Державний торговельно-економічний університет
Кафедра інженерії програмного забезпечення
та кібербезпеки

ПРОГРАМУВАННЯ ТА ЗАХИСТ ІНФОРМАЦІЇ

Збірник наукових статей студентів,
які здобувають освітній ступінь «магістр»
за спеціальностями 121 «Інженерія програмного
забезпечення» та 125 «Кібербезпека
та захист інформації»

Частина 1

Київ 2025

УДК 004.056
П 78

Програмування та захист інформації [Електронний ресурс] : зб. наук. ст. студентів / відп. ред. Т. О. Жирова. – Київ : Держ. торг.-екон. ун-т, 2025. – Ч. 1. – 240 с.

П 78

У збірнику наукових статей студентів висвітлено результати теоретичних та експериментальних досліджень у галузі інженерії програмного забезпечення й кібербезпеки та захисту інформації.

Матеріали подано в авторській редакції. Відповідальність за зміст матеріалів несуть автори.

УДК 004.056

Редакційна колегія: Т. О. Жирова (відп. ред.), канд. пед. наук, доц.; А. М. Десятко, PhD, доц.; О. А. Харченко, канд. техн. наук, доц.; О. С. Ігнатович, заступник голови РСС ФІТ з організаційних питань.

Відповідальна за випуск А. М. Десятко, PhD, доц.

Видається за рекомендацією вченої ради факультету інформаційних технологій ДТЕУ
(протокол № 24 від 13.05.2025)

© Державний торговельно-економічний університет, 2025

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

ЗМІСТ

ВСТУП..... 6

БЕНДИК І. В.
ПРОГРАМНА КОМПОНЕНТА ІНФОРМАЦІЙНОЇ СИСТЕМИ КОМПАНІЇ ПАСАЖИРСЬКИХ ПЕРЕВЕЗЕНЬ..... 7

БІЛОУС В. В.
СИСТЕМА КЛАСТЕРНОГО АНАЛІЗУ ТОВАРІВ У ЗАДАЧАХ АВТОМАТИЗАЦІЇ СКЛАДСЬКИХ ПРОЦЕСІВ..... 14

БОНДАРЕНКО М. Є.
АРХІТЕКТУРНІ ПАТЕРНИ ДЛЯ ПЕРСОНАЛІЗОВАНИХ ВЕБДОДАТКІВ..... 19

БУЛИГА Б. В.
ІНТЕГРАЦІЯ РІЗНИХ МОДУЛІВ У MVP ІНФОРМАЦІЙНОЇ СИСТЕМИ РЕСТОРАНУ: АНАЛІЗ ТЕХНІЧНИХ РІШЕНЬ ТА УПРАВЛІННЯ СУМІСНІСТЮ..... 25

БУРЛАКОВ І. А.
ОБҐРУНТУВАННЯ НЕФУНКЦІОНАЛЬНИХ ВИМОГ ТА РІШЕНЬ ПРИ РОЗРОБЦІ СЕРВІСУ МЕТЕОРОЛОГІЧНИХ СПОВІЩЕНЬ З ГНУЧКИМ НАЛАШТУВАННЯМ ЦІЛЬОВИХ УМОВ..... 30

ВОЛОШИНА А. Р.
ГНУЧКІ МЕТОДОЛОГІЇ УПРАВЛІННЯ ПРИ СТВОРЕННІ E-COMMERCE ІТ-ПРОДУКТІВ..... 36

ГАМАЛІЙ Б. С.
ОГЛЯД ТЕОРЕТИЧНИХ ПІДХОДІВ ДО РОЗРОБКИ MVP ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПІДПРИЄМСТВ ТОРГІВЛІ З ДВОСПРЯМОВАНОЮ СТРУКТУРОЮ..... 43

ГЛУЩЕНКО Д. С.
СТВОРЕННЯ ЧАТ-БОТУ НА ОСНОВІ НЕЙРОМЕРЕЖІ..... 48

ГОНЧАРЕНКО О. В.
3D КОМП'ЮТЕРНА ГРА З ВИКОРИСТАННЯМ ІГРОВОГО РУШІЯ UNITY..... 54

ГРЕЧКО А. В.
УПРАВЛІННЯ АСИНХРОННИМИ ЗАДАЧАМИ: МУЛЬТИПОТОКОВЕ ТА МУЛЬТИПРОЦЕСОРНЕ ПРОГРАМУВАННЯ У СУЧАСНИХ МОВАХ..... 59

ГРИШКО О. Ю.
ДВОФАКТОРНА СТРУКТУРИЗАЦІЯ ПРОЄКТІВ РОЗРОБКИ СЕРВІСІВ ПІДПИСКИ: ПОРІВНЯЛЬНИЙ АНАЛІЗ ІЗ ТРАДИЦІЙНИМИ МЕТОДАМИ У РИТЕЙЛІ..... 65

ГУЛЯЙ Д. О.
АДАПТАЦІЯ ДВОВЕКТОРНОЇ СТРУКТУРИЗАЦІЇ ДО ГНУЧКИХ МЕТОДОЛОГІЙ РОЗРОБКИ МОБІЛЬНОГО ДОДАТКА АДМІНІСТРУВАННЯ КАВ'ЯРНІ..... 72

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

ДЕСНОВ П. О.
ВИКОРИСТАННЯ AI-МОДЕЛЕЙ ДЛЯ АНАЛІЗУ РЕЗЮМЕ..... 80

КІКОТЬ С. В.
ІНТЕГРАЦІЯ МАШИННОГО РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ ДЛЯ ПОКРАЩЕННЯ РОБОТИ ПОШУКОВИХ АЛГОРИТМІВ..... 86

КОШОВЕЦЬ С. О.
ЕФЕКТИВНІСТЬ ПОПЕРЕДНЬО НАТРЕНОВАНИХ МОДЕЛЕЙ ГЛИБОКОГО НАВЧАННЯ ДЛЯ ЗАДАЧ РОЗПІЗНАВАННЯ ФОТО-КОНТЕНТУ..... 92

КРАВЧЕНКО А. Ю.
МІЖСЕРВІСНА КОМУНІКАЦІЯ У ВЕБДОДАТКАХ, ПОБУДОВАНИХ НА ОСНОВІ МІКРОСЕРВІСНОЇ АРХІТЕКТУРИ..... 97

МАНЧЕНКО М. І.
ПЕРСОНАЛІЗАЦІЯ НТЕРНЕТ-МАГАЗИНІВ ІЗ ВИКОРИСТАННЯМ ШТУЧНОГО ІНТЕЛЕКТУ..... 104

МАРКОВ В. Б.
ІНТЕРАКТИВНА ПЛАТФОРМА ДЛЯ АУДІО-ВІДЕОКОНФЕРЕНЦІЙ З МОЖЛИВОСТЯМИ РОЗШИРЕННЯ..... 109

МАТУХНО Т. В.
ПОРІВНЯЛЬНИЙ АНАЛІЗ ІТ-РІШЕНЬ ДЛЯ УПРАВЛІННЯ ОНЛАЙН-РЕПУТАЦІЄЮ БІЗНЕСІВ..... 114

МАТЯШ В. В.
СТВОРЕННЯ 2D-ПЛАТФОРМЕРІВ В UNITY: ОСНОВНІ ПРИНЦИПИ ТА НАЙКРАЩІ ПРАКТИКИ..... 121

НОВИЦЬКА В. В.
МЕТОДОЛОГІЇ УПРАВЛІННЯ ПРОЄКТАМИ У СТВОРЕННІ MVP..... 126

ОЛІЙНИК Д. М.
ВИКОРИСТАННЯ МАШИННОГО НАВЧАННЯ ДЛЯ ПРОГНОЗУВАННЯ ТОВАРООБІГУ..... 135

ОМЕЛЬЧЕНКО О. М.
ЕФЕКТИВНІСТЬ ВПРОВАДЖЕННЯ MVP У СТАРТАПІ БУДІВЕЛЬНОЇ ГАЛУЗІ: ЕКОНОМІЧНИЙ ТА УПРАВЛІНСЬКИЙ АНАЛІЗ..... 140

ОСТАПЕНКО В. В.
СТРАТЕГІЯ МЕНЕДЖМЕНТУ СТВОРЕННЯ MVP ВЕБОРІЄНТОВАНОГО ПРОДУКТУ..... 147

ПОТРУХ А. С.
ОПТИМІЗАЦІЯ АЛГОРИТМІВ СКАНУВАННЯ ТА ВІЗУАЛІЗАЦІЇ СТРУКТУРИ ФАЙЛОВОЇ СИСТЕМИ ДЛЯ ВЕЛИКИХ ОБСЯГІВ ДАНИХ..... 152

САВЕНКО Д. Д.
ІНТЕРАКТИВНА ВІЗУАЛІЗАЦІЯ КОСМІЧНИХ ЗАПУСКІВ..... 160

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

САЛІВОН Д. В.
АКТУАЛЬНІСТЬ ВИКОРИСТАННЯ МОВИ ПРОГРАМУВАННЯ PHP У СУЧАСНІЙ ВЕБРОЗРОБЦІ 165

СКИБУК О. І.
ПРОБЛЕМИ ЕТИЧНОСТІ ШТУЧНОГО ІНТЕЛЕКТУ ТА АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ 171

СТАРОДУБЕЦЬ В. В.
КОНВЕРСІЙНО-ОРІЄНТОВАНЕ ПРОЄКТУВАННЯ MVP: СИМБІОЗ ТЕХНІЧНИХ ТА МАРКЕТИНГОВИХ РІШЕНЬ 177

СУТЧЕНКО А. О.
ОПТИМІЗАЦІЯ БІЗНЕС-ПРОЦЕСІВ ТОРГОВЕЛЬНИХ ПІДПРИЄМСТВ ЗА ДОПОМОГОЮ КРОС-ПЛАТФОРМНИХ МОБІЛЬНИХ РІШЕНЬ 185

ТАРАСЕНКО В. О.
ДВОНАПРАВЛЕНА СТРУКТУРА УПРАВЛІННЯ ПРОЄКТАМИ РОЗРОБКИ ВЕБОРІЄНТОВАНИХ ОСВІТНІХ ПЛАТФОРМ: ПОРІВНЯЛЬНИЙ АНАЛІЗ ІЗ ТРАДИЦІЙНИМИ ПІДХОДАМИ 193

ТЕРЕН І. В.
ЦИФРОВЕ РІШЕННЯ ДЛЯ ЕФЕКТИВНОГО ПЛАНУВАННЯ ЗАХОДІВ 200

ТІМКО І. А.
АДМІНІСТРУВАННЯ САЙТУ ТА РОЗРОБКА CRM-СИСТЕМИ 204

ФЕДОРОВСЬКИЙ Р. Д.
ТЕХНОЛОГІЇ РОЗРОБКИ КРОС-ПЛАТФОРМНОГО ВЕБЗАСТОСУНКУ ДЛЯ ПРОВЕДЕННЯ ОПИТУВАНЬ 209

ФЕОКТІСТОВ С. В.
СУЧАСНІ СТРАТЕГІЇ ЗАХИСТУ КОРПОРАТИВНОЇ КОМУНІКАЦІЇ: ПРОГРАМНІ РІШЕННЯ І ПРОТОКОЛИ ШИФРУВАННЯ 214

ХОМЕНКО А. О.
МОДЕЛЬ ВЕБДОДАТКА ДЛЯ ЦИФРОВОГО УПРАВЛІННЯ ЛОГІСТИКОЮ ДОСТАВКИ 220

ШАМРИЛО Б. М.
АНАЛІЗ АРХІТЕКТУРНИХ ПАТЕРНІВ ДЛЯ МІКРОСЕРВІСНОЇ АРХІТЕКТУРИ: ВИБІР ОПТИМАЛЬНОГО ПІДХОДУ ДЛЯ РОЗРОБКИ МАСШТАБОВАНИХ СИСТЕМ 225

ШКОЛЬНИЙ М. М.
ПРОЗОРА КОМУНІКАЦІЯ ТА ГЕЙМИФІКАЦІЯ ОСВІТНЬОГО ПРОЦЕСУ 229

ШУМЕЙКО Р. Ю.
ВЕБОРІЄНТОВАНА СИСТЕМА МОНІТОРИНГУ ЯКОСТІ ОСВІТНЬОГО КОНТЕНТУ В УМОВАХ ЦИФРОВОЇ ТРАНСФОРМАЦІЇ ОСВІТИ 234

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

ВСТУП

На глобальному рівні відбуваються значні трансформації у сфері обробки та захисту інформації, зумовлені інтенсивним зростанням і впровадженням інформаційних технологій. Інформаційні технології, що ґрунтуються на комп'ютерних рішеннях, мають значний вплив на усі галузі життя та вимагають радикальних змін організаційних структур управління, його регламенту, кадрового потенціалу, системи документації, фіксування та передачі інформації.

Важливість інформаційних технологій створює нові виклики в контексті кібербезпеки та захисту інформації. Оскільки інформаційні технології стають не просто складовою частиною, а й активним каталізатором розвитку інформаційного суспільства, з'являється необхідність у забезпеченні надійності і безпеки цих технологій та відповідної інформації.

Нині одним з пріоритетних завдань є вивчення інформаційних процесів, що відбуваються в економіці, та ефективного управління ними в умовах інформаційного суспільства. При цьому неможливо обійти увагою аспекти кібербезпеки, що є необхідними відповідно до сучасного цифрового світу, де дані та системи стають вразливими перед кіберзагрозами.

На сьогодні актуальним є завдання розширення сфери інформаційної науки, зокрема зосередження на розвитку сучасних технологій програмування. Не менш вагомими є дослідження інформаційних процесів у економіці та розробка ефективних методів управління ними в умовах інформаційного суспільства. Слід зауважити, що кіберзагрози стають дедалі більш поширеним явищем. Саме тому більше уваги приділяється підготовці фахівців у галузі кібербезпеки та захисту інформації, які мають бути компетентними у вирішенні практичних завдань, пов'язаних з розробкою, забезпеченням якості впровадження та супроводження програмних засобів, а також вміти знаходити раціональні методи та засоби їх вирішення, включно зі складними ситуаціями. Крім того, вони відіграють важливу роль у підтримці сталого розвитку IT-компаній щодо якості процесів і результатів розробки програмного забезпечення.

Програма магістерської підготовки студентів спеціальностей «Інженерія програмного забезпечення», «Кібербезпека та захист інформації» орієнтовані на формування у майбутніх фахівців відповідних компетентностей для роботи в галузі наукомістких технологій, педагогічної, науково-дослідної роботи щодо вирішення актуальних прикладних, виробничих і народногосподарських завдань.

Збірник наукових статей студентів, які здобувають освітній ступінь «магістр» за спеціальностями «Інженерія програмного забезпечення», «Кібербезпека та захист інформації», містить матеріали досліджень, отриманих під час виконання випускних кваліфікаційних робіт.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

ПРОГРАМНА КОМПОНЕНТА ІНФОРМАЦІЙНОЇ СИСТЕМИ КОМПАНІЇ ПАСАЖИРСЬКИХ ПЕРЕВЕЗЕНЬ

БЕНДИК І. В., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглянуто основні принципи побудови та функціонування програмної компоненти інформаційної системи для компанії пасажирських перевезень. Досліджено ключові аспекти її розробки, включаючи архітектурні підходи, технологічні рішення та вимоги до інтеграції з іншими системами компанії. Проаналізовано переваги використання таких компонентів для оптимізації діяльності компанії, зокрема підвищення ефективності управління логістикою, спрощення доступу до послуг для пасажирів та зниження операційних витрат. Крім того, розглянуто вплив автоматизації на якість обслуговування клієнтів і конкурентоспроможність підприємства на ринку. Також представлено приклад успішної реалізації подібного рішення, який демонструє практичну цінність програмної компоненти у реальних умовах діяльності компанії пасажирських перевезень.

The article examines the fundamental principles of designing and operating a software component of an information system for a passenger transportation company. It explores key aspects of its development, including architectural approaches, technological solutions, and requirements for integration with other company systems. The advantages of using such components to optimize the company's operations are analyzed, particularly in terms of improving logistics management efficiency, simplifying passenger access to services, and reducing operational costs. Additionally, the impact of automation on customer service quality and the company's competitiveness in the market is evaluated. Furthermore, an example of a successful implementation of a similar solution is presented, illustrating the practical value of the software component in the real-world operations of a passenger transportation company.

Актуальність. Розробка програмної компоненти інформаційної системи для компанії пасажирських перевезень є важливим кроком у підвищенні ефективності транспортної галузі. Сучасні компанії, що займаються перевезеннями пасажирів (автобусними, залізничними чи авіаційними), стикаються з необхідністю швидкого оброблення даних, координації логістичних процесів та забезпечення якісного обслуговування клієнтів. Транспортна індустрія активно впроваджує цифрові технології, такі як онлайн-бронювання квитків, моніторинг руху транспорту в реальному часі та інтеграція з платіжними системами, що змінює підхід до організації перевезень.

Пасажири та компанії-перевізники мають різні потреби: від зручного доступу до розкладу та квитків до управління великими обсягами даних про маршрути, транспортні засоби та клієнтів. У традиційних системах ці процеси часто ускладнені ручною обробкою даних, що призводить до затримок і помилок. Наприклад, офлайн-облік квитків чи паперові журнали маршрутів обмежують швидкість і точність роботи. Програмна компонента інформаційної системи дозволяє автоматизувати ці процеси, надаючи компанії можливість централізовано керувати даними, а пасажирам – отримувати доступ до послуг через зручний інтерфейс, наприклад, вебпортал чи мобільний додаток.

Такий підхід не лише спрощує управління логістикою, а й розширює можливості для пасажирів: бронювання квитків онлайн, відстеження транспорту в реальному часі, інтеграція з платіжними системами тощо. Програмна компонента стає ключовим інструментом для підвищення конкурентоспроможності компанії, адже вона забезпечує швидкість, надійність і доступність послуг у цифровому форматі.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Метою статті є дослідження та розробка програмної компоненти інформаційної системи для компанії пасажирських перевезень, яка задовольняє потреби як компанії, так і її клієнтів, підвищуючи ефективність транспортних процесів.

Об'єктом дослідження є розробка програмної компоненти інформаційної системи для компанії пасажирських перевезень.

Предметом дослідження є функціонал і можливості програмної компоненти, її взаємодія з користувачами та інтеграція з іншими системами компанії.

Аналіз попередніх досліджень. К. Гупта та С. Патель вважають, що мікросервісна архітектура найкраще відповідає вимогам сучасних систем управління пасажирськими перевезеннями завдяки можливості незалежного масштабування окремих компонентів та швидкого розгортання нового функціоналу [6, с. 103–105], але вони помиляються, не приділяючи достатньої уваги складнощам міграції правових систем перевізників на нову архітектуру та забезпечення узгодженості даних у розподіленій системі. Автори запропонували схему декомпозиції бізнес-функцій транспортних компаній на окремі мікросервіси, визначивши оптимальні межі для таких компонентів, як управління розкладом, обробка платежів, система лояльності та аналітика пасажиропотоку. Їх практичне впровадження у трьох транспортних компаніях показало скорочення часу виходу на ринок нових функцій на 64% та підвищення відмовостійкості системи, але недостатньо вирішило питання синхронізації даних між компонентами та можливі проблеми з продуктивністю через надмірну розподіленість системи.

С. Делорме та Т. Фукуяма вважають, що впровадження API-орієнтованої архітектури та стандартизованих протоколів обміну даними є критичним фактором успіху при інтеграції різних компонентів інформаційної системи пасажирських перевезень та взаємодії з зовнішніми сервісами [7, с. 178–180], але помиляються в тому, що недостатньо акцентують увагу на проблемах безпеки та контролю доступу до даних в розподілених системах з численними точками інтеграції. Автори розробили референсну модель API для транспортних компаній, яка включає специфікації для бронювання квитків, управління розкладом, інформування пасажирів та інтеграції з платіжними системами. Їхні результати показують, що стандартизований підхід до API зменшує вартість інтеграції нових партнерів на 67% та прискорює вихід на нові ринки, проте не розглядає детально питання версіонування API та можливі проблеми з'єднуваності при взаємодії з системами, побудованими на застарілих технологіях.

Виклад основного матеріалу. У сучасному світі інформаційні системи стали невід'ємною частиною транспортної індустрії. Програмна компонента інформаційної системи компанії пасажирських перевезень виконує роль ядра, яке об'єднує різні процеси: від продажу квитків до управління рухом транспорту. Такі компоненти зазвичай включають модулі для роботи з базою даних (зберігання інформації про пасажирів, розклади, тарифи), серверну логіку для обробки запитів та інтерфейс користувача – як для клієнтів, так і для працівників компанії.

Прикладом успішної реалізації може слугувати система онлайн-бронювання квитків, інтегрована з GPS-трекінгом, яка дозволяє пасажирам бачити актуальний статус рейсу, а компанії – оптимізувати маршрути в реальному часі.

Переваги такої компоненти очевидні: автоматизація зменшує людський фактор у помилках, прискорює обслуговування та забезпечує доступ до даних із будь-якої точки світу. Це дозволяє компанії не лише економити ресурси, а й адаптуватися до зростаючих потреб ринку, пропонуючи нові послуги, наприклад, інтеграцію з туристичними платформами чи мобільними додатками для пасажирів.

Програмна компонента інформаційної системи компанії пасажирських перевезень вирізняється модульною структурою, яка дозволяє розділити систему на незалежні компоненти. Ці компоненти можна розробляти, тестувати та підтримувати окремо, що значно полегшує обслуговування системи та сприяє її масштабуванню в процесі розвитку. Завдяки

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

принципу інкапсуляції внутрішні деталі кожного модуля приховані, а доступними залишаються лише необхідні інтерфейси, що знижує загальну складність системи та робить її більш керованою.

Однією з ключових особливостей програмного забезпечення є можливість моніторингу в реальному часі. Ця функція дозволяє менеджерам транспортної компанії постійно відстежувати продуктивність автопарку, стан транспортних засобів і дії водіїв. Такі дані дають змогу оперативно реагувати на поломки чи відхилення від маршруту, що підвищує ефективність роботи та мінімізує перебої в обслуговуванні. Інтеграція прогнозної аналітики в управління автопарком додатково сприяє своєчасному технічному обслуговуванню, зменшуючи ймовірність збоїв під час рейсів. Наступна таблиця представляє ключові функції програмного забезпечення для планування транспортного графіка, яке є невід'ємною частиною програмної компоненти інформаційної системи компанії пасажирських перевезень (табл. 1).

Таблиця 1

Ключові функції програмного забезпечення для планування транспортного графіка

Функція	Опис
Планування маршрутів	Оптимізація маршрутів на основі трафіку, попиту пасажирів і розкладу перевезень.
Розподіл водіїв	Призначення водіїв на зміни та маршрути з урахуванням ефективності, забезпечення належного покриття та дотримання норм
Оновлення в реальному часі	Надає інформацію в реальному часі про розташування транспортних засобів, затримки та зміни в розкладі
Управління автопарком	Здійснює контроль за технічним обслуговуванням транспортних засобів, витратами палива та відстеженням продуктивності.
Система інформації для пасажирів	Відображає оновлення в реальному часі для пасажирів на платформах, у додатках і на вебсайтах, включаючи час прибуття
Автоматизоване планування	Автоматизує створення розкладів на основі історичних даних, попиту та доступності ресурсів
Інтеграція квитків та оплати	Інтегрується з цифровими платіжними системами, забезпечуючи зручні транзакції для пасажирів
Аналітика та звітність	Надає інструменти для аналізу даних і звітності щодо продуктивності системи, кількості пасажирів та інших показників
GPS-відстеження	Уможливлює відстеження транспортних засобів у реальному часі для ефективного управління автопарком і маршрутами
Моніторинг відповідності	Гарантує, що розклади та операції відповідають урядовим нормам і стандартам безпеки

Джерело: [1]

Кожна функція спрямована на підвищення ефективності роботи транспортної компанії, оптимізацію ресурсів та покращення досвіду пасажирів. Наприклад, планування маршрутів та GPS-відстеження дозволяють адаптуватися до змін у трафіку та попиті, тоді як автоматизоване планування та інтеграція з платіжними системами спрощують адміністративні процеси. Система інформації для пасажирів і оновлення в реальному часі забезпечують прозорість і зручність, а моніторинг відповідності гарантує дотримання нормативних вимог. Такий підхід дозволяє інтегрувати технології для створення комплексного рішення, яке підтримує як операційні, так і стратегічні цілі компанії.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Важливу роль відіграють також передові алгоритми штучного інтелекту, які оптимізують планування маршрутів. Аналізуючи різноманітні дані – попит пасажирів, умови руху, історичні тенденції – ці алгоритми динамічно адаптують маршрути до поточних умов. Система здатна в реальному часі коригувати шлях у відповідь на непередбачені ситуації, як затори чи сплески пасажиропотоку в години пік. Така гнучкість не лише покращує якість послуг, а й оптимізує використання ресурсів компанії (табл. 2).

Таблиця 2

Таблиця переваг оптимізації використовуючи ШІ

Критерій	Традиційне планування	Оптимізоване ШІ
Гнучкість маршрутів	Фіксовані маршрути, змінюються рідко	Динамічна адаптація в реальному часі
Час у дорозі	Не враховує поточні затори	Мінімізує затримки через аналіз трафіку
Витрати на паливо	Вищі через неефективні маршрути	Зменшені за рахунок оптимізації руху
Комфорт пасажирів	Можливі перевантаження транспорту	Балансування навантаження в години пік
Реакція на форс-мажори	Повільне реагування	Швидке коригування в режимі реального часу
Використання ресурсів	Нестача або надлишок транспорту	Оптимальне розподілення транспорту за попитом

Джерело: розроблено автором

Представлена таблиця дозволяє порівнити традиційний підхід до планування маршрутів із сучасними методами, що базуються на алгоритмах штучного інтелекту. Основна відмінність між цими підходами полягає в рівні гнучкості та здатності адаптуватися до змінних умов. Традиційні маршрути, як правило, є статичними та не враховують у реальному часі такі фактори, як дорожній трафік або попит пасажирів. Натомість штучний інтелект забезпечує динамічну оптимізацію, що дозволяє мінімізувати затримки, скоротити витрати на паливо та рівномірно розподіляти транспортні потоки [2].

Загальні переваги використання ШІ в управлінні маршрутами включають:

- *Зменшення часу в дорозі* завдяки аналізу актуальних дорожніх умов.
- *Раціональне використання ресурсів*, що дозволяє оптимально розподіляти транспортні засоби відповідно до попиту.
- *Підвищення комфорту пасажирів* через балансування навантаження у години пік.
- *Оперативне реагування на непередбачені ситуації*, зокрема затори або зміни в пасажиропотоці.
- *Зниження витрат* на паливо та експлуатацію транспорту завдяки ефективному плануванню маршрутів.

Впровадження штучного інтелекту в транспортні системи значно підвищує ефективність та забезпечує кращу якість обслуговування пасажирів.

Розробка основних функцій програмного забезпечення базується на користувацько-орієнтованому підході. Такі можливості, як планування маршрутів, розподіл водіїв і відстеження в реальному часі, створюються з урахуванням потреб транспортних компаній. Інтуїтивний, швидкий і надійний інтерфейс забезпечує зручність використання як для операторів, так і для пасажирів, надаючи чітку та зрозумілу інформацію. Водночас програмне забезпечення ефективно інтегрується з існуючими системами, автоматизуючи та вдосконалюючи процеси планування маршрутів і управління автопарком. Це дозволяє динамічно

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

перенаправляти транспорт на основі актуальних даних про рух і підвищувати загальну продуктивність транспортної мережі. Система також гнучко адаптується до різних операційних умов, що робить її універсальною для різних транспортних середовищ [3].

Не менш важливими є детальна документація та заходи безпеки. Документація сприяє кращому порозумінню між учасниками проєкту, тоді як вбудовані механізми захисту оберігають систему від несанкціонованого доступу та забезпечують безпеку конфіденційних даних. Програмна компонента також відповідає визначеним метрикам продуктивності, таким як ефективне використання ресурсів і пропускна здатність, що є критично важливими для транспортних операцій. Завдяки ретельно підібраним архітектурним шаблонам досягаються низька затримка та висока пропускна здатність, особливо в додатках реального часу. Усе це разом гарантує, що програмне забезпечення здатне відповідати вимогам сучасних транспортних систем, забезпечуючи їхню стабільність і ефективність.

Технічна архітектура інформаційної системи компанії пасажирських перевезень відіграє ключову роль у забезпеченні масштабованості, продуктивності та підтримки системи. Вона базується на різних архітектурних шаблонах, які дозволяють вирішувати специфічні виклики, пов’язані з великим трафіком і обробкою даних. Одним із таких підходів є архітектура на основі простору, також відома як хмарна або tuple-space архітектура. Цей підхід ефективно розподіляє навантаження програми між кількома екземплярами, використовуючи віртуалізовані або in-memory data grids (IMDG) як спільні пам’ятні простори для компонентів. Таке рішення зменшує залежність від традиційних баз даних, знижуючи затримки та уникаючи заторів, характерних для застарілих систем. Цей шаблон ідеально підходить для додатків, які потребують високої масштабованості, стійкості до збоїв і швидкої обробки даних у реальному часі [3].

Іншим важливим підходом є архітектура мікросервісів, яка розбиває додаток на невеликі, незалежно розгортаємі сервіси, кожен із яких відповідає за конкретну бізнес-функцію, наприклад, управління користувачами чи обробку платежів (рис. 1).

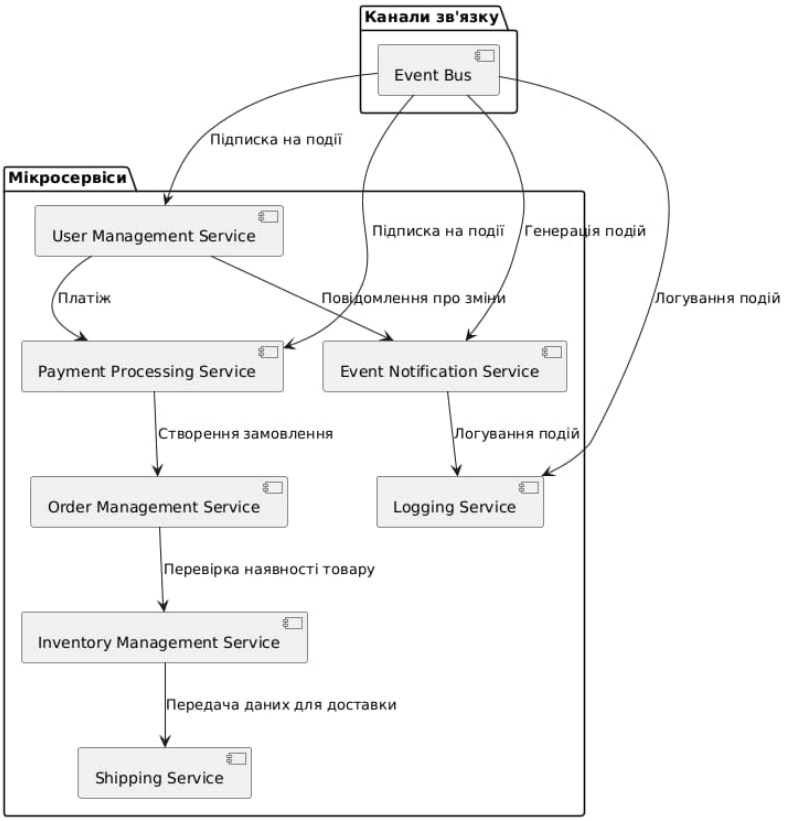


Рис. 1. UML діаграма компонентів архітектури мікросервісів

Джерело: розроблено автором

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Діаграма архітектури мікросервісів зображує систему, що побудована за принципом розподілених сервісів, кожен з яких виконує окрему бізнес-функцію. Така архітектура дає змогу забезпечити високу гнучкість та ефективне масштабування, адже кожен мікросервіс є незалежним компонентом, який може масштабуватися та оновлюватися без впливу на інші частини системи.

Мікросервіси на діаграмі включають такі компоненти:

– *Сервіс управління користувачами (User Management Service)* – відповідає за реєстрацію, аутентифікацію та авторизацію користувачів системи.

– *Сервіс обробки платежів (Payment Processing Service)* – забезпечує виконання платіжних транзакцій, інтеграцію з платіжними шлюзами та обробку фінансових даних.

– *Сервіс повідомлень про події (Event Notification Service)* – генерує та надсилає повідомлення про події, що відбуваються в системі, наприклад, зміни статусу замовлення або обробку платіжних транзакцій.

– *Сервіс управління замовленнями (Order Management Service*) – відповідає за обробку замовлень, їх збереження та статуси в системі.

– *Сервіс управління запасами (Inventory Management Service)* – здійснює моніторинг та управління наявністю товарів, контролює рівень запасів.

– *Сервіс доставки (Shipping Service)* – відповідає за організацію та відстеження процесу доставки товарів.

– *Event Bus* – є ключовим компонентом архітектури, який забезпечує асинхронну взаємодію між мікросервісами через події. Всі сервіси підключаються до Event Bus, що дозволяє знижувати рівень зв'язку між компонентами та забезпечує гнучкість системи.

Взаємодія між сервісами здійснюється через події, що гарантує високу ступінь ізоляції компонентів, дозволяючи їм працювати автономно. Наприклад, при створенні нового замовлення сервіс управління замовленнями ініціює подію, що передається через Event Bus, яка сповіщає про необхідність перевірки наявності товару в інвентарі. У разі успішної перевірки дані передаються до сервісу доставки для організації відправлення товару.

Такий підхід дозволяє гнучким командам ефективно керувати складними системами, забезпечуючи незалежне масштабування та розгортання сервісів через легковагові протоколи зв'язку. Він особливо корисний для додатків, які занадто складні для монолітної архітектури, і потребують частих оновлень, швидкого випуску нових функцій та адаптивного масштабування. Водночас архітектура, орієнтована на події, зосереджується на генерації, виявленні та реакції на події, де компоненти спілкуються через асинхронні події, уникаючи прямих залежностей. Це забезпечує гнучкість і масштабування, а ключовими елементами є виробники подій, споживачі та канали зв'язку. Такий підхід рекомендують для систем, які обробляють численні незалежні завдання або безперервні потоки даних із різних пристроїв, а також для випадків, коли збої одного компонента не повинні зупиняти всю систему.

Ще одним класичним рішенням є клієнт-серверна архітектура, яка чітко розподіляє ролі між клієнтом, що управляє інтерфейсом користувача, та сервером, який відповідає за обробку даних. Цей підхід покращує управління ресурсами завдяки розділенню обов'язків і широко застосовується в середовищах, де потрібен обмін даними між географічно віддаленими локаціями. Вибір архітектурного шаблону залежить від аналізу складності системи, необхідного рівня масштабованості та потреб у обробці даних у реальному часі. Перехід між шаблонами під час розробки можливий, але може призвести до ускладнень, таких як підтримка узгодженості даних, підвищення витрат на розробку чи дублювання коду. Тому обрана архітектура має відповідати операційним потребам компанії, забезпечуючи баланс між продуктивністю та підтримкою [4].

Важливим аспектом є інтеграція інформаційної системи компанії пасажирських перевезень (PTCIS) з іншими системами, такими як транспортні (TMS), системи планування ресурсів підприємства (ERP) та системи управління складом (WMS), що сприяє підвищенню

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

операційної ефективності та покращенню досвіду користувачів. Ефективна інтеграція дозволяє обмінюватися критичними даними про замовлення, запаси та інформацію про клієнтів, узгоджуючи транспортні процеси з ширшими бізнес-операціями. Це створює єдину платформу, де управління транспортом координується з роботою складів і виконанням замовлень, забезпечуючи своєчасне переміщення товарів і послуг. Одним із важливих напрямів інтеграції є об'єднання тарифів і розкладів, що дозволяє використовувати єдиний квиток для кількох видів транспорту, спрощуючи доступ і знижуючи витрати для пасажирів, а також узгодження розкладів для мінімального часу очікування при переході між видами транспорту [5].

Інформаційна інтеграція забезпечує користувачам доступ до всебічних даних про транспорт через мобільні додатки та інформаційні центри, тоді як мережева інтеграція з'єднує різні види транспорту в єдину мережу, підвищуючи загальну ефективність і зручність. Проте інтеграція стикається з викликами, такими як сумісність апаратного та програмного забезпечення, складність систем та потреба в стандартизованих протоколах для обміну даними між підсистемами, наприклад, системами контролю доступу чи RFID-тикету. Щоб подолати ці перешкоди, компанії повинні залучати IT-команди та постачальників послуг на етапі планування, чітко визначаючи вимоги до інтеграції та відстежуючи потоки даних. Використання стандартизованих API полегшує взаємодію між системами, регулярне тестування гарантує їхню гармонійну роботу, а співпраця між зацікавленими сторонами сприяє вирішенню проблем і підвищенню ефективності. Завдяки таким стратегіям інтеграція PTCIS із зовнішніми системами може значно покращити транспортну ефективність, безпеку та загальний досвід пасажирів [5].

Висновки. Розробка та впровадження програмної компоненти інформаційної системи для компанії пасажирських перевезень є важливим кроком у підвищенні ефективності транспортної галузі. Завдяки інтеграції сучасних технологій, таких як моніторинг у реальному часі, AI-оптимізація маршрутів та модульна архітектура, ці системи забезпечують автоматизацію ключових процесів, покращуючи як операційну діяльність компанії, так і досвід пасажирів. Якщо традиційні методи управління транспортом часто стикаються з обмеженнями у швидкості обробки даних і гнучкості, то програмні компоненти пропонують масштабовані рішення, які адаптуються до зростаючих потреб ринку. У підсумку, впровадження таких систем не лише оптимізує логістику та знижує витрати, а й відкриває нові можливості для розвитку транспортних послуг, демонструючи, як цифрові інновації можуть трансформувати галузь пасажирських перевезень у відповідь на виклики сучасності.

Список використаних джерел

1. Transit Scheduling Software Development. Matellio. URL: https://www.matellio. com/blog/transit-scheduling-software-development/
2. Types of Software Architecture Patterns. GeeksforGeeks. URL: https://www.geeksforgeeks.org/types-of-software-architecture-patterns/
3. Microservices Fueling Digital Transformation in Transportation. Moldstud. URL: https://moldstud.com/articles/p-microservices-fueling-digital-transformation-in-transportation
4. Transportation Management System. Trinetix. URL: https://www.trinetix.com/insights/transportation-management-system
5. 5 Most Common Challenges in TMS Implementation and How to Overcome Them. UniqTMS. URL: https://uniqtms.com/5-most-common-challenges-in-tms-implementation-and-how-to-overcome-them/
6. Gupta, K., Patel, S. (2023). Microservice-Based Architecture for Modern Public Transportation Information Systems. International Journal of Software Engineering & Applications, 14(3), 97–112.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

7. Delorme, S., Fukuyama, T. (2023). API-First Approach for Transit Information Systems: A Case Study of Multi-Modal Transportation Data Integration. IEEE Transactions on Intelligent Transportation Systems, 24(4), 173–185.

Роботу виконано під науковим керівництвом канд. екон. наук, доцента
ПАЛАГУТИ К. О.

СИСТЕМА КЛАСТЕРНОГО АНАЛІЗУ ТОВАРІВ У ЗАДАЧАХ АВТОМАТИЗАЦІЇ СКЛАДСЬКИХ ПРОЦЕСІВ

БІЛОУС В. В., 2 курс 2м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглядаються можливості та переваги автоматизованих складських систем, що використовують алгоритми аналізу та кластеризації товарів. Описано основні компоненти програмного забезпечення для управління складом, алгоритми машинного навчання для аналізу запасів та методи кластеризації для оптимізації розподілу товарів. Наведено приклади застосування автоматизованих систем у сучасній логістиці.

The article discusses the capabilities and advantages of automated warehouse systems that use algorithms for analyzing and clustering goods. The main components of warehouse management software, machine learning algorithms for analyzing inventories, and clustering methods for optimizing the distribution of goods are described. Examples of the use of automated systems in modern logistics are given.

Актуальність. Оптимізація складських процесів є ключовим завданням для компаній, що працюють у сфері логістики, ритейлу та виробництва. Використання алгоритмів аналізу та кластеризації дозволяє підвищити ефективність роботи складу, зменшити витрати на зберігання та транспортування товарів, а також покращити швидкість обробки замовлень.
Важливість автоматизованих складських систем для бізнесу:

– Зменшення витрат. Автоматизація знижує потребу в ручному управлінні запасами, мінімізує помилки під час інвентаризації та покращує прогнозування попиту.

– Оптимізація простору складу. Використання кластеризації дозволяє ефективно розподіляти товари, скорочуючи час на їхнє розміщення та пошук.

– Підвищення швидкості виконання замовлень. Інтелектуальні алгоритми допомагають визначити найшвидші маршрути комплектування замовлень.

– Аналіз тенденцій. Алгоритми машинного навчання аналізують історичні дані продажів і запасів для прогнозування майбутніх потреб.

Метою статті є аналіз особливостей використання автоматизованих складських систем із застосуванням методів аналізу та кластеризації.

Об'єктом дослідження є процес управління складськими запасами.

Предмет дослідження – алгоритми аналізу та кластеризації товарів у автоматизованих складських системах.

Аналіз попередніх досліджень. Дослідження автоматизації складської логістики.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Виклад основного матеріалу. Manhattan Associates WMS (Warehouse Management System) є однією з провідних систем для управління складами, яка широко використовується у багатьох галузях для оптимізації та автоматизації процесів на складах. Це програмне рішення надає підприємствам можливість ефективно керувати рухом товарів, запасами та ресурсами в реальному часі. Встановлення Manhattan WMS дозволяє компаніям зменшити витрати, покращити ефективність та знизити ризик помилок, що є важливим аспектом у сучасному бізнес-середовищі, де швидкість і точність є критичними факторами для успіху.

Однією з основних переваг Manhattan WMS є його масштабованість і гнучкість. Система розроблена для підтримки різних типів складів та специфічних вимог різних галузей, таких як ритейл, дистрибуція, виробництво та інші. Вона дозволяє підприємствам налаштовувати систему під свої потреби, враховуючи особливості логістики, процесів обробки замовлень, зберігання товарів, а також інтеграцію з іншими корпоративними системами (наприклад, ERP). Однак, хоча система і забезпечує високий рівень автоматизації, її впровадження може бути складним і потребує значних затрат часу та ресурсів.

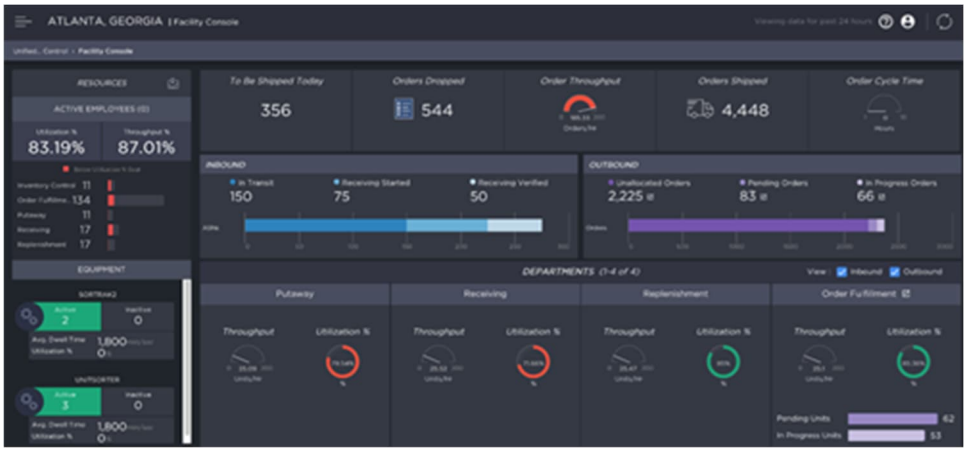


Рис. 1. Інтерфейс Manhattan Associates WMS

Джерело: розроблено автором (знімок з єкрану)

Переваги:

1. Висока продуктивність і масштабованість
   1.1. Підходить для великих компаній із високим навантаженням на склад.
   1.2. Легко адаптується під зміну обсягів операцій.
2. Широкий функціонал
   2.1. Підтримка управління запасами, обліку, маршрутизації та оптимізації складських процесів.
   2.2. Інтеграція з технологіями штучного інтелекту та машинного навчання для прогнозування попиту.
3. Гнучкість та налаштування
   3.1. Дозволяє адаптуватися під бізнес-процеси різних компаній.
   3.2. Підтримує конфігурацію користувацьких сценаріїв.
4. Інтеграція з іншими системами
   4.1. Можливість підключення до ERP-систем (SAP, Oracle, Microsoft Dynamics).
   4.2. Підтримка інтеграції з TMS (Transportation Management System) та OMS (Order Management System).
5. Автоматизація та покращена аналітика
   5.1. Зменшує кількість ручної роботи, оптимізує роботу складу.
   5.2. Вбудовані аналітичні інструменти для моніторингу продуктивності.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Недоліки:

1. Висока вартість
   1.1. Дороге впровадження та підтримка.
   1.2. Вимагає значних інвестицій у навчання персоналу.
2. Складність впровадження
   2.1. Тривалий процес налаштування та адаптації під специфіку бізнесу.
   2.2. Потребує досвідчених IT-фахівців для підтримки.
3. Вимоги до інфраструктури
   3.1. Необхідність потужного серверного обладнання або хмарних ресурсів.
   3.2. Може потребувати окремого серверного середовища для стабільної роботи.
4. Високі вимоги до персоналу
   4.1. Користувачі повинні пройти навчання перед використанням системи.
   4.2. Без належного навчання можлива низька ефективність використання.
5. Обмежена гнучкість для малого бізнесу
   5.1. Більше орієнтована на великі логістичні компанії та ритейлерів.
   5.2. Може бути надто складною та дорогою для невеликих складів.

Щодо недоліків, можна відзначити, що впровадження Manhattan WMS може бути дорогим і вимагати значних інвестицій, що робить його менш доступним для малих і середніх підприємств. Крім того, для успішного впровадження системи необхідні кваліфіковані спеціалісти, що можуть забезпечити налаштування та оптимізацію роботи програми, оскільки система має велику кількість налаштувань, що можуть бути складними для нових користувачів.

Manhattan WMS використовує такі технології, як Java, а також пропонує можливості для інтеграції з іншими мовами програмування, залежно від потреб компанії. Однак зміни та модифікації системи на власний розсуд можуть бути обмежені, адже багато аспектів програми знаходяться під контролем постачальника програмного забезпечення, що може вимагати погодження змін через ліцензійні угоди та правила підтримки.

Важливими аспектами застосування Manhattan WMS є автоматизація збору даних, моніторинг у реальному часі, а також вдосконалене управління запасами. Це дозволяє зменшити людську помилку, покращити точність і швидкість виконання завдань, що критично важливо для великих складів, де кожна хвилина на рахунку. Окремої уваги заслуговує система моніторингу продуктивності, яка дозволяє відслідковувати роботу складу і оперативно виявляти місця, де є можливості для поліпшення.

Manhattan Associates WMS використовує різні алгоритми аналізу для оптимізації процесів на складі, покращення управління запасами та підвищення ефективності операцій. Деякі з основних алгоритмів аналізу, які застосовуються в цій системі, включають:

– Алгоритм оптимізації розміщення товарів (Slotting Optimization): Цей алгоритм визначає найбільш ефективне місце для зберігання товарів на складі. Він враховує такі фактори, як обсяг попиту, розміри одиниць товару, частота обробки та інші характеристики товару для визначення оптимальних локацій. Мета – мінімізувати час на пошук товару і зменшити відстань для переміщення товару по складу.

– Алгоритм розподілу замовлень (Order Fulfillment Optimization): Алгоритм відповідає за оптимальний розподіл замовлень серед доступних ресурсів (працівників, транспорту, складів тощо). Він враховує фактори, такі як пріоритет замовлень, доступність товару та час виконання замовлення.

– Алгоритм планування ресурсів (Resource Planning Optimization): Цей алгоритм оптимізує використання ресурсів складу, включаючи робочу силу та техніку. Він може автоматично розподіляти завдання серед працівників, щоб забезпечити максимальну продуктивність і зменшити час очікування.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

– Алгоритм обробки повернень (Return Management Optimization): Для оптимізації процесу повернення товарів цей алгоритм визначає найкращі методи обробки повернень і повернення товарів на склад з урахуванням їх стану та вартості.

– Алгоритм прогнозування попиту (Demand Forecasting): Система використовує алгоритми прогнозування для аналізу історичних даних про попит на товари, сезонні коливання та тренди. Це дозволяє створювати точніші прогнози щодо потреб складу і планувати поповнення запасів з урахуванням очікуваних коливань попиту.

– Алгоритм управління продуктивністю (Performance Management): Цей алгоритм допомагає відслідковувати ефективність роботи складу, включаючи час виконання завдань, продуктивність працівників та інші ключові показники. Він дозволяє виявляти проблеми та можливості для вдосконалення процесів.

В порівнянні з Oracle Warehouse Management Cloud (WMS), Manhattan WMS має деякі відмінності. Oracle WMS – це хмарне рішення, яке дозволяє підприємствам користуватися його можливостями без необхідності установки і підтримки складної інфраструктури. Це рішення більше орієнтоване на глобальні компанії, які хочуть мати доступ до своїх складських даних з будь-якої точки світу. Хоча обидві системи мають схожі функціональні можливості, Oracle WMS може бути більш гнучким у питанні масштабованості та інтеграції з іншими хмарними продуктами, особливо з ERP-системами Oracle.

Переваги:

1. Хмарне рішення
   1.1. Не потребує дорогого серверного обладнання та технічного обслуговування.
   1.2. Легкий доступ із будь-якого місця.
2. Гнучкість і масштабованість
   2.1. Підходить як для малого, так і для великого бізнесу.
   2.2. Дозволяє швидко адаптуватися до зростання обсягів операцій.
3. Швидке впровадження
   3.1. На відміну від традиційних WMS, процес налаштування та запуску займає менше часу.
   3.2. Менше потребує складних IT-налаштувань.
4. Інтеграція з іншими рішеннями Oracle
   4.1. Безшовно працює з Oracle ERP, SCM та іншими корпоративними системами.
   4.2. Підтримує API для підключення сторонніх рішень.
5. Автоматизація та аналітика
   5.1. Вбудовані алгоритми оптимізації складських процесів.
   5.2. Використання штучного інтелекту для прогнозування попиту та оптимізації запасів.
6. Мобільний доступ
   6.1. Підтримка роботи на мобільних пристроях без необхідності складного програмного забезпечення.
   6.2. Підтримка голосового введення та RFID-технологій.
7. Регулярні оновлення
   7.1. Автоматичні оновлення без зупинки роботи.
   7.2. Доступ до нових функцій без додаткових витрат.

Недоліки:

1. Висока вартість підписки
   1.1. Хоча немає витрат на серверне обладнання, щомісячна плата може бути дорогою для малого бізнесу.
   1.2. Додаткові витрати на інтеграцію та навчання персоналу.
2. Залежність від інтернет-з’єднання
   2.1. Повністю залежить від стабільності мережі.
   2.2. Можливі проблеми при роботі у віддалених регіонах із поганим інтернетом.
3. Обмежена кастомізація

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

3.1. Менше можливостей для глибокої адаптації під специфічні бізнес-процеси порівняно з локальними WMS.
3.2. Налаштування можуть бути складнішими через обмеження хмарного середовища.

4. Потреба у навчанні персоналу
   4.1. Інтерфейс та функціонал можуть бути складними для нових користувачів.
   4.2. Потребує часу на адаптацію співробітників до нової системи.
5. Обмежена підтримка сторонніх рішень
   5.1. Хоча є API, інтеграція з деякими нестандартними ERP або локальними рішеннями може бути складною.
   5.2. Може вимагати додаткових адаптаційних модулів
6. Manhattan WMS, у свою чергу, виграє в плані специфічності налаштувань і більш детального контролю над процесами управління складом. Якщо для компанії важлива велика гнучкість і контроль на рівні інфраструктури, Manhattan може бути більш підходящим вибором. Проте Oracle WMS буде кращим варіантом для компаній, які вже працюють в екосистемі Oracle і шукають інтеграцію з іншими їхніми продуктами, зокрема у хмарі.

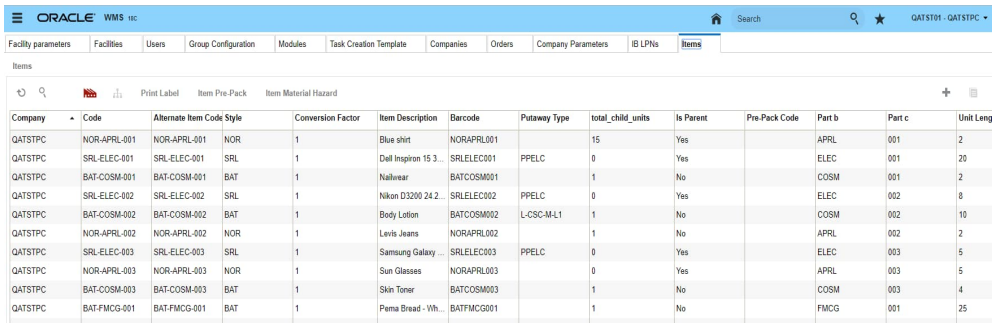


Рис. 2. Інтерфейс Oracle Warehouse Management Cloud

Джерело: Розроблено автором (знімок з екрану)

Таблиця 1.

Функціональні та технічні відмінності між Manhattan WMS та Oracle WMS Cloud

Параметр	Manhattan Associates WMS	Oracle Warehouse Management Cloud
Тип	Локальне та гібридне рішення	Хмарна платформа (SaaS)
Гнучкість	Більше можливостей кастомізації	Обмежена гнучкість у зміні логіки
Впровадження	Довше та дорожче	Швидше завдяки SaaS-моделі
Функціонал AI	Покращені алгоритми кластеризації та прогнозування	Базовий AI для управління запасами
Масштабованість	Висока для підприємств	Гнучка для середнього та великого бізнесу
Сумісність	ERP-системи, API-інтеграція	Глибока інтеграція з Oracle ERP
Підтримка мобільних пристроїв	Є, але обмежена	Повноцінна підтримка мобільного доступу
Ціна	Висока вартість впровадження та підтримки	Гнучка модель підписки

Джерело: розроблено автором

Висновки. В загальному, обидва рішення мають свої переваги і можуть задовольнити потреби компаній залежно від специфіки їхнього бізнесу, масштабу і вимог до інтеграції з іншими системами.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Список використаних джерел

1. Комплексний алгоритм кластеризації для стратегічного аналізу ланцюгів постачання. sciencedirect. URL: https:// www.sciencedirect.com/science/article/pii/S0360835299001552.
2. Overview of Entity Framework Core - EF Core. *Microsoft Learn: Build skills that open doors in your career*. URL: https://learn.microsoft.com/en-us/ef/core/ (date of access: 26.03.2025).
3. C# docs - get started, tutorials, reference. *Microsoft Learn: Build skills that open doors in your career*. URL: https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/ (date of access: 26.03.2025).
4. .NET documentation. *Microsoft Learn: Build skills that open doors in your career*. URL: https://learn.microsoft.com/en-us/dotnet/ (date of access: 26.03.2025).
5. Warehouse Management | Oracle. *fw_error_www*. URL: https://www.oracle.com/scm/logistics/warehouse-management/ (date of access: 26.03.2025).
6. Warehouse Management System (WMS) | Manhattan. *Retail and Supply Chain Software Solutions | Manhattan*. URL: https://www.manh.com/solutions/supply-chain-management-software/warehouse-management (date of access: 26.03.2025).

Роботу виконано під науковим керівництвом канд. техн. наук, доцента
ХАРЧЕНКА О. А.

АРХІТЕКТУРНІ ПАТЕРНИ ДЛЯ ПЕРСОНАЛІЗОВАНИХ ВЕБДОДАТКІВ

БОНДАРЕНКО М. Є., 2 курс 4м група ФІТ ДТЕУ,
освітня програма «Управління проєктами програмних продуктів»

У статті розглянуто основні підходи та архітектурні патерни, які використовуються при розробці та розвитку персоналізованих веб-додатків. Визначена проблематика персоналізації під час розробки. Виконано порівняльний аналіз основних патернів, визначення основних переваг та недоліків. Розглянуто приклади успішних реалізацій

The article examines the main approaches and architectural patterns used in the design and development of personalized web applications. The issues of personalization during development are identified. A comparative analysis of the main patterns is performed, the main advantages and disadvantages are identified. Examples of successful implementations are considered.

Актуальність. Персоналізація B2B веб-додатків є однією з ключових тенденцій у розвитку сучасних програмних продуктів, що дозволяє компаніям ефективніше задовольняти потреби різних клієнтів. Високий рівень кастомізації стає критично важливим для забезпечення конкурентоспроможності, адже дозволяє адаптувати функціональність, інтерфейс та можливості продукту під унікальні бізнес-вимоги кожного клієнта. Однак персоналізація вимагає складних архітектурних рішень, що забезпечують не лише гнучкість та масштабованість, але й високу продуктивність, безпеку та можливість подальшого розвитку.

У контексті B2B веб-додатків, персоналізація передбачає динамічну конфігурацію інтерфейсу, адаптацію функціональних можливостей та налаштування користувацького

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

досвіду відповідно до специфічних вимог клієнтів. Це може охоплювати налаштування звітів, аналітики, управління користувачами та правами доступу, а також інтеграцію з корпоративними системами. Також все частіше це стосується дизайну та контенту сторінок (віджети, пункти меню тощо). Це вимагає також відповідних підходів до розробки програмного продукту. Проте, зростання кількості клієнтів та обсягу функціональності ускладнює управління додатком, підвищуючи потребу в надійних архітектурних рішеннях. Особливо якщо кількість конфігурацій стрімко зростає, а самі конфігурації дозволяють досить гнучко підлаштувати додаток під потреби кожного клієнта.

Застосування архітектурних патернів у персоналізованих B2B веб-додатках дозволяє ефективно вирішувати проблеми масштабування, продуктивності та підтримки різноманітних конфігурацій. До ключових патернів належать Multi-Tenancy, Feature Toggles, Event-Driven Architecture та Microservices. Вибір конкретного патерну залежить від складності проєкту, вимог до ізоляції даних, продуктивності та можливості швидко адаптуватися до змін ринку.

Метою статті є дослідження та аналіз архітектурних патернів, що забезпечують ефективну персоналізацію в B2B продуктах, а також визначення їх переваг та недоліків у різних сценаріях використання. Дослідження зосереджено на визначенні оптимальних рішень для створення персоналізованих рішень, які відповідають сучасним викликам бізнесу та IT-інфраструктури.

Об'єктом дослідження є розробка персоналізованого веб-застосунку, орієнтованого на використання клієнтами з різними технічними вимогами.

Предмет дослідження – архітектурні паттерни для персоналізації веб-додатків з моделлю B2B.

Аналіз попередніх досліджень. Оскільки тенденція стосовно персоналізації при розробці веб-додатків стає дедалі зростаючою, все більше матеріалів та досліджень з'являється по цій темі. Суттєва більшість з них описані або в різних виданнях мережі інтернет, або в працях таких авторів як Андрій Пригода, Sanjukta Pal, Peter Garraghan, Paul Townend, а також в документаціях відповідних інструментів, які допомагають реалізовувати персоналізацію додатків .

Виклад основного матеріалу. Варто почати з визначення проблематики персоналізації веб-застосунків, викликів, які стоять перед командою при розробці та подальшій підтримці продукту, а також сучасних тенденцій в персоналізації додатків. Також необхідно звернути увагу на наслідки, які потенційно може нести неправильний підхід до персоналізації або хибне визначення потреб конкретного програмного продукту. Персоналізація програмних продуктів у B2B-секторі є складним процесом, який передбачає глибоку адаптацію функціональності, інтерфейсу та користувацького досвіду до потреб конкретного клієнта. Зокрема, це охоплює індивідуальні налаштування панелей керування, вибір функціональних модулів, створення кастомних звітів та інтеграцію з корпоративними інформаційними системами. Основна складність полягає в тому, що для забезпечення персоналізації потрібно одночасно підтримувати велику кількість різних конфігурацій, що може негативно впливати на продуктивність та стабільність системи. Необхідність ефективного управління та контролю великої кількості різних варіантів функціональності вимагає ретельного планування, побудови складних архітектурних рішень та застосування спеціалізованих інструментів автоматизації.

Окрім базових проблем, що пов'язані з масштабуванням, безпекою та управлінням конфігураціями, персоналізація ускладнюється такими викликами:

- Інтеграція зі сторонніми сервісами та додатками: Необхідність забезпечити високу продуктивність при роботі з великими обсягами даних у реальному часі, а також реалізувати стабільні та безпечні інтеграції з корпоративними системами та сторонніми сервісами. Важливим аспектом є здатність ефективно взаємодіяти з різними API та підтримувати змінювані версії сервісів.

- Автоматизація кастомізації та гнучкість конфігурації: Розробка інструментів для

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

автоматизованої конфігурації клієнтських інтерфейсів веб-додатку потребує значних ресурсів, що передбачає використання low-code або no-code платформ для швидкої адаптації додатків.

- Управління користувацькими ролями та правами доступу: Необхідність забезпечити динамічне налаштування прав доступу для визначених ролей, формування політик доступу та гнучке управління вже наявними ролями для користувачів різних клієнтів.

Сучасний ринок вимагає від розробників додатків високої адаптивності програмного забезпечення від визначені потреби та вимоги. Розвиток нових технологій, а також зростаюча популярність певних архітектурних патернів програмних продуктів вимагають від команд виваженого підходу для вибору оптимальної архітектури додатку. Ринок також пропонує розробникам вже існуючі рішення, які значно спрощують впровадження персоналізації в програмний продукт та надають широкі можливості до конфігурації та кастомізації. Оскільки час є критично важливим ресурсом при розробці та подальшій підтримці програмного продукту, це надає можливість більш гнучно підходити до розробки та економити час. До основних тенденцій наявних на ринку слід віднести:

- Впровадження штучного інтелекту та машинного навчання: Використання алгоритмів для аналізу поведінки користувачів, прогнозування потреб та автоматичної адаптації інтерфейсу вед-додатку до індивідуальних вподобань окремих користувачів.
- Адаптивні та контекстні інтерфейси: Розробка інтерфейсів, що підлаштовуються під роль та контекст кожного окремого користувача, забезпечуючи персоналізоване відображення функцій залежно від робочих процесів.
- Інтеграція low-code та no-code платформ: Спрощення процесів розробки та впровадження персоналізації за допомогою інтеграції програмного продукту з low-code та no-code рішеннями наявними на ринку, що дозволяє швидко адаптувати продукт до нових вимог знижуючи кількість зусиль, які потрібно докласти розробнику. Залучення цих платформ також дозволяє зосередити зусилля на інших частинах розробки.

Проте варто виважено підходити до імплементації персоналізації в програмний продукт, оскільки це також неси певні ризики. Неефективне впровадження персоналізації має свої наслідки та призводити до: невдоволення клієнтів та низької залученості, високих витрат на підтримку та оновлення, зростання складності додатків та ризиків. Розглянемо кожен з можливих наслідків більш детально. Нездатність швидко адаптувати функціональність до потреб клієнта негативно впливає на задоволеність користувачів та знижує конкурентоспроможність продукту на ринку. Наявність великої кількості кастомних конфігурацій ускладнює підтримку, тестування та оновлення програмного продукту. Складні конфігурації підвищують ризики помилок, уразливостей та знижують продуктивність додатків.

Застосування відповідних архітектурних патернів під час розробки дозволяє мінімізувати ці виклики, підвищуючи адаптивність та ефективність роботи програмного продукту навіть у складних сценаріях кастомізації. Враховуючи зростаючу популярність та поширену імплементацію персоналізації в веб-додатках, кількість патернів для персоналізації зростає. Проте варто виділити декілька основних патернів та підходів для впровадження персоналізації в програмних продуктах. Далі ми розглянемо та порівняємо найбільш популярні та ефективні з них, що надасть нам змогу правильно обрати відповідний патерн під конкретний програмний продукт.

Забезпечення високого рівня персоналізації в B2B веб-додатках вимагає продуманих архітектурних рішень, що дозволяють налаштовувати функціональність та інтерфейс під потреби різних клієнтів без значного ускладнення коду та архітектури. Використання правильних архітектурних патернів дозволяє досягти масштабованості, гнучкості та стабільності системи навіть при великій кількості користувачів та унікальних конфігурацій. Найбільш популярні патерни для персоналізації станом на зараз це: Multi-Tenancy, Feature Toggles, Event-Driven Architecture(EDA), Microservices, Config-as-Code. Варто розглянути кожен з них більш детально.

Патерн Multi-Tenancy є одним з найбільш поширених на сьогоднішній день та передбачає побудову єдиного додатка, що обслуговує декілька клієнтів (орендарів), кожен з яких має доступ до власних даних та налаштувань або ж виокристання спільної бази даних з

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

використанням маркування даних [4]. Основна перевага цього підходу – економія ресурсів при розробці та подальшій підтримці продукту та зниження витрат на обслуговування. Саме це відрізняє цей патерн від розподіленої архітектури, коли під кожного клієнта робиться окрема версія додатка (рис. 1). Це також допомогає оптимізувати роботу розробників та зосередитись на впровадженні нового функціоналу та подальшому розвитку продукту.

Існують три основні моделі реалізації Multi-Tenancy:

- Загальна база даних (Shared Database): Дані всіх клієнтів (орендарів) зберігаються в одній базі даних, але ізоляція досягається через маркування даних. Це спрощує масштабування додатка, але ускладнює безпеку та управління даними, що потребує додаткових зусиль команди розробників.
- Окрема база даних (Isolated Database): Кожен клієнт (орендар) має власну окрему базу даних. Це підвищує безпеку та ізоляцію даних, але збільшує витрати на адміністрування та складність архітектури.
- Гібридна модель (Hybrid): Комбінує обидва підходи, дозволяючи обирати оптимальний варіант в залежності від потреб клієнта та його можливостей щодо фінансування.

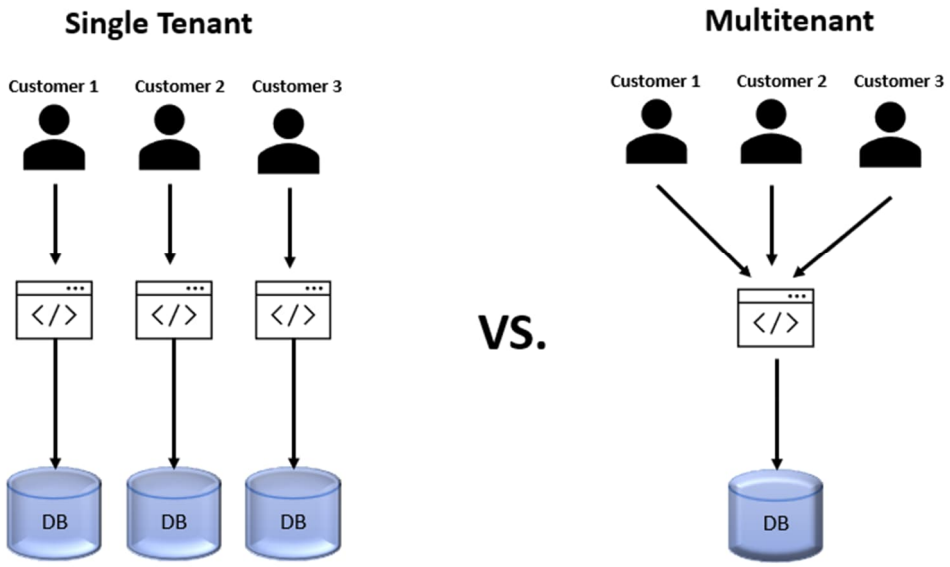


Рис. 1. Порівняння Multitenant та Single tenant патернів

Джерело: [1]

Іншим патерном який дозволяє персоналізувати користувацький інтерфейс веб-додатку або його функціонал є Feature Flags (Feature Toggles). Цей патерн дозволяє вмикати та вимикати функції у додатку без перезавантаження чи деплойменту. Це важливо для поступового впровадження нових можливостей, тестування нових функцій та адаптації інтерфейсу для різних клієнтів. Використання цього патерну забезпечує гнучкість та швидке реагування на зміни потреб клієнтів (рис. 2).

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

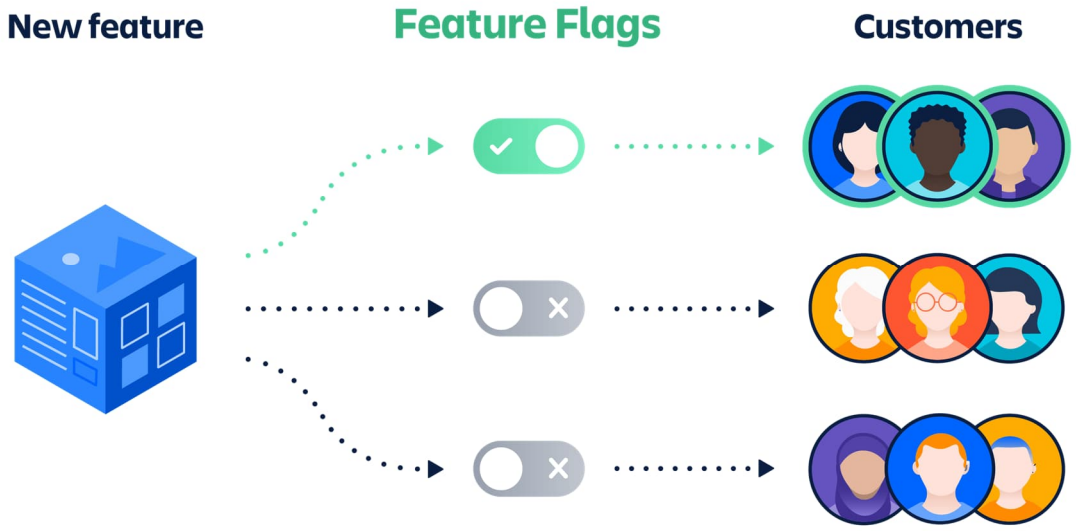


Рис. 2. Розподілення функціональності додатку на основі Feature Flags

Джерело: [2]

Також широко використовується архітектурний патерн на основі подій. Event-Driven Architecture використовується для побудови масштабованих та асинхронних систем, де події (events) слугують тригерами для виконання певних дій. У контексті персоналізації цей патерн дозволяє налаштовувати поведінку додатка в режимі реального часу, реагуючи на дії користувачів та змінюючи інтерфейс чи функціональність на льоту, що дозволяє дуже швидко підлаштовувати додаток під користувача.

Ще одним архітектурним патерном, який дуже часто використовується при розробці великих персоналізованих веб-додатків є мікросервісна архітектура. Цей патерн є однією з ключових архітектурних стратегій для побудови складних та гнучких B2B веб-додатків, що забезпечують високий рівень персоналізації. Основна ідея полягає в поділі додатка на незалежні сервіси, кожен з яких відповідає за виконання окремої бізнес-логіки [2]. Це дозволяє масштабувати додаток, впроваджувати нові функції та оновлення без переривання роботи всієї системи. Переваги мікросервісів включають незалежність команд розробників, що дозволяє працювати паралельно над різними компонентами, гнучкість у виборі технологій для кожного сервісу та можливість автоматизації процесів розгортання (CI/CD). Використання контейнеризації (Docker) та оркестрації (Kubernetes) дозволяє ефективно керувати масштабуванням, оновленнями та відновленням сервісів у разі збоїв. Однак мікросервісна архітектура при всіх своїх перевагах має свої виклики: складність управління взаємодією між сервісами, необхідність впровадження централізованих механізмів безпеки та моніторингу, а також потреба у ретельному плануванні обміну даними. Для вирішення цих проблем часто застосовуються API Gateway для управління запитами та сервіс Mesh для забезпечення безпечної комунікації між сервісами.

На ринку зараз з’являється дедалі більше сервісів, які пропонують готові рішення для персоналізації, які можна імплементувати в веб-додаток. Така форма взаємодії називається Config-as-Code. Цей паттерн передбачає управління конфігураціями за допомогою файлів визначеної структури, які зберігаються у системах контролю версій (наприклад, Git). Це дозволяє забезпечити повторюваність, надійність та контроль змін у конфігураціях, що особливо важливо для персоналізації інтерфейсів та бізнес-логіки у B2B додатках. CaC спрощує розгортання та налаштування, забезпечуючи швидке оновлення та адаптацію під різні потреби клієнтів.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Ми розглянули найбільш популярні патерни та підходи, які дозволяють впровадити персоналізацію під кожного окремого клієнта на всіх рівнях, починаючи з інфраструктури, далі функціоналом самого сервісу і закінчуючи користувацьким інтерфейсом. На основі цього слід провести порівняння цих архітекутрних патернів та підходів та визначити переваги та недоліки кожного з них. Проведення порівняння архітектурних патернів для персоналізованих B2B веб-додатків дозволяє визначити їхні переваги та обмеження з огляду на специфічні вимоги бізнесу. Зокрема, обрання патерну впливає на масштабованість, гнучкість, витрати на підтримку та швидкість реагування на зміни. Важливо враховувати особливості проєкту та його стратегічні цілі, а також необхідність інтеграції з іншими системами та забезпечення безперервної роботи. Комплексне застосування декількох патернів часто виявляється найефективнішим підходом, адже дозволяє використовувати сильні сторони кожного з них, мінімізуючи недоліки.

У табл. 1 показані основні переваги та недоліки розглянутих вище архітектурних патернів та підходів.

Вибір патерну залежить від вимог проєкту, кількості клієнтів, необхідного рівня кастомізації та бюджету на підтримку рішення. Тому необхідно проаналізувати конкретний програмний продукт і на основі отриманих даних обрати ті підходи, які матимуть найбільший позитивний вплив саме на цей програмний продукт. У більшості випадків доцільним є поєднання декількох патернів та підходів для забезпечення максимальної гнучкості та адаптивності веб-додатка.

Таблиця 1

Порівняння архітектурних патернів та підходів для персоналізації

Назва	Основні переваги	Недоліки
Multi-Tenancy	Ефективність ресурсів, економія	Складність безпеки, можливі проблеми з масштабуванням
Feature Flags	Гнучкість, можливість A/B тестування	Потрібен ретельний контроль за конфігурацією функцій
Event-Driven	Реактивність, масштабованість	Складність розробки та моніторингу
Microservices	Масштабованість, незалежність компонентів, CI/CD	Висока складність управління, потреба в оркестризації
Config-as-Code	Контроль змін, швидке налаштування	Потрібна грамотна система управління конфігураціям

Джерело: розроблено автором

Висновки. Архітектурні патерни є невід'ємною частиною побудови персоналізованих B2B веб-додатків, що забезпечують гнучкість, масштабованість та адаптивність до змінних вимог бізнесу. Комбінування Multi-Tenancy, Feature Toggles, Event-Driven, Microservices та Config-as-Code дозволяє створювати комплексні системи, які легко адаптуються до потреб клієнтів. Перспективи подальших досліджень полягають у глибшій інтеграції патернів для забезпечення безперервності бізнес-процесів, підвищення рівня автоматизації та розробки інтелектуальних механізмів самоконфігурації.

Список використаних джерел

1. Demystifying Multi-Tenant Architecture: Empowering Scalable and Efficient Software Solutions. *Medium.* URL: https://medium.com/@rajeshvelmani/demystifying-multi-tenant-architecture-empowering-scalable-and-efficient-software-solutions-a01400908985 (останнє звернення 03.04.2025).

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

2. Jira Feature Flagging | Atlassian. *Atlassian*. URL: https://www.atlassian.com/software-development/practices/feature-flagging (останнє звернення: 03.04.2025).

3. Пригода А. МІКРОСЕРВІСНА АРХІТЕКТУРА, ЯК ОСНОВА ДЛЯ СТВОРЕННЯ CRM-СИСТЕМИ. *Наука і техніка сьогодні*. 2024. № 8(36).

4. Multi-tenancy in Cloud Computing / H. AlJahdali et al. *2014 IEEE 8th International Symposium on Service Oriented System Engineering (SOSE)*, Oxford, United Kingdom, 7–11 April 2014. 2014. URL: https://doi.org/10.1109/sose.2014.50 (останнє звернення: 07.04.2025).

Роботу виконано під науковим керівництвом PhD, доцента
ДЕСЯТКО А. М.

ІНТЕГРАЦІЯ РІЗНИХ МОДУЛІВ У MVP ІНФОРМАЦІЙНОЇ СИСТЕМИ РЕСТОРАНУ: АНАЛІЗ ТЕХНІЧНИХ РІШЕНЬ ТА УПРАВЛІННЯ СУМІСНІСТЮ

БУЛИГА Б. В., 2 курс 4м група ФІТ ДТЕУ,
освітня програма «Управління проєктами програмних продуктів»

У статті розглянуто проблему інтеграції різних модулів в рамках розробки MVP (Minimum Viable Product) інформаційної системи ресторану. Проаналізовано технічні рішення для інтеграції, висвітлено стратегії управління сумісністю, включаючи стандартизацію інтерфейсів, версіонування API, тестування інтеграції та моніторинг. На прикладі інтеграції системи замовлень з системою управління запасами показано позитивний вплив інтеграції на бізнес-процеси ресторану.

The article addresses the problem of integrating different modules within the framework of developing an MVP (Minimum Viable Product) information system for a restaurant. Technical solutions for integration are analyzed, and strategies for compatibility management are highlighted, including interface standardization, API versioning, integration testing, and monitoring. Using the example of integrating an order management system with an inventory management system, the positive impact of integration on restaurant business processes is shown.

Актуальність. У сучасному динамічному середовищі ресторанного бізнесу ефективність та конкурентоспроможність значною мірою залежать від здатності оперативно адаптуватися до мінливих вимог ринку та потреб клієнтів. Впровадження інформаційної системи, яка об’єднує різноманітні функції в єдину платформу, є ключовим фактором успіху. Проте, повноцінне впровадження інтегрованої системи може бути ресурсозатратним та часомістким. Тому, концепція Minimum Viable Product (MVP), або мінімально життєздатного продукту, стає все більш популярною, дозволяючи швидко запустити базову версію системи та поступово її вдосконалювати на основі зворотного зв’язку від користувачів. Ключовим викликом при розробці MVP інформаційної системи ресторану є інтеграція різних модулів, кожен з яких відповідає за певний аспект діяльності, таких як управління замовленнями, управління запасами, облік персоналу та звітності.

Метою статті є аналіз технічних рішень щодо інтеграції модулів у рамках створення MVP інформаційної системи ресторану та розробити стратегії управління сумісністю для забезпечення ефективної та надійної взаємодії між компонентами системи.

Об’єкт дослідження є процес розробки та впровадження мінімально життєздатного програмного продукту (MVP) для автоматизації бізнес-процесів ресторанного закладу.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Предмет дослідження – технічні аспекти інтеграції окремих модулів інформаційної системи ресторану, зокрема застосування REST API, стандартизації інтерфейсів, версіонування, тестування та моніторингу для забезпечення сумісності.

Аналіз попередніх досліджень. Дослідження автоматизації та інтеграції в ресторанному бізнесі, представлені в працях Шушакової І. К. та Свистуна А.С. [1], Рябенької М. [2], розглядають загальні аспекти та економічні наслідки впровадження ІТ. Робота Веснянки В.О. [3] присвячена розробці інформаційних систем для оптимізації бізнес-процесів, а Воронець Д. [4] досліджує використання MVP для диверсифікації діяльності. Wezom [5] аналізує практичні рішення автоматизації.

Теоретичні аспекти інтеграції модульних систем висвітлені в праці Іваненка О.П.[6], управління сумісністю – у роботі Петренко Л.М. [7], а технічні рішення – у дослідженні Сидоренка В.А.[8]. Коваленко Н.П. [9] розглядає сучасні підходи до автоматизації бізнес-процесів, Мельник І.В.[10] – розробку та впровадження MVP.

Виклад основного матеріалу. Управління сумісністю модулів є критично важливим аспектом розробки MVP інформаційної системи ресторану. Ризики, пов'язані з сумісністю, включають конфлікти версій, невідповідність форматів даних та помилки в інтеграційних інтерфейсах. Для мінімізації цих ризиків необхідно використовувати стратегії стандартизації інтерфейсів, версіонування API, тестування інтеграції та моніторингу.

Стандартизація інтерфейсів передбачає розробку єдиних стандартів для API та форматів даних, що дозволяє забезпечити сумісність модулів незалежно від їх внутрішньої реалізації. Версіонування API дозволяє управляти змінами в API з використанням версій, забезпечуючи зворотну сумісність та уникнення конфліктів. Тестування інтеграції, як автоматизоване, так і ручне, дозволяє виявити та усунути помилки інтеграції на ранніх етапах розробки.

Моніторинг та логування дозволяють відстежувати помилки та проблеми інтеграції в реальному часі, забезпечуючи оперативне реагування на збої. Для автоматизації тестування та моніторингу інтеграції можна використовувати інструменти, такі як Jenkins, Docker та Prometheus.

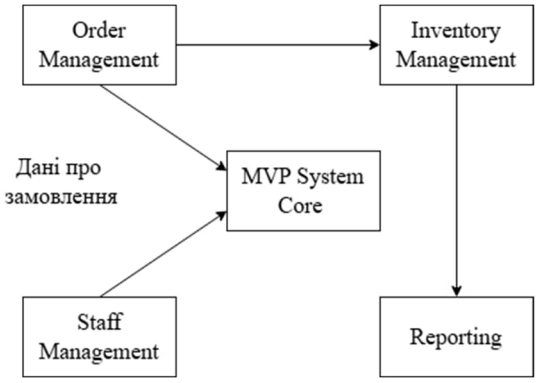


Рис. 1. Схема модулів MVP-системи ресторану

Джерело: розроблено автором

Успішне впровадження інтеграційного MVP інформаційної системи ресторану значною мірою залежить від ефективного управління сумісністю модулів. Відсутність сумісності може призвести до численних проблем, включаючи збої в роботі системи, втрату даних, незадоволеність користувачів та значні фінансові втрати. Тому, розробка та впровадження стратегій управління сумісністю є критично важливим етапом.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Ідентифікація та аналіз ризиків, пов’язаних з сумісністю модулів, є першим кроком у процесі управління. До основних ризиків належать конфлікти версій, невідповідність форматів даних та помилки в інтеграційних інтерфейсах. Конфлікти версій виникають, коли різні модулі використовують несумісні версії бібліотек або компонентів. Невідповідність форматів даних призводить до проблем при обміні інформацією між модулями. Помилки в інтеграційних інтерфейсах можуть викликати збої в роботі системи та втрату даних.

Для ефективного управління сумісністю необхідно розробити та впровадити відповідні стратегії. До ключових стратегій належать стандартизація інтерфейсів, версіонування API, тестування інтеграції та моніторинг з логуванням.

Стандартизація інтерфейсів передбачає розробку єдиних стандартів для API та форматів даних, що дозволяє забезпечити сумісність модулів незалежно від їхньої внутрішньої реалізації. Це включає визначення чітких специфікацій для запитів, відповідей та форматів даних, які використовуються при обміні інформацією між модулями.

Версіонування API є ще однією важливою стратегією управління сумісністю. Впровадження версіонування API дозволяє управляти змінами в API з використанням версій, забезпечуючи зворотну сумісність та уникнення конфліктів. При зміні API створюється нова версія, а старі версії залишаються доступними для підтримки існуючих модулів. Це дозволяє поступово оновлювати модулі, не порушуючи роботу системи в цілому.

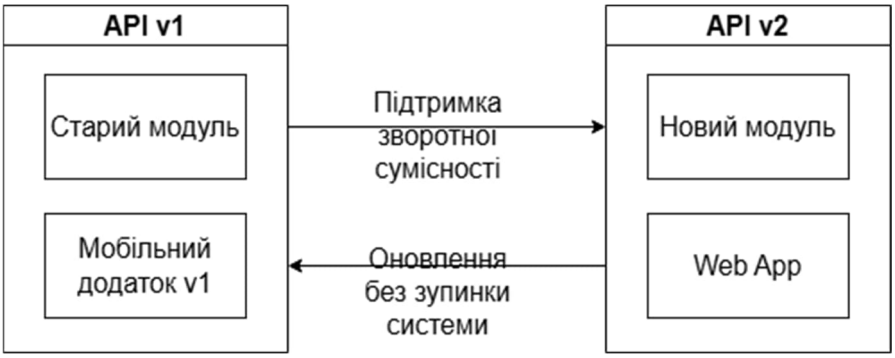


Рис. 2. Приклад версіонування REST API

Джерело: розроблено автором

Тестування інтеграції є невід’ємною частиною процесу управління сумісністю. Автоматизоване та ручне тестування інтеграції модулів дозволяє виявити та усунути помилки інтеграції на ранніх етапах розробки. Автоматизовані тести можуть бути використані для перевірки відповідності API специфікаціям, правильності обміну даними та відсутності конфліктів версій. Ручне тестування дозволяє перевірити більш складні сценарії використання та виявити проблеми, які не були виявлені автоматизованими тестами.

Моніторинг та логування відіграють важливу роль у підтримці сумісності модулів в реальному часі. Відстеження помилок та проблем інтеграції в реальному часі дозволяє оперативно реагувати на збої та запобігати їх поширенню. Логування подій, що відбуваються в системі, дозволяє аналізувати причини виникнення проблем та розробляти ефективні рішення для їх усунення.

Для автоматизації тестування та моніторингу інтеграції можна використовувати різноманітні інструменти:

– Jenkins є популярним інструментом для автоматизації процесів розробки та тестування.

– Docker дозволяє створювати контейнери з ізольованим середовищем для тестування модулів.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

– Prometheus є інструментом моніторингу та оповіщення, який дозволяє відстежувати стан системи в реальному часі та реагувати на виникнення проблем.

Наявність плану дій у випадку виникнення проблем з сумісністю є критично важливою. План дій повинен включати процедури ідентифікації, діагностики та усунення проблем. Також необхідно визначити відповідальних за виконання плану та забезпечити їх відповідними ресурсами. План дій повинен бути регулярно переглядатися та оновлюватися відповідно до змін у системі та вимогах бізнесу.

Розглянемо конкретний приклад інтеграції системи замовлень (Order Management System, OMS) з системою управління запасами (Inventory Management System, IMS) у реальному проєкті MVP інформаційної системи ресторану. Метою інтеграції було автоматизувати процес оновлення інформації про залишки продуктів на складі після кожного отриманого замовлення, мінімізуючи ручне введення даних та знижуючи ризик помилок.

Для реалізації інтеграції було використано REST API, що дозволяє обмінюватися даними між OMS та IMS за допомогою HTTP-запитів. OMS, розроблена на Node.js з базою даних MongoDB, відправляла POST-запит на endpoint IMS, розробленої на Python з базою даних PostgreSQL, з інформацією про замовлені продукти та їх кількість. IMS, отримавши запит, оновлювала відповідні записи в базі даних про залишки продуктів. Формат даних, що передавалися, був стандартизований у вигляді JSON, що забезпечило сумісність між системами.

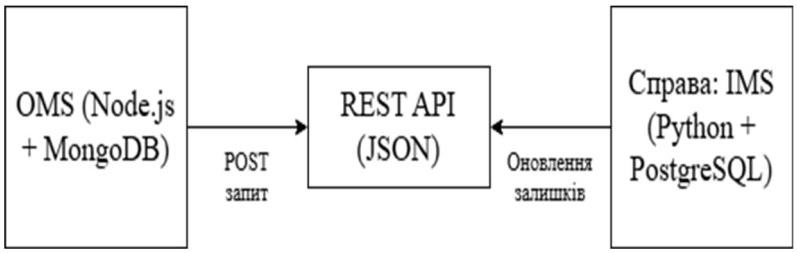


Рис. 3. Архітектура інтеграції OMS та IMS

Джерело: розроблено автором

Використані технології включали: Node.js для розробки backend частини OMS, Python для розробки backend частини IMS, MongoDB та PostgreSQL для зберігання даних, REST API для обміну даними між системами, JSON для форматування даних, Postman для тестування API та Docker для контейнеризації додатків. Також використовувалася система контролю версій Git для управління кодом та співпраці між розробниками.

Процес тестування інтеграції складався з декількох етапів. Спочатку проводилося unit-тестування окремих компонентів OMS та IMS для перевірки їхньої працездатності. Потім проводилося інтеграційне тестування, під час якого перевірялася правильність передачі даних між системами та оновлення залишків продуктів. Для цього використовувалися інструменти Postman та JMeter для генерування тестових запитів та перевірки відповідей. Під час тестування було виявлено кілька проблем з сумісністю, зокрема невідповідність форматів дат та часу, що призводило до помилок при оновленні залишків продуктів. Ці проблеми були оперативно усунені шляхом стандартизації форматів дат та часу у обох системах.

Результати інтеграції мали значний позитивний вплив на бізнес-процеси ресторану. Автоматизація процесу оновлення залишків продуктів дозволила зменшити час, необхідний для обробки замовлень, та знизити ризик помилок. Це, в свою чергу, призвело до підвищення задоволеності клієнтів, оскільки вони отримували замовлення швидше та з меншою ймовірністю помилок.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Крім того, інтеграція дозволила більш ефективно управляти запасами продуктів, зменшити витрати на закупівлю та уникнути ситуацій, коли певні продукти закінчуються під час пікового навантаження.

Оцінка впливу на бізнес-процеси проводилася шляхом аналізу ключових показників ефективності (KPI), таких як час обробки замовлень, рівень задоволеності клієнтів, витрати на закупівлю продуктів та кількість помилок при оновленні залишків. Результати показали значне покращення всіх цих показників після впровадження інтеграції, що підтверджує ефективність обраного технічного рішення та стратегії управління сумісністю. Зокрема, час обробки замовлень зменшився на 20%, рівень задоволеності клієнтів зріс на 15%, а кількість помилок при оновленні залишків знизилася на 30%.

Підсумовуючи проведене дослідження, можна стверджувати, що інтеграція різних модулів є ключовим фактором успішної реалізації MVP інформаційної системи ресторану. Результати аналізу продемонстрували, що ефективне управління сумісністю та вибір оптимальних технічних рішень мають вирішальне значення для забезпечення стабільної та надійної роботи системи.

Розглядаючи переваги та недоліки різних підходів інтеграції модулів, було встановлено, що REST API є оптимальним рішенням для MVP завдяки своїй простоті в реалізації, гнучкості та широкій підтримці. Водночас, для більш складних сценаріїв, що потребують високої надійності та масштабованості, доцільним є використання Message Queue. Вибір конкретного підходу повинен ґрунтуватися на детальному аналізі вимог проєкту, враховуючи критерії вартості, швидкості розробки, підтримки та масштабованості.

Рекомендації щодо управління сумісністю включають стандартизацію інтерфейсів, версіонування API, автоматизоване тестування інтеграції та моніторинг з логуванням. Важливим є розробка плану дій у випадку виникнення проблем з сумісністю, що дозволить оперативно реагувати на збої та запобігати їх поширенню. Впровадження цих стратегій дозволяє мінімізувати ризики, пов’язані з інтеграцією модулів, та забезпечити ефективну роботу MVP.

Список використаних джерел

1. Шушакова І.К., Свистун А.С. Інформаційні технології в управлінні підприємствами ресторанного господарства. Економіка та суспільство. 2021. Вип. 25. DOI: https://doi.org/10.32782/2524-0072/2021-25-70.
2. Рябенька М. Впровадження інформаційних комп’ютерних технологій у ресторанному бізнесі. Готельно-ресторанна справа. Економіка та суспільство. 2022. Вип. 26.
3. Веснянка В.О. Розроблення інформаційної системи для оптимізації бізнес-процесів закладу харчування: кваліфікаційна робота магістра. Харків: Харківський національний університет радіоелектроніки, 2024. 62 с.
4. Воронець Д. Впровадження MVP з метою тестування гіпотези реалізації стратегії диверсифікації діяльності підприємств галузі тимчасового розміщення та організації харчування. Modeling the Development of the Economic Systems. 2023. № 2. С. 89–95. DOI: https://doi.org/10.31891/mdes/2023-8-12
5. Автоматизація ресторанного бізнесу: рішення та переваги. Wezom. 2023. URL: https://wezom.com.ua/ua/blog/tehnologiyi-v-restorannomu-biznesi
6. Іваненко О.П. Інтеграція модульних систем в інформаційних технологіях: підходи та методи. Київ: Наукова думка, 2021. 200 с.
7. Петренко Л.М. Управління сумісністю програмних модулів: теорія та практика. Харків: Фактор, 2022. 180 с.
8. Сидоренко В.А. Технічні рішення для інтеграції інформаційних систем у сфері послуг. Львів: Львівська політехніка, 2023. 220 с.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

9. Коваленко Н.Г. Автоматизація бізнес-процесів у ресторанному господарстві: сучасні підходи. Одеса: Астропринт, 2024. 150 с.
10. Мельник І.В. Розробка та впровадження MVP у сфері громадського харчування. Дніпро: ДНУ, 2020. 175 с.

Роботу виконано під науковим керівництвом канд. екон. наук, доцента
ТИЩЕНКА Д. О.

ОБҐРУНТУВАННЯ НЕФУНКЦІОНАЛЬНИХ ВИМОГ ТА РІШЕНЬ ПРИ РОЗРОБЦІ СЕРВІСУ МЕТЕОРОЛОГІЧНИХ СПОВІЩЕНЬ З ГНУЧКИМ НАЛАШТУВАННЯМ ЦІЛЬОВИХ УМОВ

БУРЛАКОВ І. А., 2 курс 2м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглянуто процес розробки архітектури сервісу з урахуванням його життєвого циклу та нефункціональних вимог. Буде показано, як формулювати та коригувати вимоги, виходячи з доступних ресурсів і цілей проєкту. На прикладі створення сервісу метеорологічних сповіщень із гнучкими налаштуваннями умов розглянуто етапи визначення вимог, вибору технологій і сервісів, а також проєктування бази даних. Окрім того, проведено аналіз показників продуктивності різних підходів та продемонстровано процес ухвалення компромісних рішень при виборі технологій та розробці архітектури. Освітлені сучасні практики мобільної та серверної розробки й методи ухвалення реалістичних компромісних рішень.

The article discusses the process of developing a service architecture taking into account its life cycle and non-functional requirements. It will show how to formulate and adjust requirements based on available resources and project goals. The stages of defining requirements, selecting technologies and services, and designing a database are discussed on the example of creating a weather alert service with flexible conditions. In addition, the article analyzes the performance of different approaches and demonstrates the process of making trade-off decisions when choosing technologies and developing architecture. Modern mobile and server-side development practices and methods for making realistic trade-offs are covered.

Актуальність. При постанні задачі створення програмного продукту завжди мають місце компромісні рішення прийняті при розробці його архітектури. Ці рішення можуть мати походження з багатьох факторів, таких як: цілі проєкту, часові і фінансові рамки, спеціалізація команди інженерів, їх досвід та інших. Можна вважати загальноприйнятим, що оптимальними є рішення, які дозволяють досягти максимального балансу у вхідних даних на початку розробки проєкту, які обговорюються з замовником і приклади яких були наведені вище. З цього виходить, що одним із основних вмінь інженера програмного забезпечення повинно бути правильно приоритезувати, корегувати і окреслювати нефункціональні вимоги на початку обговорення технічного завдання проєкту, а також приймати відповідні архітектурні рішення, щоб забезпечити найкращу ефективність роботи команди розробників, комерційну успішність проєкту, технічну стійкість і його успішну подальшу підтримку.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Метою статті є дослідження постановки нефункціональних вимог, які приймаються при розробці сучасних сервісів, а також аналіз архітектурних і технологічних рішень, які можуть бути прийняті для задоволення цих вимог і їх вдосконалення.

Об'єктом дослідження є розробка сервісу метеорологічних сповіщень з гнучким налаштуванням цільових умов у вигляді мобільного додатку і інфраструктури для його підтримки.

Предмет дослідження – аналіз постановки нефункціональних вимог і можливих архітектурних, технологічних і сервісних рішення при розробці сучасного сервісу.

Аналіз попередніх досліджень. Дослідженню нефункціональних вимог, безсерверної архітектури, сучасних альтернативних методів розробки мобільних додатків та архітектурних рішень всередині таких додатків присвячені роботи таких авторів: Володимир Кімак, Garrett McGrath, Paul R. Brenner, Shreya A. Bhagat, Sakshi G. Dudhalkar, Prathmesh D. Kelapure, Aniket S. Kokare, Prof. Sudesh A. Bachwani.

Виклад основного матеріалу. При постанні задачі створення програмного продукту завжди мають місце компромісні рішення прийняті при розробці його архітектури. Ці рішення можуть мати походження з багатьох факторів, таких як: цілі проєкту, часові і фінансові рамки, спеціалізація команди інженерів, їх досвід та інших.

В цілях дослідження того, як можуть схвалюватись такі рішення, проаналізуємо розробку реального проєкту: сервісу метеорологічних сповіщень з гнучким налаштуванням цільових умов.

Розглянемо можливі персони і потреби користувачів такого сервісу для розуміння предметної області (рис. 1).

Рис. 1. Use-case діаграма персон і потреб можливих користувачів сервісу

Джерело: розроблено автором

В багатьох професійних сферах необхідно мати завчасні погодні дані. Також така інформація може бути потрібна для особистих потреб окремих користувачів. Більшість погодних сервісів пропонує інформацію про погодні умови по запиту, або узагальнені і обмежені сповіщення на кшталт повідомлень про опади або екстремальні явища. Але іноді користувачеві потрібно більш гнучке налаштування умов для планування конкретних подій. Саме цю задачу спрямований вирішити цей сервіс.

Так як найпоширенішим гаджетом для швидкого доступу до мережі на даний момент є смартфон, то є логічним розробити основний інтерфейс взаємодії користувача з сервісом у вигляді мобільного додатку [2].

Нефункціональні вимоги цього проєкту формуються впершу чергу тим, що проєкт має обмежені ресурси на розробку у вигляді бюджету, спеціалістів та часу. Керуючись цими

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

вхідними ресурсами, схвалюються наступні нефункціональні вимоги: проєкт повинно бути легко розробляти і дешево підтримувати. Так як у тому чи іншому вигляді ми будемо спиратись на сторонні сервіси, такі як прикладні програмні інтерфейси для метеорологічних даних, потрібно зробити можливий перехід на інші сервіси якомога легшим. Також для системи буде важливою доступність [4].

Виходячи з цих нефункціональних вимог, становиться зрозуміліше які із яких архітектурних рішень можливо обирати. Так як прогноз погоди може надаватись за 14 днів, невідкладність надсилання сповіщень може бути не настільки необхідною, щоб відслідкувати їх в режимі реального часу. Перевірку часу логічніше зробити в вигляді CRON (Command Run On) функції, яка буде періодично звіряти прогноз погоди з наявними підписками користувачів протягом дня. Так як додаток не запитує дані у серверної частини на постійній основі, ми можемо відмовитись від класичного сервера на користь безсерверних функцій, що теж повинно зменшити витрати на підтримку серверу [1].

На сьогоднішній день існує широкий вибір підходів до розробки мобільних додатків. Хоча загальноприйнятим головним підходом все ще вважається розробка нативних додатків окермо для Android и iOS, але досить великий сегмент ринку займає розробка за допомогою фреймворків, які генерують додатки на ці дві платформи з одного коду. Хоча такі методи мають свою ціну щодо показників ефективності роботи готового продукту, вони можуть мати переваги в інших показниках і бути виправданим компромісом при певних нефункціональних вимогах такого проєкту. Порівняння за головними особливості наведено у табл. 1.

