Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

ПРОЄКТУВАННЯ E-COMMERCE-ПЛАТФОРМИ З ВІРТУАЛЬНИМ СТИЛІСТОМ ШТУЧНОГО ІНТЕЛЕКТУ ТА МОДУЛЕМ ЛОГІСТИЧНОЇ АВТОМАТИЗАЦІЇ
(тема)

Виконав:
здобувач 4 року навчання,
групи ІТІНФ-22-1
Шапошник М. Є.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник ст. викл. Кобилін І. О.
(посада, прізвище, ініціали)

Допускається до захисту

Завідувач кафедри інформатики _____ Кобилін О. А.
(підпис) (прізвище, ініціали)

2026 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту

Кафедра Інформатики

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Шапошнику Микиті Євгеновичу
(прізвище, ім'я, по батькові)

1. Тема роботи Проєктування e-commerce-платформи з віртуальним стилістом штучного інтелекту та модулем логістичної автоматизації

затверджена наказом університету від 26 травня 2026 року № 435Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 20 червня 2026 р.
3. Вихідні дані до роботи науково-методична та науково-технічна література у сфері електронної комерції та штучного інтелекту, аналітичні звіти провідних компаній, офіційна технічна документація, прикладні програмні інтерфейси (API) великої мовної моделі Gemini та логістичного оператора «Нова Пошта», а також стек базових вебтехнологій (PHP, MySQL, HTML5, CSS3, JavaScript).

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд основних методів автоматизації торговельних процесів.
2.Аналіз сучасного стану та методологія оцінки інтелектуальних помічників в електронній комерції.
3.Проєктування та програмна реалізація веб-системи електронної комерції.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)Актуальність та предметна область, мета та задачі роботи, математична модель оцінки ефективності, архітектура та технологічний стек, інтеграція LLM (Віртуальний стиліст), автоматизація генерації ТТН, гібридна система клієнтської підтримки, панель адміністратора (Управління системою), Користувацький досвід (UX/UI), Апробація результатів.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	13.04.2026	
2	Аналіз завдання, підбір літератури	14.04.26-17.04.26	
3	Аналіз літератури з проблеми, що вирішується	18.04.26-22.04.26	
4	Аналіз методів автоматизації торговельних та логістичних процесів	23.04.26-29.04.26	
5	Формування кроків вирішення проблеми	30.04.26-06.05.26	
6	Програмна реалізація	07.05.26-25.05.26	
7	Аналіз отриманих результатів	26.05.26-02.06.26	
8	Оформлення пояснювальної записки	03.06.26-09.06.26	
9	Перевірка на нормоконтроль	10.06.26-19.06.26	
10	Перевірка на плагіат	11.06.26-30.06.26	
11	Рецензування	20.06.26-30.06.26	
12	Підготовка презентації та доповіді	20.06.26-30.06.26	
13	Занесення роботи в електронний архів	30.06.26-09.07.26	
14	Попередній захист кваліфікаційної роботи	30.06.26-09.07.26	

Дата видачі завдання 13 квітня 2026 р.

Здобувач _____
(підпис)

Керівник роботи _____ ст. викл. Кобилін І. О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 117 с., 18 табл., 52 рис., 3 дод., 30 джерел.

АВТОМАТИЗАЦІЯ БІЗНЕС-ПРОЦЕСІВ, ЕЛЕКТРОННА КОМЕРЦІЯ, ІНТЕЛЕКТУАЛЬНІ ПОМІЧНИКИ, КОРИСТУВАЦЬКИЙ ДОСВІД (UX), ЛОГІСТИКА, МЕТРИКИ ЕФЕКТИВНОСТІ, ПЕРСОНАЛІЗАЦІЯ, ТТН (ТОВАРНО-ТРАНСПОРТНА НАКЛАДНА), ШТУЧНИЙ ІНТЕЛЕКТ.

Об’єктом роботи є процеси обслуговування та логістики в системах електронної комерції. Метою роботи є розроблення вебсистеми магазину одягу з ШІ-асистентом та модулем автоматичної генерації ТТН. Методи, що використані у роботі: системний аналіз, математичне моделювання (ROI, CLV, CR) та обробка природної мови (NLP). Практична цінність полягає в розробленому сайті як цілісній системі з інтелектуальним підбором товарів та автоматизованою генерацією накладних. Висновки. Обґрунтовано впровадження ШІ та вдосконалено модель оцінки ефективності через «коефіцієнт довіри» (K_{trust}). Рішення знижує витрати на обслуговування до 50% і мінімізує логістичні помилки. Область застосування. Результати можуть бути використані для цифрової трансформації та автоматизації логістики малого і середнього бізнесу в роздрібній торгівлі.

ARTIFICIAL INTELLIGENCE, BUSINESS PROCESS AUTOMATION, E-COMMERCE, INTELLIGENT ASSISTANTS, LOGISTICS, PERFORMANCE METRICS, PERSONALIZATION, TTN (TRANSPORT CARD), USER EXPERIENCE (UX).

The object of the work is customer service and logistics processes in e-commerce systems. The purpose of the work is the development of a clothing store web system with an AI assistant and an automated TTN generation module. Methods used in the work: system analysis, mathematical modeling (ROI, CLV, CR), and natural language processing (NLP). The practical value lies in the developed website featuring a smart assistant for product selection and automated TTN generation. Conclusions. The implementation of AI systems was justified, and the evaluation model was improved via a "confidence coefficient" (K_{trust}). The solution cuts service costs by up to 50% and minimizes logistics errors. Field of application. The results can be used for the digital transformation and logistics automation of small and medium-sized retail enterprises.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД – база даних

ТТН – товарно-транспортна накладна

ШІ – штучний інтелект

$\triangle C$ – показник економії операційних витрат

AOV – середня вартість замовлення

API – інтерфейс програмування застосунків

CLV – довічна цінність клієнта

CR – коефіцієнт конверсії

$K_{friction}$ – коефіцієнт технічного тертя

KPI – ключовий показник ефективності

K_{trust} – коефіцієнт довіри користувача до Штучного інтилекту

LLM – велика мовна модель

NLP – обробка природної мови

$P_{escalation}$ – ймовірність передачі запиту менеджеру

Precision – точність моделі

Recall – повнота моделі

ROI – рентабельність інвестицій

$T_{resolve}$ – час вирішення запиту

UX – користувацький досвід

ВСТУП

Динаміка сучасного ринку e-commerce останніх років буквально диктує бізнесу нові, досить жорсткі правила. Сьогодні замало просто виставити якісний товар на вітрину. Ключовим фактором стає те, як швидко магазин реагує на запит і наскільки персоналізований підхід він пропонує. Особливо гостро ці виклики відчуваються у ритейлі одягу. Тут покупець завжди вагається: чи підійде розмір, як ляже фасон, чи не відрізняється колір від фото. Коли менеджер кожен день підтримує такі діалоги, це дуже гальмує процес, створює черги та, призводить до великих втрати клієнтів.

Актуальність роботи полягає у пошуку того самого балансу між повною автоматизацією та «людяним» сервісом. Впровадження ШІ-помічників у системи онлайн-торгівлі – це реальна необхідність. Проте, як показує практика, складність тут не тільки в коді, а й у вмінні оцінити реальний прибуток для бізнесу. Звичайний чат-бот, який не розуміє контексту, часто лише дратує користувача, знижуючи конверсію та створюючи зайве «технічне тертя».

Ще одним слабким місцем у ланцюжку продажів залишається логістика. Коли кількість замовлень зростає, ручне оформлення ТТН починає «з'їдати» критично багато часу. Пряма інтеграція з API логістичних операторів – це єдиний робочий спосіб виключити помилки ручного введення та пришвидшити відправки. Тому розробка системи, яка поєднує розумний фронтенд (на базі LLM) і автоматизований логістичний бекенд – це вже не просто опція, а умова виживання для українського e-commerce. Щодо об'єкта роботи, то в цій роботі розглянуто операційні процеси e-commerce платформ в епоху цифровізації. Предметом зацікавленості стали конкретні програмні інструменти та методи, що дозволяють автоматизувати сервіс і логістику за допомогою ШІ. Ключова ціль проєкту - створити власну веб-платформу магазину одягу, яка закриває весь цикл угоди: від діалогу з AI-

асистентом до автоматичного створення ТТН. Для реалізації цієї мети довелося вирішити низку завдань:Оцінити ефективність існуючих рішень на ринку ШІ. Розробити та імплементувати математичну модель «коефіцієнта довіри» (K_{trust}). Написати програмний код для модулів генерації логістичних документів. Практична цінність отриманих результатів полягає у створенні робочого інструмента, який реально знижує витрати бізнесу і позитивно впливає на лояльність клієнтів.

1 ОГЛЯД ОСНОВНИХ МЕТОДІВ АВТОМАТИЗАЦІЇ ТОРГОВЕЛЬНИХ ПРОЦЕСІВ

1.1 Сучасний стан та еволюція інтелектуальних помічників у системах електронної комерції

Класичні моделі онлайн-комунікації вже давно вичерпали власний ресурс. Цифрова економіка не пробачає затримок. Асортиментна політика більше не гарантує ринкового лідерства; справжнім полем битви стала здатність платформи генерувати миттєвий цілодобовий зворотний зв’язок [1]. Еволюція алгоритмів розгорталася поступово. Якщо перші спроби автоматизації зводилися до чітко прописаних жорстких скриптів перефразування на кшталт програми ELIZA [2], то поточна архітектура еволюціонувала і вона кардинально інша. Впровадження великих мовних моделей (LLM) зламало представлення про примітивні відповіді. Нейромережі навчилися глибоко сканувати контекст запиту, філігранно імітуючи природну людську логіку [1, 2].

Окремие місце займає корпоративна стратегія провідних технологічних гігантів. Компанії калібрування Amazon та H&M давно перестали сприймати віртуальних помічників як екзотичну маркетингову опцію. Для них це фундаментальний інструмент виживання [1]. Алгоритм безпроблемно перетравлює тисячі паралельних звернень без втрати якості. Раніше такий масштаб неодмінно вимагав би покрокового та пропорційного зростання штату живих операторів [3]. Натомість штучний інтелект нібито стає одним цілим з екосистемою інтернет-магазину. Він аналізує цифрові дані та дозволяє покращити персоналізований клієнтський досвід [4].

Певні сумніви часто викликає надмірна фіксація на сухих технічних параметрах. Лабораторні показники нейромережі нічого не варті без прямої проекції на фінансові потоки підприємства. Глибинний аудит, у такій

ситуації, вимагає пошуку неочевидних кореляцій. Дослідники прицільно вивчають, як саме мілісекундна реакція асистента пропорційно підвищує зростання коефіцієнта конверсії (CR) [1]. Ізольовані алгоритми обробки природної мови (NLP) намертво зливаються з жорсткими методами бізнес-аналітики. Цей симбіоз народжує абсолютно автономні системи обслуговування, здатні миттєво адаптуватися до будь-яких ринкових коливань.

1.2 Аналіз методів автоматизації логістичного супроводу та генерації документації

Успішно продати товар – це лише половина справи. Справжнім випробуванням для інтернет-магазину незмінно стає оперативність доставки. Звична логістична процедура формування товарно-транспортних накладних (ТТН), роками залишався найбільш вразливим етапом. Тут людський фактор є вирішальним[1]. Менеджери багато раз на день змушені вручну переносити адреси, імена та специфікації вантажу. За умов жорсткого масштабування продажів така рутина неминуче провокує критичні часові лаги та фатальні помилки у відправленнях.

Певний рівень скептицизму щодо повної автоматизації цього етапу остаточно розбивається об можливості сучасних API-інтеграцій. Відточений зв’язок із серверами кур’єрських служб створює для обох сторін «безшовний» інформаційний коридор. Дані покупця миттєво переносятся із кошика прямо у готовий до друку документ. Пряма передача пакетів через протоколи JSON або XML назавжди позбавляє персонал від необхідності у дублюванні тексту. Паралельно застосовуються приховані алгоритми калькуляції. Вони самостійно витягують габарити товару за його ідентифікатором (SKU) та миттєво генерують фінальну вартість логістики. Більше того, подальший трекінг посилки також відбувається без втручання

оператора. Система автоматично підтягує свіжі статуси, радикально знімаючи навантаження з лінії клієнтської підтримки [1].

Економічний ефект від такої інженерної модернізації важко переоцінити. Автоматична генерація ТТН безжально зрізає щоденні операційні витрати та кардинально стискає цикл обробки замовлення та дозволяє зекономити гроші. Унаслідок такої автоматизації відбувається перенесення частини операційного навантаження безпосередньо на кінцевого користувача (перехід до моделі самообслуговування). Ця швидкість безпосередньо конвертується у зростання індексу задоволеності клієнтів (CSAT) та максимізацію їхньої довічної цінності (CLV) [1, 5].

1.3 Огляд та порівняльний аналіз існуючих систем електронної комерції

Аналіз ринку цифрової торгівлі вимагає детального вивчення наявних програмних рішень. Це дає підстави об’єктивно оцінити їхні функціональні можливості та виявити технологічні прогалини. Для порівняльного аналізу обрано три репрезентативні системи. Серед них: маркетплейс Rozetka, спеціалізований fashion-ритейлер Kasta та глобальний гравець ASOS. Окремий інтерес становить архітектура їхніх клієнтських сервісів. Більшість із них спирається на застарілі парадигми взаємодії.

Платформи Kasta та ASOS демонструють потужні алгоритми рекомендацій. Вони аналізують історію покупок. З іншого боку, дослідники часто ігнорують той факт, що такі системи використовують переважно колаборативну фільтрацію. Користувач отримує статичні добірки товарів. Генерація персоналізованих стилістичних порад у режимі реального часу відсутня. Підтримка клієнтів базується на класичних сценарних чат-ботах (rule-based). Такі алгоритми жорстко обмежені заздалегідь прописаними деревами діалогів. Вони не здатні розпізнавати складний контекст запиту.

Клієнтський досвід безпосередньо страждає через надмірну шаблонність відповідей.

Масштабні маркетплейси рівня Rozetka володіють розвиненою логістичною інфраструктурою. Впровадження API поштових операторів там виконано глибоко та надійно. Водночас варто зважати на високий поріг входу для розгортання подібних систем у малому та середньому бізнесі. Типові рішення на базі готових CMS (наприклад, базові конфігурації Shopify, WordPress або OpenCart) часто вимагають ручного перенесення даних для оформлення товарно-транспортних накладних (ТТН). Менеджери змушені дублювати інформацію з адміністративної панелі до кабінету логістичного оператора. Зростає ризик механічних помилок. Швидкість обробки замовлень суттєво падає.

Технічна реалізація користувацьких інтерфейсів також потребує перегляду. Багато платформ ігнорують потребу клієнта в безшовній зміні візуальних тем (Dark/Light Mode). Інші допускають критичну втрату даних кошика під час короткочасного розриву інтернет-з’єднання. Адаптивність часто зводиться до простого масштабування CSS-сітки, без оптимізації важких елементів об’єктної моделі документа (DOM) для мобільних пристроїв. Це створює надмірне навантаження на апаратну частину смартфона та провокує затримки рендерингу. Для подолання цієї проблеми необхідно використовувати сучасні методи оптимізації Web Core Vitals, зокрема CSS-властивість content-visibility та апаратне прискорення GPU, що дозволяють суттєво розвантажити основний потік браузера та забезпечити плавність інтерфейсу [6].

Окремо постає проблема незахищеності архітектури керування станом додатків (state management) у традиційній вебкомерції. Спроби розгорнути інтерактивні інтерфейси на базі застарілих монолітних рішень часто призводять до десинхронізації даних між клієнтською та серверною частинами. Користувачі регулярно стикаються із необхідністю повторного введення інформації під час збоїв, що критично знижує конверсію. Водночас

поширені програмні платформи рідко пропонують збалансоване поєднання локальних сховищ браузера та фонових асинхронних запитів. Це суттєво обмежує можливості для безшовного оновлення контенту, перетворюючи сесію на послідовність тривалих очікувань. Певні сумніви викликає також здатність базових CMS ізолювати процеси оформлення від мережевих затримок, що робить розробку оптимізованих рішень ще більш затребуваною.

Для структурування отриманих даних результати функціонального порівняння зведено у таблицю 1.1. Вона відображає наявність ключових технічних модулів у системах та розроблюваній платформі.

Таблиця 1.1 – Порівняльний аналіз функціональних можливостей систем електронної комерції

Критерій порівняння	ASOS	Kasta	Базові CMS (Shopify/OpenCart)
Генеративний ШІ (LLM) для консультацій	Відсутній	Відсутній	Відсутній (лише сторонні плагіни)
Гібридна маршрутизація чату (ШІ + Адміністратор)	Частково (сценарний бот + людина)	Частково (сценарний бот + людина)	Відсутня
Авоматична генерація ТТН у БД магазину	Прихована логіка	Реалізовано	Потребує ручного дублювання
Асинхронне збереження стану кошика	Реалізовано	Реалізовано	Залежить від конфігурації
Безшовна підтримка динамічних тем	Відсутня	Частково	Відсутня

Аналіз даних таблиці виявляє чітку галузеву тенденцію. Існуючі гіганти ринку мають потужний функціонал, проте їхня архітектура є закритою і неповороткою. Типові CMS-системи, навпаки, потребують встановлення десятків сторонніх модулів для покриття базових логістичних потреб. Жоден з проаналізованих аналогів не пропонує нативної інтеграції генеративних мовних моделей у ролі віртуального стиліста в комплексі з глибокою автоматизацією логістики з «коробки».

Певні сумніви викликає доцільність подальшого використання жорстких сценарних ботів для обслуговування клієнтів. Натомість впровадження LLM-моделей здатне кардинально змінити рівень персоналізації сервісу. Це дає підстави вважати розробку нової системи, яка алгоритмічно поєднує ШІ-асистента, оптимізований інтерфейс та автоматизований логістичний конвеєр, технічно обґрунтованим та актуальним завданням для поточної роботи.