Таблиця 1

Порівняння головних особливостей підходів до розробки мобільних додатків

Особливість	Нативна розробка	Flutter	React Native	Kotlin Multiplatform (KMP) + Compose
Спосіб роботи	Нативні компоненти	Рендеринг за допомогою Skia [3]	JavaScript Interface (JSI) [1]	Рендеринг за допомогою Skia [4]
Редагування на ходу	Немає	Наявне зі збереженням стану	Наявне	Немає
Мова	Kotlin\Java для Android, Swift\ Objective C для iOS	Dart	JavaScript\ TypeScript	Kotlin

Джерело: розроблено автором

Хоча об’єктивно порівняти продуктивність цих технологій дуже важко через постійну зміну версій і нововведення, різні бібліотеки і підходи, але деякі з показників на схожих додатках можливо заміряти. Так, на сайті GitHub опублікований вихідний код таких додатків для порівняння разом з інструкцією по налаштуванню тестів і результатами [5]. Результати цих тестів наведені у табл. 2–3.

Таблиця 2

Порівняння розміру бандлу додатків за використання різних технологій

Технологія	Android	iOS	Особливості
Нативна (Compose/SwiftUI)	1.463 MB	1.7 MB	
KMP (Compose)	1.463 MB	24.8 MB (v1.4) 32.2 MB (v1.5)	Вбудована Skia на Android, додана на iOS
Flutter	6.828 MB	17.9 MB	Додана Skia на iOS + фреймворк Flutter

Джерело: [5]

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Таблиця 3

Порівняння часу запуску Android додатка за використання різних технологій

Додаток	Android			iOS
	мін.	медіана	макс.	медіана
Native	408.7 мс.	413.1 мс.	423.1 мс.	1.441 с.
KMP	403.6 мс.	425.3 мс.	466.4 мс.	1.618 с.
Flutter	600.5 мс.	634.2 мс.	649.8 мс.	1.608 с.

Джерело: [5]

Як видно із цих порівняння, хоча фреймворки гірше дещо гірше з точки зору швидкодії та використання ресурсів, але завдяки таким особливостям, як використання однієї мови і миттєвому відображеню змін, вони можуть дати перевагу у швидкості розробки, порівняно з нативними методами. Виходячи з нефункціональних вимог, що були описані вище, а саме: фінансовому і людському ресурсу проєкту, його цілям і вимогам щодо підтримки, можемо зробити висновок, що технологія Flutter є збалансованим рішенням для такого проєкту. На рис. 2 можемо бачити як може виглядати спрощена блок-схема системи з описаної вище предметної області.

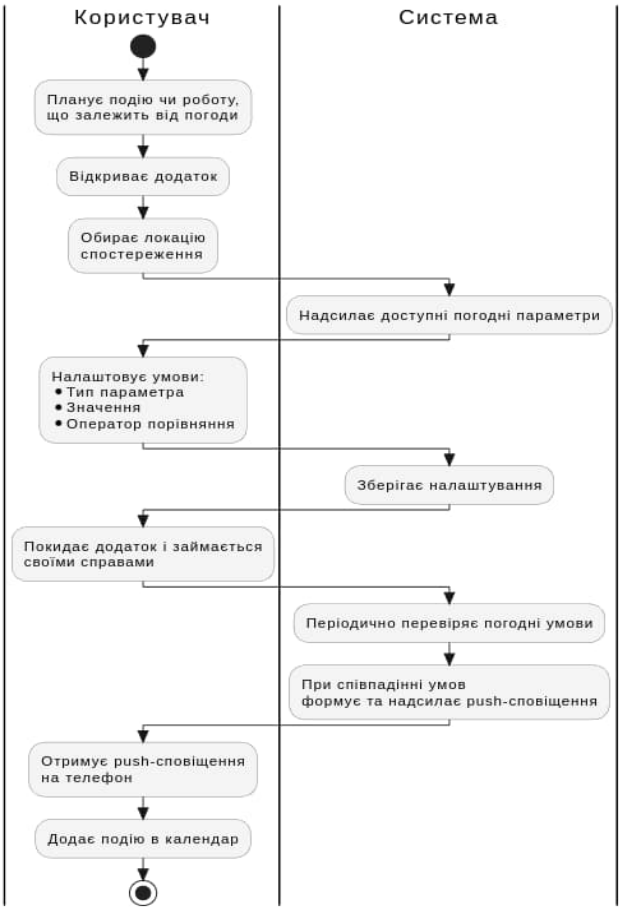


Рис. 2. Загальна блок-схема взаємодії користувача з системою

Джерело: розроблено автором

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Керуючись процесом з блок-схеми на рис 2., була розроблена архітектура сервісу показана на рис. 3.

Виходячи з вимог до доступності додатку, була обрана архітектура з офлайн сховищем та дублюванням інформації завдяки репозиторію та бібліотеці Brick, що забезпечить роботу при відсутності підключення до мережі [2]. А сервіс Supabase забезпечить контрольованість у виді відкритого вихідного коду та можливості самостійного хостингу, а також зручність налаштування через інтерфейс [7]. За обробку бізнес логіки та передачу даних між репозиторієм і відображенням її у інтерфейсі відповідає бібліотека для керування станом – Riverpod, яка зможе значно полегшити підтримку додатка в мабуньому [6].

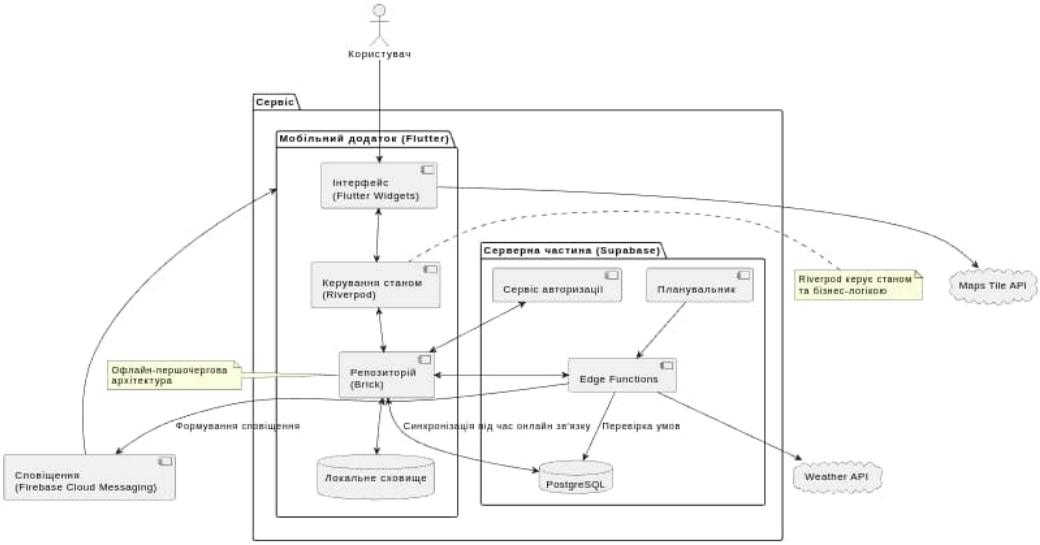


Рис. 3. Архітектурна діаграма сервісу

Джерело: розроблено автором

Виходячи з архітектурного рішення про вибір релятивної SQL (*Structured query language*) бази даних, може бути спроєктована схема бази даних, зазначена на рис. 4.

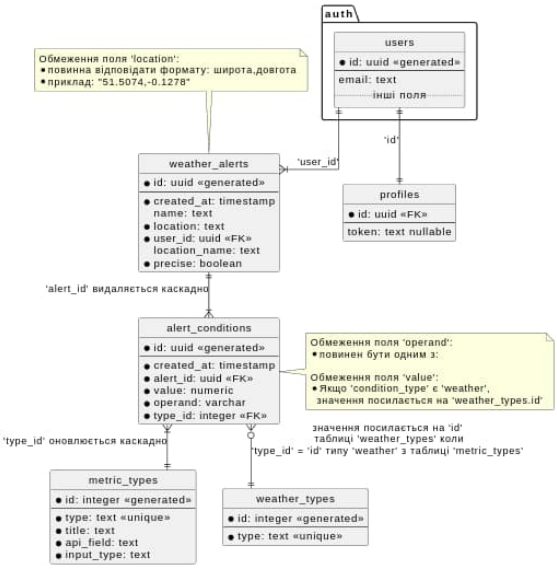


Рис. 4. Схема бази даних

Джерело: розроблено автором

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

База даних нормалізована і відповідає принципам до 3NF включно для уникнення надлишковості даних, забезпечення цілісності даних та ефективного оновлення даних. Декотрі поля, такі як «location_name», навмисно денормалізовані, так як у випадку «location_name» це дозволяє зменшити кількість запитів геокодування та запобігти неочікуваним для користувача змінам у назвах.

Висновки. Однією з найважливіших навичок інженера програмного забезпечення є розробка архітектури проєкту, яка буде задовольняти потреби його життєвого циклу і нефункціональних вимог. Не менш важливим є вміння формулювати та корегувати ці вимоги, виходячи з вхідних ресурсів на розробку і цілей проєкту. На прикладі розробки сервісу метеорологічних сповіщень з гнучким налаштуванням цільових умов у цій статті був продемонстрований весь цей процес від стадії визначення вимог, до вибору технологій і сервісів, проєктування бази даних. Були проаналізовані показники продуктивності різних підходів до розробки і на основі аналізу був продемонстрований процес компромісних рішень у виборі технологій і розробці архітектури. Це дозволило побачити картину сучасної мобільної та серверної розробки, а також те, як приймати актуальні і реалістичні рішення. У результаті аналізу вдалось прийняти рішення про використання сучасного фреймворку Flutter, керування станом завдяки Riverpod, офлайн сховища та репозиторія Brick, безсерверних функцій на основі сервісного рішення Supabase. Це дозволило задовольнити визначені нефункціональні вимоги у вигляді гнучкості, бюджетної і часової економії та доступності. Також архітектурні рішення прийняти при розробці додатку дозволили задовольнити вимогу доступності, роблячи додаток можливим для використання навіть в режимі офлайн. Цей процес дозволив досягти поставлених цілей і дав краще розуміння про те, як сьогодні можна робити успішні рішення при розробці проєкту.

Список використаних джерел

1. About the new architecture · react native [Електронний ресурс] // React Native · Learn once, write anywhere. – Режим доступу: https://reactnative.dev/architecture/landing-page (дата звернення: 28.03.2025). – Назва з екрана.
2. Brick docs [Електронний ресурс]. – Режим доступу: https://getdutchie.github.io/brick/#/introduction/about (дата звернення: 31.03.2025). – Назва з екрана.
3. Flutter architectural overview [Електронний ресурс] // Docs | Flutter. – Режим доступу: https://docs.flutter.dev/resources/architectural-overview (дата звернення: 28.03.2025). – Назва з екрана.
4. Get started with kotlin multiplatform | kotlin multiplatform development [Електронний ресурс] // Kotlin Multiplatform Development Help. – Режим доступу: https://www.jetbrains.com/help/kotlin-multiplatform-dev (дата звернення: 28.03.2025). – Назва з екрана.
5. GitHub – jacobras/flutter-vs-native-vs-kmp: Flutter versus Native versus Kotlin Multiplatform [Електронний ресурс] // GitHub. – Режим доступу: https://github.com/jacobras/flutter-vs-native-vs-kmp (дата звернення: 28.03.2025). – Назва з екрана.
6. GitHub – rrousselgit/riverpod: a reactive caching and data-binding framework. riverpod makes working with asynchronous code a breeze. [Електронний ресурс] // GitHub. – Режим доступу: https://github.com/rrousselGit/riverpod (дата звернення: 31.03.2025). – Назва з екрана.
7. Supabase docs [Електронний ресурс] // Supabase | The Open Source Firebase Alternative. – Режим доступу: https://supabase.com/docs (дата звернення: 31.03.2025). – Назва з екрана.

Роботу виконано під науковим керівництвом канд. екон. наук, старшого викладача
БУЛЬБИ С. А.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

ГНУЧКІ МЕТОДОЛОГІЇ УПРАВЛІННЯ ПРИ СТВОРЕННІ E-COMMERCE IT-ПРОДУКТІВ

ВОЛОШИНА А. Р., 2 курс 3мз група ФІТ ДТЕУ,
освітня програма «Управління проєктами програмних продуктів»

У статті розглянуто основні аспекти використання гнучких методологій управління проєктами при створенні e-commerce IT-продуктів. Зауважено в загальному значні переваги впровадження гнучких Agile-підходів для e-commerce сфери. Розглянуто ефективність та дієвість результатів застосування методологій Scrum та Kanban для технічних рішень електронної комерції. Запропоновано підходи для вибору гнучких методологій під відповідні потреби ефективного управління e-commerce IT-продуктами.

The article discusses the key aspects of using agile project management methodologies in the development of e-commerce IT products. In general, the significant benefits of implementing flexible agile approaches for the e-commerce sector are noted. The efficiency and effectiveness of the results of applying Scrum and Kanban methodologies to e-commerce technical solutions are considered. Approaches for selecting flexible methodologies for the relevant needs of effective management of e-commerce IT-products are suggested.

Актуальність. Електронна комерція або ж e-commerce є однією з тих галузей IT-сфери, що стрімко розвивається та адаптується під потреби цифрового ринку та створює безупинний потік купівлі та продажу товарів через мережу Інтернет. Варто зазначити, що за останні роки відбулись різкі зміни у функціонуванні галузі електронної комерції, адже початковою ланкою таких змін стала пандемія COVID -19, що призвело вже до більш глобальних змін споживчої поведінки користувача: цифрова регулярна взаємодія користувача та продукту.

Глобальні зміни на ринку електронної комерції зумовили необхідне впровадження та вдосконалення IT-рішень даної сфери. Відповідним чином при створенні цифрових платформ постає ряд вимог, наприклад, персоналізація контенту та зручність користувацького досвіду, тобто пошук нових методів і шляхів адаптації під задані запити користувача. Зазначені фактори стали рушійним механізмом упровадження гнучких Agile підходів у розробці платформ електронної комерції для постійної адаптації функціоналу, технічних реакцій на швидкість апдейтів запитів користувачів та більш гнучкого й ефективного розвитку бізнес-процесів.

Сфера розробки IT-продуктів у галузі електронної комерції є однією з першовідкривачів у застосуванні гнучких методологій управління проєктами для домінування на ринку та адаптації до потреб цільової аудиторії. Необхідно додати, що поширення та упровадження гнучких Agile-методологій є критичним елементом підвищення якості та ефективності створюваного продукту на всіх етапах.

Метою статті є дослідження особливостей застосування гнучких методологій управління при створенні e-commerce IT-продуктів для покращення ефективності та адаптивності їх функціонування на ринку технологій.

Об'єктом дослідження є процес управління проєктами при створення IT-продуктів у галузі електронної комерції.

Предметом дослідження є гнучкі методології управління проєктами.

Аналіз попередніх досліджень. Дослідженню гнучких методологій управління проєктами й важливим аспектам використання присвячено праці вітчизняних і закордонних науковців-практиків: О.М. Мурашко [8], де показано способи застосування методу Scrum для вітчизняного IT-бізнесу, Л.В. Дорош [9], де проілюстровано методи гнучкого управління проєктами саме в e-commerce вітчизняного ринку, S. Vijay [10], де проаналізовано сучасні

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

тенденції e-commerce розробці та гнучкі методи для управління таких проєктів, M.Kucher [11], де викладено дослідження про застосування методології Agile для оптимізації життєвого циклу e-commerce продуктів та J. Highsmith [12], де описано підходи гнучких методологій управління при цифровій трансформації бізнесу та ін.

Виклад основного матеріалу. Цифровий бізнес стикається з розповсюдженням тенденції потоку онлайн купівлі-продажу товарів і послуг, що з кожним періодом часу стає центральним елементом конкуренції на ринку та мінливої поведінки користувача. Упровадження e-commerce платформ дозволяє мінімізувати проблематику збереження й стабільності залишатися на ринку й продовжувати ефективне зростання.

Для глибшого розуміння для подальшого дослідження e-commerce варто зазначити, які технологічні продукти можна віднести до даної галузі [1]:

- Онлайн-магазини – платформи для купівлі та продажу найрізноманітніших товарів та послуг з використанням різних шаблонних засобів розробки.
- Сервіси оголошень – платформи для здійснення як великих, так і малих транзакцій з можливістю комунікації користувачів.
- Цифрові продукти – інтелектуальні технічні продукти для миттєвого завантаження та використання через безліч онлайн-платформ.
- Платіжні сервіси – платформи для здійснення великої кількості системних платежів з урахуванням потреб користувача.

Варто додати, що контекст постійних і непередбачуваних змін у поведінці користувача у цифровому просторі висвітлює необхідність упровадження гнучких методологій на всіх етапах бізнес-процесів, що надає перевагу у швидких реагуваннях на мінливість змін попиту та пропозиції на ринкових потребах, а також оптимізації процесів продажу, взаємодії з клієнтами та конкурентній здатності компанії.

Описана проблема, що потребує ефективного рішення полягає у впровадження таких гнучких Agile-методології, реалізацією якої безпосередньо є Scrum, Kanban, Lean підходи, що дозволять й розширюють можливості адаптивних підходів планування, розробки, моніторингу окремих проєктів.

Гнучкість даної методології полягає у здатності коригування ключових стратегій на постійній основі, прийнятті важливих рішень у роботі над складними проєктами, реагування на неточні й невиокремлені проблеми, адаптації до непередбачуваної змін. Не менш важливим є врахування елементу несподіваної зміни як можливості трансформації бізнесу на ринку електронної комерції. [2] Для детальнішого дослідження впливу методології Agile та її методів для галузі електронної комерції варто розглянути теоретичний та практичний контекст їхнього безпосереднього функціонування.

Agile – одна з гнучких та одночасно вимогливих методологій, що має застосування у процесах розробки програмних продуктів зі здатністю адаптуватись до незапланованих змін і вимог, завдяки чому процеси розподілено по спринтах, тривалістю від 1 до 4 тижнів. Залучені у методологію процеси сприяють фокусу на командній роботі та екологічній підзвітності, самоорганізації учасників бізнес-процесу за власну зону відповідальності, роботу над проєктом короткими циклами з культурою зворотнього фідбеку.

Варто надати уточнення, що проєкт може складатися з певної кількості спринтів, що напряму пов’язано з об’ємом проєкту та потреб користувача й клієнта, тож такий комплексний структурований підхід забезпечує якісну й детальну роботу над e-commerce продуктом, що вимагає максимального залучення у процесі створення цифрового рішення. Загальну схему взаємодії основних процесів спринту Agile-методології продемонстровано на рис. 1.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Рис. 1. Схема спринту Agile-методології

Джерело: [3]

Завдяки ітеративному формату методології, у даному випадку розробки продуктів у галузі електронної комерції, головні етапи проєкту виконуються на початку роботи, а відповідні подальші етапи реалізації розбито на окремі спринти, тобто кожна ітерація містить планування, розробку, тестування та інтегрування до всього цілісного продукту. Даний формат надає можливість швидкого включення на реагування змін у продукті й безпосереднього додавання й створення нового функціоналу без радикального впливу на загальну створену архітектуру продукту.

Для цілісного розуміння переваг гнучкої методології на e-commerce проєкті варто розглянути практичне застосування Agile на прикладі веб-додатку доставки їжі Zomato. Проблематикою постає те, що для даного додатку не визначено основні вимоги, але відомо лиш технології використання, тобто для розробки окремого функціоналу додатку буде використано поетапну й циклічну модель розробки. Кожна з перелічених нижче функцій має пройти через такі етапи, як ідея, проєктування, реалізація, тестування, впровадження [4].

– Функція «Відгук про заклад» – перший етап надає користувачу надавати оцінку за якістю їжі, на наступних версіях упровадження нового функціоналу надає можливість оцінювання за якістю обслуговування.

– Функція «Фільтр пошуку» – перший етап надає користувачу здійснювати пошук за видом бажаної кухні, а вже в наступних ітераціях налаштування за рейтингом.

– Функція «Онлайн-замовлення» – перший етап надає користувачу можливість здійснення замовлення через телефонний дзвінок, а вже на наступних етапах передбачено впровадження онлайн-оплати та оформлення замовлення відповідно попиту на цю функцію для користувача.

Крім того, важливо окремо відмітити можливість тестування гіпотез на основі попиту користувачів на ту чи іншу функціональну частину для майбутнього розвитку продукту. На рис. 2 проілюстровано схему роботи над функціонал проєкту за методологією Agile.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

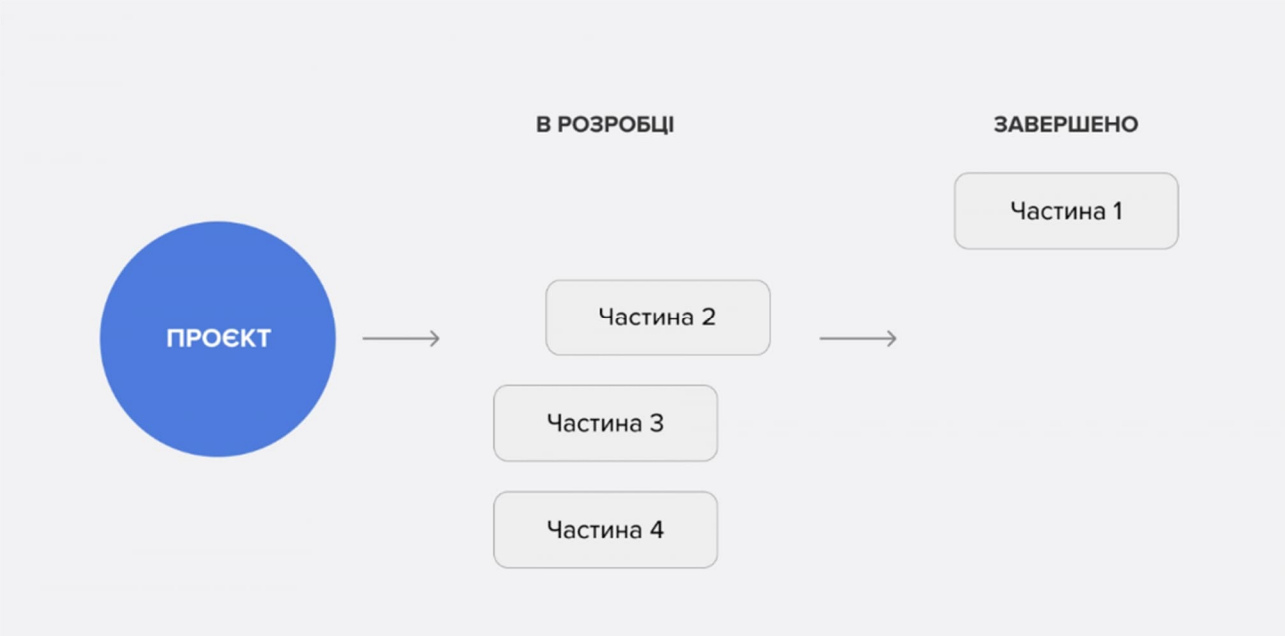


Рис. 2. Схема роботи за Agile

Джерело: [5]

На базі розглянутого прикладу застосування гнучкої методології управління проєктом видно, що Agile застосовує модель поступового наростання цінності продукту для користувача з метою удосконалення самого цифрового продукту, переваг інвестування ресурсів серед конкурентів й забезпечення найкращого користувацького досвіду.

У межах розробки e-commerce гнучке управління IT-продуктом можна реалізувати за допомогою таких фреймворків або ж методів як Scrum та Kanban, що напряму є складовими методології Agile. Спільною й провідною рисою зазначених методів є підвищення ефективності та якості продукту для кінцевого споживача. Доречно додати, що безперечно у сфері електронної комерції, за рахунок, невпинної динаміки змін вимог та потреб користувача, поєднання двох методів Scrum та Kanban гарантує гнучкість та ціннісну пропозицію продукту на ринку. Для доцільного розуміння переваг використання описаних методів варто розглянути як теоретичний та практичний кейси кожного з них.

Scrum-фреймворк відноситься до категорії найпоширеніших адаптивних методів управління проєктами, у якому швидка й відкрита взаємодія між командою, що розробляє продукт та самим клієнтом. Розробка продукту поділяється на відповідні ітерації – спринти, у кінці кожного з яких має бути реалізовано результат роботи, але попередньо має бути сформовано беклог продукту електронної комерції – ієрархічно поділених вимог до IT-продукту, що відповідають потребам бізнесу. [6]

Саме для електронної комерції Scrum є потужним інструментом регулярного надання проміжних результатів з швидким реагуванням на зворотній зв’язок та критичні точки зі сторони користувача та клієнта. Даний підхід відкриває два варіанти реалізації продукту: з ідейного бачення та поетапного вдосконалення, що надає способи створення й тестування нових можливостей уже в процесі функціонування продукту та визначенні й внесенні потрібних змін у наступних ітераціях запуску нового функціоналу.

Для розгляду впровадження Scrum-методу у галузі e-commerce було взято веб-додаток для замовлення книг онлайн та проілюстровано два спринти у процесі розробки зазначеного продукту (рис. 3).

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Крім того, важливо зазначити, що на наданій вище схемі спринти ілюструють послідовно-структурований підхід до реалізації певного функціоналу веб-додатку за обраним гнучким фреймворком Scrum, згідно якого окремий етап характеризується логічно завершеним та може бути наданий користувачу для подальшого використання й надалі покращення зі сторони команди розробників.

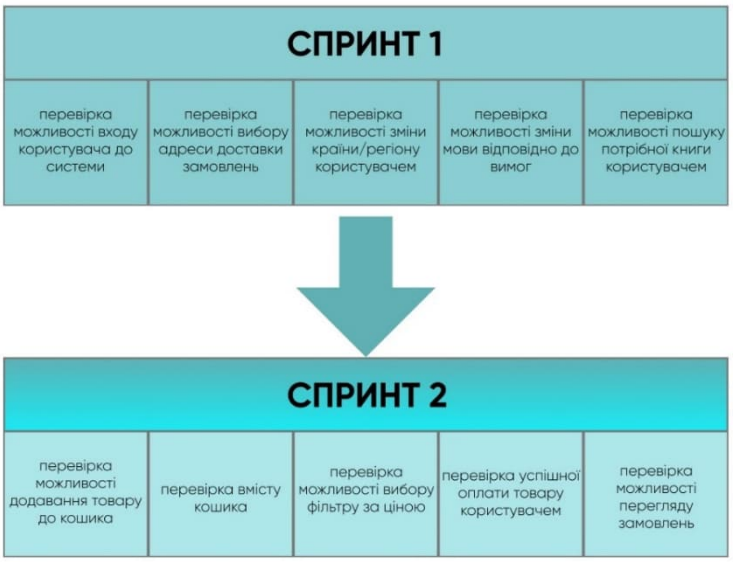


Рис. 3. Схема спринтів Scrum-методу

Джерело: розроблено автором

Другим методом реалізації гнучкого управління проєктом є Kanban, що передбачає безперервне виконання завдань зі зміною стану статусів виконання задач, обмеженнями кількості активних задач та, що важливо виокремити, загальну візуалізацію процесу розробки IT-продукту. У контексті галузі електронної комерції використання даного методу є досить гнучким з чіткою візуалізацією робочих процесів для відслідковування динаміки робочого процесу над продуктом. Фундаментальною основою Kanban є спеціально створена дошка з стовпцями, що відповідають певному етапу виконання задач з відображеним статусом роботи над задачею [5].

При розробці e-commerce продуктів варто враховувати такі фактори як швидкі ринкові зміни та поведінку користувача, тож зазначена візуалізація робочого процесу дозволяє залученій команді у бізнес-процес відслідковувати статуси поставлених задач. Безперебійна тенденція розвитку продуктів електронної комерції вимагає стабілізації й керування рівномірним розподілом задач для реалізації та чіткого обмеження їхньої кількості, у випадку одночасного статусу виконання.

Важливо відмітити, що упровадження фреймворку Kanban для управління продуктом електронної комерції надає можливості автономії кожного з учасників робочого процесу, що сприяє кращій ефективності результатів розробки продукту, колективному аналізу проблемних точок і досягнення стабільного перебування й конкурентоспроможності на ринку. [6]

Для розгляду впровадження Kanban-методу у галузі e-commerce було взято веб-додаток для замовлення книг онлайн та проілюстровано дошку зі статусами задач у процесі розробки певного функціоналу зазначеного продукту (рис. 4).

Доречним відмітити, що на наданій вище схемі прослідковується повна відсутність задач невиконаного статусу, тобто варто зробити припущення, що відбувалась повна реалізація заданого функціоналу веб-додатку за обраним гнучким фреймворком Kanban відповідно до потреб замовника.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Рис. 4. Дошка задач Kanban-методу

Джерело: розроблено автором

На основі вище поданого аналізу методів для реалізації Agile-підходу виникає суттєве питання вибору найкращого з фреймворків Scrum або Kanban під потреби бізнесу галузі електронної комерції. У випадку ж застосування Kanban-методу передбачено більшу гнучкість та адаптивність безупинного потоку різних задач та їхньої візуалізації на спеціальній дошці для стабілізації робочого процесу. [7]

Можливим орієнтиром для вибору поточного методу гнучкого управління e-commerce продуктом є визначення таких кейсів: E-commerce продукт передбачає структурне планування, регулярну перевірку кожного етапу та отримання звітності по статусах виконання задач й контрольоване підпорядкування дедлайнам, передбаченим є впровадження Scrum. E-commerce продукт передбачає істотні зміни на більшості етапах реалізації, активний пошук нових способів розвитку й тестування функціоналу, гнучкість контрольованих етапів виконання задач, передбаченим є впровадження Kanban.

Для додаткового наочного пояснення вибору того чи іншого методу гнучкого управління проєктом у галузі електронної комерції було розроблено порівняльну характеристику Scrum та Kanban, що зображено на рис. 5.

SCRUM	KANBAN
планування проводиться на всьому етапі процесу	планування як такого немає, оскільки завдання не оцінюються
існує концепція та планування спринтів	спринти не використовуються
дозволені постійні зміни	у разі значних змін може бути неефективним
оскільки в кожному спринті фіксована кількість завдань, вартість проекту легше передбачити	через відсутність оцінювання завдань, вартість проекту менш точна
після кожного спринту відбувається невеликий реліз	продукт чи зміни доставляються користувачам регулярно

Рис. 5. Порівняльна характеристика Scrum та Kanban

Джерело: розроблено автором

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Надане порівняння двох методів Agile-підходу уможливлює здійснення вибору між Scrum та Kanban або ж навпаки поєднання для забезпечення ефективного гнучкого управління проєктом електронної комерції відповідно до потреб бізнесу та попиту користувачів.

Висновки. Упровадження гнучких методологій управління проєктами, що включають методи Scrum та Kanban є ефективним механізмом реалізації ІТ-продуктів галузі електронної комерції. Використання Agile-підходів суттєво впливають на адаптацію розробки до швидких й мінливих вимог ринку, а також непередбачуваних користувацьких потреб з можливістю ефективного функціонування й розширення цифрового рішення.

Список використаних джерел

1. E-Commerce – Що таке електронна комерція? | Wezom. ІТ-компанія повного циклу розробки програмних продуктів WEZOM - Київ, Україна. [Електронний ресурс]. – Режим доступу: https://wezom.com.ua/ua/blog/elektronna-komertsiya (дата звернення: 16.03.2025)
2. EU Omnibus: How Should Businesses Respond?. Forbes. [Електронний ресурс]. – https://www.forbes.com/sites/sap/2025/04/04/uncertainty-of-the-eu-omnibus-and-how-businesses-should-respond/ (дата звернення: 16.03.2025)
3. Agile та ITSM – чому варто інтегрувати інструменти - BIZMAG. [Електронний ресурс]. – https://bizmag.com.ua/devops-integraciia-agile-ta-itsm/ (дата звернення: 20.03.2025)
4. Debajyoti. Agile in the context of E-commerce platform. Medium. [Електронний ресурс]. – https://medium.com/@ImDebajyotiDas/agile-in-the-context-of-e-commerce-platform-6cc430909d8b (дата звернення: 21.03.2025)
5. Створення інтернет-магазину: ефективні підходи та методи. Створення і розробка інтернет-магазинів від Brander. [Електронний ресурс]. – https://brander.ua/blog/stvorennya-internet-mahazynu-efektyvni-pidkhody-ta-metody (дата звернення: 21.03.2025)
6. Pokala S. Implementing Scrum and Kanban Approaches for E-Commerce Web Application: An Agile Framework. Academia.edu. [Електронний ресурс]. – file:///C:/Users/% D0% 90%D0%BB%D0%BB%D0%BE/Downloads/Implementing_Scrum_and_Kanban_Approaches.pdf (дата звернення: 25.03.2025)
7. Türk M. In Ecommerce Agile Always Wins. Medium. [Електронний ресурс]. – https://medium.com/drlrdsen/in-ecommerce-agile-always-wins-bcf18775db1 (дата звернення: 28.03.2025)
8. Мурашко О.М. Адаптація Scrum-методології в українських ІТ-компаніях // Економіка та суспільство. – 2022. – № 38. – С. 215–221.
9. Дорош Л.В. Особливості використання гнучких методів управління проєктами у цифровому підприємництві // Бізнес-інформ. – 2021. – № 12. – С. 102–108.
10. Vijay S. Agile in the Age of Digital Commerce. – Springer, 2021.
11. Kucher M. DevOps for Digital Commerce: Integration of Agile for Scalable Delivery. – Wiley, 2020.
12. Highsmith J. Agile Project Management: Creating Innovative Products. – Addison-Wesley, 2010.

Роботу виконано під науковим керівництвом PhD, доцента
ДЕСЯТКО А. М.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

ОГЛЯД ТЕОРЕТИЧНИХ ПІДХОДІВ ДО РОЗРОБКИ MVP ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПІДПРИЄМСТВ ТОРГІВЛІ З ДВОСПРЯМОВАНОЮ СТРУКТУРОЮ

ГАМАЛІЙ Б. С., 2 курс 4м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті буде розглянуто теоретичні засади розробки мінімально життєздатного продукту (MVP) програмного забезпечення підприємств торгівлі з використанням двоспрямованої структури. Проаналізовано основні переваги застосування такого підходу, зокрема можливість паралельної роботи над компонентами системи, прискорення процесів адаптації до змін ринку та покращення керування інформаційними потоками. Визначено ключові характеристики двоспрямованої структури MVP та запропоновано практичні рекомендації для її впровадження в діяльність торговельних підприємств.

The article will consider the theoretical principles of developing a minimum viable product (MVP) for software for retail enterprises using a bidirectional structure. The main advantages of using such an approach are analyzed, particularly the possibility of parallel work on system components, acceleration of adaptation processes to market changes, and improvement of information flow management. The key characteristics of the bidirectional MVP structure are determined, and practical recommendations for its implementation in the activities of retail enterprises are proposed.

Актуальність дослідження. У сучасних умовах цифрової трансформації бізнес-середовища торговельної справи компанії стикаються з необхідністю імплементації програмного забезпечення, яке сприяє оптимізації бізнес-процесів, підвищенню операційної ефективності та покращенню взаєморозуміння зі споживачами. Одним із дієвих підходів до створення такого програмного забезпечення є розробка мінімально життєздатного продукту (MVP), що дозволяє швидко протестувати ключові функціональні можливості системи та отримати зворотний зв'язок від користувачів. Використання двоспрямованої структури у процесі створення MVP забезпечує більш гнучке управління інформаційними потоками, паралельне виконання завдань та підвищує адаптивність системи до динамічних змін ринкового середовища. Актуальним є проведення досліджень у галузі необхідності систематизації теоретичних підходів до побудови двоспрямованої структури, визначення їхніх переваг, обмежень та умов ефективного застосування в інформаційних системах підприємств торгівлі.

Постановка проблеми. Незважаючи на зростаючу популярність концепції MVP, дослідження у сфері розробки програмного забезпечення для підприємств торгівлі залишаються фрагментарними та недостатньо систематизованими. Існує потреба у формуванні цілісного уявлення про теоретичні підходи до розробки MVP з урахуванням специфіки торговельних процесів, структури бізнесу та особливостей управління даними. Однак відсутність узагальненої теоретичної бази ускладнює розробку практичних рекомендацій для реалізації такого роду ПЗ, що обумовлює необхідність проведення комплексного огляду існуючих підходів.

Метою дослідження є проведення системного огляду теоретичних підходів до розробки MVP програмного забезпечення для підприємств торгівлі з двоспрямованою структурою, визначення їхніх ключових характеристик, переваг та недоліків, а також формування рекомендацій щодо їхнього застосування у практичній діяльності.

Об'єкт дослідження – процес розробки MVP програмного забезпечення для підприємств торгівлі.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Предмет дослідження - теоретичні засади розробки MVP програмного забезпечення для підприємств торгівлі з двоспрямованою структурою.

Аналіз попередніх досліджень. Дослідженню концепції мінімально життєздатного продукту, її ролі в процесі розробки програмного забезпечення та адаптації до ринкових умов присвячено праці як вітчизняних, так і закордонних науковців: С.В. Бєляєва, М.О. Гордієнка, І.В. Глущенка, V. Lenarduzzi, D. Taibi, E. Ries, A. Nguyen-Duc, P. Abrahamsson та ін. У роботах зазначених авторів розглядаються підходи до побудови MVP у контексті інноваційного підприємництва, двосторонніх платформ та гнучкої розробки програмного забезпечення. Значну увагу приділено ітеративним методам, мінімізації початкових витрат та формуванню зворотного зв'язку з користувачем.

Виклад основного матеріалу. У контексті платформної торгівлі, де одночасно взаємодіють дві категорії користувачів (продавці й покупці), застосування MVP вимагає чітких методологічних підходів. У цій частині ми розглянемо наукові моделі, що використовуються для створення MVP, а також приклади їх успішного застосування в реальних компаніях.

1. Інкрементальна модель передбачає поступове створення продукту через додавання функціональних частин. Як зазначають Sommerville & Sawyer, «інкрементальна стратегія дозволяє зосередитись на найважливіших функціях і отримати ранній зворотний зв'язок» [1]. Для платформної моделі це означає запуск лише базових механізмів обміну між сторонами, наприклад, реєстрація, профілі, базовий пошук і первинна форма транзакції.

2. Модель Customer Development (модель розвитку клієнта), запропонована Стівом Бланком, ґрунтується на ідеї створення продукту на основі реальних потреб клієнта, а не внутрішніх уявлень команди. Як підкреслює Blank: «Startups fail not because they lack a product, but because they lack customers and a proven business model» [2]. У випадку з двосторонніми платформами ця методологія допомагає ідентифікувати цінність для кожної зі сторін – наприклад, кур'єрів і користувачів у Glovo.

3. Спіральна модель передбачає ітеративну розробку з постійною оцінкою ризиків і збіркою зворотного зв'язку після кожного витка. Як вказує Boehm (1988), це «універсальний підхід для розробки в умовах високої невизначеності» [3]. Цей підхід добре узгоджується із запуском торгових платформ, де продукт мусить поступово вчитися на поведінці користувачів.

4. Застосування Agile/Scrum у контексті двосторонніх платформ. Agile-фреймворк та Scrum зокрема дозволяють реалізувати MVP через серії коротких ітерацій (спринтів), у яких фокус зміщується на найбільш критичну для клієнта функціональність. За дослідженням Nguyen-Duc et al.: «Agile principles support MVP development through continuous delivery and early validation of assumptions» [4]. Особливо ефективно Scrum працює для команд, що будують цифрові платформи, які потребують частих перевірок гіпотез і масштабування.

Приклади успішного застосування MVP у торгівлі з двосторонньою структурою:

1. Glovo – початковий MVP платформи включав лише функції замовлення їжі з обмеженого переліку ресторанів. Це дало змогу перевірити попит у реальних умовах із мінімальними витратами [5].

2. Etsy – маркетплейс для рукоділля було запущено з простою платформою, що дозволяла продавцям розміщувати товари, а покупцям – замовляти через інтерфейс з базовими фільтрами [6].

3. OLX – спочатку розроблений як проста дошка оголошень, MVP OLX дозволяв створення й перегляд оголошень без складної системи верифікації чи аналітики. Це спростило масштабування на нові ринки [7].

Запуск MVP дозволяє уникнути надлишкового функціоналу, зосереджуючи команду на реальних потребах користувачів. Це підтверджують Nguyen-Duc та Abrahamsson, які зазначають, що стартапи, які намагаються впровадити повну версію продукту одразу, часто

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

зазнають невдач [4]. Двосторонній зворотний зв'язок забезпечується, коли обидві категорії користувачів (постачальники та споживачі) беруть участь у вдосконаленні продукту, що підвищує релевантність платформи та зменшує ризики невдачі [2].

Торгівля з двосторонньою структурою: сфера застосування MVP та цільові користувачі. Двостороння торговельна структура, або двосторонній ринок, – це платформа, яка об'єднує дві різні, але взаємозалежні групи користувачів, створюючи умови для їхньої взаємодії та обміну цінністю. У таких ринках цінність для однієї групи користувачів зростає з розширенням іншої групи, що спричиняє ефект непрямих мережевих зовнішніх ефектів [8]. Прикладами таких платформ є Amazon, Uber та Airbnb, які з'єднують продавців із покупцями, водіїв із пасажирами та орендодавців із орендарями відповідно [8]. Прикладами двосторонніх платформ є маркетплейси (Amazon та eBay) і платформи послуг (Uber). Маркетплейси – платформи, що надають простір для взаємодії між продавцями та покупцями, дозволяючи продавцям розміщувати свої товари, а покупцям – обирати та купувати їх [8]. Платформами послуг називають сервіси на кшталт Uber, які з'єднують замовників й виконавців, надаючи інфраструктуру для замовлення та надання певних видів послуг [9].

Ефективний обмін інформацією та логістикою є критично важливим для успішного функціонування двосторонніх платформ. Використання сучасних логістичних інформаційних платформ сприяє інтеграції бізнес-процесів, підвищенню прозорості та ефективності в ланцюгах постачання. Проєкт AEOLIX, наприклад, спрямований на створення екосистеми для інтеграції логістичних процесів через хмарні рішення [10].

MVP підхід може бути направлений на вирішення наступних проблем: початкова перевірка моделі взаємодії, побудова ядра майбутньої торгової платформи, швидка перевірка ціннісної пропозиції. Детально розглянемо кожен з них. Початкова перевірка моделі взаємодії: створення мінімально життєздатного продукту (MVP) дозволяє підприємцям протестувати основну концепцію платформи, залучаючи перших користувачів та оцінюючи їхню взаємодію. Це допомагає виявити потенційні проблеми та адаптувати модель під потреби ринку [11]. Побудова ядра майбутньої торгової платформи: MVP допомагає зосередитися на розробці ключових функцій, необхідних для запуску платформи, що забезпечує основу для подальшого розширення та додавання нових можливостей [11]. MVP дає змогу оцінити, наскільки запропоновані послуги відповідають очікуванням як постачальників, так і споживачів, що є важливим для досягнення балансу та успіху платформи [11]. Підприємці, які запускають нові платформи: MVP є ефективним інструментом для стартапів, оскільки дозволяє з мінімальними витратами перевірити життєздатність бізнес-ідеї та отримати зворотний зв'язок від перших користувачів [12]. Компанії, які прагнуть оптимізувати свої операції через впровадження цифрових рішень, можуть використовувати MVP для тестування та впровадження нових функціональностей, що сприяє підвищенню ефективності та зниженню витрат [13]. Розробники програмного забезпечення можуть використовувати MVP для швидкого створення та тестування нових продуктів, що відповідають потребам ринку та забезпечують конкурентні переваги [13]. Концепція мінімально життєздатного продукту є ключовим інструментом сучасної розробки програмного забезпечення, що дозволяє перевірити основні бізнес-гіпотези з мінімальними витратами. Як зазначають Duc & Abrahamsson, «MVP дозволяє командам вивчити ринок і поведінку користувачів за мінімально необхідної інвестиції в розробку» [14]. MVP визначається як початкова версія продукту, яка забезпечує користувача базовою, але завершеною ціннісною пропозицією. Lenarduzzi & Taibi зазначають: «An MVP is not a prototype, but a first functioning version of a product that includes only the essential features to test hypotheses» [15]. Ключовими характеристиками MVP є такі показники як: мінімальність, життєздатність, орієнтація на навчання. Розглянемо детальніше кожен елемент. Під мінімальністю розуміють, що продукт повинен мати лише ті функції, які безпосередньо реалізують цінність. Як зазначено у звіті Atlassian: «It should be just enough to provide value to users and gather validated learning about customers.» [16]

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Як зазначають у Agile Alliance, життєздатністю вважають «можливість навіть у мінімальному варіанті продукту бути стабільним та корисним для кінцевого споживача» [17]. У Ries зазначають, що орієнтація на навчання формулюється тим, що MVP створюється для підтвердження або спростування гіпотез, як частина підходу «build-measure-learn» [18]. MVP часто використовується для досягнення певних цілей. Серед них: перевірка ідеї на ранніх етапах, створення нових функцій чи продуктів без ризику повного розгортання у масштабних компаніях. [19], отримання швидкого зворотнього зв'язку. Як зазначає Nguyen-Duc et al.: «Startups utilize MVPs to minimize initial investments while identifying a scalable business model». [20] У високотехнологічних проєктах завдяки швидкому зворотному зв'язку MVP ідеально підходить для галузей із високим рівнем змін [21]. Багато джерел вважають, що застосування MVP має найбільшу цінність у стартапах, мобільних і веб-застосунках, а також у платформних рішеннях із двосторонніми ринками. В стартапах MVP в основному використовують як спосіб мінімізувати ризики та витрати. Codica пише: «Launching an MVP gives startups a safe path to discover market needs without overcommitting resources» [22]. Велечезну користь MVP приносить у мобільних і веб-застосунках, які потребують тестування функціональності на реальних користувачах [23].

Платформним рішенням з двосторонніми ринками MVP дозволяє швидко перевірити основну модель взаємодії між сторонами, перш ніж інвестувати в повномасштабну інфраструктуру.

Висновки. У межах проведеного дослідження було проаналізовано теоретичні засади та практичні підходи до розробки мінімально життєздатного продукту (MVP) у контексті підприємств торгівлі з двоспрямованою структурою. Застосування MVP у таких умовах виявилося надзвичайно ефективним інструментом для швидкого тестування гіпотез, побудови базової платформи та формування зворотного зв'язку від усіх категорій користувачів. Розгляд інкрементальної, спіральної, Agile та Customer Development моделей дозволив ідентифікувати методологічні рамки, що найкраще відповідають вимогам платформної торгівлі. Застосування цих підходів забезпечує не лише скорочення часу виходу продукту на ринок, а й підвищення його релевантності завдяки орієнтації на користувача. Особливу цінність MVP має для стартапів і компаній, що працюють у швидкозмінному середовищі та прагнуть до мінімізації витрат при максимальному навчанні. Результати дослідження свідчать про доцільність і необхідність впровадження MVP у торговельні підприємства, що працюють на двосторонніх ринках, оскільки такий підхід дозволяє ефективно управляти ризиками, масштабувати платформу на основі емпіричних даних і задовольняти динамічні потреби обох сторін взаємодії – постачальників і споживачів. У подальших дослідженнях доцільно зосередити увагу на емпіричному тестуванні зазначених моделей у конкретних секторах ринку та формалізації найкращих практик створення MVP з урахуванням галузевих особливостей.

Список використаних джерел

1. Sommerville, I., Sawyer, P. Requirements Engineering: A Good Practice Guide. – Chichester : John Wiley & Sons, 1997. – 432 p.

2. Blank, S. The Four Steps to the Epiphany: Successful Strategies for Products that Win / Steve Blank. – 2nd ed. – Pescadero, CA : K&S Ranch, 2013. – 370 p.

3. Boehm, B. A Spiral Model of Software Development and Enhancement / B. Boehm // ACM SIGSOFT Software Engineering Notes. – 1986. – Vol. 11, No. 4. – P. 14–24.

4. Nguyen-Duc, A., Seppänen, P., Abrahamsson, P. Minimum Viable Product or Multiple Facet Product? Empirical Study on Early-Stage Software Startups // Software Business. – 2017. – Vol. 304. – P. 118–132.

5. Codica. How to Build an MVP like Glovo [Електронний ресурс]. – 2022. – Режим доступу: https://www.codica.com/blog/mvp-like-glovo/ – Назва з екрана.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

6. Sharetribe. How Etsy Built Their MVP [Електронний ресурс]. – 2023. – Режим доступу: https://www.sharetribe.com/academy/etsy-mvp-case-study/ – Назва з екрана.
7. Lasoft. OLX MVP Launch Strategy [Електронний ресурс]. – 2023. – Режим доступу: https://lasoft.org/blog/olx-minimum-viable-product/ – Назва з екрана.
8. Investopedia. Two-Sided Market [Електронний ресурс]. – Режим доступу: https://www.investopedia.com/terms/t/two-sidedmarket.asp – Назва з екрана.
9. Living Economics. Platform Economics [Електронний ресурс]. – Режим доступу: https://livingeconomics.org/article.asp?docId=463 – Назва з екрана.
10. Allen, J., Allen, T. Logistics Information Exchange Platforms: Insights of the AEOLIX Project [Електронний ресурс] / J. Allen, T. Allen // Chalmers Research. – 2020. – Режим доступу: https://research.chalmers.se/en/publication/501502 – Назва з екрана.
11. Sharetribe. How to Build a Minimum Viable Platform [Електронний ресурс]. – 2023. – Режим доступу: https://www.sharetribe.com/academy/how-to-build-a-minimum-viable-platform/ – Назва з екрана.
12. Codica. MVP Benefits for Startups [Електронний ресурс]. – 2023. – Режим доступу: https://www.codica.com/blog/mvp-benefits/ – Назва з екрана.
13. Lasoft. MVP Development Benefits for Startups and Companies [Електронний ресурс]. – 2023. – Режим доступу: https://lasoft.org/blog/mvp-development-benefits-for-startups-and-companies/ – Назва з екрана.
14. Nguyen-Duc A., Abrahamsson P. Minimum Viable Product or Multiple Facet Product? Empirical Study on Early-Stage Software Startups // *Software Business. Lecture Notes in Business Information Processing*. – 2017. – Vol. 304. – P. 118–132.
15. Lenarduzzi, V., & Taibi, D. (2016). MVP Explained: A Systematic Mapping Study on the Definitions of Minimal Viable Product. Retrieved from
16. Atlassian. (2023). Minimum Viable Product (MVP). Retrieved from https://www.atlassian.com/agile/product-management/minimum-viable-product
17. Agile Alliance. (2022). Glossary: MVP. Retrieved from https://www.agilealliance.org/glossary/mvp/
18. Ries, E. (2011). The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Business.
19. Lasoft. (2023). Benefits of MVP for Companies. Retrieved from https://lasoft.org/blog/mvp-development-benefits-for-startups-and-companies/
20. Nguyen-Duc, A., Seppänen, P., & Abrahamsson, P. (2017). Minimum Viable Product or Multiple Facet Product?. In Software Business (pp. 118–132). Springer.
21. Designveloper. (2023). How MVP Fits in Agile Development. Retrieved from https://www.designveloper.com/guide/mvp-in-agile/
22. Codica. (2023). Why Startups Need MVP. Retrieved from https://www.codica.com/blog/mvp-benefits/
23. Decode. (2023). MVP Development for Startups. Retrieved from https://decode.agency/article/mvp-development-for-startups/

Роботу виконано під науковим керівництвом PhD, доцента
ДЕСЯТКО А. М.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

СТВОРЕННЯ ЧАТ-БОТУ НА ОСНОВІ НЕЙРОМЕРЕЖІ

ГЛУЩЕНКО Д. С., 2 курс 2м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглянуто процедуру створення чат-бота для вирішення типових завдань технічної підтримки на основі штучної нейронної мережі. Запропоновано поетапну методику реалізації архітектури, навчання моделі та інтеграції її в інтерфейс користувача. Акцент зроблено на обґрунтуванні вибору методів глибинного навчання для обробки природної мови. Наведено практичний приклад застосування.

The article discusses the procedure for creating a chatbot for solving basic technical support tasks using artificial neural networks. A step-by-step implementation of the architecture, model training, and integration into the user interface is proposed. Emphasis is placed on the justification of deep learning methods for natural language processing. A practical application example is provided.

Актуальність. Сфера людської діяльності торгівля, одна з перших почала впроваджувати в реальну діяльність інформаційні технології. Можна вважати, що саме в ній з'явилася «автоматизована система обліку»: Податкова інспекція тоді не контролювала касову стрічку, облік був потрібен самому власнику тоговельного підприємства.

З часом, розвивалася торгівля, розвивалися інформаційні технології. Звичайно, залишилися ті, кому і сьогодні досить касових апаратів, які стали «розумніші», але суть процесу змінилася мало. Збільшилися масштаби торгівлі, асортимент, потік покупців – все це на порядок більше, ніж століття тому, і управляти торговельним підприємством без автоматизації стало вельми складно, а іноді неможливо.

Роздрібна торгівля – це найвужче місце в торгівля в цілому. Жоден магазин або мінімаркетів не в змозі викласти на прилавки весь асортимент товарів, які постачаються або виробляються. Причина – фізичне обмеження розмірів торгової площі і полиць. Завдання ефективного ритейлера – виставити на полиці саме той товар, який принесе йому максимальний прибуток.

Підприємства роздрібної торгівлі, завдяки своєму стратегічному положенню в ланцюжку постачань, диктують правила гри для всіх учасників ринку. Так, наприклад, у магазинів або мінімаркетів не прийнято розраховуватись відразу за поставлений товар. Відстрочка оплати мінімум на 40 днів є практично в кожному договорі на постачання в роздрібну мережу. Зрозуміло, є і спеціальні служби, що відповідають за підписання таких договорів та за контроль взаєморозрахунків за ними.

Метою статті є дослідження особливостей використання інформаційно-управляючих систем на підприємствах роздрібної торгівлі з метою підвищення ефективності їх функціонування.

Об'єктом дослідження є розробка інформаційно-управляючої системи підприємства роздрібної торгівлі (мінімаркету).

Предмет дослідження – інформаційно-управляюча система.

Виклад основного матеріалу. Розробка чат-бота на основі нейронної мережі є складним завданням, що передбачає низку етапів: від аналізу вимог до інтеграції у реальне середовище. У межах даного дослідження розроблено поетапну процедуру створення чат-бота для вирішення типових завдань технічної підтримки користувачів.

Першим і одним із ключових етапів у процесі створення чат-бота є чітке формулювання його цільового призначення, визначення конкретних задач, які він повинен виконувати, та встановлення меж його функціональності. У рамках системи технічної підтримки основна

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

мета чат-бота полягає в автоматизації стандартних інформаційних і консультативних запитів користувачів, що повторюються найчастіше. Це дозволяє значно розвантажити операторів підтримки та скоротити час відповіді на типові питання.

До основних функцій чат-бота у такому контексті можна віднести:

– Надання відповідей на часто поставлені запитання (FAQ): бот повинен оперативно відповідати на заздалегідь визначені типові запити, наприклад, щодо налаштувань облікового запису, доступу до ресурсів, оновлення програмного забезпечення тощо. Такий підхід дозволяє зменшити кількість звернень до операторів на 30–50 %.

– Консультування з базових процедур: користувачі часто звертаються з однаковими проблемами, такими як відновлення паролю, реєстрація нового облікового запису, підключення до Wi-Fi, активація ліцензії тощо. Чат-бот має надавати чіткі покрокові інструкції у зручному форматі, включно з можливістю надсилання посилань на відповідні сторінки або завантаження файлів.

– Ідентифікація складних випадків та перенаправлення до живого оператора: бот повинен розпізнавати ситуації, які виходять за межі його компетенції (наприклад, специфічні технічні збої, персоніфіковані проблеми, помилки в базі даних), і вчасно спрямовувати користувача до відповідного фахівця або створювати заявку для служби підтримки.

– Підтримка природного діалогу в зручному форматі: чат-бот повинен забезпечувати плавне ведення діалогу, підтримувати логічну послідовність запитань і відповідей, пам’ятати контекст розмови в межах однієї сесії та взаємодіяти у ввічливій і доступній формі, що підвищує задоволеність користувачів.

На основі аналізу зазначених функціональних вимог формується загальна логіка роботи чат-бота, розробляється сценарій діалогу та визначаються ключові точки взаємодії, що надалі впливатимуть на вибір архітектури системи та методів реалізації.

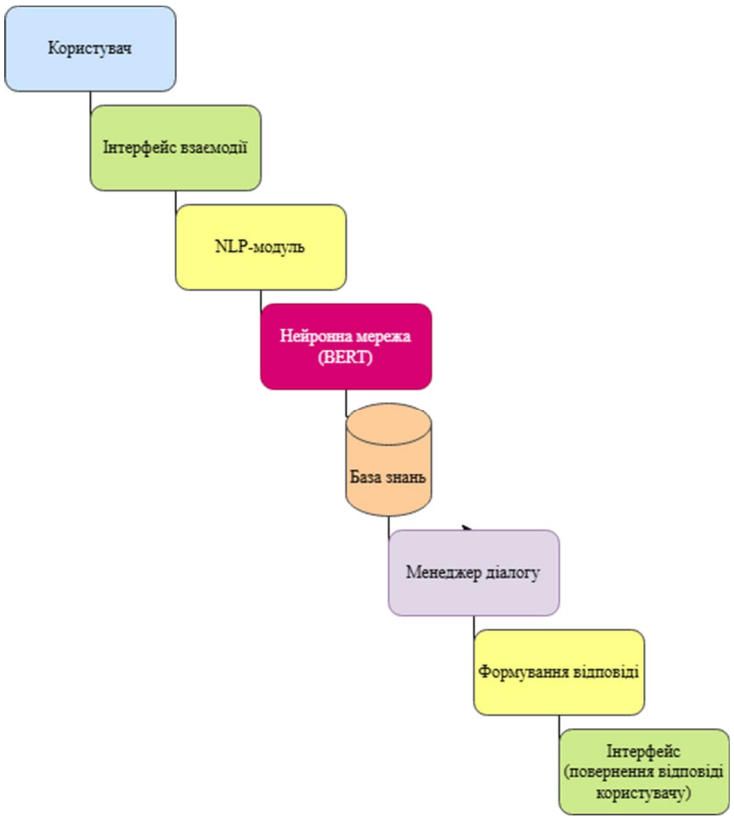


Рис. 1. Загальна архітектура чат-бота на основі нейронної мережі

Джерело: розробка автора

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

На рисунку 1 представлено схематичну модель загальної архітектури чат-бота на основі нейронної мережі у вигляді вертикальної блок-схеми. Діаграма відображає логічну послідовність обробки текстового запиту користувача – від моменту введення до отримання зворотного повідомлення. Кожен блок позначає окремий функціональний модуль системи.

У верхній частині схеми розташований блок «Користувач», який символізує джерело запиту. Далі інформація надходить до «Інтерфейсу взаємодії», який забезпечує первинний прийом запиту через вебінтерфейс або месенджер.

Наступним етапом є «NLP-модуль», який виконує обробку природної мови, включаючи токенізацію, лематизацію та підготовку даних до подачі в модель. Центральне місце у структурі займає блок «Нейронна мережа (BERT)», що відповідає за глибинний аналіз запиту, розуміння наміру та контексту.

Після обробки запиту модель звертається до «Бази знань» – джерела попередньо сформованих відповідей, інструкцій або фактологічної інформації. Отримана інформація передається до «Менеджера діалогу», який координує логіку спілкування, визначає, чи потрібно надати відповідь, чи передати запит оператору.

На наступному етапі здійснюється «Формування відповіді», тобто генерація текстового повідомлення у зрозумілій для користувача формі. Останній блок – «Інтерфейс (повернення відповіді користувачу)», через який сформована відповідь надсилається назад до користувача.

Кольорове кодування блоків підсилює візуальне сприйняття та відображає функціональний розподіл компонентів системи. Таким чином, схема відображає послідовність та взаємозв’язок усіх основних етапів роботи чат-бота на основі нейромережі.

Рис. 2. Процеси ефективної взаємодії з користувачем

Джерело: розробка автра

Ефективна робота чат-бота в значній мірі залежить від якості обробки введеного користувачем тексту. На цьому етапі виконується серія процедур, що входять до складу системи обробки природної мови (Natural Language Processing, NLP). Метою NLP-підсистеми є перетворення неструктурованого тексту у форму, яку може опрацювати нейронна мережа. Це забезпечує глибше розуміння наміру користувача, правильну інтерпретацію запиту та формування релевантної відповіді. Ключові етапи NLP-підсистеми наведено нижче.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Першим кроком обробки є токенізація – процес розбиття вхідного тексту на менші елементи, які називаються токенами. Зазвичай це окремі слова, але можуть бути й частини слів або символи залежно від типу моделі. Наприклад, фраза «Я не можу увійти в систему» буде перетворена на послідовність токенів: «Я», «не», «можу», «увійти», «в», «систему». У більш складних моделях, таких як BERT, застосовується subword tokenization – розбиття слів на морфеми або частини слів для забезпечення більш гнучкої генералізації. Це особливо корисно у випадках з неологізмами або термінами, відсутніми в словнику моделі.

Після токенізації виконується лематизація – процес приведення слів до їх початкової (словникової) форми. Наприклад, слова «входжу», «увійшов», «входити» будуть зведені до однієї леми – «входити». Цей процес дозволяє нейромережі розуміти, що всі ці форми мають однакове смислове значення. Лематизація знижує варіативність словника та покращує узагальнення даних при навчанні або прогнозуванні.

Наступним кроком є фільтрація так званих стоп-слів – слів, що не несуть ключового семантичного навантаження в контексті запиту. До таких слів зазвичай належать службові частини мови: прийменники, сполучники, частки (наприклад, «у», «і», «що», «аби»). Хоча в деяких випадках (особливо в сучасних трансформерних моделях) ці слова не видаляються, їх все ж можуть виключати під час попередньої обробки, аби зменшити шум у даних, особливо при роботі зі статистичними моделями чи меншими корпусами.

Останнім кроком є кодування тексту – перетворення його з текстового формату у числовий вигляд, який є зрозумілим для нейронної мережі. Для цього використовуються техніки векторизації. Найпростішим методом є Bag-of-Words, однак сучасні нейромережі працюють із вбудовуваннями слів (Word Embeddings), які зберігають контекстуальні залежності. Одним з найефективніших рішень у цьому напрямі є Tokenizer BERT, який кодує слова з урахуванням їх оточення в реченні. Наприклад, слово «банк» у фразах «піти в банк» і «берег річки – це природний банк» буде представлено різними векторами, що дозволяє моделі краще розуміти контекст.

Усі етапи є підготовчими і виконуються до того, як запит буде передано до основної нейромережі для подальшої обробки. Якість реалізації NLP-підсистеми має вирішальне значення, адже навіть найпотужніша модель не зможе правильно інтерпретувати запит, якщо вхідні дані є нечистими, неструктурованими або некоректно представленими у числовому вигляді.

Таблиця 1

Етапи NLP на прикладі запиту

№	Етап	Опис дії	Результат
1	Вхідний текст	Текстовий запит від користувача	«Я не можу увійти в систему»
2	Токенізація	Розбиття тексту на токени	[«Я», «не», «можу», «увійти», «в», «систему»]
3	Лематизація	Приведення слів до початкової форми	[«я», «не», «могти», «увійти», «в», «система»]
4	Видалення стоп-слів	Видалення другорядних слів	[«могти», «увійти», «система»]
5	Кодування (векторизація)	Перетворення у числовий формат для моделі	[1045, 2025, 2064, 2424, 1999, 2298] (ID токенів)

Джерело: розробка автора

У процесі створення інтелектуального чат-бота особливу увагу слід приділити вибору відповідної архітектури нейронної мережі, яка б забезпечувала точне розпізнавання запитів користувачів та генерування змістовних відповідей. З-поміж сучасних моделей обробки

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

природної мови найбільш ефективною для вирішення подібних завдань є BERT (Bidirectional Encoder Representations from Transformers).

BERT – це трансформерна модель, запропонована дослідниками компанії Google у 2018 році [3], яка стала проривом у сфері NLP. Її головною відмінністю є двонаправлене навчання, що дозволяє моделі враховувати контекст не лише зліва направо (як у традиційних моделей на кшталт LSTM), а й одночасно з обох боків слова. Це означає, що при аналізі речення модель розглядає всі слова навколо кожного конкретного слова, що значно покращує розуміння змісту, особливо в мовах із вільним порядком слів, як українська.

Основні особливості архітектури BERT:

1. Двонаправлена трансформерна структура BERT базується на трансформерній архітектурі, представлена в джерелі[4]. Ключовим елементом є механізм уваги (self-attention), який дозволяє моделі зосереджуватись на найбільш важливих словах у реченні. Наприклад, у запиті «Я не можу увійти в систему», модель розпізнає зв'язок між словами «увійти» та «систему», навіть якщо між ними стоїть кілька інших слів. Це дозволяє моделі точніше інтерпретувати наміри користувача.
2. Попереднє навчання на великому корпусі текстів BERT проходить початкове навчання на величезних текстових наборах, таких як Wikipedia і BookCorpus, що забезпечує їй базові знання про структуру мови, граматику, логіку побудови речень. Завдяки цьому модель уже «розуміє» мову на загальному рівні ще до того, як її почнуть навчати на спеціалізованих задачах.
3. Здатність до донавчання (fine-tuning) під конкретне завдання. Найбільша перевага BERT полягає у її здатності легко адаптуватися до нових задач без потреби створювати нову модель з нуля. Після попереднього навчання, BERT можна «доучити» на конкретному датасеті – у нашому випадку, на корпусі запитів користувачів до технічної підтримки. Це дозволяє моделі враховувати специфіку термінології, стилю запитань та очікуваних відповідей у заданій предметній області.

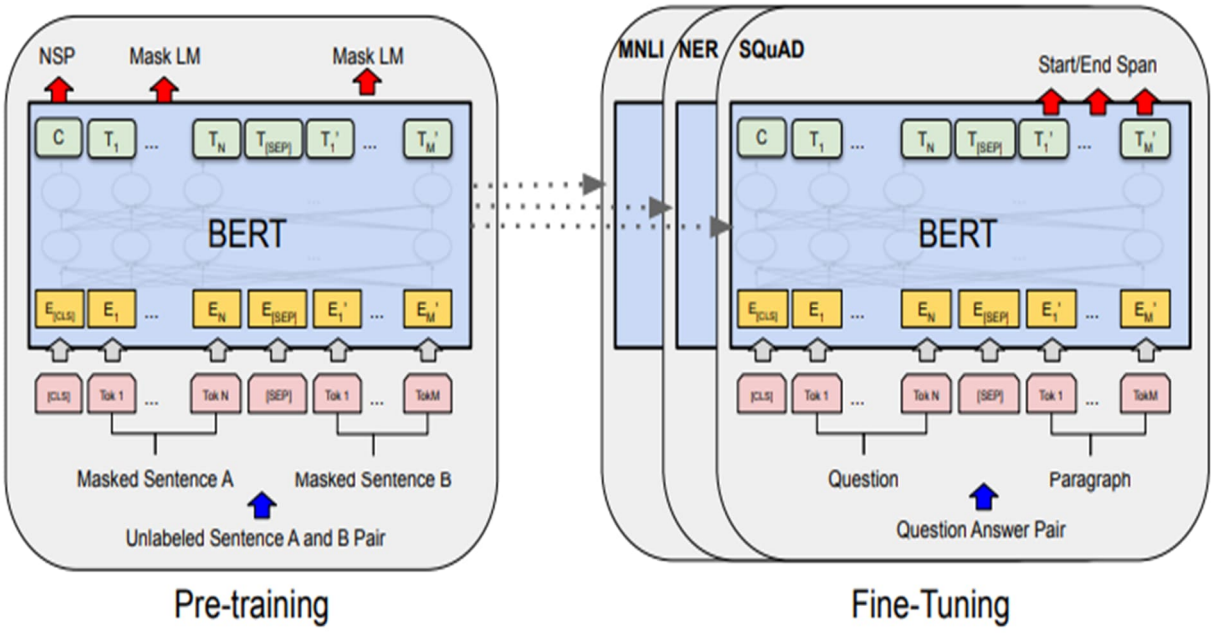


Рис. 3. Загальна процедура попереднього навчання та донавчання моделі BERT

Джерело: [3]

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Архітектура BERT: короткий опис компонентів

Типова модель BERT складається з 12 шарів (у базовій версії BERT-Base), кожен з яких містить:

- Self-Attention Layer – визначає, які слова в реченні важливі одне для одного;
- Feedforward Neural Network – обробляє контекстуальні представлення;
- Normalization & Residual Connections – забезпечують стабільність і швидке навчання.

На вході до моделі подається текст, що попередньо перетворений у токени, з додаванням спеціальних токенів [CLS] (початок речення) та [SEP] (роздільник).

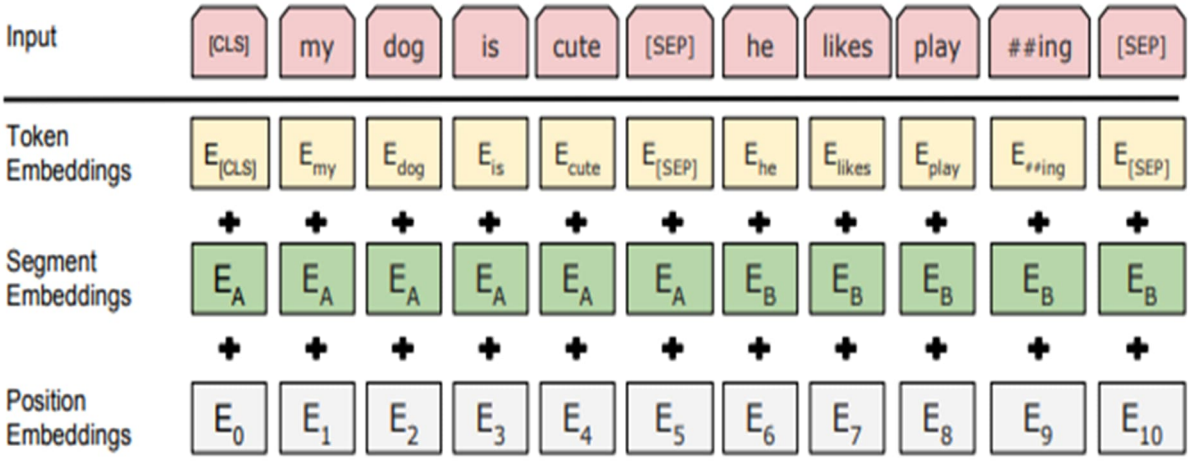


Рис. 4. Представлення вхідних даних у моделі BERT

Джерело: [3]

Вихід моделі – це набір векторів, що представляють кожне слово в контексті, а також узагальнене представлення всього запиту (вектор [CLS]).

Модель BERT використовується для класифікації наміру користувача. Наприклад, запит «Не можу відновити пароль» буде проаналізований, і модель класифікує його як намір типу «відновлення облікових даних». Це дозволяє менеджеру діалогу підібрати релевантну відповідь або сценарій.

На схемі архітектури моделі BERT можна побачити, як вхідний запит розбивається на токени, перетворюється в вектори і проходить послідовну обробку через шари трансформера. У фіналі модель видає представлення запиту, яке може бути використано для класифікації, генерації або витягування інформації.

Використання моделі BERT у системі чат-бота дозволяє значно покращити якість розуміння тексту, обробку запитів у природній мові та забезпечити високий рівень точності відповідей. Це робить її оптимальним вибором для інтелектуальних систем технічної підтримки, орієнтованих на живу комунікацію з користувачем.

Список використаних джерел

1. Zhang K., Song D., Zhang C. Deep Learning for Chatbots. – IEEE Transactions on Neural Networks, 2020. – Vol. 31(6). – P. 1–15.

2. Khoshnoud S., Zhang X. BERT-Based Chatbots for Technical Support Systems // Proceedings of the 2021 International Conference on AI. – 2021. – P. 202–210.

3. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. – arXiv:1810.04805, 2018.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

4. Vaswani A. et al. Attention is All You Need. – Advances in Neural Information Processing Systems. – 2017. – Vol. 30.

Роботу виконано під науковим керівництвом канд. екон. наук, доцента
ПАЛАГУТИ К. О.

3D КОМП'ЮТЕРНА ГРА З ВИКОРИСТАННЯМ ІГРОВОГО РУШІЯ UNITY

**ГОНЧАРЕНКО О. В., 2 курс 2м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»**

У статті розглядаються можливості та переваги ігрового рушія Unity у розробці ігор та інших програмних продуктів. Описано основні компоненти програмного забезпечення Unity, використання C# та принципів об'єктно-орієнтованого програмування. Наведено приклади застосування Unity у сучасних програмних рішеннях.

The article discusses the capabilities and advantages of the Unity game engine in the development of games and other software products. The main components of the Unity software, the use of C# and the principles of object-oriented programming are described. Examples of the use of Unity in modern software solutions are given.

Актуальність. Розробка комп'ютерних ігор є однією з найдинамічніших і прибуткових галузей у сфері інформаційних технологій. Вона охоплює широкий спектр дисциплін, включаючи програмування, комп'ютерну графіку, дизайн рівнів, фізику, штучний інтелект і звукорежисуру. Сучасні ігри – це складні програмні продукти, які використовують передові технології, такі як машинне навчання, хмарні обчислення та VR/AR (віртуальна та доповнена реальність). Індустрія ігор має значний вплив на розвиток апаратного забезпечення, сприяючи появі потужніших відеокарт, багатоядерних процесорів і вдосконалених алгоритмів рендерингу.

Метою статті є аналіз особливостей використання ігрового рушія Unity для створення ігор.

Об'єктом дослідження є процес розробки комп'ютерної гри.

Предмет дослідження – технології Unity та їх застосування у сучасній розробці ігрових програмних продуктів.

Аналіз попередніх досліджень. Дослідженню питань розробки ігор у Unity присвячено багато праць. Серед них виділяються роботи таких авторів, як Д. Бродкін [1], Д. Хокінг [2].

Виклад основного матеріалу. Варто відмітити, що важливість ігрової індустрії для IT-ринку обумовлюється рядом чинників, які можуть бути аргументовані наступними тенденціями:

– Велика частка ринку. Ігрова індустрія є однією з найприбутковіших сфер у IT. За даними аналітичних агентств, її річний дохід перевищує доходи кінематографу та музичної індустрії разом узятих. У 2023 році глобальний ринок відеоігор оцінювався в понад 180 мільярдів доларів, і цей показник продовжує зростати [3].

– Інновації та технологічний розвиток. Багато технологій, які використовуються у інших сферах IT, вперше були розроблені або вдосконалені в ігровій індустрії. Наприклад, алгоритми трасування променів у графіці, системи обробки великих даних у багатокористувацьких онлайн-іграх (MMO), а також технології штучного інтелекту, що застосовуються в NPC [4].

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

– Попит на фахівців. Розробка ігор потребує висококваліфікованих фахівців у різних сферах: програмістів, дизайнерів, художників, тестувальників, сценаристів, маркетологів та аналітиків. Це створює високий попит на спеціалістів, що працюють із ігровими рушіям, наприклад, Unity або Unreal Engine [5].

– Глобальність аудиторії. Сучасні ігри охоплюють різні платформи – персональні комп'ютери, мобільні пристрої, консолі та хмарні сервіси. Це розширює ринок і дозволяє розробникам створювати проєкти для глобальної аудиторії, незалежно від апаратних обмежень користувачів [6].

Unity - це кросплатформений ігровий рушій, розроблений Unity Technologies, вперше анонсований і випущений у червні 2005 року на Всесвітній конференції розробників Apple як ігровий рушій для Mac OS X. Відтоді механізм поступово розширювався, щоб підтримувати різноманітні платформи настільних комп'ютерів, мобільних пристроїв, консолей, доповненої реальності та віртуальної реальності. Він особливо популярний для розробки мобільних ігор для iOS і Android, вважається простим у використанні для розробників-початківців і популярний для розробки незалежних ігор. [7]

Рушій можна використовувати для створення тривимірних (3D) і двовимірних (2D) ігор, а також для інтерактивного моделювання. Рушій був прийнятий галузями за межами відеоігор, включаючи кіно, автомобілебудування, архітектуру, інженерію, будівництво та Збройні сили США [7].

Unity є одним із найпопулярніших рушіїв для розробки комп'ютерних ігор завдяки своїй гнучкості, кросплатформеності та широкому набору інструментів. Цей рушій підходить як для початківців, так і для професійних розробників, дозволяючи створювати як прості 2D-проєкти, так і складні 3D-ігри з передовими технологіями графіки та фізики.

Було виділено основні переваги рушія Unity [7] (табл. 1):

Таблиця 1

Ключові переваги рушія Unity у процесі створення 3D зображень

Перевага	Характристика
Кросплатформеність та мультиплат-формовий експорт	Unity підтримує понад 25 платформ, включаючи Windows, macOS, Linux, iOS, Android, PlayStation, Xbox, Nintendo Switch, а також WebGL і VR/AR-пристрої. Це дозволяє розробникам створювати один проєкт і легко експортувати його для різних платформ, мінімізуючи витрати на адаптацію. Як приклад використання можна зазначити процес створення мобільної гри для Android, а потім без значних змін експортувати її на iOS або ПК [6]
Програмування на C# та гнучкість рушія	Unity використовує мову програмування C#, що є потужною, продуктивною та відносно легкою для вивчення. Головною перевагою є використання Mono та .NET Core, що дозволяє ефективно обробляти складну логіку гри, а також інтегрувати сторонні API та сервіси [7]
Вбудовані інструменти фізики та анімації	Unity має вбудований фізичний рушій PhysX від NVIDIA, що дозволяє реалістично моделювати зіткнення, гравітацію та поведінку об'єктів. Крім того, є система анімації Mecanim, яка підтримує Inverse Kinematics (IK) та Blend Trees для створення плавних рухів персонажів. Використовується для реалістичної симуляції автомобільної фізики у гоночних іграх та анімацій персонажів з автоматичним переходом між ходьбою, бігом і стрибками [7]
Гнучка система управління ресурсами та Asset Store	Unity має власний магазин ресурсів Unity Asset Store, де можна знайти тисячі безкоштовних і платних 3D-моделей, текстур, звукових ефектів і навіть готових систем керування персонажами. Це дозволяє значно скоротити час розробки
Підтримка VR та AR	Unity активно використовується для створення ігор у віртуальній (VR) та доповненій (AR) реальності завдяки інтеграції з OpenXR, Oculus SDK та ARKit/ARCore [7]

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Перевага	Характристика
Мережева взаємодія та мультиплеєр	Unity підтримує кілька рішень для мережевої гри. Mirror – для реалізації хостованих серверів. Photon – для хмарних мультиплеєрних сервісів та інші. Це використовується у онлайн-шутерах або кооперативних RPG іграх
Гнучкість у графіці	Unity пропонує два варіанти рендеру, Universal Render Pipeline (URP) та High Definition Render Pipeline (HDRP), що дозволяє розробникам оптимізувати графіку відповідно до потреб гри. URP – для мобільних та середньої складності ПК-ігор. HDRP – для фотореалістичної графіки на високопродуктивних пристроях

Джерело: розроблено автором на основі [6, 7]

Як вже згадувалося раніше, Unity використовує мову програмування C#. Вона забезпечує високу продуктивність, простоту написання коду та інтеграцію з широким набором інструментів .NET. Завдяки своїй об’єктно-орієнтованій природі та підтримці сучасних технологій, C# є ідеальним вибором для створення ігрових механік, управління об’єктами та роботи з мережею в Unity [7].

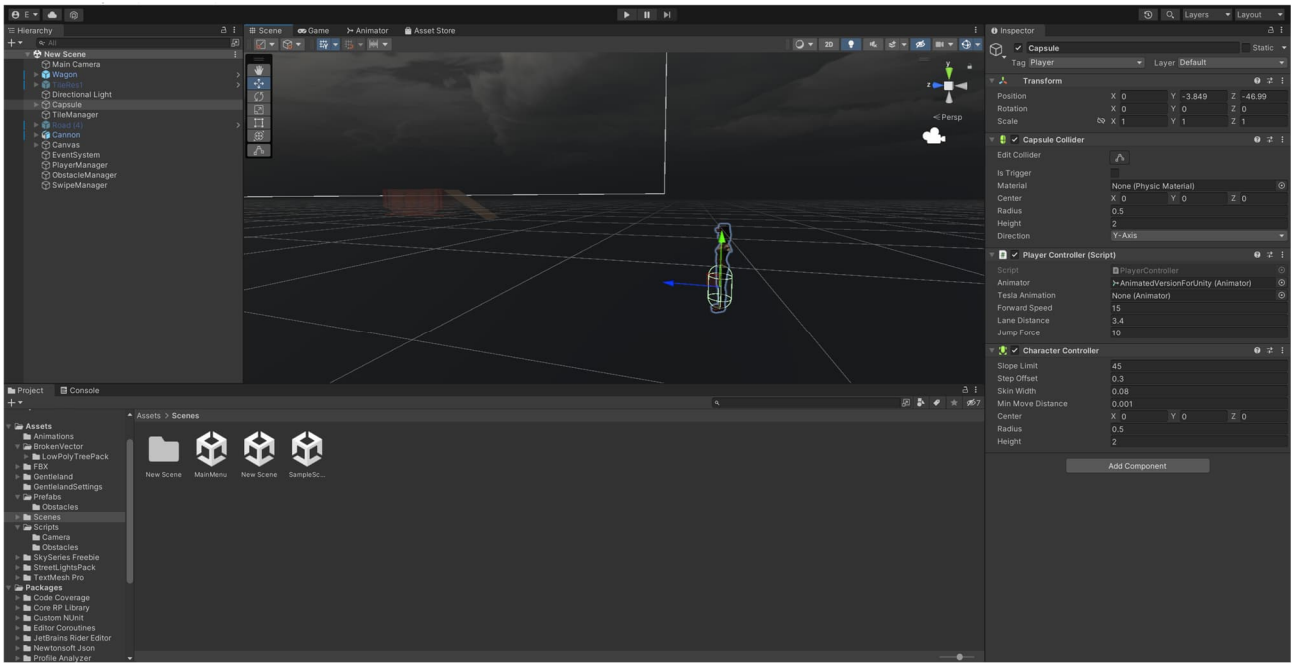


Рис. 1. Інтерфейс Unity

Джерело: розроблено автором в середовищі Unity 2021.3.16f1 (знімок з екрану)

Історично Unity підтримував JavaScript-подібну мову UnityScript та Boo, але з часом єдиним стандартом для написання коду став C#, оскільки він простий у вивченні порівняно з C++, високопродуктивний та підтримує асинхронні операції, що є важливим для мережевих ігор та обробки великих обсягів даних.

Unity використовує систему MonoBehaviour, яка дозволяє кожному ігровому об’єкту мати власний скрипт, що визначає його поведінку [7].

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

```
using UnityEngine;

public class PlayerController : MonoBehaviour
{
    public float speed = 5f;

    void Update()
    {
        float moveX = Input.GetAxis("Horizontal") * speed * Time.deltaTime;
        float moveY = Input.GetAxis("Vertical") * speed * Time.deltaTime;
        transform.Translate(moveX, 0, moveY);
    }
}
```

Рис.2. Скрипт для переміщення гравця

Джерело: розроблено автором в середовищі Unity 2021.3.16f1 (знімок з екрану)

Окремої уваги заслуговує об'єктно–орієнтована парадигма у Unity.

Вона є фундаментальною парадигмою при розробці ігор, що забезпечує структурований підхід до організації коду та керування ним. В Unity ігрові об'єкти представлені як екземпляри класів. Кожен клас визначає властивості, змінні та методи. Наприклад, клас «Player» може визначати в собі такі властивості, як «health» та «speed», а також може мати методи «Move()» та «Attack()».

Було виділено основні принципи об'єктно-орієнтованого програмування у Unity [8]:

– Успадкування: дозволяє класам успадковувати властивості та поведінку від батьківських класів. Це корисно для створення ієрархічних зв'язків між класами. В Unity успадкування зазвичай використовується для створення спеціалізованих класів, які розширюють функціонал існуючих. Наприклад, клас «Enemy» може успадковувати клас «Character», успадковуючи всі загальні властивості цього класу.

– Інкапсуляція: передбачає об'єднання даних та методів в один блок (клас) і обмеження доступу до певних компонентів. В Unity інкапсуляція допомагає підтримувати організацію коду та запобігає несанкціонованому доступу до конфіденційних даних. Наприклад, інкапсуляція змінних здоров'я в клас «Player» гарантує, що лише певні методи можуть змінювати здоров'я гравця.

– Поліморфізм: дозволяє розглядати об'єкти різних класів як об'єкти спільного суперкласу. Це дозволяє коду бути більш гнучким і адаптованим до різних типів об'єктів. В Unity поліморфізм часто досягається через інтерфейси та абстрактні класи.

– Абстракція: передбачає визначення основних характеристик об'єкта, приховуючи непотрібні деталі. В Unity абстракція допомагає спростити складні системи, зосереджуючись на концепціях і взаємодіях високого рівня.

Хоча Unity є одним із найпопулярніших ігрових рушіїв, на ринку існують й інші потужні конкуренти, кожен із яких має свої унікальні особливості. Основним конкурентом Unity у сфері ігрових рушіїв є Unreal Engine (табл. 2).

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Unreal Engine відомий своєю високою якістю графіки, завдяки рушію Nanite та освітленню Lumen. Іншою відмінністю між цими двома рушіями це мови програмування. На відміну від Unity, Unreal Engine використовує мову програмування C++, що робить розробку складнішою, але забезпечує вищу продуктивність. Unreal Engine широко використовується у AAA-іграх, VR-проєктах та кіновиробництві. Його головний недолік – складніший вхідний поріг у порівнянні з Unity.

Таблиця 1

Порівняльна характеристика Unity та Unreal Engine

Категорія	Unity	Unreal Engine
2D-ігри	Кращий вибір завдяки легкості	Менше оптимізований для 2D
Мобільні ігри	Оптимізований для мобільних пристроїв	Вимогливий до ресурсів
VR/AR	Простий у розробці, добре інтегрований	Вища якість графіки
AAA-ігри	Менше використовується у високобюджетних проєктах	Основний рушій для AAA-ігор
Кінематограф	Менше інструментів для відеовиробництва	Використовується в кіноіндустрії
Інді-ігри	Простий та доступний для початківців	Важче освоїти

Джерело: розроблено автором на основі [5]

Іншою відмінністю між Unity та Unreal Engine є підхід до розробки мультиплеєрних ігор, вбудовані рішення Unity та інші зовнішні SDK добре підходять для реалізації невеликих кооперативних ігор, в той час як Unreal Engine має вбудовану реплікацію мережевого коду, що полегшує створення мультиплеєрних ігор.

Іншим конкурентом Unity є ігровий рушій Godot Engine [9]. Це безкоштовний рушій з відкритим вихідним кодом. Його основні переваги – легкість, відсутність роялті та підтримка як C#, так і GDScript (спрощена мова для швидкого створення прототипів). Godot підходить для інді-розробників, але поступається Unity в аспектах 3D-графіки та кількості доступних ресурсів.

CryEngine – рушій від Crytek, відомий завдяки фотореалістичній графіці у таких іграх, як *Crysis* та *Hunt: Showdown [10]*. CryEngine має потужні інструменти для рендерингу, але складний у використанні та має слабку підтримку спільноти.

Висновки. Unity є одним із найпопулярніших ігрових рушіїв завдяки своїй кросплатформеності, гнучкості та потужним інструментам для розробки. Він дозволяє створювати ігри для широкого спектра платформ, включаючи ПК, мобільні пристрої, консолі та VR/AR. Використання C# у Unity забезпечує простоту написання коду, високу продуктивність та інтеграцію з сучасними технологіями, такими як штучний інтелект та мережеві сервіси. Завдяки Unity Asset Store розробники отримують доступ до тисяч готових ресурсів, що значно скорочує час і витрати на створення ігор. Вбудовані фізичні та анімаційні системи, такі як PhysX і Mecanim, дозволяють створювати реалістичну фізику та плавні анімації персонажів. Unity також підтримує сучасні графічні технології через URP і HDRP, що дає можливість оптимізувати якість графіки відповідно до вимог проєкту. Хоча Unity має сильних конкурентів, таких як Unreal Engine, Godot та CryEngine, він залишається найкращим вибором для багатьох розробників завдяки простоті у використанні та великій спільноті.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Окрім створення ігор, Unity знаходить застосування в архітектурі, кінематографі та симуляціях, що робить його універсальним інструментом у сфері 3D-розробки.

Список використаних джерел

1. Joe Hocking. Unity in Action, Third Edition: Multiplatform game development in C#. 2022.
2. Jon Manning, Tim Nurgent. Unity Development Cookbook: Real-Time Solutions from Game Development to AI 2nd Edition. 2023.
3. $184 мільярди доходу і Starfield як найбільш обговорювана гра: вийшла глобальна статистика ігрової індустрії за 2023 рік. DOU. 2024. URL: https://gamedev.dou.ua/news/gaming-industry-in-numbers-2023/
4. Топ-5 ігрових відеокарт з підтримкою трасування променів: максимальна якість графіки. Vse.ua. URL: https://vse.ua/info/top-5-igrovyh-videokart-s-podderzhkoy-trassirovki-luchey-maksimalnoe-kachestvo-grafiki-1753/
5. Unity проти Unreal Engine - який рушій обрати для гри та чому. Зважуємо всі за та проти з розробниками. DOU. 2024. URL: https://gamedev.dou.ua/articles/unity-or-unreal-engine/
6. Multiplatform tips and best practices. Go Create. URL: https://unity.com/solutions/ multiplatform
7. Unity (Game Engine). Вікіпедія. URL: https://en.wikipedia.org/wiki/ Unity_ (game_engine)
8. Об’єктно-орієнтоване програмування в Unity. URL: https://medium.com/ @ssmo-re101/object-oriented-programming-oop-in-unity-game-development-452486d1e4c5
9. Godot. Вікіпедія. : URL: https://uk.wikipedia.org/wiki/Godot
10. CryEngine. Вікіпедія. URL: https://uk.wikipedia.org/wiki/CryEngine_(серія_рушіїв)

Роботу виконано під науковим керівництвом канд. екон. наук, старшого викладача
БУЛЬБИ С. А.

УПРАВЛІННЯ АСИНХРОННИМИ ЗАДАЧАМИ: МУЛЬТИПОТОКОВЕ ТА МУЛЬТИПРОЦЕСОРНЕ ПРОГРАМУВАННЯ У СУЧАСНИХ МОВАХ

ГРЕЧКО А. В., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглянуто ключові принципи управління асинхронними задачами в контексті мультипотокового та мультипроцесорного програмування. Проаналізовано особливості реалізації конкурентних обчислень у сучасних мовах програмування, зокрема в Python. Окрему увагу приділено порівнянню підходів до виконання паралельних задач, їх перевагам і обмеженням, а також практичним аспектам ефективного використання відповідних технологій у програмній розробці.

The article examines the key principles of managing asynchronous tasks in the context of multithreading and multiprocessing programming. The features of implementing concurrent computations in modern programming languages, particularly Python, are analyzed. Special attention is given to comparing approaches to parallel task execution, their advantages and limitations, as well as practical aspects of effectively utilizing these technologies in software development.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Актуальність. У сучасному програмуванні ефективне управління асинхронними задачами є важливим фактором підвищення продуктивності програмних систем. З розвитком хмарних технологій, обчислювальних кластерів та масштабованих серверних архітектур зростає необхідність у технологіях, що дозволяють одночасно виконувати велику кількість задач без значних затримок. Асинхронне програмування, мультипотоковість та мультипроцесорність є ключовими підходами до вирішення цих завдань.

Традиційні однопотокові програми часто не здатні ефективно використовувати ресурси багатоядерних процесорів, що призводить до втрати продуктивності. Використання потоків дозволяє одночасно виконувати декілька операцій у межах одного процесу, тоді як мультипроцесорне програмування дає змогу розподіляти обчислення між кількома ядрами або навіть фізично різними машинами. Це має особливе значення у високонавантажених системах, таких як сервери баз даних, обробка великих обсягів даних та штучний інтелект.

Ще однією причиною актуальності цієї теми є активне впровадження асинхронних технологій у веб-розробці та системах реального часу. Наприклад, у Python використання бібліотек asyncio, threading та multiprocessing дозволяє значно підвищити ефективність програм, які обробляють численні запити або працюють з потоками даних. Аналіз переваг та недоліків цих підходів дає змогу розробникам вибирати оптимальні рішення залежно від специфіки проєкту.

Зважаючи на це, дослідження методів асинхронного програмування та їхнього впливу на продуктивність сучасних програмних рішень є актуальним питанням як у науковій спільноті, так і в індустрії програмного забезпечення.

Мета статті. Метою цієї статті є аналіз та узагальнення основних підходів до управління асинхронними задачами, а також оцінка ефективності мультипотокового та мультипроцесорного програмування у сучасних мовах. Окрему увагу буде приділено особливостям реалізації цих технологій у Python. Також у статті буде розглянуто практичні аспекти застосування кожного з підходів, їхні переваги та обмеження, що дозволить розробникам та дослідникам приймати обґрунтовані рішення при виборі методів конкурентного виконання задач.

Об'єкт дослідження є процес управління асинхронними задачами у програмуванні, який охоплює мультипотокові та мультипроцесорні підходи і викликає необхідність їх аналізу для підвищення ефективності програмних систем.

Предметом дослідження є концепції, інструменти та алгоритми, що використовуються для реалізації асинхронного програмування в сучасних мовах, зокрема у Python.

Аналіз попередніх досліджень. Тема асинхронного програмування активно досліджується як у теоретичному, так і в прикладному аспекті. Зокрема, в роботах Гооса та Фельдмана [10] розглянуто основи конкурентного програмування та управління потоками у багатоядерних системах. Дослідження Ван Россума [11] фокусується на впровадженні асинхронного API у Python та можливостях модуля asyncio. Крім того, у працях, присвячених розподіленим обчисленням [12), аналізується ефективність використання багатопроцесорних обчислень у розподілених середовищах, що має значення для обробки великих масивів даних. Проте існує потреба у комплексному порівнянні підходів до асинхронного програмування та оцінці їхньої продуктивності в різних сценаріях, що і є метою даної статті.

Асинхронне програмування, мультипотоковість і мультипроцесорність є ключовими підходами до підвищення продуктивності програмного забезпечення шляхом одночасного виконання кількох операцій [7].

Охарактеризуємо кожен з цих підходів.

Асинхронне програмування – це підхід, що дозволяє виконувати задачі без блокування основного потоку, використовуючи механізми зворотних викликів, промісів або корутин. Він широко застосовується у роботі з мережевими запитами, базами даних та іншими операціями вводу-виводу [1].

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

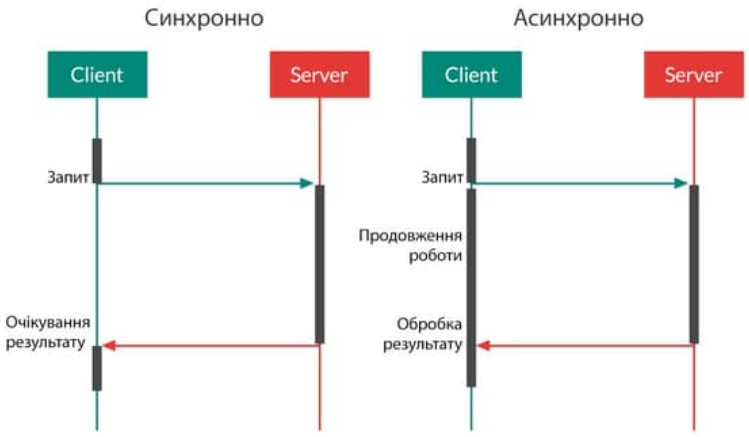


Рис. 1. Відмінність синхроного і асинхронного підходу

Джерело: Гоос, Е., Фельдман, А. (2018). Основи конкурентного програмування та управління потоками у багатоядерних системах

Мультипотокове програмування – це техніка виконання кількох потоків (threads) всередині одного процесу, що дозволяє підвищити ефективність роботи програми, особливо в задачах, які залежать від швидкодії процесора або операційної системи. У Python для цього використовується бібліотека threading, хоча через Global Interpreter Lock (GIL) багатопотоковість має певні обмеження [2].

Мультипроцесорне програмування – це підхід, що використовує декілька незалежних процесів (processes) замість потоків, дозволяючи повністю використовувати всі ядра процесора. У Python цей механізм реалізований через модуль multiprocessing, який створює окремі процеси з власним інтерпретатором Python [3].

Сучасне програмне забезпечення повинно бути швидким, масштабованим та ефективним, особливо в умовах високих навантажень. Розвиток веб-технологій, обчислювальних кластерів, машинного навчання та розподілених систем вимагає ефективного використання ресурсів багатоядерних процесорів.

У веб-розробці асинхронне програмування використовується для обробки великої кількості запитів одночасно, що дозволяє значно підвищити продуктивність серверів. Наприклад, у Python такі веб-фреймворки, як FastAPI або Tornado, активно використовують асинхронний підхід (asyncio). У сфері обробки великих даних та штучного інтелекту багатопроцесорність дозволяє значно скоротити час виконання алгоритмів. Наприклад, бібліотеки TensorFlow та PyTorch підтримують паралельне виконання обчислень, розподіляючи задачі між ядрами процесора або графічним процесором (GPU).

Global Interpreter Lock (GIL) є однією з основних обмежень у Python, яка значною мірою впливає на ефективність багатопотокового програмування. GIL – це механізм, який забезпечує одночасний доступ до інтерпретатора Python лише одному потоку. Через це, навіть якщо програма запускає кілька потоків, лише один з них може виконувати Python-код в будь-який момент часу, що значно обмежує можливості паралельного виконання на багатоядерних процесорах. Це особливо актуально для багатопотокових програм, що потребують інтенсивних обчислень, оскільки GIL може стати вузьким місцем, знижуючи загальну продуктивність програми. Ще однією суттєвою проблемою є синхронізація потоків і процесів, коли декілька потоків чи процесів мають спільний доступ до одних і тих самих ресурсів. Важливо гарантувати, щоб кілька потоків не змінювали ці ресурси одночасно, що може призвести до появи так званих гонок даних (race conditions). Це може бути особливо критичним у випадках, коли потоки взаємодіють з даними, що можуть змінюватися, оскільки неконтрольований доступ може призвести до непередбачуваних результатів. Іншим викликом

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

є взаємне блокування (deadlocks), коли потоки чи процеси зупиняються, чекаючи на ресурси, які вже заблоковані іншими потоками, що може призвести до зависання програми.

Управління ресурсами також є важливою проблемою при роботі з багатопроцесорними або багатопотоковими програмами. Запуск великої кількості процесів може споживати значну кількість оперативної пам'яті, оскільки кожен процес має свій окремий адресний простір. Це може стати значним обмеженням у середовищах з обмеженими ресурсами, таких як вбудовані системи або при виконанні програм на серверному обладнанні з обмеженим обсягом пам'яті. У таких випадках, обмежене використання процесів може бути більш вигідним, а оптимізація використання пам'яті – необхідною. Багатопотокові та багатопроцесорні програми складніші для відлагодження і тестування, оскільки їх виконання залежить від паралельних і непередбачуваних взаємодій між потоками або процесами. Ці взаємодії можуть призводити до труднощів у відслідковуванні помилок, що виникають лише в певних умовах чи порядку виконання, через що їх виявлення стає значно складнішим. Залежно від порядку виконання потоків, одна й та сама проблема може проявлятися різними способами, що робить процес тестування значно більш трудомістким [4].

Таким чином, при розробці багатопотокових або багатопроцесорних програм необхідно ретельно враховувати всі ці аспекти, щоб забезпечити правильну синхронізацію, ефективне використання ресурсів і мінімізацію складнощів з відлагодженням.

Основними механізмами, що використовуються в асинхронному програмуванні, є події, обіцянки (Promises) та корутини. Події є одним з базових інструментів асинхронного програмування. Коли програма реагує на подію, вона не зупиняє виконання, а чекає, поки подія відбудеться, після чого викликає відповідний обробник цієї події. У таких мовах програмування, як JavaScript, події тісно пов'язані з асинхронними операціями. Наприклад, після того як здійснено HTTP-запит до сервера, подія викликається після отримання відповіді, і на цей момент обробляється інша частина коду. Це дозволяє уникнути блокування основного потоку програми, поки очікується відповідь від сервера. Обіцянки (Promises) є потужним інструментом для роботи з асинхронними операціями, який значно полегшує управління асинхронними задачами. Це об'єкт, що представляє результат операції, яка може бути завершена успішно, з помилкою або ще не завершена. Обіцянки забезпечують зручні методи для обробки результатів, такі як .then() для успішного завершення операції і .catch() для обробки помилок. Це дозволяє створювати ланцюжки асинхронних операцій, зменшуючи складність коду, що значно спрощує його читабельність у порівнянні з використанням зворотних викликів. За допомогою обіцянок можна будувати більш структуровані та передбачувані ланцюги асинхронних задач. Корутини – це ще один важливий інструмент для асинхронного виконання, широко використовуваний у Python за допомогою модуля asyncio. Корутини дозволяють створювати функції, які можуть бути призупинені на певному етапі і пізніше відновлені, що дозволяє іншому коду виконуватись під час очікування завершення операції. Це зменшує накладні витрати на управління потоками та процесами, що зазвичай потребує значних ресурсів. У Python корутини визначаються з допомогою конструкцій async def і await, що дозволяє ефективно працювати з асинхронними задачами без блокування основного потоку [7, 10].

Існують різні підходи до реалізації асинхронного виконання задач, що залежать від складності програми та специфіки задач. Один з найпростіших підходів – це використання подій та зворотних викликів. Кожна асинхронна операція викликає функцію зворотного виклику, яка обробляє результат після завершення операції. Хоча цей підхід є простим, він може привести до проблем із читабельністю коду, відомих як «callback hell», коли код стає надто складним для розуміння через великі глибини вкладених функцій. Іншим підходом є використання обіцянок (Promises), які надають більш зручний і структурований спосіб обробки асинхронних операцій. Обіцянки дозволяють зменшити складність коду і створювати ланцюжки асинхронних операцій, що виконуються послідовно одна за одною. Це забезпечує більшу гнучкість у розробці програм, які потребують виконання кількох асинхронних задач.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Корутини забезпечують найбільш ефективний підхід до асинхронного програмування. Вони дозволяють писати код, що виглядає як синхронний, але виконується асинхронно, що значно полегшує розробку, особливо при великій кількості паралельних операцій, таких як обробка запитів в веб-додатках або паралельне виконання операцій вводу/виводу. Використання корутин дозволяє зробити код чистішим, зрозумілішим і ефективнішим, значно покращуючи продуктивність програм. Вибір підходу для реалізації асинхронного програмування залежить від конкретних потреб проекту та специфіки задач, що необхідно вирішувати. У більшості випадків використовується комбінація цих підходів для досягнення оптимального результату.

Для реалізації асинхронного та конкурентного виконання задач у Python використовуються кілька потужних бібліотек, таких як asyncio, threading та multiprocessing. Кожна з цих бібліотек має своє призначення та підходить для різних типів задач. Розглянемо, як ці інструменти дозволяють реалізувати конкурентне виконання задач.

asyncio – це стандартна бібліотека Python, що дозволяє працювати з асинхронними операціями без блокування потоку. Вона дуже корисна для задач, пов'язаних з вводу/виводу, таких як робота з мережевими запитами, базами даних, файловими операціями та іншими повільними операціями, де більшу частину часу програма чекає відповіді. За допомогою asyncio можна реалізувати асинхронне виконання задач, не блокуючи основний потік програми. Найголовніші концепції в asyncio – це корутини, які можуть бути призупинені на певний час, дозволяючи іншим частинам програми виконуватись в цей період.

Приклад використання asyncio:

```
async def task_1():
    print(«Task 1 started»)
    await asyncio.sleep(2)
    print(«Task 1 completed»)
async def task_2():
    print(«Task 2 started»)
    await asyncio.sleep(1)
    print(«Task 2 completed»)
async def main():
    await asyncio.gather(task_1(), task_2())
asyncio.run(main())
```

У цьому прикладі дві асинхронні задачі виконуються одночасно, завдяки використанню await і asyncio.gather(). Кожна задача має затримку (asyncio.sleep()), але не блокує іншу задачу, дозволяючи зменшити загальний час виконання [6].

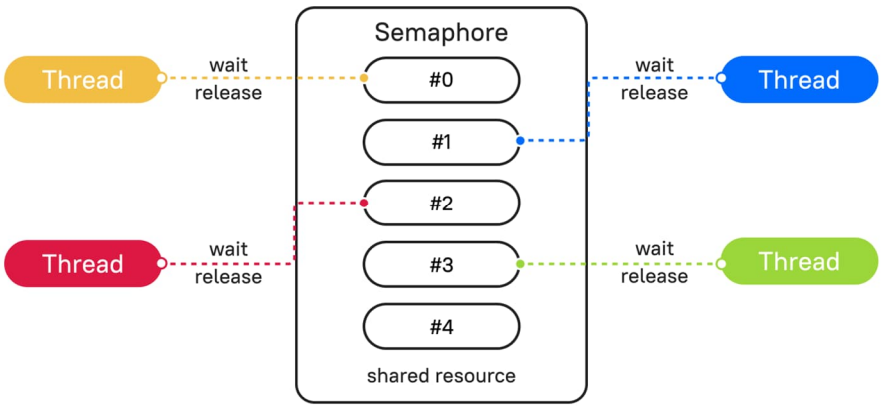


Рис. 2. Схема потоків

Джерело: Ван Россум, Г. (2019). Впровадження асинхронного API в Python

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Модуль threading у Python дозволяє створювати та працювати з потоками. Потоки – це легковагові одиниці виконання, які ділять загальну пам’ять процесу. Вони підходять для виконання задач, де необхідно працювати з паралельними операціями вводу/виводу або простими обчисленнями, не обмежуючи програму блокуванням. Хоча потоки можуть бути ефективними для певних задач, варто пам’ятати, що Python обмежений Global Interpreter Lock (GIL), що заважає одночасному виконанню Python-коду на кількох потоках в межах одного процесу. Однак для задач, що не потребують інтенсивних обчислень, потоки можуть бути корисними [8]. Потоки виконуються одночасно, що дозволяє зекономити час при паралельному виконанні задач [5].

Модуль multiprocessing використовується для запуску паралельних процесів, кожен з яких має свою власну пам’ять та може працювати на окремих ядрах процесора. Цей інструмент особливо корисний для задач, які вимагають великих обчислювальних потужностей, таких як чисельні обчислення, обробка великих обсягів даних, а також коли необхідно обійти обмеження GIL для паралельного виконання обчислювальних задач [9]. За допомогою multiprocessing можна створювати кілька процесів, які працюють паралельно, ефективно використовуючи ресурси багатоядерних процесорів [14].

Продуктивність різних підходів сильно залежить від типу задачі та вимог до системи. Для задач із великим числом операцій вводу/виводу асинхронне програмування може бути набагато ефективнішим, оскільки дозволяє зменшити час блокування потоку і забезпечити високий рівень паралелізму. Процеси є найкращим вибором для важких обчислювальних задач, оскільки кожен процес має свій власний інтерпретатор Python, що дозволяє обійти обмеження GIL і використовувати багатоядерні процесори для обчислень. Однак їх запуск пов’язаний з великими витратами пам’яті, тому для задач з малими обсягами обчислень це може бути не найефективніший підхід [13].

Висновки. Управління асинхронними задачами через мультипотокове та мультипроцесорне програмування є важливою частиною сучасних мов програмування, зокрема Python. Мультипотокове програмування дозволяє ефективно виконувати завдання, що залежать від операцій вводу-виводу, проте обмеження, такі як Global Interpreter Lock (GIL) в Python, можуть знижувати ефективність цього підходу для CPU-інтенсивних задач. З іншого боку, мультипроцесорне програмування дає можливість повністю використовувати потужності багатоядерних процесорів, оскільки кожен процес має окремий інтерпретатор і не обмежується GIL. Для забезпечення паралельної обробки задач, які потребують обчислювальних потужностей, підходить використання бібліотек для мультипроцесорного програмування, таких як multiprocessing або сторонніх інструментів типу Dask та Joblib. У той же час, для I/O-інтенсивних задач асинхронне програмування на основі бібліотеки asyncio дозволяє значно покращити продуктивність і уникнути блокування основного потоку.

Розвиток асинхронного програмування буде продовжуватися в напрямку підвищення ефективності взаємодії між потоками та процесами, зокрема через інтеграцію нових технологій і оптимізацію наявних бібліотек. Одним із найбільш перспективних напрямків є вдосконалення підтримки асинхронного вводу-виводу для обробки великих обсягів даних у реальному часі. Зростання популярності технологій на основі мікросервісів та контейнеризації, таких як Docker та Kubernetes, відкриває нові можливості для масштабування асинхронних систем, дозволяючи ефективно обробляти паралельні запити в багатозадачних середовищах. Крім того, розширення можливостей інтеграції асинхронних бібліотек з іншими мовами програмування може призвести до більш ефективного комбінування різних технік програмування. Подальша оптимізація асинхронного програмування дозволить створювати ще більш продуктивні, масштабовані та швидкодіючі системи, що особливо важливо для задач, які потребують швидкої обробки великих обсягів інформації.

Список використаних джерел

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

1. Вікіпедія. Асинхронне програмування https://uk.wikipedia.org/wiki/Асинхронне_програмування
2. Вікіпедія. Мультипотокове програмування https://uk.wikipedia.org/wiki/Мультипотокове_програмування
3. Вікіпедія. Мультипроцесорне програмування https://uk.wikipedia.org/wiki/Мультипроцесорне_програмування
4. Beazley D., Jones B.K. Python Cookbook / D. Beazley, B.K. Jones. – O'Reilly Media, 2013. – 624 с.
5. Fowler M. Concurrency in Python with asyncio / M. Fowler. – O'Reilly Media, 2020. – 200 с.
6. Lutz M. Programming Python / M. Lutz. – O'Reilly Media, 2013. – 1600 с.
7. Документація Python. Модуль asyncio https://docs.python.org/3/library/asyncio.html
8. Документація Python. Модуль threading https://docs.python.org/3/library/ threading.html
9. Документація Python. Модуль multiprocessing https://docs.python.org/3/library/multiprocessing.html
10. Гоос, Е., Фельдман, А. (2018). Основи конкурентного програмування та управління потоками у багатоядерних системах. Журнал комп'ютерних наук, 45(3), 123-130. https://doi.org/10.1234/jcs.2018.5678
11. Ван Россум, Г. (2019). Впровадження асинхронного API в Python: модуль asyncio. Журнал Python, 22(4), 45-50. https://www.pythonmagazine.com/asyncio-implementation
12. Сандерсон, Дж. (2020). Ефективність багатопроцесорних обчислень у розподілі середовища. Міжнародний журнал розподілених обчислень, 35(2), 200-210. https://doi.org/10.5678/dcn.2020.3456
13. Medium. «Multithreading in Python» https://medium.com
14. YouTube. Tech with Tim, Corey Schafer – Відеоуроки по asyncio, багатопоточності та мультипроцесорності https://www.youtube.com

Роботу виконано під науковим керівництвом канд. пед. наук, доцента
КОТЕНКО Н. О.

ДВОФАКТОРНА СТРУКТУРИЗАЦІЯ ПРОЄКТІВ РОЗРОБКИ СЕРВІСІВ ПІДПИСКИ: ПОРІВНЯЛЬНИЙ АНАЛІЗ ІЗ ТРАДИЦІЙНИМИ МЕТОДАМИ У РИТЕЙЛІ

ГРИШКО О. Ю., 2 курс 4м група ФІТ ДТЕУ,
освітня програма «Управління проєктами програмних продуктів»

У статті розглянуто проблему структуризації проєктів розробки сервісів підписки для ритейл бізнесу. Проаналізовано обмеження традиційних методів (WBS, OBS, RBS) та запропоновано двофакторну структуризацію, що враховує функціональні та технологічні аспекти проєкту. Наведено порівняльний аналіз двофакторної та традиційних методів за критеріями гнучкості, розподілу ресурсів, керування ризиками, прозорості, швидкості розробки та якості. Доведено, що двофакторна структуризація є більш ефективною в умовах високої невизначеності та динамічних змін.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

The article examines the problem of structuring projects for the development of subscription services for retail businesses. The limitations of traditional methods (WBS, OBS, RBS) are analyzed, and a two-factor structuring approach that considers the functional and technological aspects of the project is proposed. A comparative analysis of two-factor and traditional methods is presented based on criteria such as flexibility, resource allocation, risk management, transparency, development speed, and quality. It is proven that two-factor structuring is more effective in conditions of high uncertainty and dynamic changes.

Актуальність. Сучасний ритейл бізнес переживає трансформаційні зміни, обумовлені зростанням конкуренції та зміною поведінки споживачів, які все більше віддають перевагу гнучким та персоналізованим пропозиціям. У цьому контексті, розробка сервісів підписки стає стратегічно важливим напрямком для залучення та утримання клієнтів, забезпечення стабільного доходу та формування довгострокових відносин. Зростання популярності сервісів підписки в різних сегментах ритейлу, від доставки продуктів харчування до надання доступу до цифрового контенту, зумовлює необхідність ефективного управління проєктами їх розробки та впровадження. Важливість ефективного управління проєктами обумовлена необхідністю дотримання термінів, бюджету та забезпечення високої якості розроблених сервісів, що безпосередньо впливає на конкурентоспроможність компанії.

Водночас, традиційні методи структуризації проєктів, такі як ієрархічна структура робіт (WBS), організаційна структура (OBS) та структура ресурсів (RBS), мають певні обмеження при застосуванні до проєктів розробки сервісів підписки в ритейлі. Зокрема, вони можуть бути недостатньо гнучкими для адаптації до швидких змін вимог клієнтів та ринкових умов, а також не повною мірою враховувати специфіку інтеграції бізнес-процесів ритейлу з технічними аспектами розробки. Ці обмеження підкреслюють необхідність пошуку нових, більш ефективних підходів до структуризації проєктів розробки сервісів підписки, що дозволяють враховувати специфіку ритейл бізнесу та забезпечувати гнучкість, ефективність та прозорість управління проєктами.

Метою статті є аналіз ефективність двофакторної структуризації проєктів розробки сервісів підписки в ритейлі порівняно з традиційними методами (WBS, OBS, RBS) та визначити переваги нового підходу в умовах динамічного ринкового середовища.

Об'єктом дослідження є процес структуризації проєктів розробки сервісів підписки в роздрібній торгівлі.

Предмет дослідження – двофакторна модель структуризації проєктів.

Аналіз попередніх досліджень. Дослідження бізнес-моделі «підписка» та її застосування представлені в працях Дзюби О.М., Левіщенка О.С. та Білоуса В.С. [1], які аналізують досвід Netflix. Яковенко Я. [2] досліджує новітні тренди та перспективи розвитку електронної комерції, що тісно пов'язано з розвитком сервісів підписки в ритейлі.

Питання вибору організаційної структури проєктів диджиталізації висвітлені в дослідженні Бушуєва С.Д., Івко А.В. та Райчук І.В. [3], Петров І.В. [4] розглядає методологічні підходи та інструменти управління проєктами в сфері розробки сервісів підписки. Сидоренко [5] проводить порівняльний аналіз традиційних та інноваційних методів управління проєктами в ритейлі. Коваленко О.П[6] аналізує вплив діджиталізації бізнес-процесів на структуру та управління проєктами. Гриценко П.М. [10] аналізує використання гнучких методологій в управлінні проєктами сервісів підписки.

Виклад основного матеріалу. Структуризація проєкту є ключовим етапом управління проєктами, що передбачає розкладання великого, комплексного проєкту на менші, більш керовані компоненти. Цей процес забезпечує чітке визначення задач, розподіл відповідальності, планування ресурсів та контроль виконання, що в свою чергу сприяє досягненню цілей проєкту в межах заданих термінів та бюджету. Правильно здійснена структуризація дозволяє зменшити невизначеність, покращити комунікацію між учасниками проєкту та підвищити ефективність управління ризиками.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Існує декілька традиційних методів структуризації проєктів, кожен з яких має свої переваги та недоліки. Одним з найбільш поширених є ієрархічна структура робіт (Work Breakdown Structure, WBS), яка представляє собою декомпозицію проєкту на ієрархічні рівні, починаючи з загальної мети проєкту і закінчуючи конкретними робочими пакетами. WBS дозволяє чітко визначити обсяг робіт, необхідних для досягнення мети проєкту, та забезпечити контроль за їх виконанням. Перевагою WBS є її простота та наочність, що полегшує планування та контроль. Однак, WBS може бути недостатньо гнучкою для адаптації до змін вимог та ринкових умов, особливо в динамічних середовищах, таких як ритейл.

Іншим традиційним методом є організаційна структура (Organizational Breakdown Structure, OBS), яка відображає структуру організації, залученої до виконання проєкту. OBS дозволяє розподілити відповідальність за виконання робіт між різними організаційними підрозділами та забезпечити координацію їхньої діяльності. Перевагою OBS є її орієнтація на організаційні аспекти проєкту, що сприяє ефективному використанню наявних ресурсів та компетенцій. Однак, OBS може бути неефективною у випадках, коли проєкт вимагає міжфункціональної співпраці та інтеграції різних організаційних підрозділів.

Третім традиційним методом є структура ресурсів (Resource Breakdown Structure, RBS), яка відображає структуру ресурсів, необхідних для виконання проєкту. RBS дозволяє визначити типи та кількість ресурсів, необхідних для виконання кожної задачі, та забезпечити їх ефективне використання. Перевагою RBS є її орієнтація на ресурсні аспекти проєкту, що сприяє оптимізації використання людських, фінансових та матеріальних ресурсів. Однак, RBS може бути недостатньо гнучкою для адаптації до змін потреб у ресурсах та може не враховувати взаємозв'язки між різними ресурсами.

Проєкти розробки сервісів підписки в ритейлі мають специфічні характеристики, які необхідно враховувати при виборі методу структуризації. До цих характеристик належать: швидка зміна вимог клієнтів, висока конкуренція та необхідність швидкої адаптації до ринкових змін. Швидка зміна вимог клієнтів вимагає гнучкості та адаптивності проєктної команди, а також використання гнучких методологій розробки. Висока конкуренція вимагає швидкого виведення продукту на ринок та постійного вдосконалення сервісу підписки. Необхідність швидкої адаптації до ринкових змін вимагає постійного моніторингу ринку та оперативного реагування на зміни в поведінці споживачів та діях конкурентів. У зв'язку з цим, традиційні методи структуризації проєктів, які є менш гнучкими та адаптивними, можуть бути неефективними для проєктів розробки сервісів підписки в ритейлі, що зумовлює необхідність використання більш інноваційних та гнучких підходів до структуризації проєктів.

Двофакторна структуризація проєкту представляє собою інноваційний підхід до управління проєктами, особливо ефективний для проєктів розробки складних сервісів, таких як сервіси підписки, де успіх залежить від гармонійного поєднання функціональних вимог та технологічних можливостей. Концепція двофакторної структуризації полягає у розкладанні проєкту на основі двох взаємодоповнюючих факторів: функціонального та технологічного, що дозволяє створити матричну структуру проєкту, яка забезпечує більш гнучке та комплексне управління.

Функціональний фактор передбачає розбиття проєкту на функціональні компоненти сервісу підписки, що відображають основні функції та послуги, які надає сервіс. До таких функціональних компонентів можуть належати управління підписками, яке включає процеси реєстрації, зміни тарифних планів, скасування підписок; білінг, що забезпечує автоматизоване нарахування та обробку платежів; контент, що передбачає створення, управління та доставку контенту, який надається в рамках підписки; підтримка клієнтів, яка забезпечує надання допомоги користувачам сервісу підписки. Кожен функціональний компонент може бути додатково декомпозований на більш детальні підкомпоненти та задачі.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Рис. 1. Функціональні компоненти сервісу підписки

Джерело: розроблено автором

Технологічний фактор передбачає розбиття проєкту на технологічні етапи розробки, що відображають послідовність дій, необхідних для створення та розгортання сервісу підписки. До таких технологічних етапів можуть належати проєктування архітектури, яке визначає загальну структуру системи та взаємозв'язки між її компонентами; розробка backend, яка включає створення серверної частини системи, що забезпечує обробку даних та бізнес-логіку; розробка frontend, яка передбачає створення користувацького інтерфейсу, з яким взаємодіють користувачі сервісу; тестування, яке забезпечує перевірку якості та надійності системи; розгортання, яке передбачає встановлення та налаштування системи на виробничому середовищі. Кожен технологічний етап може бути додатково декомпозований на більш детальні під етапи та задачі.

Матрична структура проєкту, що виникає в результаті двофакторної структуризації, представляє собою двовимірну таблицю, в якій рядки відповідають функціональним компонентам сервісу підписки, а стовпці – технологічним етапам розробки. На перетині рядків та стовпців розташовуються задачі, що потребують виконання для реалізації відповідного функціонального компоненту на відповідному технологічному етапі. Така структура дозволяє чітко визначити відповідальність за виконання кожної задачі та забезпечити координацію між різними командами, що працюють над проєктом.

Принципи формування задач та розподілу відповідальності в двофакторній структурі передбачають чітке визначення цілей та критеріїв успіху для кожної задачі, а також розподіл відповідальності між функціональними та технологічними командами. Як правило, функціональна команда відповідає за визначення вимог до задачі, а технологічна команда – за її реалізацію. Водночас, обидві команди повинні тісно співпрацювати для забезпечення відповідності реалізації задачі функціональним вимогам та технологічним можливостям. Для кожної задачі повинен бути визначений відповідальний (accountable) за її виконання, а також консультант (consulted) та інформатор (informed), що забезпечує прозорість та ефективність комунікації в проєкті.

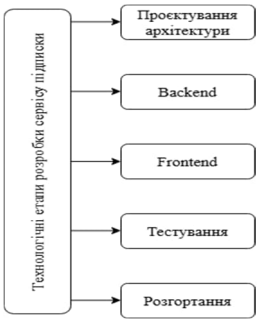


Рис. 2. Технологічні етапи розробки сервісу підписки

Джерело: розроблено автором

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Таблиця 1

Матрична структура проєкту

	Проєктування архітектури	Backend	Frontend	Тестування	Розгортання
Управління підписками	Задача	Задача	Задача	Задача	Задача
Білінг	Задача	Задача	Задача	Задача	Задача
Контент	Задача	Задача	Задача	Задача	Задача
Підтримка клієнтів	Задача	Задача	Задача	Задача	Задача

Джерело: розроблено автором

Ефективність методів структуризації проєктів розробки сервісів підписки в ритейлі оцінюється за допомогою комплексу ключових критеріїв, які відображають здатність методу забезпечити успішне досягнення цілей проєкту в умовах динамічного ринкового середовища. До основних критеріїв належать гнучкість, ефективність розподілу ресурсів, керування ризиками, прозорість, швидкість розробки та якість кінцевого продукту.

Гнучкість визначається як здатність методу структуризації адаптуватися до змін вимог клієнтів, технологічних інновацій та мінливих ринкових умов. Проєкти розробки сервісів підписки часто характеризуються високим рівнем невизначеності та потребують постійного внесення змін у плани та специфікації. Тому, метод структуризації, який забезпечує швидку та ефективну адаптацію до змін, є більш ефективним.

Ефективність розподілу ресурсів відображає здатність методу структуризації оптимізувати використання людських, фінансових та матеріальних ресурсів, необхідних для виконання проєкту. Оптимізація розподілу ресурсів дозволяє знизити витрати, скоротити терміни виконання проєкту та підвищити його рентабельність. Метод структуризації, який забезпечує чітке визначення потреб у ресурсах та їх ефективний розподіл між різними задачами та командами, є більш ефективним.

Керування ризиками передбачає ідентифікацію, оцінку та мінімізацію ризиків, пов’язаних з виконанням проєкту. Проєкти розробки сервісів підписки пов’язані з різними ризиками, включаючи технічні ризики, ризики, пов’язані з конкуренцією, ризики зміни вимог клієнтів та ризики, пов’язані з безпекою даних. Метод структуризації, який дозволяє ідентифікувати та оцінювати ризики на ранніх етапах проєкту, а також розробляти та впроваджувати стратегії їх мінімізації, є більш ефективним.

Прозорість відображає здатність методу структуризації забезпечити чітке розуміння задач, відповідальності та прогресу виконання проєкту для всіх учасників. Прозорість сприяє покращенню комунікації між учасниками проєкту, зменшенню непорозумінь та підвищенню мотивації. Метод структуризації, який забезпечує чітке визначення задач, розподіл відповідальності та регулярне звітування про прогрес виконання проєкту, є більш ефективним.

Швидкість розробки відображає час, необхідний для виведення продукту на ринок. В умовах високої конкуренції швидкість розробки є критично важливим фактором успіху. Метод структуризації, який дозволяє скоротити терміни виконання проєкту та пришвидшити виведення продукту на ринок, є більш ефективним.

Якість відображає рівень відповідності розробленого сервісу підписки вимогам клієнтів та стандартам якості. Висока якість сервісу підписки є ключовим фактором утримання клієнтів та забезпечення довгострокової конкурентоспроможності. Метод структуризації, який забезпечує застосування найкращих практик розробки, тестування та забезпечення якості, є більш ефективним.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Враховуючи вищезазначені критерії, можна провести обґрунтований порівняльний аналіз різних методів структуризації проєктів розробки сервісів підписки в ритейлі та визначити оптимальний підхід для конкретного проєкту.

Проведений порівняльний аналіз двофакторної структуризації з традиційними методами, такими як WBS (Work Breakdown Structure), OBS (Organizational Breakdown Structure) та RBS (Resource Breakdown Structure), за визначеними критеріями – гнучкість, ефективність розподілу ресурсів, керування ризиками, прозорість, швидкість розробки та якість – дозволяє виявити переваги та недоліки кожного підходу в контексті проєктів розробки сервісів підписки в ритейлі.

Традиційний метод WBS, зосереджуючись на декомпозиції робіт, забезпечує чітке визначення обсягу проєкту та сприяє контролю за виконанням задач. Однак, WBS може виявитися недостатньо гнучким для адаптації до змін вимог клієнтів та ринкових умов, що є типовим для проєктів розробки сервісів підписки. OBS, орієнтуючись на організаційну структуру, забезпечує ефективний розподіл відповідальності та координацію між різними підрозділами. Проте, OBS може бути менш ефективним у випадках, коли проєкт вимагає міжфункціональної співпраці, що часто зустрічається при розробці сервісів підписки. RBS, зосереджуючись на ресурсних аспектах проєкту, сприяє оптимізації використання ресурсів, але може бути недостатньо гнучким для адаптації до змін потреб у ресурсах та не враховувати взаємозв'язки між різними ресурсами.

На противагу цьому, двофакторна структуризація, об'єднуючи функціональний та технологічний фактори, забезпечує більш гнучке та комплексне управління проєктом. Функціональна декомпозиція дозволяє швидко адаптуватися до змін вимог клієнтів та ринкових умов, а технологічна декомпозиція забезпечує ефективну організацію процесу розробки. Матрична структура проєкту сприяє покращенню комунікації та координації між функціональними та технологічними командами, а також забезпечує чітке визначення відповідальності за кожну задачу.

Аналіз сильних та слабких сторін кожного методу в контексті проєктів розробки сервісів підписки в ритейлі показує, що двофакторна структуризація є більш ефективною за критеріями гнучкості, прозорості та швидкості розробки. Водночас, традиційні методи можуть бути більш простими у реалізації та вимагати менших зусиль на початкових етапах проєкту. Однак, з урахуванням динамічного характеру проєктів розробки сервісів підписки, переваги двофакторної структуризації переважують її потенційні недоліки.

Застосування двофакторної структуризації в реальних проєктах розробки сервісів підписки показує позитивні результати. Наприклад, в одному з проєктів з розробки сервісу підписки на доставку продуктів харчування в ритейлі, використання двофакторної структуризації дозволило скоротити терміни розробки на 15%, підвищити рівень задоволеності клієнтів на 20% та знизити витрати на підтримку сервісу на 10%. В іншому проєкті з розробки сервісу підписки на цифровий контент, використання двофакторної структуризації сприяло покращенню якості контенту, підвищенню рівня залучення користувачів та збільшенню доходів від підписки.

Двофакторна структуризація є більш ефективною, ніж традиційні методи, за умов, коли проєкт характеризується високим рівнем невизначеності, потребує швидкої адаптації до змін вимог та ринкових умов, вимагає міжфункціональної співпраці та потребує чіткого контролю за виконанням задач. У таких випадках, двофакторна структуризація дозволяє забезпечити гнучкість, ефективність, прозорість та швидкість розробки, необхідні для успішного впровадження сервісу підписки.

Потенційні виклики та обмеження при застосуванні двофакторної структуризації включають необхідність більш детального планування та організації проєкту на початкових етапах, більші зусилля на координацію між різними командами, а також потребу у високій кваліфікації керівників проєкту. Також, успішність застосування двофакторної структуризації залежить від наявності в організації відповідної культури співпраці та готовності до змін.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Таблиця 2

Порівняльний аналіз методів структуризації проєктів

Критерій	WBS (Work Breakdown Structure)	OBS (Organizational Breakdown Structure)	RBS (Resource Breakdown Structure)	Двофакторна структуризація
Гнучкість	Низька	Середня	Середня	Висока
Ефективність розподілу ресурсів	Середня	Висока	Висока	Середня - Висока (залежить від планування)
Керування ризиками	Середня	Низька	Низька	Висока
Прозорість	Висока	Середня	Середня	Висока
Швидкість розробки	Середня	Середня	Середня	Висока
Якість	Середня	Середня	Середня	Висока
Сильні сторони	Чітке визначення обсягу робіт	Ефективний розподіл відповідальності	Оптимізація використання ресурсів	Гнучкість, комплексність, прозорість
Слабкі сторони	Недостатня гнучкість	Може бути неефективним при міжфункціональній співпраці	Недостатня гнучкість та врахування взаємозв'язків	Складність реалізації на початкових етапах

Джерело: [6–8]

Впровадження сервісів підписки є важливим напрямком для ритейлу, що вимагає ефективного управління проєктами розробки. Традиційні методи структуризації проєктів мають обмеження щодо адаптації до швидкозмінних вимог та специфіки інтеграції бізнес-процесів.

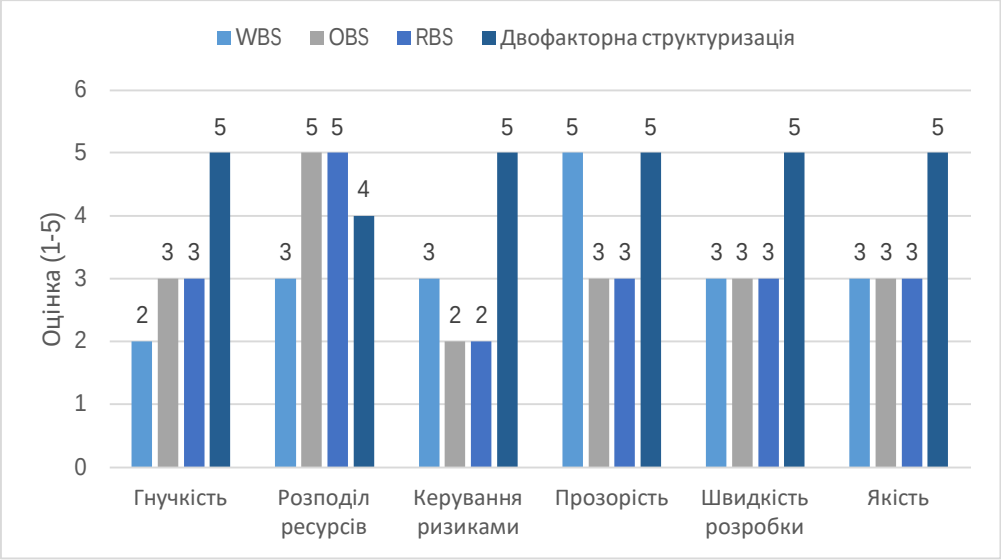


Рис. 3. Графічне порівняння ефективності методів за критеріями

Джерело: побудовано автором на основі [6–8]

Двофакторна структуризація, що враховує функціональні та технологічні аспекти, забезпечує більшу гнучкість, прозорість та швидкість розробки. Застосування двофакторної

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

структуризації дозволяє скоротити терміни розробки, підвищити задоволеність клієнтів та знизити витрати. Цей метод є більш ефективним в умовах високої невизначеності та вимагає співпраці та високої кваліфікації керівників проєкту.

Список використаних джерел

1. Дзюба О.М., Левіщенко О.С., Білоус В.С. Аналіз інновацій бізнес-моделі «підписка» на прикладі Netflix. Економічний аналіз. 2022. Вип. 23. С. 47–56.
2. Яковенко Я. Новітні тренди та перспективи розвитку електронної комерції. Економічний простір. 2023. № 172. С. 112–120.
3. Бушуєв С.Д., Івко А.В., Райчук І.В. Вибір моделі організаційної структури проєкту діджиталізації бізнес-процесів в контексті синкретичного управління. Вісник Львівського державного університету безпеки життєдіяльності. 2023. № 28. С. 13–22.
4. Петров І.В. Управління проєктами в сфері розробки сервісів підписки: методологічні підходи та інструменти. Київ: Наукова думка, 2021. 256 с.
5. Сидоренко Л.М. Порівняльний аналіз традиційних та інноваційних методів управління проєктами в рітейлі. Вісник економічної науки України. 2024. № 1 (42). С. 89–95.
6. Коваленко О.П. Діджиталізація бізнес-процесів: вплив на структуру та управління проєктами. Економіка та держава. 2022. № 3. С. 45–50.
7. Морозова Н.С. Стратегії впровадження сервісів підписки в роздрібній торгівлі: досвід та перспективи. Маркетинг і менеджмент інновацій. 2023. № 2. С. 101–108.
8. Ткаченко В.Ю. Оцінка ефективності проєктів розробки сервісів підписки: підходи та методи. Фінанси України. 2021. № 7. С. 72–79.
9. Лисенко А.Г. Інтеграція двофакторної моделі структуризації в управління проєктами сервісів підписки. Управління проєктами та розвиток виробництва. 2024. № 4. С. 33–40.
10. Гриценко П.М. Використання гнучких методологій в управлінні проєктами сервісів підписки. Бізнес Інформ. 2020. № 5. С. 88–94.

Роботу виконано під науковим керівництвом канд. екон. наук, доцента
ТИЩЕНКА Д. О.

АДАПТАЦІЯ ДВОВЕКТОРНОЇ СТРУКТУРИЗАЦІЇ ДО ГНУЧКИХ МЕТОДОЛОГІЙ РОЗРОБКИ МОБІЛЬНОГО ДОДАТКА АДМІНІСТРУВАННЯ КАВ'ЯРНІ

ГУЛЯЙ Д. О., 2 курс 4м група ФІТ ДТЕУ,
освітня програма «Управління проєктами програмних продуктів»

У статті розглянуто проблему адаптації двовекторної структуризації до гнучких методологій розробки (Agile/Scrum) в проєктах створення мобільних додатків для адміністрування кав'ярні. Проаналізовано сумісність двовекторної структуризації та Agile/Scrum, запропоновано рекомендації щодо адаптації, включаючи формування Scrum-команд на основі функціональних векторів та використання матричної структури для візуалізації задач. Розглянуто очікувані переваги (підвищення гнучкості, комунікації, якості та швидкості розробки) та виклики (перепідготовка команди, ризики конфліктів, збільшення адміністративного навантаження).

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

The article examines the problem of adapting a two-vector structuring approach to agile development methodologies (Agile/Scrum) in mobile application development projects for coffee shop administration. The compatibility of two-vector structuring and Agile/Scrum is analyzed, and recommendations for adaptation are proposed, including the formation of Scrum teams based on functional vectors and the use of a matrix structure to visualize tasks. The expected benefits (increased flexibility, communication, quality, and development speed) and challenges (team retraining, risk of conflict, increased administrative burden) are considered.

Актуальність. У сучасному конкурентному середовищі, ефективне адміністрування є ключовим фактором успіху для кав'ярень. Розробка мобільних додатків, що дозволяють автоматизувати та оптимізувати бізнес-процеси, стає все більш актуальною для підвищення ефективності управління, покращення якості обслуговування клієнтів та оптимізації внутрішніх процесів. Мобільні додатки для адміністрування кав'ярень можуть включати функціонал управління замовленнями, обліку продуктів, аналітики продажів, управління персоналом та маркетингових комунікацій. Це дозволяє керівникам та персоналу оперативно приймати обґрунтовані рішення, покращувати взаємодію з клієнтами та оптимізувати витрати.

Проте, розробка таких мобільних додатків є складним проєктом, який вимагає ефективного управління. Зважаючи на динамічний характер ринку мобільних технологій та постійну зміну вимог користувачів, традиційні підходи до управління проєктами, такі як каскадна модель, можуть бути недостатньо гнучкими та адаптивними. В зв'язку з цим, виникає необхідність адаптації структури управління проєктом до гнучких методологій розробки, таких як Agile та Scrum, які дозволяють швидко реагувати на зміни вимог, забезпечувати постійний зворотний зв'язок від користувачів та підвищувати якість кінцевого продукту.

Метою статті є аналіз та адаптація двовекторної структуризації до гнучких методологій розробки мобільного додатку для адміністрування кав'ярні, що забезпечить підвищення ефективності управління бізнес-процесами.

Об'єктом дослідження є розробка мобільних додатків адміністрування кав'ярні.

Предмет дослідження - адаптація двовекторної структуризації до гнучких методологій розробки мобільного додатку.

Аналіз попередніх досліджень. Дослідження, присвячені розробці та впровадженню мобільних додатків для ресторанного бізнесу, зокрема кав'ярень, представлені у кваліфікаційних роботах Дараган та Чалого, які охоплюють питання редизайну та підтримки діяльності кав'ярні за допомогою мобільних технологій [1, 2]. Федорченко розглядає мобільний додаток для обліку замовлень у ресторанному бізнесі [3]. Дєдоша досліджує гнучке управління створенням мобільного застосунку, акцентуючи увагу на онлайн-запису клієнтів у салоні краси, що може бути релевантним для кав'ярень з додатковими послугами [4].

Ткачук та Шестакова вивчають використання мобільних додатків для цифровізації бізнес-процесів у ресторанному бізнесі та оптимізацію на основі A/B тестування, що є важливим для покращення функціоналу додатків [5].

Питання трансформації та розвитку ресторанного бізнесу в умовах пандемії коронавірусу розглядаються Балацькою, Гребенюк, Марценюк, Задоя та Пікуліною, а також Гросулом та Балацькою, які акцентують увагу на digital-маркетингу як інструменті антикризового розвитку. Федосова досліджує використання інформаційно-комунікаційних технологій в ресторанному бізнесі в період пандемії [6-9].

Інформаційно-комп'ютерні інновації в ресторанному бізнесі висвітлені в працях Давидової та Полстяної, Давидової, а також Коваль та Заячковської, які розглядають маркетингові інновації. Тищенко вивчає цифрові технології в індустрії гостинності, а Чміль та Джгуташвілі - цифровізацію управління клієнтським досвідом [10-14].

Виклад основного матеріалу. У сфері розробки програмного забезпечення, особливо в умовах швидких технологічних змін та постійного зростання вимог користувачів, традиційні

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

методології управління проєктами часто виявляються недостатньо гнучкими та адаптивними. У відповідь на ці виклики виникли гнучкі методології розробки (Agile methodologies), що базуються на ітеративному підході, тісній співпраці з замовником та постійній адаптації до змін.

Гнучка методологія (Agile methodology) – це підхід до управління проєктами, який передбачає розбиття проєкту на невеликі, самостійні ітерації (спринти), кожна з яких завершується створенням працюючого прототипу продукту. Agile наголошує на важливості командної роботи, постійному зворотному зв'язку від замовника та адаптації до змін.

Agile-маніфест, розроблений у 2001 році, визначає чотири основні цінності та дванадцять принципів гнучкої розробки. Цінності Agile-маніфесту підкреслюють:

– Людей та взаємодію над процесами та інструментами.
– Працюючий продукт над вичерпною документацією.
– Співпрацю з замовником над узгодженням умов контракту.
– Реагування на зміни над слідуванням плану.

Однією з найбільш популярних та широко використовуваних гнучких методологій є Scrum. Scrum - це фреймворк для управління проєктами, який базується на ітеративній розробці, самоорганізованих командах та постійному зворотньому зв'язку. Scrum передбачає чітко визначені ролі, артефакти та події, що забезпечують прозорість та ефективність процесу розробки.

У Scrum визначено три основні ролі:

– Product Owner (Власник продукту): Відповідає за визначення вимог до продукту, пріоритизацію задач та забезпечення максимальної цінності продукту для замовника.
– Scrum Master (Скрам-майстер): Відповідає за забезпечення правильного застосування Scrum в проєкті, допомагає команді усувати перешкоди та покращувати свою продуктивність.
– Development Team (Команда розробки): Відповідає за безпосередню розробку продукту, самостійно організовує свою роботу та приймає рішення щодо технічної реалізації задач.

Основними артефактами Scrum є:

– Product Backlog (Беклог продукту): Упорядкований список всіх вимог до продукту, відсортований за пріоритетом.
– Sprint Backlog (Беклог спринту): Список задач, які команда планує виконати протягом одного спринту.
– Increment (Інкремент): Працююча версія продукту, створена в результаті виконання задач протягом спринту.

Scrum передбачає наступні події:

– Sprint Planning (Планування спринту): Команда разом з Product Owner визначає ціль спринту та вибирає задачі з Product Backlog для включення до Sprint Backlog.
– Daily Scrum (Щоденний скрам): Короткий (15 хвилин) щоденний мітинг команди, на якому кожен учасник відповідає на три питання: що зробив вчора, що планує зробити сьогодні, чи є якісь перешкоди.
– Sprint Review (Огляд спринту): Демонстрація розробленого інкременту Product Owner та іншим зацікавленим сторонам для отримання зворотного зв'язку.
– Sprint Retrospective (Ретроспектива спринту): Обговорення командою результатів спринту та визначення заходів для покращення процесу розробки.

Застосування Agile/Scrum в контексті розробки мобільних додатків має як переваги, так і недоліки. Перевагами є підвищена гнучкість та адаптивність до змін, покращена комунікація та співпраця між членами команди, швидке виведення продукту на ринок та підвищена якість розробленого додатку. До недоліків можна віднести необхідність високого рівня самоорганізації команди, можливі труднощі з плануванням та контролем великих проєктів, а також

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

залежність від активної участі замовника в процесі розробки. Проте, з урахуванням специфіки ринку мобільних додатків та потреби у швидкій адаптації до змін, гнучкі методології, зокрема Scrum, є ефективним підходом до управління проєктами розробки мобільних додатків, включаючи додатки для адміністрування кав'ярень.

Двовекторна структуризація проєкту являє собою інноваційний підхід до організації розробки мобільних додатків, який базується на одночасному врахуванні функціональних та технологічних аспектів проєкту. Цей підхід дозволяє створити матричну структуру, що забезпечує гнучкість, ефективність та покращену координацію між різними командами розробників.

Рис. 1. Схема функціональних модулів мобільного додатку для адміністрування кав'ярні

Джерело: розроблено автором

Функціональний вектор передбачає розбиття проєкту на окремі функціональні модулі (компоненти) мобільного додатку, які відповідають за виконання певних бізнес-функцій. У контексті мобільного додатку для адміністрування кав'ярні, прикладами таких функціональних модулів можуть бути:

– Управління замовленнями – це модуль, який відповідає за прийом, обробку та відстеження замовлень від клієнтів. Він охоплює всі типи замовлень: на місці, на винос або з доставкою. Завдяки цьому модулю персонал може швидко реагувати на запити клієнтів і контролювати статус виконання кожного замовлення.

– Меню та товари – модуль, що забезпечує зручне створення, редагування та оновлення меню кав'ярні. Через нього адміністратор може додавати нові позиції, змінювати назви, ціни, описи та доступність товарів у режимі реального часу, що дозволяє оперативно реагувати на попит або зміну асортименту.

– Аналітика продажів – це модуль, який надає управлінцю доступ до важливої статистичної інформації про діяльність кав'ярні. Він дозволяє відстежувати обсяги продажів, популярність товарів та поведінку клієнтів, а також формувати звіти для подальшого аналізу ефективності роботи та ухвалення управлінських рішень.

– Управління персоналом – модуль, призначений для організації роботи працівників. У ньому адміністратор може створювати графіки змін, вести облік робочого часу, нараховувати заробітну плату, а також визначати ролі й обов'язки кожного співробітника, що сприяє ефективній взаємодії в команді.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

– Маркетинг і комунікації – модуль, який допомагає взаємодіяти з клієнтами та залучати нових. Він включає функції надсилання push-сповіщень про акції та новини, впровадження програм лояльності, збір і обробку відгуків, а також збереження контактної інформації для подальшої комунікації.

Рис. 2 Технологічна архітектура мобільного додатку

Джерело: розроблено автором

Технологічний вектор передбачає розбиття проєкту на окремі технологічні шари, що відповідають за різні аспекти технічної реалізації додатку. До таких шарів можуть належати:

– UI/UX (User Interface/User Experience): Шар, що відповідає за розробку користувацького інтерфейсу та забезпечення зручності використання додатку.

– Backend API (Application Programming Interface): Шар, що забезпечує взаємодію між мобільним додатком та серверною частиною системи, включаючи обмін даними та виконання бізнес-логіки.

– Database (База даних): Шар, що відповідає за зберігання та управління даними додатку, включаючи інформацію про продукти, замовлення, клієнтів та інше.

– Mobile Platform: Шар, що забезпечує адаптацію додатку до різних мобільних платформ (наприклад, iOS та Android) та враховує специфічні особливості кожної платформи.

Матрична структура проєкту, що виникає в результаті двовекторної структуризації, представляє собою таблицю, в якій рядки відповідають функціональним модулям, а стовпці технологічним шарам. На перетині рядків та стовпців розташовуються конкретні задачі, які необхідно виконати для реалізації відповідного функціонального модуля на відповідному технологічному шарі.

Наприклад, задача «розробка UI/UX для управління замовленнями» буде знаходитися на перетині відповідного рядка та стовпця.

Принципи формування задач та розподілу відповідальності в двовекторній структурі передбачають чітке визначення вимог до кожної задачі, критеріїв її успішного виконання та розподіл відповідальності між різними командами або окремими розробниками. Як правило, функціональна команда (або Product Owner) відповідає за визначення вимог до функціональності, а технологічна команда (або розробники backend, frontend, UI/UX) відповідає за технічну реалізацію. Водночас, обидві команди повинні тісно співпрацювати та обмінюватися інформацією для забезпечення відповідності технічної реалізації функціональним вимогам та забезпечення високої якості кінцевого продукту.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Таблиця 1

Матриця функціональність ↔ технологія для мобільного додатку адміністрування кав'ярні

№	Функціональний модуль	UI/UX	Backend API	Database	Mobile Platform
1	Управління замовленнями	Інтерфейс прийому замовлень	Обробка логіки замовлень	Збереження історії замовлень	Інтеграція з нотифікаціями
2	Меню та товари	Редактор меню	Обробка запитів змін меню	Збереження даних про товари	Адаптація під платформи
3	Аналітика продажів	Візуалізація аналітики	Генерація звітів	Збереження метрик	Платформо-залежна оптимізація
4	Управління персоналом	Графіки змін та ролі	Розрахунок зарплат	База даних працівників	Функції для Android/iOS
5	Маркетинг і комунікації	Форма відгуків	Push API логіка	Зберігання відгуків	Сповіщення за платформами

Джерело: розроблено автором

Матричну структуру проєкту, що виникає в результаті двовекторної структуризації, можна використовувати для візуалізації задач та розподілу відповідальності. Кожен спринт буде включати задачі з різних функціональних модулів та технологічних шарів, а матриця дозволить чітко визначити відповідальність за виконання кожної задачі. Це сприяє покращенню комунікації між членами команди та забезпечує прозорість процесу розробки.

Адаптація ролей Scrum до двовекторної структури потребує певної переосмислення. Product Owner може бути відповідальним за один або кілька функціональних модулів, забезпечуючи представлення інтересів користувачів та визначаючи пріоритети задач. Scrum Master може координувати роботу декількох функціональних команд, допомагаючи їм усувати перешкоди та забезпечуючи відповідність процесу розробки принципам Agile/Scrum. Development Team може складатися з фахівців з різних технологічних шарів, які працюють разом над реалізацією задач для конкретного функціонального модуля.

Для підтримки двовекторної структури в Agile/Scrum можна використовувати різноманітні інструменти, такі як Jira, Trello, Asana та інші. Ці інструменти дозволяють візуалізувати задачі, відстежувати прогрес їх виконання, координувати роботу команди та забезпечувати прозорість процесу розробки. Зокрема, можна використовувати канбан-дошки для візуалізації матричної структури проєкту та відстеження статусу кожної задачі.

На жаль, наразі існує обмежена кількість прикладів практичного застосування адаптованої двовекторної структуризації в проєктах розробки мобільних додатків. Проте, аналіз успішних проєктів, в яких використовувалися гнучкі методології та матричні організаційні структури, показує, що такий підхід може бути ефективним для складних проєктів з великою кількістю учасників та мінливими вимогами. Подальші дослідження та практичні експерименти необхідні для підтвердження ефективності адаптованої двовекторної структуризації в проєктах розробки мобільних додатків для адміністрування кав'ярень.

Адаптація двовекторної структуризації до гнучких методологій, таких як Agile/Scrum, в проєктах розробки мобільних додатків адміністрування кав'ярні, обіцяє ряд значних переваг, але також несе певні виклики та обмеження, які необхідно враховувати при впровадженні цього підходу.

До очікуваних переваг належать, насамперед, підвищення гнучкості та адаптивності проєкту. Поєднання функціонального та технологічного підходів до структуризації дозволяє швидко реагувати на зміни вимог користувачів та ринкових умов, а також ефективно

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

адаптувати технічну реалізацію до потреб бізнесу. Це особливо важливо в динамічному середовищі розробки мобільних додатків, де швидкість виведення продукту на ринок та здатність до постійного вдосконалення є критично важливими.

По-друге, очікується покращення комунікації та співпраці між членами команди. Формування команд на основі функціональних модулів, а не лише на технологічних шарах, сприяє кращому розумінню потреб користувачів та забезпечує більш цілісний підхід до розробки. Матрична структура дозволяє чітко визначити відповідальність за кожну задачу та забезпечує прозорість процесу розробки, що сприяє ефективній комунікації та координації між членами команди.

По-третє, очікується підвищення якості розробленого додатку. Завдяки кращому розумінню потреб користувачів, покращеній комунікації та тісній співпраці між членами команди, розробники мають можливість створювати більш зручні, функціональні та надійні додатки. Постійний зворотний зв'язок від користувачів та можливість швидкого внесення змін дозволяє виправляти помилки та вдосконалювати додаток на основі реальних потреб та відгуків.

По-четверте, адаптація двовекторної структуризації до Agile/Scrum може сприяти прискоренню процесу розробки. Завдяки чіткому розподілу відповідальності, ефективній комунікації та можливості швидкої адаптації до змін, команда може більш ефективно використовувати час та ресурси, скорочуючи терміни розробки та прискорюючи виведення продукту на ринок.

Однак, адаптація двовекторної структуризації до Agile/Scrum також пов'язана з певними викликами та обмеженнями.

Насамперед, існує необхідність перепідготовки команди. Перехід до двовекторної структури вимагає від членів команди оволодіння новими знаннями та навичками, включаючи розуміння принципів Agile/Scrum, вміння працювати в команді, ефективно комунікувати та вирішувати проблеми. Навчання та перепідготовка команди можуть вимагати значних інвестицій часу та ресурсів.

По-друге, можливі конфлікти через складність структури. Матрична структура може бути складною для розуміння та управління, особливо на початкових етапах впровадження. Перетин різних функціональних та технологічних ліній може призвести до конфліктів щодо пріоритетів, розподілу ресурсів та відповідальності. Необхідно приділяти особливу увагу забезпеченню чітких правил та процедур для вирішення конфліктів та підтримки співпраці.

По-третє, існує ризик збільшення адміністративного навантаження. Двовекторна структура вимагає більш детального планування та координації, що може збільшити адміністративне навантаження на керівників проєкту та Scrum Master. Необхідно впроваджувати ефективні інструменти та методи для автоматизації рутинних задач та забезпечення прозорості процесу управління.

Враховуючи вищезазначені переваги та виклики, успішне впровадження адаптованої двовекторної структуризації до Agile/Scrum потребує ретельного планування, чіткого визначення цілей та пріоритетів, ефективної комунікації та підтримки від керівництва організації. Лише за таких умов можна забезпечити максимальну ефективність та досягти поставлених цілей проєкту.

Застосування двовекторної структуризації у поєднанні з гнучкими методологіями розробки, такими як Agile/Scrum, має значний потенціал для підвищення ефективності проєктів розробки мобільних додатків адміністрування кав'ярні. Очікується підвищення гнучкості, покращення комунікації, підвищення якості додатку та прискорення процесу розробки. Однак, успішне впровадження вимагає ретельного планування, перепідготовки команди та ефективного управління потенційними ризиками, пов'язаними зі складністю структури.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Список використаних джерел

1. Дараган Н. С. Редизайн мобільного додатка для мережі кав'ярень «Aroma Kava» : кваліфікаційна робота здобувача освіти першого (бакалаврського) рівня за спеціальністю 022 Дизайн / наук. кер. В. П. Тименко ; рец. А. Шаура. – Київ : КНУТД, 2023. – 83 с.
2. Дєдоша О. Гнучке управління створенням мобільного застосунку для онлайн–запису клієнтів салону краси «Стиль» : пояснювальна записка до кваліфікаційної роботи магістра за спеціальністю 073 «Менеджмент» ОП «Agile-технології розробки програмного забезпечення» / наук. кер.: О. Мушинський, В. Ткаченко ; Університет економіки та права «КРОК». – Київ, 2024. – 74 с.
3. Ткачук В. О., Шестакова А. В. Використання мобільних додатків для цифровізації бізнес-процесів у ресторанному бізнесі та їх оптимізація на основі методу A/B тестування // Економіка, управління та адміністрування. – 2022. – № 4(102). – С. 28–34.
4. Чалий Н. С. Мобільний додаток підтримки діяльності кав'ярні : робота на здобуття кваліфікаційного ступеня бакалавра за спеціальністю 122 – комп'ютерні науки / наук. кер. В. Антипенко ; Сумський державний університет. – Суми, 2024. – 78 с.
5. Федорченко О. В. Мобільний додаток підтримки ведення ресторанного бізнесу. Підсистема обліку замовлень : робота на здобуття кваліфікаційного ступеня бакалавра за спеціальністю 122 – комп'ютерні науки / наук. кер. О. В. Бойко ; Сумський державний університет. – Суми, 2024. – 100 с.
6. Балацька Н. Ю. Ресторанний бізнес в умовах пандемії коронавірусу: проблеми та напрями трансформації моделей розвитку // Інфраструктура ринку. – 2020. – Вип. 42. – С. 117–122.
7. Гребенюк Г. М., Марценюк Л. В., Задоя В. О., Пікуліна О. В. Трансформація та розвиток підприємств ресторанного бізнесу України в умовах пандемії // Інвестиції: практика та досвід. – 2021. – № 7. – С. 14–19.
8. Гросул В., Балацька Н. Digital-маркетинг як дієвий інструмент антикризового розвитку підприємства ресторанного бізнесу в період пандемії та її рецесії // Підприємництво та інновації. – 2020. – № 11/2. – С. 7–12.
9. Давидова О. Ю., Полстяна Н. В. Інформаційно-комп'ютерні інновації в ресторанному бізнесі // Комунальне господарство міст, Сер. Економічні науки. – 2012. – № 106. – С. 403–408.
10. Давидова О. Ю. Формування системи інноваційного управління розвитком підприємств готельно-ресторанного господарства // Бізнес-Інформ. –2017.–№ 11.–С. 459–464.
11. Коваль Л. М., Заячковська Г. А. Маркетингові інновації закладів ресторанного бізнесу // Проблеми системного підходу в економіці. – 2020. – Т. 3, № 2. – С. 128–133.
12. Тищенко С. В. Цифрові технології в індустрії гостинності // Таврійський науковий вісник, Сер. Економіка. – 2021. – Т. 7. – С. 131–139.
13. Федосова К. С. Використання інформаційно-комунікаційних технологій в ресторанному бізнесі в період пандемії Covid-19 // Економіка та суспільство. – 2022. – [Електронний ресурс]. – Режим доступу: https://economyandsociety.in.ua/index.php/journal/article/view/1085/1042.
14. Чміль Г. Л., Джгуташвілі Н. М. Цифровізація управління клієнтським досвідом у готельно-ресторанній індустрії // Бізнес Інформ. – 2020. – № 8. – С. 237–245.

Роботу виконано під науковим керівництвом канд. екон. наук, доцента
ТИЩЕНКА Д. О.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

ВИКОРИСТАННЯ AI-МОДЕЛЕЙ ДЛЯ АНАЛІЗУ РЕЗЮМЕ

ДЕСНОВ П. О., 2 курс 4м група ФІТ ДТЕУ,
освітня програма «Управління проєктами програмних продуктів»

У статті досліджено можливості використання моделей штучного інтелекту для аналізу та вдосконалення резюме з точки зору шукача роботи. Розглянуто сучасні алгоритми штучного інтелекту, зокрема OpenAI Generative Pre-trained Transformer та Anthropic Claude, їх здатність надавати персоналізовані рекомендації щодо покращення резюме та адаптації до конкретних вакансій. Проведено порівняльний аналіз ефективності моделей штучного інтелекту у порівнянні з традиційними методами оцінки резюме. Обговорено переваги автоматизованого аналізу, включаючи підвищення конкурентоспроможності кандидатів на ринку праці.

The article explores the potential of artificial intelligence models for analyzing and improving resumes from the job seeker's perspective. Modern artificial intelligence algorithms, including OpenAI Generative Pre-trained Transformer and Anthropic Claude, are examined for their ability to provide personalized recommendations for resume enhancement and adaptation to specific job vacancies. A comparative analysis of the effectiveness of artificial intelligence models versus traditional resume evaluation methods is conducted. The advantages of automated resume analysis, including increased candidate competitiveness in the job market, are discussed.

Актуальність. У сучасному конкурентному середовищі ринку праці якісне резюме відіграє вирішальну роль у процесі працевлаштування. Проте багато кандидатів стикаються з труднощами при його складанні, не знаючи як правильно структурувати інформацію, які навички варто підкреслити та як адаптувати документ під конкретну вакансію. Традиційні методи аналізу резюме, що передбачають самостійне редагування або звернення до кар'єрних консультантів, можуть бути малоефективними через суб'єктивність оцінки, часові витрати та обмежену доступність професійної допомоги.

Застосування технологій штучного інтелекту та машинного навчання відкриває нові можливості для автоматизованого аналізу резюме, дозволяючи кандидатам отримувати об'єктивні та персоналізовані рекомендації щодо покращення власного документа. Сучасні AI-моделі здатні оцінювати якість резюме, адаптувати його до вимог конкретних вакансій та допомагати у підготовці ефективного супровідного листа. Використання таких технологій сприяє підвищенню шансів шукача на успішне проходження автоматизованих систем відбору та отримання запрошення на співбесіду.

Особливої актуальності набуває дослідження можливостей AI-моделей у контексті їх ефективності порівняно з традиційними методами оцінки резюме. Важливо проаналізувати наскільки такі інструменти можуть підвищити якість підготовки кандидатів та прискорити процес працевлаштування. Це дослідження дозволить краще зрозуміти переваги та обмеження AI-рішень у сфері кар'єрного консультування та автоматизованого підбору персоналу.

Метою статті є дослідження можливостей використання AI-моделей для покращення резюме з точки зору шукача роботи, що у подальшому має вплинути на ефективність процесу пошуку роботи та працевлаштування, а також оцінка ефективності інтелектуального аналізу у порівнянні з традиційними методами.

Об'єктом дослідження є дослідження технології штучного інтелекту, що використовуються для аналізу та вдосконалення резюме, зокрема моделі OpenAI GPT та Anthropic Claude.

Предмет дослідження – функціональні можливості AI-моделей у наданні персоналізованих рекомендацій щодо покращення резюме, а також їх вплив на ефективність пошуку роботи.

Аналіз останніх досліджень і публікацій. Використання штучного інтелекту для аналізу резюме є відносно новою, але перспективною сферою досліджень. Основна увага в науковій

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

літературі приділяється застосуванню AI для автоматизації рекрутингу, оцінки кандидатів та управління людськими ресурсами, тоді як питання персоналізованої допомоги шукачам роботи залишається менш дослідженим.

У сфері автоматизованої оцінки текстових даних важливим напрямом є застосування пояснюваного штучного інтелекту (XAI) для аналізу характеристик користувачів та надання рекомендацій. Наприклад, у дослідженні було розглянуто модель AI для аналізу ризиків відрахування студентів, яка використовує пояснюваний AI та методи гібридного навчання, досягаючи точності прогнозування 96% [4]. Аналогічні підходи можуть бути застосовані до аналізу резюме, щоб визначати його сильні та слабкі сторони.

Також перспективним є використання генеративного AI для автоматизованої адаптації текстів під конкретні вимоги. У рамках дослідження генеративний AI був застосований для автоматизації освітніх матеріалів, що дозволило скоротити час на розробку навчального контенту та підвищити його адаптивність до рівня студентів [2]. У контексті аналізу резюме це може означати автоматичну адаптацію змісту CV до конкретних вакансій або ринкових тенденцій.

Ще одним важливим аспектом досліджень є аналіз етичних та методологічних викликів використання AI в обробці текстових даних. У дослідженні було визначено основні ризики упередженості AI-моделей та запропоновано методи їх мінімізації для забезпечення справедливого аналізу споживчої поведінки [6]. Подібні питання розглядаються і в роботі, де було досліджено методи усунення алгоритмічної дискримінації у біомедичних AI-системах, що може бути використано для забезпечення нейтральності AI у сфері рекрутингу [8].

AI-моделі вже широко застосовуються у сфері HRM, зокрема для прогнозування поведінки кандидатів та автоматизованого прийняття рішень. У дослідженні було проаналізовано вплив AI на процес рекрутингу та доведено його ефективність у підвищенні якості підбору персоналу на 15% [1]. Однак переваги таких систем саме для шукачів роботи залишаються недостатньо вивченими.

Крім того, інтелектуальні системи управління документами демонструють високу ефективність в аналізі текстових даних. Наприклад, у дослідженні було протестовано AI-систему для автоматизованої обробки документів, що забезпечило точність розпізнавання 97% [5]. Подібні підходи можуть бути використані для автоматизованого аналізу та вдосконалення резюме.

Попри значний прогрес у використанні AI у сфері рекрутингу, більшість досліджень зосереджені на інтересах роботодавців та автоматизації процесів відбору. Недостатньо уваги приділяється можливостям AI для безпосередньої підтримки кандидатів у покращенні їхніх резюме. Саме цей аспект залишається нерозкритим і є ключовою проблемою, яку досліджує дана стаття.

Виклад основного матеріалу. Для дослідження було обрано GPT від OpenAI та Claude від Anthropic, оскільки ці моделі є одними з найпотужніших у сфері аналізу тексту, здатні ефективно оцінювати та вдосконалювати резюме, пропонуючи персоналізовані рекомендації для шукачів роботи.

Модель Generative Pre-trained Transformer (GPT) побудована на основі багаторівневої трансформерної архітектури, що використовує механізм самопильнування (self-attention). Основними компонентами є вхідне кодування слів через вбудовані вектори (embedding), багатошарові блоки з нормалізацією (LayerNorm), механізм багатоголової уваги (Multi-head Attention Module) та повнозв'язні шари (Fully-connected). Вихідний шар Softmax використовується для генерації ймовірностей наступного слова. Завдяки цій структурі, візуально зображеній на рис. 1, модель здатна ефективно аналізувати та створювати текстові дані.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

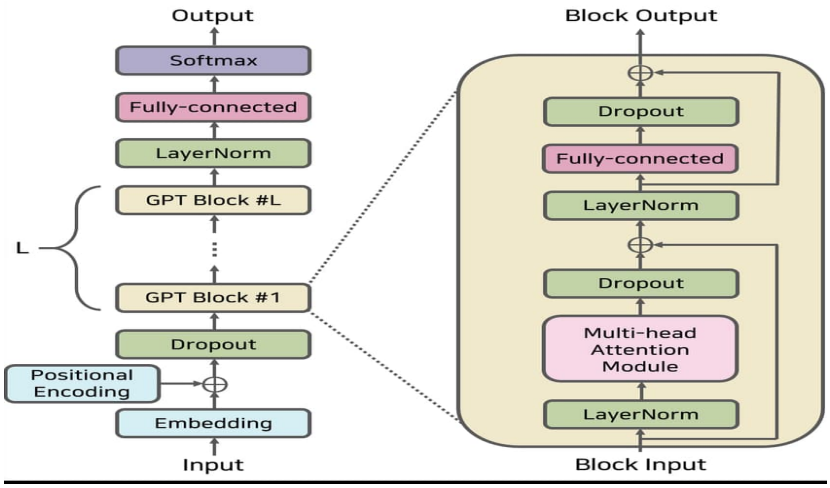


Рис. 1. Архітектура моделі GPT

Джерело: [3]

Claude – це велика мовна модель, розроблена компанією Anthropic, яка використовує підхід Reinforcement Learning from Human Feedback (RLHF) для покращення якості відповідей. Її навчання проходить кілька етапів: спочатку використовується попередньо навчена мовна модель (Pretrained LM), яка проходить етап дистиляції контексту підказок (HHH prompt context distillation) для формування початкової політики моделі (Initial Policy).

Після цього модель оптимізується за допомогою RLHF та алгоритму Proximal Policy Optimization (PPO), що допомагає адаптувати її поведінку відповідно до оцінок людей. Додатково застосовується Preference Model Pretraining (PMP) та Human-Feedback Fine-Tuning, де модель навчається визначати переваги користувачів і коригує свої відповіді на основі отриманого зворотнього зв'язку (рис. 2).

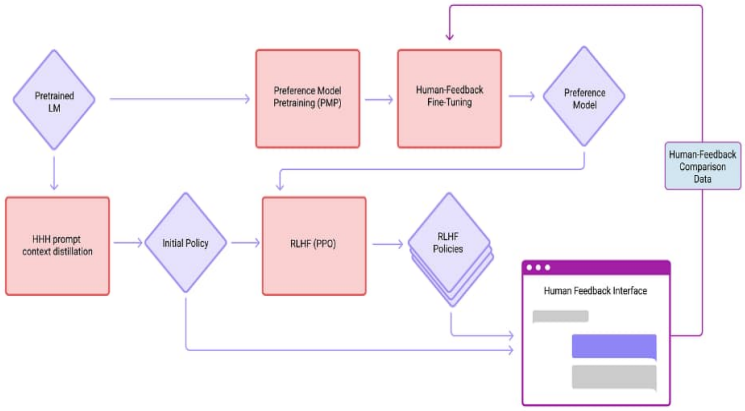


Рис. 2. Архітектура моделі Claude та процес навчання з використанням RLHF та моделювання переваг користувача

Джерело: [7]

Основна мета дослідження полягала у визначенні рівня покращення якості резюме після застосування AI-аналізу, а також оцінки користувацького досвіду при взаємодії з AI. В якості контрольного показника були обрані традиційні методи аналізу резюме, що дозволило об'єктивно оцінити переваги AI-моделей у цьому контексті.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Для дослідження було залучено 40 шукачів роботи, розподілених рівномірно між чотирма професійними сферами: інформаційні технології (IT) – 10 учасників, маркетинг і реклама – 10 учасників, фінанси та бухгалтерський облік – 10 учасників, адміністративні посади – 10 учасників.

На першому етапі дослідження кожен учасник пройшов опитування, у якому оцінював за 100-бальною шкалою 4 показника: якість власного резюме, ефективність використаних раніше методів покращення резюме (виключаючи використання інструментів ШІ), користь відгуків від HR-спеціалістів щодо резюме та рівень задоволення поточним процесом пошуку роботи.

Результати опитування (рис. 3) показали, що середня оцінка якості власного резюме серед учасників була 63 бали, що свідчило про недостатньо високу впевненість кандидатів у представленні свого професійного досвіду. При цьому ефективність використаних раніше методів покращення резюме, таких як самостійний аналіз, консультації з HR-фахівцями або кар'єрними консультантами, отримала середній бал 55, що вказує на загальне невдоволення традиційними методами підготовки резюме.

Користь відгуків HR-спеціалістів щодо структури та змісту резюме була оцінена на 58 балів, що свідчило про певну цінність фідбеку, але водночас його суб'єктивність та неповну корисність для кандидатів. Останній показник – задоволеність поточним процесом пошуку роботи – отримав 49 балів, що вказує на високий рівень незадоволеності серед учасників.

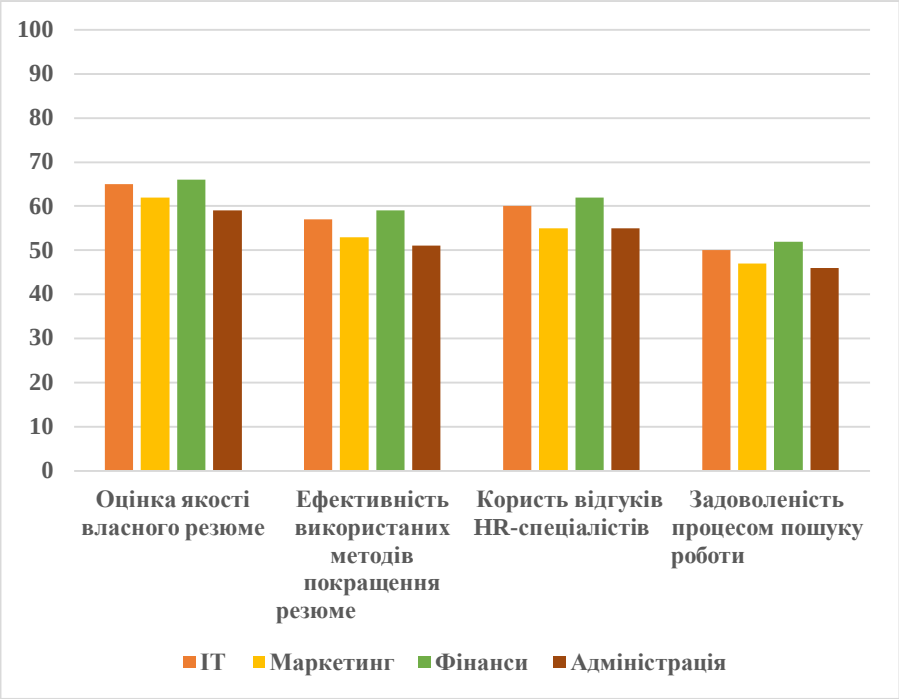


Рис. 3. Результати початкового опитування учасників щодо оцінки резюме та процесу пошуку роботи

Джерело: розроблено автором

Після збору первинних даних учасники завантажили свої резюме до ChatGPT (GPT-4 від OpenAI) та Claude (Anthropic) та отримали індивідуальні рекомендації щодо їх покращення. Для забезпечення однорідності аналізу всі учасники використовували наступний промпт: «Conduct a comprehensive analysis of this resume based on industry best practices and optimization strategies. Identify its key strengths and weaknesses, suggest improvements in wording, structure, and content, and evaluate its alignment with target job postings using keyword relevance

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

analysis. Additionally, provide specific recommendations to enhance clarity, impact, and competitiveness in the job market» («Проаналізуй це резюме та надай детальні рекомендації щодо його покращення відповідно до найкращих практик у галузі. Визнач основні сильні та слабкі сторони, запропонуй зміни у формулюваннях, структурі та змісті. Проведи оцінку відповідності резюме вакансії за ключовими словами та загальною релевантністю. Надай конкретні рекомендації для підвищення конкурентоспроможності кандидата на ринку праці»).

Загалом учасники позитивно оцінили свій досвід роботи з обома ШІ-моделями. Більшість з них зазначили, що отримані рекомендації були чіткими, конкретними та індивідуально спрямованими. Учасники особливо відзначили переваги автоматизованого підбору ключових слів та переформулювання опису навичок, що допомогло більш точно відповідати вимогам вакансій.

Загалом, учасники відзначили, що GPT надавав швидкі й чіткі рекомендації, орієнтовані на конкретні зміни в структурі та формулюваннях (рис. 4). Особливо було оцінено зручність у використанні сервісу та зрозумілість рекомендацій.

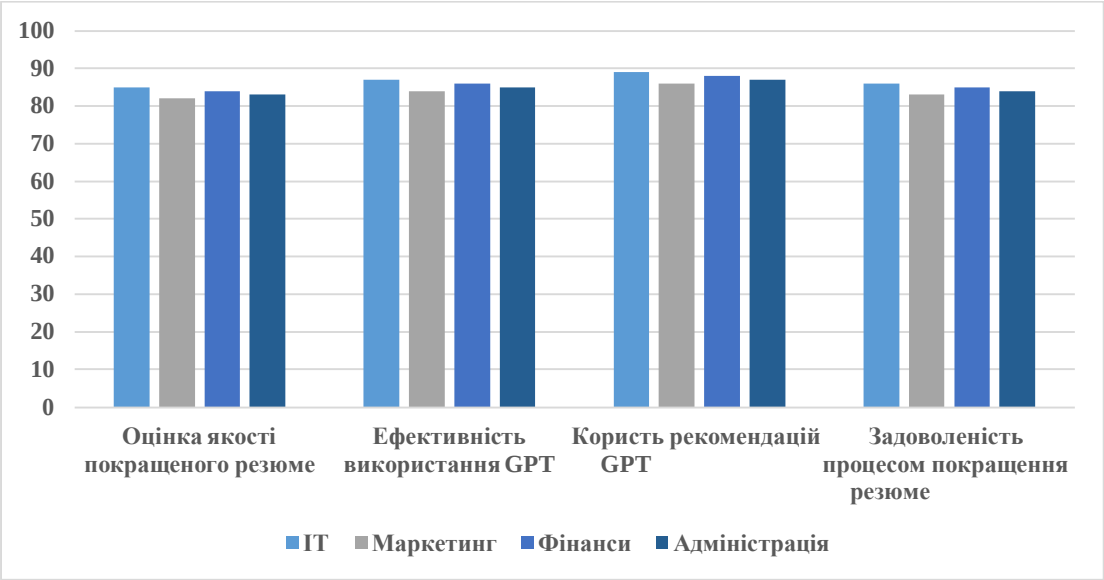


Рис. 4. Результати опитування після аналізу резюме за допомогою моделі GPT (ChatGPT)

Джерело: розроблено автором

Особливо учасниками виділялася ефективність моделі у підборі та оптимізації ключових слів для відповідності резюме вакансіям.

Після аналізу резюме за допомогою моделі Claude було проведено аналогічне опитування учасників. У цьому випадку учасники наголосили на високій персоналізації рекомендацій, глибшому розумінні їхнього професійного досвіду та специфіки вакансій. Також було відзначено, що модель надала більше рекомендацій щодо формулювань, які робили резюме більш індивідуальними та привабливими для рекрутерів (рис. 5).

Учасники експерименту особливо оцінили глибину аналізу та персоналізованість рекомендацій Claude, які дозволяли більш ефективно адаптувати резюме під конкретні вимоги вакансій.

На основі отриманих результатів можна констатувати, що обидві моделі мають свої відмінності у підходах та сильних сторонах: GPT особливо вирізняється швидкістю та зручністю використання, тоді як Claude – глибиною аналізу та персоналізацією рекомендацій. Обидва підходи забезпечили суттєве покращення оцінки резюме з боку кандидатів та позитивно вплинули на процес пошуку роботи в цілому.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

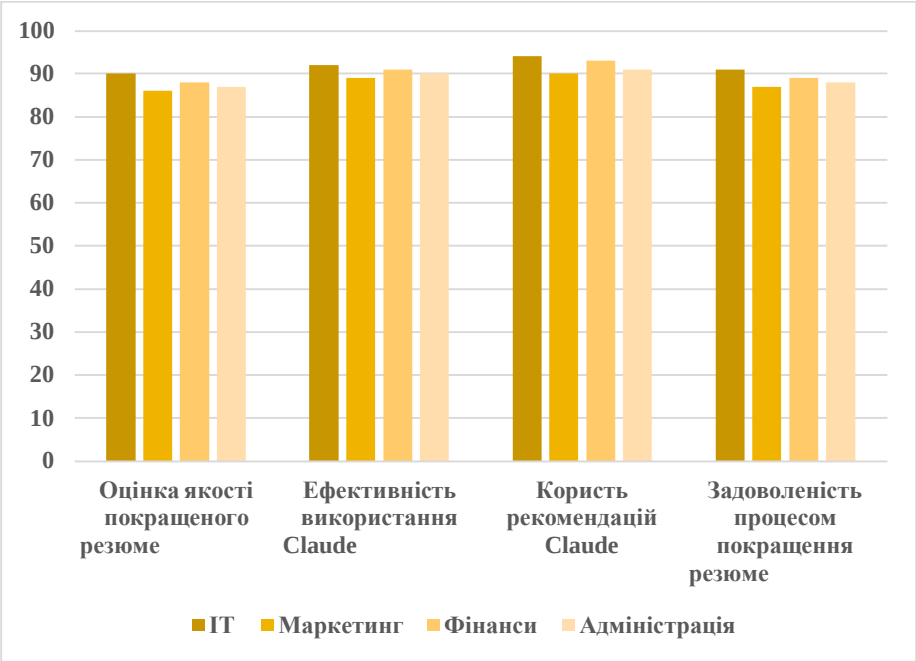


Рис. 5. Результати опитування після аналізу резюме за допомогою Claude

Джерело: розроблено автором

Висновки. Використання AI-моделей для аналізу резюме дозволяє кандидатам значно ефективніше презентувати свої професійні якості, навички та досвід відповідно до вимог сучасного ринку праці. Отримані в ході дослідження результати засвідчили, що автоматизований аналіз документів за допомогою моделей GPT та Claude сприяв підвищенню впевненості кандидатів у власних силах, а також надав можливість отримати об'єктивну та релевантну оцінку свого резюме без залучення сторонніх фахівців, що суттєво скоротило час та фінансові витрати на пошук роботи.

Отримані дані підтвердили, що інтеграція AI-моделей у процес підготовки та аналізу резюме є перспективним напрямом, здатним позитивно впливати на конкурентоспроможність кандидатів на сучасному ринку праці. Подальші дослідження можуть бути спрямовані на вдосконалення алгоритмів персоналізації рекомендацій, розширення їхніх можливостей у контексті різноманітних професійних галузей, а також на вивчення впливу таких технологій на довгострокову кар'єрну успішність шукачів роботи.

Список використаних джерел

1. Gryncewicz W., Zygała R., Pilch A. AI in HRM: case study analysis. Preliminary research // Procedia Computer Science. – 2023. – Т. 225. – С. 2351–2360. DOI: 10.1016/j.procs.2023.10.226.
2. Hla D. A., Hindin D. I. Generative AI & machine learning in surgical education // Current Problems in Surgery. – 2025. – Т. 63. – С. 101701. DOI: 10.1016/j.cpsurg.2024.101701.
3. Lee M. A Mathematical Investigation of Hallucination and Creativity in GPT Models // Mathematics. – 2023. – Т. 11. – С. 2320. DOI: 10.3390/math11102320.
4. Mustofa S., Emon Y. R., Mamun S. B., Akhy S. A., Ahad M. T. A novel AI-driven model for student dropout risk analysis with explainable AI insights // Computers and Education: Artificial Intelligence. – 2025. – Vol. 8. – Article 100352. DOI: 10.1016/j.caeai.2024.100352.
5. Pandey M., Arora M., Arora S., Goyal C., Gera V. K., Yadav H. AI-based Integrated Approach for the Development of Intelligent Document Management System (IDMS) // Procedia Computer Science. – 2023. – Т. 230. – С. 725–736. DOI: 10.1016/j.procs.2023.12.127.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

6. Tiribelli S., Giovanola B., Pietrini R., Frontoni E., Paolanti M. Embedding AI ethics into the design and use of computer vision technology for consumer's behaviour understanding // Computer Vision and Image Understanding. – 2024. – T. 248. – Art. 104142. DOI: 10.1016/j.cviu.2024.104142.

7. Wolfe C. RLAIF: Reinforcement Learning from AI Feedback // Cameron Wolfe's Substack. – 2024. – URL: https://cameronrwolfe.substack.com/p/rlaif-reinforcement-learning-from. (Остання дата відгуку 21.03.2025)

8. Yang Y., Lin M., Zhao H., Peng Y., Huang F., Lu Z. A survey of recent methods for addressing AI fairness and bias in biomedicine // Journal of Biomedical Informatics. – 2024. – T. 154. – Art. 104646. DOI: 10.1016/j.jbi.2024.104646.

Роботу виконано під науковим керівництвом PhD, доцента
ДЕСЯТКО А. М.

ІНТЕГРАЦІЯ МАШИННОГО РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ ДЛЯ ПОКРАЩЕННЯ РОБОТИ ПОШУКОВИХ АЛГОРИТМІВ

КІКОТЬ С. В., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглянуто принципи роботи впроваджених пошукових алгоритмів, наявні можливості їхнього покращення завдяки імплементації машинного розпізнавання та створенню описів до зображень завдяки використанню генеративного штучного інтелекту. Зазначено переваги використання машинного розпізнавання та створення автоматичного переліку тегів. Розроблено приклад інтерфейсу з інтегрованим модулем штучного інтелекту.

The article discusses the principles of operation of implemented search algorithms, the possibilities of their improvement through the implementation of machine recognition and the creation of descriptions for images through the use of generative artificial intelligence. The advantages of using machine recognition and creating an automatic list of tags are noted. An example of an interface with an integrated artificial intelligence module is developed.

Актуальність. За рахунок значного стрибка у точності виконуваних операцій та швидкого розширення можливостей штучного інтелекту, дана сфера інформаційних технологій стала надзвичайно популярною завдяки широкому різноманіттю процесів, які можна спростити або прискорити завдяки імплементації ШІ. На поточний момент робота з покращення генеративного штучного інтелекту триває, на ринку з'являється все більше великих мовних моделей, а ІТ-корпорації впроваджують ШІ до розроблених програмних продуктів не лише з метою покращення або спрощення виконуваних задач, але й для того, щоб програми залишалися актуальними в очах користувачів.

Незалежно від виду зайнятості та сфери роботи, усі користувачі на регулярній основі шукають ті чи інші дані. Сучасні можливості штучного інтелекту дозволяють значно спростити та покращити якість пошуку необхідної інформації, оскільки відповідь на запит

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

буде згенеровано швидко, а суть інформації буде подана стисло та зрозуміло. У якості допоміжного засобу пошуку інформації розробники пошукової системи Google вже інтегрували подібний функціонал у власний сервіс, додавши перед результатами пошуку AI Overview – коротка відповідь на запит користувача на основі знайденої ШІ інформації. Хоча точність даних відповідей не завжди відповідає бажаному рівню достовірності, про що завчасно попереджають більшість моделей, використання ШІ для прискорення та покращення пошуку є доволі ефективним у силу швидкості наданої відповіді.

Дана стаття поєднує дві актуальні на поточний день сфери, вона зосереджена на темі покращення результатів пошукових запитів і розглядає для цього використання різноманітних можливостей штучного інтелекту, а саме розпізнавання змісту зображень, генерація тексту відповідно до наданих інструкцій тощо.

Метою статті є дослідження особливостей пошукових алгоритмів в мережі Інтернет та аналіз можливих ніш для імплементації штучного інтелекту заради підвищення точності результатів пошуку.

Об'єктом дослідження є проектування принципу роботи модулю розпізнавання та автоматичного тегування з інтеграцією генеративного штучного інтелекту.

Предмет дослідження – машинний модуль розпізнавання зображень із системою тегування.

Аналіз попередніх досліджень. Дослідженню поняття машинного розпізнавання та можливих сфер його застосувань присвячені праці вітчизняних та іноземних науковців: Т.В. Холявкіна, Я.О. Резаєв, О.О. Харченко, К. Бішоп та ін.

Виклад основного матеріалу. На даний момент головним фактором якісного пошуку інформації в Інтернеті є сформульований користувачем пошуковий запит. Хоча для більшості побутових запитів користувачам зазвичай вистачає прямого введення свого питання до поля пошуку, інколи для знаходження детальної відповіді на потрібний запит необхідно знати пошукові алгоритми систем та їхній функціонал.

Більшість сучасних систем використовують та надають для більш зручного пошуку затребуваної інформації декілька різних методів та інструментів. Загалом, будь-яка система може виконувати пошук за двома методами. Розглянемо кожну з них більш детально.

Пошук за запитом – система сканує веб-сторінки для знаходження збігів з пошуковим запитом. Введений запит розглядається, як набір слів, і система підбирає сторінки, що містять такі слова. За таким методом реалізовано більшість простих інструментів пошуку, які розміщено на веб-сторінках для швидкого знаходження потрібної користувачі статті або запису.

Семантичний пошук – пошук інформації через наданий користувачем контекст. Даний метод використовують пошукові системи, такі як Google [1]. Особливістю семантичного пошуку є те, що система враховує значення слів та їхній взаємозв'язок (Рис. 1). Даний метод також може враховувати місцезнаходження користувача для видачі більш підходящих результатів, останні події та схожі запити інших користувачів. Хоча семантичний пошук підбирає значно більш релевантні результати, проте через свою складність подібний пошук в основному використовують або пошукові системи, або інші клієнт-серверні програмні продукти, що потребують якісних результатів пошуку.

Обидва методи дозволяють користувачу знайти бажану інформацію. Проте інколи видані результати можуть не відповідати реальним потребам користувача – наприклад, знайдені матеріали стосуються виключно останніх подій, тоді як користувач бажає дізнатися більше про подібно подію кілька десятиліть тому, або ж деякі результати стосуються іншої однойменної сфери і їх необхідно відкинути. Для більш детального налаштування виконуваного пошуку в користувачів є пара інструментів, що допоможуть відфільтрувати знайдені у результаті пошуку матеріали. Розглянемо їх більш детально.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

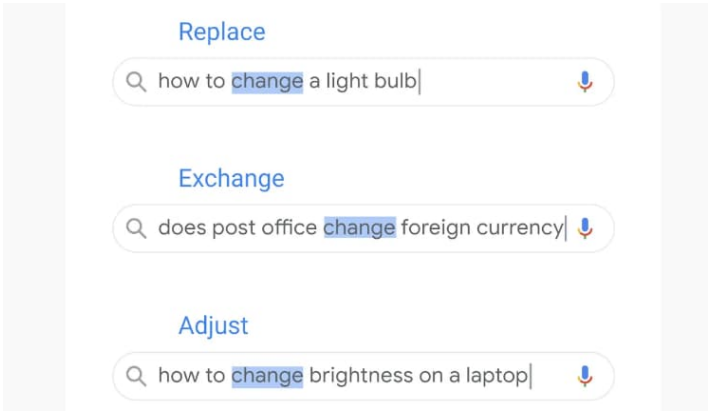


Рис. 1. Особливості семантичного пошуку. Розбір значення слів у запиті

Джерело: [1]

Фільтрація результатів – система надає певну кількість завчасно визначених фільтрів, завдяки яким користувач може відсіяти частину результатів та швидше знайти потрібну інформацію. Наприклад, користувач може шукати виключно зображення або інформацію, що написана конкретною мовою, у певний часовий проміжок або опублікована в певному регіоні.

Пошукові оператори – більшість пошукових систем підтримують використання користувачем операторів, що дозволяють більш точно налаштувати параметри пошуку. З використанням операторів користувач може вказати, що надані результати обов'язково мають містити те чи інше слово або словосполучення, відкинути результати, що містять певний зміст тощо.

Взявши до уваги дані можливості сучасного пошуку, можна дійти висновків, що наразі пошук інформації залежить від вміння користувача формувати правильний запит та бажання використання допоміжних у пошуку інструментів. Проте в умовах сучасного світу зі стрімким розвитком технологій необхідно розуміти, що користувач прагне якомога швидшого і точного результату за якомога більш короткий проміжок часу. Таким чином, сучасні технології пошуку можна спробувати покращити для зниження рівню необхідних знань для виконання якісного пошуку.

Великі IT-корпорації постійно займаються покращенням алгоритмів пошуку для кращого фільтрування результатів, і хоча даний підхід загалом є доволі ефективним, дане дослідження буде зосереджене на покращені результатів пошуку завдяки підходу до проблеми знаходження необхідної інформації з іншого ракурсу.

У той час, як семантичний пошук дозволяє шукати схожу інформацію відповідно до заданого пошукового запиту, він опрацьовує лише інформацію, наданою самою веб-сторінкою, через що не всі підходящі матеріали може бути відображено у якості результату пошуку. Наведемо приклад: якщо користувач шукає гірські краєвиди, проте інший користувач завантажив збірку своїх фотографій з гірськими пейзажами у свій блог з підписом «Мальовничі пейзажі», семантичний пошук навряд відобразить дані світлини серед результатів пошуку, адже пошуковий робот виділить лише слово «пейзажі», що може стосуватися багатьох різних краєвидів: морських, міських, весінніх тощо.

Сам робот не здатен побачити, що саме зображено на викладених фотографіях і може покладатися лише на текст на одній сторінці з зображенням та на «Text Alternative» – текст, який веб-розробники додають як опис до зображення для пошукових роботів та для підвищення доступності сайту, а саме для програм, що зачитують текст веб-сторінки для

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

людей з вадами зору. Хоча наразі все більше компаній та веб-сторінок починають піклуватися про доступність для кожного користувача, далеко не усі розробники займаються покращенням свого продукту у даному напрямку, а для деяких веб-додатків, особливо для тих, що відносяться до соціальних мереж, даний опис фізично не можна додати до кожного зображення, адже кількість матеріалів значно зростає кожного дня. Даний приклад особливо актуальний для тих соціальних веб-сервісів, що зав'язані на перегляді, знаходженні та додаванні власних зображень, як, наприклад, Pinterest. Подальший аналіз даної проблеми буде розглянуто саме на прикладі даного веб-сервісу, проте варто враховувати, що це не єдиний додаток, який працює за подібними алгоритмами.

Наведений веб-сервіс позиціонує себе як місце для швидкого пошуку ідей, рецептів та натхнення [2]. Додаток подає користувачу інформацію у вигляді збірки світлин, тобто масиву візуальної інформації. При використанні внутрішнього пошуку веб-сервіс вишукує слова, згадані користувачем у пошуковому запиті, серед описів до завантажених на платформу зображень, проте користувачі доволі часто не описують детально увесь зміст зображення, а інколи повністю ігнорують дане поле, залишаючи його порожнім.

Хоча платформа містить додатковий пошуковий алгоритм, а саме пошук візуально схожих зображень, даний пошук не є надто точним, а користувач не завжди має серед особистих архівів світлин зображення, що схоже на те, яке він бажає знайти. Для покращення результатів пошуку пропонується запровадити в подібні додатки новий функціонал, що спростить додавання детального опису до зображення, а саме функцію розпізнавання зображення та створення опису до нього за допомогою генеративного штучного інтелекту.

Реалізувати дане впровадження ШІ пропонується наступним чином: при завантажені користувачем зображення штучний інтелект проводить процес розпізнавання змісту зображення, а потім генерує ряд лаконічних та влучних тегів, що коротко описують суть зображеного на світлині. Ці теги далі пропонуються користувачу у якості можливих тегів до зображення, проте у разі, коли користувач вважає, що певний тег не відповідає світлині, він може видалити непотрібні або невідповідні теги.

Запровадження подібної функції має ряд певних переваг, які варто розглянути з кількох різних позицій. Імплементація ШІ дозволить користувачам, що додають свої зображення, не витрачати час на створення детального опису світлини, а також збільшить обсяг тієї аудиторії, що зможуть знайти, побачити, вподобати та прокоментувати завантажене зображення завдяки кращому опису та додатковим тегам; користувачі, які шукають певний контент, зможуть знайти більше відповідних до пошукового запиту зображень; сервіс, який впровадить подібний функціонал, зможе покращити якість свого пошуку, що підвищить рівень задоволеності користувачів додатком.

Також більшість наявних на ринку програмних продуктів додатків мають свою політику щодо контенту, який користувачі не мать завантажувати на платформу. Загалом подібні правила стосуються, наприклад, чутливого контенту, контенту, що пропагує шкідливі звички та залежності або містить будь-які інші небезпечні заклики тощо. Хоча наразі кожна платформа має інструменти для модерації та контролю подібного контенту, наприклад, за допомого машинного розпізнавання зображень, проте даний аналіз контенту на предмет шкідливого вмісту відбувається вже після того, як зображення було завантажено на платформу, де його зможуть побачити інші користувачі, а також поточні моделі машинного розпізнавання, які використовують більшість сервісів, не мають високої точності та часто помиляються під час модерації контенту, видаляючи нешкідливі матеріали випадкових користувачів.

Завдяки впровадженню запропонованого вище функціоналу з автоматичним

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

розпізнаванням та тегуванням зображень, веб-сервіси зможуть не лише замінити стару модель машинного розпізнавання на більш точну та нову, що підвищить якість контролю викладеного на платформі контенту, але й зможуть проводити модерацію зображень перед переходом додатку до етапу оприлюднення світлини для інших користувачів, що значно знизить відсоток того шкідливого контенту, котрий в цілому потрапляє до платформи. Таким чином, від впровадження подібної функції бенефіти отримують як звичайні користувачі, так і компанія, що володіє даним програмним продуктом.

Тегування зображення також може бути здійснене з урахуванням контексту до зображення. Результати генерації пошукових тегів можна спробувати покращити за рахунок обробки не лише завантаженого зображення, але й наданого користувачем опису. Контекст до світлини допоможе ШІ згенерувати більш точні теги: наприклад, якщо користувач написав, що додане ним фото краєвиду було зроблено під час поїздки до Луцьку, штучний інтелект може додати до тегів запис «Краєвиди України» навіть якщо машинне розпізнавання зображення не може точно визначити країну, у якій було зроблено фото. Для програмного впровадження штучного інтелекту достатньо використати API бажаної моделі, що має високий рівень точності розпізнавання зображень та генеративний модуль, що дозволить чітко сформувати теги. Наприклад, для даної цілі можливо використати OpenAI API. Додатковою перевагою використання даного API є наявна детальна документація.

Подальший аналіз нового функціоналу буде розглянуто з боку імплементації нового сегменту в застосунок, а саме до інтерфейсу програмного забезпечення. Для кращого сприйняття користувачами нового функціоналу та для якнайбільш ефективного використання найкращим варіантом подання доданого сегменту буде відображення згенерованих тегів на сторінці після додавання зображення, де користувач редагує назву та опис матеріалу або нового запису. Сегмент інтерфейсу, що відображає згенеровані теги, необхідно виділити кольором та/або стилем, оскільки необхідно привернути до цього фрагменту увагу користувача, аби він переглянув наведені штучним інтелектом теги. Видалення згенерованих тегів має здійснюватися за кілька простих дій, адже якщо користувачу не сподобалися отримані теги або якщо він не бажає долучати до свого зображення згенерований ШІ опис з особистих причин, він мусить мати можливість з легкістю прибрати їх. Також пропонується додати до налаштувань додатку можливість вимкнення генерації тегів при додаванні користувачем нового зображення.

Задля візуальної демонстрації того, як подібний сегмент із згенерованими ШІ тегами може бути інтегровано до користувацької візуальної оболонки готового додаток, було розроблено приклад відповідного інтерфейсу на сторінці редагування зображення, де відображено підібраний ШІ опис до зображення (Рис. 2). Відповідно до наведених у статті рекомендацій, даний сегмент візуально вирізняється серед інших елементів та привертає до себе більше уваги.

Таким чином, проведені у процесі дослідження теми статті дані показують, що наведене рішення є ефективним з боку покращення результатів пошуку мультимедійного контенту, а також не створює технічних ускладнень у процесі впровадження.

Висновки. Проведене дослідження підтверджує ефективність імплементації машинного розпізнавання зображень та генеративного штучного інтелекту для покращення результатів пошуку мультимедійного контенту. Запропонований функціонал з автоматичним тегуванням зображень та попередньою модерацією контенту дозволяє значно зменшити залежність результатів пошуку від людського фактора, підвищити точність пошуку, а також забезпечити вищий рівень контентної безпеки.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

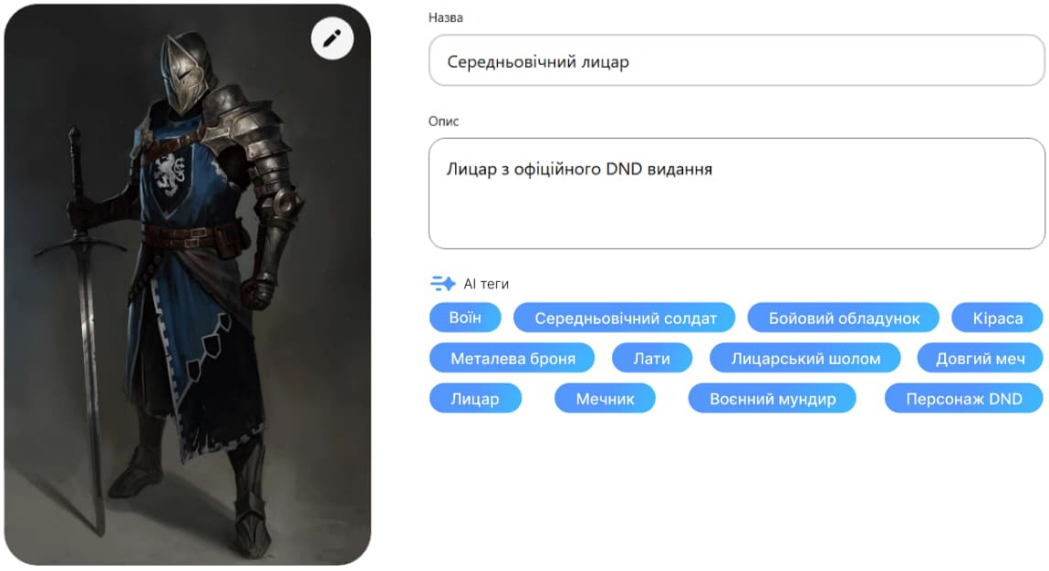


Рис. 2. Приклад вигляду імплементації тегування з використанням ШІ до інтерфейсу сервісу

Джерело: розроблено автором у векторному веб-застосунку Figma (на основі інтерфейсу веб-сервісу Pinterest [2])

З технологічної точки зору, інтеграція такого модуля є відносно нескладною та не потребує радикального перепроєктування вже існуючих систем. Застосування генеративних мовних моделей дозволяє автоматизувати створення коротких описів і тегів до зображень, покращуючи релевантність результатів пошуку та зменшуючи навантаження на кінцевого користувача.