1.4 Аналіз методів інтеграції штучного інтелекту в системи електронної комерції

Трансформація інструментів клієнтської підтримки пройшла стрімкий шлях від примітивних скриптів до когнітивних моделей. Тривалий час стандартом автоматизації виступали детерміновані системи або сценарні чат-боти. Їхні алгоритми будуються на жорстких деревах рішень. Попри поширену думку про надійність подібних архітектур, їхнє використання супроводжується значними обмеженнями. Найбільш очевидним дефектом є абсолютна негнучкість при обробці нестандартного синтаксису користувача. Система виявляється нездатною інтерпретувати запит через найменшу орфографічну помилку. Сучасна альтернатива полягає у використанні великих мовних моделей (Large Language Models). Вони оперують

імовірнісними методами та векторним представленням слів. Це дозволяє алгоритму глибоко аналізувати семантику тексту, самостійно розпізнавати синоніми та формувати природну відповідь без жорстко заданих правил.

Проте безконтрольне впровадження мовних моделей несе серйозні репутаційні ризики. Окремий інтерес становить феномен «штучних галюцинацій». Генеративний алгоритм може створити фактологічно хибне твердження з високим ступенем переконливості, вигадавши неіснуючі знижки або с потворивши характеристики товару. Для подолання цього явища застовають методологію сувого конструювання підказок (Strict Prompt Engineering). Поведінка моделі жорстко інкапсулюється всередині прихованого системного контексту. Алгоритм примусово наділяється роллю віртуального стиліста і блокується від обговорення сторонніх тем. Такий підхід перетворює хаотичну нейромережу на стабільний та безпечний комерційний інструмент.

Іншим архітектурним викликом є організація збереження стану діалогу. Базові вебпротоколи не мають вбудованої пам’яті (stateless). Відповідно, кожне нове звернення до ШІ розглядається сервером як ізольований запит. Для відновлення контексту розробляються спеціалізовані структури у базі даних. Йдеться про таблиці історії чату. Накопичені репліки динамічно серіалізуються та передаються моделі. Водночас виникає необхідність впровадження жорстких сесійних квот на використання API. Це дозволяє збалансувати фінансові витрати на інфраструктуру та захистити сервер від перевантажень, зберігаючи високу швидкість обробки клієнтських запитів.

1.5 Аналіз проблематики логістичних процесів та автоматизації ТТН

Обслуговування клієнтів не завершується на етапі оплати кошика. Класична модель обробки замовлень у малому та середньому бізнесі містить критичне «пляшкове горлечко» процес формування супровідної транспортної

документації. Зазвичай адміністратор змушений вручну копіювати дані клієнта з панелі управління магазином до особистого кабінету поштового оператора. При невеликих обсягах продажів такий підхід здається прийнятним. З іншого боку, дослідники часто ігнорують той факт, що при масштабуванні бізнесу рутинні операції експоненціально збільшують операційне навантаження на персонал. Швидкість обробки одного замовлення падає, що провокує загальні затримки у відправленнях.

Ключовою проблемою ручного перенесення даних є неминучий людський фактор. Помилки при введенні прізвища, номеру телефону або індексу відділення призводять до логістичних колапсів. Товар відправляється за хибною адресою. Бізнес зазнає прямих фінансових витрат через необхідність оплачувати повернення та переадресацію посилок. Цілком імовірно, що єдиним надійним механізмом усунення цієї проблеми є перенесення відповідальності за коректність даних безпосередньо на покупця під час оформлення замовлення. Це вимагає глибокої інтеграції прикладних інтерфейсів (API) логістичних компаній безпосередньо в архітектуру сторінки чекауту (checkout).

Впровадження зовнішніх логістичних API, зокрема сервісів Нової Пошти, породжує нові інженерні виклики. Пошук міста або відділення в реальному часі генерує величезну кількість HTTP-запитів до віддалених серверів при кожному натисканні клавіші користувачем. Без належної оптимізації це призводить до вичерпання лімітів API та зависання клієнтського інтерфейсу. Певні сумніви викликає доцільність прямої трансляції всіх користувацьких вводів на сервер пошти. Рішенням стає імплементація механізмів дебаунсингу (debouncing) на стороні фронтенду. Штучна затримка відправки запиту на кілька сотень мілісекунд дозволяє або об’єднати введені символи, або мінімізувати надлишковий мережевий трафік.

Фінальним етапом логістичної автоматизації є програмна генерація товарно-транспортної накладної (ТТН). Після вибору валідних

ідентифікаторів (UUID) міста та відділення, серверна частина магазину має сформувати стандартизований JSON-пакет і відправити його через захищене з'єднання до оператора. Такий конвеєр повністю виключає необхідність ручної роботи адміністратора. Готовий номер накладної автоматично записується в базу даних магазину та стає доступним для трекінгу. Це дає підстави розглядати розробку модуля автоматизованої логістики як один із найважливіших етапів створення сучасної e-commerce платформи, що безпосередньо впливає на її комерційну життєздатність.

1.6 Аналіз метрик оцінки ефективності та порівняльний аналіз підходів

1.6.1 Критерії оцінки результативності інтелектуальних систем

Впровадження технологій штучного інтелекту без попереднього економічного обґрунтування супроводжується високим ризиком фінансових збитків під час інтеграції нових функціональних рішень. Будь-яка технологічна модернізація вимагає чіткої економічної формалізації. Рентабельність інвестицій (ROI) у цій парадигмі є центральним об'єктом. Цей індикатор жорстко зіштовхує згенерований чистий прибуток із реальними витратами на розробку, безпомилково фіксуючи життєздатність проєкту [1].

Цілком імовірно, що найбільший прихований резерв криється в оптимізації щоденної рутини. Делегування типових клієнтських запитів алгоритмам зрізає операційні витрати ($\triangle C$). Нейромережа поглинає від 50% до 80% навантаження живого персоналу. Кадровий голод перестає бути проблемою. Бізнес отримує фундаментальну базу для масштабування, тоді як зарплатні фонди залишаються практично незмінними або навіть зменшуються.

Проте критерії оцінки ефективності проєкту не обмежуються виключно показниками скорочення фінансових витрат. Справжній вплив системи на шлях покупця найточніше відстежують поведінкові маркери. Віртуальний асистент безпосередньо втручається у коефіцієнт конверсії (CR), цілеспрямовано стимулює зростання середньої вартості замовлення (AOV) та закладає міцну основу для максимізації довічної цінності клієнта (CLV).

З іншої перспективи, суто інженерна верифікація традиційно спирається на суворий баланс точності (Precision) та повноти (Recall). Альтернативний погляд полягає в тому, що гола технічна статистика часто виявляється сліпою до психології споживача. Проф аудит вимагає обов'язкової інтеграції якісних факторів. Лише постійно відстежуючи суб'єктивний «коефіцієнт довіри» (K_{trust}) та випалюючи залишкове «технічне тертя» ($K_{friction}$), аналітик здатен побачити дійсно повноцінну картикну з точки зору комерційної ефективності.

1.6.2 Порівняння часових витрат при традиційному та автоматизованому обслуговуванні

Вже звичне архітурне рішення клієнтського сервісу містить фундаментальну вразливість. Живий персонал фізично не здатний обробляти весь об'єм запитів у години пікового навантаження, через що черги очікування формуються майже миттєво, суттєво знижуючи лояльність аудиторії. Хронометраж базової транзакції пошуку інформації та формування відповіді виявляє значний технологічний розрив між людиною та автоматизованими алгоритмами. Штучний інтелект мінімізує тривалість кожного мікроетапу обробки звернення. Фіксація текстового запиту та генерація релевантної відповіді тривають мілісекунди. Водночас людина-оператор зазнає значного когнітивного навантаження під час паралельного ведення кількох діалогів, що неминуче призводить до затримок та

десинхронізації даних. Інтелектуальні модулі повністю позбавлені цього обмеження завдяки здатності до асинхронного багатопотокового аналізу неструктурованих масивів без втрати точності. Зіставлення часових витрат на різних етапах для порівнюваних підходів наведено у таблиці 1.2.

Таблиця 1.2 – Порівняльний аналіз швидкості обробки запиту

Етап операції	**Традиційний метод (Людина)**	**Метод з ШІ-асистентом**	**Прискорення**
Прийом запиту	2-5 хв (очікування на лінії)	< 0.5 сек (миттєво)	х600
Аналіз контексту	3-5 хв (читання історії)	< 0.1 сек (авто-аналіз)	х1000+
Пошук рішення	5-10 хв (пошук у базі/CRM)	1-2 сек (векторний пошук)	х300
Генерація відповіді	3-5 хв (набір тексту)	1-3 сек (генерація)	х100
РАЗОМ	~15-25 хвилин	~3-5 секунд	~ у 400 разів

Нейромережа прискорює комунікацію в середньому у 400 разів. Певний скептициз може підказувати, що вважати такі гіпершвидкості надлишковими для звичайного ритейлу. На противагу цій думці спрацьовує

жорстка комерційна аналітика. Миттєва реакція платформи утворює найміцніший зв’язок із ймовірністю успішного закриття угоди. Розумний алгоритм за допомогою прискорення ритму обслуговування органічно блокує можливий перехід клієнта на сайти конкурентів.

1.6.3 Моделювання логіки взаємодії: від лінійних до інтелектуальних процесів

Класична логіка комунікації працює не досить ефективно. Кожне клієнтське звернення неминуче застрягає в системі очікування, поки живий оператор вручну шукає потрібну інформацію. Під час пікових навантажень такі часові лаги стають фатальними для бізнесу. Таку взаємодію структуруємо на рисунку 1.1.