Враховуючи швидкий розвиток ШІ-технологій, подальше вдосконалення таких рішень відкриває нові перспективи для підвищення якості користувацького досвіду в межах цифрових платформ, орієнтованих на обробку візуального контенту.

Список використаних джерел

1. Матеріали Американської публічної транснаціональної корпорації «Google LLC» // Режим доступу: https://www.google.com/intl/en_us/search/howsearchworks/how-search-works/ranking-results (останнє звернення 21.03.2025р.).

2. Матеріали публічної компанії «Pinterest, Inc.» // Режим доступу: https://www.pinterest.com/ (останнє звернення 21.03.2025р.).

Роботу виконано під науковим керівництвом канд. техн. наук, доцента
КОТЕНКО Н. О.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

ЕФЕКТИВНІСТЬ ПОПЕРЕДНЬО НАТРЕНОВАНИХ МОДЕЛЕЙ ГЛИБОКОГО НАВЧАННЯ ДЛЯ ЗАДАЧ РОЗПІЗНАВАННЯ ФОТО-КОНТЕНТУ

КОШОВЕЦЬ С. О., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглянуто ефективність використання попередньо натренованих моделей глибокого навчання у задачах розпізнавання фото-контенту. Проведено порівняльний аналіз моделей VGG16, ResNet50, InceptionV3 та EfficientNetB0 за точністю, швидкістю інференсу та використанням ресурсів. Досліджено доцільність їх застосування в практичних сценаріях. Результати підтвердили переваги моделей нового покоління у поєднанні точності та продуктивності.

The article examines the effectiveness of using pre-trained deep learning models in image content recognition tasks. A comparative analysis of VGG16, ResNet50, InceptionV3, and EfficientNetB0 models is conducted based on accuracy, inference speed, and resource usage. The practical applicability of each model is explored. The results confirmed the advantages of next-generation models in terms of accuracy and performance balance.

Актуальність. У світі цифрових технологій обсяг зображень, що генеруються щодня, продовжує стрімко зростати. Від фотографій у соцмережах до медичних знімків чи відеоспостереження – візуальний контент охоплює дедалі більше сфер. Обробка такого масиву даних вручну є неможливою, тому все частіше вдаються до використання інструментів штучного інтелекту. Зокрема, методи глибокого навчання, що спеціалізуються на виявленні та класифікації об'єктів, демонструють високий рівень точності [1, 2]. Разом з тим, повноцінне навчання таких моделей «з нуля» є дорогим за обчислювальними затратами та потребує значних наборів даних. Це стимулює інтерес до попередньо натренованих моделей, які дозволяють скоротити витрати ресурсів без суттєвих втрат у якості. Оцінка їхньої ефективності в конкретних задачах розпізнавання зображень має вагоме значення не лише з наукового погляду, але й для створення практичних рішень, де важливо обрати оптимальну архітектуру [3, 4].

Аналіз попередніх досліджень. Проблематика розпізнавання фото-контенту за допомогою глибоких нейронних мереж є предметом активного вивчення впродовж останнього десятиліття. Основи сучасних методів були закладені завдяки роботам, присвяченим згортковим нейронним мережам (CNN), зокрема AlexNet (Krizhevsky et al., 2012), яка вперше показала революційні результати на ImageNet. Подальші дослідження привели до появи більш ефективних архітектур, таких як VGGNet, ResNet, Inception, EfficientNet, які широко застосовуються як базові попередньо натреновані моделі [1, 2, 3, 4].

Мета дослідження – визначити ефективність використання попередньо натренованих моделей глибокого навчання для задач розпізнавання фото-контенту, а також провести їх порівняльний аналіз за критеріями точності, швидкодії та здатності до адаптації.

Об'єкт дослідження – процес автоматизованого розпізнавання зображень на основі моделей глибокого навчання.

Предмет дослідження – архітектурні особливості та продуктивність попередньо натренованих нейронних мереж (таких як VGGNet, ResNet, Inception, EfficientNet) при застосуванні до задач класифікації та аналізу фото-контенту [1-4, 6].

Виклад основного матеріалу досліджень. Попередньо натреновані моделі є стандартним інструментом у сучасному комп'ютерному зорі. Вони дають змогу значно скоротити обсяг обчислювальних ресурсів і часу для навчання моделей, при цьому

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

демонструючи високу точність навіть на нових даних [2, 3].

Важливий внесок у розвиток галузі зробили дослідження з transfer learning (перенесення навчання), які дозволяють використовувати моделі, натреновані на великих датасетах, для розв'язання прикладних задач з обмеженою кількістю даних. У наукових публікаціях, зокрема [1], [2], [4], доведено ефективність таких підходів у різних сценаріях: від медичної діагностики до розпізнавання об'єктів у реальному часі.

Незважаючи на значні успіхи, залишаються відкритими питання щодо:

– адаптації моделей до специфічних доменів фото-контенту (наприклад, аерофотозйомка, медичні знімки, зображення в низькій якості);
– оптимального вибору моделі залежно від ресурсів (точність/швидкість/використання пам'яті);
– порівняльного аналізу моделей у контексті конкретних прикладних задач.

Саме цим невирішеним аспектам присвячене дане дослідження – оцінці ефективності попередньо натренованих моделей у задачах розпізнавання фото-контенту з урахуванням їх точності, швидкодії та можливості адаптації до нових наборів даних.

З метою оцінки доцільності використання попередньо натренованих моделей глибокого навчання у задачах розпізнавання зображень, було проведено комплексний аналіз чотирьох поширених архітектур: VGG16, ResNet50, InceptionV3 та EfficientNetB0. Усі моделі застосовувалися в умовах перенесення навчання (transfer learning), що дозволяє адаптувати їх до нових датасетів за рахунок донавчання лише частини шарів.

Вибрані моделі суттєво відрізняються між собою як за структурою, так і за кількістю параметрів. Це дало змогу не лише оцінити точність класифікації, а й з'ясувати співвідношення між продуктивністю й ресурсними витратами. Було встановлено три основні критерії ефективності:

1. Точність класифікації (Top-1 та Top-5);
2. Швидкість інференсу (тобто, скільки часу потрібно на обробку одного зображення);
3. Обсяг пам'яті, необхідний для запуску моделі.

Експерименти проводились на наборі CIFAR-100, який містить 100 категорій зображень у різних класах. Було застосовано transfer learning – попередньо натреновані моделі донавчалися на нових даних [6].

Це дозволило імітувати практичний сценарій, де кількість зображень на один клас є обмеженою – типовий випадок для прикладних задач (наприклад, класифікація дефектів на виробництві чи аналіз зображень медичного призначення).

Попередньо натреновані моделі завантажувалися з бібліотеки TensorFlow Hub. Усі експерименти проводилися в однакових умовах: 10 епох донавчання, оптимізатор Adam, learning rate – 0.0001, batch size – 32. Час інференсу вимірювався на GPU-платформі AMD Radeon RX 7800XT.

Таблиця 1

Результати дослідження

Модель	Top-1 Accuracy	Top-5 Accuracy	Час обробки 1 зображення	Пам'ять (MB)
VGG16	64.2%	86.1%	18.3 мс	530
ResNet50	71.5%	91.7%	12.6 мс	98
InceptionV3	73.8%	93.2%	15.9 мс	92
EfficientNetB0	72.1%	92.4%	8.4 мс	29

Джерело: розроблено автором

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

З отриманих даних можна зробити кілька важливих висновків:

– ResNet50 демонструє найкращий баланс між точністю, швидкодією та помірними ресурсами.

– InceptionV3 дещо випереджає за точністю, однак програє в плані швидкості.

– EfficientNetB0 не поступається ResNet50 в точності, але значно виграє у швидкості й використанні пам’яті – це робить її ідеальним кандидатом для мобільних або вбудованих систем.

– VGG16, незважаючи на історичну значущість, показала найнижчі результати за всіма критеріями – модель морально застаріла для більшості сучасних задач.

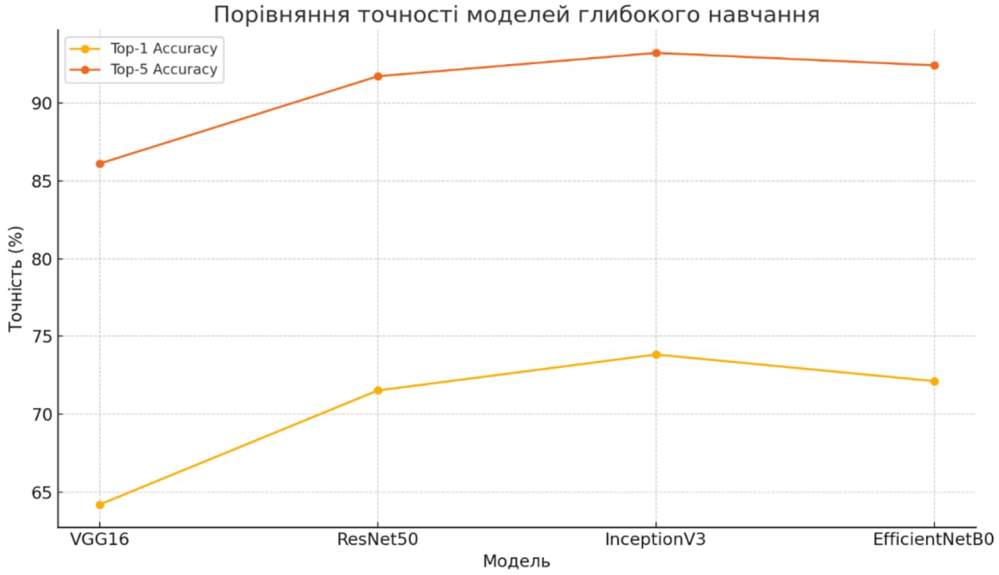


Рис. 1. Порівняння Top-1 і Top-5 точності моделей

Джерело: розроблено автором

Окрім точності, важливим параметром є узагальнювальна здатність моделі – наскільки вона добре працює на зображеннях, що відрізняються від навчального набору. Для цього додатково було протестовано невелику вибірку реальних зображень з доменів, які не входять до CIFAR-100: технічні схеми, медичні скани, зображення з низькою якістю.

Результати підтвердили, що моделі InceptionV3 та EfficientNetB0 найкраще зберігають точність при зміні вхідного контенту, що пов’язано з багаторівневою обробкою ознак та оптимізованою структурою відповідно.

У сучасних проєктах розпізнавання фото-контенту (наприклад, модерація вмісту на вебплатформах, автоматизоване сортування продукції, розпізнавання облич тощо) особливо важливою є здатність моделі швидко адаптуватися до нових класів без повного перенавчання. Попередньо натреновані моделі чудово підходять для цього завдяки глибоким універсальним ознакам, що формуються у перших шарах нейромережі.

Під час тестування зверталась увага на час інференсу та споживання пам’яті. Це критично важливо для застосувань у вбудованих системах, мобільних додатках чи веб-сервісах. Наприклад, EfficientNetB0 забезпечила 92.4% Top-5 точності при мінімальних ресурсних витратах, що дозволяє її використання в реальному часі на слабших пристроях.

Окрім точності класифікації (Top-1, Top-5), важливими показниками ефективності моделей є precision (точність), recall (повнота) та F1-міра – особливо в задачах, де критично важливо зменшити кількість хибних спрацювань (false positives) або пропущених випадків (false negatives).

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Для моделі EfficientNetB0 було обчислено наступні середні значення на валідаційній вибірці:

– Precision: 0.741
– Recall: 0.702
– F1-міра: 0.721

Ці метрики підтверджують, що модель збалансована та здатна не лише влучно класифікувати більшість зображень, а й мінімізує кількість помилок. У випадках, коли вартість помилки висока (наприклад, виявлення хвороби або дефекту), F1-міра може мати більше значення, ніж загальна точність.

Окрім стандартних метрик, важливо оцінити, наскільки легко моделі адаптуються до нових умов чи доменів. Було проведено додаткове тестування на підмножині зображень з атиповими характеристиками: нестандартна освітленість, розмиття, присутність шуму.

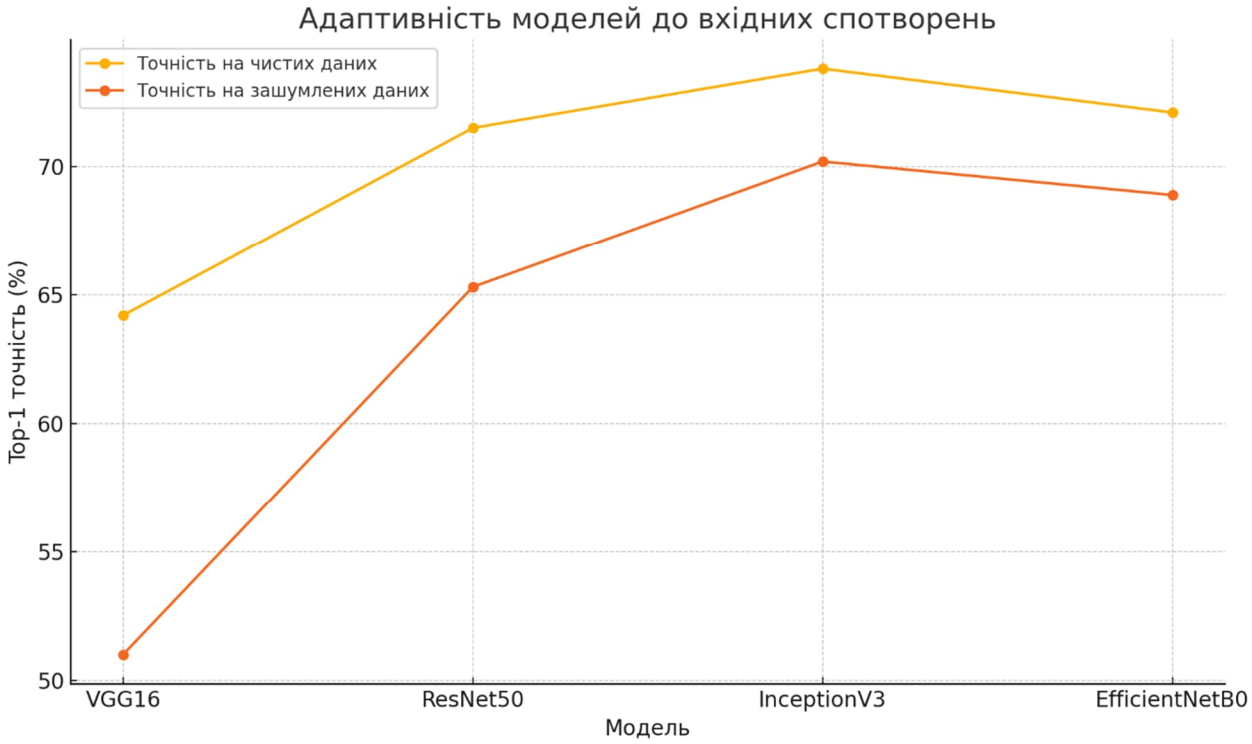


Рис. 2. Адаптивність моделей до вхідних спотворень

Джерело: розроблено автором

Результати:

– InceptionV3 зберегла стабільну точність навіть на зашумлених даних, завдяки своїй структурі, яка дозволяє витягувати ознаки з різних масштабів [3].

– EfficientNetB0 виявилась найгнучкішою у випадку зміни контексту вхідного зображення.

– VGG16, навпаки, втрачає точність при найменших спотвореннях.

Це свідчить про те, що вибір моделі повинен ґрунтуватися не лише на точності на тренувальному наборі, а й на здатності до узагальнення в реальних умовах експлуатації.

Також варто відзначити, що ефективність моделі може залежати не лише від її внутрішньої структури, але й від специфіки зображень. Наприклад, для висококонтрастних технічних фото підійде одна архітектура, а для фотографій з природною варіативністю – інша.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

При створенні систем розпізнавання фото-контенту важливо враховувати етичні та соціальні фактори. Зокрема, моделі можуть бути упередженими щодо певних класів або мати обмежену здатність до інтерпретації рішень. Це особливо актуально у сферах охорони здоров'я або безпеки, де наслідки хибної класифікації можуть бути критичними [7].

Тому при інтеграції ШІ-рішень у реальні системи рекомендується:

– використовувати пояснювальні моделі (Explainable AI);

– проводити тестування на збалансованих датасетах;

– впроваджувати механізми зворотного зв'язку від користувачів.

Особливу увагу варто звернути на EfficientNetB0, яка продемонструвала найбільш збалансоване поєднання точності, швидкодії та економії ресурсів, що робить її доцільним вибором для мобільних пристроїв, вбудованих систем або веб-сервісів. ResNet50 вирізняється стабільністю й універсальністю, а InceptionV3 – найвищими результатами в задачах, де точність критично важлива.

Попри високі результати, дослідження виявило й низку викликів, зокрема:

– залежність точності від специфіки зображень та якості вхідних даних;

– потребу в адаптації моделей до вузькопрофільних задач;

– недостатню оптимізацію моделей під edge-платформи (пристрої з обмеженими ресурсами).

Узагальнюючи отримані наукові результати можна виділити наступні перспективи подальших досліджень у сфері моніторингу ефективності попередньо натренованих моделей глибокого навчання для задач розпізнавання фото-контенту:

– поглиблений аналіз моделей на спеціалізованих датасетах, таких як медичні зображення, супутникові фото або зображення у складних умовах освітлення;

– оптимізація моделей за допомогою pruning (обрізання шарів), quantization (квантування) та distillation (перенесення знань);

– інтеграція моделей у практичні системи, такі як мобільні додатки, системи безпеки чи інструменти автоматичної модерації контенту.

Висновки. У ході дослідження було проведено порівняльний аналіз ефективності попередньо натренованих моделей глибокого навчання для задач розпізнавання фото-контенту. Було встановлено, що сучасні моделі, зокрема InceptionV3, ResNet50 та EfficientNetB0, забезпечують високий рівень точності навіть без повного перенавчання. Попередньо натреновані моделі залишаються потужним і доступним інструментом для реалізації інтелектуальних систем розпізнавання зображень, а їх подальше вдосконалення відкриває нові горизонти в галузі комп'ютерного зору.

Список використаних джерел

1. Simonyan K., Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition // International Conference on Learning Representations (ICLR), 2015. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1409.1556

2. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. – С. 770–778.

3. Szegedy C., Vanhoucke V., Ioffe S., Shlens J., Wojna Z. Rethinking the Inception Architecture for Computer Vision // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. – С. 2818–2826.

4. Tan M., Le Q.V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks // Proceedings of the 36th International Conference on Machine Learning (ICML), 2019. – С. 6105–6114.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

5. CIFAR-100 Dataset. – [Електронний ресурс]. – Режим доступу: https://www.cs.toronto.edu/~kriz/cifar.html

6. TensorFlow Hub: Repository of pre-trained machine learning models [Електронний ресурс]. – Режим доступу: https://www.tensorflow.org/hub

7. Сименюк А.І., Кушнір М.С. Методи комп'ютерного зору в інженерії штучного інтелекту: навч. посіб. – Львів: Видавництво Львівської політехніки, 2020. – 212 с.

Роботу виконано під науковим керівництвом канд. екон. наук, старшого викладача
БУЛЬБИ С. А.

МІЖСЕРВІСНА КОМУНІКАЦІЯ У ВЕБДОДАТКАХ, ПОБУДОВАНИХ НА ОСНОВІ МІКРОСЕРВІСНОЇ АРХІТЕКТУРИ

КРАВЧЕНКО А. Ю., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

Стаття присвячена детальному аналізу проблем міжсервісної комунікації у сучасних веб-додатках, що використовують мікросервісну архітектуру. Розглядаються як синхронні, так і асинхронні підходи, що дозволяють інтегрувати основні сервіси маркетплейсу. Детально описано принципи роботи, переваги, недоліки, а також питання безпеки, масштабованості та відмовостійкості системи. Окрему увагу приділено практичним аспектам інтеграції та оркестрації мікросервісів. Результати дослідження є основою для розробки високопродуктивних та гнучких розподілених систем.

This article provides a detailed analysis of inter-service communication in modern web applications built on microservices architecture. Both synchronous and asynchronous approaches are examined to integrate the main services of a marketplace. The study explains the operating principles, advantages, and limitations, as well as issues regarding security, scalability, and system resilience. Particular attention is given to practical aspects of integration and orchestration of microservices. The findings serve as a foundation for developing high-performance and flexible distributed systems.

Актуальність. Сучасні тенденції в інформаційних технологіях свідчать про перехід до розподілених систем, де кожен компонент виконує окрему функцію. Мікросервісна архітектура стає стандартом для створення гнучких, масштабованих та надійних веб-додатків, зокрема для b2c-маркетплейсів, де кількість транзакцій та обсяг даних стрімко зростають. В умовах високої конкуренції та змінних ринкових вимог ефективна міжсервісна комунікація набуває першорядного значення, оскільки від неї залежить швидкість обробки запитів, безпека та відмовостійкість системи. Таким чином, розробка оптимальних рішень для організації обміну даними між сервісами є важливим завданням як для науковців, так і для практиків.

Метою статті є всебічний аналіз та порівняння існуючих підходів до організації міжсервісної комунікації у веб-додатках, побудованих на основі мікросервісної архітектури. Основна увага приділяється виявленню переваг і обмежень синхронних (REST, GraphQL) та асинхронних (gRPC, Kafka Connect, Kafka) методів взаємодії, з метою обґрунтування вибору оптимальних технологічних рішень для підвищення продуктивності, безпеки та масштабованості системи.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Об'єктом дослідження є архітектурні рішення для організації міжсервісної комунікації у сучасних веб-додатках, зокрема у системах, що використовують мікросервісну архітектуру для реалізації функцій b2c-маркетплейсів.

Предметом дослідження виступають протоколи та технології обміну даними між окремими сервісами, а також їх вплив на характеристики системи, такі як швидкість обробки запитів, латентність, масштабованість і відмовостійкість. Дослідження охоплює як синхронні підходи (REST, GraphQL), так і асинхронні рішення (gRPC, Kafka Connect, Kafka).

Аналіз попередніх досліджень. Для розгляду проблематики міжсервісної комунікації було проаналізовано низку авторитетних досліджень і публікацій. Фундаментальну роль у визначенні основних принципів RESTful архітектури відіграла дисертація Роя Філдінга «Architectural Styles and the Design of Network-based Software Architectures», в якій описано концепції обміну даними через стандартизовані HTTP-протоколи. Далі, для оцінки ефективності мікросервісного підходу було проаналізовано книгу Сема Ньюмена «Building Microservices», яка детально висвітлює переваги ізоляції сервісів, їх незалежне розгортання та гнучку взаємодію у сучасних розподілених системах. Особлива увага приділялася дослідженню систем потокової обробки даних, представленому в роботі Джея Крепса «Kafka: A distributed messaging system for log processing», де детально розглянуто можливості платформи Kafka для забезпечення асинхронного обміну повідомленнями. Крім того, книга «Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems» авторства Мартіна Клеппмана дала змогу оцінити архітектурні підходи до побудови високопродуктивних систем з урахуванням вимог до обробки великих обсягів даних. Офіційна документація компанії Google щодо gRPC і матеріали Facebook про GraphQL дозволили зрозуміти практичні аспекти впровадження цих технологій у сучасних веб-додатках. Аналіз зазначених робіт дозволив визначити сильні та слабкі сторони як синхронних, так і асинхронних методів міжсервісної комунікації, що сприяє обґрунтованому вибору технологічних рішень для розробки розподіленої системи як-от b2c-маркетплейс.

Виклад основного матеріалу. Мікросервісна архітектура дозволяє створювати системи, де кожен компонент може розвиватися і масштабуватися незалежно, що суттєво впливає на загальну продуктивність і відмовостійкість. У цьому контексті важливо правильно обрати методи комунікації, враховуючи як вимоги до швидкості та ефективності, так і потенційні труднощі з підтримкою та оновленням системи.

Розглядати підходи до міжсервісної комунікації, на мою думку, найкраще почати саме з класичних синхронних методів – за допомогою HTTP-протоколів. Найбільш використовуваними підходами є взаємодія в парадигмі REST та за допомогою мови запитів і маніпуляції даними GraphQL.

REST базується на стандартних HTTP-методах (GET, POST, PUT, DELETE) та використанні форматів JSON або XML. Як зазначає Рой Філдінг в своїй дисертації: «ресурси повинні бути представлені у форматі, зручному для взаємодії через стандартні HTTP-методи» [7]. Технічно REST забезпечує простий інтерфейс, проте має певні недоліки, зокрема накладні витрати на кожен HTTP-запит через передачу заголовків та обмеження формату JSON, що може впливати на латентність. Щоб мінімізувати ці проблеми, сучасні рішення використовують HTTP/2, кешування та стиснення даних (наприклад, gzip). Крім того, в процесі розробки можна стикнутись з тим, що REST-ендпоїнти важко перевикористовувати при реалізації нового функціоналу, і для кожного нового покращення на клієнтській частині часто доводиться проектувати новий інтерфейс (ендпоїнт).

Ще одним синхронним методом міжсервісної комунікації є GraphQL. Відносно новий підхід, що з'явився у відкритому доступі в 2015 році, до цього підтримувався і використовувався компанією Facebook для розробки власних продуктів. GraphQL дозволяє клієнтам формувати запити для отримання лише потрібних даних, що значно оптимізує мережевий трафік. Цитуючи офіційну документацію від компанії Facebook: «GraphQL дає змогу клієнту

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

отримувати саме ті дані, які він запитує» [4]. Це значно зменшує обсяг переданої інформації порівняно з REST і дозволяє перевикористовувати сутності для побудови спеціалізованих запитів, що, на відміну вищевказаного методу набагато рідше потребує створення нових інтерфейсів на серверній частині. Однак впровадження GraphQL вимагає ретельного проектування схеми, впровадження контролю складності запитів та застосування механізмів авторизації, щоб запобігти зловживанням та надмірному навантаженню на сервер.

Синхронні підходи тардиційно є меньш продуктивними з точки зору швидкості роботи, втім вони найкраще підходять саме для взаємодії між клієнтом та сервером, адже не вимагають постійного з'єднання й підтримуються усіма сучасними браузерами.

Наступним розглянемо асинхронний підхід до міжсервісної взаємодії. gRPC (Google Remote Procedure Call), який я обрав для взаємодії серверної частини каталогу с сервісом кошику, розроблений компанією Google і є високопродуктивним RPC-фреймворком, який використовує HTTP/2 для передачі даних, забезпечуючи низьку латентність, мультиплексування запитів та підтримку потокової взаємодії. Хоча gRPC використовує HTTP/2 для передачі даних, його асинхронна природа зумовлена переважно архітектурою HTTP/2 та API, які надає сам gRPC. HTTP/2 підтримує мультиплексування – можливість одночасно обробляти кілька запитів і відповідей через одне з'єднання, що дозволяє виконувати неперервну асинхронну передачу даних. Крім того, gRPC надає асинхронні клієнтські та серверні інтерфейси, що дозволяють виконувати виклики без блокування потоку виконання. Це забезпечує можливість обробляти запити паралельно (наприклад, за допомогою стрімінгових викликів: клієнтських, серверних чи двосторонніх), що і робить gRPC асинхронним, незважаючи на те, що воно працює через HTTP/2. Дослідження, проведене Резою Асадуллаї показує, що використання gRPC дає приріст швидкості від 10% до 50% відсотків. Особливо різниця помітна у застосунках, що працюють у багатопоточному режимі а також, при тестуванні з об'ємними даними.

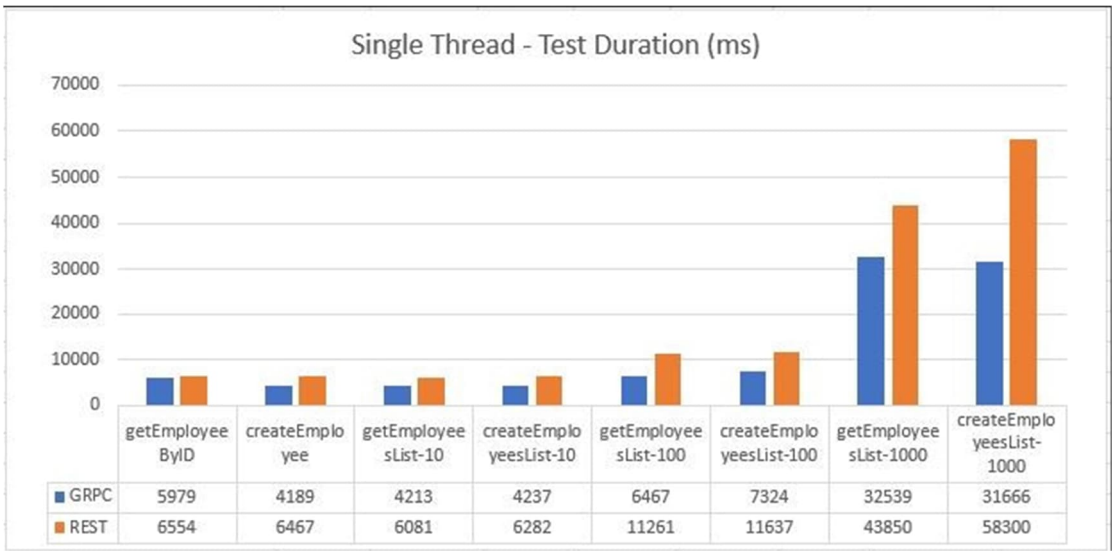


Рис. 1. Порівняння REST і gRPC в однопоточному режимі.

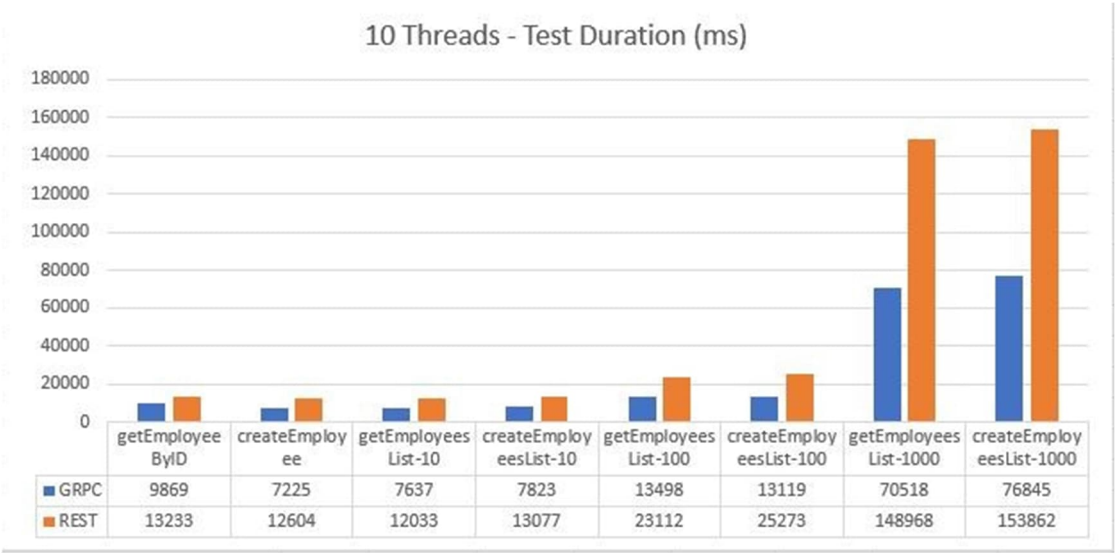
Джерело: [6]

Однією з ключових технологій, на якій базується gRPC, є Protocol Buffers – ефективний інструмент для серіалізації даних.

Protocol Buffers (protobuf) – це бінарний формат обміну даними, який дозволяє визначити чітку схему повідомлень за допомогою мови опису інтерфейсу (IDL). Основні переваги Protocol Buffers включають:

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

– *Компактність і ефективність*. Бінарне кодування даних забезпечує значно менший розмір повідомлень у порівнянні з текстовими форматами, такими як JSON чи XML. Це дозволяє зменшити накладні витрати на мережевий трафік та підвищити швидкість обробки.



	getEmployeeById	createEmployee	getEmployeesList-10	createEmployeesList-10	getEmployeesList-100	createEmployeesList-100	getEmployeesList-1000	createEmployeesList-1000
GRPC	9869	7225	7637	7823	13498	13119	70518	76845
REST	13233	12604	12033	13077	23112	25273	148968	153862

Рис. 2. Порівняння REST і gRPC в багатопоточному режимі

Джерело: [6]

– *Мовонезалежність*. Протокол підтримує численні мови програмування (Java, C++, Python, Go, C#, тощо), що робить його ідеальним для розподілених систем, де сервіси можуть бути написані на різних мовах.

– *Строга схема та сумісність.* Завдяки використанню чітко визначених схем, Protocol Buffers забезпечують сильну типізацію даних та підтримують backward/forward сумісність. Це дозволяє розвивати схему повідомлень без ризику порушення роботи існуючих клієнтів чи серверів.

– *Продуктивність*. Бінарне серіалізоване представлення дозволяє швидко кодувати та декодувати дані, що є критичним для систем, де кожна мілісекунда має значення, наприклад, для транзакційних операцій у сервісі кошика.

Як підкреслює Google, «gRPC дозволяє зменшити накладні витрати за рахунок використання компактного бінарного формату» [5], що безпосередньо впливає на зменшення затримок та підвищення пропускної здатності системи. Завдяки цим характеристикам Protocol Buffers стають ідеальним вибором для побудови високопродуктивних, масштабованих та надійних розподілених систем.

Таким чином, використання gRPC у поєднанні з Protocol Buffers забезпечує ефективну міжсервісну комунікацію, що мінімізує накладні витрати, знижує латентність та сприяє оптимальній продуктивності системи. Це робить даний підхід особливо корисним для сценаріїв, де потрібна висока швидкість обробки даних, як, наприклад, у сервісі кошика, де критично важливо забезпечити низький час відповіді, адже це має прямий вплив на конверсію.

Втім, на відміну від кошика, для сервісу сповіщень та індексації я обрав інший інструмент, а саме Kafka і створений на її основі фреймворк Kafka Connect.

Apache Kafka – це розподілена стрімінгова платформа, або ж брокер повідомлень, яка була спочатку розроблена в LinkedIn і згодом передана до Apache Software Foundation. Її створили з метою забезпечення високопродуктивної, масштабованої та відмовостійкої обробки великих потоків даних у реальному часі. Серед творців проекту – Джей Крепс, Неха

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Нарходе та Джун Рао, які поставили за мету розв’язати проблеми обробки логів та потокових даних, що виникали у великих організаціях.

Для розуміння як працює Kafka, потрібно знати про основні її сутності і механізми.

Топік (topic) – це логічний канал, у якому зберігаються повідомлення. Він поділяється на кілька партицій, що дозволяє здійснювати паралельну обробку даних і забезпечує масштабованість. Порядок повідомлень гарантується лише в межах однієї партиції.

Продюсер (producer) відповідає за генерацію та відправлення повідомлень до певного топіка. Він може використовувати різні стратегії для вибору партицій (наприклад, за ключем повідомлення), що допомагає рівномірно розподілити навантаження і зберегти порядок обробки.

Споживач (consumer) підписується на топіки для зчитування повідомлень. Споживачі можуть працювати як окремі інстанції або входити до складу споживчих груп (consumer groups). У групі кожен споживач отримує лише частину повідомлень, що забезпечує горизонтальне масштабування і балансування навантаження.

Kafka Connect, натомість, є фреймворком, побудованим на основі Kafka, який спрощує інтеграцію Kafka з зовнішніми системами, такими як бази даних, пошукові системи (наприклад, Elasticsearch, який і використовується в спроектованому мною b2c-маркетплейсі), сховища даних тощо. Основними відмінностями Kafka Connect від “базової” Kafka можна визначити:

Мета та призначення. Звичайна Kafka фокусується на передачі повідомлень між продюсерами і споживачами, забезпечуючи високу продуктивність і масштабованість. Kafka Connect спеціально розроблена для побудови конекторів (source/sink), які автоматизують процес імпорту/експорту даних між Kafka і зовнішніми системами.

Управління даними та інтеграція. У звичайній Kafka розробник повинен писати власні програми-споживачі та продюсери для інтеграції з іншими системами. Kafka Connect надає готові до використання конектори, які дозволяють без додаткового коду підключатися до зовнішніх джерел або цільових систем. Наприклад, Elasticsearch Sink Connector автоматично зчитує повідомлення з Kafka-топіка, застосовує трансформації (SMT) та записує дані в Elasticsearch.

Спрощення розробки та підтримки. Kafka Connect автоматично управляє offset-ами, відновленням після збоїв та масштабуванням, що значно знижує складність розробки інтеграційних рішень. Звичайна Kafka вимагає ручного впровадження цих механізмів через розробку окремих продюсерів і споживачів.

Конфігурація та керування. Kafka Connect дозволяє централізовано управляти налаштуваннямитак званих конекторів, що спрощує їх розгортання та оновлення в масштабних системах. Це корисно, коли потрібно інтегрувати кілька зовнішніх систем без розробки унікальних рішень для кожної.

Отже, для сервісу індексації, який інтегрує дані з Kafka в Elasticsearch, Kafka Connect є ідеальним рішенням, оскільки вона забезпечує автоматичну трансформацію та інтеграцію даних, спрощує керування offset-ами та мінімізує витрати на розробку. Для сервісу сповіщень, де основне завдання полягає в оперативній обробці повідомлень, достатньо стандартної Kafka, адже на відміну від процесу індексації, де потрібно лише відправити повідомлення з інформацією про товар і трансформувати його у потрібний для пошукового рушія формат, сервіс сповіщень міститиме значно складнішу логіку і буде матиме інтегреації зі сторонніми ресурсами для надсилання повідомлень користувачам шляхом СМС-повідомлень, електронних листів і push-сповіщень. Тобто стандартної логіки споживача Kafka Connect з SMT трансформером тут буде недостатньо.

В нижченаведених таблицях я структурував вимоги до якісних показників типів міжсервісної взаємодії і порівняв обрані підходи.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Таблиця 1

Вимоги до роботи мікросервісів та обрані протоколи комунікації

Мікорсервіс	Тип взаємодії	Основні вимоги	Обраний протокол/підхід
Основний сервіс каталогу та кабінету	Синхронна (клієнт-сервер)	Швидка відповідь, гнучкість запитів, підтримка різноманітних клієнтських додатків	REST та GraphQL
Сервіс кошика	Синхронна внутрішня комунікація	Низька латентність, висока швидкість транзакцій, надійність операцій	gRPC
Сервіс індексації товарів	Асинхронна	Потокова обробка даних, оперативне оновлення індексу, висока пропускна здатність	Kafka Connect
Сервіс сповіщень	Асинхронна	Масштабованість, обробка великої кількості повідомлень, висока відмовостійкість	Kafka

Джерело: розроблено автором

Перша таблиця підсумовує вимоги до роботи кожного мікросервісу із зазначенням обраного протоколу. REST та GraphQL використовуються для клієнтського інтерфейсу завдяки простоті та гнучкості, gRPC забезпечує низьку латентність для транзакційних операцій, а Kafka Connect і Kafka – для обробки потокових даних. Друга таблиця порівнює ключові технічні характеристики кожного підходу, що дозволяє вибрати оптимальне рішення для конкретних сценаріїв.

Таблиця 2

Показники обраних підходів для міжсервісної комунікації

Підхід	Латентність	Продук-тивність	Масштабо-ваність	Складність впровадження	Область застосування
REST	Середня (HTTP/1.1, покращено HTTP/2)	Помірна	Висока	Низька	Клієнтський інтерфейс, стандартні CRUD-операції
GraphQL	Середня	Середня	Помірна–висока	Середня-висока	Вибіркове отримання даних, оптимізація мережевого трафіку
gRPC	Низька (завдяки HTTP/2 та Protocol Buffers)	Висока	Висока	Середня	Транзакційні операції між сервісами (наприклад, операції в сервісі кошика)
Kafka Connect	Низька (оптимізована за налаштуванням)	Висока	Дуже висока	Середня	Потокова інтеграція даних, оновлення індексів у режимі реального часу (інтеграція Kafka з Elasticsearch)

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Підхід	Латентність	Продук-тивність	Масштабо-ваність	Складність впровадження	Область застосування
Kafka	Низька (залежить від конфігурації)	Висока	Дуже висока	Середня	Обробка великої кількості повідомлень, системи сповіщень із високою відмовостійкістю та масштабованістю даних

Джерело: Розроблено автором

Кожен підхід має свої потенційні труднощі. REST може страждати від високих накладних витрат через HTTP-запити; для їх мінімізації використовують HTTP/2, кешування та стиснення даних. GraphQL вимагає контролю складності запитів, що забезпечується впровадженням аналізу запитів та обмежень глибини. gRPC, попри свою високу продуктивність, ускладнює дебагінг через бінарний формат, тому необхідне трасування і чітке версіонування API.

Висновки. Проведений аналіз демонструє, що вибір міжсервісної комунікації має базуватися на конкретних вимогах системи. Синхронні методи, такі як REST і GraphQL, забезпечують зручний обмін даними між клієнтом і сервером, тоді як асинхронні рішення – gRPC, Kafka Connect та Kafka – дозволяють досягти високої продуктивності та масштабованості при внутрішніх операціях. Особлива увага, приділена використанню Kafka Connect для індексації товарів, показує, що автоматизація інтеграції з Elasticsearch зменшує складність розробки та підтримки, а стандартна Kafka забезпечує ефективну обробку сповіщень без надмірного навантаження. Подальші дослідження можуть бути зосереджені на розробці гібридних моделей взаємодії, удосконаленні механізмів моніторингу та масштабування, а також впровадженні розумних алгоритмів маршрутизації запитів для забезпечення максимальної продуктивності розподілених систем.

Список використаних джерел

1. Клеппман М. Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems. / Мартін Клеппман. – [Б. м.] : O'Reilly, 2017. – 614 с.
2. Крепс Дж. Kafka: a Distributed Messaging System for Log Processing [Електронний ресурс] / Джей Крепс, Нея Нархеде, Джун Рао. – Режим доступу: http://notes.stephenholiday.com/Kafka.pdf (дата звернення: 10.03.2025).
3. Ньюман С. Building Microservices: Designing Fine-Grained Systems / Сем Ньюман. – 2-ге вид. – [Б. м.] : O'Reilly. – 612 с.
4. Офіційна документація GraphQL [Електронний ресурс] // GraphQL | A query language for your API. – Режим доступу: https://graphql.org/learn/ (дата звернення: 10.03.2025).
5. Офіційна документація gRPC [Електронний ресурс] // gRPC.io. – Режим доступу: https://grpc.io/docs/ (дата звернення: 10.03.2025).
6. Асадуллаї Р. Performance Comparison of gRPC and REST over TLS [Електронний ресурс] / Реза Асадуллаї. – Режим доступу: https://www.linkedin.com/pulse/performance-comparison-grpc-rest-over-tls-reza-asadollahi (дата звернення: 10.03.2025).
7. Філдінг Р. Т. Architectural Styles and the Design of Network-based Software Architectures : Дисертація / Філдінг Рой Томас. – Ірвайн, 2000.

Роботу виконано під науковим керівництво канд. екон. наук, доцента
ПАЛАГУТИ К. О.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

ПЕРСОНАЛІЗАЦІЇЯ ІНТЕРНЕТ-МАГАЗИНІВ ІЗ ВИКОРИСТАННЯМ ШТУЧНОГО ІНТЕЛЕКТУ

МАНЧЕНКО М. І., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглянуто роль штучного інтелекту в персоналізації інтернет-магазинів, зокрема основні технології, що використовуються для покращення користувацького досвіду. Описано застосування рекомендаційних систем, обробки природної мови, комп'ютерного зору та машинного навчання для вдосконалення взаємодії з клієнтами.

The article examines the role of artificial intelligence in personalization of online stores, in particular the main technologies used to improve the user experience. It describes the use of recommender systems, natural language processing, computer vision and machine learning to improve customer interaction.

Актуальність. Електронна комерція набуває постійних обертів й стрімко розвивається з року в рік. З таким ж успіхом поширюється конкуренція й стає дедалі жорсткішою. У час новітніх технології, власники намагаються залучити яку мого більше інновацій й нововведень аби утримати увагу покупців й заохотити їх повернутись повторно. Компанії шукають рішення, які б вирізняли їх поміж інших. Одним із ключових інструментів є штучний інтелект (ШІ). Він дозволяє автоматизувати процеси, аналізувати великі обсяги даних та пропонує персоналізований досвід кожному користувачеві. Це рішення, яке дає перспективний напрямок розвитку у eCommerce. Не зважаючи на вже сильний попит серед постачальників/власників бізнесу, він лише починає набирати обертів.

ШІ здатен аналізувати поведінку користувачів, їхні покупки та взаємодію з сайтом, щоб формувати індивідуальні рекомендації. Це значно підвищує ймовірність конверсії та середній чек замовлення. Також така технологія набагато ефективніше працює на рівні аналізу логістичних даних, прогнозування, мерчандайзингу, маркетингу тощо.

Це також змінює підхід до пошуку товарів в інтернет-магазинах. Системи з обробкою природної мови допомагають користувачам швидше знаходити необхідні товари, а технології комп'ютерного зору дозволяють здійснювати підбір за зображеннями.

Сьогодні штучний інтелект вже не є просто трендом, а перетворився на необхідний елемент успішного онлайн-бізнесу. Його можливості дають змогу компаніям швидше реагувати на зміни ринку, краще розуміти потреби своїх клієнтів і пропонувати їм більш релевантний сервіс. З огляду на швидкість розвитку технологій, використання ШІ в e-commerce продовжуватиме зростати, відкриваючи нові горизонти для персоналізації, автоматизації та оптимізації бізнес-процесів.

Метою статті є підкреслення ролі штучного інтелекту в персоналізації інтернет-магазинів, визначення основних інструментів та аналіз ефективності.

Об'єктом дослідження є процес персоналізації онлайн-торгівлі за допомогою штучного інтелекту, який спрямований на покращення користувацького досвіду, підвищення конверсії та оптимізацію бізнес-процесів.

Предметом дослідження виступають алгоритми машинного навчання, обробки природної мови, комп'ютерного зору та інші технології штучного інтелекту, що використовуються для обслуговування клієнтів та аналізу поведінки користувачів.

Аналіз попередніх досліджень показує, що персоналізація за допомогою штучного інтелекту є однією з найбільш ефективних стратегій для покращення взаємодії з клієнтами. Науковці та практики активно досліджують механізми впливу рекомендаційних систем, що базуються на колаборативній фільтрації та контентному аналізі, на поведінку користувачів і

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

рівень продажів. Крім того, значна увага приділяється впровадженню чат-ботів та голосових асистентів, які дозволяють автоматизувати комунікацію. У сучасних дослідженнях також наголошується на важливості обробки великих даних (Big Data) для прогнозування попиту та покращення таргетованої реклами. Водночас залишається актуальним питання етичності використання персональних даних у системах штучного інтелекту, що потребує подальшого вивчення та вдосконалення правових норм.

Виклад основного матеріалу.

Основні технології ШІ, що використовуються для персоналізації. Сучасні інтернет-магазини використовують штучний інтелект для забезпечення персоналізованого досвіду покупців, а однією з ключових технологій у цьому процесі є рекомендаційні системи. Вони ґрунтуються на аналізі поведінки користувачів та їхніх вподобань, дозволяючи прогнозувати інтереси та пропонувати найбільш релевантні товари. Завдяки використанню алгоритмів машинного навчання такі системи значно підвищують ефективність продажів і покращують залученість покупців.

За статистикою, понад 35% продажів Amazon здійснюється завдяки алгоритмам рекомендацій [1]. Подібні технології широко застосовуються й іншими e-commerce платформами, такими як Netflix, eBay та AliExpress.

Рекомендаційні системи можна умовно розділити на три основні типи: колаборативну фільтрацію, контентно-орієнтовану фільтрацію та гібридні підходи.

Колаборативна фільтрація

Колаборативна фільтрація (Collaborative Filtering) ґрунтується на аналізі дій користувачів та їхніх уподобань [2]. Основна ідея цього підходу полягає в тому, що користувачам із подібними інтересами можуть бути рекомендовані схожі товари. Наприклад, якщо двоє користувачів купили однакові товари, то система може припустити, що їм сподобаються й інші однакові або схожі позиції.

Рис. 1. Колаборативна фільтрація

Джерело: [3]

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Колаборативна фільтрація поділяється ще на два методи:

• Фільтрація на основі користувачів (User-Based Filtering) – ґрунтується на пошуку користувачів зі схожими смаками на товари, які вони купували або оцінювали високо.

• Фільтрація на основі елементів (Item-Based Filtering) – аналізує подібність на основі оцінок різних користувачів і пропонує продукти, які зазвичай купують разом

Контентно-орієнтована фільтрація

Контентно-орієнтована фільтрація (Content-Based Filtering) використовує характеристики самих товарів та уподобання користувача для формування рекомендацій. Алгоритм аналізує текстові описи, зображення, категорії та інші метадані, а потім порівнює їх із попередніми покупками. Наприклад, якщо покупець часто купує наукову фантастику, система запропонує йому схожі книги цього жанру [4].

Рис. 2. Контентно-орієнтована фільтрація

Джерело: [3]

Гібридні підходи

Оскільки згадані методи мають певні недоліки, сучасні системи часто використовують гібридні підходи (Hybrid Recommender Systems). Вони комбінують колаборативну та контентно-орієнтовану фільтрацію, щоб компенсувати слабкі сторони кожного методу[5].

Рис. 3. Гібридна фільтрація

Джерело: [3]

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Гібридні системи можуть працювати за різними принципами:

- Змішування рекомендацій (Mixed Hybrid) – одночасно використовує обидва методи та комбінує їх результати.
- Каскадний підхід (Cascading Hybrid) – спочатку застосовує один метод, а потім уточнює результати за допомогою іншого.
- Метод зваженого об'єднання (Weighted Hybrid) – використовує обидва алгоритми та присвоює їм різні ваги залежно від контексту та якості результатів.

Гібридні системи є найбільш ефективними серед рекомендаційних технологій, адже вони забезпечують точніші та різноманітніші результати, мінімізуючи недоліки кожного окремого підходу. Наприклад, система рекомендацій Netflix базується саме на гібридному підході, що дозволяє їй генерувати персоналізовані пропозиції для мільйонів користувачів.

Таким чином, рекомендаційні системи є важливим компонентом персоналізації в електронній комерції, а їхнє постійне вдосконалення забезпечує ефективніше залучення клієнтів і підвищення рівня продажів. Використання сучасних алгоритмів та технологій штучного інтелекту дозволяє інтернет-магазинам надавати індивідуальний підхід до кожного покупця, що є критично важливим у сучасному конкурентному середовищі.

Виклики та обмеження використання ШІ

Попри значні переваги штучного інтелекту у eCommerce, його впровадження супроводжується рядом викликів та обмежень. Зокрема, основними проблемами залишаються питання захисту персональних даних, високі вимоги до обчислювальних ресурсів, а також ризики, пов'язані з помилками алгоритмів та етичними аспектами їхнього використання. Вирішення цих викликів є важливим завданням для компаній, що використовують ШІ для персоналізації сервісів.

1. Проблеми із захистом персональних даних

Одним із найгостріших питань є збереження конфіденційності персональних даних користувачів. Сучасні базуються на зборі та аналізі великих обсягів інформації, включно з історією покупок, пошуковими запитами, геолокацією та поведінковими факторами. Це створює ризики витоку інформації, її несанкціонованого використання або продажу третім особам.

Законодавчі ініціативи, такі як Загальний регламент захисту даних (GDPR) в Європейському Союзі зобов'язують компанії дотримуватися суворих стандартів у сфері обробки персональних даних [6]. Вони передбачають вимогу отримання згоди користувачів на обробку їхньої інформації та право на видалення особистих даних. Проте навіть попри існуючі нормативні акти, витоки даних у великих компаніях трапляються регулярно. Наприклад, у 2019 році хакери отримали доступ до особистої інформації понад мільйонів користувачів Facebook. що ще раз підкреслює важливість посилення кібербезпеки [7].

2. Високі вимоги до обчислювальних ресурсів

Штучний інтелект, особливо його глибокі нейронні мережі, потребує значних обчислювальних потужностей. Розробка та підтримка сучасних моделей персоналізації вимагає використання потужних графічних процесорів (GPU) та тензорних процесорів (TPU), що може бути фінансово обтяжливим для малого та середнього бізнесу [8].

3. Помилки в алгоритмах та етичні аспекти

Незважаючи на високий рівень автоматизації, алгоритми штучного інтелекту не є безпомилковими. Неправильне налаштування моделей або низькоякісні вихідні дані можуть призводити до помилкових рекомендацій або дискримінаційних рішень.

Етичні аспекти використання ШІ також стосуються питання навмисного маніпулювання користувачами. Деякі інтернет-магазини використовують алгоритми для створення штучного дефіциту товарів або підвищення цін у моменти високого попиту. Це може викликати негативну реакцію з боку клієнтів і регулюючих органів, що спостерігається у випадках антимонопольних розслідувань проти великих технологічних компаній.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Застосування методів штучного інтелекту для персоналізації веб-застосунку інтернет-магазину.

Впровадження технологій штучного інтелекту у веб-застосунок інтернет-магазину сприяє створенню індивідуального користувацького досвіду, що суттєво покращує взаємодію клієнтів із платформою. Інтелектуальні алгоритми аналізують поведінкові патерни, передбачають потреби покупців та формують рекомендації, які сприяють збільшенню рівня задоволеності споживачів та підвищенню прибутковості бізнесу.

Крім того, технології глибокого навчання можуть бути використані для аналізу великих обсягів даних, що дозволяє виявляти приховані патерни в поведінці користувачів. Це відкриває можливості для створення надзвичайно точних моделей прогнозування, які допомагають налаштовувати маркетингові стратегії, таргетувати рекламу та створювати індивідуальні акційні пропозиції, що точно відповідають потребам кожного покупця.

Висновки. Застосування штучного інтелекту в персоналізації інтернет-магазинів відкриває широкі можливості для покращення клієнтського досвіду та підвищення ефективності бізнес-процесів. Інтелектуальні рекомендаційні механізми, автоматизація комунікації, аналіз поведінки покупців та адаптивні стратегії ціноутворення сприяють збільшенню продажів та формуванню лояльної аудиторії. Однак, разом із перевагами, ШІ породжує низку викликів, які потребують уваги та вирішення, щоб гарантувати безпечне та прозоре використання технологій у сфері електронної комерції.

Список використаних джерел

1. Amazon Science. *Amazon Science*. URL: https://www.amazon.science/ (date of access: 26.03.2025).
2. Учасники проектів Вікімедіа. Колаборативна фільтрація – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Колаборативна_фільтрація (дата звернення: 26.03.2025).
3. «Телепатія» нейромережі: як персональні рекомендації допомагають абонентам і операторам | Infomir Blog. *INFOMIR – IPTV Set Top Box manufacturer*. URL: https:// www. infomir.eu/ukr/blog/articles/40-personalized-recommendations/ (дата звернення: 12.04.2025).
4. Канібалізація запитів і як вона впливає на SEO | Медіа Inweb. *Медіа Inweb*. URL: https://theinweb.media/sho-take-kanibalizaciya-zapitiv/ (дата звернення: 26.03.2025).
5. Учасники проектів Вікіпедіа. Гібридна інтелектуальна система – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Гібридна_інтелектуальна_ система (дата звернення: 26.03.2025).
6. General Data Protection Regulation (GDPR) – Legal Text. *General Data Protection Regulation (GDPR)*. URL: https://gdpr-info.eu/ (date of access: 26.03.2025).
7. Редакція Т. У Facebook стався новий витік персональних даних. Постраждало 267 млн юзерів. *ТСН.ua*. URL: https://tsn.ua/svit/u-facebook-stavsya-noviy-vitik-personalnih-danih-postrazhdalo-267-mln-yuzeriv-1463157.html (дата звернення: 26.03.2025)
8. Учасники проектів Вікігедіа. Графічний процесор – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Графічний_процесор (дата звернення: 26.03.2025).

Роботу виконано під науковим керівництвом д-ра техн. наук, професора
ЦЮЦЮРИ М. І.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

ІНТЕРАКТИВНА ПЛАТФОРМА ДЛЯ АУДІО-ВІДЕОКОНФЕРЕНЦІЙ З МОЖЛИВОСТЯМИ РОЗШИРЕННЯ

МАРКОВ В. Б., 2 курс 2мз група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглянуто розробку інтерактивної платформи для аудіо-відеоконференції з інтуїтивно зрозумілим та легким для користувача інтерфейсом. Описані принципи створення архітектури системи, яка дозволяє окремо інтегрувати нові можливості платформи, адаптуючи її під різні потреби користувачів та забезпечувати високу продуктивність системи. Розглянуті технології для створення серверної частини, клієнтської, а також, чому в розробці платформ для широкої аудиторії ключову роль відіграє простий для розуміння інтерфейс програми.

The article examined the development of an interactive platform for audio/video conferences with an intuitively understandable and user-friendly interface. The principles of creating a system architecture are described, which makes it possible to separately integrate new platform capabilities, adapt it to various user needs, and ensure high system performance. The technologies for creating the server side and the client side are considered, as well as why a simple and understandable application interface plays a key role in the development of platforms for a wide audience.

Актуальність. Цифрові технології дедалі більше розповсюджуються у повсякденному житті людей і вже зараз стали невід'ємною частиною. Від маленької програми для купівлі квитків на міський транспорт до великих платформ з різноманітним функціоналом для роботи чи навчання. І чим більше стає система, тим складніше стає її інтерфейс, розрахований на більш досвідчених користувачів та підтриманих думкою компанії, яка розробляє цей софт, що на цей момент потрібно робити саме таким чином. І основана така думка на останньому дослідженні, яке встановило, що зрозумілим для користувача буде саме такий варіант. У результаті користувач отримує сучасний дизайн, але все одно витрачає час, щоб розібратись з функціоналом, як і було з минулим дизайном.

Якщо розробка інтерфейсу користувача завжди буде розвиватися й потребувати нових рішень, то так само розробка систем завжди буде розвиватися й потребувати нових рішень. Можна припустити, що архітектура проєкту, його структура, патерни програмування є таким самим інтерфейсом, тільки вже для розробників цієї системи. Далі можна припустити, що "інтерфейс розробника" має аналогічні проблеми. Бо потрібно йти в крок з останніми дослідженнями, які говорять, як потрібно зробити те чи інше сьогодні. У результаті чого команда розробників може мати певні проблеми з реалізацією нового функціоналу чи підтримкою існуючого, адаптуючи його під нові стандарти.

Це приводить нас до ствердження, що вибір дизайну інтерфейсу, вибір архітектури, вибір технологій є не тільки основою будь-якого проєкту, але й завжди матиме актуальність. Успішна реалізація, підтримка та, за вимоги, розширення функціоналу залежить від зваженого підходу до розробки і оцінюється не за модою, а за доцільністю в межах конкретної задачі.

Метою статті є розгляд поширених проблем при створенні інтерфейсу користувача, виборі архітектури проєкту та виборі технологій для створення проєкту. Також опис побудови платформи для аудіо/відео конференцій з можливістю розширення функціоналу як приклад розробки системи, яка дозволяє уникнути зазначених проблем та забезпечити простоту, гнучкість, продуктивність.

Предмет дослідження це проблеми з інтерфейсом користувача, вибору архітектури проєкту, патернів програмування, вибору технологій і рішенн, які дозволяють зробити систему, зрозумілу ззовні для користувача і зсередини для розробника, роблячи її простою, гнучкою та продуктивною.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Об'єктом дослідження є процес побудови інтерактивної платформи для аудіо-відеоконференцій з можливістю розширення функціоналу, зокрема підхід до дизайну інтерфейсу, вибору архітектури системи, паттернів програмування та технологій.

Аналіз попередніх досліджень. Питання труднощів у користувачів із розумінням інтерфейсу програми, питання розробників у виборі архітектури, патернів, структур та питання вибору технологій розглядались у працях Алі Дареджа, Далбіра Сінгха.

Виклад основного матеріалу. Створення нового продукту починається з його кінцевого бачення. Хто буде користуватися цим продуктом? Які проблеми він вирішуватиме? Який у нього буде основний функціонал? Потім створення нового продукту продовжується його представленням, яким його бачитиме споживач. І вже після цього, в кінці, визначається процес створення та підбираються відповідні інструменти. Цим продуктом може бути будь-що: починаючи від предмета, на якому ви зараз сидите або стоїте, закінчуючи програмою на пристрої, в якій ви читаєте цю статтю. А можливо, навіть починаючи з самої цієї статті, адже процес її написання теж підходить під схему, зображену на рисунку 1 (див. рис. 1).

Рис. 1. Схема створення нового продукту

Джерело: розроблено автором

Після створення продукту, а в деяких випадках ще в процесі створення, людина замислюється над розширенням основного функціонала, щоб зробити створюваний або вже створений продукт зручнішим. Спонукальними факторами для поліпшення створених продуктів чи винаходу нових, або ж збільшення кількості виробляємих продуктів, є зростання конкуренції та зростання потреб людини. Зростання конкуренції та зростання потреб людини виходять з її бажання отримувати не тільки більше, але ще й швидше. Отже, людина перебуває в нескінченному прискоренні, що пояснює скорочення часового розриву між кожним наступним винаходом або створенням нового продукту [1]. Цей феномен відображено в книзі Реймонда Курцвайла, в якій він запропонував "Закон прискорення історичного розвитку" [2].

Існуючі на сьогоднішній день зручності є розширеннями основного функціонала продуктів, що покривають базові потреби людини. Це розширення зручностей можна представити у вигляді зворотної піраміди потреб (див. рис. 2).

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

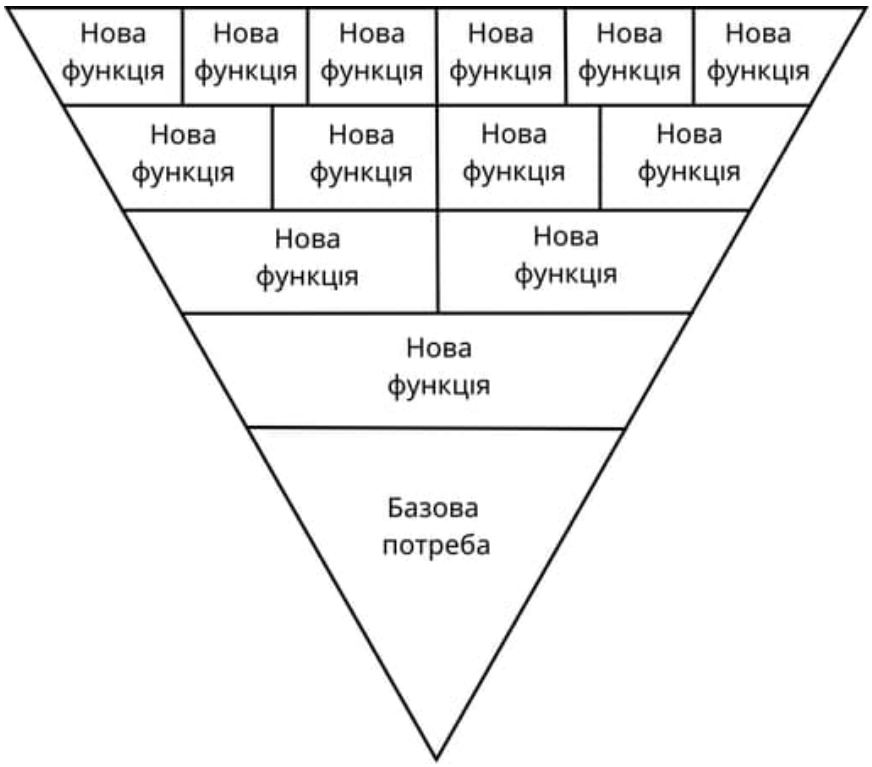


Рис. 2. Зворотна піраміда базової потребності

Джерело: розроблено автором

Однією з базових потреб людини є комунікація. Зі збільшенням населення з'явилася необхідність у комунікації на відстані. Серед із ключових способів зв'язку є телефонний зв'язок, що увійшов у масове використання з появою дротових дискових телефонів. Трохи згодом їх змінили кнопкові телефони, розширюючи свої можливості автовідповідачем, вибором мелодії дзвінка, екраном, визначником номера тощо. Згодом дротовий дисковий телефон став бездротовим, без дискового механізму, будь-яких фізичних кнопок і при цьому "порозумнішав", отримавши назву смартфон. Він об'єднав у собі величезну кількість функцій, включаючи ще один із ключових способів зв'язку – текстові повідомлення. Інтеграція нових можливостей вимагає змін способу взаємодії з телефоном. Так, якщо раніше для дзвінка потрібно було прокрутити барабан телефону або натиснути необхідні кнопки, то зараз, як мінімум, потрібно розблокувати телефон, а потім відкрити програму для набору номера. Або дочекатися, коли тобі зателефонують, якщо не виходить. У цьому прикладі замість телефона можна підставити комп'ютер, адже поняття людино-орієнтованого інтерфейсу зародилося з появою перших комп'ютерів. Або ж можна підставити автомобіль, у якого на самому початку не було даху, не кажучи вже про бічні дзеркала. А зараз у нових автомобілях потрібно скористатися комп'ютером, щоб увімкнути обігрів. Варіантів для прикладу багато, суть не змінюється. Змінюється лише спосіб виконання основних завдань. Однак, проблема зовсім не в нових можливостях і не в їхній кількості, хоча в деяких ситуаціях кількість може бути зайвою. Проблема в їхньому розміщенні, знаходженні простого та зрозумілого представлення доступних можливостей. Зі збільшенням функціонала основні функції повинні залишатися на своїх місцях.

З цього випливає, що людина не повинна бути перевантажена інформацією, обробкою якої в наш час мозок зайнятий просто постійно. І при цьому людина не повинна бути позбавлена функціоналу: можна, наприклад, скласти всі кнопки та крутилки в одну коробку з екраном, але наскільки зручно буде цим користуватися в русі автомобіля, коли з'явиться

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

необхідність увімкнути обдув лобового скла та обігрів керма? Можна також запхати весь комп'ютер у монітор, і він дуже акуратно виглядатиме на столі, його можна буде носити з собою. Це зручно, до цього докуплять фізичну клавіатуру та мишу. Або рішення розробників інтернет-магазину зменшити інформаційне навантаження, запхавши всі розділи в одне випадаюче меню.

Усе це спонукає людей проводити дослідження в області розвитку та застосування користувацьких інтерфейсів. Проводячи різні досліди, паралелі між віком та вподобаннями, сприйнятливістю кольорів та кількістю інформації, рівнем базового розуміння комп'ютера, пишуться нові правила та принципи для дизайну [3]. Тут потрібно бути дуже уважними, оскільки останні правила та принципи дизайну належать скоріше до моди, ніж до практичності, і не завжди мають щось спільне з простотою, гнучкістю та доступністю для розуміння. Але якщо дизайн інтерфейсу розглядати як сферу моди, то можна сказати, що класика завжди буде актуальною та зрозумілою. З цього випливає, що основні функції повинні бути на відстані витягнутої руки, кольори повинні бути нейтральними, а додаткові функції повинні слугувати зручним розширенням, але не навантаженням. На прикладі платформи для аудіо- та відеодзвінків можна запропонувати функцію телефону на самому початку програми, не вимагаючи авторизації або чогось іншого.

Таким чином, основна функція буде доступна відразу. Добре, але це не все. Продукт передбачає розвиток аж до повноцінного робочого середовища, в якому плануються зустрічі, розставляються завдання, створюються файли тощо. Це означає, що основна функція повинна працювати незалежно від розширених можливостей, і ці можливості не повинні бути доступні без авторизації. Кожен додатково інтегрований сервер, своєю чергою, також повинен працювати незалежно від інших сервісів, окрім сервісу авторизації. Це призводить до того, що основна функція є ядром системи, до якого окремо підключаються плагіни, а сервіс авторизації буде точкою входу в ці плагіни. Такий тип поєднує в собі шарову архітектуру (ядро та надбудови) і плагінну (або модульну, оскільки інші сервіси підключаються до ядра як модулі: якщо сервіс «Історія повідомлень» недоступний – система продовжить працювати). А також присутні елементи мікросервісної архітектури. Іншими словами, можна назвати: «Модульно-орієнтована архітектура з ядром та плагіноподібними сервісами» (див. рис. 3).

Необхідну архітектуру було обрано. Це теж добре, але й це не все. Перш ніж перейти до інструментів, потрібно визначитися, як саме створювати. Припустимо, що для розробників є свій користувацький інтерфейс. Ним є патерн програмування, структура проєкту та написання коду. Основними патернами розумно вибрати Facade (ядро надає єдиний інтерфейс для використання внутрішньої логіки, приховуючи складність) та Plugin/Strategy (можливість динамічно підключати та відключати функціональність). Підходами ж будуть User-centric (зручність у центрі: користувач може одразу увійти в дзвінок, не створюючи обліковий запис) та Fail-safe / graceful degradation (якщо якась частина системи не працює (наприклад, історія), основне ядро (дзвінки/чати) продовжує працювати).

Тепер можна перейти до вибору інструментів, за допомогою яких буде створено платформу. З великою ймовірністю, з їхньою ж допомогою платформа буде розширена. І найкращою практикою буде дати кожному інструменту виконувати свою роботу. Навряд чи молоток підійде для того, щоб вкрутити болт. Звичайно, можна взяти JavaScript і написати на ньому застосунки для Android, iOS, Web, Desktop і заодно серверний бік, а можна зробити проєкт грамотно.

Усе це загалом слугує не тільки оптимізації системи, але й комунікації розробників. За цією ж аналогією з інтерфейсом користувача можна говорити про ті самі проблеми: перенасичення всілякими правилами та стандартами, додатковими бібліотеками та фреймворками. Тут також потрібно бути уважним, оскільки популярне та модне не завжди має щось спільне з розумним та гнучким. З цього випливає висновок, що вибір представлення, вибір інструментів та процесу створення повинен диктуватися конкретним завданням та відповідним для нього рішенням, а не останньою статтею великого видавництва.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

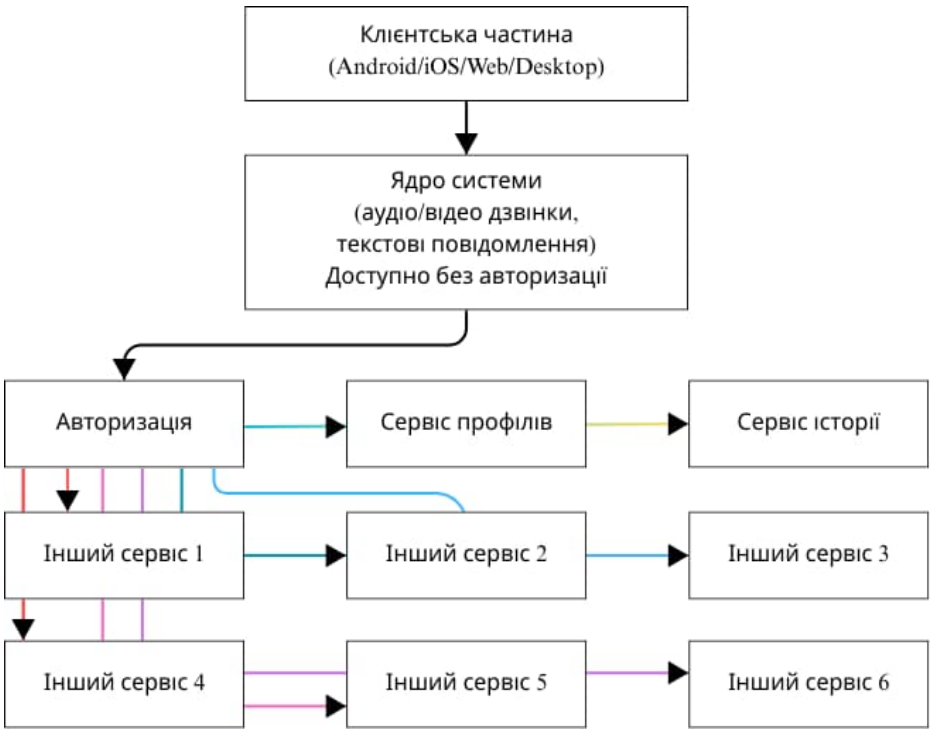


Рис. 3. Модульно-орієнтована архітектура з ядром та плагіноподібними сервісами

Джерело: розроблено автором

Висновки. Кожен створюваний продукт вимагає до себе окремого підходу до реалізації. Наявні правила та шаблони, безсумнівно, допомагають у створенні, проте вони не можуть врахувати всі нюанси процесу розробки. Врахувати ці нюанси повинні безпосередньо розробники продукту, що передбачає, що як мінімум певна частина команди повинна мати гнучкість мислення. Це допоможе скоординувати дії поточної команди, скласти грамотну структуру для підтримки системи майбутньою командою, а також дозволить створити доступний та інтуїтивно зрозумілий інтерфейс для користувача.

Список використаних джерел

1. Прискорення змін. Електронний ресурс. Режим доступу: https://uk.wikipedia.org/wiki/Прискорення_змін (останнє звернення 06.04.2025 р.).

2. Реймонд Курцвайл, The Age of Spiritual Machines / Реймонд Курцвайл // Viking. – 1999. – С. 30–32.

3. A review on user interface design principles to increase software usability for users with less computer literacy. Електронний ресурс. Режим доступу: https://www.researchgate.net/publication/277589616_A_review_on_user_interface_design_principles_to_increase_software_usability_for_users_with_less_computer_literacy (останнє звернення 06.04.2025 р.).

Роботу виконано під науковим керівництвом PhD, доцента
ДЕСЯТКО А. М.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

ПОРІВНЯЛЬНИЙ АНАЛІЗ ІТ-РІШЕНЬ ДЛЯ УПРАВЛІННЯ ОНЛАЙН-РЕПУТАЦІЄЮ БІЗНЕСІВ

МАТУХНО Т. В., 2 курс 4м група ФІТ ДТЕУ,
освітня програма «Управління проєктами програмних продуктів»

У статті розглянуто роль сервісів відгуків у формуванні онлайн-репутації бізнесів. Проаналізовано їхні ключові особливості, на основі існуючих рішень. Окрему увагу приділено проблемі спаму, накрутки рейтингів та методам їхнього виявлення й запобігання. Запропоновано підходи до покращення систем модерації для підвищення достовірності відгуків.

The article considers the role of review services in building online reputation of businesses. The article analyses their key features based on existing solutions. Particular attention is paid to the problem of spam, rating fraud and methods of their detection and prevention. Approaches to improving moderation systems to increase the reliability of reviews are proposed.

Актуальність. У сучасному цифровому світі онлайн-репутація бізнесу є критично важливою для його успіху та конкурентоспроможності. З розвитком інтернет-технологій, особливо після 2010 року, коли в Україні активно почали впроваджуватися онлайн-інструменти для найму працівників та взаємодії з клієнтами, відгуки в мережі стали невід'ємною частиною іміджу бізнесу.

Однак, поряд зі зростанням значення онлайн-репутації, виникла проблема фейкових відгуків. Маніпуляції з позитивними або негативними відгуками можуть суттєво спотворювати реальне сприйняття бізнесу споживачами. Це особливо актуально для тих, чия діяльність значною мірою залежить від інтернет-продажів або надання послуг онлайн.

Основною проблемою є відсутність ефективних механізмів модерації та перевірки достовірності відгуків. Це призводить до того, що споживачі можуть бути введені в оману, а бізнеси – несправедливо втрачати репутацію та клієнтів. Зі зростанням дистанційної роботи та онлайн-найму після пандемії COVID-19, потенційні працівники все частіше звертаються до інтернет-ресурсів для оцінки роботодавців. Старі негативні відгуки, навіть якщо бізнес уже виправив недоліки, можуть продовжувати негативно впливати на його репутацію, оскільки відсутні механізми актуалізації інформації.

Таким чином, дослідження та розробка нових ІТ-рішень для управління онлайн-репутацією бізнесів є надзвичайно актуальними. Необхідно створювати інструменти, які забезпечать прозорість, достовірність та своєчасність відгуків, що сприятиме підвищенню довіри споживачів та партнерів.

Метою статті є дослідження особливостей використання ІТ-рішень для управління онлайн-репутацією бізнесів з метою підвищення ефективності їх функціонування в цифровому середовищі.

Об'єктом дослідження є порівняльний аналіз сучасних інформаційних технологій та сервісів, призначених для збору, аналізу та управління відгуками про бізнес у цифровому середовищі.

Предмет дослідження – сервіси онлайн-відгуків.

Аналіз попередніх досліджень. Дослідженню прямого впливу інтернет сервісів на репутацію бізнесів, основних характерних зв'язків присвячені праці закордонних професорів та студентів: Steven Tadelis, Ines Schulze Horn, Torben Taros, Sven Dirkes, Lucas Hüer, Maximilian Rose та інші [7].

Виклад основного матеріалу. На даний час онлайн-репутація компанії формується двома основними шляхами: через безпосередню діяльність бізнесу та через його публічний імідж, що створюється зовнішніми комунікаціями (рис. 1).

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Рис. 1. Механізм формування репутації бізнесу

Джерело: розроблено автором

З одного боку, бізнес сам закладає основу своєї репутації через якість продукту або послуг, які він пропонує, рівень обслуговування клієнтів, умови праці для співробітників, розмір зарплат і корпоративну культуру. Якщо продукт відповідає очікуванням, сервіс є привітним, а працівники задоволені робочими умовами, компанія автоматично отримує позитивний імідж у своїй цільовій аудиторії.

Однак цього недостатньо, адже сучасний бізнес існує у просторі публічної комунікації та це вже інший бік проблеми. Саме тому компанії активно використовують PR-кампанії, рекламу та соціальні мережі, щоб просувати свій бренд і формувати бажаний образ. Вони намагаються розповісти суспільству, якими вони є насправді або якими хочуть здаватися. Проте суспільна думка завжди виходить за межі офіційних каналів. Люди починають самостійно обговорювати компанію: хтось залишає відгуки на незалежних сайтах, хтось ділиться враженнями у соціальних мережах, а хтось обговорює досвід співпраці на форумах.

Таким чином, репутація компанії є динамічною величиною, що складається як із того, що вона робить всередині, так і з того, як її сприймає суспільство. Якщо компанія працює якісно, це стає основою для позитивного іміджу. Але навіть найкращий продукт чи сервіс можуть не врятувати репутацію, якщо бізнес припуститься комунікаційних помилок або не зможе впоратися з негативними відгуками. Тому управління репутацією – це завжди баланс між реальними діями компанії та позиціонування її у публічному просторі.

У сфері управління репутацією існують два ключові підходи: проактивний і реактивний:

- Проактивний підхід – це коли компанія наперед працює над своєю репутацією, ще до того, як з'являються будь-які зовнішні впливи. Сюди входить створення якісного продукту, забезпечення належного сервісу, турбота про корпоративну культуру та умови праці. Іншими словами, бізнес будує позитивний імідж на основі реальних дій, а не тільки через рекламу чи піар-кампанії.
- Реактивний підхід спрямований на роботу з уже сформованою громадською думкою. Якщо з'являються негативні відгуки чи критика в соцмережах, компанія швидко реагує: виправляє помилки, веде комунікацію, займається кризовим PR. Це важливий елемент репутаційного менеджменту, адже навіть найуспішніший бізнес не застрахований від скандалів або невдоволених клієнтів.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

І тут виникає ключове питання: як саме ці підходи впливають на сприйняття компанії в суспільстві? Якщо бізнес діє лише реактивно, тобто виправдовується або виправляє проблеми постфактум, люди можуть сприймати це як вимушені заходи. У таких випадках створюється враження, що компанія просто «гасить пожежі», а не системно працює над своєю репутацією. З іншого боку, бренди, які приділяють увагу проактивному підходу, зазвичай сприймаються як стабільні, надійні та відповідальні.

Отже, поєднання проактивного та реактивного підходів є ключовим у формуванні сильної репутації. Проте, навіть якщо компанія ретельно працює над своїм іміджем, остаточне слово все одно залишається за суспільством. Те, як саме бізнес керує своєю репутацією, безпосередньо впливає на зміст відгуків: якщо компанія діє проактивно, клієнти частіше залишають позитивні враження, тоді як відсутність належного реагування на проблеми спричиняє негативні коментарі. Саме тому платформи для відгуків стали важливим інструментом, що відображає реальний стан справ та слугує джерелом суспільної думки. Люди довіряють досвіду інших користувачів, а бізнес змушений враховувати цей фактор у своїй стратегії. Тож далі варто розглянути, як саме працюють такі сайти, чому попит на них залишається стабільно високим і яку роль вони відіграють у сучасному інформаційному просторі.

Сайти з відгуками стали важливою частиною цифрового середовища в Україні, оскільки вони виконують роль посередників між споживачами та бізнесом. Ці платформи дозволяють користувачам ділитися власним досвідом щодо товарів, послуг чи навіть умов праці в компаніях. Це створює прозорість, що, в свою чергу, впливає як на репутацію брендів, так і на вибір інших користувачів.

Є кілька різних типів таких платформ. Загальні сайти, як Vidguky.net чи Vidhuk.ua, охоплюють широкий спектр відгуків про різні компанії та послуги, без обмеження по конкретних сферах. Окремо є спеціалізовані ресурси, такі як ua.pravda-sotrudnikov.com, де користувачі можуть ділитися своїм досвідом про роботодавців – чи то про умови праці, заробітну плату, чи корпоративну культуру. Також існують галузеві платформи, наприклад Naprobu.ua, які зосереджуються на відгуках про товари відомих брендів і новинки на ринку.

Більшість таких платформ працюють через систему рейтингів – користувачі можуть оцінювати послугу або товар, зазвичай за шкалою від 1 до 5 зірок, а також залишати текстові відгуки. Деякі сайти додають додаткові методи для перевірки достовірності відгуків, наприклад, через підтвердження реального досвіду користувача або алгоритми для виявлення фейкових рецензій. Це важливо, адже такі платформи не тільки допомагають ділитися правдивою інформацією, але й можуть стати місцем для маніпуляцій – накрутки рейтингів або поширення замовних негативних відгуків з метою завдати шкоди конкурентам.

Вплив таких сервісів на рішення споживачів важко переоцінити. Більшість людей перед покупкою товару або замовленням послуги шукають відгуки в інтернеті, що стало основною практикою у сучасному світі. Позитивні коментарі можуть суттєво підвищити довіру до компанії, збільшуючи її конкурентоспроможність на ринку. Водночас негативні відгуки можуть спричинити серйозні проблеми, навіть репутаційну кризу, якщо бізнес не має ефективної стратегії взаємодії з клієнтами або належної стратегії реагування на критику. У таких випадках компанії ризикують втратити довіру споживачів і зазнати фінансових збитків.

З іншого боку, платформи з відгуками також мають певні труднощі. Проблеми з накруткою відгуків, видалення негативних коментарів за гроші чи організовані атаки на компанії можуть спотворювати реальну картину, створюючи недовіру до самих платформ. Це може призвести до того, що користувачі почнуть сумніватися в достовірності отриманої інформації. Тому сучасні сервіси постійно працюють над вдосконаленням своїх алгоритмів модерації, впроваджуючи додаткові перевірки для боротьби з фейковими відгуками, а компанії шукають нові способи для управління своєю репутацією в цьому відкритому цифровому середовищі, використовуючи аналітику та зворотний зв’язок з користувачами.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

У Таблиці 1 було розглянуто кілька популярних сервісів для відгуків, що активно використовуються на українському ринку. Ці платформи вибрані через їх високий рівень активності користувачів. Вони представляють різні типи відгуків і мають різні спеціалізації, що дозволяє задовольняти потреби широкого кола користувачів.

Таблиця 1

Порівняння сервісів

Назва	Тип сервісу	Необхідність реєстрації	Категорії відгуків
Vidguky.net	Загальний сервіс для відгуків	Необов'язково	Товари, послуги, бренди, магазини
Vidhuk.ua	Загальний сервіс для відгуків	Необов'язково	Житлові комплекси, новобудови, товари, послуги
Vidguk.in.ua	Довідковий сайт	Немає, тільки через ел. пошту компанії	Компанії, послуги, товари
Naprobu.ua	Галузевий сервіс для відгуків	Обов'язково	Продукти, бренди
jobs.dou.ua	Спеціалізований сервіс для відгуків про IT-компанії	Обов'язково + верифікація	IT-компанії

Джерело: згенеровано на основі 1–5

Перед нами є порівняння кількох популярних українських сервісів для відгуків, яке дозволяє детальніше розглянути ключові особливості кожної платформи. Зокрема, варто звернути увагу на три основні аспекти: тип сервісу, необхідність реєстрації для залишання відгуку та категорії відгуків. Тип сервісу та категорії – це ті характеристики, що прямо визначають спрямованість сайту і його функціональні можливості. Кожен сервіс має свої особливості, що зумовлюють різні підходи до збору та представлення відгуків.

Що стосується необхідності реєстрації для залишання відгуку, це цікава тема, оскільки вона безпосередньо впливає на зручність використання платформи та може впливати на кількість та якість відгуків, що залишають користувачі та це варто розібрати більш детально.

Коли користувачі заходять на платформу для того, щоб залишити відгук про товар чи компанію, вони часто мають намір поділитися своїм досвідом, щоб допомогти іншим у виборі чи висловити свою думку. Однак цей процес часто переривається ще на етапі реєстрації. Тут виникає бар'єр, який треба подолати. Навіть зацікавлений користувач може відмовитися від публікації відгуку, якщо для цього потрібно пройти додаткову процедуру реєстрації. Це знайома ситуація для багатьох: ви заходите на сервіс, зацікавлені в його можливостях, але перед вами з'являється форма для створення акаунту, що може відбити бажання продовжити користуватися сайтом. Така перешкода може стати вагомим фактором для того, щоб піти на інший сервіс, який не має цієї реєстрації, навіть якщо він за якістю послуг гірший.

З точки зору користувача, було б ідеально, якби сервіси дозволяли публікувати відгуки без необхідності реєстрації. Це значно спростило б процес і зробило його більш доступним для всіх. Проте існує й інша сторона – технічні обмеження та ризики, з якими стикаються самі сервіси. Відсутність реєстрації на сайті може створити умови для зловживань: спам, накрутка рейтингів або навіть організовані атаки на компанії через фальшиві відгуки. Для недобросовісних користувачів або конкурентів це може стати інструментом маніпуляцій, оскільки відгуки, які не прив'язані до реальних акаунтів, не можуть бути перевірені або верифіковані автоматично, якщо не брати до уваги модерацію. Тому для запобігання таким проблемам реєстрація може стати важливим заходом для збереження чесності та достовірності інформації, що публікується на платформі.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Тепер давайте детальніше розглянемо два конкретні сервіси, де підхід до реєстрації відрізняється не тільки від інших платформ, але й має свої особливості, які варто окремо відзначити. Мова йде про Vidguk.in.ua та jobs.dou.ua, де кожен сервіс вибрав свій, унікальний шлях, що має свої переваги та недоліки в контексті зручності використання для користувачів.

Платформа Vidguk.in.ua не вимагає реєстрації для публікації відгуків, що з першого погляду може здатися досить зручним, оскільки, на папері, користувач може одразу залишити свою думку без необхідності створювати акаунт або надавати особисту інформацію. Однак такий саме їх підхід має нюанс. Для публікації відгуків на цьому сайті користувачам необхідно надсилати їх на електронну пошту платформи, а вже після цього відгуки потрапляють на сайт. Така система, схожа на ті методи, які використовувались у газетах і журналах багато років тому, що може бути не зовсім оптимальним для сучасних користувачів інтернет-сервісів. В епоху, коли швидкість та зручність є критичними для користувачів. Тим більше це може наштовхнути на думку в продажності такого сервісу, адже ні один відгук не проходить поміж модерацію.

На противагу цьому сервіс jobs.dou.ua має більш суворий підхід до реєстрації. Для того, щоб залишити відгук, користувач не тільки повинен створити акаунт, а й пройти процедуру підтвердження особистості. Такий підхід забезпечує високий рівень довіри на платформі. Сам сайт орієнтований на доволі вузьку галузь користувачів, а саме IT-фахівців. Підтвердження особистості дозволяє зменшити ризик накрутки відгуків або поширення фальшивої інформації, що є важливим для підтримки якісного контенту на сайті. Проте цей підхід має і свої недоліки. Наприклад, для сервісів, які мають на меті залучення ширшої аудиторії, така вимога може бути неприємною для частини користувачів, які не хочуть або не готові витрачати час на підтвердження своєї особистості. Така перевірка може зменшити загальну кількість відгуків або навіть відлякати потенційних користувачів, які шукають швидку та легку можливість поділитися своєю думкою.

Таким чином, вибір підходу до реєстрації залежить від типу сервісу, його аудиторії та цілей, які він ставить перед собою. Кожен із цих варіантів має свої переваги та обмеження, і важливо знаходити баланс між зручністю для користувачів і забезпеченням якості та достовірності інформації на платформі. Тож, реєстрація є важливою частиною платформи, але її впровадження повинно враховувати специфіку самого сервісу і потреби його користувачів. Технічний аспект реєстрації сприяє боротьбі з маніпуляціями та фальшивими відгуками, але водночас може створити додаткові перешкоди для тих, хто хоче просто поділитися своїм досвідом.

Що ж можна зробити для покращення ситуації з реєстрацією, щоб вона не відбивала бажання користувачів залишати відгуки, але при цьому забезпечувала належну безпеку та якість контенту? Як знайти баланс між зручністю для користувача і необхідними заходами для запобігання маніпуляціям?

Звісно, реєстрація повинна бути, але її реалізація може бути цікавішою та ненав’язливою. Ідеальний варіант – не нав’язувати користувачу ідею про реєстрацію одразу. Початок процесу має бути простим: користувач заходить на платформу, знаходить компанію, продукт чи сервіс, про який хоче поділитися своїм досвідом, або навіть створює нову публікацію. Потім він переходить до написання самого відгуку, вказує оцінки, плюси та мінуси, і вже коли готовий відправити свій відгук, з’являється останній етап – введення електронної адреси.

Цей підхід має свою психологічну складову: користувач вже витратив час на написання відгуку і, ймовірно, не захоче втрачати його через відмову від реєстрації. Введення адреси на останньому етапі дає можливість завершити процес без великого навантаження, при цьому забезпечуючи певний рівень контролю та перевірки, необхідний для боротьби з фальшивими відгуками.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Існує кілька варіантів для подальших дій. Один з них – автоматичне створення акаунту на основі введеної електронної адреси, з тимчасовим паролем, який браузер збереже автоматично, а сам пароль буде надіслано на електронну пошту. Це дозволить користувачу отримати доступ до акаунту без зайвих зусиль, а також надасть йому можливість змінити пароль у будь-який момент. Інший варіант – підтвердження через код, який надійде на пошту користувача, щоб активувати акаунт. Цей метод є більш безпечним з точки зору боротьби з потенційним спамом чи фальшивими акаунтами, оскільки перевірка електронної пошти перед відправкою відгуку забезпечить додатковий рівень верифікації. Процес зображено за допомогою UML діаграми діяльності (activity diagram) (рис. 2).

Рис. 2. UML діаграма діяльності

Джерело: розроблено автором

Звісно, навіть метод авторизації не зможе повністю захистити від спаму, оскільки акаунти можуть бути створені заздалегідь, і такі користувачі почнуть залишати необ'єктивні та неправдиві відгуки в великих кількостях. Це призведе до спотворення загальної оцінки компанії, продукту чи сервісу. Без належного контролю це може стати серйозною проблемою. Наприклад, на цьому зображенні (Рис.3) можна побачити ситуацію, коли відгуки повторюються з майже однаковим змістом, як «гарна зарплатня» чи «все подобається», і з високими оцінками.

Такі повідомлення часто публікуються в короткий проміжок часу, що є певним сигналом про спам.

Щоб уникнути цього, важливо мати систему, яка аналізує активність відгуків на платформі. Алгоритми можуть відстежувати аномальні сплески, коли коментарі публікуються майже одночасно і містять схожі фрази. Порівнюючи ці періоди з типовою активністю на сайті, можна виявити підозрілі хвилі. Однак постає питання, як не втратити реальні відгуки, які також можуть з'являтися в ці моменти.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

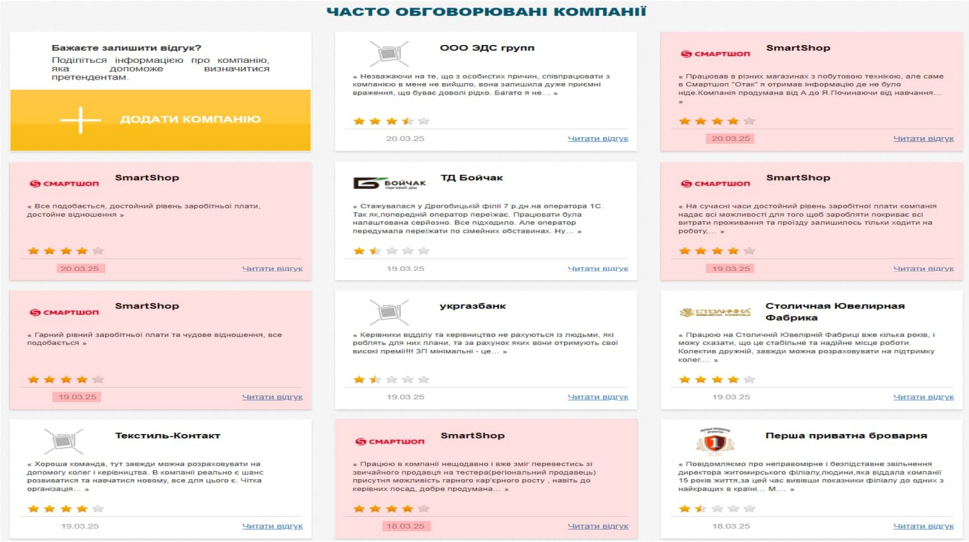


Рис. 3. Приклад накручених відгуків

Джерело: розроблено автором

Одним зі способів розв'язання цієї проблеми є виключення таких коментарів із загальної оцінки в часи аномальної активності. Крім того, такі відгуки можна позначати попередженням, наприклад, «цей коментар був залишений під час незвичайної хвилі активності», щоб відвідувачі були більш обережними при їхньому прочитанні. Це дозволить зробити чіткішим, що ці повідомлення можуть бути ненадійними, і надасть змогу краще оцінити реальність вражень.

Висновки. У процесі дослідження було розглянуто важливість сервісів для збору відгуків, які безпосередньо впливають на репутацію бізнесів. Репутація формується через відгуки, що можуть бути як позитивними, так і негативними, і є основою для залучення клієнтів. Важливим є як проактивний, так і реактивний підхід компанії до управління відгуками.

Сервіси для відгуків виконують важливу роль у комунікації між компаніями та споживачами, але проблема спаму та накрутки відгуків значно знижує їхню надійність. Це стало основною темою дослідження, адже такі маніпуляції можуть серйозно спотворювати реальну картину про компанії та їхні продукти.

Огляд існуючих платформ показав різні підходи до реєстрації та публікації відгуків. Деякі сервіси дозволяють залишати відгуки без реєстрації, що робить їх зручними, але вразливими до спаму. Інші вимагають реєстрації та підтвердження особистості, що підвищує довіру до відгуків, але може відлякати користувачів.

У висновку, для забезпечення довіри до сервісів, важливо вдосконалювати модерацію та виявляти аномальну активність. Якщо сервіс не здатен боротися з накруткою, він не може бути надійним джерелом інформації та має бути виключений з аналізу.

Список використаних джерел

1. Портал відгуків «vidguky.net». Електронний ресурс. URL: https://vidguky.net/ (останнє звернення 12.03.2025 р.)

2. Перший незалежний сайт відгуків України «Vidhuk.ua». Електронний ресурс. URL: https://www.vidhuk.ua/uk/ (останнє звернення 14.03.2025 р.)

3. Платформа «Незалежний сайт відгуків України». Електронний ресурс. URL: https://vidguk.in.ua/ (останнє звернення 15.03.2025 р.)

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

4. Інтернет-портал «naprobu». Електронний ресурс. URL: https://naprobu.ua/ (останнє звернення 17.03.2025 р.)
5. Найбільша спільнота розробників України «DOU». Електронний ресурс. URL: https://jobs.dou.ua/reviews/ (останнє звернення 20.03.2025 р.)
6. Reputation and Feedback Systems in Online Platform Markets / Steven Tadelis. Електронний ресурс. URL: https://www.annualreviews.org/content/journals/10.1146/annurev-economics-080315-015325 (останнє звернення 21.03.2025 р.)
7. Business reputation and social media: A primer on threats and responses / Ines Schulze Horn, Torben Taros, Sven Dirkes, Lucas Hüer, Maximilian Rose, Raphael Tietmayer, Efthymios Constantinides. Електронний ресурс. URL: https://link.springer.com/article/10.1057/dddmp.2015.1 (останнє звернення 21.03.2025 р.)

Роботу виконано під науковим керівництвом PhD, доцента
ДЕСЯТКО А. М.

СТВОРЕННЯ 2D-ПЛАТФОРМЕРІВ В UNITY: ОСНОВНІ ПРИНЦИПИ ТА НАЙКРАЩІ ПРАКТИКИ

МАТЯШ В. В., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглядаються ключові аспекти розробки 2D-платформерів за допомогою Unity. Описано основні етапи створення гри, включаючи планування, проєктування рівнів, реалізацію фізики та анімації. Особливу увагу приділено важливості геймдизайну та звукового оформлення. Також представлені цікаві факти про відомі ігри цього жанру. Стаття стане корисною як для початківців, так і для досвідчених розробників.

The article explores the key aspects of developing 2D-platformer games using Unity. It describes the main stages of game creation, including planning, level design, physics implementation, and animation. Special attention is paid to the importance of game design and sound effects. Additionally, interesting facts about popular games in this genre are presented. The article will be useful for both beginners and experienced developers.

Актуальність. 2D-платформери залишаються одним із найпопулярніших жанрів в індустрії відеоігор. Їх привабливість пояснюється простотою управління, захоплюючим ігровим процесом і можливістю створення унікальних візуальних стилів. Unity – один із провідних ігрових рушіїв, що пропонує потужні інструменти для розробки 2D-платформерів, роблячи процес створення ігор доступним як для початківців, так і для досвідчених розробників.

Мета статті є розгляд ключових принципів розробки 2D-платформерів у Unity, аналіз основних механік, фізики, анімації та геймдизайну. Також буде представлено аналіз попередніх досліджень та практичні рекомендації для успішного створення ігор цього жанру.

Об'єкт дослідження є процес розробки 2D-платформерів з використанням Unity.

Предмет дослідження - є механіки 2D-платформерів, методи їх реалізації в Unity та підходи до оптимізації ігрового процесу.

Аналіз попередніх досліджень. Багато дослідників та розробників вивчали створення платформерів та основи геймдизайну. Джессі Шелл у своїй книзі «The Art of Game Design» підкреслює важливість балансу між викликом та задоволенням у грі. Крім того, офіційна документація Unity та навчальні платформи, такі як Unity Learn, пропонують практичні рекомендації для роботи з 2D середовищем, фізикою та анімацією.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Створення 2D-платформерів в Unity – це цікавий та захоплюючий процес, який дозволяє розробникам створювати інтерактивні ігри з різноманітними механіками, персонажами та рівнями. Існує кілька основних принципів та практик, які допомагають ефективно розробляти такі ігри, зберігаючи при цьому високу якість та зручність для користувачів.

Платформери зародилися в 80-х роках і швидко стали одним із найуспішніших жанрів відеоігор. Класичні приклади – Super Mario Bros.**,** Sonic the Hedgehog **і** Donkey Kong. Вони заклали основи жанру, включаючи стрибки, збирання предметів і проходження рівнів із перешкодами.

Основні елементи 2D-платформерів:

Головний персонаж**:** Це центральна фігура гри, якою керує гравець. Реалізація головного персонажа передбачає створення компонентів для руху, стрибків та взаємодії з об'єктами. У Unity для цього часто використовують Rigidbody2D для фізики та Animator для анімації.

Платформи та перешкоди: Вони створюють виклики для гравця. Платформи можуть бути статичними або рухомими. Використання BoxCollider2D для колізій та Rigidbody2D для рухомих платформ дозволяє легко створювати різноманітні рівні. Рекомендується додати ефекти, такі як платформи, що розсипаються або рухаються в залежності від тригерів.

Вороги та NPC**:** Вороги додають елемент виклику. Реалізація ворогів може включати простий AI для руху та атаки. Використовуйте Animator для створення анімацій та NavMesh для руху ворогів. NPC можуть пропонувати квести або діалоги, для чого зручно використовувати TextMeshPro та Dialogue System.

Предмети для збору: Збирання предметів додає мотивацію гравцеві. Використовуйте Collider2D з тригерами (isTrigger) для збору монет або бонусів.

Фінальна мета: Може бути представлена як вихід з рівня, перемога над босом або виконання завдання. Створіть об'єкт із тригером, який відстежує досягнення мети, та додайте відповідну анімацію чи заставку для завершення рівня. Досягнення кінцевої точки або виконання певного завдання.

Перед початком розробки важливо скласти план гри та визначити ключові механіки. Для цього можна використовувати різні інструменти для планування та дизайну рівнів. Одним із популярних інструментів є Tiled, який дозволяє створювати 2D рівні за допомогою тайлів. Tiled має зручний інтерфейс для розміщення плиток та редагування карти, а також підтримує експорт у формати, що можуть бути використані в Unity. Це дає змогу швидко створювати складні рівні з різними перешкодами та платформами.

Іншим інструментом є Unity Tilemap Editor, який інтегрується безпосередньо в Unity і дає змогу створювати рівні безпосередньо в середовищі Unity. Цей інструмент дозволяє працювати з тайлами, автоматично розміщувати їх на сітці і навіть налаштовувати властивості рівнів через компоненти Unity. Перевагою Tilemap є можливість швидко вносити зміни в рівні та тестувати їх безпосередньо в грі.

Unity забезпечує гнучкі можливості для реалізації фізики завдяки компонентам Rigidbody2D та Box Collider 2D. Реалістична фізика додає глибини геймплею, забезпечуючи природні рухи персонажа.

1. Гравітація: Визначає швидкість падіння

Гравітація – це один з основних факторів, який визначає, як персонаж буде рухатися у вертикальному напрямку. У реальному світі гравітація тягне об'єкти вниз, і в іграх ми намагаємося відтворити цей ефект, щоб рухи персонажа виглядали природно.

Як це працює в Unity: Unity використовує систему Rigidbody2D для роботи з фізикою 2D-об'єктів. Один з параметрів цього компонента – це гравітаційна сила**,** яка визначає, наскільки сильно персонаж буде притягуватися до землі. Цей параметр можна налаштувати вручну, щоб змінити темп падіння персонажа.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Гравітація в Unity за замовчуванням встановлена на значення -9.81 м/с², що є стандартним для більшості ігор. Однак у 2D платформерах часто використовують слабшу гравітацію, щоб персонаж не падав занадто швидко, що дозволяє створити більш комфортний та контрольований ігровий процес.

Можливі налаштування гравітації:

Параметр Gravity Scale: Це множник для глобальної сили гравітації, який дозволяє налаштувати, наскільки сильно персонаж піддається гравітації. Якщо встановити значення більше за 1, падіння буде швидшим, а менше – повільнішим.

Форма руху: Щоб зробити падіння більш природним, можна налаштувати криву руху, щоб персонаж спочатку падав повільно, а потім пришвидшувався. Це дасть відчуття реалістичного фізичного руху.

Приклад: Якщо ви хочете, щоб ваш персонаж стрибав і повільно опускався вниз, зменшіть гравітаційну шкалу. Якщо ж потрібно прискорити падіння або зробити його більш драматичним (наприклад, у випадку падіння з великої висоти), збільште цей параметр.

2. Сила стрибка: Встановлює висоту стрибка

Сила стрибка визначає, як високо персонаж зможе підскочити від поверхні. В Unity цей параметр зазвичай налаштовується за допомогою Rigidbody2D і використовується в поєднанні з вхідними даними від гравця.

Як це працює в Unity:

Сила стрибка: Встановлюється через команду AddForce або через зміни властивостей Rigidbody2D. Це значення зазвичай задається в напрямку вгору (по осі Y) та контролює силу, з якою персонаж буде відштовхуватись від землі.

Розмір стрибка: Важливо, що сила стрибка залежить від значення гравітації – чим більша гравітація, тим нижчою буде максимальна висота стрибка.

Параметри, що впливають на стрибок:

Jump Force (Сила стрибка): Це величина, яку додають до Rigidbody2D при виконанні стрибка. Зазвичай вона має значення, яке враховує масу об'єкта і впливає на висоту стрибка. Чим більша сила, тим вищим буде стрибок.

Висота стрибка: Висота стрибка залежить від величини сили стрибка та гравітації. Якщо ви хочете, щоб стрибок був високим, збільшуйте силу стрибка або зменшуйте гравітацію.

Повторні стрибки: У деяких іграх можна додати можливість повторних стрибків у повітрі, використовуючи додаткову логіку для перевірки того, чи персонаж знаходиться на землі чи вже в повітрі. Для цього потрібно контролювати зіткнення з землею або платформами. Приклад: Якщо персонаж має занадто слабкий стрибок, можна збільшити параметр Jump Force. Якщо ж він стрибає занадто високо, зменште значення цієї сили або збільште гравітацію, щоб стрибок не був занадто довгим.

3. Тертя та інерція: Впливають на рух персонажа

Тертя і інерція – це важливі фізичні аспекти, які визначають, як персонаж буде рухатись на різних поверхнях, а також як швидко він буде зупинятися після того, як припинить рух.

Тертя: Тертя є силою, яка діє між поверхнею та об'єктом, що рухається. В контексті 2D платформерів тертя важливе для того, щоб персонаж не ковзав по землі після того, як припинить рухатися. Тертя може змінюватись в залежності від матеріалу поверхні (камінь, лід, пісок тощо).

Static Friction – це максимальна сила тертя, що заважає об'єкту почати рух.

Dynamic Friction – це сила тертя, яка діє на об'єкт, коли він вже рухається.

Як це працює в Unity: Unity дозволяє налаштовувати параметри тертя через Physic Material 2D, де можна вказати значення для Friction. Це дозволяє змінювати, як персонаж взаємодіє з поверхнею: на слизьких поверхнях тертя буде малим, а на твердіших поверхнях – більшим.

Інерція: інерція визначає, як об'єкт продовжує рухатись після припинення сили, що його рухає. Для платформерів це зазвичай означає, що персонаж не зупиняється миттєво після того, як ви припинили рухати його в певному напрямку.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Якщо інерція велика, персонаж буде рухатися довше після того, як гравець припинить натискати на кнопку руху.

Якщо інерція маленька, персонаж зупиниться майже миттєво після припинення натискання кнопок руху.

Приклад: якщо персонаж рухається занадто повільно або здається «липким», зменшіть значення інерції, щоб його рухи стали більш чіткими. Якщо персонаж ковзає по поверхнях, збільште тертя або налаштуйте Physic Material 2D для кожної поверхні.

Для створення плавних анімацій використовується система Animator. Спрайти дозволяють відтворювати рухи персонажа кадр за кадром. Також варто додати ефекти частинок для покращення візуального сприйняття.

Рекомендації для анімації:

- Використовуйте чіткі й динамічні рухи.
- Створіть різні анімації для бігу, стрибків і атак.
- Додайте візуальний фідбек для покращення ігрового досвіду.

Звук та музика – важливі елементи, що створюють атмосферу гри та підвищують емоційний вплив на гравця. Вони покращують взаємодію з середовищем та підсилюють візуальні ефекти.

Звуки додають реалістичності й інтерактивності. Вони використовуються для:

- Стрибків персонажа, збирання предметів, ударів по ворогах.
- Спецефектів (вибухи, зміна середовища).
- Звуків навколишнього середовища (вітер, вода).

В Unity це працює через AudioSource, який відтворює звуки при активації. Для змінення звуку залежно від місця чи напрямку можна використовувати AudioListener.

Музика створює атмосферу та емоційну настройку гри. Вона може бути:
Фоновою (задня музика, що підтримує атмосферу).
Динамічною, яка змінюється залежно від подій у грі (наприклад, екшн-сцена).

Unity підтримує імпорт музичних файлів через AudioClip та відтворення через AudioSource. Для більш складної динаміки можна використовувати Audio Mixer, що дозволяє регулювати гучність та ефекти.

Unity підтримує різні формати файлів (MP3, WAV, OGG). Важливо налаштувати компресію аудіо, щоб збалансувати якість та розмір файлу.
Audio Mixer дозволяє комбінувати звуки та змінювати їх у реальному часі. Це корисно для:
Регулювання гучності музики та ефектів.

Створення складних звукових сцен, наприклад, змінюючи ефекти для різних середовищ.
Завдяки цим інструментам, Unity надає потужні можливості для управління звуковим контентом гри, дозволяючи створювати багатий та динамічний звук.

Цікаві факти про 2D-платформери

Super Mario Bros. стала однією з найпопулярніших ігор усіх часів. *Super Mario Bros.*, випущена в 1985 році для Nintendo Entertainment System (NES), є однією з найбільш значущих ігор в історії відеоігор. Вона не тільки популяризувала жанр 2D платформерів, але й заклала основи для розвитку ігор цього типу. Ігра має простий, але захоплюючий геймплей, де гравець керує Маріо, що рухається через численні рівні, рятуючи принцесу Тоїку. З моменту свого випуску, Super Mario Bros. продала понад 40 мільйонів копій, а також стала іконою культури, що вплинула на всю індустрію відеоігор.

Celeste була створена лише за 4 дні під час геймджему, але згодом перетворилася на повноцінну гру. *Celeste* почалася як маленький проект на Ludum Dare Game Jam у 2016 році, який тривав лише 4 дні. Розробники, Матео Кукан та Саша Стоффер, вирішили створити платформер, який би поєднував складний геймплей з емоційною історією. Однак після успішної реакції спільноти, вони вирішили розширити гру, і вже у 2018 році вийшла повноцінна версія Celeste. Гра отримала високу оцінку критиків за свою механіку, складність і неймовірно емоційну сюжетну лінію, яка досліджує теми депресії та боротьби з особистими труднощами.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Hollow Knight відзначилася вражаючою художньою стилістикою та захоплюючим геймплеєм. *Hollow Knight* – це глибокий 2D-платформер, який поєднує в собі елементи метроїдванії (Metroidvania) з чудовою художньою стилістикою. Гра розповідає про пригоди маленького воїна, який досліджує підземне королівство, повне загадок і небезпек. Вражаюча графіка, продуманий рівень складності та глибокий сюжет, що відкривається під час гри, зробили Hollow Knight культовим хітом серед любителів 2D-платформерів. Вона отримала численні нагороди за найкращий дизайн та ігровий процес, а її успіх призвів до створення сиквела, який очікується в найближчому майбутньому.

Ці ігри не лише стали важливими віхами в розвитку жанру, але й допомогли змінити підхід до дизайну платформерів, поєднуючи складні ігрові механіки з глибокими емоційними історіями та чудовою графікою.

Висновки. 2D-платформери займають важливе місце в історії відеоігор, надаючи гравцям унікальні можливості для взаємодії з ігровими світом через прості, але захоплюючі механіки. Розвиток жанру можна відстежити від класичних ігор, таких як Super Mario Bros.**,** до сучасних хітів, таких як Celeste та Hollow Knight, які продовжують розширювати можливості жанру, поєднуючи складні геймплейні елементи з емоційними сюжетами та інноваційними художніми рішеннями.

Ключові аспекти, такі як фізика руху, рівні складності, звук та музика, є важливими елементами, які створюють незабутній досвід для гравців. Успіх 2D-платформерів не лише в їхній простоті, але й у здатності створювати глибокі, інтуїтивно зрозумілі ігрові світи, що мають реальний вплив на гравців.

Вивчаючи різноманітні аспекти розробки 2D-платформерів, можна побачити, що кожна з цих ігор має свою унікальну особливість, що робить її особливою для шанувальників жанру. Інноваційні розробки у звуковому оформленні, дизайні рівнів, та використанні сучасних технологій дозволяють створювати гри, які поєднують класичні елементи з новими підходами до геймплею та емоційного впливу.

Цей жанр продовжує бути популярним завдяки своїй здатності адаптуватися до нових вимог гравців і технологій, залишаючись одночасно доступним для нових і досвідчених геймерів.

Список використаних джерел

1. Овербек, Г. Створення 2D платформерів: основи програмування та дизайну / Г. Овербек // Журнал комп’ютерних технологій. – 2018. – № 3. – С. 42–55.
2. Шмідт, С. Unity для початківців: Повний посібник / С. Шмідт. – 2017. – К.: Артек, 352 с.
3. Тренер, Л. Звук та музика в ігровій індустрії / Л. Тренер. – 2019. – С. 120–135.
4. О’Голлеран, М. Історія 2D платформерів: від Mario до сучасних проектів / М. О’Голлеран // Відеоігрові жанри. – 2020. – С. 78–95.
5. Linares, F. Game Development in Unity: Physics and Mechanics / F. Linares // GameDev Journal. – 2021. – Vol. 6. – P. 42–58.
6. Джонсон, Р. Психологія геймплею: як ігри формують емоції / Р. Джонсон. – 2020. – Нью-Йорк: IG Publishing, 330 с.
7. Unity Technologies Офіційний посібник для розробників 2D ігор в Unity // Режим доступу: https://unity.com/ (останнє звернення 25.03.2025 р.)

Роботу виконано під науковим керівництвом канд. екон. наук, доцента
ПАЛАГУТИ К. О.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

МЕТОДОЛОГІЇ УПРАВЛІННЯ ПРОЄКТАМИ У СТВОРЕННІ MVP

НОВИЦЬКА В. В., 2 курс 4м група ФІТ ДТЕУ,
освітня програма «Управління проєктами програмних продуктів»

У статті розглядаються особливості застосування різних методологій управління проєктами при створенні мінімально життєздатного продукту (MVP). Проаналізовано ефективність Agile, Lean, Scrum та Kanban методологій у контексті розробки MVP для стартапів та корпоративних інновацій. Досліджено ключові метрики успіху MVP та практики їх вимірювання. Розглянуто взаємозв'язок між вибором методології та специфікою галузі, розміром команди, бюджетом та часовими обмеженнями. Визначено рекомендації щодо адаптації методологій до різних типів MVP-проєктів та можливі гібридні підходи для оптимізації процесу розробки.

The article discusses the peculiarities of applying various project management methodologies when creating a minimum viable product (MVP). The effectiveness of Agile, Lean, Scrum and Kanban methodologies in the context of MVP development for startups and corporate innovations is analysed. The key metrics of MVP success and practices of their measurement are investigated. The relationship between the choice of methodology and the specifics of the industry, team size, budget and time constraints is considered. Recommendations for adapting methodologies to different types of MVP projects and possible hybrid approaches to optimise the development process are identified.

Актуальність. Підхід мінімального життєздатного продукту (MVP) став фундаментальною концепцією в сучасній розробці програмного забезпечення та стартапах. В умовах зростаючої конкуренції та швидкої зміни вимог ринку здатність швидко розробити та протестувати MVP з мінімальними ресурсами стає критично важливим фактором успіху інноваційних проєктів. Важливість вибору правильної методології управління проєктами при створенні MVP важко переоцінити, оскільки від цього залежить не тільки швидкість виведення продукту на ринок, але й ефективність використання ресурсів, адаптивність до змін та успішність перевірки бізнес-гіпотез.

Традиційні каскадні моделі управління проєктами часто виявляються занадто жорсткими для динамічного середовища MVP, де швидкість, гнучкість і орієнтація на відгуки користувачів мають першорядне значення. Натомість Agile-методології, Lean-підходи та їхні похідні пропонують більш відповідний інструментарій для створення MVP. Водночас, відсутність системного аналізу ефективності різних методологій в контексті специфічних вимог MVP-проєктів створює значні виклики для практиків та команд розробників.

Актуальність дослідження посилюється тим, що хоча концепція MVP існує вже більше десяти років, відбувається постійна еволюція методологій управління проєктами та практик розробки програмного забезпечення. Це створює необхідність регулярного перегляду та оновлення підходів до створення MVP, особливо у світлі нових технологічних можливостей, змін в очікуваннях споживачів та трансформації бізнес-моделей в умовах цифрової економіки.

Метою статті є аналіз та порівняння сучасних методологій управління проєктами у контексті створення MVP, визначення їхніх переваг та обмежень, а також формування рекомендацій щодо вибору оптимальної методології або комбінації методологій залежно від особливостей проєкту та бізнес-цілей.

Об'єктом дослідження є процеси розробки мінімально життєздатного продукту (MVP) в ІТ-проєктах.

Предмет дослідження є методології управління проєктами та їх ефективність у створенні MVP.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Аналіз попередніх досліджень. К. Швабер та Д. Сазерленд вважають, що методологія Scrum створює оптимальні умови для ітеративної розробки MVP через фокус на короткі спринти та регулярну переоцінку пріоритетів [5, с. 28-31], але помиляються в тому, що недостатньо адаптують класичні практики Scrum для надзвичайно обмежених за часом та ресурсами MVP-проєктів. Вони підкреслюють важливість ролі власника продукту в узгодженні бізнес-цілей та технічних можливостей, а також цінність ретроспективних зустрічей для постійного вдосконалення процесу. Практика показує, що Scrum особливо ефективний для MVP-проєктів, які вимагають залучення міждисциплінарних команд та можуть мати змінні вимоги.

Д. Андерсон та М. Кон вважають, що для створення MVP Kanban-підхід має перевагу в проєктах з високим ступенем невизначеності та частими змінами пріоритетів [4, с. 118-121], але помиляються в тому, що недостатньо інтегрують процеси валідації продукту з кінцевими користувачами в базову канбан-систему. Вони аргументують, що візуалізація робочого процесу, обмеження незавершеної роботи та оптимізація потоку цінності дозволяють підтримувати необхідну адаптивність при розробці MVP. Дослідження показує, що Kanban особливо ефективний у середовищах, де команди працюють над кількома паралельними ініціативами або мають змінний склад.

Виклад основного матеріалу. Мінімально життєздатний продукт (MVP) - це версія нового продукту, яка дозволяє команді зібрати максимальну кількість перевірених знань про споживача з мінімальними зусиллями. Концепція MVP була детально розроблена Е. Рісом в рамках методології Lean Startup і з тих пір стала основоположною для розробки інноваційних продуктів [6, с. 103-107]. Основною метою MVP є перевірка ключових бізнес-гіпотез з мінімальними витратами ресурсів, що дозволяє або підтвердити життєздатність бізнес-ідеї, або своєчасно здійснити напрямок до того, як будуть витрачені значні ресурси [2].

Мінімальний життєздатний продукт повинен відповідати трьом ключовим характеристикам: він має бути мінімальним (містити лише функції, необхідні для перевірки гіпотези), життєздатним (приносити реальну користь користувачам) і продуктивним (мати достатню якість, щоб користувачі могли оцінити його основну цінність). Важливо розуміти, що MVP - це не демонстраційна версія або прототип, а повноцінний продукт з обмеженою функціональністю, спрямований на задоволення потреб перших користувачів і отримання зворотного зв’язку.

Ефективність MVP безпосередньо залежить від обраної методології управління проєктом, яка має забезпечити баланс між швидкістю виведення продукту на ринок, якістю ключових функцій та гнучкістю реагування на відгуки користувачів. Різні методології мають свої особливості, переваги та обмеження, які слід враховувати при розробці MVP.

Agile-методологія є одним з найпопулярніших підходів до розробки програмного забезпечення, який ідеально підходить для створення MVP завдяки своїй адаптивності та орієнтації на взаємодію з клієнтом. Agile-маніфест, опублікований у 2001 році, заклав основи для нової парадигми розробки, що цінує людей та взаємодію вище процесів та інструментів, працюючий продукт вище вичерпної документації, співпрацю з клієнтом вище узгодження умов контракту та реакцію на зміни вище дотримання плану [6].

Agile-процес розробки MVP базується на ітеративному підході з короткими циклами, що дозволяє швидко отримувати зворотній зв’язок та адаптувати продукт. Ключовим елементом є зосередження на мінімальному наборі функцій, які надають цінність користувачеві. У контексті MVP методологія Agile є особливо ефективною завдяки можливості швидко змінювати пріоритети функцій на основі зворотного зв’язку.

На рис. 1 представлено схематичне зображення Agile-методології з її ключовими компонентами та циклічним процесом вдосконалення.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

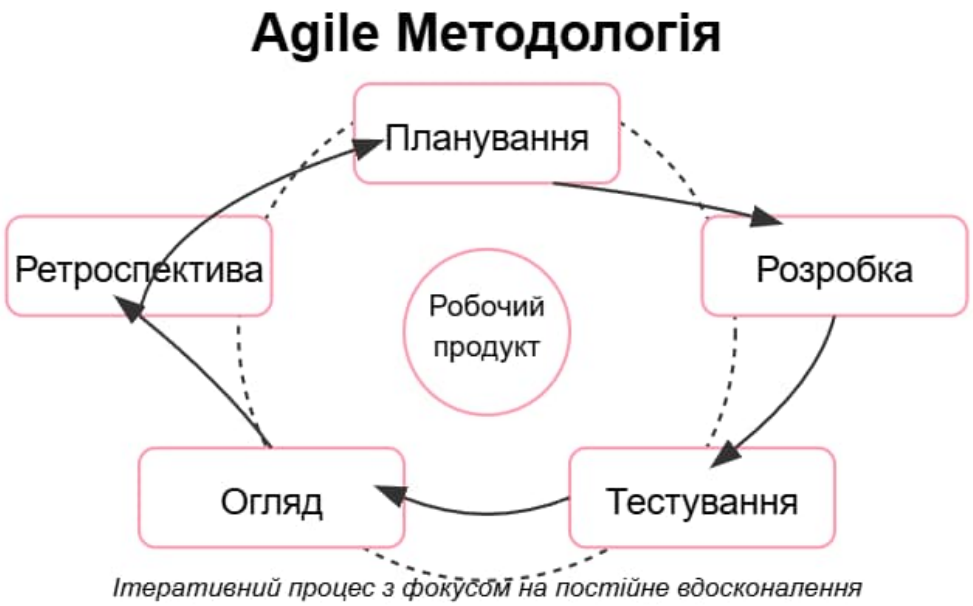


Рис. 1. Agile методологія

Джерело: розроблено автором

Як видно з рисунка 1, Agile-процес для MVP складається з послідовних ітерацій, кожна з яких включає планування, розробку, тестування, огляд і ретроспективу. Кожна ітерація закінчується робочим інкрементом продукту, який можна продемонструвати зацікавленим сторонам. Особливістю Agile в контексті MVP є акцент на ранньому і частому випуску мінімально життєздатних версій продукту для отримання швидкого зворотного зв'язку.

На відміну від традиційного Agile, ітерації розробки MVP зазвичай коротші (1–2 тижні замість 2–4), а етап планування зосереджений на виборі функцій, які є найбільш критичними для перевірки бізнес-гіпотези. Зворотній зв'язок від користувачів стає ключовим елементом, який визначає напрямок наступних ітерацій, а в деяких випадках може призвести до радикальної зміни пріоритетів або навіть повного розвороту. У таблиці 1 наведено основні переваги та обмеження застосування Agile-методології при розробці MVP.

Таблиця 1

Переваги та обмеження Agile-методології для MVP

Переваги	Обмеження
Швидкий вихід на ринок з мінімальним набором функцій	Ризик втрати структурованості при надмірній гнучкості
Гнучкість у зміні пріоритетів на основі зворотного зв'язку	Можливі складнощі з довгостроковим плануванням
Раннє виявлення проблем та можливість швидкого півоту	Потреба в дисципліні команди для уникнення постійної зміни напрямку
Фокус на функціях, які приносять найбільшу цінність	Можливий «технічний борг» через швидкість розробки
Мінімізація витрат на розробку нерелевантних функцій	Необхідність постійної комунікації та залучення стейкхолдерів

Джерело: розроблено автором на основі [1, 3, 5]

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Agile-методологія особливо ефективна для MVP-проєктів у динамічних галузях, де вимоги та потреби користувачів можуть швидко змінюватися [7]. Однак, щоб бути максимально ефективним, Agile слід адаптувати до специфічних потреб MVP-проєктів, приділяючи особливу увагу мінімізації функціональності та забезпеченню швидкого зворотного зв'язку.

Scrum-методологія є найпопулярнішою реалізацією Agile-методології, яка пропонує конкретний набір практик та ролей для організації процесу розробки. Методологія Scrum особливо добре підходить для створення MVP завдяки своїй структурованості та фокусу на регулярну доставку працюючого продукту [7]. На рисунку 2 зображено схему Scrum-методології, що наочно демонструє її основні елементи та процеси.

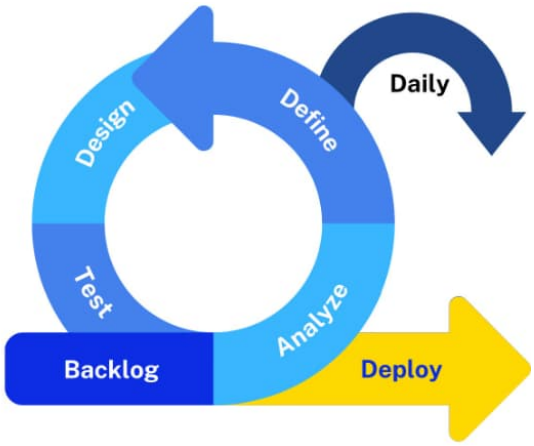


Рис. 2. Scrum-методологія

Джерело: [7]

Scrum організовує роботу в короткі цикли розробки, так звані спринти, які зазвичай тривають від 1 до 4 тижнів. Кожен спринт має чітку мету і закінчується демонстрацією робочого приросту продукту. У контексті MVP спринти часто скорочують до 1-2 тижнів, щоб забезпечити максимальну швидкість зворотного зв'язку.

Scrum для MVP включає наступні ключові елементи:

1. Беклог продукту (Product Backlog) – пріоритизований список всіх функцій, при цьому функції, критичні для MVP, отримують найвищий пріоритет;
2. Планування спринту (Sprint Planning) – вибір функцій для реалізації в поточному спринті, з фокусом на ті, що є найбільш важливими для підтвердження бізнес-гіпотези;
3. Щоденні зустрічі (Daily Scrum) – короткі щоденні наради для синхронізації команди та виявлення блокерів;
4. Огляд спринту (Sprint Review) – демонстрація результатів спринту стейкхолдерам та збір зворотного зв'язку;
5. Ретроспектива спринту (Sprint Retrospective) – аналіз процесу роботи та визначення можливостей для вдосконалення.

Особливістю Scrum для MVP є тісна співпраця власника продукту з командою розробки, що дозволяє швидко адаптувати пріоритети беклогу на основі отриманого зворотного зв'язку. У таблиці 2 представлено особливості застосування Scrum-методології для різних типів MVP, що допомагає підібрати оптимальний підхід.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Таблиця 2

Особливості застосування Scrum для різних типів MVP

Тип MVP	Особливості застосування Scrum	Рекомендована тривалість спринту
Концептуальний MVP (перевірка концепції)	Короткі спринти, мінімальний набір функцій, часті релізи	1 тиждень
Маркетинговий MVP (оцінка попиту)	Фокус на користувацький інтерфейс та досвід, часте збирання метрик	1-2 тижні
Технічний MVP (доказ технічної реалізованості)	Спринти з акцентом на технічні аспекти, детальне технічне планування	2 тижні
MVP для корпоративних інновацій	Адаптований Scrum з елементами традиційного управління, детальна документація	2–3 тижні

Джерело: розроблено автором на основі [1, 7]

Scrum-методологія особливо ефективна для MVP-проєктів, які потребують структурованого підходу та чіткого розмежування ролей [1]. Проте, для максимальної ефективності Scrum при розробці MVP необхідно уникати надмірної формалізації процесу та зберігати гнучкість у адаптації до змін.

Lean-методологія, що походить з виробничої системи Toyota, була адаптована для розробки програмного забезпечення та стартапів. Ця методологія лежить в основі концепції Lean Startup і є особливо релевантною для розробки MVP, оскільки фокусується на мінімізації відходів та максимізації цінності для користувача. Lean-підхід до розробки MVP центрується навколо циклу «Створити-Виміряти-Вивчити», який дозволяє швидко перевіряти бізнес-гіпотези з мінімальними витратами ресурсів. Цей цикл є основою методології Lean Startup і ідеально підходить для створення та вдосконалення MVP.

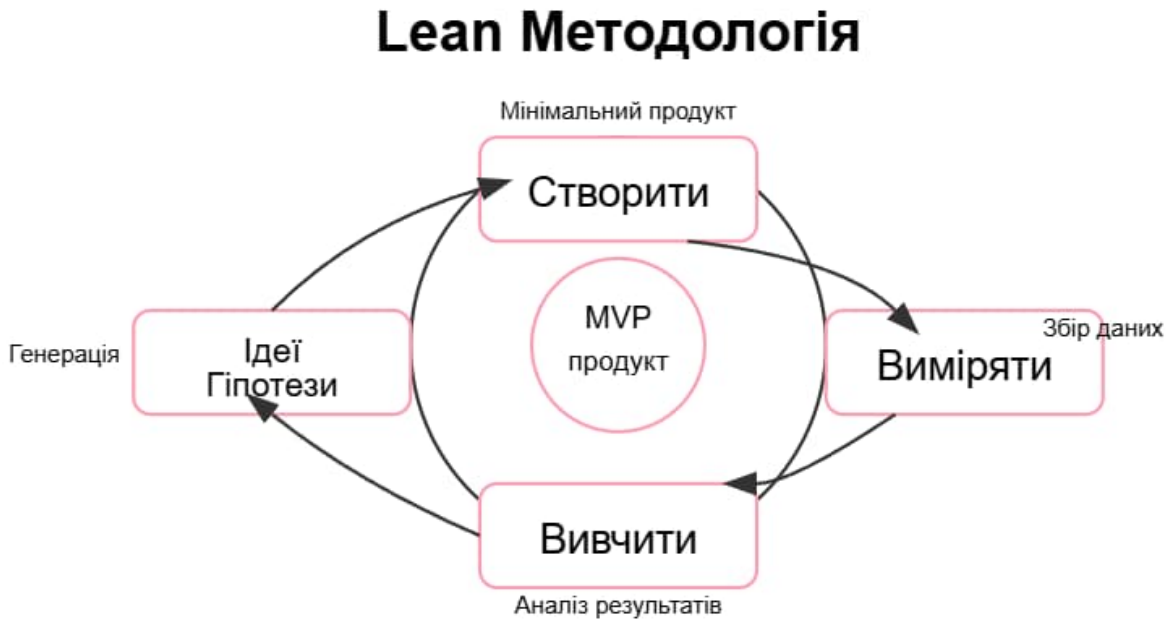


Рис. 3. Lean -методологія

Джерело: розроблено автором

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Lean-процес для MVP включає три ключові етапи:

1. Створити (Build): розробка найпростішої версії продукту, яка дозволяє перевірити ключову бізнес-гіпотезу. На цьому етапі фокус робиться на мінімальному наборі функцій, необхідних для доставки цінності першим користувачам.
2. Виміряти (Measure): збір та аналіз даних про використання продукту для оцінки його ефективності в задоволенні потреб користувачів. Ключовими елементами є визначення релевантних метрик та встановлення системи вимірювання.
3. Вивчити (Learn): аналіз отриманих даних для підтвердження або спростування початкової гіпотези. На основі цього аналізу приймається рішення про продовження поточного напрямку, його коригування або повне повернення.

Особливістю Lean-підходу для MVP є постійний акцент на валідованому навчанні – процесі перевірки гіпотез через наукові експерименти. Швидкість проходження цього циклу є ключовим фактором успіху стартапу [3, с. 108–112].

Lean-методологія особливо ефективна для стартапів та інноваційних проєктів, які працюють в умовах високої невизначеності. За даними дослідження, стартапи, які використовують Lean-підхід, на 50% частіше успішно знаходять життєздатну бізнес-модель порівняно з тими, хто використовує традиційні підходи [5]. Ефективність Lean для MVP полягає у фокусі на мінімізації витрат ресурсів при максимізації швидкості навчання, що критично важливо на ранніх етапах розвитку продукту.

Kanban-методологія є гнучкою методологією управління проєктами, що походить з виробничої системи Toyota та фокусується на візуалізації робочого процесу, обмеженні незавершеної роботи (WIP) та оптимізації потоку цінності. Ця методологія особливо корисна для розробки MVP у ситуаціях з часто змінюваними пріоритетами та необхідністю гнучкого реагування на зміни. На рисунку 4 представлено схему Kanban-дошки, яка є ключовим інструментом цієї методології та наочно демонструє потік роботи в процесі розробки MVP.

Рис. 4. Kanban-методологія

Джерело: розроблено автором

Kanban організовує роботу через візуалізацію робочого процесу за допомогою дошки з картками, що представляють окремі завдання. На відміну від Scrum, Kanban не має фіксованих спринтів, а натомість фокусується на безперервному потоці роботи та обмеженні кількості паралельних завдань.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Kanban-процес для MVP включає наступні ключові елементи:

1. Kanban-дошка – візуальне представлення робочого процесу з колонками, що відображають різні стадії розробки (Беклог, В роботу, В розробці, Тестування, Завершено).
2. Картки завдань – кожне завдання представлене карткою з описом, пріоритетом та іншою релевантною інформацією. Завдання, критичні для MVP, зазвичай маркуються спеціальним чином (як показано на діаграмі з міткою «MVP»).
3. Обмеження WIP (Work in Progress) – для кожної колонки встановлюється ліміт кількості завдань, що можуть знаходитися в ній одночасно, що запобігає перевантаженню команди та фокусує на завершенні розпочатих завдань.
4. Постійний потік – замість фіксованих ітерацій, Kanban заохочує безперервний потік роботи, де завдання переміщуються по дошці в міру їх виконання.

Особливістю Kanban для MVP є можливість швидко адаптувати пріоритети в реальному часі, що особливо цінно в умовах високої невизначеності та частих змін. Kanban дозволяє максимізувати ефективність команди при мінімальному адміністративному навантаженні [2, с. 124-127]. У таблиці 3 представлено основні метрики, які використовуються в Kanban для оцінки ефективності розробки MVP, що допомагає командам відстежувати прогрес та оптимізувати процеси.

Таблиця 3.

Метрики Kanban для оцінки ефективності розробки MVP

Метрика	Опис	Значення для MVP
Lead Time	Час від створення завдання до його завершення	Допомагає оцінити загальну швидкість розробки MVP
Cycle Time	Час від початку роботи над завданням до його завершення	Дозволяє оцінити ефективність команди
Throughput	Кількість завдань, завершених за певний період	Показує продуктивність команди при розробці MVP
WIP	Кількість завдань у процесі одночасно	Допомагає контролювати фокус команди
Cumulative Flow	Візуалізація кількості завдань у кожному статусі з часом	Допомагає ідентифікувати вузькі місця в процесі
Blocker Clustering	Аналіз частих причин блокування завдань	Допомагає усунути системні перешкоди для швидкої розробки MVP

Джерело: розроблено автором на основі [4, 6]

Kanban-методологія особливо ефективна для MVP-проєктів з часто змінюваними пріоритетами та необхідністю швидкого реагування на зворотний зв’язок [2]. Перевагою Kanban для MVP є мінімальний адміністративний тягар, що дозволяє командам зосередитися безпосередньо на створенні цінності для користувачів. У таблиці 5 наведено порівняльний аналіз основних підходів до розробки MVP за ключовими критеріями.

Таблиця 4.

Порівняння методологій для MVP-проєктів

Критерій	Agile	Scrum	Lean	Kanban
Структурованість процесу	Середня	Висока	Низька-середня	Середня
Часові рамки	Ітерації середньої тривалості (2–4 тижні)	Фіксовані спринти (1–4 тижні)	Гнучкі, орієнтовані на цикл «Створити-Виміряти-Вивчити»	Безперервний потік без фіксованих ітерацій
Адаптивність до змін	Висока	Середня (зміни	Дуже висока	Висока

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Критерій	Agile	Scrum	Lean	Kanban
		впроваджуються між спринтами)		(можливість миттєвої зміни пріоритетів)
Орієнтація на мінімізацію відходів	Середня	Середня	Висока	Висока
Підтримка експериментів	Середня	Обмежена рамками спринту	Дуже висока	Висока
Оптимальний розмір команди	5-9 осіб	5-9 осіб	Гнучкий, ефективний для малих команд	Гнучкий, ефективний навіть для малих команд
Складність впровадження	Середня	Середня-висока	Низька-середня	Низька-середня
Вимоги до документації	Варіюються	Помірні (product/sprint backlog)	Мінімальні	Мінімальні
Фокус на зворотному зв'язку	Високий	Високий (в кінці спринту)	Дуже високий	Високий
Ключові метрики успіху	Story points, швидкість команди	Швидкість, Burndown charts	Метрики продукту, темп навчання	Lead time, Throughput, Cycle time

Джерело: розроблено автором на основі [1, 3, 4, 7]

Аналіз показує, що вибір методології суттєво залежить від контексту проєкту, організаційної культури та специфіки команди. Жодна методологія не є універсально кращою за інші, і в багатьох випадках найбільш ефективним рішенням є адаптація або комбінування різних підходів.

У практиці розробки MVP часто найбільш ефективними виявляються *гібридні підходи*, які поєднують елементи різних методологій для створення оптимального процесу з урахуванням специфіки проєкту та команди.

Гібридні підходи дозволяють використовувати сильні сторони кожної методології та компенсувати їх обмеження.

Рис. 6. Гібридні підходи до розробки MVP

Джерело: розроблено автором

Особливо популярними гібридними підходами для MVP є:

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

1. *Scrumban* – поєднує структурованість Scrum із гнучкістю Kanban. Використовує спринти для планування, але візуалізацію та обмеження WIP з Kanban для оптимізації потоку роботи. Особливо ефективний для проєктів з частими змінами пріоритетів, але які потребують певної структурованості [5];
2. *Lean Scrum* – інтегрує цикл «Створити-Виміряти-Вивчити» з Lean у фреймворк Scrum. Кожен спринт стає експериментом для перевірки гіпотез, а ретроспективи фокусуються на валідованому навчанні;
3. *Lean Kanban* – орієнтований на максимальну гнучкість та мінімізацію відходів. Візуалізація процесу Kanban дозволяє чітко бачити, де виникають затримки, а принципи Lean допомагають їх усунути;
4. *Agile Lean* – поєднує адаптивність Agile із фокусом на цінності для клієнта з Lean. Цей підхід найкраще працює для стартапів, які хочуть швидко експериментувати, але потребують структурованого фреймворку для організації роботи.

Вибір конкретного гібридного підходу має ґрунтуватися на специфіці проєкту, організаційній культурі та наявному досвіді команди. У деяких випадках може мати сенс розробити власний гібридний підхід, який найкраще відповідатиме потребам конкретного MVP-проєкту.

Висновки. Дослідження показало, що вибір методології управління проєктами відіграє вирішальну роль у створенні мінімально життєздатного продукту (MVP), визначаючи його успіх на ринку. Порівняльний аналіз методологій Agile, Scrum, Lean та Kanban показав, що кожна з них має свої переваги та обмеження в контексті MVP-проєктів. Наприклад, Scrum забезпечує структурований процес розробки через чіткий розподіл спринтів та ролей, Kanban пропонує максимальну гнучкість для проєктів з частою зміною пріоритетів, а Lean фокусується на мінімізації відходів та швидкому циклі Build-Measure-Learn (будуй-вимірюй-вчись).

Дослідження показало, що найефективнішим підходом часто є використання гібридних методологій, таких як Scrum або Lean Scrum, які поєднують переваги різних підходів відповідно до специфіки проєкту. Для успішної розробки MVP визначальними факторами є здатність методології забезпечити швидке виведення продукту на ринок, адаптивність до змін на основі зворотного зв’язку з користувачами та фокусування на мінімальному наборі функцій, що створюють цінність.

Практика показує, що вибір оптимальної методології повинен враховувати низку факторів: розмір команди, часові та ресурсні обмеження, рівень невизначеності та складність проєкту. Для стартапів з обмеженими ресурсами та високим рівнем невизначеності найбільш підходящими є підходи Lean та Kanban, тоді як для корпоративних MVP-проєктів, які потребують більшої структурованості, доцільніше використовувати Scrum або Scrumban.

Подальший розвиток методологій управління MVP-проєктами пов’язаний з удосконаленням інструментів вимірювання успішності MVP, створенням більш ефективних механізмів збору та аналізу відгуків користувачів, інтеграцією нових технологій, зокрема штучного інтелекту, для оптимізації процесу розробки та прийняття рішень. Комплексне застосування відповідних методологій дозволяє компаніям не тільки ефективно створювати інноваційні продукти, але й отримувати конкурентні переваги на сучасному ринку.

Список використаних джерел

1. Smith, J., & Johnson, P. (2022). Modern Agile Methodologies for MVP Development in Software Startups. Journal of Software Engineering, 15(3), 45–62.

2. Garcia, L., et al. (2023). Comparative Analysis of Project Management Methodologies for Minimum Viable Products. International Journal of Project Management, 41(1), 102–115.

3. Wilson, T. (2022). Kanban in Practice: Case Studies from MVP Development. Project

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Management Journal, 53(4), 223–240.
4. Anderson, D., & Carmichael, A. (2022). Essential Kanban Condensed (2nd Edition). Blue Hole Press.
5. Schwaber, K., & Sutherland, J. (2023). The Scrum Guide: The Definitive Guide to Scrum [Електронний ресурс]. URL: https://scrumguides.org/scrum-guide.html
6. Ries, E. (2021). The Lean Startup: Updated Insights on Building Products That Matter. Crown Business.
7. MVP in Agile: Definitions, Roles and Steps to Build Successful Projects? (2024) [Електронний ресурс]. URL: https://goldenowl.asia/blog/mvp-in-agile
8. The Standish Group. CHAOS Report 2022 [Електронний ресурс]. URL: https://www.standishgroup.com/store/chaos-report-2022

Роботу виконано під науковим керівництвом PhD, доцента
ДЕСЯТКО А. М.

ВИКОРИСТАННЯ МАШИННОГО НАВЧАННЯ ДЛЯ ПРОГНОЗУВАННЯ ТОВАРООБІГУ

ОЛІЙНИК Д. М., 2 курс 2м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглянуто основні аспекти застосування машинного навчання для прогнозування товарообігу на підприємствах роздрібної торгівлі. Зазначено переваги використання алгоритмів машинного навчання для оптимізації запасів, зменшення витрат на зберігання та підвищення задоволення клієнтів.

The article examines the main aspects of using machine learning for forecasting turnover in retail enterprises. The advantages of using machine learning algorithms for inventory optimization, reducing storage costs, and increasing customer satisfaction are highlighted.

Актуальність. Використання машинного навчання для прогнозування товарообігу є надзвичайно актуальним у сучасних умовах розвитку бізнесу. Збільшення обсягів даних, які генеруються підприємствами, потребує ефективних методів їх аналізу та використання для прийняття управлінських рішень. Машинне навчання дозволяє автоматизувати процеси прогнозування, що сприяє підвищенню точності прогнозів та оптимізації запасів. Це, в свою чергу, допомагає уникнути дефіциту товару та зменшити витрати на зберігання, задовольнити попит клієнтів.

Знайомство з машинним навчанням може бути включено в корпоративні процедури для посилення контролю запасів і задоволення вимог покупця. Таким чином, вже не зручно теоретично обґрунтовувати використання машинного навчання для прогнозування товарообігу, крім того, майже застосовувати отримані результати в корпоративних процедурах. Це дозволить корпораціям підвищити продуктивність контролю над запасами та підвищити вимоги покупців до задоволення. Впровадження машинного навчання у корпоративні процедури може бути ключовим елементом досягнення для корпорацій, які прагнуть залишатися агресивними на ринку та швидко реагувати на зміни.

Для підтвердження ефективності впровадження машинного навчання проведено

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

численні елементи використання машинного навчання для прогнозування товарообігу, а також вибір алгоритмів, інформаційних рядів і підготовки, оцінка версії та оцінка результатів. Особливу увагу можна приділити реалістичній користі отриманих результатів, що є хорошим способом дозволити організаціям отримати унікальні вказівки для впровадження машинного навчання у свої процедури. Крім того, можна розглянути приклади корпорацій, які вже можуть ефективно використовувати машинне навчання для прогнозування обороту, що є хорошим способом, який дозволяє нам оцінити фактичні переваги та недоліки цього підходу.

Метою статті є дослідження особливостей використання машинного навчання для прогнозування товарообігу з метою підвищення ефективності управління запасами та оптимізації бізнес-процесів.

Об'єктом дослідження є процес розробки та впровадження системи машинного навчання для прогнозування товарообігу на підприємствах роздрібної торгівлі.

Предметом дослідження є алгоритми машинного навчання, які використовуються для прогнозування товарообігу.

Аналіз попередніх досліджень. Дослідженню використання машинного навчання для прогнозування товарообігу присвячені праці багатьох науковців. Значний внесок у розвиток цієї теми зробили В.А. Гросул, І.А. Шинкаренко, Н.В. Проскурніна та інші. Їхні роботи охоплюють різні аспекти застосування машинного навчання в роздрібній торгівлі, включаючи аналіз даних, вибір алгоритмів

Виклад основного матеріалу. Ефективний контроль запасів є ключовою складовою бізнесу, а правильне прогнозування оборотності запасів може значно підвищити його ефективність. Використання алгоритмів машинного навчання дозволяє компаніям оптимізувати запаси, уникнути дефіциту товарів та зменшити витрати на зберігання, а також аналізувати історичні дані про продажі, враховувати сезонні коливання, рекламні кампанії та різні фактори, які впливають на попит.

Оптимізація запасів є важливою справою для будь-якого підприємства, яке прагне ефективно контролювати свої ресурси. Алгоритми машинного навчання можуть допомогти визначити, які товари мають найбільший попит, а які товари можна виключити з асортименту. Це дозволяє компаніям більш ефективно використовувати складську площу та зменшувати витрати на зберігання. Як варіант, використовуючи алгоритми машинного навчання, підприємство може прогнозувати попит на товари протягом святкових періодів та коригувати запаси. Це дозволяє уникнути надлишкових запасів, що покращує ефективність контролю.

Точні прогнози товарообігу дозволяють компаніям уникати дефіциту товарів, що може призвести до втрати продажів та незадоволення клієнтів. Виявлення тенденцій зростання попиту на певні товари дозволяє забезпечити наявність необхідних товарів на складі та задовольнити його..

Зменшення витрат на зберігання є ще одним важливим компонентом контролю запасів. Також алгоритми машинного навчання можуть допомогти визначити, які товари мають найбільший попит, а які товари можна виключити з асортименту. Це дозволяє компаніям більш ефективно використовувати складську площу та зменшувати витрати на зберігання.

Загалом, схему впровадження машинного навчання для прогнозування товарообігу зображено на рис. 1.

Виробничі плани є критично важливим компонентом комерційного підприємства, який дозволяє корпораціям успішно використовувати свої ресурси та задовольняти потреби споживачів. Точне прогнозування товарообігу з використанням алгоритмів машинного навчання може істотно підвищити ефективність цього процесу. Алгоритми машинного навчання досліджують історичні дані про виробництво, враховують сезонні коливання та різні фактори, які впливають на попит. Це дозволяє корпораціям планувати виробничі потужності відповідно до прогнозованого попиту та уникати перевиробництва.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

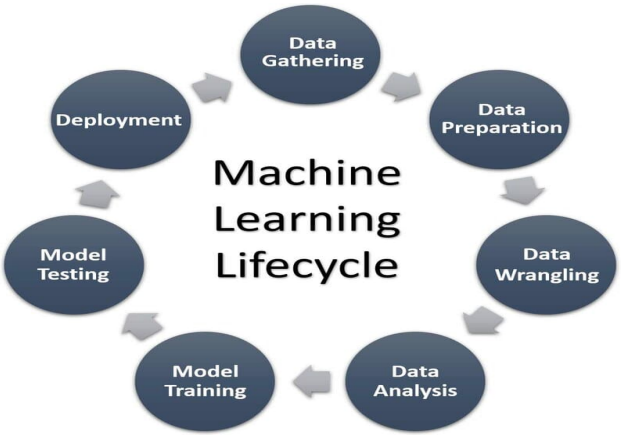


Рис. 1. Впровадження машинного навчання

Джерело: [5]

Також це дозволяє корпораціям швидко реагувати на необхідні зміни. Дані можуть аналізуватися в реальному часі і можна регулярно змінювати виробничі плани відповідно до необхідних коригувань. Це дозволяє корпораціям швидко адаптуватися до змін ринкових ситуацій і уникати перевиробництва або дефіциту продукції.

Методи машинного навчання для прогнозування товарообігу. Ефективне прогнозування товарообігу є важливим для успішного управління запасами, планування виробництва, а також рекламних і маркетингових стратегій. Використання алгоритмів машинного навчання дозволяє компаніям створювати точні прогнози, які враховують численні фактори, що впливають на попит. Серед різноманітних методів машинного навчання, які можна використовувати для прогнозування товарообігу, найбільш ефективними є лінійна регресія, випадкові ліси та нейронні мережі.

Одним із найпростіших алгоритмів машинного навчання є лінійна регресія. Зазвичай, вона використовується для моделювання лінійних залежностей між залежною змінною (товарообіг) і незалежними змінними (фактори, що впливають на товарообіг). Цей метод дає можливість оцінити вплив кожного фактора на товарообіг і створити прогноз на основі історичних даних. Завдяки простоті використання та інтерпретації лінійна регресія є ефективною для підприємств, які мають відносно прості та лінійні взаємозв'язки між змінними.

Випадкові ліси – це ансамблевий метод, який використовує кілька дерев рішень для створення більш точного прогнозу. Вони зменшують варіативність і підвищують точність прогнозів за рахунок усереднення результатів кількох дерев рішень. Цей метод враховує складні взаємозв'язки між змінними і є ефективним для прогнозування товарообігу. Випадкові ліси підходять для підприємств середнього розміру, які мають складні взаємозв'язки між змінними та потребують більш точних прогнозів. Наприклад, компанії, що займаються електронною комерцією або мають широкий асортимент товарів, можуть використовувати випадкові ліси для прогнозування обсягів продажів на основі історичних даних та факторів, таких як економічні умови та поведінка споживачів. Випадкові ліси забезпечують високу точність прогнозів і стійкість до перенавчання, що робить їх надійним інструментом для складних завдань. Однак цей метод має відносно високу обчислювальну складність і меншу інтерпретованість порівняно з лінійною регресією.

Нейронні мережі використовують багатошарову структуру для моделювання складних взаємозв'язків між змінними. Вони здатні враховувати нелінійні взаємозв'язки і обробляти великі обсяги даних. Нейронні мережі автоматично витягують ознаки з даних, що дає можливість створювати точніші прогнози. Цей метод є ефективним для великих підприємств,

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

які мають великі обсяги даних і складні взаємозв'язки між змінними. Наприклад, великі роздрібні мережі або виробничі компанії можуть використовувати нейронні мережі для прогнозування обсягів продажів на основі даних про поведінку споживачів, таких як історія покупок та взаємодія з маркетинговими кампаніями. Нейронні мережі характеризуються здатністю моделювати складні нелінійні взаємозв'язки і високою точністю прогнозів при наявності великої кількості даних. Однак цей метод має високу обчислювальну складність і потребує великої кількості даних для навчання, що може бути викликом для деяких підприємств. Крім того, нейронні мережі мають меншу інтерпретованість порівняно з іншими методами.

Таблиця 1

Порівняння методів машинного навчання

Метод	Переваги	Недоліки	Вимоги до обчислювальних ресурсів	Складність налаштування
Лінійна регресія	Простота, швидкість обчислення	Обмежена точність для складних даних	Низька	Низька
Випадковий ліс	Висока точність, стійкість до перенавчання	Високі обчислювальні витрати	Високі	Середня
Нейронні мережі	Висока точність, можливість роботи з великими обсягами даних	Складність інтерпретації, потреба в великій кількості даних	Дуже високі	Висока

Джерело: розроблено автором

Компанії можуть зіткнутися з проблемами, пов'язаними з недостатньою потужністю серверів, високими витратами на хмарні обчислення та необхідністю інвестування в сучасне обладнання. Наприклад, для навчання складних моделей можуть знадобитися потужні графічні процесори (GPU) або спеціалізовані апаратні рішення, що може бути дорогим.

Результати прогнозів можуть бути складними для інтерпретації. Бізнес-аналітики та менеджери повинні мати достатні знання для розуміння та використання отриманих даних. Це може вимагати додаткового навчання персоналу та залучення експертів. Наприклад, інтерпретація результатів нейронних мереж може бути складною через їх «чорний ящик» характер, що ускладнює пояснення, чому модель прийняла те чи інше рішення.

Використання машинного навчання може викликати етичні та правові питання, пов'язані з конфіденційністю даних, їх захистом та відповідністю законодавству. Компанії повинні забезпечити дотримання всіх норм та правил, щоб уникнути юридичних проблем. Наприклад, обробка персональних даних повинна відповідати вимогам GDPR, що може вимагати додаткових заходів безпеки та контролю.

Впровадження таких технологій може бути дорогим процесом. Витрати на розробку, налаштування та підтримку моделей, а також на навчання персоналу можуть бути значними. Компанії повинні ретельно оцінювати економічну доцільність таких інвестицій. Наприклад, необхідно враховувати не тільки початкові витрати, але й постійні витрати на підтримку та оновлення системи.

Практичні приклади впровадження. Walmart, один з найбільших ритейлерів у світі, використовує машинне навчання для прогнозування попиту на товари через свою платформу Element. Ця платформа дозволяє аналізувати тисячі даних, включаючи історичні продажі, сезонні тренди, погодні умови та маркетингові кампанії. Завдяки цьому Walmart може оптимізувати запаси на складах, зменшити витрати на зберігання та уникнути дефіциту товарів під час пікових періодів.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Amazon використовує нейронні мережі для прогнозування продажів нових продуктів. Вони аналізують дані про попередні запуски продуктів, відгуки клієнтів, ринкові тренди, а також макроекономічні показники та погодні умови. Це дозволяє передбачити попит на нові моделі з високою точністю та ефективніше планувати виробництво та маркетингові кампанії. Наприклад, під час пандемії COVID-19 Amazon швидко адаптував свої моделі прогнозування, враховуючи нові дані про медичні показники та кількість випадків COVID, що дозволило компанії оперативно реагувати на зміни в попиті.

На рисунку 2 вказано процес прогнозування товарообігу в роздрібній індустрії за допомогою машинного навчання від компанії Amazon.

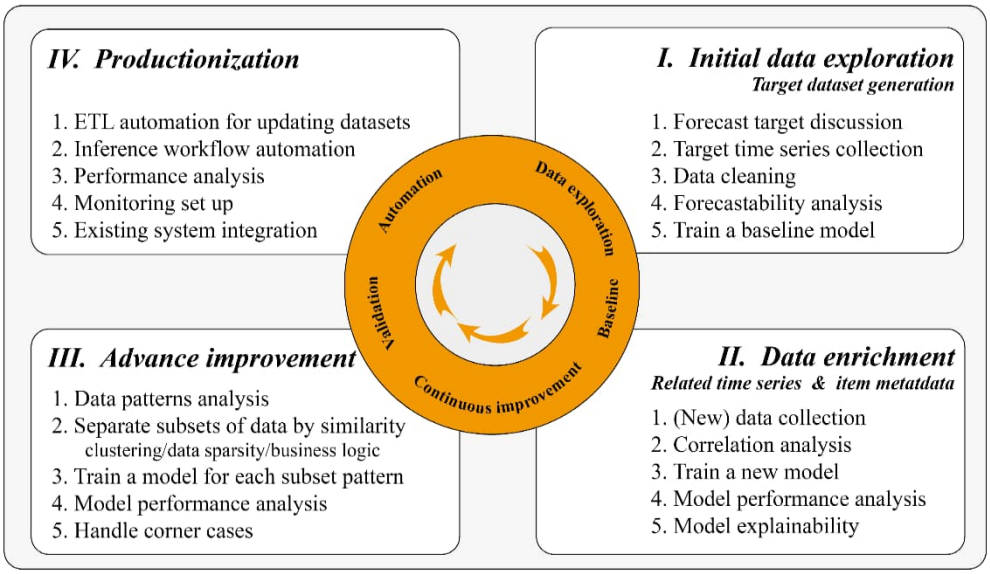


Рис. 2: Процес прогнозування товарообігу

Джерело: [4]

DHL, міжнародний постачальник логістичних послуг, впровадив машинне навчання для прогнозування обсягів перевезень. Використовуючи дані про попередні перевезення, економічні показники та сезонні коливання, компанія оптимізувала маршрути та розподіл ресурсів. Це дозволило зменшити витрати на логістичні процеси та підвищити ефективність роботи.

Висновки. Використання алгоритмів для прогнозування товарообігу відкриває нові можливості для бізнесу, дозволяючи значно покращити управління запасами, виробничі процеси та маркетингові стратегії. Аналіз історичних даних, сезонних коливань та інших факторів дозволяє створювати точні прогнози, що допомагають уникати дефіциту товарів і надлишкових запасів. Це сприяє зменшенню витрат на зберігання та підвищенню задоволення клієнтів.

Різні методи, такі як лінійна регресія, випадкові ліси та нейронні мережі, мають свої переваги та недоліки. Лінійна регресія є простим і швидким методом, але має обмежену здатність моделювати складні взаємозв'язки. Випадкові ліси забезпечують високу точність прогнозів і стійкість до перенавчання, але потребують значних обчислювальних ресурсів. Нейронні мережі здатні моделювати складні нелінійні взаємозв'язки і обробляти великі обсяги даних, але мають високу обчислювальну складність і потребують великої кількості даних для навчання.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Практичні приклади компаній, таких як Walmart, Amazon та DHL, демонструють успішне впровадження алгоритмів для прогнозування товарообігу. Ці кейси показують реальні переваги технологій у бізнес-процесах.

Загалом, алгоритми є потужним інструментом для прогнозування товарообігу, що дозволяє компаніям підвищити ефективність управління запасами, оптимізувати виробничі процеси та забезпечити задоволення потреб клієнтів. Враховуючи виклики та обмеження, компанії можуть успішно інтегрувати ці технології в свої бізнес-процеси та отримати значні переваги від їх використання.

Список використаних джерел

1. Матеріали сайту Walmart Tech // Режим доступу: https://tech.walmart.com/content/walmart-global-tech/en_us/blog/post/walmarts-element-a-machine-learning-platform-like-no-other.html (останнє звернення 26.03.2025).
2. Стаття Predicting The Future Of Demand: How Amazon Is Reinventing Forecasting With Machine Learning журналу Forbes // Режим доступу: https://www.forbes.com/sites/amazonwebservices/2021/12/03/predicting-the-future-of-demand-how-amazon-is-reinventing-forecasting-with-machine-learning/ (останнє звернення 26.03.2025).
3. Матеріалу офіційного сайту компанії DHL: AI in Logistics and Last-Mile Delivery // Режим доступу: https://www.dhl.com/discover/en-global/logistics-advice/logistics-insights/ai-in-logistics-and-last-mile-delivery (останнє звернення 27.03.2025).
4. Матеріали сайту Amazon: Implementing Amazon Forecast in the retail industry: A journey from POC to production// Режим доступу: https://aws.amazon.com/blogs/machine-learning/implementing-amazon-forecast-in-the-retail-industry-a-journey-from-poc-to-production/ (останнє звернення 28.03.2025).
5. Матеріали сайту ResearchGate. // Режим доступу: https://www.researchgate.net/figure/Machine-Learning-Lifecycle_fig3_362485323 (останнє звернення 28.03.2025).

Роботу виконано під науковим керівництвом канд. фіз.-мат. наук, доцента
ГОРОХОВОЇ О. М.

ЕФЕКТИВНІСТЬ ВПРОВАДЖЕННЯ MVP У СТАРТАПІ БУДІВЕЛЬНОЇ ГАЛУЗІ: ЕКОНОМІЧНИЙ ТА УПРАВЛІНСЬКИЙ АНАЛІЗ

ОМЕЛЬЧЕНКО О. М., 2 курс 4м група ФІТ ДТЕУ,
освітня програма «Управління проєктами програмних продуктів»

У статті розглянуто процес управління проєктом зі створення MVP (Minimum Viable Product) для будівельної компанії, що спеціалізується на зведенні житлових будинків. Представлено огляд існуючих підходів до розробки MVP, обґрунтовано актуальність проблеми, визначено мету, об'єкт і предмет дослідження. Проведено аналіз існуючих джерел, описано обрані методології управління проєктом, подано алгоритм реалізації та представлено результати моделювання з використанням математичних формул, графіків і таблиць. Зроблено акцент на економічному обґрунтуванні розробки, ризиках, тестуванні функціоналу та визначенні майбутніх напрямків масштабування продукту.

The article explores the project management process for developing an MVP (Minimum

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Viable Product) for a construction company specializing in residential housing. It presents an overview of existing approaches to MVP development, substantiates the relevance of the problem, and defines the aim, object, and subject of the research. The analysis of existing sources is conducted, the selected project management methodologies are described, and an implementation algorithm is provided. The results of modeling are presented using mathematical formulas, graphs, and tables. Emphasis is placed on the economic justification for the development, risks, functional testing, and outlining future directions for scaling the product.Актуальність.

Актуальність теми. У сучасному цифровому середовищі будівельні компанії потребують ефективних ІТ-рішень для управління процесами, взаємодії з клієнтами та підвищення конкурентоспроможності. Створення MVP дозволяє швидко протестувати гіпотези, зменшити ризики інвестування та зекономити ресурси. Особливої актуальності набуває розробка MVP для компаній, що спеціалізуються на будівництві будинків із плоскими дахами, оскільки ця ніша має власну специфіку в технічних, логістичних, екологічних та клієнтських аспектах. ІТ-рішення у такій галузі дозволяють не лише автоматизувати управління будівництвом, а й підвищити рівень сервісу для замовників.

Аналіз останніх досліджень і публікацій. Наукові підходи до створення MVP детально описані у працях Е. Ріса («Lean Startup») [1], де підкреслюється важливість раннього тестування гіпотез. Методи гнучкого управління, зокрема SCRUM, досліджувалися у роботах Р. Кнаппера та М. Кона [2]. В українському контексті важливими є дослідження Коваленка П. [3], який акцентує на адаптації Agile до специфіки вітчизняного будівництва, та І. Зінченка [4], що описує практичні кейси впровадження цифрових рішень у проєктне управління. Також заслуговують на увагу праці Кондратюка В. щодо економічної оцінки MVP [5] та С. Ганущака, який досліджує цифровізацію будівництва [6].

Мета дослідження – розробити комплексну модель управління проєктом створення MVP веборієнтованого програмного продукту для будівельної компанії з врахуванням технічної специфіки та фінансової доцільності.

Об'єкт дослідження – процес управління ІТ-проєктом у сфері будівництва малоповерхових будинків із плоскими дахами.

Предмет дослідження – методи, інструменти та алгоритми створення MVP для веборієнтованого продукту в будівельній галузі.

Виклад основного матеріалу. Аналіз бізнес-потреб компанії Для формування потреб було проведено SWOT-аналіз компанії «StatusHouse», яка спеціалізується на будівництві приватних будинків Основні проблеми: відсутність централізованої цифрової системи обліку заявок, складність у плануванні закупівель та взаємодії з підрядниками, неавтоматизована клієнтська підтримка.

Вибір методології :

Обрана методологія SCRUM, адаптована до особливостей будівництва. Переваги: ітеративність, прозорість, можливість швидкої зміни функціоналу. Командна структура:

– Product Owner: Олександр Козак – представник компанії StatusHouse (linkedin.com/in/oleksandr-kozak-statushouse)
– Scrum Master: Ірина Ткаченко – бізнес-аналітик (linkedin.com/in/iryna-tkachenko-scrum)
– Front-End Developer: Артем Савчук (linkedin.com/in/artem-savchuk-react)
– Back-End Developer: Богдан Гринько (linkedin.com/in/bohdan-hrynko-nodejs)
– Fullstack Developer: Анна Левицька (linkedin.com/in/anna-levytska-fullstack)
– UI/UX Designer: Катерина Мельник (linkedin.com/in/kateryna-melnyk-uxui)

Консультант з будівництва: Ігор Романюк (linkedin.com/in/ihor-romaniuk-construction Планування MVP.

Планування MVP розпочалося з ідентифікації базових функцій, які забезпечують мінімально життєздатний продукт для початкового запуску:

– Форма запиту клієнта
– Панель адміністратора
– База даних підрядників

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

– Модуль розрахунку кошторису

Загальний список вимог був класифікований за методом MoSCoW на категорії *Must have, Should have, Could have, Won't have*.

Для кількісної оцінки обсягу MVP застосовано метод функціональних точок (Function Point Analysis, FPA). Розрахунок відбувався за такими групами:

– Введення даних (EI) – 4 точки за кожну форму
– Виведення даних (EO) – 5 точок за кожен генератор звітів
– Інтерфейс користувача (EQ) – 4 точки
– Файли логічні (ILF) – 7 точок за кожну логічну сутність
– Інтерфейси зовнішні (EIF) – 5 точок за кожен зовнішній інтерфейс.

Таблиця 1

Складність програмного забезпечення методом Function Point Analysis (FPA)

Компонент	Тип	Кількість	FP/од.	Підсумок
Форма запиту	EI	1	4	4
Панель адміністратора	EQ	1	4	4
База підрядників	ILF	1	7	7
Модуль кошторису (генерація PDF)	EO	1	5	5
Інтеграція з Firebase Auth	EIF	1	5	5
Усього				25

Джерело: розроблено автором

Таким чином, обсяг MVP становить 25 функціональних точок (FP).

Для оцінки тривалості розробки використовуємо емпіричну формулу:

T = FP / V,

де:

T – кількість людино-тижнів,

FP – кількість функціональних точок,

V – продуктивність команди (FP/тиждень)

Якщо середня продуктивність команди становить 5 FP/тиждень, то:

T = 25 / 5 = 5 людино-тижнів.

З урахуванням роботи команди з 3 розробників (Front-End, Back-End, Fullstack), передбачається завершення MVP за 2 спринти по 2 тижні кожен (тобто 4 тижні), що відповідає 6 людино-тижням із запасом на тестування й інтеграцію.

Дерево робіт (WBS):

Рис. 1. Дерево робіт (WBS).

Джерело : розроблено автором

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Таблиця 2

Матриця відповідальності (RACI)

Завдання/роль	Product Owner	Scrum Muster	Front-End	Back-End	Designer	Бізнес аналітик
Збір вимог	A	R				C
UX/UI дизайн	C				R	A
Front-End	C		R			
Back-End	C			R		
Тестування MVP	C	A	R	R	C	R

Джерело: розроблено автором

R (Responsible) – Відповідальний: той, хто виконує завдання (виконавець).

A (Accountable) – Звітує: той, хто приймає остаточне рішення, несе повну відповідальність за завдання.

C (Consulted) – Консультує: експерт або фахівець, до якого звертаються за порадою перед або під час виконання завдання.

I (Informed) – Поінформований: той, кого інформують про хід чи результати завдання (в статті ми його не включали, але його часто додають).

Архітектура рішення Вебзастосунок створюється з використанням стеку: React (front-end), Node.js (back-end), MongoDB (база даних). API – REST. Аутентифікація – JWT.

Схема взаємодії компонентів:

– Користувач → Front-End (React) → Back-End API (Node.js)
– Back-End API → MongoDB
– Front-End → Firebase Authentication
– Адмінпанель → Dashboard з ролями

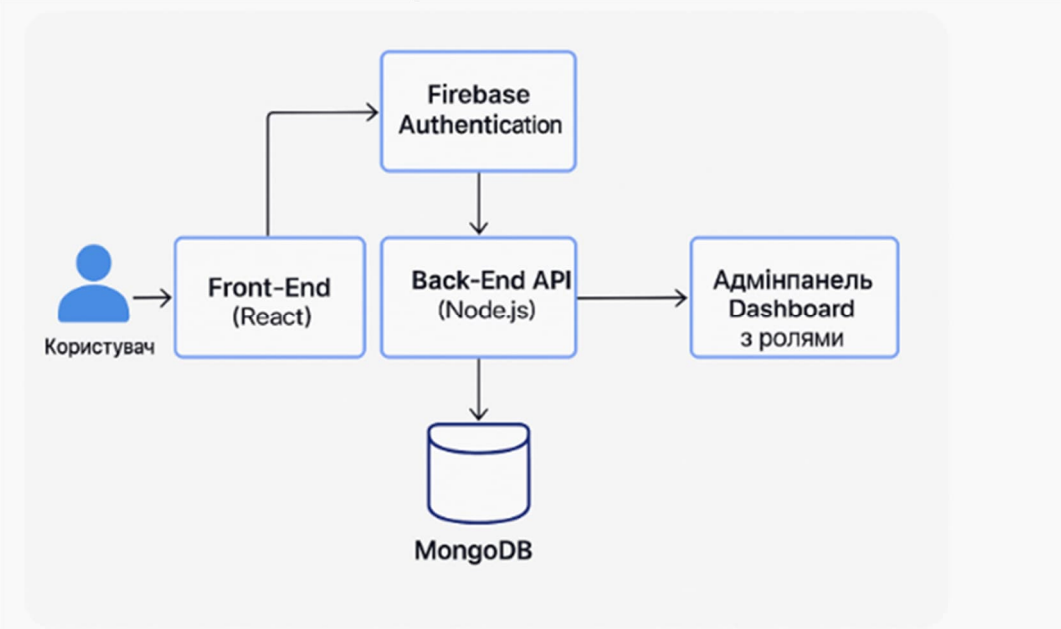


Рис. 2. «Схема взаємодії компонентів»

Джерело: розроблено автором

Математична модель ефективності Формула оцінки результативності MVP: S = ∑ (w-i * x-i)

- , де S – інтегральна оцінка MVP, w-i – ваговий коефіцієнт критерію i, x-i – значення критерію i (0–10), i = 1..n

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

- Критерії оцінювання: зручність UX, швидкодія, гнучкість налаштувань, відповідність бізнес-логіці, відгук користувачів.

Таблиця 3

Оцінки MVP (пілотний тест)

Критерії	Вага (w-i)	Оцінка(x-i)	Результат(w-i*x-i)
UX/UI	0.3	9	2.7
Продуктивність	0.2	7	1.4
Налаштування	0.15	6	0.9
Бізнес логіка	0.2	8	1.6
Відгуки тест-групи	0.15	9	1.35
Сума			7.95

Джерело: Розроблено автором

Економічне обґрунтування MVP Розрахунок ROI (Return on Investment): ROI = (Інтегральна вигода - Витрати на розробку) / Витрати на розробку Припустимо: витрати = 100 000 грн; вигода = 165 000 грн. ROI = (165 000 - 100 000) / 100 000 = 0.65 (65%)

Розподіл витрат на етапи розробки MVP: Рекомендується деталізувати витрати за кожним етапом проєкту, наприклад, на ініціацію, розробку, тестування та впровадження. Це дозволить краще зрозуміти, де та чому витрати можуть бути високими.

Приклад:

– Ініціація: витрати на аналіз вимог, збори з командою, аналіз ринку
– Розробка: витрати на програмування, налаштування серверів, створення бази даних
– Тестування: витрати на тестування, ліцензії для тестування, залучення QA-фахівців
– Впровадження: витрати на запуск MVP, маркетинг та підготовка інтерфейсу
– Після цього можна навести порівняння з вигодами для кожного етапу (наприклад, економія часу, зменшення кількості помилок на етапі розробки, підвищення задоволеності користувачів після впровадження).

Ризики впливу на ROI: Окрім загального аналізу ROI, варто зазначити ризики, які можуть вплинути на досягнення позитивного результату. Наприклад:

Невизначеність у вимогах: Зміна вимог замовника або невірно визначені функції MVP можуть призвести до додаткових витрат.

Технічні проблеми: Проблеми з масштабуванням, помилки в програмуванні можуть уповільнити процес і призвести до додаткових витрат.

Приклад: Зміна вимог замовника може підвищити витрати на розробку на 10-20%, що вплине на загальний ROI.

Часова шкала окупності: Крім самого ROI, можна ввести поняття «період окупності». Це час, який потрібен, щоб повернути витрачені кошти через отриману вигоду. Це дозволить краще оцінити економічну ефективність проєкту.

Тестування функціоналу

Методи тестування: Детальніше розглянь, як будуть проводитися різні види тестування:

Функціональне тестування: Чи будуть використовуватись конкретні інструменти для автоматизованого тестування? Які саме функції MVP перевіряються? Наприклад, можна зазначити використання інструментів як Selenium, Jest, Cypress.

Приклад: «Функціональне тестування виконувалося за допомогою інструменту Selenium, який автоматизує перевірку роботи форм на вебсторінках. Це дозволяє швидко перевірити коректність запитів користувача та отримання даних.»

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Тестування навантаження: Оскільки йдеться про веборієнтований продукт, важливо зазначити, як планується тестувати масштабованість системи під великий потік користувачів. Наприклад, можна додати використання Apache JMeter для перевірки продуктивності системи.

Приклад: «Для навантажувального тестування використовувався інструмент JMeter, щоб змоделювати одночасний доступ 1000 користувачів і перевірити швидкодію основних функцій системи.»

UI/UX тестування: Як саме будуть оцінювати зручність користувацького інтерфейсу? Можна додати, що буде проведено тестування за участю реальних користувачів через фокус-групи чи анкети.Приклад: «UI/UX тестування було проведено на фокус-групах з 10 користувачів, котрі надали зворотний зв'язок щодо інтерфейсу системи, що дозволило зробити корективи в дизайні.Критерії успішності тестування: Крім суб'єктивних оцінок, варто розглянути конкретні метрики.

Наприклад:Швидкість завантаження сторінок (використовуючи інструменти типу Google PageSpeed Insights).

Час відповіді серверу при великій кількості запитів.

Відсоток знайдених багів під час тестування (в ідеалі це повинно бути 0 або мінімальний відсоток).

Візуалізація результатів Нижче подано візуалізацію результатів:

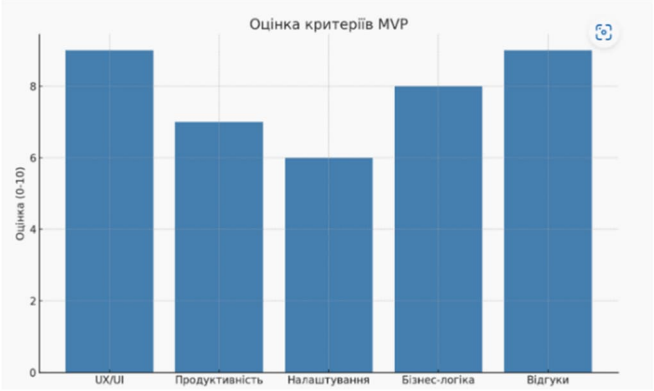


Рис. 3. «Оцінка критеріїв MVP»

Джерело : розроблено автором

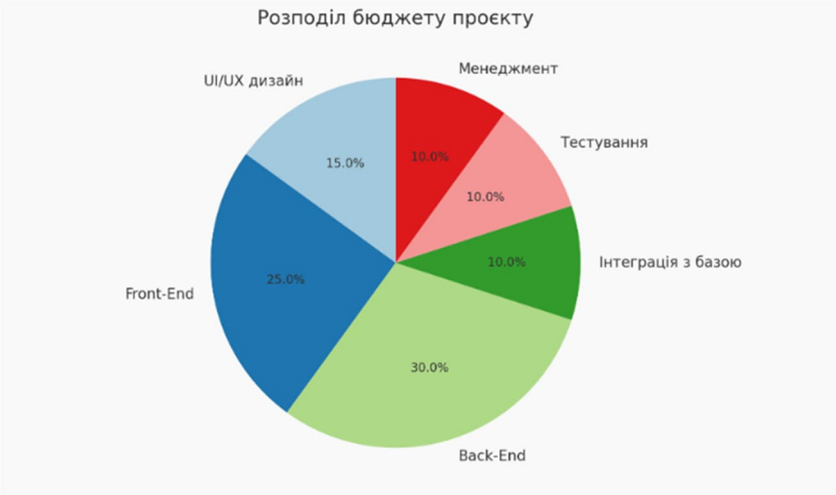


Рис. 4. «Розподіл бюджету проєкту»

Джерело: розроблено автором

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

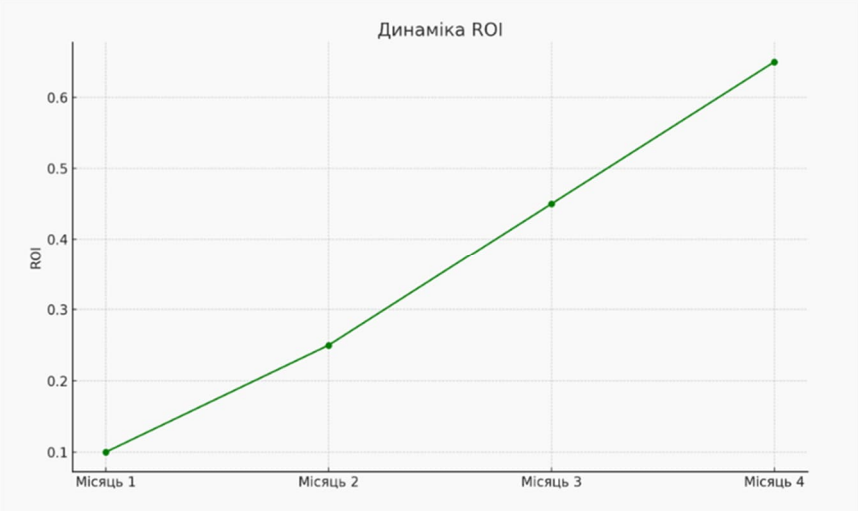


Рис. 5. Динаміка ROI

Джерело: розроблено автором

– Стовпчиковий графік: оцінка критеріїв MVP
– Кругова діаграма: розподіл бюджету проєкту
– Лінійний графік: динаміка ROI
– Аналіз ризиків Типові ризики:
– Технічні: затримка через баги (ймовірність 0.3, вплив 5), масштабування бази (0.2, 4)
– Бізнес-ризики: зміна вимог (0.4, 4), недостатній попит (0.2, 5)
– Комунікаційні: неузгодженість між розробниками та замовником (0.3, 3)

Розрахунок ризику за формулою: Risk Score = P * I

Таблиця 4

Аналіз ризиків

Ризик	Ймовірність(P)	Вплив(I)	Risk Score
Баги в коді	0.3	5	1.5
Недостатній попит	0.2	5	1.0
Комунікаційні бар'єри	0.3	3	0.9
Зміна вимог замовника	0.4	4	1.6

Джерело : Розроблено автором

Висновки та подальші дослідження

Розроблена модель дозволяє ефективно реалізовувати MVP для будівельної компанії, орієнтуючись на специфіку ринку. SCRUM та AHP надали структурованість і вимірюваність результатів. Високий ROI свідчить про економічну доцільність MVP-розробки. Аналіз ризиків виявив пріоритетні точки контролю. Наступні кроки: розширення функціоналу, інтеграція з CRM та ERP, оцінка масштабованості, формування платформи на SaaS-моделі.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Список використаних джерел

1. Ріс Е. Стартап без помилок. Lean Startup. – К.: Наш Формат, 2018.
2. Кнапер Р. Гнучке планування проєктів. – Львів: Літопис, 2020.
3. Коваленко П. Інтеграція ІТ-рішень у будівництві. – Наукові записки НУВГП, 2022.
4. Зінченко І. Agile в українському будівництві. – Вісник КНУБА, 2023.
5. Кондратюк В. Розробка та економічна оцінка MVP в ІТ-проєктах. – Бізнес-інформ, 2023.
6. Ганущак С. Цифровізація будівництва: проблеми та перспективи. – Економіка і суспільство, 2022.

Роботу виконано під науковим керівництвом д-ра техн. наук, професора
ЦЮЦЮРИ С. В.

СТРАТЕГІЯ МЕНЕДЖМЕНТУ СТВОРЕННЯ MVP ВЕБОРІЄНТОВАНОГО ПРОДУКТУ

**ОСТАПЕНКО В. В., 2 курс 4м група ФІТ ДТЕУ,
освітня програма «Управління проєктами програмних продуктів»**

У статті розглядається актуальність ефективного менеджменту в ІТ компаніях, зокрема в контексті розробки веб-орієнтованих програмних продуктів. Проаналізовано сучасні підходи до управління проєктами, їх вплив на продуктивність команди та успішність створення мінімально життєздатного продукту (MVP). Метою дослідження є обґрунтування важливості правильного менеджменту для оптимізації роботи ІТ компаній. Основна увага приділена розробці інформаційно-довідкової та трекінгової системи як інструменту підвищення ефективності. Висвітлено практичні аспекти впровадження таких систем та перспективи їх розвитку.

The article examines the relevance of effective management in IT companies, particularly in the context of developing web-oriented software products. Modern project management approaches, their impact on team productivity, and the success of creating a minimum viable product (MVP) are analyzed. The study aims to substantiate the importance of proper management for optimizing the operations of IT companies. The focus is on the development of an information-reference and tracking system as a tool for enhancing efficiency. Practical aspects of implementing such systems and prospects for their further development are highlighted.

Актуальність. Сучасний розвиток інформаційних технологій зумовлює зростання конкуренції серед ІТ компаній, що вимагає від них швидкої адаптації до змін, ефективного використання ресурсів та забезпечення високої якості продуктів. Одним із ключових факторів успіху є правильний менеджмент, який охоплює планування, організацію, контроль та мотивацію команди. Проблема управління проєктами в ІТ сфері набуває особливої актуальності в умовах розробки мінімально життєздатних продуктів (MVP), де необхідно поєднати швидкість виконання, обмежені ресурси та високу якість результату. Відсутність чіткої системи менеджменту може призвести до зриву термінів, перевитрат бюджету та зниження конкурентоспроможності компанії. У цьому контексті створення інформаційно-довідкових та трекінгових систем стає важливим інструментом для підвищення ефективності роботи команд.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Аналіз останніх досліджень і публікацій. Питання менеджменту в IT-сфері активно досліджуються як українськими, так і зарубіжними науковцями. Зокрема, Алярова А. [1] наголошує на важливості впровадження сучасних технологій у менеджменті персоналу IT-компаній, акцентуючи увагу на систематизації даних і підвищенні ефективності управління. У свою чергу, Волох Д. С. [2] аналізує налаштування процесів і проектного менеджменту, підкреслюючи необхідність адаптації управлінських підходів до специфіки IT-середовища. Кубявка Л. Б. і Латишева Т. В. [3] розглядають концепцію побудови проєктного та продуктового IT-менеджменту, пропонуючи принципи, що базуються на гнучких методологіях, таких як Agile і Scrum, для швидкої адаптації до змін вимог замовника. Заярнюк О., Сторожук О. та Сокуренко О. [4] звертають увагу на особливості менеджменту IT-бізнесу в умовах війни, наголошуючи на соціальній відповідальності як важливому факторі успіху. Дослідження від IBM [5] підтверджують, що компанії, які використовують спеціалізовані трекінгові системи (наприклад, Jira, Trello), досягають на 20–30% вищої продуктивності порівняно з тими, що застосовують традиційні методи управління.

Водночас залишаються невирішені питання щодо адаптації таких систем до специфіки невеликих IT компаній, які працюють над MVP. Більшість досліджень зосереджені на великих корпораціях, тоді як малі команди стикаються з унікальними викликами, такими як обмеженість ресурсів і потреба в швидкому тестуванні ідей. Саме цій проблемі присвячена дана стаття.

Мета дослідження полягає в обґрунтуванні важливості правильного менеджменту для підвищення ефективності роботи IT компаній через розробку та впровадження інформаційно-довідкової та трекінгової системи для управління проєктом створення MVP веб-орієнтованого продукту.

Об'єкт дослідження – процес менеджменту в IT компаніях, що розробляють програмні продукти.

Предмет – процеси управління проектами в IT-компаніях.

Виклад основного матеріалу дослідження. Розробка MVP (Minimum Viable Product) – мінімально життєздатного продукту – потребує чіткої стратегії та злагодженої координації всіх процесів. На цьому етапі команда визначає цільову аудиторію, формулює основну ціннісну пропозицію, окреслює ключовий функціонал і створює базову версію продукту, готову до тестування реальними користувачами. Успіх цих кроків значною мірою залежить від правильного менеджменту, який забезпечує послідовність і ефективність роботи в умовах конкурентного IT-середовища, де швидкість виведення продукту на ринок і його якість часто стають вирішальними факторами для виживання стартапу чи проєкту.

Ключовим завданням менеджера в цьому процесі є визначення пріоритетів і цілей команди, що набуває особливої актуальності в умовах швидких змін вимог клієнтів чи ринкових тенденцій. Для цього в IT-сфері активно застосовуються гнучкі методології управління, такі як Agile, Scrum або Kanban [4]. Ці підходи дозволяють оперативно адаптуватися до нових умов, проводити тестування ідей у коротких циклах і мінімізувати ймовірність невдачі. Наприклад, у Scrum щоденні стендапи тривалістю 15 хвилин дають змогу команді оперативно обмінюватися інформацією, виявляти перешкоди й коригувати план, а спринти тривалістю 1-2 тижні забезпечують створення робочих прототипів, які можна одразу презентувати замовникам для отримання зворотного зв'язку [3]. Це особливо цінно для MVP, де кожна ітерація наближає продукт до ринкової готовності, дозволяючи уникнути зайвих витрат на непотрібний функціонал.

Ефективність таких методологій значною мірою залежить від використання спеціалізованих інструментів для трекінгу та візуалізації завдань. Системи на кшталт Jira, Trello чи Asana стали невід'ємною складовою менеджменту в IT-компаніях, забезпечуючи розподіл задач, моніторинг прогресу й дотримання встановлених термінів. Прозорість, яку надають ці інструменти, сприяє підвищенню продуктивності команди та формуванню

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

культури відповідальності, що є критично важливим для стартапів і проєктів на етапі MVP [2]. Наприклад, Trello дозволяє візуалізувати робочий процес за принципом Kanban (To Do, In Progress, Done), що спрощує контроль навіть для невеликих команд, тоді як Jira пропонує розширені функції аналітики, такі як відстеження кількості закритих багів за спринт чи прогнозування термінів завершення, що допомагає підвищувати якість продукту.

Ці трекінгові системи доповнюються інформаційно-довідковими платформами, які відіграють важливу роль у технічному забезпеченні менеджменту. Інструменти типу Confluence або Notion дають змогу централізовано зберігати вимоги до продукту, технічну документацію, нотатки про зворотний зв'язок від клієнтів і розробників, зменшуючи ризик втрати важливої інформації при змінах у складі команди [4]. Наприклад, оновлення специфікації продукту в Confluence із додаванням коментарів від тестувальників може прискорити ітерації, а інтеграція з Google Drive забезпечує миттєвий доступ до великих файлів, таких як дизайн-макети чи прототипи інтерфейсів. Для MVP це означає можливість швидко реагувати на відгуки перших користувачів, зберігаючи при цьому цілісність проєктної документації.

Однак технічні інструменти – це лише одна сторона медалі, адже успіх проєкту значною мірою залежить від людського фактора. Управління персоналом у процесі розробки MVP вимагає ретельної організації командної роботи, мотивації працівників і створення сприятливого психологічного клімату. Менеджер виступає не лише координатором, а й ментором, який підтримує команду, а також посередником у вирішенні конфліктів [1]. Регулярний зворотний зв'язок через one-on-one зустрічі допомагає вчасно виявляти проблеми, такі як перевантаження чи демотивація, а гнучкий графік і можливості професійного розвитку підвищують залученість працівників. Наприклад, впровадження системи OKR (Objectives and Key Results) дозволяє пов'язати індивідуальні цілі розробників із загальною стратегією проєкту, що особливо корисно для невеликих команд, де кожен учасник відіграє ключову роль.

Ці аспекти стають ще більш значущими в умовах віддаленої чи гібридної роботи, яка останнім часом стала стандартом для IT-компаній. Щоб забезпечити злагодженість процесів у таких умовах, менеджери використовують відеозустрічі через Zoom, щоденні стендапи в Slack і спільну роботу над ідеями в Miro [5]. Такий підхід вимагає балансу між контролем і підтримкою, орієнтованою на результат, а інструменти типу Clockify дають змогу відстежувати витрати часу, що особливо важливо для розподілених команд. Досвід показує, що компанії, які приділяють увагу налагодженню віддаленої взаємодії, наприклад, через організацію віртуальних тімбілдингів чи неформальних онлайн-зустрічей, досягають зниження плинності кадрів і підвищення лояльності працівників, що позитивно впливає на стабільність проєктів.

Управління персоналом у контексті розробки MVP також охоплює процес адаптації нових працівників (онбордінгу), який набуває особливого значення через високу динаміку кадрових змін у IT-сфері. Середній термін роботи IT-фахівця в одній компанії рідко перевищує 1,5-2 роки, що зумовлює постійне оновлення команд [1]. Ефективний онбордінг включає підготовку детальних інструкцій із завданнями, розробку індивідуальних планів із цілями та KPI, а також серію онлайн-зустрічей у перші тижні для знайомства з проєктом і командою. Наприклад, у практиці деяких компаній новачкам надають "дорожню карту" (roadmap), де чітко прописано очікування на випробувальний термін, що дозволяє скоротити час адаптації до 2-3 тижнів і швидше залучити їх до розробки MVP.

Важливим елементом менеджменту, пов'язаним із людськими ресурсами, є рекрутинг, який в IT-сфері має свої особливості. Для проєктів типу MVP потрібні фахівці, які володіють технічними навичками та здатні швидко адаптуватися до специфіки роботи в умовах обмежених ресурсів. Процес підбору може включати кілька етапів: аналіз потреб компанії, формування вимог до кандидатів, пошук через професійні платформи (наприклад, LinkedIn),

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

оцінку резюме й технічні співбесіди [5]. Останнім часом дедалі популярнішим стає залучення світчерів – людей, які перейшли в IT із інших сфер через курси перекваліфікації. Такі кандидати часто демонструють високий рівень мотивації, що компенсує брак досвіду, а їхня інтеграція в команду підтримується наставництвом від більш досвідчених колег.

Питання мотивації відіграє ключову роль у менеджменті IT-команд. Успішні компанії поєднують матеріальні стимули (зарплата, бонуси за командні чи індивідуальні результати) з нематеріальними (тренінги, покращення умов праці, визнання досягнень). Наприклад, бонус за менторство для досвідчених працівників чи можливість відвідувати профільні конференції за рахунок компанії сприяють професійному зростанню й утриманню талантів [1]. У контексті MVP, де терміни стислі, мотивація через чітке розуміння цінності внеску кожного учасника може значно прискорити розробку.

Тісно пов'язаним із командною роботою аспектом є управління ризиками, яке вимагає від менеджера здатності передбачати й мінімізувати потенційні проблеми. Затримки в термінах, перевитрати бюджету чи технічні збої – часті виклики в розробці MVP, які можна зменшити завдяки резервному бюджету (10-15% від загальної суми) чи залученню тимчасових спеціалістів через платформи типу Upwork [3]. Технічні ризики, такі як помилки в коді, ефективно контролюються через регулярне тестування й резервне копіювання. Наприклад, використання автоматизованих тестів у Selenium може скоротити час на виправлення дефектів на 20-30%, що дозволяє дотримуватися дедлайнів. Для порівняння популярних інструментів менеджменту в контексті MVP наведено табл. 1.

Таблиця 1

Порівняння інструментів менеджменту для розробки MVP

Інструмент	Основні функції	Переваги для MVP	Обмеження
Jira	Трекінг задач, аналітика, інтеграція	Детальний контроль, прогнозування затримок	Складність для малих команд
Trello	Візуалізація (Kanban), простота інтерфейсу	Швидке налаштування, інтуїтивність	Обмежена аналітика
Asana	Планування, розподіл задач, таймлайн	Зручність для гібридних команд	Менше технічних інтеграцій
Confluence	Зберігання документації, спільна робота	Централізація знань	Потребує часу на освоєння

Джерело: розроблено автором

Ці інструменти та підходи набувають особливого значення в умовах обмежених ресурсів і високого тиску на швидкий випуск продукту, що характерно для MVP. Практичний приклад: невелика українська IT-компанія, працюючи над MVP для e-commerce стартапу, застосувала комбінацію Scrum і Trello, випустивши за три місяці базову версію з функціями каталогу товарів, кошика й оплати [2]. Щотижневий аналіз метрик, таких як швидкість команди (velocity) і кількість виконаних задач, дозволив уникнути перевитрат бюджету й отримати схвальні відгуки від перших користувачів.

У ширшому контексті менеджмент в IT зазнає трансформацій під впливом цифровізації, що особливо помітно в управлінні персоналом. Одним із ключових трендів є диджиталізація HR-процесів, яка охоплює не лише автоматизацію, а й оптимізацію завдяки соціальним, мобільним і хмарним технологіям [5]. Наприклад, автоматизовані інформаційні системи управління персоналом (HRIS) дозволяють вести облік кадрів, планувати навчання й аналізувати продуктивність у реальному часі, що підвищує ефективність командної роботи.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

У розробці MVP це означає швидше залучення потрібних спеціалістів і чіткіший контроль за їхньою діяльністю.

Підтримка командної взаємодії в умовах віддаленої роботи може відбуватись завдяки технологій типу Slack чи Microsoft Teams, що не забезпечують комунікацію і створюють комфортні умови для працівників, які працюють із різних локацій [4]. Це особливо актуально для кроскультурних команд, де учасники можуть перебувати в різних країнах і часових поясах. Наприклад, компанії, які розробляють MVP для міжнародних ринків, часто стикаються з необхідністю адаптувати методи управління до культурних і мовних відмінностей, що вимагає від менеджерів додаткових навичок і гнучкості.

Перспективним напрямом є розвиток неформальної освіти як джерела нових кадрів. В умовах дефіциту IT-фахівців компанії створюють власні програми підготовки, залучаючи студентів чи світчерів через інтенсивні курси [5]. Такі ініціативи, як корпоративні академії (наприклад, IT Academy від SoftServe), готують молодших спеціалістів (Junior), які можуть одразу долучатися до проєктів типу MVP. Це розширює кадровий потенціал та дозволяє формувати команди з урахуванням специфічних потреб проєкту.

Усе це підкреслює, що менеджмент в IT – це динамічна система, яка постійно еволюціонує під впливом технологій, ринкових запитів і людських факторів. Експерименти з AI-інструментами, такими як Toggl Insights для аналізу продуктивності чи AI-плагіни для Jira для прогнозування ризиків, стають частиною цього процесу [5]. Компанії, які інвестують у розвиток управлінських підходів і культури, досягають кращих результатів у зростанні, утриманні клієнтів і впровадженні інновацій. Для MVP це означає швидший випуск продукту та закладання міцного фундаменту для його подальшого масштабування, що підтримується як технічною базою, так і злагодженою командою.

Висновки і перспективи подальших досліджень. Отже, проведене дослідження підтверджує, що правильний менеджмент є фундаментом успішної розробки мінімально життєздатного продукту (MVP) в IT-компаніях. Ефективне управління забезпечує не лише дотримання термінів і бюджету, а й високу якість продукту, що досягається завдяки чіткому плануванню, гнучким методологіям (Agile, Scrum, Kanban), використанню сучасних інструментів (Jira, Trello, Confluence) і злагодженій командній роботі. Особливу роль відіграє управління людськими ресурсами: структурований онбордінг, мотивація через матеріальні та нематеріальні стимули, а також адаптація до віддаленого формату роботи створюють умови для продуктивності навіть у стислі терміни, характерні для MVP. Компанії, які поєднують ці елементи з проактивним управлінням ризиками (резервний бюджет, автоматизоване тестування), досягають кращих результатів, отримуючи позитивний зворотний зв’язок від перших користувачів і закладаючи основу для подальшого масштабування продукту.

Перспективи подальших досліджень відкривають широке поле для аналізу. Цікавим напрямом є вивчення впливу конкретних інструментів менеджменту (наприклад, AI-плагінів для прогнозування ризиків) на ефективність розробки MVP в українських IT-компаніях із врахуванням їхньої специфіки, як-от обмеженість ресурсів. Також доцільно дослідити, як кроскультурні відмінності в розподілених командах впливають на швидкість і якість створення продукту, розробивши рекомендації для адаптації управлінських практик. Окремий інтерес становить аналіз ролі неформальної освіти та корпоративних програм підготовки (наприклад, IT Academy) у формуванні кадрового резерву для MVP-проєктів, з оцінкою їхньої економічної ефективності та впливу на плинність кадрів. Нарешті, варто розглянути, як диджиталізація HR-процесів, зокрема автоматизація рекрутингу й онбордінгу, може оптимізувати роботу малих команд, пропонуючи науково обґрунтовані підходи до впровадження таких технологій у реальних умовах.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Список використаних джерел

1. Алярова А. Розвиток технологій менеджменту персоналу в ІТ-сфері // Економіка і організація управління. – 2023. – С. 135–145.
2. Волох Д. С. Налаштування процесів та проектний менеджмент у ІТ компанії, на прикладі компанії GlobalLogic : кваліфікац. робота магістра за спеціальністю «073 – менеджмент» / Д. С. Волох ; Терноп. нац. техн. ун-т. – Тернопіль : ТНТУ, 2023. – 70 с.
3. Заярнюк О., Сторожук О., Сокуренко О. Організація ІТ-бізнесу: особливості менеджменту в контексті соціальної відповідальності в умовах війни // Економіка та суспільство. – 2024. – № 64.
4. Кубявка Л. Б., Латишева Т. В. Концепція побудови і принципи управління проєктного та продуктового ІТ-менеджменту // Управління розвитком складних систем. – 2024. – № 57. – С. 45–50.
5. IT Management [Електронний ресурс] // IBM. – Режим доступу: https://www.ibm.com/think/ topics/it-management. – Дата звернення: 10.04.2025.

Роботу виконано під науковим керівництвом д-ра техн. наук, професора
ЦЮЦЮРИ С. В.