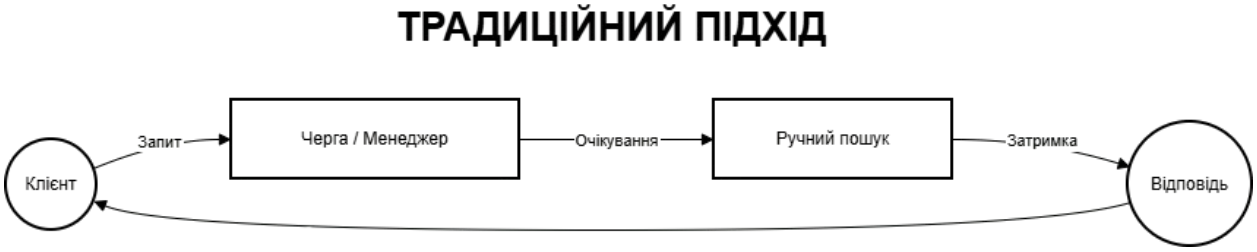

Рисунок 1.1 – Графічне пояснення підвищення автоматизації процесу в залежності від навантаження

Впровадження інтелектуального помічника кардинально змінює традиційну лінійну схему взаємодії. Завдяки алгоритмам обробки природної мови (NLP) тривалий процес послідовного аналізу трансформується у високошвидкісну транзакцію. Нейромережеві моделі інтерпретують безпосередні наміри покупця, забезпечуючи автоматичне формування релевантної відповіді з мінімальним часовим лагом. При цьому вебплатформа переходить на подійно-орієнтовану модель, за якої типові запити розподіляються в асинхронних потоках без блокування інтерфейсу

користувача. Таке рішення дозволяє паралельно обслуговувати велику кількість клієнтських сесій, повністю виключаючи людський фактор із контуру обробки інформації та усуваючи обмеження пропускної здатності системи як її головного «вузького місця». Графічне порівняння структурних переваг автоматизованого підходу наведено на рисунку 1.2.

Рисунок 1.2 – Графічне пояснення підвищення автоматизації процесу в залежності від навантаження

Наведена схема демонструє оптимізований цикл обробки звернень, у якому залучення інтелектуального асистента та бази знань скорочує час генерації фінальної відповіді до однієї секунди. Застосування такого автономного алгоритму забезпечує стабільну пропускну здатність платформи незалежно від пікових коливань користувацького навантаження.

1.7 Постановка задачі

Метою даної кваліфікаційної роботи є проєктування та розробка комплексної вебплатформи електронної комерції для магазину одягу, що включає інтегрованого інтелектуального помічника (віртуального стиліста) та модуль автоматизації логістичних процесів.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

– провести аналіз предметної області, виявити недоліки існуючих рішень та обґрунтувати вибір стека технологій;

– спроєктувати масштабовану архітектуру вебзастосунку та структуру реляційної бази даних, здатну витримувати пікові навантаження;

– інтегрувати модуль на базі алгоритмів обробки природної мови (NLP) для надання персоналізованих стилістичних консультацій у режимі реального часу;

– розробити логістичний модуль із прямою інтеграцією API поштових сервісів для автоматизації процесів валідації адрес та генерації товарно-транспортних накладних (ТТН);

– виконати експериментальну валідацію розробленої системи із застосуванням математичного апарату та комерційних метрик для підтвердження її працездатності.

Об’єктом роботи є процеси обслуговування клієнтів та логістичного супроводу в сучасних системах електронної комерції.

Предметом роботи є методи та програмні засоби автоматизації взаємодії з користувачами й механізми формування супровідної документації на основі інтелектуальних алгоритмів.

2 АНАЛІЗ СУЧАСНОГО СТАНУ ТА МЕТОДОЛОГІЯ ОЦІНКИ ІНТЕЛЕКТУАЛЬНИХ ПОМІЧНИКІВ В ЕЛЕКТРОННІЙ КОМЕРЦІЇ

2.1 Технологічна трансформація клієнтського сервісу: від лінійних до інтелектуальних моделей

Функціонування в умовах цифрової економіки вимагає високої швидкості обробки даних та мінімізації часових затримок. Сьогоденна електронна комерція має конкурувати не тільки асортиментом, скільки еталонною якістю сервісу, та набором функцій. Швидкість реакції та глибока персоналізація безпосередньо визначають життєздатність проєкту. Проте звичні моделі підтримки, жорстко прив’язані до живого персоналу, вже вичерпали свій ресурс. Залежність від операторів створює великі втрати часу та не менш колосальні фінансові витрати. Масштабувати такий штат під раптові напливи покупців дуже складно та майже неможливо. Забезпечити ж миттєвий зворотний зв’язок посеред ночі – майже нереально.

Еволюція інтелектуальних систем, у сучасному світі, має солідне теоретичне підґрунтя. Від фундаментальних концепцій А. Тюрінга[7] щодо природи машинного мислення до перших наївних спроб хоча б зімітувати діалог в алгоритмі ELIZA Дж. Вайзенбаума [2] минула ціла епоха.

Сучасний етап розвитку характеризується кардинальним зсувом парадигми. Сьогодні базовою технологією для глибокого розуміння природної мови в таких системах виступають архітектури трансформерів, здатні коректно обробляти складні лінгвістичні конструкції [8]. Завдяки цьому примітивні чат-скрипти остаточно еволюціонували в автономні стратегічні інструменти. Аналітики McKinsey [5] та Gartner [9] одностайно фіксують переломний етап у цій трансформації. Перехід до великих мовних моделей (LLM) фактично ліквідував головне «вузьке місце» клієнтського

сервісу, оскільки ручний контроль перестав бути обов’язковим для кожної транзакції.

Традиційна архітектура центрів обробки викликів характеризується наявністю системних обмежень. Запит проходить сувору лінійну ієрархію, застрягаючи в системі черг та очікуючи на повільне опрацювання людиною. Наслідки є цілком передбачувані. У періоди пікових навантажень накопичуються критичні часові лаги та затримки, які невідворотно руйнують лояльність покупця, або простими словами, дратують. Впровадження алгоритмів обробки природної мови (NLP) агресивно ламає цю послідовну схему. Впровадивши такий алгортим можна зробити стрімкий крок у майбутнє. Обмежена комунікація перетворюється на паралельну та миттєву. ШІ-система здатна аналізувати тисячі звернень одночасно. Сервіс масштабується за долі секунди. Автоматизація процесів дозволяє уникнути необхідності оперативного збільшення кількості персоналу. Формується безперервний цикл взаємодії, який надійно фіксує увагу користувача та посилює конкурентні позиції платформи.

Практичні метрики повністю підтверджують доцільність таких змін. Делегування рутинних процесів алгоритмам успішно закриває від 60% до 80% діалогів абсолютно автономно [1, 9]. Показовим індикатором ефективності цього підходу виступає досвід інтеграції рішень Anthropic у платформу Intercom [10]. Там нейромережі здатні самостійно закривати 86% клієнтських проблем.

2.2 Формалізація комплексної математичної моделі оцінки ефективності

Оцінка ефективності впровадження систем штучного інтелекту потребує обов’язкового переходу від теоретичних описів до кількісного аналізу. Для об’єктивного порівняння різних корпоративних стратегій та

мінімізації похибок у даних застосовано комплексну методологічну базу. Даний підхід дозволяє систематизувати результати та оцінити ефективність інтегрованих алгоритмів за трьома основними критеріями: фінансовими показниками, технічною стабільністю та метриками клієнтського досвіду [1].

Інтегральний показник впливу ШІ (AI_{impact})визначається як функція від трьох параметрів, як показано у формулі (2.1):

$$AI_{impact} = f(T_{tech}, E_{econ}, C_{client}), \quad (2.1)$$

де – T_{tech} – технічний параметр (цілодобова доступність, стабільність роботи, швидкість обробки запитів);

E_{econ} – економічний параметр (зменшення витрат на персонал, підвищення ефективності обслуговування);

C_{client} – клієнтський параметр (покращення комфорту, довіри та лояльності клієнтів).

2.2.1 Фінансові та операційні метрики оцінки

Впровадження інноваційних технологій вимагає обов'язкового проведення фінансового оцінювання. Рентабельність інвестицій (ROI) виступає тут безапеляційним суддею життєздатності ШІ-проєкту. Математично цей індикатор кристалізується у базове співвідношення чистого прибутку до загальної вартості впровадження, що відображає формула (2.2):

$$ROI_{AI} = \frac{R_{AI} - C_{AI}}{C_{AI}} \times 100\%. \quad (2.2)$$

У цій моделі змінна R_{AI} акумулює весь додатковий дохід, згенерований

виключно завдяки алгоритмам – від раптового сплеску продажів до успішного утримання лідів. Натомість C_{AI} фундаментально фіксує сукупні витрати. Сюди лягають не лише чеки за розробку, але й кошти на системну інтеграцію та подальшу підтримку інфраструктури.

Певна вузькість мислення часто змушує менеджерів зводити ефективність інтелектуальних систем виключно до прямих грошових потоків. Професійний аудит вимагає чітко розмежовувати два виміри окупності. Перший – це «жорсткий ROI» (tangible gains). Він рахує абсолютно конкретні метрики, такі як радикальне урізання бюджетів чи зростання конверсії.