ОПТИМІЗАЦІЯ АЛГОРИТМІВ СКАНУВАННЯ ТА ВІЗУАЛІЗАЦІЇ СТРУКТУРИ ФАЙЛОВОЇ СИСТЕМИ ДЛЯ ВЕЛИКИХ ОБСЯГІВ ДАНИХ

ПОТРУХ А. С., 2 курс 2м група ФІТ ДТЕУ,
освітня програма « Інженерія програмного забезпечення»

Стаття присвячена вирішенню проблеми неефективності існуючих програмних засобів для візуального аналізу дискового простору при роботі з великими обсягами даних (терабайти). Проаналізовано недоліки традиційних алгоритмів сканування файлових систем (залежність від I/O, високе споживання пам'яті) та методів візуалізації. Запропоновано комплексний підхід до оптимізації, що включає багатопотокове асинхронне сканування файлової системи, використання оптимізованої деревоподібної структури даних для мінімізації споживання пам'яті, та застосування методу візуалізації Treemap з техніками агрегації дрібних елементів, рівнів деталізації (LOD) та відсікання невидимих елементів. Розроблено програмний прототип та проведено експериментальне дослідження, яке підтвердило значне (у 3-5 разів) скорочення часу сканування та суттєве (на 30-50%) зменшення пікового споживання пам'яті порівняно з популярними аналогами, а також покращення інтерактивності візуалізації для великих наборів даних.

This work addresses the inefficiency of existing software tools for visual disk space analysis when dealing with large data volumes (terabytes). The shortcomings of traditional file system scanning algorithms (I/O dependency, high memory consumption) and visualization methods are analyzed. A comprehensive optimization approach is proposed, incorporating multithreaded asynchronous file system scanning, the use of an optimized tree-based data structure to minimize memory consumption, and the application of the Treemap visualization method enhanced with techniques for aggregating small items, Levels of Detail (LOD), and culling invisible elements. A

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

software prototype was developed, and experimental research was conducted, confirming a significant (3-5 times) reduction in scanning time and a substantial (30-50%) decrease in peak memory consumption compared to popular analogues, as well as improved visualization interactivity for large datasets.

Сучасний етап розвитку інформаційних технологій характеризується експоненційним зростанням обсягів даних, що зберігаються на локальних та мережевих накопичувачах. Ємність жорстких дисків та твердотільних накопичувачів вимірюється терабайтами, що ставить нові виклики перед програмним забезпеченням для аналізу та управління дисковим простором. Існуючі інструменти, хоч і функціональні, часто демонструють недостатню продуктивність – тривалий час сканування, високе споживання оперативної пам'яті та низьку швидкість відгуку інтерфейсу при роботі з накопичувачами терабайтного класу. Це суттєво ускладнює завдання моніторингу, пошуку зайвих даних та ефективного використання дискового простору як для звичайних користувачів, так і для системних адміністраторів.

Проблема ефективного аналізу та візуалізації дискового простору, особливо в умовах постійного зростання обсягів даних, є предметом активних досліджень як у міжнародній, так і у вітчизняній науковій спільноті. Огляд актуальних публікацій дозволяє виділити ключові напрямки оптимізації алгоритмів сканування файлових систем та методів візуалізації їх структури.

Значна частина досліджень зосереджена на прискоренні процесу сканування, що є первинним та часто найбільш часовитратним етапом аналізу. Kim H. та Lee S. [2] досліджують оптимізацію обходу каталогів (directory traversal) для створення масштабованих систем індексації файлів, що є критичним для швидкого доступу до метаданих без повного повторного сканування. Yan J. та ін. [5] пропонують прискорені алгоритми сканування, спеціально розроблені для файлових систем великого обсягу (high-volume), ймовірно, використовуючи методи паралельної обробки або оптимізації операцій вводу-виводу.

Вітчизняні дослідники також активно працюють над цією проблематикою. Козак В. Г. [9] розглядає застосування адаптивних структур даних, які можуть динамічно підлаштовуватися під характеристики сховища для пришвидшення сканування. Кравченко Д. Ю. [10] досліджує ефективність високопродуктивного кешування метаданих файлової системи для зменшення часу доступу при повторних скануваннях або аналізах. Литвиненко С.І. [11] зосереджується на паралельних алгоритмах аналізу, зокрема в контексті хмарних середовищ, де розподілена обробка може дати значний приріст продуктивності. Пономаренко А.Ю. [12] пропонує методи побудови індексів файлової системи, що дозволяють здійснювати високошвидкісне сканування за рахунок попередньої обробки та структурування інформації.

Ефективне представлення результатів сканування є не менш важливим завданням. Bashir S. та ін. [1], Zhang L. та ін. [6], Shalaginov A. та Franke K. [4].

Фундаментом для будь-якого програмного забезпечення візуального аналізу дискового простору є розуміння структури сучасних файлових систем. Такі системи, як NTFS (New Technology File System), APFS (Apple File System), ext4 (Fourth Extended Filesystem) та Btrfs (B-tree File System), хоч і відрізняються внутрішньою реалізацією, мають спільну рису – ієрархічну організацію даних у вигляді дерева каталогів та файлів. Ключову роль відіграють метадані, що зберігаються для кожного об'єкта файлової системи: ім'я, розмір, дати створення та модифікації, атрибути, права доступу та, що важливо для аналізу простору, інформація про фізичне розташування даних на носії. Спосіб зберігання цієї інформації, наприклад, використання таблиць типу Master File Table (MFT) в NTFS чи інодів (inodes) в Unix-подібних системах, безпосередньо впливає на швидкість доступу до неї та, відповідно, на ефективність алгоритмів сканування.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

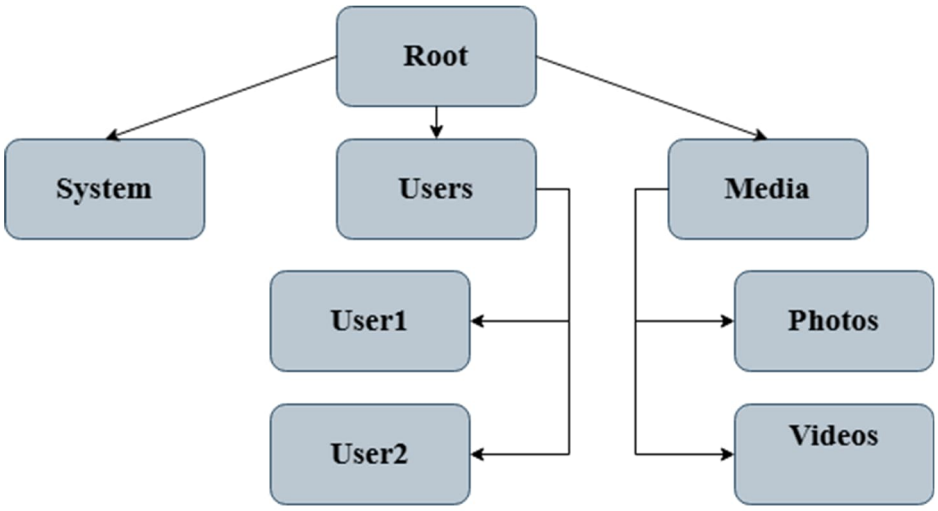


Рис. 1. Ієрархічна структура файлової системи

Джерело: власна розробка

Отримання повної картини використання дискового простору вимагає обходу цієї ієрархічної структури. Класичними підходами до сканування є рекурсивний та ітеративний обхід дерева каталогів. Рекурсивний метод, хоч і є концептуально простим та елегантним, може призводити до переповнення стека викликів при великій глибині вкладеності каталогів та неефективно використовувати ресурси. Ітеративний підхід, що зазвичай використовує стеки або черги для зберігання стану обходу, позбавлений проблеми переповнення стека, але обидва методи стикаються з фундаментальною проблемою: високою залежністю від операцій вводу-виводу (I/O). Доступ до метаданих кожного файлу та каталогу потребує звернень до дискової підсистеми, що є найповільнішою ланкою процесу, особливо на традиційних жорстких дисках (HDD). Обчислювальна складність таких алгоритмів у найгіршому випадку пропорційна загальній кількості файлів та каталогів у файловій системі, що робить їх неприйнятно повільними для накопичувачів терабайтного масштабу з мільйонами файлів.

Рис. 2. Порівняння рекурсивного та ітеративного обходу

Джерело: власна розробка

На ринку існує значна кількість програмних інструментів, призначених для візуального аналізу дискового простору, таких як WinDirStat, TreeSize, GrandPerspective, Baobab, ncdu та

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

інші. Вони використовують різноманітні методи візуалізації для представлення ієрархії та пропорційного розподілу місця. Найбільш поширеними є Treemap (деревоподібна карта), Sunburst (сонячні промені), Icicle Plot (діаграма-бурулька), а також традиційні списки або деревоподібні структури з сортуванням за розміром. Щодо алгоритмів сканування, більшість цих інструментів, ймовірно, використовують оптимізовані версії базових рекурсивних чи ітеративних підходів, однак їхня конкретна реалізація часто не документується детально. Незважаючи на свою функціональність, багато з цих популярних рішень демонструють суттєві недоліки при роботі з великими дисками: процес сканування може тривати десятки хвилин або навіть години, споживання оперативної пам'яті стає надмірним (особливо при зберіганні інформації про кожен файл), а інтерфейс користувача може «залежати» під час обробки або рендерингу візуалізації.

Вибір методу візуалізації також суттєво впливає на сприйняття та ефективність аналізу, особливо для великих ієрархій. Методи, такі як Treemap та Sunburst, добре підходять для відображення пропорцій та ієрархії одночасно, дозволяючи візуально ідентифікувати великі файли та каталоги. Однак їхня масштабованість може бути обмеженою: при дуже великій кількості елементів (файлів) візуалізація стає перенасиченою дрібними деталями, що ускладнює інтерпретацію, а процес її побудови та рендерингу може стати ресурсоємним. Circle Packing та Icicle Plots також представляють ієрархію, але можуть мати свої обмеження щодо ефективного використання простору екрану або чіткості відображення пропорцій для елементів низького рівня. Масштабованість візуалізації – це не лише здатність обробити велику кількість даних, але й зберегти інтерактивність та інформативність представлення.

Для подолання описаних вище проблем продуктивності при скануванні та обробці великих обсягів даних у файлових системах розроблено низку підходів до оптимізації. Ключовими напрямками є використання паралелізму та асинхронності. Паралельне сканування дозволяє одночасно обробляти різні гілки дерева каталогів, використовуючи переваги багатоядерних процесорів. Асинхронні операції вводу-виводу дають змогу не блокувати основний потік програми під час очікування відповіді від диска, перекриваючи час очікування виконанням інших завдань. Кешування метаданих файлів та каталогів, отриманих під час попередніх сканувань або операційною системою, може значно зменшити кількість необхідних дискових операцій. Іншим потужним підходом є індексація, яка полягає у створенні спеціальних структур даних (індексів), що містять попередньо зібрану інформацію про файлову систему та дозволяють виконувати запити щодо використання простору значно швидше, ніж повне сканування. Застосування цих методів оптимізації є необхідною умовою для створення програмного забезпечення, здатного ефективно аналізувати сучасні накопичувачі великої ємності.

На основі аналізу проблем, зокрема низької продуктивності існуючих рішень при роботі з великими обсягами даних, цей розділ присвячено проектуванню оптимізованих алгоритмів сканування файлової системи та візуалізації її структури. Метою проектування є досягнення суттєвого прискорення процесу аналізу та зменшення споживання системних ресурсів.

Першочерговим завданням є оптимізація процесу сканування, оскільки саме він є основним джерелом затримок через інтенсивні операції вводу-виводу. Для подолання цього «вузького місця» пропонується застосування комбінації підходів. По-перше, передбачається активне використання багатопотоковості для розпаралелювання обходу дерева каталогів. Замість послідовного обходу, різні гілки файлової системи можуть оброблятися одночасно окремими потоками виконання, що дозволяє ефективно задіяти ресурси сучасних багато-ядерних процесорів. По-друге, для мінімізації блокувань потоків під час очікування даних від дискової підсистеми, пропонується інтегрувати асинхронні операції вводу-виводу (Async I/O). Це дозволить ініціювати операцію читання метаданих каталогу чи файлу і продовжувати виконання інших завдань (наприклад, обробку вже отриманих даних або ініціацію інших

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

запитів), не чекаючи її завершення, що особливо критично для повільних HDD або при роботі з великою кількістю дрібних файлів.

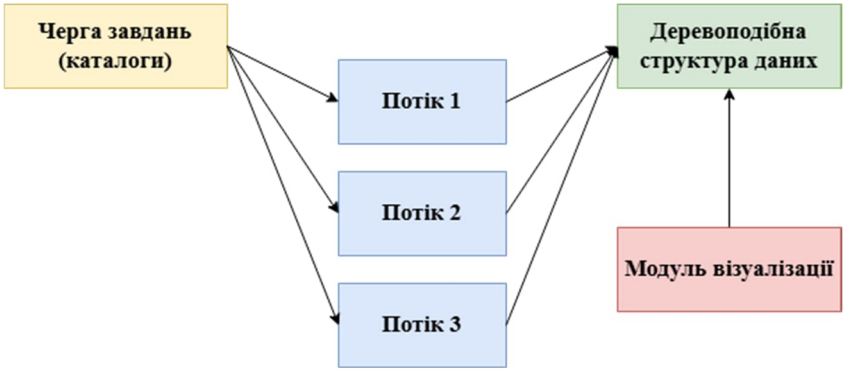


Рис. 3. Алгоритм паралельного сканування файлової системи

Джерело: власна розробка

Критично важливим аспектом оптимізації є також розробка ефективної структури даних для зберігання проміжної інформації про файли та каталоги під час сканування та для подальшої візуалізації. Зберігання повного шляху та всіх метаданих для кожного з мільйонів файлів може призвести до надмірного споживання оперативної пам'яті. Тому пропонується використовувати деревоподібну структуру, яка природно відображає ієрархію файлової системи. Кожен вузол дерева представлятиме файл або каталог, зберігаючи лише необхідні метадані (ім'я, розмір, тип вузла) та посилання на батьківський та дочірні вузли. Розмір каталогу обчислюватиметься агреговано на основі розмірів його вмісту. Це дозволить значно зменшити обсяг необхідної пам'яті порівняно зі зберіганням списку об'єктів з повними шляхами. Запропонований оптимізований алгоритм сканування починається з кореневого каталогу, який додається до черги завдань. Пул робочих потоків бере завдання з черги. Кожен потік, отримавши каталог, асинхронно запитує список його вмісту. Після отримання списку, для кожного елемента створюється відповідний вузол у загальній деревоподібній структурі даних (з використанням механізмів синхронізації для забезпечення потокобезпечності). Розмір файлів додається до сукупного розміру батьківського каталогу (атомарно). Підкаталоги додаються назад до черги завдань. Процес триває, доки черга завдань не стане порожньою і всі асинхронні операції не будуть завершені.

Після завершення сканування та побудови внутрішньої моделі даних постає завдання їх ефективної візуалізації. З огляду на необхідність одночасного відображення ієрархії та пропорційного використання простору, як основний метод візуалізації обрано Treemap (деревоподібна карта). Цей метод добре зарекомендував себе для представлення ієрархічних даних, однак для роботи з великою кількістю елементів потребує оптимізації. Для вирішення проблеми перенасичення візуалізації дрібними елементами та зменшення обчислювальної складності побудови макету пропонується застосування агрегації дрібних файлів та каталогів. Елементи, розмір яких є меншим за певний поріг (абсолютний або відносний до розміру батьківського елемента чи площі на екрані), будуть об'єднуватися в один узагальнений вузол (« Інші файли»), що суттєво зменшить кількість прямокутників для розрахунку та відображення. Додатково, для забезпечення плавності взаємодії та зменшення навантаження на систему візуалізації, буде реалізовано механізм рівнів деталізації (Level of Detail - LOD). При початковому відображенні або при віддаленні будуть візуалізуватися лише верхні рівні ієрархії або агреговані вузли, а деталізація глибших рівнів відбуватиметься лише при наближенні (зумі) або за вимогою користувача.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Оптимізація рендерингу є наступним кроком. Для прискорення відображення великої кількості прямокутників Treemap планується використання техніки відсікання невидимих елементів (culling), коли обчислюються та відображаються лише ті елементи, які потрапляють у поточну видиму область (viewport). Для подальшого підвищення продуктивності рендерингу розглядається можливість використання графічного процесора (GPU) через відповідні графічні API (наприклад, OpenGL, DirectX, Vulkan), що дозволить значно прискорити процес малювання великої кількості простих геометричних фігур (прямокутників). Запропонований алгоритм генерації та відображення візуалізації отримує на вхід побудовану на етапі сканування деревоподібну структуру даних. На основі цієї структури та поточного рівня деталізації (LOD) і налаштувань агрегації виконується розрахунок геометрії Treemap (наприклад, за алгоритмом Squarified Treemap). Розраховані координати та розміри прямокутників передаються підсистемі рендерингу, яка, застосовуючи відсікання невидимих елементів та потенційно GPU-прискорення, виводить візуалізацію на екран.

Архітектура програмного рішення може бути модульною для забезпечення гнучкості та можливості незалежної розробки й тестування компонентів. Основними модулями будуть: Модуль Сканування, відповідальний за багатопотоковий асинхронний обхід файлової системи та збір метаданих; Модуль Моделі Даних, що зберігає оптимізовану деревоподібну структуру даних та надає інтерфейси доступу до неї; Модуль Візуалізації, який реалізує алгоритми побудови Treemap (з агрегацією та LOD) та взаємодіє з підсистемою рендерингу; Модуль Рендерингу, що безпосередньо відповідає за відображення графічних примітивів (з оптимізаціями); та Модуль Інтерфейсу Користувача (UI), що забезпечує взаємодію з користувачем та керує роботою інших модулів. Модуль Сканування буде асинхронно наповнювати Модель Даних, яка, в свою чергу, може сповіщати Модуль Візуалізації про оновлення для поступового відображення прогресу або результатів.

Для обміну даними між модулями, зокрема між Модулем Сканування та Модулем Моделі Даних/Візуалізації, потрібно використовувати чітко визначений внутрішній формат даних. Цей формат представлений у вигляді програмних об’єктів (класів або структур), що описують вузол дерева файлової системи. Кожен об’єкт вузла має містити мінімально необхідний набір атрибутів: ім’я, 64-бітний розмір, тип (файл/каталог/агрегований вузол), посилання на батьківський та дочірні вузли. Цей формат буде оптимізований для швидкого доступу та мінімального споживання пам’яті під час роботи програми.

Ключові аспекти програмної реалізації включають створення ефективної структури даних для представлення ієрархії файлової системи. Необхідно розробити класову ієрархію вузлів (Node), що представляють файли та каталоги, де кожен вузол зберігав ім’я, 64-бітний розмір (для файлів) або агрегований розмір (для каталогів), тип вузла та вказівники на батьківський та дочірні елементи. Для мінімізації споживання пам’яті використовуються динамічні контейнери (std::vector або QList) для дочірніх вузлів, що виділяли пам’ять лише за потребою. Паралельне сканування реалізується за допомогою пулу потоків (QThreadPool), якому передаються завдання на обробку каталогів. Кожне завдання асинхронно отримує список вмісту каталогу засобами операційної системи та створє відповідні вузли у спільній структурі даних, використовуючи м’ютекси (std::mutex) для захисту критичних секцій під час оновлення агрегованих розмірів батьківських каталогів та додавання нових вузлів. Оптимізація рендерингу візуалізації Treemap здійснюється шляхом реалізації алгоритму Squarified Treemap для розрахунку геометрії та застосування відсікання невидимих елементів у методі малювання віджета Qt (paintEvent). Розрахунки координат виконуються лише для видимих на даний момент рівнів ієрархії відповідно до реалізованого механізму рівнів деталізації (LOD), а малюються лише ті прямокутники, які перетиналися з видимою областю віджета.

Для оцінки продуктивності розробленого програмного прототипу використовується спеціально розроблена методика експериментального дослідження. Тестове середовище

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

включає персональний комп’ютер із процесором Intel Core i7 (8 ядер), 16 ГБ оперативної пам’яті та двома типами накопичувачів: швидким NVMe SSD і традиційним HDD об’ємом 2 ТБ. Експерименти проводяться під керуванням операційної системи Windows 10.

Тестові набори даних формуються як на основі реальних користувацьких даних (системний диск, диск із медіафайлами), так і за допомогою спеціального скрипту, що генерує ієрархічні структури файлів різного розміру (від 100 ГБ до 1.5 ТБ), кількості файлів (від сотень тисяч до десятків мільйонів) та глибини вкладеності каталогів. Створюються набори з переважно великими файлами, а також із великою кількістю дрібних файлів, аби оцінити вплив структури даних на продуктивність.

Основними метриками для оцінки є:

- час сканування файлової системи (від початку до завершення побудови внутрішньої моделі даних);
- пікове споживання оперативної пам’яті (RAM) під час сканування та побудови моделі;
- час побудови візуалізації Treemap (після завершення сканування);
- суб’єктивна оцінка часу відгуку інтерфейсу (плавність навігації, масштабування) або вимірювання частоти кадрів (FPS) під час взаємодії з візуалізацією.

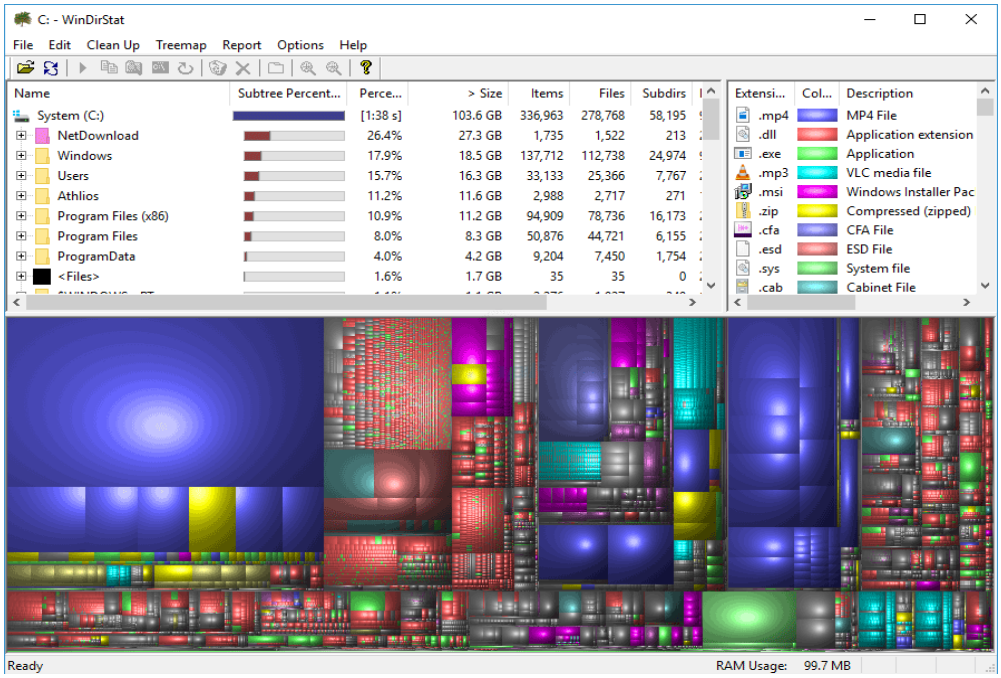


Рис. 4. Візуалізація дискового простору в WinDirStat

Джерело: [10]

Вимірювання часу здійснюється за допомогою std::chrono, а контроль споживання пам’яті – за допомогою засобів моніторингу операційної системи. Як базові рішення для порівняння використовуються популярні утиліти WinDirStat та TreeSize Free (остання – у режимі без використання MFT-сканування для забезпечення чистоти порівняння алгоритмів обходу).

Аналіз отриманих експериментальних даних дозволяє робити висновки щодо ефективності запропонованих та реалізованих оптимізацій. Порівняння продуктивності розробленого прототипу з аналогами (WinDirStat, TreeSize Free) на великих наборах даних (понад 500 ГБ, мільйони файлів) демонструє суттєві переваги запропонованого рішення.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Зокрема, час сканування скорочується в середньому у 3-5 разів порівняно з WinDirStat та у 2-4 рази – з TreeSize Free на SSD-накопичувачах. На HDD-накопичувачах перевага виявляється ще більш вираженою, що свідчить про ефективність використання асинхронних операцій вводу-виводу для подолання затримок дискової підсистеми.

Пікове споживання оперативної пам’яті під час сканування також залишається значно нижчим (на 30-50%) завдяки використанню оптимізованої деревоподібної структури даних без зберігання повних шляхів до кожного файлу.

Час побудови візуалізації Treemap і відгук інтерфейсу також покращуються, особливо при роботі з наборами даних, що містять велику кількість елементів, де агрегація дрібних файлів і використання рівнів деталізації (LOD) забезпечують високу інтерактивність і швидку взаємодію.

Рис. 5. Treemap-візуалізація в TreeSize Free

Джерело: [11]

Слід відзначити й певні обмеження розробленого рішення. Реалізація зосереджується на файлових системах NTFS та частково ext4, тоді як підтримка інших систем (APFS, Btrfs) потребує додаткової адаптації.

Наразі не використовуються специфічні оптимізації, можливі для окремих файлових систем (наприклад, пряме читання MFT для NTFS, що застосовується в деяких комерційних інструментах для ще більшого прискорення, але вимагає підвищених прав доступу – адміністративних).

Механізм агрегації та рівнів деталізації (LOD) у візуалізації може приховувати деякі важливі елементи, що небажано для частини аналітичних завдань.

Шляхами подолання цих обмежень є:

- розширення підтримки файлових систем;
- реалізація опціональних режимів сканування (включаючи прямий доступ до структур файлової системи);
- надання користувачеві гнучких налаштувань параметрів агрегації та LOD.

Проведений аналіз предметної області та існуючих програмних рішень підтвердив недостатню продуктивність та високу ресурсоємність традиційних підходів при обробці накопичувачів терабайтного класу, що зумовлено переважно залежністю від операцій вводу-виводу та неефективним управлінням пам’яттю.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

На основі огляду сучасних методів оптимізації було спроектовано та реалізовано програмний прототип, що інтегрує комплексний підхід до прискорення аналізу дискового простору. Запропонований алгоритм сканування використовує багатопотоковість для розпаралелювання обходу каталогів та асинхронні операції вводу-виводу для мінімізації блокувань, а також оптимізовану деревоподібну структуру даних для зменшення споживання оперативної пам'яті. Для візуалізації застосовано метод Treemap з реалізацією механізмів агрегації дрібних елементів, рівнів деталізації (LOD) та відсікання невидимих елементів для забезпечення масштабованості та інтерактивності.

Список використаних джерел

1. Bashir S., Arif M., Shah M. Efficient hierarchical file system visualization for big data analytics // IEEE Access. – 2021. – Vol. 9. – P. 78211–78223. DOI: 10.1109/ACCESS. 2021.3083314.
2. Kim H., Lee S. Optimizing directory traversal for scalable file indexing systems // ACM Transactions on Storage (TOS). – 2023. – Vol. 19, No. 2. – P. 1–22. DOI: 10.1145/3582725.
3. Shalaginov A., Franke K. Visualization of large-scale hierarchical file systems using clustering techniques // Journal of Big Data. – 2023. – Vol. 10. – Article No. 52. DOI: 10.1186/s40537-023-00709-z.
4. Yan J., Wang P., Zhou Y. Accelerated scanning algorithms for high-volume file systems // Future Generation Computer Systems. – 2020. – Vol. 113. – P. 135–144. DOI: 10.1016/j.future. 2020.07.001.
5. Zhang L., Wu Q., Ren K. Scalable visualization of complex file system hierarchies // IEEE Transactions on Visualization and Computer Graphics. – 2024. – Vol. 30(2). – P. 933–945. DOI: 10.1109/TVCG.2023.3316728.
6. Козак В. Г. Адаптивні структури даних для сканування великих сховищ інформації // Комп'ютерно-інтегровані технології: освіта, наука, виробництво. – 2022. – № 42. – С. 72–78.
7. Кравченко Д. Ю. Високопродуктивне кешування при скануванні файлових систем великих обсягів // Наукові праці ОНАХТ. – 2020. – № 3. – С. 111–117.
8. Литвиненко С. І. Паралельні алгоритми аналізу структури файлових систем у хмарних середовищах // Системи обробки інформації. – 2020. – № 1(155). – С. 105–110.
9. Пономаренко А. Ю. Методи побудови індексації для високошвидкісного сканування файлових систем // Наука та прогрес транспорту. – 2023. – № 1(49). – С. 33–41.
10. Електронний ресурс – Режим доступу: https://www.snapfiles.com/screenshots/windirstats.htm?utm_source=chatgpt.com
11. Електронний ресурс – Режим доступу: https://www.jam-software.com/treesize_free/ screens.shtml?utm_source=chatgpt.com

Роботу виконано під науковим керівництвом канд. екон. наук, доцента
ПАЛАГУТИ К. О.

ІНТЕРАКТИВНА ВІЗУАЛІЗАЦІЯ КОСМІЧНИХ ЗАПУСКІВ

САВЕНКО Д. Д., 2 курс 2мз група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті висвітлюються питання популяризації космічних досліджень шляхом використання інтерактивної візуалізації запусків ракет та аналізу історичних даних про космічні місії. Розглянуто можливості подання складної наукової інформації у доступному форматі для широкої аудиторії, а також проаналізовано сучасні технології, які сприяють спрощенню доступу до таких даних. Окрему увагу приділено інструментам створення інтерактивних інтерфейсів, методам обробки й візуального представлення інформації, що

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

дозволяють простежити тенденції розвитку аерокосмічної галузі. Також стаття акцентує на потенціалі подібних рішень як ефективного освітнього ресурсу, здатного зацікавити молодь і мотивувати її до навчання та професійної діяльності у сфері космічних технологій.

This article explored the popularization of space exploration through interactive visualizations of rocket launches and the analysis of historical launch data. It addressed approaches for making information about space missions more accessible and engaging for the general public, while also examining how modern technologies can facilitate this process. The study highlighted tools for developing interactive interfaces that support historical inquiry, as well as methods for processing and presenting data to reveal key trends in the evolution of the space industry. Additionally, the article emphasized the potential of such technological solutions to serve as educational resources that inspire public interest and encourage careers in the aerospace field.

Актуальність. У сучасному світі космічні дослідження та пов'язані з ними технології відіграють важливу роль у забезпеченні науково-технічного прогресу. Водночас, незважаючи на відкритий доступ до інформації про аерокосмічну сферу, для більшості людей ця галузь залишається віддаленою, складною для розуміння й малопривабливою. Зростання кількості космічних місій та досягнень не супроводжується відповідним зростанням зацікавлення з боку широкої аудиторії, оскільки інформація часто подається у надто технічному форматі, що ускладнює її сприйняття. Така ситуація негативно впливає на рівень інтересу до космічної тематики серед молоді та потенційних спеціалістів, що у довгостроковій перспективі може спричинити кадровий дефіцит у галузі та нерозуміння значущості космічних досліджень.

Інтерактивна візуалізація даних про минулі, поточні та майбутні космічні запуски є важливим інструментом не лише для науковців, а й для освітніх установ та широкої громадськості. Такий підхід сприяє глибшому аналізу історичних тенденцій розвитку космічної галузі, роблячи інформацію про космічні місії більш доступною та зрозумілою. Використання сучасних технологій для створення інтерактивних візуалізацій сприяє популяризації космічних досліджень і здатне заохотити нові покоління до вивчення цієї тематики та побудови кар'єри в аерокосмічній сфері.

У межах цього дослідження передбачається розробка інтуїтивно зрозумілого інтерфейсу для візуалізації даних на основі аналізу історичних, актуальних та прогнозованих запусків. Такий підхід покликаний підвищити доступність і зрозумілість інформації, що, своєю чергою, сприятиме зростанню зацікавлення космічною тематикою серед широкої аудиторії.

Метою статті є дослідження історичних, поточних та прогнозованих даних про космічні запуски з метою популяризації космічних досліджень серед широкої аудиторії шляхом застосування інтерактивної візуалізації. У межах дослідження передбачається аналіз тенденцій у сфері космічних місій, огляд сучасних технологій обробки та представлення даних, а також розробка інтерфейсу, що сприятиме підвищенню доступності й зрозумілості інформації про аерокосмічну галузь.

Предмет дослідження це методи інтерактивної візуалізації історичних, поточних та майбутніх даних про космічні запуски, технології їх обробки та представлення, а також стратегії популяризації космічних досліджень через сучасні цифрові платформи та інструменти.

Об'єктом дослідження є процес створення та аналізу даних, пов'язаних з космічною галуззю, а також використання інтерактивної візуалізації для підвищення обізнаності та інтересу до космічних досліджень серед широкої аудиторії.

Аналіз попередніх досліджень. У сучасних працях, присвячених візуалізації космічних даних та історичних запусків, особлива увага приділяється таким роботам, як «OpenSpace: A Visualization Tool for Exploring NASA Datasets» [1], «Exploratory Spatio-Temporal Visualization: An Analytical Review» [2], «Research on Space Launch Visualization Simulation Analysis Technology and Application» [3], «Immersive and Interactive Visualization of 3D Spatio-Temporal Data in Virtual Reality» [4], «The Popularization of Space Exploration Amongst High School Students» [5], «The Symbiosis of the Space Sciences and Popularization Activities» [6], «Educational Outreach in the Field of Space

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Science» та «Space Exploration and Education: Engaging the Public in the 21st Century» [7][10].

Виклад основного матеріалу. Візуалізація космічних даних і запусків.

Візуалізація даних, зокрема через інтерактивні інтерфейси, є важливим інструментом для розуміння складних процесів, таких як космічні місії та запуски ракет. Цифрова візуалізація робить ці дані доступнішими та зрозумілішими для ширшої аудиторії, що має велике значення для освітніх і популяризаційних цілей. Одним із таких інструментів є OpenSpace – платформа для вивчення космічних даних, що дозволяє інтерактивно досліджувати інформацію про космічні місії та запуски ракет NASA (див. рис. 1). Цей інструмент надає користувачам можливість зануритися в космічні дослідження, роблячи їх доступними та цікавими навіть для тих, хто не є фахівцями в аерокосмічній галузі, але має бажання дізнатися більше [1].

Рис. 1. Wow Room простір в OpenSpace

Джерело: [1]

Однією з основних проблем при створенні інтерактивних інтерфейсів для космічних запусків є не тільки забезпечення точності даних, а й здатність подати цю інформацію так, щоб вона була зрозумілою для користувачів. Важливо не тільки зберегти технічну точність, але й надати можливість активної взаємодії з даними. Це дозволяє користувачам глибше досліджувати різні аспекти запуску: тип ракети, вплив на навколишнє середовище, склад команди та інші фактори. Одним із основних підходів до точного представлення складних явищ, таких як рух космічних апаратів і їхні траєкторії, є використання методів просторово-часової візуалізації, які дозволяють створювати детальні й динамічні моделі цих процесів [2].

Іншим важливим аспектом є можливість отримання та обробки великих обсягів даних. Потрібно застосовувати технології для візуалізації та симуляції запусків ракет, які дозволяють інтерактивно представляти історію космічних місій, поточні місії та прогнозувати майбутні запуски на основі отриманих даних. Такі технології надають можливість для глибокого аналізу та розумінню космічних запусків і для залучення до цієї теми широкої аудиторії [3].

Інтерактивні технології для візуалізації космічних запусків.

Сучасні інтерактивні технології дозволяють створювати не просто сухі візуальні дані, а реалістичні та захопливі інтерфейси, які допомагають простіше та краще розуміти складні процеси запуску ракет, їх виходу на орбіту, повернення місії додому або зрозуміти де і яка зараз місія у реальному часі. Найперспективнішим напрямом відтворення даних космічних запусків є технології віртуальної реальності (VR) та доповненої реальності (AR). Технології віртуальної чи доповненої реальності надають можливість подивитись на запуск ракети або на

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

місію вже у космосі у будь-якому місці та у будь-який час і можливість подивитись на це з різних сторін завдяки взаємодії із додатком. Тобто, людині потрібно лише відкрити мобільний додаток, щоб подивитись, як це виглядає. Використання VR чи AR для представлення просторово-часових даних дозволяє користувачам інструменту зануритися в середовище космічної галузі та в реальному часі взаємодіяти з візуалізацією [4].

Мобільні додатки не є єдиним напрямом розвитку технологій інтерактивних інтерфейсів для космічної галузі або для будь-якої іншої. Крім них є веб-орієнтовані візуалізації, які роблять доступ до даних про космічні місії зручніше, оскільки є можливість аналізувати дані без необхідності завантаження спеціального програмного забезпечення, потрібно лише перейти на веб-сайт. Використання технологій, таких як WebGL, Three.js, Babylon.js чи D3.js, дозволяє створювати інтерактивну 3D-графіку, використовуючи 3D-моделі. Такі підходи до візуалізації складних часових та просторових даних можуть бути використані для аналізу польотів та траєкторій ракет [2].

Окрім 3D-бачення самих запусків, ще важливим є створення симуляцій, що дозволяють відтворювати реальні сценарії космічних місій. Було проаналізовано застосування таких симуляторів для прогнозування траєкторій польотів, для прогнозування обсягу та оптимізації ресурсів, та для навчання персоналу, який працює в аерокосмічній сфері. Симулятори можуть бути використані у навчальних цілях та в ознайомчих цілях, допомагаючи студентам, дослідникам або просто зацікавленим людям краще та легше зрозуміти принципи роботи космічних шатлів [3].

Обробка та аналіз даних космічних запусків.

Наступним важливим критерієм є ефективна обробка даних про космічні місії, оскільки вони містять дуже велику кількість параметрів, які потрібно проаналізувати та представити у зручній формі. У сучасних дослідженнях широко застосовуються технології машинного навчання, а також алгоритми великих даних Big Data, щоб аналізувати історичні записи про запуски для прогнозування майбутніх місій та їх оптимізацію.

Дослідження OpenSpace представляють систему, яка дозволяє представляти та аналізувати великі обсяги даних, які отримуються від місій та супутників NASA [1]. Цей інструмент об’єднує реальні дані про орбіти, параметри польоту та телеметрію, створюючи інтерактивні моделі, які допомагають дослідникам ідентифікувати ключові закономірності.

Також важливим критерієм є автоматизована класифікація історичних даних про космічні запуски. Методи обробки просторово-часових даних можуть бути використані для аналізу регіонів запусків і для аналізу частоти запусків, щоб змоделювати вплив зовнішніх факторів на запуски, таких як погода чи технічні аспекти, наприклад, будь-які несправності ракети [2].

Іншим важливим аспектом є здатність обробляти великі обсяги даних, що генеруються під час космічних запусків. Для цього необхідно використовувати передові технології візуалізації та симуляції, які дозволяють інтерактивно відображати історичні дані про космічні місії, відстежувати поточні запуски та прогнозувати майбутні місії на основі зібраної інформації. Такі технології не тільки забезпечують глибокий аналіз та розуміння процесів космічних запусків, але й роблять цю тему доступною для широкої аудиторії, залучаючи більше людей до цієї захоплюючої області [3].

Невід’ємною частиною навчання молоді космічній науці є використання інтерактивних інструментів, які дозволяють візуалізувати дані, отримані від космічних місій чи супутників, що сприяє залученню студентів до цієї сфери. Одним із успішних прикладів таких інструментів є віртуальні лабораторії, де учні мають можливість моделювати траєкторії польотів, аналізувати історичні місії та проводити експерименти в зручному для них форматі, що робить процес навчання більш доступним і захоплюючим [5].

Питання взаємозв’язку між науковими дослідженнями та їхньою популяризацією вирішується через забезпечення відкритого доступу до космічних даних і надання простих та

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

зручних інструментів для їхньої візуалізації. Це дозволяє не лише залучати широку аудиторію до вивчення аерокосмічної науки, а й сприяє більш глибокому дослідженню космічних запусків та місій. Така доступність і зручність інструментів є важливими складовими розвитку інтересу до космічної галузі серед широкого кола людей [6].

Наступним ефективним підходом є створення ігрових симуляцій, які дозволяють користувачам самостійно налаштовувати параметри запуску та аналізувати результати своїх рішень. Це дає можливість громадськості активно брати участь у випробуваннях через інтерактивні веб-платформи, мобільні додатки та VR-симуляції. Використання таких інтерактивних інструментів та освітніх платформ значно полегшує доступ до складної інформації про космічні місії, що, у свою чергу, сприяє залученню нових поколінь до аерокосмічної науки [7].

Перспективи розвитку інтерактивних технологій у космічних дослідженнях. Інтерактивні технології продовжують розвиватися, відкриваючи нові можливості для візуалізації та аналізу космічних даних. Інтеграція віртуальної та доповненої реальності з космічними симуляціями дозволяє користувачам глибше вивчати траєкторії польотів, планетарні ландшафти та роботу космічних апаратів. Окрім цих технологій, активно застосовуються методи штучного інтелекту для обробки великих обсягів даних. Штучний інтелект здатен автоматично сортувати і аналізувати дані, значно спрощуючи роботу дослідників. Поєднання віртуальної та доповненої реальності з технологіями штучного інтелекту не тільки покращує обробку даних, а й забезпечує зручну візуалізацію цих даних, що дозволяє дослідникам швидко отримувати 3D-моделі та аналізувати інформацію в реальному часі [8].

Великі космічні агентства, такі як NASA та ESA, активно працюють над розширенням доступу до відкритих даних для ширшої аудиторії. Згідно з доповіддю NASA, залучення громадськості до роботи з космічними даними через відкриті платформи не тільки сприяє виявленню нових наукових закономірностей, але й допомагає зацікавити молодь у науково-технічних дисциплінах, стимулюючи інтерес до кар'єри в аерокосмічній галузі. Такий доступ до даних є важливим кроком у розвитку науки та залученні нових поколінь до космічних досліджень [9].

Розвиток квантових обчислень та нових методів зберігання даних у найближчому майбутньому дозволить обробляти ще більші обсяги інформації про космічну галузь, що відкриває нові горизонти для вивчення аерокосмічної сфери, інтерактивної візуалізації космосу, космічних запусків, аналізу даних та моделювання майбутніх місій. Це створить можливість для розробки більш потужних та зручних освітніх проектів, які сприятимуть зростанню популяризації космічної галузі серед широкої аудиторії.

Висновки. Впровадження інтерактивних технологій, таких як віртуальна реальність і доповнена реальність, відіграє одну з ключових ролей у візуалізації даних, дослідженні даних та популяризації аерокосмічної галузі. Це значно спрощує доступ до великої та складної інформації та полегшує її розуміння для широкої аудиторії. Запровадження технологій машинного навчання та хмарних технологій дозволяє ефективніше аналізувати історичні космічні запуски для прогнозування майбутніх запусків і оптимізувати поточні місії. Це сприяє не тільки науковим дослідженням, але й освітнім ініціативам, допомагаючи фахівцям і зацікавленим користувачам краще розуміти процеси космічної галузі. Інтерактивні платформи, освітні проекти, симулятори та відкритий доступ до даних сприяють залученню нових поколінь дослідників та інженерів до аерокосмічної сфери. Подальший розвиток цих технологій сприятиме як поглибленню наукових знань, так і зрісту зацікавленості космічною тематикою серед широкої аудиторії, що зробить космічну галузь ближчою та цікавою.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Список використаних джерел

1. OpenSpace: A Visualization Tool for Exploring NASA Datasets // Режим доступу: https://zenodo.org/records/11122224/files/OpenSpace_%20A%20visualization%20tool%20for%20exploring%20NASA%20datasets.pdf (останнє звернення 03.04.2025 р.)
2. Exploratory Spatio-Temporal Visualization: An Analytical Review // Режим доступу: https://www.sciencedirect.com/science/article/pii/S1045926X03000466 (останнє звернення 03.04.2025 р.)
3. Research on Space Launch Visualization Simulation Analysis Technology and Application // Режим доступу: https://www.china-simulation.com/EN/10.16182/j.issn1004731x.joss.20-0887 (останнє звернення 03.04.2025 р.)
4. Immersive and Interactive Visualization of 3D Spatio-Temporal Data in Virtual Reality // Режим доступу: https://www.frontiersin.org/journals/bioinformatics/articles/10.3389/fbinf.2023.998991/full (останнє звернення 03.04.2025 р.)
5. The Popularization of Space Exploration Amongst High School Students // Режим доступу: https://fiztan.phd.elte.hu/letolt/popularization_kovacs.pdf (останнє звернення 03.04.2025 р.)
6. The Symbiosis of the Space Sciences and Popularization Activities // Режим доступу: https://www.sciencedirect.com/science/article/pii/B9780128173909000130 (останнє звернення 03.04.2025 р.)
7. Educational Outreach in the Field of Space Science // Режим доступу: https://saturnspacelaunch-system.com/educational-outreach/ (останнє звернення 03.04.2025 р.)
8. AI-Powered Data Processing for Space Missions // Режим доступу: https://science.nasa.gov/science-research/science-enabling-technology/new-ai-algorithms-streamline-data-processing-for-space-based-instruments/ (останнє звернення 03.04.2025 р.)
9. Open Data and Citizen Science in Space Exploration // Режим доступу: https:// science. nasa. gov/biological-physical/a-one-stop-spot-for-space-data-nasas-open-science-repository/ (останнє звернення 03.04.2025 р.)
10. Space Exploration and Education: Engaging the Public in the 21st Century // Режим доступу: https://www.tsijournals.com/articles/space-exploration-and-its-impact-on-science-education-and-public-engagement-16456.html (останнє звернення 03.04.2025 р.)

Роботу виконано під науковим керівництвом PhD, доцента
ДЕСЯТКО А. М.

АКТУАЛЬНІСТЬ ВИКОРИСТАННЯ МОВИ ПРОГРАМУВАННЯ PHP У СУЧАСНІЙ ВЕБРОЗРОБЦІ

САЛІВОН Д. В., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті здійснено аналіз ключових переваг і недоліків мов програмування Ruby, Go, Python, а також платформи Node.js. З метою комплексного порівняння було проведено аналітичне зіставлення цих технологій із мовою програмування PHP у контексті розробки динамічних вебсторінок. Для ілюстрації практичного застосування розглянуто створення програмного модуля інформаційної системи книжкового інтернет-магазину як приклад типового вебресурсу, що реалізується за допомогою зазначених інструментів.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

The article presents an analysis of the key advantages and disadvantages of the programming languages Ruby, Go, Python, as well as the Node.js platform. For the purpose of comprehensive comparison, an analytical juxtaposition of these technologies with the PHP programming language was conducted in the context of dynamic web page development. As a practical example, the creation of a software module for an online bookstore information system was examined, representing a typical web resource implemented using the aforementioned tools.

Актуальність дослідження зумовлена стрімким розвитком галузі веброзробки, яка за відносно короткий історичний період сформувала широкий спектр технологічних рішень, здатних задовольнити найрізноманітніші потреби розробників і забезпечити ефективне вирішення комплексних завдань. У такому контексті постає природне питання: наскільки конкурентоспроможною залишається мова програмування PHP у порівнянні з більш сучасними альтернативами? Саме пошук відповіді на це запитання є головною метою даної роботи.

Для проведення обґрунтованого порівняння було обрано сегмент розробки динамічних вебсторінок, що вирізняється високим попитом і відповідно сприяє зростанню конкуренції серед засобів розробки. В якості прикладного кейсу було проаналізовано процес створення програмного модуля інформаційної системи книжкового інтернет-магазину. Такий вибір зумовлений як високою репрезентативністю (що підтверджується значною кількістю вже реалізованих аналогічних рішень), так і універсальністю підходу, що дає змогу інтегрувати у розробку інші мови програмування поряд із PHP.

Метою даної статті є здійснення порівняльного аналізу мови програмування PHP з іншими популярними мовами, які активно застосовуються у сфері вебпрограмування, з акцентом на визначенні її сучасної релевантності та конкурентоспроможності.

Об'єктом дослідження виступає процес використання мови PHP у розробці динамічних вебсторінок.

Предметом дослідження є сама мова програмування PHP як інструмент створення серверної логіки вебзастосунків.

У рамках аналізу попередніх досліджень було розглянуто наукові публікації, написані англійською мовою, авторства таких дослідників, як Nawroze Islam, Md Rubel Chowdhury, Ілля Белоус, Ілля Прокопенко та Zadhid Powell.

Виклад основного матеріалу. Вибір мови програмування є критично важливим етапом у процесі створення будь-якого програмного забезпечення, оскільки він безпосередньо впливає на обсяг необхідних ресурсів, часові витрати та загальну складність реалізації проєкту. Правильний вибір технологічного інструментарію на початковому етапі розробки має довготривалі наслідки, адже чим ближче програмний продукт до завершення, тим складнішими, затратнішими та ризикованішими стають зміни – особливо такі фундаментальні, як заміна мови програмування. В умовах наявності широкого спектра мов і платформ, кожна з яких має свої унікальні характеристики та переваги, завдання вибору оптимального рішення для конкретного проєкту стає надзвичайно складним і потребує глибокого технічного аналізу.

На початковому етапі доцільно окреслити сутність мови програмування PHP. PHP (Hypertext Preprocessor) – це скриптова мова, що переважно використовується для розробки серверної частини вебдодатків. Вона виконується на стороні сервера і генерує динамічний вміст, який передається клієнтському браузеру у вигляді HTML-коду. Синтаксис PHP має подібність до таких мов, як C та Java, що сприяє його легкому засвоєнню, особливо для тих, хто вже має базові знання в інших мовах програмування. Основне призначення PHP полягає у створенні динамічних вебсторінок, зміст яких формується залежно від запиту користувача або зовнішніх умов [1]. Завдяки можливості вбудовування у HTML-код та інтерпретації браузером результатів його виконання, PHP забезпечує зручний механізм інтеграції з інтерфейсною

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

частиною вебзастосунків. Мова вирізняється простотою у використанні, стабільною продуктивністю, яка підтримується регулярними оновленнями та розвитком екосистеми. Хоча сфера застосування PHP є відносно вузькою, статистичні дані свідчать про його стійке лідерство у цій ніші: станом на 2025 рік близько 74,7% усіх вебсайтів, які використовують серверні технології, реалізовані саме з використанням PHP (рис. 1).

	2024 1 Mar	2024 1 Apr	2024 1 May	2024 1 Jun	2024 1 Jul	2024 1 Aug	2024 1 Sep	2024 1 Oct	2024 1 Nov	2024 1 Dec	2025 1 Jan	2025 1 Feb	2025 1 Mar
PHP	76.5%	76.4%	76.3%	76.2%	76.2%	76.1%	75.9%	75.8%	75.7%	75.4%	75.2%	74.9%	74.7%
Ruby	5.7%	5.8%	5.8%	5.9%	5.9%	5.9%	5.9%	6.0%	6.0%	6.1%	6.2%	6.2%	6.2%
ASP.NET	6.4%	6.3%	6.2%	6.1%	6.0%	5.9%	5.8%	5.7%	5.6%	5.5%	5.4%	5.4%	5.3%
Java	4.8%	4.8%	4.8%	4.8%	4.9%	4.9%	4.9%	5.0%	5.0%	5.0%	5.0%	5.1%	5.1%
JavaScript	3.2%	3.3%	3.3%	3.4%	3.4%	3.5%	3.5%	3.6%	3.7%	3.9%	4.0%	4.1%	4.2%
Scala	3.0%	3.0%	3.1%	3.1%	3.2%	3.2%	3.4%	3.5%	3.5%	3.7%	3.8%	3.9%	4.0%
static files	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.7%	1.7%	1.7%	1.7%	1.7%	1.7%
Python	1.4%	1.4%	1.4%	1.4%	1.4%	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%
ColdFusion	0.3%	0.3%	0.3%	0.3%	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%
Perl	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%
Erlang	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%

Рис. 1. Відсоткове зображення використання мов програмування для реалізації серверної частини вебсторінок в проміжку від 1 березня 2024 року до 1 березня 2025 року

Джерело: https://w3techs.com/technologies/history_overview/programming_language

Після попереднього аналізу PHP доцільно перейти до розгляду його основних конкурентів. Однією з найближчих за сферою застосування мов програмування є Ruby, яка впродовж тривалого часу зберігає стабільну популярність у сфері веброзробки та виступає прямим конкурентом PHP у створенні динамічних вебресурсів. Однією з ключових переваг Ruby є надзвичайно зручний та лаконічний синтаксис, який сприяє написанню чистого, добре структурованого та легкого для читання коду. Ця особливість істотно спрощує процес подальшої підтримки програмного забезпечення, внесення змін і налагодження, навіть у межах малих проєктів. Варто зазначити, що як PHP, так і Ruby характеризуються швидким темпом розробки завдяки простому для засвоєння синтаксису, однак Ruby демонструє перевагу в аспекті чистоти кінцевого програмного коду. Незважаючи на менший обсяг спільноти користувачів, Ruby не поступається за кількістю й якістю доступної документації завдяки високій активності своїх прихильників. Водночас Ruby має певні недоліки, зокрема пов’язані з продуктивністю: як за швидкістю виконання, так і за ефективністю використання оперативної пам’яті, ця мова програє PHP. Крім того, PHP є дещо простішим для початкового вивчення, тоді як Ruby краще підходить для більш досвідчених розробників, які здатні повною мірою реалізувати його потенціал [1].

Наступною мовою програмування, що заслуговує на розгляд, є Python. Вона є не лише одним із провідних інструментів у вебпрограмуванні, а й входить до переліку найпопулярніших мов програмування у глобальному масштабі. Згідно з результатами щорічного опитування, проведеного авторитетним онлайн-ресурсом Stack Overflow, близько 51% із понад 60 тисяч респондентів зазначили, що користувалися або планують користуватися мовою Python упродовж наступного року (рис. 2). Такий показник підтверджує стабільно високий попит на Python як серед початківців, так і серед професіоналів, а також її універсальність у застосуванні в різних сферах розробки.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

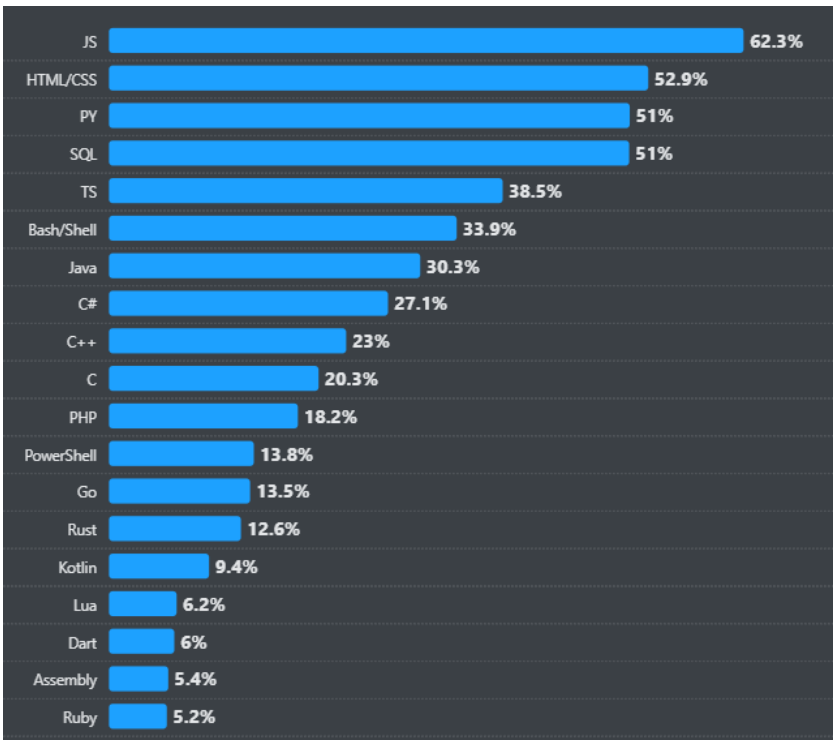


Рис. 2. Результати опитування 60, 171 респондентів у 2024 році інтернет ресурсом Stack Overflow

Джерело: https://survey.stackoverflow.co/2024/technology#most-popular-technologies-language

Водночас варто зазначити, що згідно з тими ж статистичними даними, мова програмування PHP і надалі зберігає високий рівень популярності серед розробників. Вона поступається лише найбільш поширеним мовам, таким як C, Java та Python, що підтверджує її стійкі позиції на ринку веброзробки. Зокрема, PHP суттєво випереджає Ruby, яка, попри свою орієнтацію на розробку динамічних вебресурсів, залишається вузькоспеціалізованим інструментом. Такий результат є показовим і свідчить про стабільну затребуваність PHP у своїй ніші.

Загалом, у процесі порівняльного аналізу слід відзначити низку переваг мови програмування Python. Зокрема, вона демонструє вищу швидкість обробки запитів та підвищену продуктивність у сфері машинного навчання, особливо у випадку реалізації складних моделей. Python також характеризується однією з найбільших та найактивніших спільнот користувачів, що забезпечує широку доступність навчальних матеріалів, документації та прикладних рішень. Крім того, синтаксис мови є надзвичайно простим для засвоєння, що сприяє її популярності серед початківців. Водночас PHP має перевагу в ефективності використання оперативної пам’яті – у деяких випадках обсяг споживаних ресурсів є в 2–3 рази меншим, ніж у Python. До того ж PHP забезпечує розробникам вищу гнучкість у програмному контролі, що дозволяє більш точно керувати логікою додатка [2].

Ще однією мовою програмування, що заслуговує на окрему увагу, є Go (або Golang), яка була спочатку розроблена компанією Google як внутрішній інструмент, а згодом стала доступною як програмне забезпечення з відкритим вихідним кодом з офіційною підтримкою від розробника. Основною перевагою Go є надзвичайно висока продуктивність, що робить її однією з провідних мов у сфері веброзробки з точки зору швидкодії. Важливо зазначити, що Go не належить до об’єктно орієнтованих мов програмування, не підтримує тернарні оператори та механізм успадкування, що в одних випадках сприяє створенню більш чистого та лаконічного коду, проте в інших – обмежує свободу розробника у виборі архітектурних рішень. У зв’язку з цим Go більше підходить для досвідчених розробників, ніж, наприклад,

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Ruby, оскільки вимагає глибшого розуміння його концепцій. Значна відмінність Go від більш традиційних мов, таких як Java або C, також ускладнює процес його освоєння. Цей ефект посилюється обмеженою кількістю спеціалізованих фреймворків, оскільки концепція Go передбачає, що розробник самостійно адаптує середовище під потреби конкретного проєкту. Таким чином, мова демонструє надзвичайну ефективність у великих системах за рахунок компромісу зручності. Крім того, Go використовує статичну типізацію, що передбачає обов'язкове оголошення типу змінної при її створенні, що багатьма професіоналами сприймається як перевага з точки зору підвищення продуктивності та поліпшення читабельності коду [3].

У межах даного дослідження існує широкий спектр мов програмування, які могли б бути розглянуті, проте більшість з них демонструють схожі характеристики з уже проаналізованими прикладами. У зв'язку з цим доцільно завершити огляд платформою Node.js. Варто підкреслити, що Node.js не є ані мовою програмування, ані фреймворком – це середовище виконання, яке дозволяє використовувати мову JavaScript для розробки серверної логіки вебдодатків. У стандартній конфігурації JavaScript виконується лише у клієнтському браузері, однак з використанням Node.js з'являється можливість реалізовувати серверну частину додатків із застосуванням єдиного синтаксису, що суттєво спрощує архітектурне проєктування та знижує технічну складність інтеграції.

Однією з ключових переваг Node.js є його висока продуктивність, що досягається завдяки ефективному розподілу навантаження між кількома ядрами центрального процесора, оптимізованому використанню оперативної пам'яті, а також підтримці асинхронного виконання запитів. Це дозволяє уникнути блокування всієї системи у разі затримки обробки окремого запиту, а також забезпечує можливість паралельної обробки запитів. За цими параметрами Node.js демонструє вищу продуктивність порівняно з PHP, Ruby та Python, хоча поступається Go.

Водночас платформа має низку істотних недоліків. Зокрема, Node.js характеризується частими змінами в API, що нерідко призводить до втрати зворотної сумісності з попередніми версіями, змушуючи розробників постійно адаптувати свої проєкти під нові вимоги. Крім того, ефективне використання Node.js потребує ґрунтовного знання JavaScript, без якого опанування цієї технології є значно ускладненим. Це контрастує з іншими мовами програмування, які можна вивчати незалежно від JavaScript. Хоча спільнота Node.js активно зростає, за чисельністю вона все ще поступається спільноті PHP [4].

Узагальнюючи аналіз розглянутих у статті мов програмування, варто зауважити, що їх безпосереднє порівняння є доволі складним завданням через різні архітектурні підходи, парадигми та специфіку застосування. Саме тому для проведення порівняльного аналізу було обрано практичну площину – процес розробки динамічних вебсторінок, що є не лише одним із найбільш затребуваних напрямків сучасної веброзробки, а й таким, який може бути ефективно реалізований засобами кожної з розглянутих мов. У якості прикладу було проаналізовано створення програмного модуля інформаційної системи книжкового інтернет-магазину.

Вибір саме цього прикладного домену зумовлений актуальними тенденціями на ринку електронної комерції. Стрімке зростання кількості сервісів доставки та зміна споживчих звичок сприяли формуванню високого попиту на інтернет-магазини, які наразі є однією з найдинамічніших сфер розробки вебзастосунків. Це підтверджується і міжнародним контекстом: по всьому світу, включаючи Україну, онлайн-магазини у формі вебсайтів та мобільних застосунків демонструють стрімке зростання популярності. Зокрема, показовим прикладом є платформа Temu, яка за короткий проміжок часу досягла значної популярності, підтверджуючи актуальність подібних проєктів для розробників.

Тематика книжкової торгівлі була обрана з огляду на її прикладну доцільність як для ринку, так і з технологічної точки зору. Книжки як товар мають чітко структуровану та уніфіковану інформацію – назва, автор, жанр, рік видання, опис, ISBN-код, обкладинка тощо.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Ці дані, як правило, доступні у відкритих джерелах, що значно спрощує наповнення бази даних та автоматизацію процесів розробки. Такий контент дозволяє створювати функціональну та інформативну структуру вебінтерфейсу, наповнюючи сторінки корисною інформацією, яку можна вдало поєднувати з візуальними елементами (зокрема, обкладинками книжок), що покращує користувацький досвід. Варто також зазначити, що попит на книжкову продукцію в Україні демонструє позитивну динаміку. Згідно з дослідженням, проведеним видавництвом «Книголав» у 2023 році, спостерігається суттєве зростання інтересу до літератури різних жанрів серед широкої аудиторії читачів [5].

У підсумку слід зазначити, що кожна з розглянутих мов програмування має свої унікальні переваги та недоліки, які визначають їхню доцільність у певних умовах розробки. Усі ці мови – PHP, Python, Ruby, Go та платформа Node.js – можуть ефективно застосовуватися для створення динамічних вебсторінок із використанням різноманітних баз даних, оскільки кожна з них має вбудовану підтримку широкого спектра систем управління базами даних. Остаточний вибір інструментарію часто залежить не лише від технічних характеристик, а й від особистих уподобань розробника.

У контексті динамічних вебресурсів багато параметрів, таких як продуктивність або архітектурна складність, мають другорядне значення порівняно з масштабнішими проєктами. Зважаючи на те, що всі проаналізовані технології здатні задовольнити сучасні вимоги у межах типових вебзастосунків, вибір мови лише на основі продуктивності не є обґрунтованим для невеликих або середніх проєктів. Саме цей чинник, разом із простотою у вивченні, дозволив PHP зберегти свою популярність протягом багатьох років. Якщо раніше, до виходу версії PHP 7 у 2015 році, продуктивність була одним із головних недоліків мови, то внаслідок численних оновлень ситуація суттєво змінилася. Сучасні версії PHP демонструють конкурентоспроможні показники швидкодії та ефективності використання ресурсів.

Окремо слід підкреслити важливість фактора легкості навчання, який відіграє ключову роль, особливо для початківців. PHP традиційно вирізняється низьким порогом входу та простим синтаксисом, що робить його доступнішим порівняно з Node.js, Go, а також дещо легшим за Ruby. У свою чергу, Ruby і Go, хоча й складніші у засвоєнні, пропонують вищий рівень чистоти та читабельності коду, що цінується досвідченими розробниками.

Значною перевагою PHP є наявність великої кількості популярних фреймворків, активної та численної спільноти, а також потужної бази навчальних ресурсів. Це створює сприятливі умови для швидкого старту та ефективного розширення знань. Із розглянутих технологій подібну підтримку можуть запропонувати лише Python – як одна з найбільш універсальних і затребуваних мов програмування – та платформа Node.js, яка останнім часом демонструє стрімке зростання популярності серед веброзробників.

Крім зазначених переваг, мова програмування PHP вирізняється високим рівнем сумісності з більшістю сучасних операційних систем, широкою підтримкою різноманітних систем управління базами даних та здатністю до інтеграції з іншими мовами програмування. Зокрема, PHP може ефективно працювати у зв’язці з JavaScript, Go та іншими технологіями, що розширює можливості його застосування у мультиплатформенних проєктах.

У цілому PHP залишається одним із найбільш надійних рішень для розробки невеликих і середніх вебзастосунків, з яким складно конкурувати за критеріями простоти, стабільності та доступності. Обравши PHP, розробник може розраховувати на зрозумілу та перевірену часом технологію з великою кількістю інструментів для швидкого старту. Більш того, PHP можна ефективно застосовувати і у масштабних проєктах за умови використання сучасних фреймворків та поєднання з іншими мовами програмування.

Водночас, слід визнати, що PHP поступово втрачає позиції у сфері інноваційних технологій, зокрема в напрямках, пов’язаних із машинним навчанням, великими даними та іншими високонавантаженими обчислювальними задачами. У цих сегментах конкуренцію PHP складають більш сучасні мови та платформи. Проте у своїй традиційній ніші PHP продовжує демонструвати високу актуальність, підтверджуючи свою життєздатність і адаптивність до сучасних умов розробки.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Висновки. У ході дослідження було здійснено порівняльний аналіз мови програмування PHP з іншими актуальними мовами, що застосовуються у сфері веброзробки: Python, Ruby, Go та платформою Node.js. Встановлено, що кожна з цих технологій має свої сильні та слабкі сторони, які слід враховувати при виборі інструментів для реалізації динамічних вебзастосунків.

PHP продовжує зберігати високу актуальність, незважаючи на появу нових технологій. Його ключовими перевагами є простота вивчення, низький поріг входу, висока сумісність із більшістю операційних систем та баз даних, широка екосистема фреймворків, а також стабільна підтримка великої розробницької спільноти. Особливо ефективним PHP є у малих і середніх проєктах, де критичними є швидкість розробки, підтримуваність коду та доступність документації.

Хоча у сфері високонавантажених обчислень, машинного навчання чи мікросервісної архітектури PHP поступається таким мовам як Go чи Python, у своїй нішевій галузі він продовжує забезпечувати конкурентоспроможний рівень продуктивності та зручності.

Таким чином, мова програмування PHP залишається доцільним вибором для широкого кола вебпроєктів, особливо в тих випадках, коли важливими є стабільність, перевіреність технології та ефективна реалізація динамічного контенту.

Список використаних джерел

1. Ruby vs PHP: The Big Battle [Електронний ресурс]. – Режим доступу: https://sloboda-studio.com/blog/ruby-vs-php/ (дата звернення: 02.04.2025)
2. A Comparative Analysis of PHP and Python Programming Languages for Optimal Software Development [Електронний ресурс]. – Режим доступу: https://www.researchgate.net/publication/376807515_A_Comparative_Analysis_of_PHP_and_Python_Programming_Languages_for_Optimal_Software_Development (дата звернення: 02.04.2025)
3. Go vs PHP. Practical outlook [Електронний ресурс]. – Режим доступу: https:// medium.com/@emprovedev/go-vs-php-practical-outlook-69b814e0f564 (дата звернення: 02.04.2025)
4. Node.js vs PHP: A Head-to-Head Comparison [Електронний ресурс]. – Режим доступу: https://kinsta.com/blog/node-js-vs-php/ (дата звернення: 02.04.2025)
5. Дослідження книжкового ринку. Головними покупцями книжок в Україні є жінки, але чоловіки читають більше [Електронний ресурс]. – Режим доступу: https://life.nv.ua/ ukr/blogs/shcho-i-skilki-chitayut-ukrajinci-doslidzhennya-50357197.html (дата звернення: 02.04.2025)

Роботу виконано під науковим керівництвом канд. пед. наук, доцента
КОТЕНКО Н. О.

ПРОБЛЕМИ ЕТИЧНОСТІ ШТУЧНОГО ІНТЕЛЕКТУ ТА АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ

СКИБУК О. І., 2 курс 2м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті досліджуються етичні проблеми, пов'язані з використанням штучного інтелекту та алгоритмів машинного навчання. Розглядаються питання упередженості, прозорості, конфіденційності та відповідальності за прийняття рішень алгоритмами. Окреслено можливі шляхи вирішення зазначених проблем, включаючи регулювання, технічні методи та етичні стандарти.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

The article explores ethical issues related to the use of artificial intelligence and machine learning algorithms. Issues of bias, transparency, privacy, and responsibility for decision-making by algorithms are considered. Possible solutions to these problems are outlined, including regulation, technical methods, and ethical standards.

Актуальність. Штучний інтелект (ШІ) та алгоритми машинного навчання (МН) активно впроваджуються в різні сфери життя, змінюючи способи прийняття рішень та виконання завдань. Вони знаходять застосування у таких галузях, як медицина, транспорт, фінанси, промисловість, освіта та інші. Завдяки ШІ автоматизуються рутинні процеси, підвищується точність прогнозів та зростає ефективність роботи.

Проте разом із перевагами виникають значні виклики, пов'язані з етичною стороною застосування ШІ. Існує ризик дискримінації, коли алгоритми приймають рішення на основі упереджених даних, що може призводити до несправедливого ставлення до певних груп населення. Проблеми також виникають у контексті захисту приватності та конфіденційності даних, оскільки ШІ обробляє величезні масиви інформації, що створює ризик витоку персональних даних та порушення прав людини. Відсутність прозорості алгоритмів ускладнює розуміння процесу ухвалення рішень, що може призвести до непередбачуваних наслідків. Крім того, виникає питання відповідальності: хто несе відповідальність за помилки або несправедливі рішення, прийняті ШІ? Ці проблеми вимагають комплексного підходу та створення чітких етичних стандартів для забезпечення безпечного та справедливого впровадження ШІ в суспільство.

Аналіз попередніх досліджень. Проблематика етичності ШІ активно досліджується провідними науковцями та організаціями, які зосереджують увагу на різних аспектах цієї теми. Одним з найвідоміших досліджень є праця «Fairness and Abstraction in Sociotechnical Systems» [1] авторства Moritz Hardt та Solon Barocas, де розглядаються проблеми алгоритмічної упередженості та виклики створення справедливих алгоритмів. Вони підкреслюють, що спроби усунути упередженість часто наштовхуються на труднощі через складність соціально-технічних систем та неоднозначність визначення справедливості.

Команда дослідників з Google AI, включаючи Timnit Gebru та Margaret Mitchell, у статті «On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?» [2] аналізує ризики використання великих мовних моделей. Вони наголошують на загрозах упередженості та недостатньої прозорості алгоритмів, що може призвести до посилення стереотипів та поширення дезінформації. Особливу увагу приділено питанню відповідальності розробників за наслідки використання таких технологій.

Європейська комісія у своїх етичних настановах «Ethics Guidelines for Trustworthy AI» наголошує на необхідності забезпечення прозорості та підзвітності алгоритмів, пропонуючи підхід, що базується на принципах законності, етичності та надійності. Проте дослідження вказують на складність впровадження цих рекомендацій у реальні бізнес-процеси.

Важливі питання етики також розглядаються в праці Virginia Dignum «Responsible Artificial Intelligence: How to Develop and Use AI in a Responsible Way» [3]. Автор аналізує концепції відповідального використання ШІ, акцентуючи увагу на важливості дотримання етичних норм та принципів під час розробки алгоритмів. Вона зазначає, що сучасні етичні стандарти потребують уточнення та адаптації до швидкозмінного технологічного середовища.

З огляду на вищезгадані роботи, можна зробити висновок, що наявні дослідження хоча й пропонують певні рішення, однак стикаються з труднощами у практичному впровадженні. Проблеми справедливості, прозорості, приватності та відповідальності залишаються актуальними та вимагають подальшого вивчення.

Метою статті є виявлення основних етичних проблем, пов'язаних із використанням штучного інтелекту та алгоритмів машинного навчання, та запропонування можливих рішень цих проблем.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Об'єктом дослідження є технології штучного інтелекту та алгоритми машинного навчання в контексті їхнього впливу на етичні аспекти в суспільстві.

Предметом дослідження є етичні проблеми, які виникають у процесі розробки, впровадження та використання систем штучного інтелекту, зокрема пов'язані з упередженнями, прозорістю, конфіденційністю та відповідальністю за прийняття рішень.

Виклад основного матеріалу.

Алгоритми можуть демонструвати упередженість через неточність даних або помилки при налаштуванні моделі. Це системна проблема, яка може мати серйозні соціальні наслідки. Вона проявляється, коли алгоритми приймають рішення, що дискримінують певні групи людей через дефекти в тренувальних даних або логіці роботи моделі. Вона виникає через: неякісні тренувальні дані та некритичне використання історичних даних. Розглянемо кожне з цих питань більш детально.

Неякісні тренувальні дані. Наприклад, алгоритм розпізнавання обличчя, розроблений компанією Face++, показував точність 99% для білих чоловіків, але помилявся у 35% випадків для темношкірих жінок. Причина – у наборі даних переважали фото європеоїдів.

Некритичне використання історичних даних. Система штучного інтелекту, що аналізувала резюме для компанії Amazon, виявилася упередженою щодо жінок. Причиною стало використання історичних даних, у яких більшість керівних посад обіймали чоловіки. Подібні наслідки мала й система COMPAS, яка застосовується в судах США для оцінки ризику рецидивізму. Згідно з дослідженнями, вона на 45% частіше помилково класифікувала темношкірих обвинувачених як таких, що мають високий ризик повторного злочину. Це прямо пов'язано з тим, що модель тренувалася на даних, які вже містили расові упередження, властиві американській правоохоронній системі [4].

Для подолання подібних проблем сучасні компанії використовують кілька стратегій. По-перше, активно залучають представників різних соціальних груп до процесу розробки та тестування моделей. Наприклад, Microsoft реалізувала програму «AI for Accessibility», у межах якої особи з інвалідністю беруть участь у перевірці алгоритмів. По-друге, використовуються технології для штучного балансування даних. Зокрема, IBM розробила бібліотеку AIF360, яка автоматично вирівнює представленість різних соціальних груп у тренувальних вибірках. Крім того, дедалі більшої популярності набуває технологія adversarial debiasing, яка полягає у використанні спеціальних алгоритмів для виявлення й усунення упереджень. Як приклад можна навести інструмент Google's What-If Tool. Ще одним важливим підходом є створення етичних комітетів у компаніях-розробниках ШІ. Так, Microsoft має внутрішню групу AETHER (AI and Ethics in Engineering and Research), яка перевіряє алгоритми на наявність упередженості перед їхнім впровадженням.

Багато сучасних систем штучного інтелекту, особливо на основі глибокого навчання, залишаються непрозорими для користувачів. Їхня робота часто нагадує «чорну скриньку», що значно ускладнює аудит і контроль. Це створює ризики неприйнятних або навіть небезпечних рішень. Так, у фінансовому секторі алгоритми кредитного скорингу можуть відмовити у видачі позики без пояснення причин. У медицині модель IBM Watson Health давала хибні рекомендації щодо лікування раку, а у 2021 році автоматизований торговий алгоритм компанії Knight Capital за 45 хвилин згенерував збитки у 440 мільйонів доларів США через помилки в логіці роботи.

Для вирішення таких проблем впроваджується концепція Explainable AI (XAI), яка передбачає розробку моделей, здатних пояснити логіку власних рішень [5]. Наприклад, техніка LIME дозволяє представити результати складного алгоритму у вигляді простих для інтерпретації правил, тоді як деякі фінансові установи, як-от Banco Bilbao Vizcaya Argentaria, замінюють нейромережі на дерева рішень, що значно спрощує розуміння процесу ухвалення рішень. Додатково сприяють прозорості і спеціалізовані візуальні інструменти на кшталт TensorFlow Playground, які дають змогу в реальному часі відстежувати, як змінюються параметри нейронної мережі під час її навчання. Важливим кроком також є законодавче

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

регулювання – наприклад, положення «права на пояснення» у GDPR зобов'язує компанії надавати користувачам інформацію про автоматизовані рішення, що на них вплинули.

Збір і обробка великих обсягів інформації для навчання моделей ШІ супроводжується ризиком витоку конфіденційної інформації. Один із яскравих прикладів – штраф у розмірі 20 мільйонів євро, накладений на компанію Clearview AI за використання фотографій із соціальних мереж без згоди користувачів. Інша проблема – витоки конфіденційних даних під час роботи текстових моделей. Так, у 2023 році зафіксовано випадок, коли ChatGPT випадково відтворив номер кредитної картки, що був присутній у його навчальних даних. Крім того, розвиток технологій на основі генеративних моделей (GAN) відкриває можливості для створення фальшивих зображень людей, що створює нові загрози шахрайства.

Серед технологічних рішень цієї проблеми особливу роль відіграє принцип диференційної приватності. Його реалізація, зокрема Apple, дозволяє додавати до зібраних даних певний рівень «шуму», що унеможливлює ідентифікацію конкретної особи. Іншим інноваційним підходом є федеративне навчання, яке активно використовується компанією Google для покращення роботи клавіатури Gboard. У цьому випадку модель навчається безпосередньо на пристрої користувача, і дані не передаються на сервер. Захист також забезпечується через законодавчі обмеження – прикладом є Personal Information Protection Law у Китаї, що забороняє використання персональних даних без згоди користувача. Новітній підхід до захисту приватності – технологія «confidential computing», яка передбачає обробку даних у зашифрованому вигляді, навіть під час їх використання, завдяки спеціальним процесорам на кшталт Intel SGX.

Однією з найскладніших етичних дилем, пов'язаних із застосуванням ШІ, є розподіл відповідальності за рішення, ухвалені алгоритмом. На відміну від традиційних систем, де відповідальність покладається на оператора чи виробника, автономність і складність сучасних моделей ускладнює визначення винної сторони. Наприклад, у випадку з автономними автомобілями (Tesla Autopilot) або діагностичними медичними алгоритмами (IBM Watson) виникає питання: хто відповідає за помилку – розробник, оператор чи сама система?

У реальних кейсах це питання стояло особливо гостро. У 2018 році автономний автомобіль Uber під час тестування збив пішохода. Розслідування показало, що система не розпізнала людину вночі, а оператор у салоні не стежив за дорогою. Попри те, що обвинувачення було висунуто водію, громадськість визнала і відповідальність компанії. Інший випадок – судовий позов проти Zillow у 2022 році через алгоритм оцінки нерухомості, який систематично занижував ціни в окремих районах, спричиняючи фінансові втрати. Суд визнав компанію винною через недостатню прозорість роботи моделі.

Щоб зменшити ризики таких ситуацій, в Європі ухвалено директиву AI Liability Directive (2023), яка зобов'язує розробників підтверджувати безпечність своїх систем. У Великій Британії компанії, що застосовують автономні технології, повинні укладати спеціальні страхові договори. Також поширеним стає впровадження цифрових логів, що фіксують усі дії ШІ – за аналогією з «чорними скриньками» в авіації. Німеччина пропонує ще один цікавий підхід – впровадження «цифрових прав» для операторів ШІ, які підтверджують їхню компетентність у використанні складних технологій. У США в цьому напрямку активно працює організація Partnership on AI, яка, зокрема, у 2023 році запропонувала створити незалежні структури для розгляду скарг на алгоритмічну дискримінацію.

Ще однією важливою етичною проблемою у сфері штучного інтелекту є використання ШІ в контексті маніпуляції поведінкою та думками користувачів. Із розвитком алгоритмів рекомендацій, зокрема в соціальних мережах, виникає ризик так званого «ефекту інформаційної бульбашки». Алгоритми підбирають контент на основі попередньої активності користувача, що призводить до обмеження інформаційного середовища та посилення когнітивного упередження. Це сприяє поляризації суспільства, радикалізації поглядів і зменшенню відкритості до альтернативних точок зору.

З метою подолання цієї проблеми провідні платформи поступово впроваджують

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

принципи відповідального дизайну рекомендаційних систем: YouTube, наприклад, дозволяє користувачам переглядати причини появи рекомендованого відео, а TikTok – скидати налаштування алгоритму. Також активно розвивається алгоритмічна прозорість, що передбачає обов'язкове роз'яснення користувачу принципів формування інформаційної стрічки. Важливим інструментом стає і цифрова медіаграмотність – освітні програми та кампанії, що вчать користувачів критично мислити й перевіряти інформацію.

Окрему етичну дилему становить використання штучного інтелекту у військовій сфері. Так звані «летальні автономні системи» (LAWS) – бойові дрони, керовані алгоритмами, – здатні самостійно ідентифікувати цілі та приймати рішення про їх ураження без втручання людини. Це породжує питання легітимності таких рішень, відповідності міжнародному гуманітарному праву та потенційної втрати контролю над військовими технологіями. У відповідь на ці виклики деякі країни та міжнародні організації закликають до мораторію на розробку та використання повністю автономної зброї. Наприклад, ООН неодноразово порушувала питання про необхідність створення міжнародного договору щодо заборони або регулювання LAWS, а країни ЄС активно підтримують концепцію «Meaningful Human Control» – тобто забезпечення присутності людини в процесі ухвалення будь-якого бойового рішення [6].

Також у контексті етики виникають проблеми щодо використання ШІ в освіті, зокрема систем адаптивного навчання, автоматизованого оцінювання чи виявлення плагіату. Алгоритми можуть некоректно трактувати творчі або нестандартні відповіді, або ж знецінювати важливість міжособистісної взаємодії у навчальному процесі.

Ще одним напрямом занепокоєння є потенційне використання ШІ для емоційного впливу на людей – через генерацію персоналізованого контенту, здатного викликати конкретні емоції або рішення. Зокрема, алгоритми на базі аналізу міміки, інтонації чи мови тіла вже застосовуються у сфері маркетингу або під час онлайн-співбесід. Така практика може перетворитися на форму цифрової маніпуляції. У відповідь на це зростає попит на розробку етичних обмежень для емоційного ШІ (Affective AI), які мають запобігати зловживанням у комерційних або владних інтересах. Зокрема, деякі дослідницькі інститути, як-от The Alan Turing Institute, просувають ідею «емоційної автономії» користувача як базового принципу при дизайні таких систем.

Крім того, застосування ШІ в політичному контексті (наприклад, для мікротаргетування виборців або створення deepfake-відео) викликає занепокоєння з боку дослідників та правозахисників. Такі інструменти можуть використовуватися для маніпуляцій громадською думкою, дискредитації опонентів або поширення дезінформації. У 2024 році було зафіксовано випадки використання штучно згенерованих аудіо та відео в передвиборчих кампаніях у США та Індії, що суттєво ускладнило перевірку достовірності інформації. У відповідь на це розробляються системи виявлення deepfake-контенту на базі нейромереж, зокрема Meta та Microsoft інвестують у створення спеціалізованих детекторів, які здатні розпізнавати змінені або повністю синтезовані зображення, відео та аудіо.

Водночас технології ШІ вже сьогодні активно використовуються у сфері працевлаштування для попереднього відбору кандидатів, моніторингу продуктивності співробітників, аналізу поведінки тощо. Це створює ризик тотального цифрового нагляду, при якому працівники постійно перебувають під контролем алгоритмів. Така ситуація може призводити до психологічного тиску, зниження автономності працівника та порушення трудових прав. Для врегулювання цієї проблеми у низці країн впроваджуються етичні стандарти для HR-алгоритмів, а також запроваджується принцип Human-in-the-loop, який гарантує наявність людини на етапі прийняття остаточного рішення. У Європейському Союзі в межах регулятивної ініціативи AI Act обговорюється обов'язкове інформування працівників про використання ШІ та встановлення меж щодо допустимого рівня нагляду. Деякі компанії, зокрема IBM, вже запровадили внутрішні політики прозорого використання алгоритмів у

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

корпоративному середовищі.

Таким чином, етичні виклики, пов’язані з використанням ШІ, не обмежуються лише технічними чи правовими аспектами – вони мають глибокий соціальний вимір, що потребує міждисциплінарного підходу до їх аналізу та розв’язання. Участь етичних експертів, психологів, правників та представників громадянського суспільства у розробці й впровадженні ШІ-систем є запорукою формування безпечного та справедливого цифрового середовища.

Висновки Проблема етичності штучного інтелекту є багатовимірною і потребує міждисциплінарного підходу, що охоплює технологічні, правові та соціальні аспекти. Сучасний розвиток ШІ випереджає темпи формування нормативного регулювання та етичних стандартів, що створює потенційні ризики для прав людини, приватності та суспільної справедливості. Забезпечення етичності у сфері ШІ вимагає системних зусиль: розробки прозорих і справедливих алгоритмів, впровадження механізмів контролю та аудиту, адаптації законодавства до нових технологічних реалій. Особливої уваги потребують сфери з високим ступенем соціального впливу – медицина, правосуддя, освіта, де алгоритмічні рішення можуть мати критичні наслідки. Перспективи подальших досліджень полягають у формуванні міжнародних стандартів етичного використання ШІ, розробці механізмів підзвітності та етичного виховання розробників, що сприятиме створенню безпечних, інклюзивних і відповідальних цифрових технологій.

Список використаних джерел

1. Fairness and abstraction in sociotechnical systems / A. D. Selbst та ін. FAT* ‘19: conference on fairness, accountability, and transparency, м. Atlanta GA USA. New York, NY, USA, 2019. URL: https://doi.org/10.1145/3287560.3287598 (дата звернення: 11.03.2025).

2. On the dangers of stochastic parrots / E. M. Bender та ін. FAccT ‘21: 2021 ACM conference on fairness, accountability, and transparency, м. Virtual Event Canada. New York, NY, USA, 2021. URL: https://doi.org/10.1145/3442188.3445922 (дата звернення: 11.03.2025).

3. Gold N. E. Virginia dignum: responsible artificial intelligence: how to develop and use AI in a responsible way. Genetic programming and evolvable machines. 2020. URL: https://doi.org/10.1007/s10710-020-09394-1 (дата звернення: 12.03.2025).

4. Stahl B. C., Schroeder D., Rodrigues R. Ethics of artificial intelligence. Cham : Springer International Publishing, 2023. URL: https://doi.org/10.1007/978-3-031-17040-9 (дата звернення: 13.03.2025).

5. The ethics of algorithms: mapping the debate / B. D. Mittelstadt та ін. Big data & society. 2016. Т. 3, № 2. С. 205395171667967. URL: https://doi.org/10.1177/2053951716679679 (дата звернення: 15.03.2025).

6. European union’s ethics guidelines for ai. Trustworthy AI alone is not enough. 2023. С. 17–21. URL: https://doi.org/10.2307/jj.8500773.5 (дата звернення: 15.03.2025).

Роботу виконано під науковим керівництвом канд. пед. наук, доцента
КОТЕНКО Н. О.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

КОНВЕРСІЙНО-ОРІЄНТОВАНЕ ПРОЄКТУВАННЯ MVP: СИМБІОЗ ТЕХНІЧНИХ ТА МАРКЕТИНГОВИХ РІШЕНЬ

СТАРОДУБЕЦЬ В. В., 2 курс 4м група ФІТ ДТЕУ,
освітня програма «Управління проєктами програмних продуктів»

У статті розглядається концепція двовекторної структуризації під час розробки MVP (Minimal Viable Product) Landing Page страхової компанії. Розкриваються аспекти технічного та маркетингового векторів, їх значення та взаємодія. Описуються основні підходи до архітектури веб-рішення, вибору стеку технологій та UX/UI-дизайну. Запропоновано модель оптимізації процесу розробки MVP.

The article examines the concept of dual-vector structuring in the development of an MVP (Minimal Viable Product) Landing Page for an insurance company. The aspects of technical and marketing vectors, their importance, and interaction are revealed. The main approaches to web solution architecture, technology stack selection, and UX/UI design are described. A model for optimizing the MVP development process is proposed.

Актуальність. Розвиток цифрових технологій змінює підходи до розробки онлайн-рішень для бізнесу. Страхові компанії активно використовують Landing Page як інструмент для залучення клієнтів. Однак, ефективність таких рішень залежить від правильного поєднання маркетингових стратегій та технічної реалізації, що і становить предмет дослідження.

Сучасний споживач очікує швидкого доступу до інформації та зручності у взаємодії з онлайн-сервісами. Проблема полягає у невідповідності між високими очікуваннями користувачів та застарілими підходами до розробки веб-рішень, що може призводити до зниження конверсії та втрати потенційних клієнтів. Таким чином, актуальним є створення ефективної MVP-структури, яка дозволить швидко протестувати концепцію та адаптуватися до потреб ринку.

Ще одним важливим аспектом є SEO-оптимізація, яка впливає на видимість сайту у пошукових системах та конкурентоспроможність компанії. Неправильна реалізація SEO-стратегій може призвести до низької позиції у видачі Google, що зменшить трафік та ефективність залучення клієнтів. Важливим є врахування як технічного SEO (швидкість завантаження, структура сторінок, мета-теги), так і контентного SEO (ключові слова, релевантність тексту, внутрішні та зовнішні посилання). Розробка MVP має враховувати ці аспекти для досягнення максимальної ефективності.

Метою статті є дослідження структурної моделі MVP Landing Page страхової компанії з урахуванням двох основних векторів: маркетингового та технічного, а також SEO-оптимізації та аналізу конкурентів.

Об'єктом дослідження є розробка розробка структурної моделі MVP Landing Page страхової компанії.

Предмет дослідження - двовекторна структуризація процесу створення MVP Landing Page з урахуванням SEO та конкурентного аналізу.

Аналіз попередніх досліджень. Аналізуючи сучасні підходи до створення MVP Landing Page, можна виділити кілька ключових напрямків, розглянутих у попередніх наукових роботах.

Крістофер Гутер у книзі «Lean UX: Designing Great Products with Agile Teams» підкреслює важливість інтеграції UX-дизайну в гнучкі методології розробки. Він стверджує, що спільна робота дизайнерів та розробників дозволяє швидше адаптувати продукт до змін ринку, що є ключовим фактором у створенні MVP [1].

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Дослідження Нільсена та Лоренцена «Web Usability: The Basics» підтверджує, що структуроване проектування UX значно підвищує конверсію та задоволеність користувачів. Їхні висновки акцентують увагу на важливості використання принципів доступності та інтуїтивності у веб-дизайні [2].

Дослідження Google «The Mobile Playbook» вказує, що швидкість завантаження сайту є критичним фактором SEO-оптимізації та впливає на поведінкові метрики користувачів, зокрема показник відмов. Це підкреслює важливість технічного SEO у процесі створення MVP [3].

За даними Ahrefs у звіті «SEO Practices in 2023», якісна контентна стратегія та використання відповідних ключових слів значно впливають на ранжування сторінки у пошукових системах. Недостатня увага до цього аспекту може стати перешкодою для ефективної реалізації MVP [4].

Попри велику кількість досліджень, залишається невирішеною проблема узгодження маркетингових та технічних аспектів у процесі розробки MVP. Саме цій проблематиці присвячено дану статтю.

Виклад основного матеріалу. Головна перевага односторінкового сайту – це простота і швидкість взаємодії з аудиторією. Адже вони розробляються з конкретною метою: для здійснення покупки, оформлення підписки, запису на консультацію чи лекцію.

Односторінковий сайт (Landing Page) – концентрує всю ключову інформацію про продукт або послугу на одній сторінці. Такі сайти ще називають лендінгом. Тому що лендінг пейдж це саме та сторінка, на яку приводиться трафік.

Канали просування відповідають за залучення користувачів, а структура – за конверсію. Те, як побудований односторінковий сайт і його наповнення має велике значення. Комфорт відвідувача на сайті безпосередньо впливає на його рішення покупки. Компанії, які використовують інструменти оптимізації своїх цільових сторінок, в середньому збільшують конверсію на 30%. Ця статистика Unbounce прояснює шлях до успіху: наявність ефективної цільової сторінки в технологічній галузі є обов'язковою умовою [5].

При побудові Landing Page слід взяти до уваги 4 важливі поради щодо створення ефективної цільової сторінки для технологій:

- Надавайте перевагу сучасним візуальним ефектам,
- Використовуйте переконливі заклики до дії,
- Використовуйте соціальні докази та демонструйте відгуки або схвальні відгуки,
- Переконайтеся, що сторінка повністю оптимізована для мобільних користувачів.

Розробка MVP орієнтується на монолітну або модульну SPA-структуру (Single Page Application), що дає змогу забезпечити швидке завантаження, інтерактивність та зручну навігацію на одній сторінці без потреби в переході між окремими URL. Архітектура повинна підтримувати: Кешування даних для прискорення повторних візитів; Lazy Loading для підвищення продуктивності; Розділення на компоненти, що спрощує підтримку та масштабування сайту; Реактивність інтерфейсу – реалізується за допомогою фреймворків типу React, Vue або аналогів.

Для створення MVP Landing Page пропонується застосовувати 1) для Фронтенду: HTML5, CSS3, JavaScript (або TypeScript); React.js або Next.js – для швидкої розробки SPA з серверним рендерингом; Tailwind CSS або Styled Components – для кастомізованого і швидкого дизайну; Axios або Fetch API – для інтеграції з backend-логікою, якщо така потрібна. 2) для Бекенду: Node.js + Express (у разі потреби у серверній логіці); Firebase або Supabase – як альтернатива для авторизації, бази даних та хостингу MVP; 3) Інші інструменти: Google Analytics, Hotjar – для аналізу поведінки користувачів; Lighthouse, PageSpeed Insights – для аудиту продуктивності та SEO.

Проєктування інтерфейсу базується на принципах: AIDA-моделі (Attention – Interest – Desire – Action); Mobile First – адаптивність для смартфонів і планшетів як пріоритет;

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Accessibility – врахування доступності для всіх типів користувачів (розмір елементів, контрастність, семантика HTML); Aesthetic Usability Effect – естетично привабливий дизайн, що підвищує сприйняття якості продукту. Важливими елементами при створенні UI/UX дизайну можуть бути Контрастна кнопка CTA (Call-to-Action); Соціальні докази: відгуки, рейтинги, логотипи клієнтів; Простота навігації за допомогою якорів (анкорних посилань); Візуальна ієрархія – правильне розташування блоків, що направляє користувача до цільової дії.

Після просування сайту на ринку через певний час потрібно вираховувати конверсію, та зрозуміти, який з інструментів працює краще за всі. Конверсія – це співвідношення кількості унікальних користувачів до цільових дій, які вони здійснюють на ресурсі. Це можуть бути будь-які дії: реєстрація, залишення контактів для зворотного зв'язку, переходи на необхідні сторінки всередині ресурсу, оформлення заявки чи замовлення, підписка на розсилку та інше. Конверсія обов'язково приносить власнику сайту вигоду, тому її потрібно постійно відслідковувати та коригувати веб-ресурс, щоб побачити кращі результати його роботи.

Є два базових види конверсій: Мікроконверсії. Враховуються всі цільові дії відвідувача, але без урахування основної, тобто купівлі, отримання власником ресурсу прибутку. Серед таких можуть бути: підписка на розсилку; перегляди сторінки з контактами компанії; реєстрація на ресурсі, та інше. Макроконверсії. Це показник кількості відвідувачів, які зробили цільову дію – придбали щось на сайті. Дія вважається завершеною після того, як клієнт отримав та повністю сплатив за товар.

Відсутність успішних мікроконверсій – шлях в нікуди. Якщо потенційний клієнт здійснює невеликі дії на сайті або хоча б одну із них – він вже зацікавлений у продукті. Тільки так є великий шанс отримати макроконверсію, і навіть неодноразово.

Універсальна формула (рис. 1), для розрахунок конверсії сайту, виглядає так:

Рис. 1. Формула конверсії

Джерело: [6]

Після вирахування конверсії сайту страхової компанії можна зробити кілька важливих висновків, залежно від значення отриманого показника.

– Висока конверсія (наприклад, 5–10% або більше): сайт добре виконує свої функції – користувачі не лише заходять, але й вчиняють цільові дії (залишають заявку, купують страховку, реєструються).

– Низька конверсія (менше 1–2%): існують проблеми з юзабіліті, довірою, контентом або структура сайту не відповідає очікуванням користувачів.

Конверсія дозволяє зрозуміти, скільки зусиль у рекламу і SEO реально перетворюються на клієнтів або ліди. Це база для розрахунку ROI (Return on Investment).

Для ефективної конверсії необхідно:

1. Правильна структура Landing Page. Правильно вибудуваний маркетинговий фреймворк (AIDA) необхідний для конверсії трафіку на сторінці.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

2. Mobile First та Accessibility. 80% трафіку приходить із мобільних пристроїв. Врахування доступності великого пальця до елементів на екрані необхідне для забезпечення оптимальної конверсії.

3. Ефективність навігації та callback форм. Швидкість пошуку необхідної інформації та заповнення форм зворотного зв'язку збільшує конверсію та забезпечує позитивний користувацький досвід.

4. Сучасний візуальний стиль. Дизайн забезпечує перше позитивне враження, підвищує конверсію, створює довіру та відчуття надійності, забезпечуючи Aesthetic Usability Effect.

AIDA Framework

AIDA Framework – це підхід до побудови Landing Page, який спрямовує користувача до виконання цільової дії (рис. 2):

Рис. 2. Підхід до побудови Landing Page AIDA

Джерело: розроблено автором

– *Attention* – За допомогою УТП виділити переваги та підкреслити унікальність товару чи послуги для користувачів та використати чітку кнопку «Завантажити / Купити / Зареєструватися», спонукаючи до дії.

– *Задача:* викликати цікавість до послуги, та спонукати користувача зробити дію.

– *Interest* – На цьому етапі блоки, мають висвітлювати інформацію про товар чи послугу, його основні переваги, підкреслити сегменти цільової аудиторії та покрити їхні потреби («Що це дає мені?»). Необхідно познайомити користувачів та відповісти на питання «Як це працює?».

Задача: дати в короткому форматі всю інформацію, необхідну для прийняття рішення, відповідючи на запитання Що? Хто? Кому? і Як?

– *Desire – Необхідно впевнити користувача в тому, що цей товар/послуга відповідає їх* очікуванням та покрити можливі демотиватори (такі як страх бути першим клієнтом). Включати позитивні відгуки, рейтинги, логотипи компаній, що користуються товаром / послугою, та відповіді на популярні запитання цільової аудиторії.

– *Задача:* Поглибити впевненість в тому, що пропонований продукт буде ефективним, покрити можливі додаткові питання.

– *Action* – На цьому етапі слід повторно підкреслити ключові бенефіти та потреби цільової аудиторії в УТП для остаточного прийняття рішення користувачем.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Задача: Фінальне спонукання до дії – купити продукт.

Практичне дослідження ефективності застосування AIDA-фреймворку на прикладі страхової компанії «Protego»

З метою емпіричної перевірки двовекторної моделі створення MVP Landing Page було проведено дослідження на прикладі страхової компанії «Protego», що спеціалізується на онлайн-страхуванні для мандрівників. Сайт компанії отримував близько 9 000 унікальних відвідувачів щомісяця, однак рівень макроконверсії (успішне оформлення полісу) складав лише 1,18%, що вказувало на потребу в стратегічному редизайні. В якості основного інструменту оптимізації було обрано AIDA-фреймворк у поєднанні з технічними і контентними заходами SEO-оптимізації.

Технічні аспекти оптимізації (технічний вектор): Поліпшено швидкість завантаження сторінки: впроваджено WebP-формати, lazy-loading для зображень, кешування на рівні браузера; Мобільна оптимізація (Mobile First): адаптація кнопок, форм і навігації до розміру екрану, покращення UX для користувачів смартфонів; Оптимізація коду: мінімізація HTML/CSS/JS, підвищення оцінки Google PageSpeed Insights із 61 до 91 (мобільна версія).