На противагу йому працює «м'який ROI» (qualitative benefits). Оцифрувати рівень лояльності аудиторії чи психологічний комфорт покупця, дісно, вкрай складно. Проте саме ці глибинні якісні зсуви формують довгостроковий фундамент довіри до бренду, захищаючи його від відтоку клієнтів.

Найбільш відчутним наслідком автоматизації майже завжди стає показник економії коштів ($\triangle C$). Цей фінансовий маркер може безпомилково відстежувати пряме скорочення щоденних операційних витрат підприємства після відмови від ручного обслуговування. Математична логіка обчислення цієї збереженої дельти зведена у формулу (2.3):

$$\triangle C = C_{before} - C_{after} = Savigns_{AI}, \tag{2.3}$$

де C_{before} – витрати до впровадження ШІ;

C_{after} – витрати після впровадження ШІ.

Ця метрика є особливо актуальною для оцінки ШІ в автоматизації клієнтського сервісу та оптимізації внутрішніх процесів, де технологія замінює або доповнює ручну працю, наприклад, у кол-центрах чи юридичних департаментах.

2.2.2 Поведінкові та клієнтські метрики

Алгоритми рекомендацій виконують прогнозування споживчих потреб на основі зібраних даних. Вони ак стимулюють додаткові (cross-selling) та преміальні (upselling) продажі. Оцінити реальну фінансову успішність такої поведінки штучного інтелекту допомагає показник середньої вартості замовлення (AOV). Математична модель цього індикатора зводимо до відповідного розрахунку.

Для забезпечення максимальної релевантності персоналізованих пропозицій віртуального стиліста, система повинна одночасно аналізувати значну кількість різноманітних параметрів користувача. Для вирішення таких завдань прийняття надійних рішень на основі безлічі ефективних факторів успішно застосовуються технології нечіткої логіки [11].

$$AOV = \frac{Загальний\_дохід}{Кількість\_замовлень}. \quad (2.4)$$

Проте фокусуватися виключно на спорадичних чеках – буде цілком хибним кроком. Професійна аналітика неминуче зміщується на довгострокову перспективу. Тут на авансцені з’являється довічна цінність клієнта (Customer Lifetime Value, CLV). Така метрика відкидає миттєву вигоду, даючи приорітет прогнозуванню сумарного доходу від покупця за весь період співпраці з платформою. Її архітектура відбита у формулі (2.5):

$$CLV = AOV \times F \times T, \quad (2.5)$$

де AOV – представляє із себе середній чек;

F – фіксує частоту транзакцій;

T – визначає загальну тривалість лояльності клієнта.

У наукових та маркетингових дослідженнях відсутня єдина думка щодо визначення універсального індикатора ефективності. Більшість фахівців сходиться на думці, що фундаментальним маркером залишається коефіцієнт конверсії (Conversion Rate, CR). Він дозволяє відсіяти «випадкових перехожих». Алгоритм вираховує точний відсоток відвідувачів, які успішно завершили цільову дію(покупку), що відображено у формулі (2.6):

$$CR(\frac{Кількість\_конверсій}{Загальна\_кількість\_відвідувачів})\times 100\%. \quad (2.6)$$

Коефіцієнт конверсії є найбільш репрезентативним показником впливу штучного інтелекту на процес продажів. Вона чітко фіксує кожен крок: від влучної персоналізації пропозицій на головній сторінці до філігранної роботи чат-бота, який рятує покинутий кошик безпосередньо на етапі оплати.

З іншого боку, специфіка підпискових бізнес-моделей диктує абсолютно власні правила. Для великих учасників ринку, зокрема стрімінгових платформ – Netflix[3], агресивно залучає нових лідів часто відходить на другий план. Критичною метрикою виживання тут стає коефіцієнт відтоку клієнтів (Churn Rate). Він миттєво сигналізує про відсоток аудиторії, яка розірвала відносини з компанією, в повному обсязі демонструючи реальну ціну технічних помилок чи нерелевантних рекомендацій системи. Для бізнес-моделей, заснованих на підписці, як ми вже зазначили коефіцієнт відтоку клієнтів (ChurnRate) є дуже важливим.

$$ChurnRate = (\frac{Клієнти\_втрачені\_за\_період}{Клієнт\_на\_початок\_періоду})\times 100\%. \quad (2.7)$$

2.2.3 Технічні метрики продуктивності та фактор довіри

Технічний аудит класифікаційних алгоритмів неможливий без двох базових індикаторів: точності (Precision) та повноти (Recall).

$$\Pr esicsion = \frac{TP}{TP + FP}. \tag{2.8}$$

Формула повноти(2.9).

$$\mathrm{Re}\, call = \frac{TP}{TP + FN}. \tag{2.9}$$

Їхня математична архітектура спирається на три фундаментальні змінні. Система фіксує істинні спрацювання (True Positive, TP), відстежує хибні (False Positive, FP) та виявляє небезпечні пропуски (False Negative, FN).

Технічна реалізація вимагає знаходження компромісу між конфліктуючими метриками. Максимальна точність агресивно відсікає хибні спрацювання. Вона є гарантом того, що алгоритм випадково не заблокує легітимну транзакцію покупця. Водночас абсолютна повнота діє взагалі інакше. Вона фокусується на тому, щоб жодна шахрайська операція не прослизнула крізь побудовані фільтри, навіть ціною періодичних помилкових блокувань. Знайти цей баланс – критично важливе завдання для розробників систем управління ризиками.

Звернемо увагу на пошук математичної «золотої середини» між цими крайнощами. Щоб звести дві конфліктуючі метрики до єдиного знаменника допомагає F1-міра (F1 Score). Вона діє як гармонійне середнє. Цей показник ідеально збалансовує обидва типи помилок, що формалізовано у рівнянні (2.10).

$$F1 = 2 \times \frac{\Pr ecision \times \mathrm{Re} call}{\Pr ecision \times \mathrm{Re} call}, \quad (2.10)$$

де саме F1-міра дозволяє аналітикам адекватно порівнювати різні моделі у ситуаціях, коли ціна хибної тривоги та пропущеної загрози є однаково високою.

Фундаментальною основою для побудови відмовостійких інтелектуальних систем є використання адаптивних моделей ідентифікації. Застосування робастних алгоритмів для обробки нестаціонарних систем дозволяє ефективно виявляти приховані закономірності в потоках даних [12, 13].

Водночас виключно кількісні алгоритмічні показники не враховують психологічних факторів поведінки споживача. Зіставлення результатів платформи «до» та «після» розгортання інтелектуального помічника підсвічує очевидну прогалину. Технічно навіть сама ідеальна нейромережа може не генерувати продажі, якщо відвідувач немає до неї довіри. Це дає підстави інтегрувати в наявні моделі новий вимір – Коефіцієнт довіри користувачів (K_{trust}). Цей специфічний маркер неможливо витягнути з логів сервера. Він формується виключно шляхом глибинного аналізу якісних опитувань аудиторії.

Відтак, класична оцінка конверсії зазнає глибокої модифікації. Аналітичний рушій починає враховувати не лише сухий факт здійснення цільової дії, але й рівень психологічного комфорту, який безпосередньо спровокував конкретну транзакцію. Оновлена архітектура оцінювання закріплена у формулі (2.11).

$$CR_{effective} = CR_{base} \times (1 + K_{trust} - K_{friction}), \quad (2.11)$$

де – $K_{friction}$ – коефіцієнт «тертя», що виникає через помилки ШІ або затримки у відповідях.

Часова ефективність вирішення запиту описується рівнянням(2.12):

$$T_{resolve} = T_{ai\_session} + (P_{escalation} \times T_{human\_wait}), \quad (2.12)$$

де $P_{escalation}$ – ймовірність ескалації (частка звернень);

T_{human_wait} – середній час очікування живого оператора.

Метою впровадження ШІ є мінімізація обох $P_{escalation}$ та $T_{ai\_session}$ що веде до загального зниження $T_{resolve}$ і, як наслідок, до скорочення операційних витрат та підвищення задоволеності клієнтів.

2.3 Класифікація сфер впливу та функціональних переваг ШІ-асистентів

Окрім кількісних метрик, обов'язковим етапом роботи є структурування якісного впливу інтелектуальних помічників на бізнес-процеси електронної комерції. Функціонування асистентів не обмежується виключно прямою текстовою комунікацією; цей інструмент інтегрується у весь цикл взаємодії з користувачем – від початкового етапу пошуку товару до етапу формування стійкої лояльності.

Комплексна оцінка ефективності вимагає врахування багатовимірності сучасних торговельних платформ. Традиційна воронка продажів трансформується під впливом алгоритмів штучного інтелекту, здійснюючи перехід від реактивної моделі обслуговування до проактивної персоналізованої взаємодії. Впровадження інтелектуальних систем глибоко модифікує архітектуру клієнтського шляху (Customer Journey). Зокрема, алгоритмізація базових процесів дозволяє оптимізувати навігаційні сценарії, автоматично адаптувати контентну видачу під специфічні запити споживача та забезпечити безперервність циклу технічної підтримки. Впровадження

адаптивних матричних нейро-фаззі мереж для кластеризації багатовимірних часових рядів дозволяє значно підвищити точність сегментації користувачів. Використання самоорганізовних мереж забезпечує динамічну адаптацію системи до мінливих уподобань клієнтів [14, 15]. У цьому контексті якісні зміни в операційній діяльності потребують чіткої класифікації за функціональними напрямками. Диференціація сфер впливу створює необхідну методологічну базу для подальшого обґрунтування рентабельності інтегрованих програмних модулів.

На основі проведеного аналізу [1] систематизовано ключові напрямки впливу ШІ-асистентів, їхню функціональну сутність та результативні переваги, які наведено у Таблиці 2.1. Зазначена класифікація формує базовий критеріальний апарат для подальшого проєктування системи. Вона дозволяє чітко формалізувати технічні вимоги до інтегрованих програмних модулів.

Таблиця 2.1 – Сфери впливу, суть впливу та ключові переваги інтелектуальних помічників

№	Сфера впливу	Суть впливу	Ключові переваги
1	**2**	**3**	**4**
2.1	Оптимізація пошуку товарів і послуг	Асистент швидко знаходить потрібний товар, розуміє не лише ключові слова, а й контекст запиту.	Економія часу користувача, зручний діалоговий пошук.
2.2	Персоналізовані рекомендації	Пропонує товари з урахуванням попередніх покупок, переглядів, інтересів.	Підвищення ймовірності повторних покупок, зростання середнього чеку.
2.3	Автоматизація обслуговування клієнтів	Миттєво відповідає на типові запитання, зменшує навантаження.	Швидке обслуговування 24/7, зниження витрат на персонал.

Продовження таблиці 2.1

1	2	3	4
2.4	Підвищення доступності сервісу	Працює цілодобово, незалежно від часових поясів та людського фактора.	Безперервний сервіс, відсутність втрати клієнтів через затримки у відповідях.
2.5	Зниження витрат бізнесу	Замінює частину операторів, автоматизуючи до 60-80% звернень.	Економія ресурсів компанії при збереженні якості обслуговування.
2.6	Формування лояльності та позитивного досвіду	Створює відчуття турботи завдяки персоналізації, швидкості та зручності взаємодії.	Зростання довіри, та лояльності, формування користувацького досвіду.

2.4 Порівняльна верифікація стратегій впровадження інтелектуальних систем

Довіряти виняткова математичним індикаторам часто приховує глибинні зміни всередині бізнесу. Голі цифри не здатні описати якісну еволюцію платформи. Тому виникає гостра потреба структурувати невидимий, але потужний вплив штучного інтелекту на архітектуру електронної комерції. Доволі поширеною помилкою серед аналітиків є зведення ролі нейромереж до банальної імітації живого оператора в текстовому вікні. Проте реальний масштаб інтеграції значно ширший.

Щоб об’єктивно виміряти цю багатовимірну еволюцію, розкладення впливу ШІ на три фундаментальні критерії. Перший вектор – технологічна та операційна ефективність (T_{tech}). Цей показник фіксує безпрецедентну швидкість опрацювання запитів та абсолютну автономність системи, яка

надійно розвантажує персонал від рутини. Другий напрямок цілком концентрований на якості клієнтського досвіду (C_{client}). Інтелектуальний помічник значно підвищує зручність навігації, паралельно забезпечуючи покращену персоналізацію для кожної товарної пропозиції. Фінальний критерій відображає пряму економічну та стратегічну результативність (E_{econ}). Інтегровані алгоритми гарантують безперебійну масштабованість магазину під час пікових розпродажів та генерують стабільне зростання фінансових показників.

Аналіз саме цих точок дотику чітко пояснює, як абстрактні алгоритми врешті-решт перетворюються на реальні фінансові дивіденди. Проте, спираючись на результати попередніх експериментальних зрізів [1] та функціональну сутність базових критеріїв ($T_{tech}, E_{econ}, C_{client}$), вкрай небезпечно ігнорувати психологію покупця. Гола статистика фіксує лише сухий факт транзакції. Натомість глибинний аналіз корпоративних стратегій підсвічує значно цінніший актив – рівень сформованої довіри до інтелектуального сервісу (K_{trust}). Цей суб'єктивний маркер виступає абсолютним фундаментом для максимізації довічної цінності клієнта (CLV), утримуючи його від переходу до конкурентів. Узагальнені результати такого дескриптивного аналізу, базовані на реальному досвіді провідних організацій, структуровано в Таблиці 2.2.

Таблиця 2.2 – Результати аналізу компаній та організацій (за матеріалами [1])

Компанія (Сектор)	**Тип стратегії та Рішення**	**Ключові технології**	**Результат та Рівень довіри**
1	**2**	**3**	**4**
Amazon (E-commerce)	Гібридна: голосова комерція Alexa, рекомендації	NLP, глибоке навчання	-40% часу на пошук. *Довіра:* Висока

Продовження таблиці 2.2

1	2	3	4
H&M (Ритейл)	Гібридна: прогнозування попиту, рекомендації	ML (XGBoost), GAN	+15% до середнього чека. Довіра: Середня
Sephora (Косметика)	Клієнтська: чат-бот для персональних консультацій	NLP, діалоговий ШІ	+11% до конверсії. Довіра: Висока
Netflix (Розваги)	Гібридна: персоналізована фільтрація контенту	Колаборативна фільтрація	>80% переглядів через ШІ. *Довіра:* Критична
Intercom (CRM)	Операційна: ШІ-асистент Fin для відповідей	LLM, NLP	-50% навантаження на підтримку. *Довіра:* Висока
Bank of America	Гібридна: віртуальний фінансовий асистент Erica	NLP, аналітика даних	>10млн активних користувачів. *Довіра:* Дуже висока
Естонія/Дія (Держсектор)	Гібридна: чат-боти, інтелектуальна навігація	NLP, діалоговий ШІ	Скорочення бюрократії. *Довіра:* Критична

Аналіз показників Таблиці 2.2 підтверджує, що впровадження моделей глибокої індивідуалізації клієнтського шляху (на прикладі та Sephora) корелює з високими показниками довіри користувачів. Водночас прагматичний підхід до операційної інтеграції, реалізований на платформі

Intercom, підтверджує економічну ефективність систем штучного інтелекту за рахунок цілеспрямованої оптимізації операційних витрат підприємства.

Проте не можна зупинятися виключно на описових категоріях. Повноцінна валідація побудованої математичної моделі вимагає проектування цих абстрактних результатів на конкретні фінансові та поведінкові індикатори. Масив теоретичних формул (2.1–2.12) має пройти суворе випробування реальними кейсами. Тільки так аналітик здатен відстежити неочевидний та спецефічний взаємозв'язок. Стає кристально ясно, як саме ювелірна технічна точність нейромережі провокує лавиноподібне зростання комерційних показників. Цей фінальний метричний зріз комплексно агреговано у Таблиці 2.3.

Таблиця 2.3 - Результати метричного аналізу компаній (верифікація моделі [1])

Компанія	**Ціль впровадження ШІ**	**Економічний ефект (ROI, $\triangle C$, AOV)**	**Клієнтський ефект (CR, CLV) та Результат**
1	**2**	**3**	**4**
Amazon	Персоналізація в масштабі	Оптимізація логістики, зростання AOV	Підвищення CR та CLV. Результат: -40% часу на пошук
H&M	Оптимізація запасів	AOV +15%, зменшення надлишків	Помірний вплив на CLV. Результат: Зростання чека на 15%
Sephora	Масштабування консультацій	Зниження навантаження на персонал	Підвищення лояльності. Результат: Зростання конверсії (CR) на 11%
Netflix	Максимізація утримання	*Не є пріоритетом*	Зниження відтоку (Churn Rate). Результат: >80% переглядів через ШІ

Продовження таблиці 2.3

1	2	3	4
Intercom	Ефективність сапорту	*Економія до 50% часу ($\triangle C$)*	Покращення сервісу. Результат: -50% навантаження
Duolingo	Ефективність навчання	*Не застосовується*	Утримання аудиторії. Результат: Баланс залученості через власні метрики
Coursera	Результативність курсів	*Не застосовується*	Підвищення мотивації. Результат: +25-35% до завершення курсів
Дія	Оптимізація послуг	Економія адміністративних витрат	*Не застосовується*.Результат: Кардинальне прискорення послуг

Синтезування емпіричних даних із таблиць 2.2 та 2.3 остаточно розвіює теоретичні сумніви щодо ефективності роботи нейромереж. На цьому кроці ми фіксуємо абсолютно пряму залежність між агресивною автоматизацією та різким обвалом операційних витрат $\triangle C$. Корпоративні кейси Intercom та Bank of America яскраво ілюструють це найкраще. Навантаження на живий персонал там фактично скоротилося вдвічі. Цей прецедент бездоганно верифікує математичну логіку, закладену в основу формули (2.3).

Проте оцінка виключно за фінансовими критеріями є недостатньою для комплексного аналізу. Глибший зріз демонструє силу емпатії в цифровому середовищі. Впровадження діалогових інтерфейсів, здатних філігранно імітувати живу людську експертність (як це реалізовано у Sephora), миттєво каталізує зростання суб'єктивного «Коефіцієнта довіри» (K_{trust}). У аслідок такого психологічного комфорту виявляється цілком прибудкови та матеріальним. Конверсія платформи автоматично стрибає на 11%.