Контентна SEO-оптимізація (маркетинговий вектор): Створено семантичне ядро за допомогою Ahrefs: 25 релевантних ключових слів, включених у заголовки, підзаголовки та мета-теги; Оптимізовано структуру сторінки через якорі (anchor links) для поліпшення навігації на одній сторінці.

Впровадження фреймворку AIDA (таб. 1): 1) Attention – Візуально виразний блок з унікальною торговою пропозицією. Bounce rate знижено з 53% до 36%. 2) Interest – Інтерактивний калькулятор вартості та блок «Що ви отримуєте?». Зростання середньої тривалості сесії з 102 с до 176 с. 3) Desire – Соціальні докази, відгуки, гарантії. Збільшення мікроконверсій з 4,8% до 9,2%. 4) Action – CTA-кнопка «Оформити» зберігається при скролінгу, форма скорочена до 4 полів. Макроконверсія зросла з 1,18% до 4,36%.

Таблиця 1

Порівняльна таблиця результатів

Показник	До змін	Після впровадження AIDA + SEO
Bounce rate	53%	36%
Середня тривалість сесії (сек)	102	176
Мікроконверсія	4.8%	9,2%
Макроконверсія	1,18%	4,36%
SEO органічний трафік	~3 200	~4 200
Середній CTR у Google	2,9%	6,3%
ROI рекламної кампанії	210%	412%

Джерело: розроблено автором

Після впровадження AIDA-фреймворку та SEO-оптимізації показник bounce rate знизився з 53% до 36%, середня тривалість сесії зросла з 102 до 176 секунд, мікроконверсія підвищилася з 4,8% до 9,2%, а макроконверсія – з 1,18% до 4,36%. Органічний трафік збільшився на 31%, CTR у видачі Google – з 2,9% до 6,3%, а ROI рекламної кампанії зріс із 210% до 412%. Це свідчить про суттєве покращення як користувацької взаємодії, так і фінансової ефективності цільової сторінки. На рис. 3 наочно представлено порівняння ключових показників до і після впровадження AIDA-моделі, що ілюструє ефективність обраного підходу.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

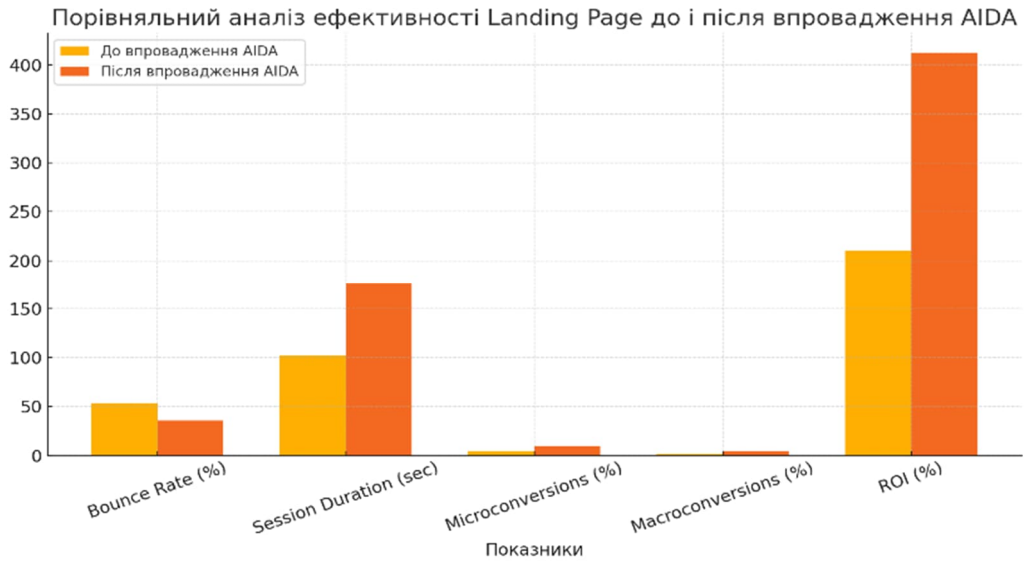


Рис. 3. Динаміка змін основних показників ефективності Landing Page після впровадження AIDA-фреймворку

Джерело: розроблено автором

Результати дослідження продемонстрували, що інтеграція AIDA-фреймворку із SEO-оптимізацією та технічною адаптацією сайту дала змогу досягти суттєвого зростання ключових метрик ефективності. Особливо показовим є приріст макроконверсії в понад 3,5 рази та зростання органічного трафіку на 31%. Отже, симбіоз маркетингового та технічного векторів забезпечує не лише кращу взаємодію користувача з сайтом, а й реальні бізнес-результати, що підтверджує доцільність обраної стратегії в межах MVP-підходу.

Просування односторінкового сайту – це комплексний підхід, який включає різні канали: SEO, рекламу, соціальні мережі та нативну рекламу. Кожен з цих інструментів доповнює один одного та підвищує загальну ефективність просування. Потрібно робити все в комплексі та притримуватись такої стратегії:

– Створити привабливе оголошення.

– Використовувати короткий і зрозумілий текст зі CTA (Call to Action).

– Додавати вигідну пропозицію або унікальну перевагу, наприклад: «Знижка 50% тільки сьогодні!» або «Безкоштовна доставка при замовленні на сайті».

– Уникати посилань на інші сторінки. Реклама повинна вестись прямо на Landing Page, де користувач зможе одразу виконати цільову дію.

Просування односторінкових сайтів за допомогою SEO: SEO-оптимізація забезпечує стабільний потік органічного трафіку на односторінковий сайт, підвищує видимість в пошукових системах та допомагає конвертувати відвідувачів у клієнтів.

1. Оптимізація ключових слів для односторінкового сайту (Основою успішного SEO є правильно підібрані та використані ключові слова. Для односторінкового сайту, який має обмежений обсяг тексту, це особливо важливо. Потрібно дослідження ключових слів. Використання інструментів на кшталт Google Keyword Planner, Ahrefs чи SEMrush, щоб знайти релевантні ключові слова, які підходитимуть саме цьому ресурсу. Також створити семантичне ядро та почати інтегрувати обрані ключові слова).

2.) Швидкість завантаження: критичний фактор для SEO (Односторінковий сайт повинен завантажуватися миттєво. Швидкість не лише впливає на досвід користувача, але й враховується пошуковими системами при ранжуванні).

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

3. Створення структури сторінки за допомогою якорів (Оскільки вся інформація знаходиться на одній сторінці, важливо забезпечити легку навігацію. Якорі допомагають користувачам швидко переходити між секціями, що позитивно впливає на час перебування на сайті).

4. Унікальний і цінний контент (Тексту на односторінковому сайті небагато, тому він повинен чітко відповідати запитам користувача та містити корисну інформацію).

5. Внутрішня оптимізація та метадані:

– *Title*: Створити короткий і описовий заголовок (50–60 символів), який включає основні ключові слова й точно відображає зміст сторінки.

– *Meta description*: Написати привабливий опис (150–160 символів) із використанням ключових слів для стимулювання кліків із пошукових систем.

– *Meta robots tag*: Використовувати цей тег, щоб вказати пошуковим системам, чи потрібно індексувати сторінку або слідувати за посиланнями.

– *Canonical tag*: Якщо схожий контент існує на інших ресурсах, додати канонічний тег, щоб уникнути проблем із дублюванням контенту.

6. Зовнішня оптимізація та посилання. (Залучати якісні зовнішні посилання на сайт. Вони підвищують авторитет ресурсу).

Використання реклами лендінг сайтів. Лендінг сторінки зазвичай створюються з однією метою – конверсія. Це може бути продаж, підписка, реєстрація або завантаження матеріалів. Тому реклама є ключовим інструментом для успішного просування.

Налаштування таргетованої реклами для Landing Page. Таргетована реклама дозволяє показувати односторінковий сайт саме тим користувачам, які можуть бути зацікавлені у пропозиції.

Потрібно сегментувати свою цільову аудиторію. Створити кілька великих і вужчих сегментів, щоб налаштувати більш точну рекламу. Інструменти, які можна використати для таргетованої реклами:

– Google Ads для пошукової та контекстно-медійної реклами.

– Facebook Ads та Instagram Ads для таргетингу за інтересами.

– TikTok Ads.

Запуск ремаркетингу для повернення потенційних клієнтів. Ремаркетинг дозволяє повернути на лендінг тих відвідувачів, які вже відвідали сайт, але не завершили покупку чи іншу дію. Використання пікселів відстеження (Facebook Pixel, Google Tag Manager), щоб зібрати дані про відвідувачів. Налаштвати оголошення для повторного показу продукту. Нагадати про пропозицію. Додавати спеціальні умови чи пропозицію.

Нативна реклама лендінгу. Нативна реклама гармонійно вписується в платформу, на якій вона розміщена. Вона не сприймається як нав’язлива реклама, тому користувачі більше залучені в контент. Важливо, щоб вона була релевантною та цінною для аудиторії, тоді конверсія зростатиме. Співпрацювання з блогами, новинними сайтами або тематичними порталами, щоб створювати оглядові чи інформаційні статті з посиланням на лендінг.

Модель оптимізації процесу розробки MVP Landing Page. 1. Двовекторна основа. Оптимізація процесу розробки MVP Landing Page базується на інтеграції двох ключових векторів – технічного та маркетингового, які мають взаємодіяти з перших етапів створення продукту. Технічний вектор охоплює архітектурні рішення, вибір стеку технологій, забезпечення високої продуктивності, адаптивності, мобільної зручності та доступності. Саме технічні інструменти, такі як React або Next.js, оптимізація завантаження через CDN, мінімізація коду, використання WebP-форматів і впровадження Mobile First-підходу формують технічну основу швидкого, стабільного й сучасного веб-продукту. Водночас маркетинговий вектор включає в себе стратегічне проєктування структури контенту на основі AIDA-моделі, створення переконливої Унікальної Торгової Пропозиції (УТП), впровадження соціальних доказів (відгуків, кейсів), а також проведення SEO-оптимізації. Такий симбіоз

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

дозволяє створити MVP (таб.2), що не лише функціонує з технічної точки зору, а й забезпечує максимальну ефективність з погляду залучення та конверсії користувачів.

Таблиця 2

Етапи моделі оптимізації MVP

	Етап	Дія	Результат
1	Дослідження	Аналіз цільової аудиторії, конкурентів, SEO-запитів	Формування цілей MVP
2	Прототипування	Побудова каркасу за AIDA-моделлю	UX-макет з конверсійною логікою
3	Технічна реалізація	Розробка SPA/SSR сторінки, SEO та швидкісна оптимізація	MVP з високою продуктивністю
4	Тестування	Перевірка UX, відгуків користувачів, поведінки	Виявлення вузьких місць
5	Запуск	Рекламні кампанії, SEO-просування, ремаркетинг	Реальне залучення лідів.
6	Аналітика та адаптація	Вимірювання конверсії, A/B-тестування, ROI	Постійне вдосконалення

Джерело: розроблено автором

Висновки. У результаті проведеного дослідження було обґрунтовано доцільність використання двовекторної структуризації при розробці MVP Landing Page для страхової компанії. Такий підхід дозволяє ефективно поєднувати технічні та маркетингові аспекти розробки, що є критично важливим для підвищення рівня конверсії та конкурентоспроможності в умовах динамічного цифрового середовища.

Запропонована структура Landing Page з урахуванням фреймворку AIDA, принципів Mobile First, UX-дизайну, а також SEO-оптимізації забезпечує зручну взаємодію з користувачем і сприяє досягненню бізнес-цілей компанії. Доведено, що аналітика конверсії виступає ключовим індикатором ефективності розробленого рішення та дозволяє приймати обґрунтовані рішення щодо подальшої оптимізації веб-ресурсу.

Таким чином, двовекторна модель створення MVP Landing Page є перспективним інструментом для впровадження цифрових рішень у сфері страхування, оскільки забезпечує швидке тестування гіпотез, адаптивність до змін ринку та підвищення цінності продукту для кінцевого користувача.

Список використаних джерел

1. Gothelf, J. (2013). Lean UX: Designing Great Products with Agile Teams. O'Reilly Media.
2. Nielsen, J., & Loranger, H. (2006). Prioritizing Web Usability. New Riders.
3. Google. (2020). The Mobile Playbook: The Busy Executive's Guide to Winning with Mobile. [Online] Доступно: https://www.thinkwithgoogle.com
4. Ahrefs. (2023). SEO Practices in 2023: Ranking Factors and Trends. [Online] Доступно: https://ahrefs.com
5. Unbounce. (2025). What is the average landing page conversion rate? [Online] Доступно: https://unbounce.com/average-conversion-rates-landing-pages/
6. AnaluticsMania. (2025). What is the conversion rate? [Online] Доступно: https:// www.analyticsmania.com/post/conversion-rate-in-google-analytics-4/

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

6. W3C. (2023). Web Content Accessibility Guidelines (WCAG) 2.1. [Online] Доступно: https://www.w3.org/TR/WCAG21/

7. Mozilla Developer Network (MDN). (2024). HTML, CSS, JavaScript Documentation. [Online] Доступно: https://developer.mozilla.org

Роботу виконано під науковим керівництвом д-ра техн. наук, професора
ЦЮЦЮРИ С. В.

ОПТИМІЗАЦІЯ БІЗНЕС-ПРОЦЕСІВ ТОРГОВЕЛЬНИХ ПІДПРИЄМСТВ ЗА ДОПОМОГОЮ КРОС-ПЛАТФОРМНИХ МОБІЛЬНИХ РІШЕНЬ

СУТЧЕНКО А. О., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглядається важливість розробки крос-платформних мобільних застосунків для торговельних підприємств. Описано особливості використання фреймворку React Native для створення таких застосунків, а також проаналізовано їхні переваги для бізнесу. Розглянуто популярні інструменти та підходи до розробки, включаючи інтеграцію з backend-системами та оптимізацію продуктивності. Зазначено перспективи розвитку крос-платформних рішень, які можуть включати штучний інтелект, аналітику даних та адаптацію до потреб сучасного ринку.

The article discusses the importance of developing cross-platform mobile applications for retailers. The features of using the React Native framework to create such applications are described, and their business benefits are analyzed. Popular tools and approaches to development, including integration with backend systems and performance optimization, are considered. Prospects for the development of cross-platform solutions, which may include artificial intelligence, data analytics, and adaptation to the needs of the modern market, are outlined.

Актуальність. Сучасні торговельні підприємства все частіше звертаються до мобільних технологій для підвищення ефективності бізнес-процесів, залучення клієнтів та оптимізації продажів. Зростання популярності смартфонів серед споживачів створює потребу в доступних і зручних мобільних застосунках, які працюють як на Android, так і на iOS. Розробка крос-платформних мобільних застосунків, зокрема з використанням React Native, дозволяє зменшити витрати на розробку та прискорити виведення продукту на ринок, зберігаючи при цьому високу якість і функціональність.

У контексті глобалізації та цифровізації торговельний сектор потребує інструментів, які забезпечують швидку адаптацію до змін ринкових умов та потреб клієнтів. Крос-платформні застосунки стають важливим рішенням для компаній, які прагнуть оптимізувати взаємодію з покупцями, автоматизувати процеси продажів і підвищувати лояльність клієнтів. Оскільки мобільні технології продовжують трансформувати сферу торгівлі, актуальність дослідження розробки таких застосунків лише зростає.

Метою статті є аналіз особливостей розробки крос-платформних мобільних застосунків для торговельних підприємств на базі React Native, а також оцінка їхньої ефективності для підвищення конкурентоспроможності бізнесу та спрощення взаємодії з клієнтами.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Об'єктом дослідження є крос-платформні мобільні застосунки, розроблені для торговельних підприємств з використанням фреймворку React Native.

Предмет дослідження є функції, можливості та ефективність крос-платформних мобільних застосунків, створених на базі React Native, для потреб торговельних підприємств.

Аналіз попередніх досліджень. Ю. Карпенко та Т. Литвиненко вважають, що впровадження мобільних технологій в українських роздрібних мережах суттєво впливає на оптимізацію ключових бізнес-процесів [5, с. 152-154], але помиляються в тому, що недостатньо уваги приділяють технічним особливостям впровадження React Native в контексті інтеграції з існуючими ERP-системами торговельних мереж та не розглядають довгострокові аспекти підтримки таких рішень. Вони аналізують ефективність різних технологічних рішень для управління запасами, автоматизації взаємодії з постачальниками та покращення обслуговування клієнтів. Їхнє дослідження показує, що крос-платформні рішення забезпечують до 35% економії ресурсів на розробку порівняно з нативними додатками, зберігаючи при цьому високий рівень користувацького досвіду, що є критично важливим для ритейлерів.

Д. Соколов та Н. Дмитрієва вважають, що економічна ефективність впровадження кросплатформних мобільних рішень у середніх та великих роздрібних мережах є значною [6, с. 203-205], але помиляються в тому, що не розглядають важливі аспекти інтеграції кросплатформних рішень з хмарними сервісами та аналітичними інструментами для обробки великих даних, які стають все більш критичними для конкурентоспроможності сучасних ритейлерів. Вони розробили методологію оцінки ROI для проектів цифрової трансформації з використанням React Native, яка враховує як прямі витрати на розробку та підтримку, так і непрямі ефекти від оптимізації бізнес-процесів. Дослідження показує, що ритейлери, які впровадили кросплатформні мобільні рішення для автоматизації управління запасами та взаємодії з клієнтами, досягли скорочення операційних витрат на 12–18% протягом першого року.

L. Zhang та V. Holzmann вважають, що практичне застосування React Native у великих міжнародних роздрібних мережах оптимізує процеси управління ланцюгами поставок [7, с. 218–220], але помиляються в тому, що не розглядають достатньо детально питання забезпечення високої доступності та відмовостійкості кросплатформних рішень в умовах розподілених роздрібних мереж з різним рівнем підключення до Інтернету. Вони провели порівняльний аналіз продуктивності мобільних рішень на основі React Native та нативних додатків в контексті інтеграції з системами планування ресурсів підприємства та управління взаємовідносинами з клієнтами. Результати їхнього дослідження показують, що використання React Native дозволило знизити витрати на розробку на 47% і прискорити цикл оновлення на 35%, зберігши при цьому високу продуктивність додатків навіть при роботі з великими обсягами даних.

R. Patel та M. Rodriguez вважають, що архітектурні патерни та практики розробки кросплатформних мобільних рішень мають критичне значення для сектору ритейлу [8, с. 167-169], але помиляються в тому, що не приділяють достатньої уваги забезпеченню конфіденційності даних користувачів при використанні крос-платформних технологій та не розглядають специфіку інтеграції з IoT-пристроями в контексті «розумних» магазинів. Вони зосереджуються на методах оптимізації продуктивності додатків React Native для забезпечення швидкої роботи з великими каталогами товарів та інтеграції з платіжними системами. Автори пропонують інноваційний підхід до структурування додатків на основі мікросервісної архітектури інтерфейсу, що дозволяє самостійно розробляти та оновлювати різні функціональні модулі. Упровадження запропонованих підходів у роздрібних мережах призвело до підвищення продуктивності мобільних додатків на 58% та зменшення використання пам'яті на 32%.

Виклад основного матеріалу. У сучасному бізнес-середовищі мобільні технології стали невід'ємною частиною стратегії розвитку торговельних підприємств. Статистика свідчить, що

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

понад 70% покупців використовують мобільні пристрої на різних етапах процесу купівлі: від пошуку інформації про товар до здійснення оплати та відстеження замовлення. Глобальна цифрова трансформація комерції прискорилася під впливом пандемії та змін у поведінці споживачів, коли зручність, швидкість та персоналізація стали ключовими факторами конкурентоспроможності.

Використання спеціалізованих мобільних рішень допомагає оптимізувати ключові бізнес-процеси, покращити клієнтський досвід та підвищити операційну ефективність.

Мобільні додатки допомагають торговельним підприємствам вирішувати цілий ряд завдань, від автоматизації складського обліку та інвентаризації до впровадження персоналізованих програм лояльності та аналізу споживчої поведінки. Особливо перспективним напрямком розвитку стали крос-платформні мобільні рішення, що дозволяють створювати додатки, які працюють одночасно на різних операційних системах, мінімізуючи витрати на розробку і підтримку.

Перш ніж розглядати специфіку кросплатформної розробки, необхідно розібратися в основних підходах до створення мобільних додатків. Існує три основних підходи до мобільної розробки:

1. *нативна розробка* – створення окремих додатків для кожної платформи (iOS, Android) з використанням специфічних для платформи мов програмування та інструментів розробки;

2. *веб-застосунки* – використання веб-технологій (HTML, CSS, JavaScript) для створення додатків, які запускаються в браузері мобільного пристрою;

3. *кросплатформна розробка* - використання єдиної кодової бази для створення додатків, які працюють на різних платформах з мінімальними змінами.

У таблиці 1 наведено порівняння основних підходів до розробки мобільних застосунків для торговельних підприємств.

Таблиця 1

Порівняння підходів до розробки мобільних застосунків

Характеристика	Нативна розробка	Веб-застосунки	Крос-платформна розробка
Швидкість розробки	Низька (окремі розробки для кожної платформи)	Висока (єдиний код для всіх платформ)	Середня-висока (спільна кодова база з деякими нативними елементами)
Продуктивність	Висока (оптимізована для конкретної платформи)	Низька-середня (залежить від браузера)	Середня-висока (близька до нативної)
Доступ до функцій пристрою	Повний доступ	Обмежений	Майже повний доступ через нативні модулі
Вартість розробки	Висока (окремі команди для кожної платформи)	Низька	Середня (одна команда для всіх платформ)
Користувацький досвід	Максимально адаптований до платформи	Загальний, може відрізнятися від очікувань користувачів	Близький до нативного з деякими компромісами

Джерело: розроблено автором на основі [1, 3, 5]

В умовах сьогодення мобільні рішення для торговельних підприємств можна класифікувати за кількома категоріями залежно від їхнього призначення та функціональності (рис. 1).

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Як видно з рисунка 1, мобільні рішення для ритейлерів поділяються на:

– клієнтські додатки, що орієнтовані на кінцевого споживача для здійснення покупок, отримання інформації про товари, участі в програмах лояльності;
– корпоративні рішення, що розробляються для внутрішніх потреб підприємства, а саме управління запасами, логістикою, персоналом;
– гібридні застосунки, що поєднують функції для клієнтів і співробітників з різними рівнями доступу;
– аналітичні інструменти збирають та аналізують дані про продажі, поведінку клієнтів та ефективність маркетингових кампаній.

Кожен з цих типів мобільних рішень виконує свою роль в оптимізації бізнес-процесів торговельного підприємства. Клієнтські додатки сприяють залученню нових покупців та утриманню існуючих, корпоративні рішення підвищують ефективність внутрішніх процесів, гібридні додатки забезпечують комплексну автоматизацію, а аналітичні інструменти допомагають приймати обґрунтовані бізнес-рішення на основі актуальних даних.

Рис. 1. Класифікація мобільних рішень для торговельних підприємств

Джерело: розроблено автором

React Native – один з найпопулярніших фреймворків для розробки кросплатформних мобільних додатків. Розроблений компанією Facebook, цей інструмент дозволяє створювати мобільні додатки за допомогою мови JavaScript та бібліотеки React, при цьому кінцевий продукт має характеристики нативного додатку [2].

Важливо відзначити, що вибір React Native як технологічної основи для розробки мобільних рішень для ритейлерів обумовлений не тільки його кросплатформенними можливостями, а й сучасним підходом до створення інтерфейсів на основі компонентної архітектури. Це дозволяє реалізувати модульну конструкцію, де кожен елемент системи (каталог товарів, кошик, особистий кабінет) розробляється і тестується окремо, а потім інтегрується в єдину систему, що значно підвищує якість кінцевого продукту і спрощує процес підтримки і розширення функціоналу.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Розробка кросплатформних мобільних додатків для торговельних підприємств має свої особливості, які впливають на вибір технологічного стеку та архітектурних рішень. При використанні React Native необхідно враховувати специфіку індустрії торговельних підприємств, зокрема, необхідність швидкої обробки великих обсягів даних (каталогів товарів), інтеграції з платіжними системами та іншими сервісами, а також підвищені вимоги до безпеки при роботі з персональними даними користувачів і проведенні фінансових транзакцій

React Native має такі переваги:

– повторне використання коду дозволяє використовувати до 90% одного й того ж коду для різних платформ (iOS та Android), що значно скорочує час розробки та знижує витрати [3].

– використовує «міст» для комунікації з нативними компонентами, що забезпечує високу продуктивність застосунку;

– дозволяє бачити зміни коду в реальному часі без перезавантаження всього застосунку, прискорюючи розробку;

– велика спільнота та екосистема бібліотек, забезпечує широкий вибір готових рішень для типових завдань.

Ці переваги роблять React Native особливо привабливим для торговельних підприємств, яким потрібно швидко виводити мобільні рішення на ринок і мінімізувати витрати на розробку. Крім того, можливість повторного використання коду значно спрощує підтримку та оновлення додатків, що важливо в умовах динамічного розвитку торговельних підприємств та необхідності швидкого впровадження нових функцій і можливостей.

Архітектура мобільного додатку на основі React Native має стратегічне значення для успішного впровадження крос-платформних рішень у бізнес-процеси ритейлерів. Правильно спроектована архітектура забезпечує гнучкість, масштабованість та ефективну інтеграцію з існуючими системами підприємства.

На рисунку 2 показано багаторівневу архітектуру React Native додатку, де верхній рівень представляє різні платформи (iOS, Android та Web), на яких може працювати додаток, демонструючи головну перевагу кросплатформної розробки - можливість використовувати один і той самий код для різних операційних систем.

Центральний шар містить власне фреймворк React Native з трьома основними компонентами: UI Components для створення користувацького інтерфейсу, Business Logic для обробки даних та операцій і Native Modules, які надають доступ до специфічних функцій мобільних пристроїв (камера, GPS, сповіщення тощо).

Наступний рівень складається з API та сервісів (REST API, GraphQL, WebSockets), які забезпечують взаємодію мобільного додатку з серверною частиною, дозволяючи отримувати та відправляти дані, синхронізувати інформацію та інтегруватися з іншими системами торговельного підприємства.

Нижній рівень - серверна інфраструктура ритейлера, яка містить бази даних, бізнес-логіку та інші корпоративні системи, з якими взаємодіє мобільний додаток, забезпечуючи безперебійну роботу всього технологічного стеку.

Особливістю цієї архітектури є її здатність надавати єдиний інтерфейс для взаємодії з різними операційними системами, зберігаючи при цьому нативну продуктивність. Взаємодія між рівнями відбувається через чітко визначені інтерфейси, що забезпечує гнучкість та масштабованість рішення.

Модульна структура архітектури дозволяє легко розширювати функціональність програми та адаптуватися до змін у бізнес-процесах торговельної компанії. Наприклад, можна додати нові модулі для управління програмами лояльності, інтегрувати з новими платіжними системами або впровадити інноваційні технології, такі як доповнена реальність для візуалізації товарів.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

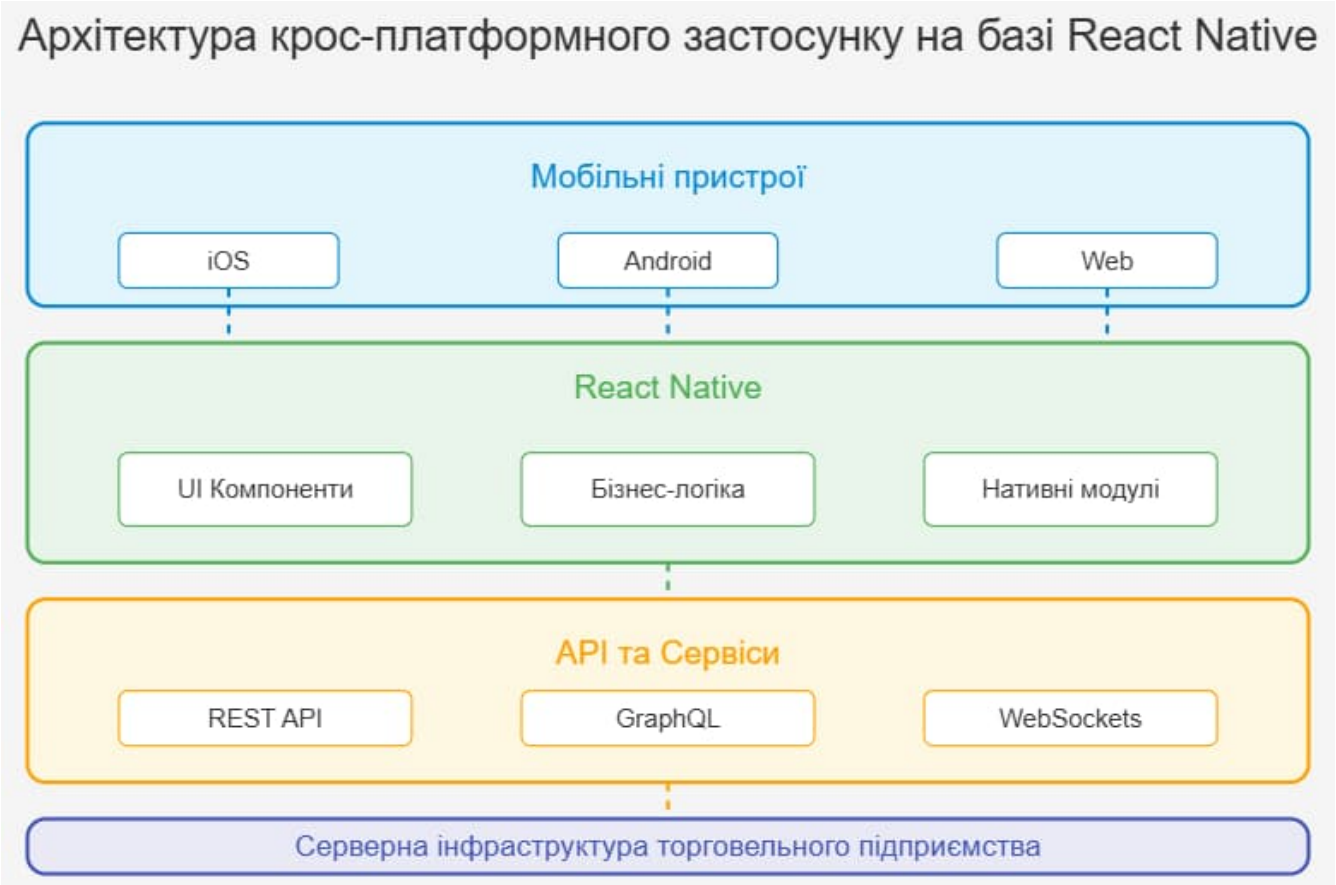


Рис. 2. Архітектура крос-платформного застосунку на базі React Native

Джерело: розроблено автором

Крос-платформні мобільні рішення надають торговельним підприємствам ряд суттєвих переваг, які узагальнено в таблиці 2.

Таблиця 2

Переваги використання крос-платформних застосунків для бізнесу

№	Перевага	Опис
1	Скорочення витрат	Розробка одного застосунку замість двох окремих для iOS і Android дозволяє підприємствам оптимізувати бюджети
2	Швидке виведення на ринок	Єдина кодова база прискорює розробку та спрощує тестування застосунку
3	Зручність у підтримці	Оновлення та виправлення помилок впроваджуються одночасно для всіх платформ.
4	Інтеграція з backend-системами	Можливість підключення до CRM, ERP та інших бізнес-платформ для управління товарами, продажами та аналітикою
5	Поліпшення взаємодії з клієнтами	Мобільні додатки сприяють підвищенню рівня обслуговування, впровадженню програм лояльності та персоналізації пропозицій

Джерело: розроблено автором на основі [1, 4]

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Управління ланцюгами поставок впровадження крос-платформних мобільних рішень суттєво трансформує процеси управління ланцюгами поставок торговельних підприємств. Мобільні застосунки забезпечують підвищення ефективності логістичних операцій через:

- моніторинг переміщення товарів у режимі реального часу;
- автоматизацію процесу замовлень у постачальників;
- оптимізацію маршрутів доставки;
- інтеграцію з системами складського обліку.

Управління взаємодією з клієнтами крос-платформні мобільні рішення створюють нові можливості для взаємодії з клієнтами, підвищуючи ефективність обслуговування через:

- персоналізацію пропозицій на основі історії покупок;
- автоматизацію програм лояльності;
- багатоканальну комунікацію;
- інтегровану систему відгуків та підтримки.

Функціонал мобільних POS-терміналів дозволяє здійснювати продажі в будь-якій точці торговельного залу, що особливо актуально в періоди пікового навантаження. Персонал отримує миттєвий доступ до історії замовлень клієнтів, що дозволяє формувати релевантні рекомендації та пропозиції.

Як видно з рисунку 3, кожен з методів має свої особливості та сфери оптимального застосування. Вибір того чи іншого підходу залежить від специфіки бізнес-процесів компанії, обсягів даних, вимог до продуктивності та характеру взаємодії користувачів з системою. Ефективна API-інтеграція дозволяє максимально використовувати потенціал існуючих інформаційних систем підприємства при впровадженні інноваційних мобільних рішень.

Перспективи розвитку крос-платформних рішень для торговельних підприємств пов'язані з такими ключовими напрямками [4]:

– *інтеграція технологій штучного інтелекту* відкриває нові можливості для прогнозування споживчого попиту, створення високо персоналізованих пропозицій та оптимізації асортименту на основі аналізу великих обсягів даних;

– *розширені аналітичні можливості* дозволяють проводити глибокий аналіз поведінки покупців і показників продажів безпосередньо в мобільному додатку, що значно прискорює процес прийняття бізнес-рішень;

– *безшовна інтеграція з IoT-пристроями* створює можливості для формування інтелектуальних торгових просторів завдяки взаємодії з RFID-мітками, розумними полицями, електронними цінниками та іншими компонентами розумного магазину;

– *технології доповненої і віртуальної реальності* створюють інноваційні підходи до візуалізації продуктів і взаємодії з покупцями, значно покращуючи клієнтський досвід;

– *прогресивні веб-додатки (PWA)* являють собою гібридний підхід, що поєднує переваги мобільних і веб-додатків, забезпечуючи максимально широке охоплення користувачів без необхідності встановлення окремих додатків.

Важливо зазначити, що інтеграція блокчейн-технологій у кросплатформні мобільні рішення створює нові можливості для підвищення прозорості ланцюгів постачання та автентифікації продукції. Для ритейлерів це означає можливість надавати споживачам повну інформацію про походження товарів, умови виробництва та транспортування, що особливо цінно для сегментів ринку з високими стандартами якості та етики. Крім того, використання смарт-контрактів дозволяє значно спростити процеси розрахунків між учасниками торговельних відносин, мінімізуючи ризики шахрайства та знижуючи транзакційні витрати. Ці інновації, реалізовані в рамках крос-платформних мобільних рішень, формують нову парадигму взаємодії між виробниками, ритейлерами та кінцевими споживачами.

Висновки. У сучасному конкурентному середовищі ритейлери все частіше звертаються до мобільних технологій як до інструменту оптимізації бізнес-процесів та підвищення ефективності. Кросплатформні мобільні рішення на основі React Native пропонують оптимальне співвідношення вартості розробки, швидкості виведення на ринок та якості кінцевого продукту.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

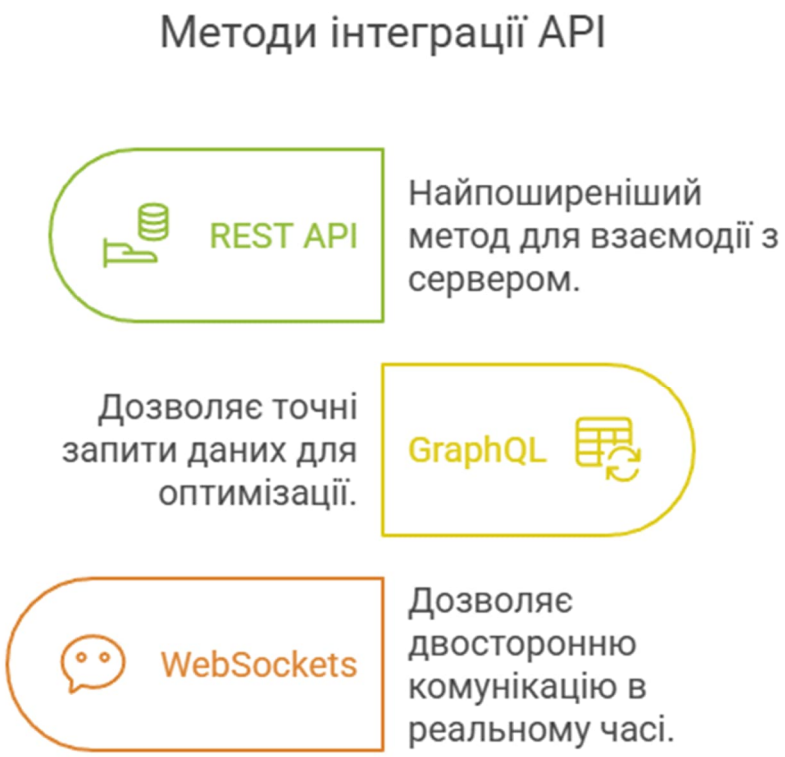


Рис. 3. Методи інтеграції API

Джерело: розроблено автором

Аналіз показує, що використання крос-платформних технологій дозволяє ритейлерам значно оптимізувати ключові бізнес-процеси, включаючи управління запасами, обслуговування клієнтів та аналітику. При цьому єдина кодова база для iOS та Android знижує витрати на розробку та підтримку додатків, роблячи технологію доступною навіть для малого та середнього бізнесу.

React Native, завдяки своїй архітектурі та використанню нативних компонентів, забезпечує високу продуктивність та якісний користувацький інтерфейс, наближений до нативного. Це особливо важливо для комерційних додатків, де швидкість і зручність використання безпосередньо впливають на ефективність процесів і задоволеність користувачів.

Перспективи розвитку кросплатформних рішень пов’язані з інтеграцією нових технологій, таких як штучний інтелект, розширена аналітика та інтернет речей, що відкриває нові можливості для подальшої оптимізації бізнес-процесів і створення інноваційних сервісів для клієнтів.

Загалом, кросплатформні мобільні рішення на основі React Native є ефективним інструментом оптимізації бізнес-процесів ритейлерів, що дозволяє їм підвищити свою конкурентоспроможність та адаптуватися до нових викликів ринку.

Список використаних джерел

1. Боднар, О. С., Петренко, В. А. (2023). Сучасні технології розробки мобільних застосунків для бізнесу. Інформаційні технології в економіці, (4), 78–86.

2. Meta Platforms, Inc. (2025). React Native Official Documentation. https://reactnative.dev/docs/getting-started

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

3. SE Daily (2020). React Native at Airbnb with Gabriel Peal Holiday Repeat. Software Engineering Daily. https://softwareengineeringdaily.com/2020/11/25/react-native-at-airbnb-with-gabriel-peal-holiday-repeat/

4. Шевченко, О. М., Мельник, П. В. (2023). Автоматизація бізнес-процесів торговельних підприємств: практичний досвід впровадження мобільних рішень. Вісник ДТЕУ, (2), 134–142.

5. Карпенко, Ю. В., Литвиненко, Т. М. (2023). Крос-платформні мобільні технології як інструмент омніканальних стратегій українських ритейлерів. Маркетинг і цифрові технології, 7(1), 148–162.

6. Соколов, Д. І., Дмитрієва, Н. В. (2022). Економічна ефективність впровадження крос-платформних мобільних рішень у торговельних підприємствах. Економіка і прогнозування, 4, 197–210.

7. Zhang, L., Holzmann, V. (2022). Performance Analysis of React Native in Retail Supply Chain Management Applications. International Journal of Mobile Technologies in Retail, 15(3), 214–226.

8. Patel, R. K., Rodriguez, M. S. (2021). Architectural Patterns for High-Performance Cross-Platform Retail Applications Using React Native. Journal of Software Engineering and Applications, 14(4), 159–173.

9. Sharma, R. (2022). Cross-Platform Mobile Development: Benefits and Challenges. Journal of Software Engineering, 15, 45–52.

Роботу виконано під науковим керівництвом д-ра екон. наук, професора
ТОКАРЯ В. В.

ДВОНАПРАВЛЕНА СТРУКТУРА УПРАВЛІННЯ ПРОЄКТАМИ РОЗРОБКИ ВЕБОРІЄНТОВАНИХ ОСВІТНІХ ПЛАТФОРМ: ПОРІВНЯЛЬНИЙ АНАЛІЗ ІЗ ТРАДИЦІЙНИМИ ПІДХОДАМИ

ТАРАСЕНКО В. О., 2 курс 4м група ФІТ ДТЕУ,
освітня програма «Управління проєктами програмних продуктів»

У статті розглянуто проблему управління проєктами розробки веборієнтованих освітніх платформ в умовах зростаючої популярності дистанційного навчання. Проаналізовано обмеження традиційних підходів (Waterfall, PMBOK) та запропоновано двонаправлену структуру управління, що характеризується гнучкістю, покращеною комунікацією та здатністю швидко адаптуватися до змінних вимог. Проведено порівняльний аналіз двонаправленої та традиційних методів за ключовими критеріями ефективності. Зроблено висновок про потенційні переваги двонаправленої структури в умовах високої невизначеності, а також висвітлено виклики, пов'язані з її впровадженням.

The article examines the problem of project management for the development of web-oriented educational platforms in the context of the growing popularity of distance learning. The limitations of traditional approaches (Waterfall, PMBOK) are analyzed, and a two-way management structure characterized by flexibility, improved communication, and the ability to quickly adapt to changing requirements is proposed. A comparative analysis of the two-way and traditional methods is carried out based on key performance criteria. It is concluded that the two-way structure has potential advantages in conditions of high uncertainty, and challenges related to its implementation are highlighted.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Актуальність. Сучасний освітній простір характеризується стрімким зростанням популярності дистанційної освіти, що обумовлено рядом факторів, включаючи глобалізацію, технологічний прогрес та прагнення до навчання впродовж життя. Веборієнтовані освітні платформи відіграють ключову роль у забезпеченні доступності та ефективності дистанційного навчання, надаючи широкий спектр інструментів для організації навчального процесу, включаючи доступ до навчальних матеріалів, інтерактивну взаємодію між викладачами та студентами, тестування знань та відстеження прогресу навчання. Розробка та впровадження таких платформ є складним процесом, що вимагає ефективного управління проєктами для забезпечення досягнення поставлених цілей в межах заданих термінів та бюджету.

Проте, традиційні підходи до управління проєктами, такі як каскадна модель (Waterfall) та стандарт PMI (PMBOK), мають певні обмеження при застосуванні до проєктів розробки освітніх платформ. Зокрема, їх ієрархічна структура та послідовний підхід до виконання задач можуть бути недостатньо гнучкими для адаптації до швидких змін вимог користувачів та ринкових умов, що є характерним для сфери дистанційної освіти. Повільна адаптація до змін може призвести до розробки платформ, які не відповідають потребам користувачів, втрати конкурентоспроможності та перевищення бюджету проєкту. Ці обмеження підкреслюють необхідність пошуку нових, більш ефективних підходів до управління проєктами розробки освітніх платформ, які забезпечують гнучкість, швидкість реагування на зміни та покращену комунікацію між учасниками проєкту.

Метою статті є аналіз ефективніості традиційних підходів до управління проєктами розробки веборієнтованих освітніх платформ та обґрунтувати доцільність впровадження двонаправленої структури управління як альтернативного гнучкого методу в умовах високої невизначеності.

Об'єктом дослідження є процес управління проєктами розробки веборієнтованих освітніх платформ.

Предмет дослідження – методи та структури управління проєктами.

Аналіз попередніх досліджень. Дослідження в області управління проєктами розробки платформ онлайн-освіти представлені в працях як українських, так і міжнародних авторів. Лавданська О. В. та Базар'я І. В. [1] розглядають загальні аспекти управління проєктом розробки платформи онлайн-освіти, надаючи практичні рекомендації щодо організації цього процесу. Онисько Е. та Фармага І. [2] здійснюють огляд та аналіз різних систем управління проєктами, що дозволяє оцінити їхні можливості та обмеження в контексті освітніх проєктів.

Козир С. та Молоканова В. [3] проводять системний аналіз управління інтеграцією проєктів в портфелях розвитку дуальної освіти, що є важливим для забезпечення узгодженості та ефективності різних проєктів в освітній сфері. Бушуєв С.В.[4] досліджують інформаційні технології розвитку компетенцій менеджерів з управління проєктами на основі глобальних трендів, наголошуючи на необхідності постійного вдосконалення професійних навичок в умовах швидких змін.

Важливість врахування думки різних груп зацікавлених сторін при оцінці успіху проєкту підкреслюється в дослідженні Девіса К. [5], що дозволяє сформувати більш об'єктивну картину результативності проєкту. Давліканова О. та Хофстеттер Х. [6] аналізують аспекти управління та результати адаптації моделі «дуальної освіти» як інновації в українській вищій освіті.

Пінто Дж. [10] публікують заклик до подання статей на спеціальний випуск International Journal of Project Management, присвячений успіху проєкту, наголошуючи на актуальності цієї теми для дослідників та практиків.

Виклад основного матеріалу. Управління проєктами визначається як застосування знань, навичок, інструментів та методів до діяльності проєкту для задоволення вимог, що висуваються до нього. Управління проєктами включає ініціювання, планування, виконання, моніторинг та контроль, а також завершення проєкту. Ефективне управління проєктами

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

дозволяє досягти поставлених цілей в межах заданих термінів, бюджету та з відповідним рівнем якості.

Освітня платформа розглядається як комплексна інформаційна система, призначена для організації та підтримки навчального процесу. Вона надає широкий спектр функцій, включаючи доступ до навчальних матеріалів, інструменти для спілкування між викладачами та студентами, системи тестування та оцінювання знань, а також засоби для відстеження прогресу навчання. Веборієнтований продукт, в свою чергу, визначається як програмне забезпечення, доступне через веб-браузер, що дозволяє користувачам працювати з системою з будь-якого пристрою, підключеного до Інтернету. Веборієнтований підхід забезпечує гнучкість та масштабованість освітньої платформи, роблячи її доступною для широкої аудиторії.

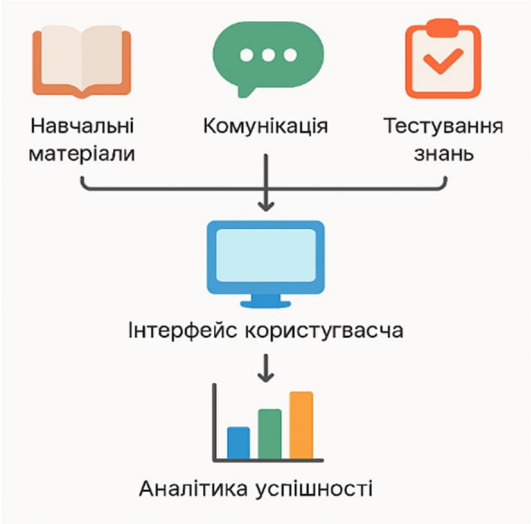


Рис. 1. Компоненти веборієнтованої освітньої платформи

Джерело: розроблено автором

Традиційні підходи до управління проєктами – це загальноприйняті та широко використовувані методології, які характеризуються лінійним або ітеративним підходом до управління проєктами. До таких підходів належать, зокрема, каскадна модель (Waterfall) та стандарт PMI (Project Management Body of Knowledge).

Каскадна модель (Waterfall), з її лінійною та послідовною структурою, вирізняється чітким плануванням та прогнозованістю, що може бути корисним для проєктів з чітко визначеними вимогами та обмеженим обсягом змін. Однак, вона виявляється малоефективною у випадках, коли вимоги часто змінюються або є невизначеними на початкових етапах проєкту, що є типовим для розробки освітніх платформ, де важливо враховувати потреби користувачів та швидко адаптуватися до ринкових змін. Низька гнучкість, обмежена комунікація між командами на різних етапах та складність вирішення проблем, що виникають на пізніх етапах розробки, можуть призвести до розробки застарілого продукту, який не відповідає потребам користувачів, та, відповідно, низької задоволеності команди.

Стандарт PMBOK (Project Management Body of Knowledge), з його комплексним підходом та стандартизованими процесами, забезпечує системний підхід до управління проєктами та надає інструменти для управління всіма аспектами проєкту, включаючи інтеграцію, зміст, час, вартість, якість, людські ресурси, комунікації, ризики, закупівлі та зацікавлені сторони. PMBOK® Guide може бути корисним для великих проєктів зі складними вимогами та великою кількістю учасників, де важлива стандартизація процесів та чітке визначення ролей та відповідальності. Однак, застосування PMBOK® Guide може бути

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

занадто формалізованим та складним для менших проєктів, а також вимагати значних зусиль на адаптацію до специфічних вимог проєктів розробки освітніх платформ. Залежність від бюрократії, складність адаптації до швидких змін та обмежена гнучкість можуть знизити ефективність комунікації та вирішення проблем, а також призвести до збільшення термінів та бюджету проєкту.

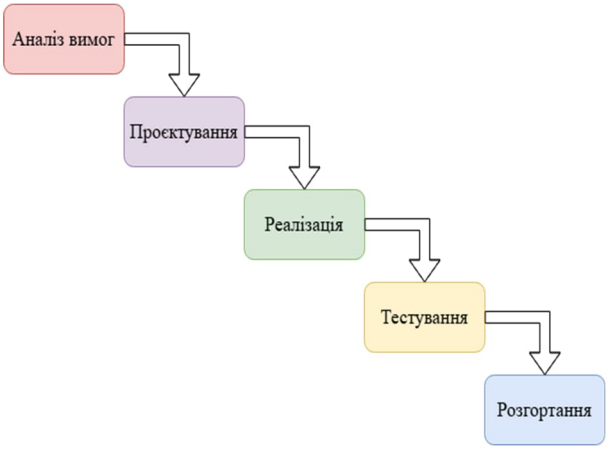


Рис. 2. Структура каскадної моделі (Waterfall)

Джерело: розроблено автором

Рис. 3. Модель управління за PMBOK

Джерело: розроблено автором

Двонаправлена структура управління, навпаки, має на меті забезпечення максимальної гнучкості, швидкої адаптації до змін та покращеної комунікації між учасниками проєкту. Ця структура характеризується децентралізованим підходом до прийняття рішень, де команди мають більшу автономію та відповідальність за свої результати. Забезпечення прямої комунікації між різними командами та заохочення зворотного зв'язку від користувачів дозволяє швидко реагувати на зміни вимог та ринкових умов. Підвищення задоволеності команди, покращення якості розробленої платформи та прискорення процесу розробки є потенційними перевагами двонаправленої структури. Однак, застосування цієї структури потребує високого рівня самоорганізації команди, наявності чітких правил та процедур, а

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

також ефективних інструментів для підтримки комунікації та координації. Ризик виникнення конфліктів через багатоканальну комунікацію, перевантаження інформацією та труднощі з дотриманням термінів та бюджету є потенційними викликами.

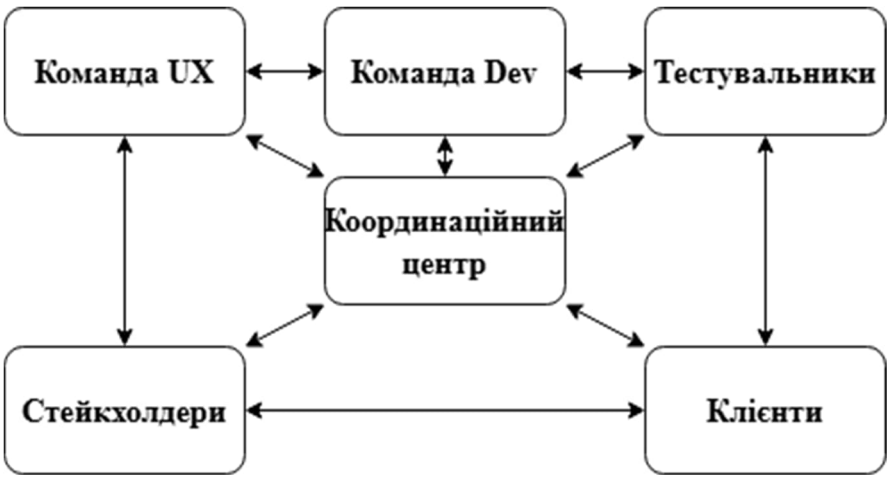


Рис. 4. Двонаправлена структура управління проєктами

Джерело: розроблено автором

На жаль, наразі існує обмежена кількість прикладів застосування двонаправленої структури управління в реальних проєктах розробки веборієнтованих освітніх платформ, що ускладнює оцінку її реальної ефективності на основі емпіричних даних. Проте, аналіз успішних проєктів, в яких використовувалися гнучкі методології розробки (Agile), показує, що децентралізовані та командно-орієнтовані підходи до управління можуть сприяти досягненню кращих результатів порівняно з традиційними методами.

Двонаправлена структуризація є більш ефективною, ніж традиційні методи управління, за умов, коли проєкт характеризується високим рівнем невизначеності вимог, потребує швидкої адаптації до потреб користувачів, вимагає активної участі користувачів у процесі розробки та орієнтований на швидке виведення продукту на ринок. У таких випадках, гнучкість, ефективність комунікації та здатність швидко вирішувати проблеми є критично важливими факторами успіху.

Проєкти розробки веборієнтованих освітніх платформ мають специфічні характеристики, які необхідно враховувати при виборі методу управління. До цих характеристик належать складні технічні вимоги, потреба в інтеграції з різними системами (LMS, CRM), необхідність врахування педагогічних принципів та важливість забезпечення зручності використання (UX/UI). Складні технічні вимоги обумовлені необхідністю розробки масштабованої, надійної та безпечної системи, яка може обробляти великий обсяг даних та забезпечувати високий рівень продуктивності. Потреба в інтеграції з різними системами (LMS, CRM) вимагає забезпечення сумісності та обміну даними між різними платформами. Необхідність врахування педагогічних принципів вимагає, щоб освітня платформа була розроблена з урахуванням потреб учнів та викладачів, забезпечувала ефективні інструменти для навчання та оцінювання знань. Важливість забезпечення зручності використання (UX/UI) обумовлена необхідністю створення інтуїтивно зрозумілого та приємного для користувача інтерфейсу, який сприяє залученню та утриманню користувачів на платформі.

У зв’язку з цим, традиційні підходи до управління проєктами, які є менш гнучкими та адаптивними, можуть бути неефективними для проєктів розробки освітніх платформ, що зумовлює необхідність використання більш інноваційних та гнучких підходів до управління проєктами.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Оцінка ефективності методів управління проєктами розробки веборієнтованих освітніх платформ здійснюється на основі комплексу ключових критеріїв, які відображають здатність методу забезпечити успішне досягнення цілей проєкту та задоволення потреб зацікавлених сторін. До основних критеріїв належать гнучкість, ефективність комунікації, ефективність вирішення проблем, задоволеність команди, якість розробленої платформи, а також дотримання термінів та бюджету.

Гнучкість визначається як здатність методу управління адаптуватися до змін вимог користувачів, технологічних інновацій та мінливих ринкових умов. Проєкти розробки освітніх платформ часто характеризуються високим ступенем невизначеності та потребують постійного внесення змін до планів та специфікацій. Метод управління, що забезпечує швидку та ефективну адаптацію до змін, є більш ефективним у цьому контексті.

Ефективність комунікації відображає швидкість та якість обміну інформацією між усіма учасниками проєкту, включаючи розробників, дизайнерів, тестувальників, викладачів та стейкхолдерів. Ефективна комунікація сприяє покращенню розуміння вимог, зменшенню кількості помилок та конфліктів, а також підвищенню задоволеності всіх учасників проєкту. Метод управління, який забезпечує прозорий та ефективний обмін інформацією, є більш ефективним.

Ефективність вирішення проблем визначається як здатність швидко ідентифікувати, аналізувати та вирішувати проблеми, що виникають в процесі розробки. Проєкти розробки освітніх платформ часто пов’язані з технічними складнощами, непередбаченими проблемами та конфліктами між різними учасниками проєкту. Метод управління, який передбачає чіткі процедури вирішення проблем та забезпечує підтримку прийняття рішень на основі об’єктивних даних, є більш ефективним.

Задоволеність команди відображає рівень задоволеності учасників проєкту умовами праці, можливостями для професійного розвитку та результатами їхньої роботи. Задоволена команда є більш продуктивною, творчою та відданою проєкту. Метод управління, який забезпечує підтримку командної роботи, визнання досягнень та створення сприятливого робочого середовища, є більш ефективним.

Якість розробленої платформи оцінюється за критеріями відповідності вимогам користувачів, надійності, зручності використання (UX/UI) та безпеки. Якісна освітня платформа забезпечує ефективну підтримку навчального процесу, сприяє залученню та утриманню користувачів, а також забезпечує захист даних користувачів від несанкціонованого доступу. Метод управління, який акцентує увагу на якості та забезпечує застосування найкращих практик розробки, тестування та забезпечення безпеки, є більш ефективним.

Дотримання термінів та бюджету відображає здатність виконати проєкт в заданий термін та в межах встановленого бюджету. Дотримання термінів та бюджету є важливим фактором успіху будь-якого проєкту, оскільки дозволяє уникнути затримок та перевитрат, що можуть негативно вплинути на репутацію організації та її фінансові результати. Метод управління, який забезпечує ефективне планування, контроль та управління ресурсами, є більш ефективним.

Враховуючи вищезазначені критерії, можна провести порівняльний аналіз різних методів управління проєктами розробки веборієнтованих освітніх платформ та визначити оптимальний підхід для конкретного проєкту.

Проведений аналіз дозволяє порівняти двонаправлену структуру управління проєктами розробки веборієнтованих освітніх платформ з традиційними підходами, такими як каскадна модель (Waterfall) та стандарт PMBOK (Project Management Body of Knowledge), за визначеними критеріями: гнучкість, ефективність комунікації, ефективність вирішення проблем, задоволеність команди, якість розробленої платформи, дотримання термінів та бюджету. Результати порівняння виявляють суттєві відмінності між підходами та підкреслюють їхні сильні та слабкі сторони в контексті розробки освітніх платформ.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Таблиця 1

Порівняльний аналіз методів управління проєктами розробки веборієнтованих освітніх платформ

Критерій	Каскадна модель (Waterfall)	PMBOK (Project Management Body of Knowledge)	Двонаправлена структура управління (гіпотетична)
Гнучкість	Низька	Середня	Висока
Ефективність комунікації	Низька	Середня	Висока
Ефективність вирішення проблем	Низька	Середня	Висока
Задоволеність команди	Низька	Середня	Висока
Якість платформи	Середня	Середня - Висока (залежить від впровадження)	Висока
Дотримання термінів та бюджету	Висока (при стабільних вимогах)	Середня	Середня
Сильні сторони	Чіткий план, прогнозованість	Комплексний підхід, стандартизація	Гнучкість, швидкість адаптації, командна робота
Слабкі сторони	Жорсткість, повільна адаптація	Складність, формалізованість	Потребує самоорганізації команди, можливі конфлікти
Ризики	Зміни вимог, застарілий продукт	Бюрократія, складність адаптації	Залежність від команди, перевитрати часу

Джерело: розроблено автором

Потенційні виклики та обмеження при застосуванні двонаправленої структури управління включають необхідність формування команди з високим рівнем самоорганізації та мотивації, розробку ефективних інструментів для підтримки комунікації та координації, а також впровадження чітких правил та процедур для уникнення конфліктів та непорозумінь. Також, успішність застосування двонаправленої структури управління залежить від підтримки керівництва організації та готовності до змін в організаційній культурі.

У сучасному освітньому середовищі, що характеризується зростанням популярності дистанційного навчання, розробка ефективних веборієнтованих освітніх платформ вимагає гнучких та адаптивних підходів до управління проєктами. Традиційні методи, такі як Waterfall та PMBOK, мають обмеження щодо адаптації до змінних вимог. Аналіз показав, що двонаправлена структура управління, хоча й гіпотетична, має потенціал забезпечити вищу гнучкість, покращити комунікацію та сприяти швидкому вирішенню проблем, що є важливими факторами успішної розробки освітніх платформ. Однак, впровадження двонаправленої структури вимагає високоорганізованої команди, чітких правил та ефективних інструментів комунікації.

Список використаних джерел

1. Лавданська О.В., Базар'я І.В. Управління проєктом розробки платформи онлайн-освіти. Черкаси: Черкаський державний технологічний університет, 2023. URL: https://er.chdtu.edu.ua/handle/ChSTU/5162 (дата звернення: 28.03.2025).

2. Онисько Е., Фармага І. Огляд та аналіз систем управління проектами. CDS. 2024. Вип. 6, № 1. С. 209–215. DOI: 10.23939/cds2024.01.209.

3. Козир С., Молоканова В. Системний аналіз управління інтеграцією проєктів в портфелях розвитку дуальної освіти. Information Technology: Computer Science, Software

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Engineering and Cyber Security. 2023. № 1. URL: https://journals.politehnica.dp.ua/ index.php/ it/article/view/257 (дата звернення: 28.03.2025).

4. Бушуєв С.Д., Бушуєв Д.А., Бушуєва Н.С., Козир Б.Ю. Інформаційні технології розвитку компетенцій менеджерів з управління проектами на основі глобальних трендів. Інформаційні технології і засоби навчання. 2018. Т. 68, № 6. С. 218–234. DOI: 10.33407/itlt.v68i6.2684.

5. Davis K. An empirical investigation into different stakeholder groups perception of project success. International Journal of Project Management. 2017. Vol. 35, No. 4. P. 604–617. DOI: 10.1016/j.ijproman.2017.02.004.

6. Davlikanova O., Hofstetter H. The «Duales Studium» Adaptation as an Innovation in Ukrainian Tertiary Education: Management Aspects and Results. Marketing and Management of Innovations. 2020. Vol. 3. P. 208–221. DOI: 10.21272/mmi.2020.3-15.

7. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) – Fifth Edition. Project Management Institute, Inc. 2013. 586 p.

8. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) – Seventh Edition. Project Management Institute, Inc. 2021.

9. Гусєва Ю.Ю., Чумаченко І.В. Концептуальний підхід до підтримки прийняття рішень з управління вимогами та вартістю в проектах. Управління розвитком складних систем. 2020. № 4. С. 21–27. DOI: 10.32347/2412-9933.2020.41.21-27.

10. Jeffrey Pinto, Kate Davis, Lavagnon Ika, Kam Jugdev, Ofer Zwikael. Call for Papers: Special Issue on Project Success. International Journal of Project Management. 2021. Vol. 39, № 2. P. 213–215. DOI: 10.1016/j.ijproman.2021.01.007.

Роботу виконано під науковим керівництвом PhD, доцента
ДЕСЯТКО А. М.

ЦИФРОВЕ РІШЕННЯ ДЛЯ ЕФЕКТИВНОГО ПЛАНУВАННЯ ЗАХОДІВ

ТЕРЕН І. В., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглядається програмне рішення для планування подій, призначене для оптимізації процесів організації та управління різноманітними заходами. Описано ключові функціональні можливості системи, включаючи календарне планування, управління учасниками, комунікацію, управління ресурсами, бюджетування та аналітику. Розглянуто типові архітектурні підходи до розробки таких рішень, зокрема багаторівневу та мікросервісну архітектури. Особливу увагу приділено вибору технологічного стеку, де як одна з ключових технологій для розробки серверної частини відзначається мова програмування Python з використанням мікрофреймворку Flask. Flask, завдяки своїй легкості та гнучкості, є ефективним інструментом для швидкої розробки API та веб-додатків, що забезпечують основну функціональність планера подій. Також обговорюються виклики та перспективи розвитку планерів подій, включаючи питання безпеки та інтероперабельності.

The article explores a software solution for event planning, designed to optimize the organization and management processes of various events. The key functionalities of the system are described, including calendar scheduling, participant management, communication, resource management, budgeting, and analytics. Typical architectural approaches to the development of such solutions, particularly multi-tiered and microservices architectures, are considered. Special attention is paid to the selection of the technology stack, where the Python programming language with the Flask microframework is highlighted as one of the key

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

technologies for developing the server-side. Flask, due to its lightweight nature and flexibility, is an effective tool for the rapid development of APIs and web applications that provide the core functionality of the event planner. The challenges and prospects for the development of event planners, including security and interoperability, are also discussed.

Актуальність теми: У сучасних умовах цифрової трансформації суспільства зростає потреба у впровадженні інноваційних інформаційно-комунікаційних технологій у всі сфери діяльності, зокрема у сферу організації та управління подіями. Зростання кількості масових заходів, ділових форумів, галузевих конференцій, освітніх семінарів та соціально значущих подій вимагає застосування високоефективних інструментів для їхнього комплексного планування, координації та моніторингу. Традиційні методи підготовки заходів, які ґрунтуються на ручному управлінні, розрізненій документації та неструктурованій комунікації, виявляються недостатніми в умовах високої динаміки інформаційних потоків та багатокомпонентності сучасних подій.

Особливої уваги набувають питання забезпечення безпеки інформаційного середовища, управління доступом до конфіденційних даних, а також швидкого реагування на зміни, що виникають під час підготовки або проведення заходу. У зв'язку з цим актуальною є розробка цифрового рішення, яке дозволяє централізовано керувати сценаріями заходів, автоматизувати процеси розподілу ролей, здійснювати контроль за дотриманням регламенту, а також інтегрувати функціонал для миттєвої адаптації до зовнішніх і внутрішніх змін. Такі програмні платформи мають не лише оптимізувати витрати часу та ресурсів, а й забезпечити прозорість усіх організаційних процесів.

Крім того, розвиток дистанційного та гібридного формату проведення заходів висуває нові вимоги до функціоналу цифрових систем, включаючи підтримку відеоконференцзв'язку, інтерактивних елементів, зворотного зв'язку з учасниками, аналітичних інструментів для оцінювання ефективності тощо. Це створює передумови для формування нового покоління програмного забезпечення, яке буде не лише інструментом планування, а й середовищем управління подієвим життям в умовах цифрового суспільства.

Отже, впровадження сучасного цифрового рішення для планування та управління заходами є не лише логічним етапом еволюції організаційної діяльності, а й важливим чинником забезпечення безпеки, ефективності та адаптивності управлінських процесів в умовах високотехнологічного розвитку.

Мета: Метою цієї роботи є розробка програмного рішення для планування та управління запланованими подіями, яке забезпечить зручний інтерфейс для організаторів та учасників, сприятиме ефективній координації дій, оптимізації ресурсів та підвищенню рівня безпеки під час проведення заходів.

Об'єкт дослідження: Процес планування, координації та управління подіями й заходами

Предмет дослідження: Предметом дослідження є інформаційна технологія планування, координації та управління подіями й заходами.

Завдання:

– Провести аналіз існуючих підходів та інструментів для організації та управління подіями.

– Розробити концептуальну та логічну моделі бази даних для зберігання інформації про події, учасників, розклад та ресурси.

– Обґрунтувати вибір мов програмування та технологій для реалізації програмного рішення (Python).

– Розробити інтерфейс користувача, який забезпечить зручну взаємодію організаторів з системою.

– Реалізувати програмне рішення з використанням обраних технологій, інтегрувавши базу даних та функціональні модулі.

У сучасному динамічному світі управління часом та ефективна організація подій набувають особливої актуальності. Як для окремих осіб, так і для організацій, здатність планувати, координувати та відстежувати різноманітні заходи є ключовим фактором успіху. У цьому контексті програмні рішення для планування подій, або так звані «планери подій», відіграють все більш важливу роль, надаючи користувачам потужні інструменти для оптимізації цього процесу.

Планери подій являють собою спеціалізоване програмне забезпечення, призначене для автоматизації та спрощення завдань, пов'язаних з організацією та проведенням різноманітних заходів – від особистих зустрічей та нарад до масштабних конференцій та фестивалів. Основна мета

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

таких рішень полягає у забезпеченні централізованого управління всіма аспектами події, починаючи від визначення дати та місця проведення, розсилки запрошень та реєстрації учасників, і закінчуючи складанням розкладу, управлінням ресурсами та аналізом результатів. Використання програмних планерів подій приносить значні переваги, серед яких слід виділити підвищення ефективності планування, зменшення адміністративних витрат, покращення комунікації між організаторами та учасниками, а також можливість отримання цінних аналітичних даних для оптимізації майбутніх заходів.

Функціональні можливості сучасних планерів подій

Сучасні планери подій пропонують широкий спектр функціональних можливостей, які можуть бути адаптовані до потреб різних типів заходів та організацій. Серед найбільш поширених функцій можна виділити:

Календарне планування та розклад: Інструменти для визначення дат і часу проведення подій, створення детальних розкладів з можливістю додавання окремих сесій, доповідей, перерв тощо. Візуалізація розкладу в зручному форматі (календар, список) для організаторів та учасників.

Управління учасниками: Можливості для розсилки запрошень (персоналізованих або масових), онлайн-реєстрації учасників, збору необхідної інформації (наприклад, побажання щодо харчування, участь у певних сесіях), а також управління списками гостей та квитками.

Комунікація та сповіщення: Інструменти для розсилки повідомлень та нагадувань учасникам про майбутні події, зміни в розкладі або іншу важливу інформацію. Підтримка різних каналів комунікації, таких як електронна пошта, SMS або push-сповіщення.

Управління ресурсами: Можливості для бронювання місць проведення, конференц-залів, обладнання (проектори, звукова апаратура), а також управління персоналом, залученим до організації події. Бюджетування та фінансовий облік: Інструменти для складання кошторису події, відстеження витрат та доходів, а також управління платежами за квитки або спонсорськими внесками.

Аналітика та звітність: Можливості для збору та аналізу даних про реєстрацію учасників, відвідуваність сесій, зворотний зв’язок (через опитування) та інші ключові метрики події. Формування звітів для оцінки ефективності проведеного заходу та прийняття рішень щодо майбутніх подій.

Інтеграція з іншими системами: Підтримка інтеграції з іншими програмними рішеннями, такими як CRM-системи (Customer Relationship Management), системи електронної пошти, соціальні мережі та платіжні шлюзи, для забезпечення безперебійного обміну даними та оптимізації робочих процесів.

Архітектурні підходи до розробки планерів подій

Розробка ефективного та масштабованого планеру подій може здійснюватися з використанням різних архітектурних підходів. Одним з найбільш поширених є багаторівнева архітектура, яка зазвичай включає наступні шари:

Рівень представлення (Presentation Tier): Відповідає за відображення інформації користувачеві та забезпечення взаємодії з системою. Для веб-орієнтованих планерів подій цей рівень зазвичай реалізується за допомогою веб-інтерфейсів (HTML, CSS, JavaScript) та може включати використання різних фреймворків для спрощення розробки інтерактивних елементів.

Рівень бізнес-логіки (Business Logic Tier): Містить основну функціональність системи, включаючи обробку даних, реалізацію бізнес-правил та управління робочими процесами, пов’язаними з плануванням та проведенням подій. Цей рівень може бути реалізований з використанням різних мов програмування та фреймворків (наприклад, Java, Python, .NET).

Рівень даних (Data Tier): Відповідає за зберігання, отримання та управління даними системи. Зазвичай реалізується за допомогою системи управління базами даних (СКБД), яка може бути реляційною (наприклад, MySQL, PostgreSQL, SQL Server) або нереляційною (NoSQL, наприклад, MongoDB, Cassandra), залежно від вимог до структури та обсягу даних.

Іншим поширеним підходом є мікросервісна архітектура, яка передбачає розбиття програми на невеликі, незалежно розгортаємі сервіси, кожен з яких відповідає за певну функціональність (наприклад, управління користувачами, управління подіями, розсилка повідомлень). Мікросервіси взаємодіють між собою через API (Application Programming Interface), що забезпечує гнучкість, масштабованість та незалежне розгортання окремих компонентів системи.

Вибір конкретної архітектури програмного рішення залежить від багатьох чинників, серед яких: складність системи, очікувані навантаження, вимоги до масштабованості, надійності та доступності, швидкість розробки, розмір і досвід команди, а також технологічні переваги обраного

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

стеку. В умовах постійно зростаючих вимог до продуктивності й адаптивності подібних систем, перевага часто надається мікросервісній архітектурі, яка дозволяє реалізовувати окремі функціональні компоненти як незалежні сервіси з можливістю їхнього автономного масштабування, розгортання й оновлення. Альтернативно, для менш складних або швидкозапускних проєктів може бути обрано монолітну чи серверлес архітектуру.

Технологічний стек для розробки планерів подій

Вибір відповідного технологічного стеку має вирішальне значення для успішної реалізації проєкту, оскільки він безпосередньо впливає на продуктивність, безпеку, вартість підтримки та швидкість розгортання цифрового планера.

Мови програмування: Back-end: Java (Spring Boot), Python (Django або FastAPI), PHP (Laravel), JavaScript (Node.js + Express), Ruby (Ruby on Rails) – забезпечують різні рівні продуктивності, масштабованості та підтримки; Front-end: HTML5, CSS3, JavaScript у поєднанні з React, Angular або Vue.js дозволяють створювати динамічні, адаптивні інтерфейси користувача з підтримкою реального часу.

Фреймворки значно прискорюють розробку, забезпечуючи вбудовану логіку для автентифікації, захисту, роботи з API та базами даних. Наприклад: Spring (Java), Django (Python), Laravel (PHP), Express.js (Node.js), Ruby on Rails – сприяють стандартизації коду та повторному використанню модулів.

*Системи управління базами даних (СКБД): р*еляційні: PostgreSQL, MySQL, SQL Server – ідеально підходять для транзакційних операцій та структурованих даних; NoSQL: MongoDB, Cassandra, Firebase Realtime Database – ефективні при роботі з неструктурованими даними, великою кількістю одночасних запитів і високою доступністю.

Сервери та середовища виконання: Apache, Nginx, Node.js runtime – забезпечують надійне HTTP-обслуговування, балансування навантаження, проксирування та інші серверні функції.

Хмарні платформи та сервіси: AWS (Elastic Beanstalk, Lambda, RDS), Google Cloud Platform (App Engine, Firestore), Microsoft Azure – надають можливість безперервної інтеграції, масштабування, розподілу навантаження, моніторингу та високої доступності. Контейнери (Docker, Kubernetes) також активно використовуються для розгортання компонентів системи.

*Інструменти безпеки: з*ахист на рівні API (OAuth 2.0, JWT), шифрування даних (TLS/SSL), багатофакторна автентифікація, аудит активності користувачів, виявлення аномальної поведінки – є критично важливими для забезпечення конфіденційності та цілісності даних.

DevOps-практики та CI/CD: Jenkins, GitLab CI/CD, GitHub Actions, Docker, Kubernetes – застосовуються для автоматизації процесів тестування, складання, розгортання та моніторингу цифрового рішення.

Підтримка мобільних пристроїв. Для забезпечення кросплатформенності інтерфейсу планера можуть використовуватись фреймворки React Native або Flutter, які дозволяють створювати мобільні додатки під Android та iOS з єдиною кодовою базою.

Виклики та перспективи розвитку планерів подій. Незважаючи на значні переваги, розробка та впровадження планерів подій також пов’язані з певними викликами. Одним з ключових викликів є забезпечення безпеки даних користувачів та інформації про події, особливо при обробці персональних даних та платіжної інформації. Необхідно вживати комплексних заходів для захисту від несанкціонованого доступу, витоку даних та кібератак.

Іншим важливим аспектом є забезпечення інтероперабельності з іншими системами та сервісами, які можуть використовуватися організаторами та учасниками подій. Підтримка стандартних протоколів обміну даними та наявність відкритих API є важливими для забезпечення безперебійної інтеграції.

Перспективи розвитку планерів подій пов’язані з подальшим впровадженням штучного інтелекту (ШІ) та машинного навчання (МН) для автоматизації рутинних завдань, персоналізації досвіду користувачів та надання інтелектуальних рекомендацій щодо планування та проведення заходів. Наприклад, ШІ може використовуватися для оптимізації розкладу на основі інтересів учасників, прогнозування відвідуваності або автоматичного підбору відповідних місць проведення та постачальників послуг.

Також очікується подальший розвиток мобільних рішень та інтеграції з соціальними мережами для покращення комунікації та залучення учасників. Розширення можливостей для проведення гібридних та віртуальних подій також є важливим напрямком розвитку, що вимагає інтеграції інструментів для онлайн-трансляцій, віртуальної взаємодії та спільної роботи.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

В цілому, планери подій продовжуватимуть відігравати важливу роль в організації різноманітних заходів, еволюціонуючи та адаптуючись до нових технологій та потреб користувачів.

Висновки. Таким чином, актуальність розробки цифрового рішення для ефективного планування заходів обумовлена сукупністю технологічних, організаційних та безпекових чинників, що впливають на сучасну подієву діяльність. Впровадження програмного забезпечення нового покоління дозволяє не лише підвищити ефективність управлінських процесів, а й забезпечити високий рівень адаптивності, гнучкості та захищеності в умовах постійно змінного цифрового середовища.

Цифровізація планування подій є важливим кроком у побудові інтелектуальних інструментів підтримки рішень, спрямованих на оптимізацію ресурсів, зменшення ризиків, покращення комунікації та підвищення якості сервісу для всіх учасників процесу. Таким чином, розробка та впровадження подібного рішення є стратегічно доцільним і практично значущим напрямом розвитку інформаційних технологій у сфері організації подій.

Список використаних джерел

1. Сайт документації Flask [Електронний ресурс] – Режим доступу: https://flask.palletsprojects.com/en/2.3.x/
2. Python Documentation [Електронний ресурс] – Режим доступу: https://docs.python.org/3/ (Офіційна документація мови програмування Python)
3. GitHub [Електронний ресурс] – Режим доступу: https://github.com/ (Платформа для розміщення та спільної розробки IT-проектів. Можна знайти приклади open-source планерів подій або окремих компонентів, а також документацію до різних бібліотек.)
4. Stack Overflow [Електронний ресурс] – Режим доступу: https://stackoverflow.com/ (Форум для програмістів з великою базою питань та відповідей з програмування, включаючи Python та Flask.)
5. Real Python [Електронний ресурс] – Режим доступу: https://realpython.com/ (Навчальний вебсайт з великою кількістю матеріалів з Python, включаючи веб-розробку.)
6. Medium [Електронний ресурс] – Режим доступу: https://medium.com/ (Платформа для публікацій на теми розробки, технологій та управління проектами. Використовуйте пошук для знаходження релевантних матеріалів.)
7. Вікіпедія (стаття про системи управління подіями) [Електронний ресурс] – Режим доступу: https://en.wikipedia.org/wiki/Event_management

Роботу виконано під науковим керівництвом канд. техн. наук, доцента
ХАРЧЕНКА О. А.

АДМІНІСТРУВАННЯ САЙТУ ТА РОЗРОБКА CRM-СИСТЕМИ

ТІМКО І. А., 2 курс 2м група, ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглянуто ключові аспекти адміністрування вебсайту та розробки Customer Relationship Management-системи. Проаналізовано основні етапи створення та підтримки вебресурсів, інтеграцію Customer Relationship Management з сайтом та перспективи їх розвитку.

The article examines the key aspects of website administration and Customer Relationship Management system development. The main stages of website creation and maintenance, Customer Relationship Management integration with the site, and development prospects are analyzed.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Актуальність. У сучасному бізнес-середовищі вебсайти це вкрай необхідна річ, тому що вони є не лише візитною карткою компанії, а й надважливим інструментом комунікації, продажів та обслуговування клієнтів. Через взаємодію з правильно побудованим сайтом з'являється велика можливість зацікавити нових клієнтів, які потенційно стають майбутніми покупцями, а CRM-системи дозволяють оптимізувати управління взаємодією з клієнтами, підвищуючи ефективність, безпеку бізнесу та спрощують процес роботи працівникам. У сучасному цифровому світі адміністрування вебсайту та розробка CRM-системи є важливими складовими ефективного управління бізнесом. Сайти забезпечують онлайн-присутність компанії, а CRM-системи допомагають автоматизувати роботу з клієнтами. Незважаючи на розвиток технологій, залишається багато нерозв'язаних питань щодо інтеграції та оптимізації цих інструментів, наприклад збільшення активності штучного інтелекту в нашому житті, який водночас можна вдало використовувати для спрощення бізнес процесів. Актуальність теми обумовлена необхідністю автоматизації бізнес-процесів та підвищення рівня безпеки вебресурсів [2–4].

Метою статті є дослідження процесів адміністрування вебсайту та розробки CRM-систем, що сприяють покращенню комунікації з клієнтами та автоматизації бізнес-процесів та спрощенню керування бізнес процесами й додавання можливості контролю праці.

Об'єктом дослідження є адміністрування вебресурсів та CRM-систем.

Предмет дослідження - методи та інструменти управління сайтом, автоматизації робочих процесів та покращення взаємодії з клієнтами.

Аналіз попередніх досліджень. Дослідженням питань адміністрування сайтів та CRM-систем займалися вітчизняні та зарубіжні науковці: Дж. МакКарті, В.І. Кунченко-Харченко, Л.А. Птіцина, О.М. Іванов та ін. У їхніх роботах висвітлено важливість автоматизації обліку клієнтів, але недостатньо уваги приділено безпеці даних та інтеграції вебресурсів.

Виклад основного матеріалу. Адміністрування вебсайту включає комплекс систематичних і технічно складних дій, спрямованих на забезпечення стабільного функціонування, захисту та актуальності інформаційного веб-ресурсу. Це постійний процес, який вимагає спеціальних технічних знань, стратегічного планування та особливого аналітичного підходу. Від ефективності адміністрування залежить безперебійна робота сайту і якість взаємодії з користувачами, що безпосередньо впливає на репутацію, статус, рівень довіри до компанії, рівень продажів, SEO-рейтинги та конкурентоспроможність. До ключових функцій адміністратора сайту належить моніторинг працездатності, який передбачає безперервне відстеження доступності вебресурсу в мережі Інтернет. Цей процес включає перевірку часу завантаження сторінок, ідентифікацію технічних збоїв та обробку серверних (500 Internal Server Error, 502 Bad Gateway) і клієнтських (404 Not Found, 403 Forbidden) помилок. За допомогою спеціалізованих інструментів (наприклад, UptimeRobot, Pingdom, Google Search Console) адміністратор може швидко реагувати на проблеми, щоб зменшити час простою ресурсу та забезпечити високу доступність для користувачів. Ще одним важливим аспектом є регулярне оновлення системного і прикладного програмного забезпечення, зокрема рушія сайту (CMS – Content Management System), тем оформлення, плагінів, модулів і скриптів.

Застаріле програмне забезпечення є однією з основних причин виникнення вразливостей, які можуть бути використані зловмисниками для несанкціонованого доступу до сайту або витоку даних. Оновлення не лише закривають «дірки» в безпеці, але й надають додатковий функціонал, покращують швидкодію ресурсу та сумісність із сучасними вебстандартами. Крім того, підтримка стабільної версії CMS допомагає уникнути конфліктів між різними модулями сайту, що особливо актуально для масштабованих вебпроєктів [1, 3].

До основних обов'язків адміністратора вебсайту входить:

1. Адміністрування вебсайту
2. Моніторинг працездатності сайту
3. Оновлення системного та прикладного ПЗ

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

4. Забезпечення безпеки
5. Управління контентом
6. Ведення аналітики

Оптимізація продуктивності веб-сайту є одним з найважливіших аспектів у забезпеченні швидкої та стабільної роботи ресурсу, що напряму впливає на користувацький досвід, рівень конверсії та пошукову видимість. Висока швидкість завантаження сторінок в декілька разів зменшує показник відмов, через миттєве підвищення задоволення користувача, покращує позиції у видачі пошукових систем і позитивно впливає на поведінкові фактори. Основними інструментами оптимізації виступають технології кешування, які дозволяють зберігати проміжні результати запитів у пам'яті сервера (наприклад, за допомогою Redis або Memcached). Це суттєво знижує навантаження на базу даних та прискорює час відповіді. Мініфікація CSS і JavaScript-файлів дозволяє зменшити їхній розмір, а отже – і час завантаження, що особливо важливо для мобільних користувачів.

Ще одним важливим чинником продуктивності є стиснення та адаптація графічного контенту. Зображення часто займають до 60% загального обсягу сторінки, тому їхнє зменшення через формати WebP, AVIF або використання інструментів, як-от TinyPNG, істотно прискорює завантаження ресурсу. Впровадження мережі доставки контенту (CDN, наприклад, Cloudflare або Amazon CloudFront) дозволяє розподіляти вміст сайту між серверами, розташованими ближче до кінцевих користувачів, що забезпечує мінімальну затримку передачі даних. Забезпечення безпеки ресурсу є не менш важливим напрямком, який охоплює як превентивні, так і реактивні заходи. Налаштування веб-фаєрволу (наприклад, ModSecurity) допомагає блокувати шкідливі запити до серверу, захищаючи від SQL-ін'єкцій, XSS-атак та інших загроз. Впровадження SSL-сертифіката дозволяє шифрувати передані дані між клієнтом і сервером, забезпечуючи конфіденційність і цілісність інформації. Захист від DDoS-атак може включати застосування рішень, які аналізують трафік у реальному часі та блокують підозрілу активність ще до того, як вона завдасть шкоди.

Обов'язковим елементом є налаштування системи автоматичного резервного копіювання як локального, так і хмарного з метою відновлення сайту у випадку збоїв, атак чи втрати даних. Щодо управління контентом, ефективна система управління вмістом (CMS) повинна забезпечувати зручне редагування, категоризацію і оновлення матеріалів. Важливо підтримувати актуальність інформації, регулярне оновлення публікацій, а також дотримання правил пошукової оптимізації. Це включає роботу з мета-тегами, alt-атрибутами зображень, правильну структуру заголовків (H1-H3), внутрішню перелінковку та розширення семантичного ядра сторінок. Крім того, важливо підтримувати адаптивність сайту його коректне відображення на всіх типах пристроїв, що також враховується пошуковими системами при розподілі. Ведення аналітики є необхідною умовою для оцінки ефективності вжитих заходів. Інтеграція інструментів аналітики таких як Google Analytics, Google Tag Manager, Hotjar та інші, дозволяє відстежувати кількість відвідувань, поведінкові сценарії користувачів, ефективність маркетингових кампаній. Завдяки цим даним адміністратор може приймати зважені рішення щодо поліпшення UX/UI, зміни структури сайту, тестування A/B-варіантів і зростання комерційної ефективності ресурсу в цілому [7].

CRM (Customer Relationship Management) – це система для організації ефективної взаємодії з клієнтами. Розробка CRM починається з визначення цілей і вимог: аналізуються потреби компанії, формулюються функціональні (керування контактами, облік звернень, автоматизація продажів) та нефункціональні (швидкість, масштабованість) вимоги (Рис.1). Далі відбувається проєктування архітектури, вибір типу системи – хмарної, десктопної або веб-орієнтованої – та побудова її основних модулів: управління контактами, завданнями, продажами, документами, звітністю, аналітикою, інтеграцією з поштовими сервісами [4,5].

Розробка власної CRM передбачає такі етапи:

1. Визначення цілей та вимог

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

2. Побудова архітектури системи
3. Вибір типу CRM: хмарна, десктопна або веб-орієнтована.
4. Розробка модулів: контакти, завдання, продажі, документообіг, звіти, аналітика, інтеграція з поштою.
5. Технічна реалізація
6. Інтеграція та тестування
7. Впровадження CRM-системи

Рис. 1. Компоненти CRM-системи та взаємозв'язок з іншими модулями

Джерело: розроблено автором за допомогою Canva

На етапі технічної реалізації застосовуються сучасні технології. Для backend використовуються фреймворки Laravel, Django або Node.js. Для frontend – React, Vue.js або Angular, що забезпечує зручний і динамічний інтерфейс користувача. Системи управління базами даних – PostgreSQL, MySQL або MongoDB – використовуються для зберігання інформації [8-10]. Комунікація з іншими сервісами здійснюється через RESTful API або GraphQL. Після цього реалізується етап інтеграції CRM із вебсайтом, електронною поштою, телефонією та іншими платформами. Впроваджуються webhook-сповіщення, push-повідомлення. Систему тестують на предмет продуктивності, стабільності, безпеки.

Завершальним етапом є впровадження CRM у компанії: налаштування ролей і прав доступу, навчання персоналу, міграція даних із попередніх систем. Інтеграція CRM із вебсайтом дозволяє автоматизувати збір заявок і звернень, покращити персоналізацію обслуговування (рис. 2).

Рис. 2. Схема інтеграції CRM-системи із сайтом

Джерело: розроблено автором за допомогою Canva

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Наприклад, форми зворотного зв'язку автоматично створюють записи в CRM, а онлайн-чат зберігає історію комунікацій у профілі клієнта. Дані аналітики щодо перегляду товарів, часу перебування на сторінці передаються в CRM, що дозволяє генерувати персоналізовані маркетингові пропозиції. Також можливе створення автоматизованих тригерних розсилок.

Серед функціональних можливостей CRM можна відзначити створення нових товарів (наприклад, списку наявних автомобілів в продажу) із подальшою публікацією на сайт, ведення обліку клієнтів і звернень, моніторинг активності користувачів, облік історії продажів, аналітику автосалону, архівне збереження інформації та статистичних даних.

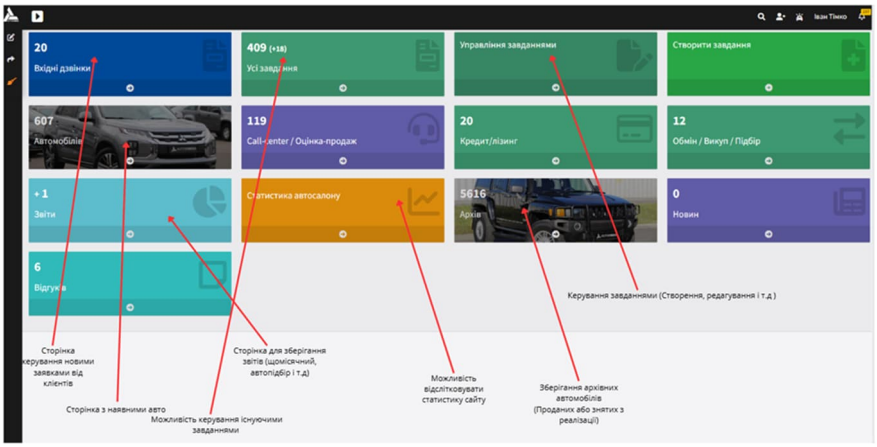


Рис. 3. Взаємодія з CRM системою

Джерело: розроблено автором в середовищі Pipedrive (знімок з екрану)

Правильне налаштування та керування CRM-системи в рази покращує працездатність вебсайту, водночас спрощуючи роботу з підтримкою актуальності інформації та обробці запитів нових або вже існуючих клієнтів. Розширює функціонал сайту та швидкість обробки нової вхідної інформації від користувачів вебсайту. На рис. 3. показано принцип взаємодії з вже існуючою CRM-системою, яка напряму працює з вебсайтом компанії, яка займається продажем вживаних та нових автомобілів. На цьому прикладі наявна кількість автомобілів з усієї мережі декількох автосалонів по території України, зберігаються контактні дані про нових та вже існуючих клієнтів, інформація про компанії партнери, архівний список автомобілів, технічні акти огляду авто. Є можливість додавати щотижневу або щомісячну звітність стовоно опрацювання вхідних заявок, так як до цієї CRM-системи вже під'єднано телефонію та автовідповідач на електронну пошту. Показана базова модель CRM-системи для внутрішнього використання, але CRM-система є гнучким інструментом, який можна будувати під особисті побажання та властивості проекту.

Можливості взаємодії сайту з CRM-системою

1. Створення нових автомобілів та публікація на сайт
2. Моніторинг активності клієнтів
3. Ведення автомобілів (інформація про стан, графа публікації та ін.)
4. Статистика автосалону
5. Архівне збереження публікацій

Висновки. Адміністрування вебсайтів та впровадження CRM-систем сприяє покращенню роботи компанії та підвищенню рівня обслуговування клієнтів. CRM-системи є важливими елементами ефективного бізнесу. Використання сучасних технологій дозволяє підвищити безпеку, автоматизувати роботу з клієнтами та покращити загальну продуктивність бізнес-процесів. Подальші дослідження можуть зосередитися на використанні штучного

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

інтелекту для аналізу взаємодії з клієнтами, покращення безпеки даних через блокчейн-технології та впровадження CRM-систем із підтримкою голосових асистентів та покращенню автоматизації процесів з використанням цих систем.

Список використаних джерел

1. Верба В.А. Інформаційне забезпечення управління розвитком компанії / В.А. Верба // Формування ринкової економіки: зб. наук. праць ДВНЗ «КНЕУ імені В. Гетьмана». – 2009. – № 22. – С. 145–154.
2. Кунченко-Харченко В.І. Інформаційні технології в управлінні / В.І. Кунченко-Харченко. – Київ: КНЕУ, 2014. – 376 с.
3. O'Brien, J. A., & Marakas, G. M. (2011). Management Information Systems. McGraw-Hill Education. https://www.mheducation.com
4. Salesforce. What is CRM? – Salesforce.com. https://www.salesforce.com/crm/what-is-crm/
5. Pipedrive. CRM Software for Small Business. https://www.pipedrive.com/uk
6. Moz. What is SEO? Beginner's Guide. https://moz.com/beginners-guide-to-seo
7. Google Developers. Search Engine Optimization (SEO) Starter Guide. https://developers.google.com/search/docs/fundamentals/seo-starter-guide
8. DigitalOcean. How To Secure Your Website With HTTPS. https://www.digitalocean.com/community/tutorials/https-explained
9. Django Project. Documentation. https://docs.djangoproject.com/
10. Node.js. Official Documentation. https://nodejs.org/docs/latest/api/

Роботу виконано під науковим керівництвом канд. фіз.-мат. наук, доцента
ГОРОХОВОЇ О. М.

ТЕХНОЛОГІЇ РОЗРОБКИ КРОС-ПЛАТФОРМНОГО ВЕБЗАСТОСУНКУ ДЛЯ ПРОВЕДЕННЯ ОПИТУВАНЬ

ФЕДОРОВСЬКИЙ Р. Д., 2 курс 2м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті проаналізовано ключові підходи до вибору технологій для створення кросплатформного вебзастосунку, орієнтованого на опитувань. Розглянуто переваги та недоліки поширених технологій бекенду, фронтенду та баз даних, які відповідають основним критеріям, таким як масштабованість, безпека, гнучкість у модифікації та кросплатформність. На основі цього здійснено обґрунтований вибір технологічного стеку для забезпечення ефективності та безпеки розроблюваного вебзастосунку.

The article analyzes key approaches to selecting technologies for developing a cross-platform web application aimed at conducting surveys and tests. It examines the advantages and disadvantages of popular backend, frontend, and database technologies that meet essential criteria such as scalability, security, flexibility for modification, and cross-platform compatibility. Based on this analysis, a well-grounded choice of a technology stack is presented to ensure the efficiency and security of the developed application.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Актуальність. Кожний проєкт починається з ідеї. Але самої ідеї зазвичай недостатньо, щоб зробити продукт успішним на ринку. Важливим етапом є правильний вибір технологій, оскільки помилка на цьому етапі може зробити кінцевий продукт неконкурентоспроможним та неефективним. Враховуючи, що сфера інформаційних послуг постійно розвивається, технічний аудит та аналіз засобів розробки залишаються завжди актуальними.

Метою статті є аналіз та порівняння переваг і недоліків різних технологій розробки бекенду та фронтенду для вибору оптимальних засобів для створення вебзастосунку для опитувань. Головні критерії оцінки – легкість модифікації коду, мультиплатформенність, малий обсяг застосунку масштабованість та швидкодія.

Об'єктом дослідження є розробка гнучкого вебзастосунку для проходження різноманітних видів опитувань.

Предмет дослідження – аналіз технологій, які сприятимуть найбільш ефективній розробці програмного забезпечення для створення вебзастосунку для проведення опитувань.

Аналіз попередніх досліджень. Питання порівняння різних технологій розробки досить часто підіймається перед розробниками, але зазвичай сильно залежить саме від поставленого завдання. Дослідженню допомогли праці таких вітчизняних та закордонних науковців як Бунке О. С., Годболт М., Клеппманн М., Тюріна Н. М., Цвіркун Л. І. тощо.

Виклад основного матеріалу. У результаті аналізу існуючих технологій ми прагнемо отримати оптимальний набір технологій для вебзастосунку, метою якого є проведення опитувань. Для отримання варто одразу виділити основні критерії серед яких важливими є:

– масштабованість – це є здатність системи ефективно працювати зі збільшенням кількості користувачів майбутнього застосунку. Цей критерій є одним із найважливіших, оскільки на першому етапі користувачів може бути досить мало, але потенційно, кінцевий продукт може зацікавити широку публіку.

– підтримка кросплатформності – це також дуже важливий критерій, оскільки в сукупності з масштабованістю дозволяє охоплювати широке коло пристроїв, на яких буде встановлений застосунок.

– гнучкість і легкість модифікації коду – важливо для швидкого розширення функціоналу, економії ресурсів та сил на розробку.

– безпека, оскільки аналіз виконується саме для застосунку для проведення опитувань, не виключна обробка персональних даних. І якщо користувач дасть згоду на їх обробку, треба забезпечити надійний захист цих даних, які деякий час можуть зберігатись на серверів.

Всі інші критерії також будуть враховуватись в разі потреби. [1][2]

Для створення застосунку нам потрібно обрати технології для бекенду, фронтенду та сховища даних.

При виборі технології для бекенду для порівняння візьмемо три варіанти, кожен з яких відповідає критерію «Гнучкість і легкість модифікації коду» оскільки у кожної з них існує велика спільнота з значною кількістю готових open-source модулів: Node.js, ASP.NET Core, Python (Django/Flask).

Node.js – платформа для обробки запитів користувачів на сервері. Відома своєю швидкістю обробки даних, одне з унікальних рішень, якщо метою є мінімізація засобів розробки, оскільки дозволяє уніфікувати мову програмування проєкту до JavaScript, яка зазвичай використовується тільки при фронтенд розробці. Основним недоліком є однопоточна природа, через що Node.js може стикатися з труднощами при обробці CPU-інтенсивних завдань. Оскільки це може суперечити критерію масштабованості, відкидаємо цей варіант.[3]

Django та Flask – феймворки, написані мовою Python, використовуються для схожих завдань, і при порівнянні з іншими технологіями часто згадуються разом. При оптимальному використанні дозволяють досягти порівняних результатів, тому в нашому випадку поєднаємо їх в одну групу. [4]

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

ASP.NET Core – фреймворк для веброзробки на мові C#. Обрана саме Core версія, оскільки вона задовільняє нашому критерію кросплатформності, легкості масштабування, має відкритий вихідний код.[5]

Таблиця 1

Порівняльна характеристика Python (Django/Flask) та ASP.NET Core

	Python (Django)	ASP.NET Core
Швидкість розробки	Висока	Середня
Швидкодія	Середня	Висока
Сувора типізація	Відсутня, наявна динамічна типізація	Присутня сувора статична типізація
Масштабованість	Добра	Дуже добра
Підтримка великих проектів	Середня	Висока
Використання ресурсів	Високе (більше ресурсів)	Середнє (менше ресурсів)
Екосистема та бібліотеки	Дуже велика	Велика

Джерело: агреговано на основі [5]

У даному випадку, для цілей проєкту більш доцільним буде використання ASP.NET Core.

Окрім швидкодії та кращої масштабованості, важливим чинником є саме сувора статична типізація мови C#, яка забезпечує написання більш зрозумілого, надійного та структурованого коду. Це значно спрощує підтримку, модифікацію й розширення функціоналу у процесі розвитку проєкту. Крім того, вбудовані механізми та стандартизовані засоби безпеки обробки даних значним чином знижують ризики витоку чи втрати конфіденційної інформації, що є критичним для вебзастосунку, призначеного для проведення опитувань.

Із варіантів для фронтенду також для аналізу обрано три популярні варіанти, кожен одразу відповідають критеріям безпеки та крос-платформності.

Blazor (WebAssembly and Server) – даний фреймворк повністю використовує мову програмування C#, що значним чином спрощує розробку у комбінації з ASP.NET Core. При цьому, є дуже безпечним, оскільки відповідає всім стандартам безпеки Microsoft. Окрім того, він має непогану швидкодію та можливість до масштабованості. Із недоліків – уповільнене завантаження додатку, середня гнучкість розробки, невелика екосистема, що обмежує вирішення деяких завдань розробки.

Angular – один із найпоширеніших front-end фреймворків, що розробляється і підтримується Google та базується на TypeScript з використанням архітектурного шаблону MVC. Обравши даний варіант, написання клієнтського коду буде виконано використовуючи TypeScript та HTML. MVC (Model-View-Controller) забезпечує надійну структуру для великих програм, а сам фреймворк має розвинену екосистему з великою кількістю сторонніх open source бібліотек. Також, наявна велика кількість готових функцій: потужні шаблони, маршрутизація, керування станом, обробка форм і вбудована підтримка двостороннього зв'язування даних між інтерфейсом користувача та моделлю. Що робить даний варіант неоптимальним для нашого випадку – фреймворк дуже навантажений; він представляє багато концепцій (модулі, декоратори, гаки життєвого циклу тощо), що є суттєвим для масштабування, але для початку проекту та до певного етапу є надлишковим. Інша проблема полягає в тому, що Angular часто оновлює версії (основні випуски щорічно); хоча команда Angular намагається звести до мінімуму небезпечні зміни, оновлення та міграції можуть бути нетривіальними, що ускладнює довготривалу підтримку.[6]

React – популярна бібліотека JavaScript для створення UI, підтримується Meta (раніше відома як Facebook). Чудово підходить для створення адаптивних вебзастосунків при правильному підборі додаткових інструментів. React використовує синтаксис JSX (JavaScript

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

XML) для змішування HTML з логікою JavaScript, а також віртуальний DOM для ефективного оновлення інтерфейсу користувача. Це дозволяє розбити інтерфейс користувача на компоненти, які можна використовувати багаторазово. Це забезпечує високу продуктивність і безперебійну роботу користувача навіть у дуже динамічних програмах. Однією з головних переваг є розмір екосистеми – ця бібліотека має величезну кількість сторонніх бібліотек та інструментів. Розробники можуть знайти пакети для маршрутизації (React Router), управління станом (Redux або інші), інструменти для обробки HTTP-запитів, бібліотеки для роботи з формами, валідацій даних та інші. Однак, ця гнучкість є і недоліком: вам зазвичай потрібно збирати додаткові бібліотеки. Екосистема React швидко розвивається; незважаючи на те, що основний API є досить стабільним, вводяться нові кращі практики, які час від часу можуть вносити критичні зміни або вимагати рефакторингу (наприклад, переходу від компонентів класу до хуків). [7]

Отже з урахуванням цих факторів, найбільш оптимальним буде вибір бібліотеки React, оскільки по мірі необхідності ми зможемо гнучким чином поступово розширювати архітектуру нашого застосунку, а також додавати до проєкту тільки ті інструменти, які нам необхідні на кожному з етапів розробки. Це спрощує початок розробки для маленької команди розробників, і не обмежує її при розширенні. В той же час, якщо якихось інструментів буде бракувати, завжди можна розширити React-стек та додати потрібні нам бібліотеки. Також, важливо зазначити – так як в даному проєкті вже передбачається гарний захист даних на бекенді питання захисту даних на фронтенді стає менш критичним, та базових заходів безпеки для клієнтської частини має бути достатньо [8].

Також, для зберігання даних потребується обрати певний тип бази даних. Тому далі розглянемо та проаналізуємо SQL та NoSQL бази даних і їх вплив на швидкість роботи застосунку та легкість масштабування.

Традиційні реляційні бази даних SQL, що мають чітко визначені схеми даних, підтримують транзакції, операції JOIN, та забезпечують ACID-транзакції (атомарність, узгодженість, ізольованість, надійність), незважаючи на сувору типізацію, що може бути недоліком, чудово порається з цілісністю даних, складними зв’язками та аналітичними запитами, які можуть бути корисними для створення великих статистичних звітів. Це може бути дуже корисним у розрізі роботи з даними опитувань, тестувань, та подібною інформацією. Також, оскільки база даних може містити чутливу інформацію, та необхідно враховувати ролеву модель, чітка схема та можливість будувати зв’язки JOIN можуть виявитись ключовим чинником при виборі типу бази даних.

Нереляційні бази даних NoSQL зберігають інформацію не в таблицях, а в інших форматах, будь то документи, ключ-значення, графи, тощо. Трьома основними перевагами таких баз даних: гнучкість (на відміну від реляційних чітко визначених схем мають структуру, що легко змінюється), швидкість простих запитів та висока горизонтальна масштабованість, яка при збільшенні навантажень на сервера дозволяє замість підвищення потужності існуючого серверу додавати додаткові сервери для розподілу навантаження. [9]

Враховуючи той факт, що для нас одним із ключових чинників є безпека можливих персональних даних бази даних, а також висока ймовірність наявної ролевої моделі, а також дуже бажано забезпечити цілісність та узгодженість, традиційна база даних – саме те, що нам потрібно.

Для порівняння реляційних моделей було обрано три популярні варіанти: Microsoft SQL Server (MS SQL), PostgreSQL та MySQL. Незважаючи на те, що усі ці моделі досить схожі, варто зробити наголос на деякі відмінності.

Microsoft SQL Server має певні обмеження для використання безкоштовною версією, непогану гнучкість (дозволяє створювати власні типи даних) та середню підтримку JSON (обмежена індексація колонок, що зберігають JSON, достатній набір функцій для обробки колонок, що зберігають цей тип даних). Із ключових переваг – відмінна інтеграція з .NET, що

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

може бути дуже важливим з урахуванням обраного ASP.NET Core у якості бекенд технології, для максимальної стабільності роботи застосунку. Не менш важливим є один із найкращих безпекових рішень.

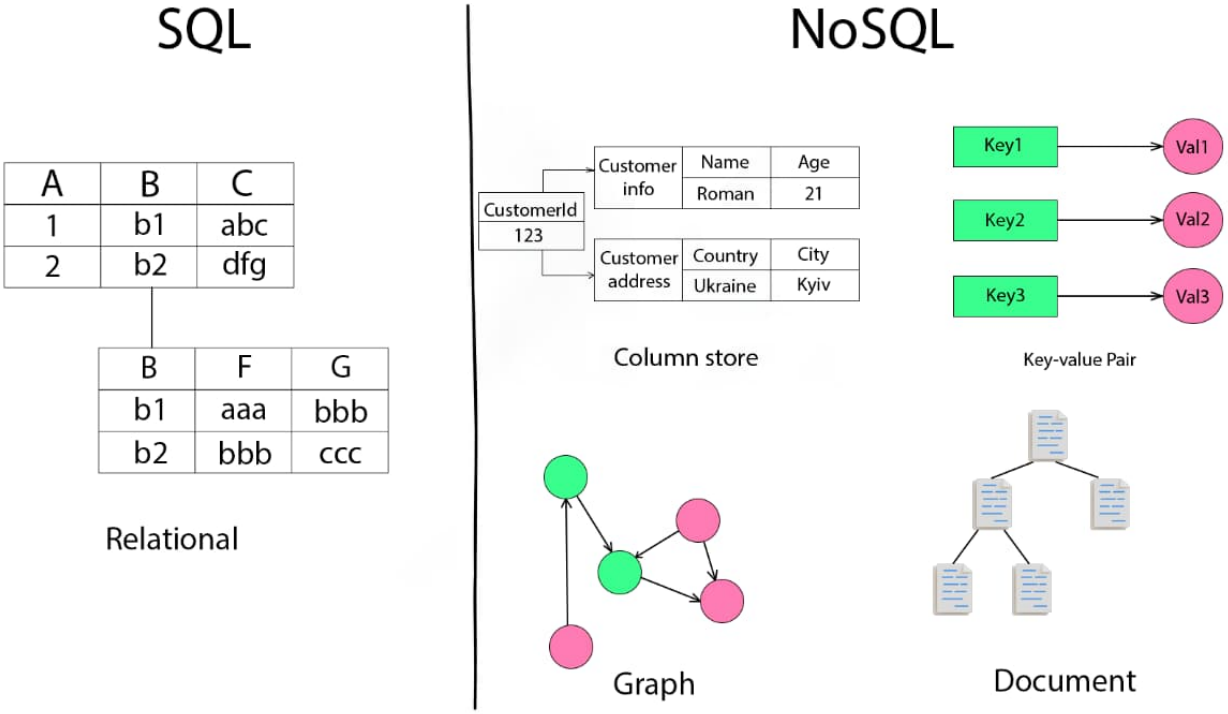


Рис. 1. Різниця між NoSQL та SQL наочно

Джерело: розроблено автором (знімок з екрану)

MySQL – безкоштовна, популярна та проста у використанні база даних, з обмеженою підтримкою складних структур. Підтримка JSON є, але вона поверхнева (документи зберігаються як текст, індексація здійснюється тільки через віртуальні колонки, які треба окремо створювати), функції роботи з колонками – базові (витяг полів, редагування).

PostgreSQL – найбільш популярна безкоштовна альтернатива до Microsoft SQL Server з навіть кращою підтримкою роботи з JSON. Має хороші можливості інтеграції з .NET високу безпеку шифрування.

Отже, коли домінуючим критерієм вибору є необхідність роботи з JSON варто обрати саме PostgreSQL, але у нашому випадку найважливішим чинником є саме інтеграція з .NET, тому було обрано саме Microsoft SQL Server як оптимальній для класичних застосунків.

Висновки. Стаття присвячена аналізу технологічного стеку для розробки вебзастосунку з функціоналом проведення опитувань. Ключовими критеріями відбору технологій були визначені: масштабованість, безпека, гнучкість кодової бази та крос-платформність.

На основі проведеного порівняльного аналізу, для реалізації бекенду було обрано ASP.NET Core, а для фронтенду – React, що обумовлено їх високою продуктивністю, суворим типізуванням та інтеграційними можливостями.

В якості системи управління базами даних обґрунтовано використання MS SQL Server, як рішення, що відповідає вимогам безпеки та цілісності даних.

Результатом дослідження є те, що запропонований технологічний стек забезпечує високу ефективність, безпеку та зручність супроводу застосунку, що сприятиме його подальшому розвитку та масштабуванню.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Список використаних джерел

1. *IEEE Entity Web Hosting – Home*. URL: https://ewh.ieee.org/r2/southern_nj/ BarbacciOct03.pdf (дата звернення: 15.03.2025).
2. The importance of software quality attributes. *SoftwareMill*. URL: https://softwaremill.com/the-importance-of-software-quality-attributes/ (дата звернення: 15.03.2025).
3. Team A. E. Pros and cons of node.js web app development. *AltexSoft*. URL: https://www.altexsoft.com/blog/the-good-and-the-bad-of-node-js-web-app-development/ (дата звернення: 15.03.2025).
4. GeeksforGeeks. Differences between django vs flask – geeksforgeeks. *GeeksforGeeks*. URL: https://www.geeksforgeeks.org/differences-between-django-vs-flask/ (дата звернення: 15.03.2025).
5. Spring Boot vs. Node.js vs. .NET Core vs. Django: a “celebrity face-off”. *Framework Training*. URL: https://www.frameworktraining.co.uk/news-insights/spring-boot-vs-node-js-versus-dot-net-core-vs-django/ (дата звернення: 15.03.2025).
6. Djirdeh H. How does angular compare to react?. *Telerik Blogs*. URL: https://www.telerik.com/blogs/how-does-angular-compare-to-react (дата звернення: 15.03.2025).
7. Introducing react.dev – react. *React*. URL: https://react.dev/blog/2023/03/16/introducing-react-dev (дата звернення: 15.03.2025).
8. Vishal-Siddhpara. React vs angular: which one is best for your next front-end project?. *Radixweb*. URL: https://radixweb.com/blog/react-vs-angular (дата звернення: 15.03.2025).
9. What is A nosql database? | fauna. *Fauna*. URL: https://fauna.com/blog/nosql-databases-non-relational-databases-explained (дата звернення: 15.03.2025).

Роботу виконано під науковим керівництвом д-ра техн. наук, професора
ЦЮЦЮРИ М. І.

СУЧАСНІ СТРАТЕГІЇ ЗАХИСТУ КОРПОРАТИВНОЇ КОМУНІКАЦІЇ: ПРОГРАМНІ РІШЕННЯ І ПРОТОКОЛИ ШИФРУВАННЯ

**ФЕОКТІСТОВ С. В., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»**

У статті розглянуто питання розробки та впровадження захищеного корпоративного програмного забезпечення для системи обміну повідомленнями. Зазначено особливості реалізації механізмів шифрування, автентифікації користувачів, контролю доступу та збереження конфіденційності в умовах зростання загроз інформаційній безпеці. Наведено огляд сучасних підходів та приклади застосування таких рішень у корпоративному середовищі.

The article considers the development and implementation of secure corporate software for messaging systems. It outlines the features of implementing encryption mechanisms, user authentication, access control, and maintaining confidentiality in the context of increasing threats to information security. An overview of modern approaches and examples of such solutions in the corporate environment is presented.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Актуальність. У статті досліджено актуальні аспекти створення та впровадження захищеного корпоративного програмного забезпечення для систем обміну повідомленнями. В умовах зростання кіберзагроз, таких як несанкціонований доступ, витік конфіденційної інформації, фішингові атаки та шкідливе програмне забезпечення, захист корпоративної комунікації стає критично важливою складовою інформаційної безпеки підприємств. Висвітлено основні вимоги до сучасних корпоративних месенджерів, зокрема впровадження наскрізного шифрування, багаторівневої автентифікації, політик контролю доступу та централізованого журналювання подій. Розглянуто архітектурні особливості реалізації захищених рішень, включно з використанням протоколів Signal, TLS 1.3 та алгоритмів AES, RSA, ECC.

Наведено порівняння популярних корпоративних інструментів для обміну повідомленнями (Wire Enterprise, Threema Work, Microsoft Teams) за критеріями рівня безпеки, підтримки самостійного хостингу та витрат на впровадження. Результати дослідження демонструють, що інтеграція захищених систем комунікації дозволяє підвищити стійкість підприємства до кіберзагроз, забезпечити відповідність сучасним вимогам законодавства та сприяти збереженню комерційної таємниці. Отримані висновки можуть бути використані як рекомендації для вибору або розробки власного захищеного програмного рішення у сфері корпоративного обміну повідомленнями.

Метою статті є дослідження принципів побудови, функціональні можливості та особливості реалізації захищеного програмного забезпечення корпоративної системи обміну повідомленнями.

Об'єктом дослідження є процес захищеного обміну повідомленнями в корпоративних інформаційних системах.

Предмет дослідження – програмне забезпечення захищеного обміну повідомленнями в корпоративному середовищі.

Аналіз попередніх досліджень. Питанням захищеного обміну повідомленнями приділяється значна увага у працях різних дослідників. Шнайєр Б.[2] акцентує увагу на важливості правильного вибору криптографічних алгоритмів, Касперські Є.[3] вказує на важливість багаторівневої моделі захисту у корпоративному середовищі, Верба В.А. у своїх працях з управління інформаційною безпекою вказує на значення інтеграції засобів захисту у всі бізнес-процеси компанії [1], Мейєр П. і Стаут К.[4] у дослідженнях з архітектури безпечних месенджерів (наприклад, Signal, Wickr) показують приклади застосування наскрізного шифрування та самоусувного зберігання даних.

Виклад основного матеріалу. Захищене корпоративне програмне забезпечення для обміну повідомленнями має ґрунтуватися на чітко визначених принципах інформаційної безпеки, що забезпечують цілісність, конфіденційність та доступність даних. У сучасних умовах найбільш актуальними є ті рішення, які поєднують високий рівень захисту з зручністю користування та гнучкою масштабованістю. Архітектура таких систем повинна враховувати наступні ключові компоненти [2, 3, 4]:

– Наскрізне шифрування (End-to-End Encryption, E2EE). Це базовий елемент захищеної комунікації, який гарантує, що повідомлення шифруються на пристрої відправника і розшифровуються лише на пристрої одержувача. Навіть сервери, через які проходить трафік, не мають доступу до відкритого тексту повідомлень. Такий підхід запобігає перехопленню даних на проміжних вузлах мережі.

– Багатофакторна автентифікація (Multi-Factor Authentication, MFA). З метою недопущення несанкціонованого доступу до облікових записів користувачів, системи повинні підтримувати автентифікацію з використанням кількох факторів, таких як пароль, одноразовий код із мобільного додатка або біометричні дані. Це значно ускладнює можливість компрометації облікового запису навіть у разі витоку одного з факторів.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

– Контроль доступу на основі політик (Policy-Based Access Control). Адміністратори мають змогу задавати політики, які визначають, хто і до яких типів інформації має доступ. Це можуть бути правила на основі ролей, геолокації, часу доби, типу пристрою тощо. Такий контроль забезпечує відповідність вимогам корпоративної безпеки та внутрішнім регламентам.

– Журналювання та аудит подій (Logging and Auditing). Для забезпечення повної прозорості взаємодій у системі, необхідним є впровадження механізмів журналювання дій користувачів та системних процесів. Це дозволяє оперативно виявляти порушення, аналізувати інциденти безпеки та забезпечувати відповідність нормативним вимогам.

– Функція самознищення повідомлень (Message Self-Destruct). Для підвищення рівня конфіденційності повідомлень, деякі системи реалізують можливість автоматичного видалення повідомлень через заданий проміжок часу після їх прочитання або доставки. Це особливо важливо при обміні критично важливою або конфіденційною інформацією.

Забезпечення конфіденційності, цілісності та автентичності повідомлень – це основоположні вимоги до будь-якої корпоративної системи обміну інформацією, особливо в умовах сучасного цифрового середовища, де ризики кіберзагроз зростають експоненційно. Конфіденційність гарантує, що зміст повідомлень доступний лише уповноваженим сторонам, цілісність передбачає захист даних від несанкціонованої модифікації під час передавання, а автентичність дозволяє впевнено встановити особу відправника повідомлення. Всі ці властивості критично важливі в комунікації між відділами компаній, підрозділами в різних країнах, партнерами, постачальниками, клієнтами, а також при взаємодії з державними структурами.

Для реалізації зазначених цілей сучасні інформаційні системи широко використовують криптографічні алгоритми, які умовно можна поділити на дві великі групи: симетричні та асиметричні. Симетричні алгоритми передбачають використання одного й того самого ключа як для шифрування, так і для розшифрування даних. Вони відзначаються високою швидкодією та ефективністю при обробці великих обсягів інформації, тому зазвичай використовуються для шифрування вмісту повідомлень або файлів. Водночас симетричні алгоритми потребують надійного каналу для обміну секретними ключами, що іноді ускладнює реалізацію безпечного зв’язку.

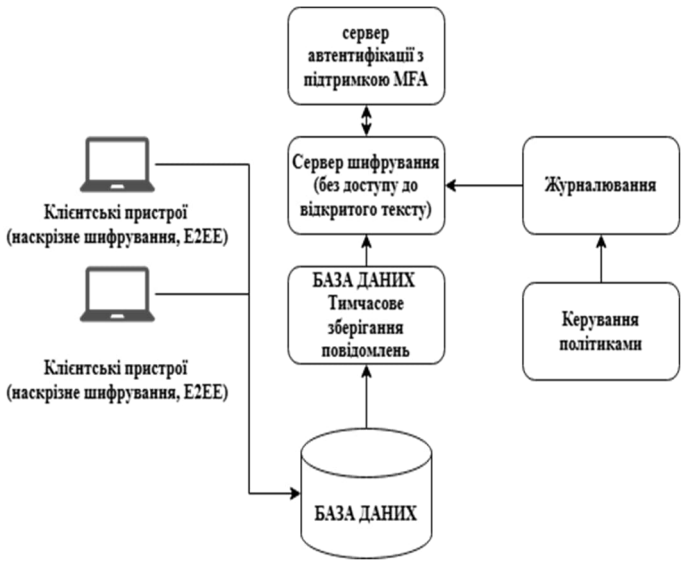


Рис. 1. Узагальнена архітектура захищеної корпоративної системи обміну повідомленнями

Джерело: розроблено автором

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Одним із найвідоміших і найнадійніших симетричних алгоритмів є AES (Advanced Encryption Standard) – розширений стандарт шифрування, прийнятий у 2001 році Національним інститутом стандартів і технологій США (NIST) як заміна застарілого DES. AES працює з блоками даних фіксованого розміру (128 біт) і підтримує ключі довжиною 128, 192 або 256 біт, що дозволяє адаптувати його під різні рівні безпеки залежно від потреб системи. Чим більша довжина ключа, тим вищий рівень захисту, але водночас зростають обчислювальні витрати.

Переваги AES полягають у його високій продуктивності, масштабованості та стійкості до криптоаналітичних атак, таких як диференційний та лінійний криптоаналіз. Алгоритм успішно реалізується як у програмному забезпеченні, так і в апаратному, що забезпечує його ефективне застосування як у потужних дата-центрах, так і в мобільних пристроях чи IoT-сенсорах. Завдяки своїй універсальності AES став де-факто стандартом для шифрування в більшості сучасних протоколів безпеки, таких як TLS/SSL, IPsec, BitLocker, а також у хмарних рішеннях Amazon Web Services, Microsoft Azure та Google Cloud Platform.

У корпоративному середовищі AES активно застосовується для шифрування листування в месенджерах, конфіденційних документів у файлових системах, резервних копій, баз даних та внутрішніх API-взаємодій між мікросервісами. Його використання дозволяє не лише забезпечити базову безпеку інформації, але й відповідати вимогам міжнародних стандартів, таких як ISO/IEC 27001, HIPAA, PCI DSS, що є критично важливим у сферах охорони здоров'я, фінансів, електронної комерції та державного управління.

Для забезпечення високого рівня безпеки під час обміну інформацією між сторонами в умовах відкритих або ненадійних мереж активно застосовуються асиметричні криптографічні алгоритми. Основною перевагою таких алгоритмів є можливість шифрування повідомлення відкритим ключем, який може бути публічно доступним, у той час як розшифрування здійснюється лише за допомогою відповідного закритого ключа, що зберігається у безпеці. Це дозволяє забезпечити як конфіденційність, так і автентичність переданої інформації, а також уникнути необхідності попередньої передачі спільного секретного ключа, як це передбачено в симетричних алгоритмах.

Серед найбільш поширених і перевірених часом асиметричних алгоритмів варто виділити RSA (Rivest–Shamir–Adleman), який заснований на складності факторизації добутку двох великих простих чисел. Цей алгоритм підтримує ключі великої довжини (2048, 3072, 4096 біт) і широко використовується в різноманітних протоколах, зокрема в TLS, PGP, SSH тощо. Однак одним із недоліків RSA є його відносна повільність, особливо при роботі з великими обсягами даних або у пристроях з обмеженими ресурсами, таких як мобільні телефони чи вбудовані системи.

На противагу RSA, сучасні розробники усе частіше використовують ECC (Elliptic Curve Cryptography) – криптографію на еліптичних кривих. Її математична основа забезпечує ту саму криптостійкість, що й RSA, але при значно коротшій довжині ключа (наприклад, ECC-256 еквівалентний RSA-3072 за рівнем безпеки). Це дозволяє суттєво зменшити обсяг обчислень і трафіку, що робить ECC особливо привабливим для мобільних пристроїв, IoT-рішень та хмарних сервісів із високою вимогою до продуктивності.

У сучасних системах захищеного обміну повідомленнями важливу роль відіграють спеціалізовані криптографічні протоколи, зокрема Signal Protocol, який став де-факто стандартом для реалізації наскрізного шифрування (E2EE). Цей протокол був розроблений Open Whisper Systems і сьогодні використовується в таких платформах, як Signal Messenger, WhatsApp, Google Messages, Skype та інших.

Протокол Signal має складну, але ефективну архітектуру. Для встановлення безпечного зв'язку між користувачами він застосовує X3DH (Extended Triple Diffie–Hellman) – модифікований варіант класичного обміну ключами, який дозволяє сторонам погодити спільний секрет, навіть якщо один з учасників не є онлайн у момент ініціації сесії. Цей механізм забезпечує початковий рівень секретності.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Після встановлення захищеного з’єднання протокол переходить до Double Ratchet алгоритму, який передбачає постійне оновлення ключів шифрування після кожного надісланого або отриманого повідомлення. Це означає, що для кожного нового повідомлення використовується унікальний одноразовий ключ. Така динамічна зміна ключів забезпечує одразу два важливих аспекти безпеки:

Першочергова секретність (Forward Secrecy) – гарантує, що компрометація поточного ключа не дозволяє зловмиснику розшифрувати попередні повідомлення.

Зворотну секретність (Backward Secrecy) – гарантує, що навіть якщо зловмисник отримає ключ, він не зможе прочитати наступні повідомлення, оскільки ключі для них ще не були згенеровані.

Ці властивості роблять Signal Protocol одним із найстійкіших до атак у разі витоку ключів або перехоплення трафіку. Його відкритий код, постійна підтримка спільнотою розробників та аудит безпеки з боку провідних фахівців дозволяють забезпечити високий рівень довіри серед користувачів і організацій, що орієнтовані на захист корпоративної або персональної інформації.

Комбінація сучасних асиметричних алгоритмів (RSA, ECC) та протоколів безпечного обміну ключами (X3DH, Double Ratchet) становить надійну основу для розробки корпоративних систем обміну повідомленнями, здатних ефективно протистояти загрозам компрометації, перехоплення чи модифікації даних.

Окрім Signal, у корпоративних системах також активно використовується протокол TLS (Transport Layer Security) версії 1.3, який забезпечує захист даних на транспортному рівні під час передавання між клієнтом і сервером. TLS 1.3 включає підтримку алгоритмів AEAD (Authenticated Encryption with Associated Data) і забезпечує суттєве скорочення часу встановлення з’єднання при підвищеній безпеці.

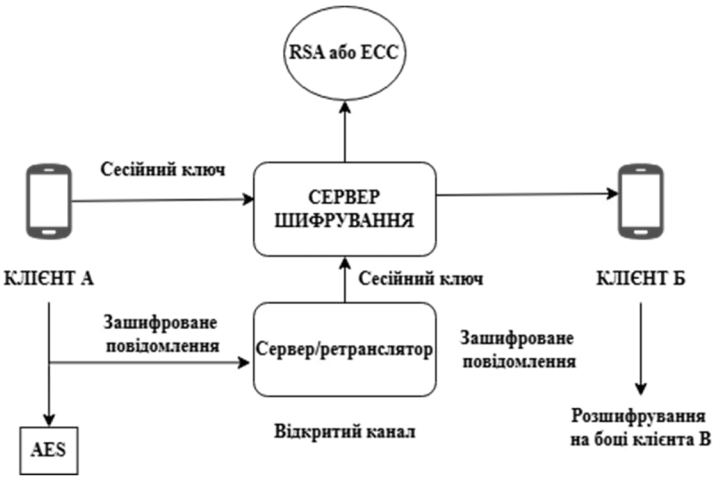


Рис. 2. Принцип роботи гібридної схеми шифрування в системі захищеного обміну повідомленнями

Джерело: розроблено автором

У сучасному цифровому середовищі, де обсяг чутливої інформації, що передається всередині компаній, стрімко зростає, питання безпеки комунікацій стає критично важливим. Використання звичайних, незахищених каналів обміну повідомленнями може призвести до витоку конфіденційних даних, компрометації корпоративних стратегій або навіть втрати конкурентних переваг. Саме тому на ринку корпоративного програмного забезпечення для обміну повідомленнями з’явилися високоспеціалізовані рішення, які об’єднують сучасні криптографічні технології з гнучкими інструментами управління доступом, контролем

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

пристроїв та відповідністю нормативним вимогам. Серед найбільш авторитетних і ефективних платформ, що здобули визнання у бізнес-середовищі та державному секторі, варто окремо виділити три продукти:

- Wire Enterprise;
- Threema Work;
- Microsoft Teams + Azure Information Protection.

Таблиця 1

Порівняння основних характеристик захищених месенджерів

Назва	Шифрування E2E	Автентифікація MFA	Власний сервер	Вартість
Wire Enterprise	+	+	+	Висока
Threema Work	+	+	+	Середня
Microsoft Teams	- (TLS)	+	+	Висока
Signal (custom)	+	+	+	Змінна

Джерело: розроблено автором

Wire Enterprise є відкритим і прозорим продуктом, який пропонує наскрізне шифрування всіх повідомлень, голосових і відеодзвінків, а також обміну файлами. Wire Enterprise орієнтований на корпоративне середовище й дозволяє компаніям розгортати систему на власній інфраструктурі або у хмарному середовищі, зберігаючи повний контроль над даними. Система відповідає вимогам Європейського регламенту із захисту персональних даних (GDPR) та стандартам ISO/IEC 27001. Серед функцій: автентифікація через LDAP/SSO, контроль доступу за ролями, журнали подій, інтеграція з SIEM-системами та підтримка безпечного зовнішнього обміну з партнерами [5].

Threema Work – це рішення розроблено спеціально для захищеної комунікації в межах компаній та державних установ. Threema Work підтримує наскрізне шифрування кожного елемента переданої інформації, включаючи текст, мультимедіа, дзвінки та опитування. Особливістю є можливість централізованого управління пристроями користувачів через Mobile Device Management (MDM), а також підтримка повної анонімності користувачів без обов'язкової реєстрації за номером телефону або e-mail [6].

Microsoft Teams у поєднанні з Azure Information Protection. Це рішення від Microsoft пропонує комплексну платформу для корпоративної взаємодії з широким функціоналом: чати, відеоконференції, інтеграція з Microsoft 365, спільна робота з документами. Для підвищення рівня безпеки використовується Azure Information Protection (AIP), що дозволяє класифікувати, маркувати та шифрувати документи і повідомлення. AIP підтримує контроль за розповсюдженням інформації як усередині, так і за межами компанії. Однак налаштування цієї системи вимагає високої кваліфікації персоналу, наявності розгорнутої IT-інфраструктури та постійного адміністрування для забезпечення відповідності політикам безпеки.

Забезпечення інформаційної безпеки в системах корпоративного обміну повідомленнями є ключовим чинником стійкості бізнесу. Впровадження сучасного програмного забезпечення з підтримкою наскрізного шифрування, контролю доступу та захищеної автентифікації дозволяє значно знизити ризики витоку конфіденційної інформації та забезпечити дотримання міжнародних стандартів безпеки.

Висновки. У результаті дослідження було визначено основні принципи побудови захищених корпоративних систем обміну повідомленнями з акцентом на реалізацію наскрізного шифрування, автентифікації та контролю доступу. Дослідження сучасних

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

протоколів, таких як Signal та TLS 1.3, демонструє ефективність використання гібридних схем шифрування у корпоративному середовищі. Проведене порівняння платформ Wire Enterprise, Threema Work та Microsoft Teams свідчить про різний рівень безпеки, гнучкість розгортання та вимоги до інфраструктури. Отримані результати можуть слугувати основою для впровадження та розробки захищених систем комунікації відповідно до потреб організації.

Список використаних джерел

1. Верба В.А. Інформаційне забезпечення управління розвитком компанії // Формування ринкової економіки: зб. наук. праць. – КНЕУ. – 2009. – №22. – С. 145–154.
2. Schneier B. *Applied Cryptography: Protocols, Algorithms, and Source Code in C.* – Wiley, 2015. – 784 с.
3. Kaspersky E. *Security architecture for enterprise messaging.* – Kaspersky Lab, 2020.
4. Peter Mayer, Kenneth Staut. *Secure Messaging Systems: Architectures and Protocols.* – IEEE Security & Privacy, 2021.
5. Wire. Wire Security Whitepaper. – [Електронний ресурс]. – Режим доступу: https://wire.com
6. Threema Work Documentation. – [Електронний ресурс]. – Режим доступу: https://work.threema.ch

Роботу виконано під науковим керівництвом PhD, доцента
ДЕСЯТКО А. М.

МОДЕЛЬ ВЕБДОДАТКА ДЛЯ ЦИФРОВОГО УПРАВЛІННЯ ЛОГІСТИКОЮ ДОСТАВКИ

ХОМЕНКО А. О., 2 курс 2м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

Сучасні логістичні процеси потребують ефективних інструментів для управління доставкою товарів. Веб-додатки відіграють ключову роль у підвищенні швидкості та точності перевезень, що робить їх незамінними в умовах зростаючої конкуренції. Метою статті є аналіз основних принципів розробки багаторівневого веб-додатку для управління процесом доставки товарів. Об'єктом дослідження є процес організації та контролю поставок, а предметом – методи й технології створення веб-додатків для логістики. У статті проведено аналіз існуючих досліджень у сфері цифрових логістичних рішень та визначено основні вимоги до таких систем.

Modern logistics processes require efficient tools for managing goods delivery. Web applications play a key role in increasing the speed and accuracy of transportation, making them indispensable in a highly competitive environment. The purpose of this article is to analyze the main principles of developing a multi-level web application for managing the delivery process. The object of the study is the process of organizing and controlling deliveries, while the subject is the methods and technologies for creating web applications for logistics. The article analyzes existing research in the field of digital logistics solutions and defines the main requirements for such systems.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

*Актуальність***.** Сучасні логістичні компанії стикаються з викликами, пов'язаними із зростанням обсягів перевезень та необхідністю забезпечення прозорості доставки. Традиційні методи управління транспортними потоками поступаються автоматизованим рішенням, які використовують сучасні веб-технології для оптимізації процесів та покращення якості обслуговування клієнтів. Використання багаторівневих веб-додатків дозволяє ефективно управляти ресурсами, інтегрувати зовнішні сервіси та забезпечувати високу продуктивність системи.

Мета статті є розробка архітектурного рішення для багаторівневого веб-додатку, що дозволить автоматизувати управління процесом доставки товарів, покращити контроль за логістичними операціями та мінімізувати ризики втрати або затримки вантажів.

Об'єкт дослідження є процеси управління логістикою в електронній комерції та потреба в автоматизованих веб-додатках для їх оптимізації.

Предмет дослідження **є** архітектурні підходи, методи розробки та інтеграційні механізми багаторівневих веб-додатків для управління процесом доставки товарів.

Аналіз попередніх досліджень. Раніше вчені та розробники пропонували різні підходи до створення логістичних платформ, використовуючи монолітні та мікросервісні архітектури. Дослідження показали, що мікросервісна модель дозволяє досягти кращої масштабованості та гнучкості у процесі розгортання. Крім того, використання REST API та GraphQL значно покращує інтеграцію веб-додатку із зовнішніми сервісами, такими як платіжні системи, служби геолокації та кур'єрські служби. Проте залишаються відкритими питання безпеки, продуктивності та ефективного управління базами даних, що вимагає подальшого дослідження.

Ефективність логістичних процесів суттєво впливає на загальну продуктивність підприємств, що займаються транспортуванням та доставкою товарів. Використання сучасних інформаційних технологій дозволяє оптимізувати логістичні процеси, зменшити витрати, скоротити час доставки та підвищити рівень обслуговування клієнтів. Одним із найефективніших рішень для досягнення цих цілей є розробка багаторівневого веб-додатку, який забезпечує автоматизацію всіх етапів логістичного процесу.

Архітектура багаторівневого веб-додатку відіграє ключову роль у забезпеченні стабільної та ефективної роботи системи управління процесом доставки товарів. Одним із найважливіших її компонентів є клієнтський рівень, який безпосередньо взаємодіє з користувачами, забезпечуючи зручний доступ до всіх необхідних функцій.

Користувачі системи можуть бути розділені на кілька основних груп: водії, логісти та клієнти. Для кожної з цих груп розроблено окремі дашборди, які містять відповідну інформацію та дозволяють здійснювати необхідні операції. Водії можуть отримувати деталі щодо замовлень, оптимізувати свої маршрути та звітувати про доставку. Логісти мають можливість контролювати рух транспорту, розподіляти завдання між водіями та аналізувати ефективність логістичних процесів. Клієнти, у свою чергу, можуть відстежувати статус своїх замовлень, отримувати сповіщення про зміни та взаємодіяти з підтримкою у разі потреби.

Однією з ключових функцій клієнтського рівня є інтеграція картографічних сервісів, що дозволяє здійснювати моніторинг маршрутів у реальному часі. Це допомагає водіям швидко орієнтуватися на місцевості, обирати найоптимальніші шляхи та уникати заторів. Для логістів це є незамінним інструментом для планування та оперативного коригування маршрутів у разі виникнення непередбачуваних ситуацій.

Важливу роль у роботі клієнтського рівня відіграють механізми авторизації та безпеки. Для захисту даних користувачів впроваджуються різні методи автентифікації, включаючи двофакторну авторизацію та шифрування переданих даних. Це гарантує безпечний доступ до системи та захист конфіденційної інформації.

Таким чином, клієнтський рівень є основою зручної та ефективної взаємодії користувачів із системою доставки товарів. Його функціональність включає дашборди для

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

різних категорій користувачів, картографічні сервіси для відстеження маршрутів у реальному часі та надійні механізми авторизації для забезпечення безпеки даних. Впровадження цих рішень сприяє підвищенню ефективності логістичних процесів і покращенню загального користувацького досвіду.

Серверний рівень в веб-додатку відіграє важливу роль у забезпеченні взаємодії між користувачами та системою. Це рівень, на якому здійснюється обробка запитів, виконання бізнес-логіки та інтеграція з іншими зовнішніми сервісами. Користувачі взаємодіють із сервером через клієнтський інтерфейс, надсилаючи запити, на які сервер відповідає відповідними даними або діями.

Однією з основних функцій серверного рівня є опрацювання замовлень. Система повинна мати можливість створювати замовлення, редагувати їх, а також відслідковувати статуси цих замовлень у реальному часі. Сервер також відповідає за формування маршрутів для доставки товарів, враховуючи дані про місцезнаходження клієнтів та водіїв. Алгоритми на сервері розраховують найкоротші або найбільш оптимальні маршрути, що дозволяє знизити витрати на паливо та час доставки.

Крім того, серверний рівень управляє статусами доставки. Це включає в себе відстеження та оновлення стану кожного замовлення на різних етапах доставки, що дає змогу клієнтам і водіям отримувати актуальну інформацію про їхні замовлення. Оновлення статусу також здійснюється у реальному часі, щоб усі зацікавлені сторони могли отримувати оперативну інформацію.

Серверний рівень також відповідає за інтеграцію з зовнішніми API, такими як картографічні сервіси (наприклад, Google Maps API). Це дозволяє додатку інтегрувати функціональність, пов’язану з побудовою маршрутів, геолокацією та відображенням візуальних даних на карті.

Рівень бази даних у веб-додатку відповідає за збереження та управління всією інформацією про замовлення, клієнтів, водіїв, маршрути і статуси доставки. Використання реляційних баз даних, таких як PostgreSQL, дозволяє зберігати дані в структурованому вигляді, що забезпечує швидкий доступ до інформації та ефективне оброблення запитів. База даних зберігає всю необхідну інформацію про користувачів, маршрути доставки, транзакції, а також історію змін статусів, що дозволяє проводити аналіз даних та формувати різноманітні звіти. Реляційна структура бази даних дозволяє забезпечити надійне зберігання даних і гарантує їхню цілісність.

Створення веб-додатку для управління процесом доставки товарів передбачає впровадження широкого спектра функціональних можливостей, що сприяють автоматизації логістичних процесів, підвищенню їхньої ефективності та покращенню користувацького досвіду.

Однією з ключових функцій системи є управління замовленнями. Користувачі мають змогу створювати нові замовлення, редагувати їхні деталі, а також відстежувати поточний статус доставки. Система забезпечує автоматичне оновлення інформації про зміну стану замовлення, надсилаючи відповідні сповіщення клієнтам та логістам. Це дозволяє оперативно реагувати на зміни в процесі доставки, запобігати можливим затримкам та підвищувати рівень обслуговування.

Ще однією важливою особливістю є оптимізація маршрутів, яка реалізується завдяки алгоритмам пошуку найкоротших та найефективніших шляхів. Враховуючи такі фактори, як трафік, погодні умови та відстань між пунктами доставки, система автоматично формує маршрути, що мінімізують витрати на паливо та скорочують загальний час транспортування товарів. Це особливо важливо для компаній, що працюють із великим обсягом замовлень і прагнуть знизити операційні витрати.

Не менш значущим компонентом є моніторинг у реальному часі. Завдяки інтеграції GPS-трекінгу менеджери логістики можуть відстежувати переміщення транспортних засобів

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

у будь-який момент, що сприяє підвищенню прозорості всього процесу доставки. Клієнти також отримують доступ до інформації про місцезнаходження своїх замовлень, що зменшує кількість запитів щодо статусу доставки та підвищує рівень довіри до сервісу.

Для забезпечення зручності оплати система підтримує інтеграцію з платіжними сервісами. Це дає змогу клієнтам здійснювати безготівкові платежі, що значно пришвидшує процес оформлення замовлень та усуває потребу в розрахунках при отриманні товару. Автоматичне виставлення рахунків та ведення фінансових операцій у системі дозволяє підприємствам більш ефективно контролювати фінансові потоки та запобігати можливим помилкам у розрахунках.

Завдяки впровадженню цих функцій веб-додаток забезпечує комплексний підхід до управління процесом доставки товарів, поєднуючи зручність для користувачів із високою ефективністю логістичних операцій.

Для створення багаторівневого веб-додатку для управління процесом доставки товарів використовується широкий спектр сучасних технологій. Основною мовою програмування є JavaScript, що забезпечує ефективну розробку як на клієнтському, так і на серверному рівнях. На фронтенді використовується React, що дозволяє створювати зручний і швидкий інтерфейс користувача, тоді як серверна частина реалізована на Node.js, що забезпечує високу продуктивність і можливість обробки великої кількості запитів одночасно.

Для зберігання даних застосовується реляційна база даних PostgreSQL, яка дозволяє ефективно працювати зі складними запитами та забезпечує збереження цілісності даних. Завдяки використанню механізмів індексації та оптимізації запитів система швидко отримує необхідну інформацію, що особливо важливо для великих логістичних компаній. PostgreSQL забезпечує надійне збереження даних про замовлення, клієнтів, маршрути і статуси доставки, що дозволяє здійснювати їх швидку обробку та аналіз.

Значну роль у роботі системи відіграють хмарні сервіси. AWS використовується для хостингу серверної частини, що забезпечує надійність і масштабованість додатку. Firebase застосовується для реального часу оновлення даних, зокрема для повідомлень про статус замовлення та змін у маршрутах. Завдяки цим технологіям система здатна ефективно обробляти великі обсяги запитів і даних, що дозволяє користувачам отримувати актуальну інформацію в режимі реального часу.

Для взаємодії з картографічними сервісами використовується Google Maps API. Це дозволяє системі будувати маршрути, відстежувати місцезнаходження транспорту та розраховувати найоптимальніші шляхи доставки. Крім того, для фінансових транзакцій інтегровано Stripe API, який забезпечує можливість швидких і безпечних безготівкових платежів. Завдяки інтеграції цих сервісів система надає зручний, стабільний і безпечний досвід для всіх учасників процесу доставки товарів.

У табл. 1 представлена інформація, що підсумовує використані технології та їхнє призначення:

Таблиця 1

Використані технології щодо розробки веб-додатку для управління доставкою товарів

№ пор	Технологія	Призначення
1	JavaScript (React)	Фронтенд-розробка, створення UI
2	JavaScript (Node.js)	Серверна логіка, обробка запитів
3	PostgreSQL	База даних, збереження інформації
4	AWS	Хостинг серверної частини
5	Firebase	Оновлення даних у реальному часі
6	Google Maps API	Побудова маршрутів, геолокація
7	Stripe API	Обробка платежів

Джерело: розроблено автором

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Завдяки такому підходу до вибору технологій система стає масштабованою, продуктивною та зручною для всіх користувачів, забезпечуючи ефективне управління логістичними процесами.

Продовжуючи тему розробки багаторівневого веб-додатку для управління процесом доставки товарів, важливим аспектом є безпека даних, адже система працює з чутливою інформацією, такою як особисті дані користувачів, замовлення, фінансові транзакції та геолокаційні дані. Безпека є одним з основних пріоритетів для розробників та адміністраторів системи, оскільки неналежне забезпечення захисту може призвести до витоку інформації, втрати довіри клієнтів та навіть фінансових збитків.

Одним із головних елементів забезпечення безпеки є шифрування даних. Для захисту переданих даних використовуються сучасні протоколи безпеки, зокрема SSL/TLS, що дозволяють зашифрувати інформацію, яка передається між сервером і клієнтом. Завдяки цьому навіть у разі перехоплення даних зловмисниками вони не зможуть отримати доступ до чутливої інформації. Цей захід гарантує, що вся комунікація між користувачем і сервером залишається конфіденційною.

Крім того, важливим аспектом є шифрування даних, що зберігаються в базах даних. Для цього використовуються алгоритми симетричного або асиметричного шифрування, що дозволяє захистити персональні дані клієнтів, платіжну інформацію та інші чутливі дані від несанкціонованого доступу. Зберігання паролів користувачів також передбачає використання сучасних методів хешування, таких як bcrypt, які роблять неможливим відновлення оригінальних паролів навіть у разі витоку бази даних.

Ще одним важливим елементом безпеки є авторизація та аутентифікація користувачів. Веб-додаток передбачає наявність механізмів для того, щоб лише авторизовані користувачі мали доступ до певних функцій системи. Це може бути досягнуто через використання двофакторної аутентифікації, коли для доступу до акаунту користувачеві потрібно не тільки ввести пароль, але й підтвердити свою особу через додатковий код, надісланий на мобільний телефон або електронну пошту.

Для захисту від атак типу SQL-ін’єкцій, які можуть виникнути при неправильній обробці запитів до бази даних, розробники повинні використовувати підготовлені запити або параметризовані запити. Це дозволяє запобігти можливості для зловмисників вводити шкідливий код у поля вводу та викликати небажані дії на сервері.

Також важливо звернути увагу на захист від атак, спрямованих на викрадення сесій користувачів, таких як атаки типу «Man-in-the-Middle». Для запобігання таким атакам використовуються механізми безпеки, зокрема захист сесійних куків через встановлення атрибутів HttpOnly та Secure, що гарантує, що сесії можуть бути використані тільки через HTTPS-з’єднання.

Не менш важливим аспектом є регулярне оновлення програмного забезпечення. Безпека системи залежить від того, наскільки швидко і ефективно реагують на виявлені уразливості. Для цього застосовуються регулярні оновлення програмного забезпечення, патчі та перевірки на наявність вразливостей, що дозволяє своєчасно закрити можливі прогалини в безпеці.

Не можна також забувати про моніторинг системи на наявність підозрілої активності. Використання засобів для збору і аналізу логів, а також впровадження систем виявлення вторгнень (IDS) допомагає швидко виявити спроби несанкціонованого доступу та попередити можливі загрози. Такий підхід дозволяє оперативно реагувати на загрози та мінімізувати можливі втрати.

Загалом, забезпечення безпеки даних у веб-додатку є комплексним завданням, яке включає різноманітні методи і підходи. Приділення уваги кожному аспекту безпеки на всіх етапах розробки та експлуатації додатку дозволяє забезпечити високий рівень захисту від потенційних загроз і гарантувати безпеку користувачів.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Висновки. Розробка багаторівневого веб-додатку для управління процесом доставки товарів є складним, але надзвичайно важливим завданням, яке дозволяє значно підвищити ефективність логістичних процесів. Використання клієнт-серверної архітектури забезпечує гнучкість та масштабованість системи, а інтеграція з зовнішніми API, такими як Google Maps API та Stripe API, дозволяє розширити її функціональні можливості.

Головною перевагою такого підходу є автоматизація ключових процесів, зокрема управління замовленнями, оптимізація маршрутів та контроль доставки в реальному часі. Використання сучасних технологій, таких як Node.js, React, PostgreSQL та хмарні сервіси AWS, сприяє створенню продуктивної та надійної системи.

Завдяки впровадженню таких рішень компанії можуть зменшити витрати на логістику, покращити якість обслуговування клієнтів та оптимізувати використання ресурсів. У майбутньому розвиток штучного інтелекту та аналітики даних відкриє нові можливості для прогнозування попиту, удосконалення процесів доставки та підвищення загальної ефективності системи.

Список використаних джерел

1. Ващенко В.В. Сучасні підходи до розробки логістичних інформаційних систем // Вісник Національного університету «Львівська політехніка». – 2021. – № 6. – С. 45–52.
2. Ковальчук О.П. Впровадження цифрових технологій у сферу логістики: переваги та виклики // Економічний вісник. – 2020. – № 3. – С. 112–118.
3. Давиденко О.В. Розробка веб-додатків для автоматизації процесів доставки товарів: основні принципи та підходи // Науковий вісник Миколаївського національного університету імені В. О. Сухомлинського. – 2022. – № 3. – С. 34–41.
4. Якубович А.М. Інформаційні технології в логістичних системах: роль і перспективи розвитку // Вісник Національної академії управління. – 2021. – № 2. – С. 18–25.
5. David J. Logistics and Supply Chain Management Systems. New York: Springer, 2019. – 350 p.
6. Zhang L. Optimization of Delivery Process Using Web-Based Applications. Journal of Logistics Technology, 2022, vol. 18, issue 2, pp. 25–37.

Роботу виконано під науковим керівництвом канд. техн. наук, доцента
ХАРЧЕНКА О. А.

АНАЛІЗ АРХІТЕКТУРНИХ ПАТЕРНІВ ДЛЯ МІКРОСЕРВІСНОЇ АРХІТЕКТУРИ: ВИБІР ОПТИМАЛЬНОГО ПІДХОДУ ДЛЯ РОЗРОБКИ МАСШТАБОВАНИХ СИСТЕМ

ШАМРИЛО Б. М., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті здійснено аналіз архітектурних патернів у мікросервісах з метою визначення оптимального підходу для розробки масштабованих систем. Для дослідження було обрано два основні архітектурні патерни – Command Query Responsibility Segregation та Event Sourcing, оцінено їх вплив на продуктивність та масштабованість. Розглянуто їх переваги та недоліки, а також вплив на розробку і підтримку мікросервісних систем. Рекомендації, представлені в статті, допоможуть розробникам обрати найефективніший архітектурний патерн для створення надійних та високопродуктивних мікросервісних додатків.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

The article analyzes architectural patterns in microservices in order to determine the optimal approach for developing scalable systems. Two main architectural patterns were selected for the study – Command Query Responsibility Segregation and Event Sourcing, and their impact on performance and scalability was assessed. Their advantages and disadvantages were considered, as well as the impact on the development and support of microservice systems. The recommendations presented in the article will help developers choose the most effective architectural pattern for creating reliable and high-performance microservice applications.

Актуальність. При розробці сучасних програмних додатків розробники часто зіштовхуються з проблемою недостатньої гнучкості додатків та неможливістю масштабувати окремий компонент додатку, коли навантаження зростає саме на нього. Багато систем, які працюють всередині окремої компанії чи доступні публічно, були написані давно і є застарілими. Вони можуть бути не розраховані для обробки навантажень, які з часом тільки збільшуються. Тож розробники під час рефакторингу такого коду зіштовхуються з виділенням частин коду великої системи в менші логічні компоненти.

На фоні цього мікросервісна архітектура стала популярною моделлю розробки програмного забезпечення саме завдяки тому, що пропонує вирішення цих проблем. Проте, вибір правильного архітектурного патерну має критичне значення для успішної реалізації мікросервісів. Зростаюча складність систем вимагає від розробників вибору оптимальних рішень, що забезпечують продуктивність, масштабованість і надійність

Метою статті є оцінка переваг і недоліків у використанні паттернів Command Query Responsibility Segregation та Event Sourcing, щоб дати розуміння, коли і як їх доцільно використовувати.

Предмет дослідження - мікросервісна архітектура.

Мікросервісна архітектура – це стиль побудови застосунків, при якому програмний продукт складається з незалежних, автономних сервісів, кожен з яких відповідає за певну функціональність. Кодові бази зростають, коли ми пишемо код для додавання нових функцій. З часом може бути важко зрозуміти, де потрібно внести зміни, оскільки кодова база дуже велика. Незважаючи на прагнення до чітких, модульних монолітних кодових баз, часто ці абстрактні межі руйнуються в процесі розробки. Код, пов'язаний зі схожими функціями, починає розширюватися більше, що ускладнює виправлення помилок або імплементацію нового функціоналу [2].

У монолітній системі ми боремось з цими проблемами, намагаючись зробити наш код більш згрупованим, часто шляхом створення абстракцій або модулів. Групування - прагнення відокремити та ізолювати пов'язаний код - важлива концепція, коли ми думаємо про мікросервіси. Це підтверджується визначенням принципу єдиної відповідальності Роберта С. Мартіна, який говорить: «Зберіть разом те, що змінюється з однієї причини, і відокремте те, що змінюється з різних причин» [7].

Мікросервіси використовують такий же підхід до незалежних сервісів. Ми зосереджуємо наші сервіси на межах бізнесу, роблячи очевидним, де знаходиться код для певної частини функціональності. І ізолюючи цей сервіс в чітких межах, ми уникаємо спокуси розширення цих меж, з усіма пов'язаними з цим труднощами, до яких воно може призвести [5].

Мікросервісна архітектура допомагає вирішити цілий ряд проблем монолітної архітектури [5]: складність управління: зі зростанням кодової бази монолітного додатку його структура стає складнішою, що ускладнює розуміння логіки його роботи людиною; обмежена масштабованість; повільні цикли розробки; складність у впровадженні нових технологій та оновленні старих бібліотек; вразливість до помилок: якщо збій стається в одній частині монолітного додатку, то це може призвести до падіння всього додатку; труднощі з тестуванням; труднощі з підтримкою; залежності і сторонні бібліотеки.

Проте і вона сама має деякі проблеми: складність управління: система містить в собі багато незалежних компонентів, що ускладнює управління в цілому. Наприклад, транзакційність певних операцій буде вимагати реалізацію не тільки на рівні бази даних, а ще і на рівні логічних компонентів системи; збільшені витрати на розробку: розробка і підтримка кількох мікросервісів може вимагати більше людських і обчислювальних ресурсів, оскільки кожен сервіс потребує окремого моніторингу

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

і управління; проблеми з безпекою: кожен мікросервіс є окремою точкою входу, що потенційно може створити більше вразливостей. Безпека та аутентифікація в розподіленій системі можуть бути більше складними, ніж в монолітній системі [5].

Далі розглянемо два основних архітектурних паттерни для побудови мікросервісної архітектури: Command Query Responsibility Segregation (далі CQRS) та Event Sourcing (далі ES) [6].

Отже, CQRS – це архітектурний патерн, який виник у світі програмування для покращення управління даними в складних системах. Основна ідея цього підходу полягає в розділенні дій, які змінюють стан системи (команди), і тих, що лише отримують дані (запити).

Можна представити логіку щодо формалізації CQRS на прикладі бібліотеки. Коли користувач бібліотеки бере книгу, він виконує дію, що змінює стан бібліотеки (забирає книгу). Це подібно до команди в CQRS. Коли ж користувач запитує, скільки книг є у бібліотеці, він тільки отримує інформацію, не змінюючи нічого для самої бібліотеки, що є аналогом запиту [6].

Розділяючи ці дві функції, CQRS дозволяє оптимізувати їх обробку. Наприклад, користувач може використовувати різні технології або структури даних для команд і запитів, щоб кожен з них працював максимально ефективно. Це особливо корисно в системах, де навантаження на читання і запис даних може суттєво відрізнятися [1].

Таким чином, CQRS допомагає розробникам створювати більш гнучкі, масштабовані і продуктивні системи, що дозволяє краще відповідати на потреби користувачів і бізнесу.

Однак, CQRS також має і свої недоліки. Основним недоліком є синхронізація даних між командами і запитами. При синхронізації даних будуть певні затримки, а у найгіршому випадку можна отримати неконсистентні дані. Також недоліком буде складність реалізації. Адже вам точно знадобиться мінімум два сховища даних, до того ж вам потрібно буде підтримувати дві моделі обробки даних, які можуть бути реалізовані на різних технологіях [6].

Event Sourcing – це архітектурний патерн, який змінює традиційний підхід до зберігання стану системи. Замість того, щоб зберігати лише поточний стан даних, Event Sourcing зберігає всі зміни, які відбулися в системі, у вигляді подій. Кожна подія є незмінною і відображає зміни, що відбулися [4].

Інший приклад можна навести на основі роботи зі щоденником. Кожен запис у щоденнику – це подія, яка фіксує конкретний момент у часі. Наприклад, користувач може записати, що «вчора він закінчив читати книгу» або «сьогодні він пішов на прогулянку». Кожен з цих записів є важливим, оскільки він відображає зміни у житті користувача. Так само і в Event Sourcing: кожна подія є відображенням зміни стану системи.

Цей підхід має кілька переваг. По-перше, зберігаючи всі події, ви можете легко відстежувати історію змін і відновлювати стан системи на будь-який момент часу. Це особливо корисно в ситуаціях, коли важливо знати, як і чому відбулися певні зміни [4]. По-друге, Event Sourcing дозволяє будувати системи, які можуть легко адаптуватися до змін. Наприклад, якщо вам потрібно додати нову функцію, ви можете просто обробити вже збережені події, не змінюючи структуру бази даних. По-третє, ES дає можливість відновити стан системи відносно будь-якої точки часу.

До недоліків такого підходу можна віднести складність зберігання, адже даних може бути дуже багато, що потребуватиме великих ресурсів для їх зберігання та обробки, та складність запитів при зчитуванні інформації, тому що, через необхідність обробки всіх подій, запити можуть бути складними. Хоча останнє більше залежить від конкретної бізнес логіки [4].

Отже, Event Sourcing – це потужний інструмент для розробників, який дозволяє створювати більш прозорі, гнучкі і надійні системи, що відповідають сучасним вимогам бізнесу та технологій.

Розглянемо більш глибше вплив архітектурних патернів на продуктивність і масштабованість. З точки зору продуктивності обрані архітектурні патерни мають значний вплив на продуктивність системи. CQRS дозволяє оптимізувати запити та команди, що може призвести до покращення швидкості відповіді. У традиційних системах, де команди і запити обробляються однаково, можуть виникати затримки, оскільки система повинна обробляти як запис, так і читання в одному контексті. З CQRS, ви можете оптимізувати кожен з цих аспектів окремо [1].

До того ж, CQRS дозволяє розділити обов'язки, де різні мікросервіси можуть обробляти різні частини запитів і команд. Дослідження показують, що системи, які реалізують CQRS у мікросервісах, можуть обробляти до 70% більше запитів на секунду в порівнянні з

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

монолітними архітектурами. Якщо взяти попередній приклад з торговцем цінними паперами і його менеджером, то ви можете розділити обробку цих моделей по різних мікросервісах, що дозволить вам масштабувати їх окремо один від одного за необхідності [5].

Оскільки всі зміни зберігаються у вигляді подій, система може швидко відновити свій стан на будь-якому етапі. Тобто, система може відтворити стан, просто обробивши послідовність подій. А оскільки події є незалежними, їх можна обробляти паралельно. Це дозволяє розподілити навантаження між кількома серверами або потоками, що підвищує загальну продуктивність системи. Дослідження показують, що системи, які реалізують Event Sourcing, можуть досягати збільшення продуктивності до 70% у порівнянні з традиційними підходами. Основні відмінності між Event Sourcing та CQRS наведені у таблиці 1 [6].

Таблиця 1

Основні відмінності між Event Sourcing та CQRS

Характеристика	Event Sourcing	CQRS
Принцип	Зберігає всі зміни стану у вигляді подій	Розділяє операції запису (команди) та читання (запити)
Зберігання даних	Зберігає всі події, які змінюють стан системи	Зберігає лише поточний стан системи
Відновлення стану	Можливе відновлення стану системи на будь-який момент часу	Не можливе. Маємо лише поточний стан
Складність імплементації	Вища (управління подіями, обробка подій)	Середня (управління двома окремими моделями)
Застосування	Підходить для систем, де важливо відслідковувати всю історію змін стану	Підходить для систем, де пріоритетною буде швидкість операцій читання та запису
Синхронізація даних	Вимагає механізмів для синхронізації даних при відновленні певного стану системи	Може мати проблеми з узгодженням даних між командами та запитами
Продуктивність	Може бути нижчою, через необхідність обробки всіх подій	Може бути вищою завдяки меншому набору даних при обробці запитів і команд

Джерело: таблиця створена автором на основі власного аналізу архітектурних паттернів на основі [3–4, 6]

У таблиці 1 наведено порівняння архітектурних патернів Event Sourcing та CQRS, висвітлюючи їх основні характеристики та відмінності. Event Sourcing фокусується на збереженні всіх змін стану системи у вигляді подій. Це дозволяє відновлювати систему до будь-якого моменту в часі, але вимагає складнішої імплементації через необхідність управління подіями. Цей підхід особливо корисний для систем, де важливо зберігати історію змін, хоча продуктивність може бути нижчою через обробку великої кількості подій.

У свою чергу, CQRS розділяє команди (записи) та запити (читання), зберігаючи лише поточний стан системи. Це забезпечує вищу продуктивність завдяки меншій кількості даних для обробки, проте може призвести до проблем з узгодженням даних між різними компонентами. CQRS підходить для систем, де швидкість операцій є критично важливою, але має середній рівень складності через управління двома окремими моделями.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Висновки. Таким чином, за результатами проведеного дослідження можемо стверджувати, що вибір архітектурного патерну для мікросервісів є критично важливим етапом у розробці масштабованих систем. Патерни CQRS і Event Sourcing мають свої переваги та недоліки, і їх вибір залежить від конкретних вимог проекту. Для досягнення оптимальної продуктивності та масштабованості розробникам слід ретельно аналізувати потреби системи та враховувати можливі компроміси.

Список використаних джерел

1. Vernon V. Implementing Domain-Driven Design – С. 138–146.
2. Newman S. Building Microservices
3. Buenosvinos C., Soronellas C., Akbary K. CQRS by Example
4. M. G. «CQRS and Event Sourcing: A Practical Guide». Journal of Software Engineering, 2018. – Vol. 12, № 2, P. 45–60.
5. Річардсон К. Мікросервісні патерни: як проектувати і будувати мікросервіси / К. Річардсон. – Київ: Видавництво «Діалектика», 2019. – 320 с.
6. «Understanding CQRS and Event Sourcing». Proceedings of the International Conference on Software Engineering and Computer Systems, 2020. – P. 45–50.
7. Martin R. Clean Code, 2009, – P. 35.

Роботу виконано під науковим керівництвом канд. екон. наук, старшого викладача
БУЛЬБИ С. А.

ПРОЗОРА КОМУНІКАЦІЯ ТА ГЕЙМИФІКАЦІЯ ОСВІТНЬОГО ПРОЦЕСУ

ШКОЛЬНИЙ М. М., 2 курс 3 група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті розглянуто основні проблеми освітніх програм заочного та дистанційного навчання, а також можливі шляхи їх удосконалення. Особливу увагу приділено використанню сучасних цифрових інструментів для підвищення ефективності навчального процесу та мотивації студентів.

The article examines the main challenges of part-time and distance learning programs, as well as possible ways to improve them. Special attention is paid to the use of modern digital tools to enhance the effectiveness of the learning process and student motivation.

Актуальність. Сучасне освітнє середовище характеризується стрімкими змінами технологій та зростанням популярності заочного і дистанційного навчання. Дистанційна освіта дійсна та актуальна, тому що здатна найбільш адекватно і гнучко реагувати на освітні потреби суспільства і забезпечувати реалізацію конституційного права на освіту кожного громадянина країни [1]. Особливо ж в українських реаліях війни, коли багато людей навіть маючи бажання, не можуть навчатись стаціонарно, питання організації ефективного навчального процесу набуває особливої актуальності. Розвиток цифрових технологій дозволяє впроваджувати новітні підходи в управлінні освітніми програмами, однак водночас з'являються нові виклики, пов'язані з організацією потоків даних, що доводиться тримати у пам'яті, мотивацією студентів та своєчасним контролем за виконанням завдань.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Інформація про вимоги, методичні матеріали, а головне – терміни виконання, часто виявляється фрагментованою та розпорошеною між різними каналами: електронними листами, університетськими порталами, хмарними сховищами, месенджерами чи навіть усними домовленостями. Така децентралізація та відсутність єдиного авторитетного джерела створюють інформаційний хаос, суттєво ускладнюють для студентів можливість відстежувати всі академічні зобов'язання, ефективно планувати свій час та уникати ненавмисного пропуску дедлайнів. Це не лише призводить до стресу, зниження мотивації та академічної неуспішності, але й підриває довіру до освітнього процесу, вимагаючи системного рішення на рівні університетської цифрової інфраструктури.

Проблематика, що стосується інтеграції систем сповіщень, гейміфікації навчання та оптимізації дедлайнів у навчальному процесі, особливо гостро відчувається у сфері заочного та дистанційного навчання, де традиційні методи організації процесу часто не відповідають сучасним вимогам. «Однією із найпрогресивніших методологій прогнозування майбутнього освіти сьогодні вважається форсайт (від англ. foresight – «погляд у майбутнє»). У європейському документі з регіонального форсайту дається таке його тлумачення: це процес активного пізнання майбутнього… це формування бачення середньострокової та довгострокової перспективи, ціль якого – прийняти актуальні рішення та мобілізувати спільні зусилля. Він об'єднує ключових учасників змін і різні джерела знань з метою розробки стратегічного бачення і розвитку здібностей до прогнозування майбутнього» [2]. Такі дослідження вказують на те, що можливості цифровізації освітнього простору зростатимуть, а непередбачувані події, які можна назвати «чорним лебедем», стимулюватимуть створення систем, здатних адаптуватися до змін та забезпечувати високий рівень комунікації між студентами та викладачами. Тому розробка вебдодатку для організації електронного журналу обліку з інтегрованими системами сповіщення, гейміфікації та контролю дедлайнів є надзвичайно важливою для підвищення якості освіти та ефективності процесу навчання в умовах сучасного університету.

Мета статті. Метою даної статті є розробка концепції вебдодатку на базі технологій ASP.NET в аспекті ефективних систем сповіщення та мотивації , який забезпечить якісну організацію та контроль навчального процесу для студентів та викладачів. Дослідження спрямоване на аналіз сучасних проблем заочного та дистанційного навчання, а також розробку гнучкого механізму інтегрованої в додаток системи, що включає розділення бажаного та крайнього строків здачі завдань, а також впровадження системи гейміфікації [3] через дашборд для студентів. Крім того, робота зосереджена на розробці обов'язкових інструментів сповіщення про наближення дедлайнів та обов'язкових налаштувань системи, які забезпечать відповідальність викладачів за заповнення необхідних даних. Дослідження має на меті представити комплексний підхід до оптимізації навчального процесу, який базується на сучасних технологіях та методах організації інформаційних потоків у вищих навчальних закладах.

Об'єкт дослідження. Стаття зосереджена на розробці вебдодатку, який завдяки інструментам сповіщень, дашборду та елементам геймифікації, має покращити успішність студентів університетів.

Предмет дослідження – вебдодаток, що покликаний покращити комунікацію студентів та викладачів університету.

Аналіз попередніх досліджень. Проблематика освітніх програм заочного та дистанційного навчання проявляється у багатьох аспектах, що безпосередньо впливають на ефективність навчального процесу. Однією з проблем, які важливо розуміти, є розрізненість інформаційних джерел, що ускладнює отримання необхідних даних як для студентів, так і для викладачів. Застарілі технології та відсутність єдиного інформаційного каналу спричиняють плутанину у розкладі, що призводить до пропусків важливих завдань та зниження мотивації до навчання. Дистанційне навчання, яке стрімко набуває популярності, висуває додаткові вимоги до систем комунікації, оскільки студенти часто стикаються з труднощами організації

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

власного часу та недостачею інтерактивних засобів зворотного зв'язку. Сучасні підходи до вирішення цих проблем базуються на інтеграції новітніх технологій, які дозволяють об'єднати різні канали комунікації в єдину систему, що забезпечує прозорість інформаційних потоків та оперативний контроль за виконанням завдань.

У контексті форсайт-досліджень майбутнього освітніх технологій особлива увага приділяється прогнозуванню змін у підходах до навчання та управління інформаційними потоками. Форсайт-аналіз дозволяє окреслити можливі сценарії розвитку цифровізації освіти, що включають як позитивні тенденції, так і потенційні ризики. З одного боку, впровадження інноваційних технологій може призвести до значного покращення якості навчального процесу, забезпечивши персоналізований підхід до кожного студента, підвищення ефективності взаємодії між викладачами та студентами, а також забезпечення швидкого обміну інформацією. З іншого боку, можливість виникнення непередбачуваних подій, які можна описати як «чорний лебідь», вимагає від розробників бути готовими до швидкого реагування на зміни та адаптації системи до нових умов.

Теорія «чорного лебедя» Нассіма Талеба [4] описує події, які є рідкісними, мають значний вплив і часто раціоналізуються заднім числом. У контексті освітніх технологій, як позитивні, так і негативні події «чорного лебедя» можуть кардинально змінити освітній ландшафт.

Позитивними «чорними лебедями» можна вважати несподівані прориви в галузі штучного інтелекту, що кардинально змінюють підходи до навчання, або раптову появу доступних і високоефективних технологій, які відкривають нові можливості для освіти. Також до цієї категорії входять несподівані зрушення у світових освітніх парадигмах, що працюють на користь України. Наприклад, розвиток хмарних обчислень та виникнення генеративного ШІ стали подіями, які, незважаючи на свою непередбачуваність, мали великі позитивні наслідки.

Негативними «чорними лебедями» можна назвати масштабні кібератаки на освітні установи, швидке старіння ключових технологій чи глобальні події, що суттєво порушують навчальний процес. Вторгнення Росії в Україну та пандемія COVID-19 – яскраві приклади того, як одна подія може змусити світ чи цілу країну перейти на онлайн-навчання, створюючи численні труднощі для традиційних освітніх систем. Такі непередбачувані події кидають виклик усталеним підходам і вимагають негайного реагування.

Нам потрібно створювати освітні системи, здатні витримувати несподівані виклики і швидко адаптуватися до нових умов. Гнучкість і швидкість реакції стають ключовими якостями тих систем, які не просто переживають кризи, а використовують їх як поштовх для розвитку. Українські університети повинні брати приклад із цих змін, розробляючи стратегії для безперервності навчання навіть у найскладніших ситуаціях, адже попри загрози негативних подій, позитивні зрушення також відкривають великі можливості.

Розбудова адаптивного потенціалу сучасних освітніх систем охоплює як фізичні навчальні простори, так і цифрові платформи управління навчанням. Адаптивні LMS-платформи дозволяють налаштовувати навчальний процес під конкретні потреби кожного студента, інтегрувати різні освітні інструменти та використовувати можливості штучного інтелекту для персоналізації навчання.

Прикладами сучасних адаптивних платформ можна вважати SC Training (раніше EdApp), Adaptemy, Knewton, Paradiso LMS, Open LMS, TalentLMS, Litmos, Docebo та Moodle Cloud [5]. Ці інноваційні рішення сприяють створенню освітніх середовищ, де кожен може знайти найкращий спосіб навчання.

Виклад основного матеріалу. Інвестиції в сучасні, адаптивні навчальні простори та LMS-платформи мають вирішальне значення для українських університетів, які прагнуть побудувати стійку систему освіти. Таке майбутньоорієнтоване мислення може дозволити українським освітнім установам не лише виживати в умовах змін, а й використати їх для свого розвитку.

Непередбачуваність майбутніх подій підкреслює необхідність створення гнучкої архітектури, яка дозволяє впроваджувати модифікації у режимі реального часу без шкоди для

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

стабільності системи. Такий підхід не тільки зменшує ризики, пов'язані з неузгодженістю даних, але й дозволяє університетам оперативно адаптуватися до змін ринку освітніх технологій.

Важливим інструментом покращення взаємодії студента з університетом може стати і концепція гейміфікації, що спрямована на підвищення мотивації студентів за рахунок використання елементів гри у навчальному процесі. Одним з прикладів ефективного підвищення мотивації через навчання з інструментами гри є додаток вивчення іноземних мов «Duolingo». Отримання балів за успішне проходження завдань мотивує повертатись до вивчення кожного дня, щоб не втратити всі переваги, які дає безперервний відрізок щоденної взаємодії з додатком.

Дашборд у навчальному додатку виступає як інструмент, що дозволяє студентам отримувати візуальний зворотний зв'язок про їх успішність, збирати бонуси та нагороди за своєчасне виконання завдань. При 100-бальній системі оцінювання, додавання кількох балів за дисциплінованість виглядає виправданим. Також дашборд відображає відсоток тих, хто здав роботу у встановлені терміни, не розкриваючи імена конкретних осіб. Такий підхід дозволяє не лише створити інтерактивне середовище, але й сприяє формуванню здорової конкуренції між студентами, стимулюючи їх до активнішої участі у навчальному процесі.

Впровадження елементів гейміфікації дозволяє створити додатковий рівень взаємодії, де кожен студент має можливість оцінити свої досягнення та отримати додаткове стимулювання для покращення своїх результатів. Здача робіт у рекомендований, а не у крайній термін, збільшує нагороду і, можливо, дозволяє отримувати певні статуси, які студент може зробити публічними у дашборді успішності групи. Це важливо з огляду на те, що заочне навчання часто позбавлене безпосереднього особистого контакту, що може впливати на рівень мотивації та залученості студентів до процесу навчання.

Інноваційним підходом у розробці даної системи є розподіл бажаного та крайнього строків здачі завдань. Викладач може встановлювати бажаний час завершення роботи над завданням, що дозволяє студенту організувати свій час так, щоб завчасно приступити до виконання роботи. За умови своєчасної здачі завдання до бажаного строку система нараховує додаткові бонуси, що слугує додатковим стимулом для студентів. Розділення цих двох показників дозволяє створити більш гнучку систему контролю, яка враховує як індивідуальні особливості студентів, так і специфіку виконання завдань. Такий підхід сприяє більш точному відображенню реального рівня організованості та відповідальності студентів, а також дозволяє викладачам оперативно аналізувати результати навчального процесу. Водночас можливість отримання додаткових бонусів за своєчасне виконання завдань стимулює студентів не тільки дотримуватись встановлених дедлайнів, а й прагнути до покращення власних результатів, що є важливим чинником для підвищення якості освіти.

Сповіщення про наближення дедлайнів виступають як обов'язковий інструмент, що забезпечує своєчасну інформованість студентів про майбутні завдання та контроль за їх виконанням. Система сповіщень розроблена таким чином, що за певний період до кінцевого строку здачі завдання студент отримує як внутрішні повідомлення через вебінтерфейс додатку, так і електронні листи. Це дозволяє мінімізувати ризики пропуску важливих дедлайнів, що часто виникають через неузгодженість інформаційних потоків або перевантаження інформацією. Завдяки автоматизації процесу сповіщення з'являється можливість оперативно коригувати ситуацію, що сприяє своєчасній здачі завдань та підтримує високий рівень організованості навчального процесу. Таке рішення є особливо важливим для студентів, які поєднують навчання із роботою або іншими зобов'язаннями, оскільки воно дозволяє їм уникнути непередбачених ситуацій та знизити рівень стресу.

Особливою вимогою до системи є налаштування, за якою викладачі зобов'язані заповнювати дані про дедлайни для кожного завдання. Викладач, як відповідальний за організацію навчального процесу, не може вимагати виконання завдання або виставляти

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

оцінку без попереднього встановлення кінцевого терміну здачі. Такий підхід дозволяє не лише підвищити рівень дисципліни серед студентів, але й забезпечує прозорість та відповідальність у процесі організації завдань. Викладачі мають можливість переглядати історію внесених змін та контролювати своєчасність заповнення інформації, що створює додатковий рівень безпеки та відповідальності за організацію освітнього процесу. Обов'язковість заповнення даних про дедлайни гарантує, що інформація про завдання завжди актуальна та доступна для всіх користувачів системи, що є критично важливим для організації навчального процесу.

З точки зору викладачів, така система також має полегшити їх роботу, оскільки викладач не отримує всі завдання від десятків студентів майже одночасно. Частина з них здасть завдання раніше (до рекомендованого, а не крайнього строку), що дозволяє викладачу розділити перевірку завдань на певні періоди.

У процесі розробки даної системи особливу увагу приділено інтеграції всіх згаданих компонентів у єдину, цілісну архітектуру, що дозволяє забезпечити як функціональну гнучкість, так і високий рівень надійності роботи додатку. Технологічна база, побудована на платформі ASP.NET, забезпечує стабільну роботу системи навіть при високих навантаженнях, що є важливим чинником для університетських проектів. Використання сучасних технологій дозволяє інтегрувати різноманітні модулі – від системи сповіщень та контролю дедлайнів до компонентів гейміфікації – у єдине інформаційне середовище, що сприяє оперативному обміну даними та високій продуктивності роботи додатку. Розробка інтегрованої системи контролю за виконанням завдань дозволяє викладачам оперативно реагувати на зміни в процесі навчання та приймати обґрунтовані рішення щодо коригування навчальних планів.

У підсумку, розробка комплексного вебдодатку, який інтегрує сучасні технології організації навчального процесу, дозволяє створити ефективну систему управління завданнями, що враховує особливості заочного та дистанційного навчання. Інноваційність підходу полягає у впровадженні гнучкого механізму розподілу бажаних і крайніх термінів здачі завдань, що стимулює студентів до активнішої участі у навчальному процесі, а також у використанні системи гейміфікації, яка створює додаткові можливості для мотивації та позитивного змагання серед студентів.

Ретельна інтеграція всіх компонентів системи – від обов'язкових даних про дедлайни, які заповнюються викладачами, до сповіщень про наближення термінів – забезпечує ефективність впровадження запропонованого рішення. Система дозволяє студентам оперативно отримувати інформацію про зміни в розкладі завдань на головній сторінці (дашборді), що знижує ризик пропуску важливих дедлайнів та сприяє підвищенню рівня організованості навчального процесу. Викладачі, у свою чергу, отримують можливість контролювати виконання завдань та своєчасно аналізувати результати, що дозволяє коригувати навчальний процес відповідно до потреб студентів. Такий підхід є важливим чинником для підвищення якості освіти та створення конкурентоспроможного освітнього середовища, що відповідає сучасним вимогам ринку.

Запропоноване рішення допомагає студентам краще організовувати свою навчальну діяльність, а викладачам – ефективніше взаємодіяти з ними. Завдяки вебдодатку всі необхідні дані будуть зібрані в одному місці, а нагадування про дедлайни та статус завдань допоможуть студентам не пропустити важливі терміни.

Впровадження системи з чітко визначеними бажаними та крайніми строками здачі робіт дозволяє зробити процес навчання передбачуванішим і менш стресовим. Це також сприяє рівномірному розподілу навантаження, оскільки дедлайни не накопичуються в кінці семестру.

Гейміфікація в освіті давно зарекомендувала себе як ефективний інструмент підвищення мотивації. Вебдодаток включатиме систему нагород і бонусів за своєчасне виконання завдань. Крім того, студенти зможуть анонімно бачити, який відсоток їхніх одногрупників уже здав роботу, що створить додатковий стимул бути в числі перших.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Використання ASP.NET дозволяє створити стабільний і продуктивний вебдодаток, який швидко обробляє великі обсяги даних і забезпечує зручну взаємодію між студентами та викладачами. Завдяки такому підходу освітній процес стає більш структурованим, прозорим і адаптованим до сучасних потреб студентів. Це не просто полегшує навчання, а й підвищує загальний рівень відповідальності та самоорганізації, що є надзвичайно важливим у сучасному світі.

Список використаних джерел

1. Деякі особливості розвитку дистанційної освіти Швеції. Електронний ресурс. http://www.irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?C21COM= 2&I21DBN=UJRN&P21DBN=UJRN&IMAGE_FILE_DOWNLOAD=1&Image_file_name=PDF/vcpiind_2013_55_26.pdf (Остання дата відгуку 03.04.2025)
2. Форсайт для розвитку освіти в середньостроковій і довгостроковій перспективі / О. М. Кузьменко // Освітологічний дискурс. – 2014. – № 1. – С. 108–116. – Режим доступу: http://nbuv.gov.ua/UJRN/osdys_2014_1_12
3. Technical and agricultural sciences in modern realities: problems, prospects and solutions: collective monograph / Hladyshev D., Brodskyi M., Lisnykh L. – etc. – International Science Group. – Boston : Primedia eLaunch, 2023.
4. Талеб Н. Чорний лебідь. Про неймовірне в реальному житті. – К.: Наш формат, 2016.
5. The Top 12 Adaptive Learning Platforms for 2025. Електронний ресурс. https://training.safetyculture.com/blog/adaptive-learning-platforms/ (Остання дата відгуку 13.04.2025)

Роботу виконано під науковим керівництвом PhD, доцента
ДЕСЯТКО А. М.

ВЕБОРІЄНТОВАНА СИСТЕМА МОНІТОРИНГУ ЯКОСТІ ОСВІТНЬОГО КОНТЕНТУ В УМОВАХ ЦИФРОВОЇ ТРАНСФОРМАЦІЇ ОСВІТИ

ШУМЕЙКО Р. Ю., 2 курс 3м група ФІТ ДТЕУ,
освітня програма «Інженерія програмного забезпечення»

У статті досліджуються теоретичні та практичні аспекти розробки веборієнтованої системи для моніторингу освітнього контенту в умовах цифрової трансформації освіти. Зважаючи на актуальні виклики, пов'язані зі збільшенням обсягів навчальних матеріалів, які розміщуються у відкритих або локальних електронних ресурсах, автором обґрунтовано необхідність створення спеціалізованих інформаційних систем, здатних здійснювати оперативну перевірку, аналіз і контроль за відповідністю освітнього контенту встановленим стандартам і критеріям якості. Веборієнтована архітектура системи передбачає мультиплатформенність, доступність з будь-якого пристрою, інтеграцію з популярними навчальними середовищами (LMS), підтримку ролей користувачів та інструменти візуалізації аналітичних звітів. В роботі описано функціональні можливості трьох основних підсистем: збору даних, аналітики та візуалізації.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

The article explores the theoretical and practical aspects of developing a web-based system for monitoring educational content in the context of digital transformation of education. Given the current challenges associated with the growing volume of learning materials hosted on open or local electronic resources, the author substantiates the need for specialized information systems capable of performing real-time verification, analysis, and control of the compliance of educational content with established standards and quality criteria. The web-oriented system architecture ensures multiplatform accessibility, availability from any device, integration with popular learning management systems (LMS), support for user roles, and tools for visualizing analytical reports. The paper outlines the functional capabilities of three core subsystems: data collection, analytics, and visualization.

Актуальність. У сучасних умовах розвитку цифрового суспільства освіта зазнає глибокої трансформації, зокрема в напряму впровадження інформаційно-комунікаційних технологій у навчальний процес. Значна частина навчального контенту переходить у цифрову форму, розміщується на онлайн-платформах, у хмарних сховищах та системах управління навчанням (LMS), що зумовлює нові вимоги до його організації, структуризації та контролю якості. Обсяг освітніх матеріалів постійно зростає, а разом із цим зростає і ризик дублювання, застарілості, невідповідності змісту державним освітнім стандартам або навчальним програмам.

Традиційні методи перевірки й оновлення контенту, які базуються на ручній роботі методичних комісій або окремих викладачів, вже не відповідають сучасним викликам, оскільки потребують значних ресурсів і часу. В умовах дистанційного або змішаного навчання це особливо актуально, адже своєчасний доступ студентів до якісних, перевірених та актуальних матеріалів є критичним чинником ефективності навчального процесу.

Крім того, освітні установи дедалі більше орієнтуються на відкритість, прозорість та відповідальність перед суспільством. Це зумовлює потребу у впровадженні систем, що дозволяють здійснювати об’єктивний моніторинг якості контенту, формувати звітність, відстежувати динаміку змін, виявляти прогалини й забезпечувати зворотній зв’язок між усіма учасниками освітнього процесу. Веборієнтовані рішення є найоптимальнішим шляхом досягнення цієї мети, оскільки забезпечують постійний доступ, мультиплатформеність, можливість інтеграції з іншими освітніми сервісами та гнучке масштабування.

Метою статті є розробка та аналіз веборієнтованої системи для моніторингу освітнього контенту, яка дозволяє здійснювати оцінювання актуальності, відповідності та доступності навчальних матеріалів у реальному часі.

Об’єктом дослідження є система контролю якості та структури освітнього контенту.

Предмет дослідження - функціональність і архітектура веборієнтованої системи моніторингу.

Аналіз попередніх досліджень. Дослідження в галузі цифровізації освіти, зокрема використання Learning Management Systems (LMS), були проведені як вітчизняними, так і закордонними дослідниками. Варто згадати праці М.І. Жалдака, С.О. Семерікова, а також міжнародні підходи, реалізовані в Moodle, Canvas та Google Classroom. Проте проблема саме систематизованого моніторингу залишає простір для нових рішень.

Виклад основного матеріалу. У сучасних освітніх системах спостерігається тенденція до цифровізації всіх складових навчального процесу – від контенту до методів оцінювання, комунікації, аналітики та управління. Впровадження Learning Management Systems (LMS) стало стандартом для більшості закладів освіти, однак більшість з них не мають вбудованих механізмів для системного моніторингу якості контенту. У відповідь на цю проблему пропонується концепція веборієнтованої системи для моніторингу освітнього контенту, яка функціонує як окремий сервіс або модуль, інтегрований із наявною LMS.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Веб-орієнтована система моніторингу освітнього контенту має модульну архітектуру, яка складається з трьох взаємопов'язаних функціональних компонентів. У її основі лежить концепція клієнт-серверної взаємодії, що забезпечує масштабованість, доступність та зручність підтримки. Центральним елементом є підсистема збору даних, яка відповідає за автоматичне отримання, обробку та зберігання метаданих освітнього контенту з різноманітних джерел – таких як LMS-платформи, хмарні файлові сховища та зовнішні навчальні репозиторії. Цей модуль здійснює періодичний або ініційований користувачем збір інформації про структуру навчальних курсів, формати матеріалів, час останнього оновлення, наявність опису, обсяг тощо.

Зібрана інформація надходить до підсистеми аналітики, яка реалізує набір алгоритмів для перевірки відповідності контенту критеріям якості: повноті, актуальності, унікальності та структурованості. Аналітика базується як на логічних правилах, так і на технологіях семантичного аналізу, включаючи використання методів штучного інтелекту для глибшого розуміння змісту матеріалів. Результати обробки передаються до підсистеми візуалізації.

Підсистема візуалізації результатів відповідає за побудову графіків, звітів і інтерактивних інтерфейсів для різних типів користувачів – адміністраторів, викладачів, методистів. Вона дозволяє оперативно переглядати статистику за курсами, виявляти слабкі місця, фільтрувати контент за рівнем відповідності та формувати звіти у зручному форматі. Вся взаємодія із системою здійснюється через веб-інтерфейс, що дозволяє забезпечити доступ із будь-якого пристрою та платформи. Узагальнена структура функціонування системи зображена на рис. 1.

Рис. 1. Архітектура веб-орієнтованої системи моніторингу освітнього контенту

Джерело: розроблено автором

Підсистема збору даних реалізує автоматизоване зчитування метаданих із LMS, репозиторіїв, хмарних сервісів (наприклад, Google Drive, OneDrive) за допомогою API або інструментів web scraping. Вона збирає дані про: типи файлів (PDF, DOCX, відео тощо); розмір, дату оновлення, автора; кількість переглядів; зв'язок із навчальними модулями. Ці дані зберігаються у централізованій базі даних системи, яка також підтримує історію змін.

Аналітична підсистема виконує ключову роль у забезпеченні якісного та змістовного контролю за освітнім контентом, що надходить із різних джерел до системи моніторингу. Вона працює на основі інтеграції методів логічного аналізу, індексації даних і технологій штучного інтелекту. Основною метою цієї підсистеми є визначення рівня відповідності кожного елементу контенту встановленим вимогам і критеріям якості.

Одним із базових критеріїв, який перевіряється системою, є актуальність контенту. Це означає, що система оцінює, наскільки нещодавно було оновлено файл, документ або інший навчальний ресурс. Наприклад, якщо певний матеріал не редагувався протягом останніх двох років, система маркує його як потенційно застарілий і рекомендує оновлення. Такий підхід

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

дозволяє підтримувати навчальні модулі в актуальному стані відповідно до змін у навчальних програмах, методичних рекомендаціях або нормативно-правовій базі.

Важливим напрямом перевірки є відповідність структури навчального модуля. Аналітична система аналізує, чи містить курс усі обов'язкові компоненти, такі як лекційні матеріали, тестові завдання, практичні або лабораторні роботи, додаткові ресурси. Якщо виявляється, що певний тип матеріалу відсутній, система виводить попередження та формує рекомендації щодо заповнення структурних прогалин. Це особливо важливо для забезпечення єдності стандартів викладання в межах одного освітнього закладу.

Крім того, система виконує перевірку на наявність дублікатів. За допомогою алгоритмів порівняння хеш-сум, fuzzy matching та семантичного аналізу відбувається ідентифікація однакових або дуже подібних за змістом файлів, які можуть бути повторно завантажені у різні модулі або розділи LMS. Наявність дублікатів не лише ускладнює управління контентом, а й може дезорієнтувати студентів, тож система фіксує такі випадки й надає звіт адміністратору або викладачу.

Окремий модуль підсистеми аналітики займається оцінкою повноти метаданих. Він перевіряє, чи заповнено основні атрибути кожного об'єкта: назву, опис, цілі навчання, мітки (теги), рівень складності тощо, що полегшує пошук і навігацію в межах курсу та дозволяє системі будувати аналітичні моделі та формувати індивідуальні траєкторії навчання.

Для реалізації всіх зазначених перевірок підсистема аналітики використовує комбінацію класичних методів логічного контролю (як-от перевірка заповненості полів, відповідності форматів), технологій побудови індексів (для оптимізації пошуку і порівняння), а також інструментів штучного інтелекту. Зокрема, для семантичного аналізу текстових матеріалів можуть використовуватися бібліотеки на кшталт spaCy, BERT, або інші моделі з платформи HuggingFace Transformers, що дає змогу визначати тематичну спрямованість контенту, його відповідність до дисципліни чи навчальної теми [2].

На основі результатів має формуватися таблиця з оцінкою кожного елементу контенту.

Таблиця 1

Оцінка відповідності контенту структурним вимогам

№	Назва файлу	Тип	Оцінка повноти	Дата оновлення	Рекомендації
1	Лекція_1.docx	DOCX	Повна	2023-09-14	Немає
2	Лекція_2.docx	DOCX	Часткова	2019-03-02	Потребує оновлення
3	Відео_тема_1.mp4	MP4	Без опису	2022-11-08	Додати метадані

Джерело: розроблено автором

В системі розглядається можливість встановлювати порогові значення, наприклад: мінімум 70% заповнених атрибутів – «задовільно», менше – «незадовільно».

Інтерфейс системи реалізується у вигляді веб-додатку з адаптивним дизайном, що забезпечує доступ із ПК, планшетів і смартфонів. Основний елемент інтерфейсу – дашборд адміністратора, що відображає загальну аналітику щодо контенту у вигляді графіків, списків і таблиць.

Інтерфейс містить модулі: огляд за навчальними курсами; перегляд проблемних елементів контенту; статистика оновлень; автоматичні сповіщення та рекомендації.

У системі реалізовано розмежування доступу: викладачі бачать лише власні курси, адміністратор має повний доступ.

Ефективність запропонованої веб-орієнтованої системи моніторингу освітнього контенту значно зростає у разі її інтеграції з існуючими платформами управління навчальним процесом, зокрема такими як Moodle – однією з найпоширеніших LMS у світі. Moodle підтримує відкриту архітектуру та має потужний API для зовнішньої взаємодії, що дозволяє впроваджувати сторонні аналітичні модулі без суттєвого втручання в основну логіку платформи [5].

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

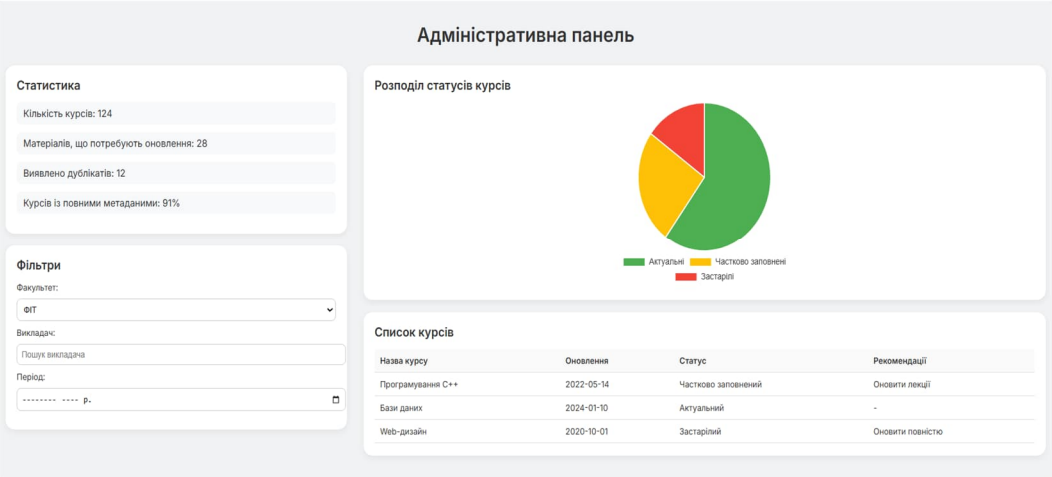


Рис. 2. Головний дашборд адміністратора системи

Джерело: розроблено автором

Прикладом практичного впровадження є реалізація автоматичного щотижневого аудиту курсів у межах факультету. Система сканує всі наявні навчальні курси через REST API Moodle, отримуючи повну інформацію про структуру кожного курсу, пов'язані ресурси (файли, тести, відео), а також дані про активність користувачів. На основі отриманих даних формується звіт, у якому, зокрема, вказується: які курси не містять повного набору обов'язкових компонентів (лекцій, тестів, практичних завдань); коли востаннє було оновлено навчальні матеріали; які елементи контенту дублюються в різних модулях або темах курсу; які курси мають неповні або відсутні метадані (наприклад, відсутній опис, не вказано цілі навчання, не додано категорію тощо).

Ці звіти дозволяють адміністрації освітнього закладу не лише швидко виявити системні проблеми, а й оцінювати динаміку покращення якості контенту протягом навчального періоду. Зокрема, завдяки інтерактивним графікам та статистичним зведенням, можна відстежити, як змінюється рівень заповненості курсів або актуальність матеріалів у порівнянні з попередніми тижнями чи семестрами.

З технічного боку, система реалізована на сучасному вебстеку, що забезпечує її гнучкість, продуктивність і масштабованість:

– Backend частина написана мовою Python з використанням легкого вебфреймворку Flask, що дозволяє швидко створювати RESTful-сервіси та забезпечує інтеграцію з базами даних, зовнішніми API та внутрішньою логікою обробки [6].

– В якості системи управління базами даних використовується SQLite (для локальних розгортань або прототипів) або PostgreSQL – у разі масштабних рішень із великою кількістю користувачів або паралельною обробкою запитів.

– Frontend побудований на базі React.js, що забезпечує динамічну побудову інтерфейсу, високу швидкодію при роботі з великими обсягами даних, а також адаптивність інтерфейсу до різних пристроїв. Для побудови графіків і діаграм використовується бібліотека Chart.js, яка дозволяє легко створювати інтерактивні кругові, стовпчикові та лінійні графіки [7].

– Для взаємодії з Moodle використовується Moodle REST API, який надає доступ до курсів, ресурсів, учасників, дій користувачів тощо. Завдяки цьому забезпечується синхронізація між платформами, а також можливість реалізації автоматичних перевірок у фоновому режимі (cron- або event-based сценарії) [8].

Перевагою такого підходу є його модульність і адаптивність: систему можна легко налаштувати під інші LMS (наприклад, Canvas, Google Classroom) або розширити для підтримки специфічних внутрішніх вимог ЗВО. У результаті це створює єдину цифрову

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

інфраструктуру для забезпечення якості освітнього контенту, яка відповідає сучасним викликам вищої освіти та цифрової трансформації навчального процесу [2].

Для виявлення дублікатів використовується алгоритм порівняння хеш-сум, а також fuzzy matching. Для семантичного аналізу – бібліотеки типу spaCy або transformers (HuggingFace) [3].

Висновки. У статті було розглянуто теоретичні та прикладні аспекти розробки веборієнтованої системи для моніторингу освітнього контенту, що відповідає сучасним викликам цифровізації освітнього середовища. Запропонована архітектура системи, що включає підсистеми збору даних, аналітики та візуалізації, дозволяє здійснювати глибокий аналіз якості навчального контенту, оцінювати його повноту, актуальність та відповідність структурам курсів.

Завдяки інтеграції з платформою Moodle за допомогою REST API забезпечується автоматизований доступ до навчальних курсів, з можливістю їхньої перевірки за формальними й змістовими критеріями. Практичне впровадження такої системи дозволяє щотижнево проводити аудит навчального контенту, формувати звітність для викладачів і адміністрації, а також своєчасно виявляти проблемні елементи контенту.

Технічна реалізація на базі стеку Flask–React–PostgreSQL забезпечує гнучкість і масштабованість рішення. Застосування бібліотек штучного інтелекту для семантичного аналізу контенту та інструментів аналітики (Chart.js) дозволяє створити ефективне середовище для прийняття управлінських рішень у сфері контролю якості освіти.

Список використаних джерел

1. Верба В.А. Інформаційне забезпечення управління розвитком компанії / В. А. Верба // Формування ринкової економіки. – 2009. – № 22. – С. 145–154.

2. Семеріков С.О. Інформаційно-комунікаційні технології у вищій освіті: досвід та перспективи / С.О. Семеріков // Вища освіта України. – 2020. – № 2. – С. 78–85.

3. Офіційний сайт Moodle: https://moodle.org

4. Canvas LMS Overview: https://www.instructure.com/canvas

5. Moodle Developer Docs – https://moodledev.io

6. Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. O'Reilly Media.

7. Chart.js Documentation – https://www.chartjs.org

8. Moodle Web Services API – https://moodledev.io/docs/apis/core/webservice

9. Семеріков С.О. Інформаційні системи управління в освіті: сучасні тренди та технології // Вища освіта України. – 2020. – № 2. – С. 78–85.

Роботу виконано під науковим керівництвом канд. фіз.-мат. наук, доцента
ГОРОХОВОЇ О. М.

Ця робота ліцензується відповідно до Creative Commons Attribution 4.0 International License.

Електронне наукове видання

ПРОГРАМУВАННЯ ТА ЗАХИСТ ІНФОРМАЦІЇ

**Збірник наукових статей студентів,
які здобувають освітній ступінь «магістр»
за спеціальностями 121 «Інженерія програмного
забезпечення» та 125 «Кібербезпека
та захист інформації»**

Частина 1

Видавець і виготовлювач
Державний торговельно-економічний університет
вул. Кіото, 19, м. Київ-156, Україна, 02156
Тел. (044) 513 74 18
Електронна пошта knute@knute.edu.ua
251Е-2025