Кардинально інша картина вимальовується у площині підписковихбізнес-моделей. Для стрімінгових гігантів калібрування Netflix суха технічна точність (Precision/Recall) перетворюється на питання лише базового виживання. Алгоритм не має права на помилку. Лише еталонна релевантність контентних рекомендацій здатна зупинити критичний для системи відтік розчарованої аудиторії (Churn Rate).

Загалом проблематика коректного оцінювання штучного інтелекту вже давно вийшла за межі лабораторних експериментів. Сучасна економіка буквально знеживлена дефіцитом кваліфікованих кадрів, вимагаючи екстремально швидкої трансформації процесів. За таких жорстких умов ігнорування глибокої аналітики гарантовано призводить до спалювання багатомільйонних інвестицій. Саме тому запропонована методологія пропонує бізнесу міцну точку опори. Органічне поєднання психологічного «Коефіцієнта довіри» з об’єктивним інтегральним показником ефективності AI_{impact} створює надійний запобіжник від управлінських помилок, перетворюючи хаотичні інновації на чітко прогнозований результат.

2.5 Проєктування архітектури відмовостійкого інтеграційного шлюзу (API Gateway)

Пряма взаємодія базових модулів електронної комерції із зовнішніми сервісами приховує критичну архітектурну вразливість. Зазвичай розробники імплементують синхронні запити безпосередньо з клієнтської частини або ядра бекенду. Такий підхід здається зручним на етапі прототипування. З іншого боку, дослідники часто ігнорують той факт, що сторонні сервери не гарантують стабільного часу відгуку. Раптова затримка на стороні хмарної нейромережі або логістичного провайдера миттєво блокує головний потік виконання вебзастосунку. Інтерфейс завмирає. Користувач не може

завершити оформлення замовлення. Це призводить до катастрофічного падіння конверсії та втрати довіри до платформи.

Для усунення цієї проблеми було спроєктовано виділений інтеграційний шлюз (API Gateway). Це ізольований програмний прошарок на серверній стороні. Його головне завдання полягає у централізованому управлінні всіма вихідними та вхідними мережевими запитами. Жоден клієнтський скрипт не має прямого доступу до Gemini API чи сервісів нової пошти. Усі звернення маршрутизуються виключно через цей шлюз. Запропонована архітектура дозволяє інкапсулювати складну логіку авторизації, управління токенами, обробки помилок та форматування даних в єдиному модулі. Системне ядро платформи залишається чистим і повністю незалежним від раптових змін у специфікаціях зовнішніх провайдерів.

Окремий інтерес становить механізм забезпечення відмовостійкості (Fault Tolerance). Інтеграційний шлюз реалізує патерн Запобіжник (Circuit Breaker) та сувору політику таймаутів. Якщо API нової пошти не відповідає протягом визначеного часу (наприклад, 2000 мілісекунд), шлюз примусово перериває з’єднання. Система не чекає нескінченно. Замість критичної помилки сервер повертає на фронтенд коректний резервний відгук (Fallback). Для логістичного модуля це означає активацію резервного поля для ручного введення адреси. Для інтелектуального стиліста – генерацію стандартного повідомлення про тимчасову перевантаженість консультанта. Певні сумніви щодо складності розробки такого механізму нівелюються тим, що користувач завжди залишається у робочому інтерфейсі та може продовжити покупки незалежно від стану сторонніх серверів.

Водночас шлюз автоматично керує фоновими спробами повторного підключення (Retry Policy). У разі тимчасової відмови Gemini API (наприклад, через помилку HTTP 429 Too Many Requests), інтеграційний модуль не транслює цю помилку клієнту миттєво. Замість цього ініціюється повторний запит із застосуванням алгоритму експоненційної затримки. Кожна наступна спроба виконується через збільшений проміжок часу, щоб

не створювати надлишкового навантаження на мережу. Лише після вичерпання ліміту спроб сесія маркується як неуспішна.

Іншим критичним аспектом є двостороння валідація інформаційних пакетів. Сторонні сервіси можуть періодично змінювати структуру JSON-відповідей. Якщо бекенд магазину спробує розпарсити невалідний або пошкоджений об’єкт, виникне фатальна помилка виконання коду. Інтеграційний шлюз діє як санітарний бар’єр. Кожна відповідь від мовної моделі або поштового оператора спочатку проходить сувору типізацію. Алгоритм перевіряє наявність обов’язкових ключів та формат типів даних (рядки, масиви, цілі числа). Лише після успішної структурної валідації інформація передається до модуля клієнтського відображення. Некоректні відповіді жорстко блокуються, а їхній технічний дамп записується в ізольований системний журнал для подальшого аналізу адміністратором.

Графічна репрезентація архітектури інтеграційного шлюзу та логіки обробки зовнішніх запитів подана на рисунку 2.1.

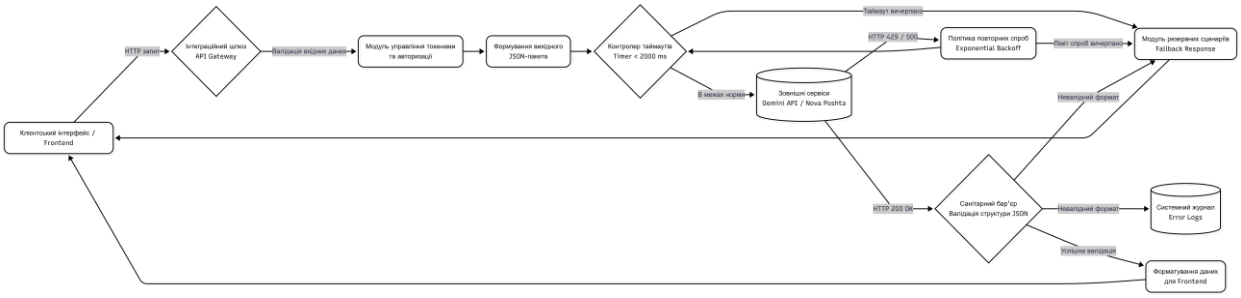

Рисунок 2.1 – Структурна схема відмовостійкого інтеграційного шлюзу (API Gateway)

2.6 Синтез результатів аналітичної роботи як передумова практичної реалізації

Здійснено повну верифікацію запропонованої математичної моделі. Аналітика світових гігантів калібрування Amazon, Netflix та Intercom не

залишає простору для дискусій. Інтеграція нейромереж в електронну комерцію давно перестала бути подихом моди, перетворившись на серьозний економічний крок. Цілком закономірно, що найвищу результативність генерують виключно гібридні екосистеми. Вони категорично відмовляються обирати між комфортом покупця та оптимізацією внутрішніх процесів компанії, наздоганяючи двох зайців.

На основі аналізу статистики визначено чіткі архітектурні вимоги до проєктованої системи. Фронтенд-взаємодія не має права залишатися пасивною. Вона вимагає розгортання розумного NLP-асистента. Алгоритм повинен стимулювати зростання середнього чека (AOV) та загальної конверсії (CR), паралельно зводячи до абсолютного мінімуму будь-яке користувацьке «тертя» ($K_{friction}$). Втім, навіть найзручніший інтерфейс зазнає краху без надійного тилу. Відтак, бекенд-інфраструктура бере на себе завдання з радикального урізання операційних витрат ($\triangle C$).Програмне ядро самостійно генеруватиме логістичну документацію, зокрема товарно-транспортні накладні (ТТН). Вплив людського фактора під час обробки даних мінімізується, оскільки відповідальність за коректність введеної інформації переноситься на користувача. Це дозволяє суттєво скоротити тривалість циклу обробки та відправлення замовлень.

Для об’єктивного моніторингу роботи системи передбачено інтеграцію модуля безперервного метричного контролю. Платформа агрегуватиме дані для регулярного перерахунку інтегрального показника AI_{impact} та тонкої валідації психологічного «Коефіцієнта довіри» (K_{trust}).

Викладений матеріал завершує теоретичне обґрунтування проблеми. Сформована методологія перестає бути просто набором формул і перетворюється на інженерний каркас. Оптимізація користувацького інтерфейсу на основі методів адаптивної кластеризації об’єктів дозволяє створювати персоналізовані сценарії взаємодії, що безпосередньо впливає на показники конверсії платформи. Попереду на етапі прямого проєктування та написання коду для вебплатформи інтернет-магазину одягу.

3 ПРОЄКТУВАННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ-СИСТЕМИ ЕЛЕКТРОННОЇ КОМЕРЦІЇ

3.1 Обґрунтування вибору середовища програмної реалізації та інструментальних засобів

Ефективність функціонування розроблюваної системи електронної комерції визначається архітектурною цілісністю та обґрунтованим вибором технологічного стека. Проєктування платформи вимагало синхронізації інтелектуального ШІ-асистента та модуля логістичної автоматизації в межах єдиного контуру. Для реалізації клієнтської частини (Frontend) обрано класичний стек HTML5, CSS3 та чистий JavaScript (Vanilla JS). Використання базових технологій без громіздких сторонніх фреймворків дозволяє мінімізувати час первинного рендерингу сторінок, знизити об’єм завантажуваних скриптів та забезпечити максимальну гнучкість при маніпуляціях з DOM-деревом під час асинхронної взаємодії з сервером.

Серверна частина (Backend) побудована на мові програмування PHP [16]. Вибір зумовлений її високою швидкістю обробки HTTP-запитів, нативною підтримкою сесій та простотою інтеграції із зовнішніми REST API через вбудовану бібліотеку cURL. Як система управління базами даних (СУБД) застосовано реляційну модель MySQL [17]. Вона забезпечує високу швидкість виконання складних транзакційних запитів, підтримує механізми зовнішніх ключів (Foreign Keys) для збереження цілісності даних та гарантує надійне ізолювання персональної інформації користувачів, номенклатури товарів та логістичних логів.

3.1.1 Інтеграція інтелектуального модуля та логістичного сервісу

Аріхтектуру даних треба жорстко впорядковувати. Спроби поєднати товарну номенклатуру, профілі клієнтів, масиви ШІ-діалогів та логістику в єдину структуру закономірно привели до реляційної моделі – СУБД MySQL. Авжеж, існує цілком невтомна спокуса застосувати трендові NoSQL-рішення для збереження, наприклад, неструктурованих текстів чатів. Проте надійність та структурованість надійність оброкбки замовлень та цілісність каталогу товарів тут важить набагато більше. Класичний SQL-підхід завдяки архітектурі зовнішніх ключів гарантує залізобетонну цілісність інформації. Це не просто технічна забаганка. Сувора зв’язність таблиць виступає критичною передумовою для точного розрахунку CLV та решти аналітичних метрик, які ми препарували у другому розділі. Жодних втрат даних чи аномалій при багатошарових запитах.

Програмний комплекс OSPanel (Open Server) елегантно закриває всі базові потреби інжинірингу. Замість виснажливого ручного конфігурування розрізнених вузлів, система отримує монолітний WAMP-стек «з коробки» (Windows, Apache, MySQL, PHP). Розробник створює код в умовах, які практично тотожно відтворюють конфігурацію бойового хостингу. Відповідно, левова частка інфраструктурних багів відловлюється ще до релізу. Швидкість ітерацій зростає кратно. Водночас рутинне адміністрування таблиць легко перекладається на плечі вбудованих інструментів формату phpMyAdmin. Мінімум часу на налаштування оточення, максимум – на імплементацію складної бізнес-логіки.

3.2 Проєктування інтерфейсу та функціональної структури веб-платформи

Візуальний каркас платформи слугує єдиній жорсткій меті. Йдеться про тотальне знищення коефіцієнта тертя($K_{friction}$), з механікою якого ми вже похнайомилися. Користувацький досвід (UX) просто не має права перетворюватися на заплутаний лабіринт. Хибним є коли дизайнери за інерцією перевантажують стартові екрани масивами контенту, сподіваючись затримати увагу клієнта. Проте стратегія кардинально інакша. Архітектура головної сторінки свідомо форсує конверсію. Усе вирішує синергія двох чинників: миттєвого доступу до флагманських пропозицій та постійної готовності інтелектуального помічника підхопити діалог. Кожна зайва секунда пошуку буквально вбиває продаж. Тому інтерфейс (UI) прокладає для покупця максимально пряму та очевидну траєкторію без жодних візуальних перешкод.

3.2.1 Структура та динамічні модулі головної сторінки

Архітектура головної сторінки реалізована за модульним принципом. Формування інтерфейсу відбувається динамічно, без використання статичного (жорстко закодованого) контенту. Серверні PHP-скрипти на льоту витягують дані з MySQL, формуючи гнучкий інтерфейс. Відверто кажучи, ручне оновлення категорій давно стало моветоном в індустрії. Тому інтелектуальний Header відтворює дворівневі випадаючі меню «Каталог» та «Бренди» на основі поточного даних з бази. Це повністю автоматизує навігацію. Одразу під нею відвідувача зустрічає масивна Hero-секція з чітким та відвертим закликом до дії (CTA) та приваблює промо-банером WELCOME10, що відразу формує бажання клієнта хоча б зайти до каталогу.

Впровадження прямого стимулювання є ефективним інструментом для максимізації показника конверсії.

Далі фокус уваги зміщується на інноваційну складову платформи. Виділений блок цілеспрямовано «продає» послуги персонального стиліста, провокуючи клієнта на запуск діалогу через Gemini API. Паралельно працює інша психологія збуту. Інтерактивна JS-карусель «Луки на кожен день» візуалізує не поодинокі речі, а цілісні готові образи. Такий технічний маневр гарантовано тягне вгору середній чек (AOV). Природні сумніви щодо «холодного» трафіку жорстко та цілеспрямовано нейтралізує система відгуків. Свіжі модеровані коментарі із зірковим рейтингом буквально цементують коефіцієнт довіри. Фінальним штрихом виступає блок «Про нас», де філософія бренду зливається з лінками на Instagram і Telegram для омніканальної підтримки.

Загальну архітектуру та візуальне представлення модуля наочно демонструє рисунок А.1, який надано у додатку А.

3.2.2 Система ідентифікації та управління станом користувача

Безперервність користувацького досвіду жорстко контролюється механізмом PHP-сесій. Це фундаментальний запобіжник від втрати контексту. Перевірка системних прапорців $isLoggedIn та $isAdmin не просто фіксує факт авторизації, а на ще і миттєво перезбирає інтерфейс під конкретний рівень доступу. Існує типова для e-commerce проблема раптового обнулення кошика під час навігації. Ми повністю усунули цю вразливість.

JavaScript регулярно пакує фронтенд-дані у JSON та примусово синхронізує їх із серверною сесією. Обрані товари гарантовано залишаються на місці навіть після «хард-рефрешу» сторінки. Не менш критичним є рішення свідомої відмова від тотальної автоматизації сервісу. Як тільки логіниться адміністратор, система одразу розгортає приховані елементи

панелі, параметри меню та панелі керування чатами. Так активується гібридна модель підтримки: живий оператор та діалог з нейромережею.

3.3 Проєктування системи ідентифікації та управління профілем користувача

Модуль авторизації виконує функцію базового контролю доступу до вебплатформи. Процес автентифікації спирається на перевірку стану PHP-сесій. За умови ідентифікації активної змінної isset($_SESSION[’user’]) здійснюється миттєве перенаправлення до особистого кабінету. Для оптимізації взаємодії форму входу (рис. А.2) зведено до двох критичних параметрів, що суттєво зменшує час відгуку системи. Водночас збереження користувацьких налаштувань, зокрема візуальних тем оформлення, делеговано клієнтському JavaScript-модулю theme.js. Він асинхронно застосовує обрані параметри ще до моменту фактичної авторизації на сервері.

На відміну від модуля входу, підсистема реєстрації (рис. А.3) орієнтована на збір розширеного масиву персональних даних. Окрім стандартних параметрів, обов’язковим елементом є поле для введення повного імені (ПІБ). Попри ймовірний ризик зниження первинної конверсії, такий підхід жорстко продиктований архітектурними вимогами. Зібрані дані зберігаються у профілі та автоматично експортуються у форми фінального оформлення замовлення. Це є фундаментальною технічною передумовою для безшовної інтеграції з API логістичного сервісу «Нова Пошта» та генерування товарно-транспортних накладних (ТТН). Усі описані процеси спираються на єдину реляційну модель. Детальну структуру таблиці користувачів наведено у додатку Б (табл. Б.1). Саме ця схема даних гарантує стабільність авторизаційних сесій та надійний транзит логістичних параметрів.

Керування обліковим записом реалізовано через модуль налаштувань (рис. А.10), який використовує багаторівневу систему обробки інформації для мінімізації серверного навантаження. Оновлені текстові параметри (ім’я чи логін) миттєво синхронізуються з масивом $_SESSION, що виключає необхідність надлишкових звернень до бази даних під час рендерингу інтерфейсу. Зміна пароля супроводжується криптографічною перевіркою поточного ключа та валідацією мінімальної довжини. Робота з мультимедіа організована за принципом суворої файлової гігієни. Завантаження нового аватара автоматично ініціює скрипт видалення попереднього зображення, запобігаючи засміченню дискового простору. Серверна валідація пропускає виключно MIME-типи JPG, PNG та WEBP розміром до 5 мегабайт, тоді як об’єкт FileReader на стороні JavaScript миттєво генерує прев’ю без перезавантаження сторінки.

3.4 Проєктування та функціональні можливості особистих кабінетів

Особистий кабінет є центральним вузлом взаємодії користувача з платформою. Він забезпечує персоналізацію клієнтського досвіду та надає доступ до ключових сервісів системи: від історії замовлень до інтелектуального асистента.

3.5 Профіль користувача (User Profile)

Персональний кабінет (profile.php) функціонує як головний операційний хаб покупця. Його бекенд-архітектура спирається на жорстке розмежування прав доступу через механізм PHP-сесій. Варто системі не знайти валідний маркер $_SESSION[’user’], як бекенд миттєво відхиляє запит, примусово повертаючи відвідувача на екран логіну. Дехто з інженерів

відкладає перевірку ролей на пізніші етапи виконання коду. Тут підхід кардинально інакший. Маршрутизатор аналізує системний прапорець isadmin безпосередньо під час ініціалізації сторінки. Якщо база даних валідні дані, скрипт автоматично перенаправляє користувача у закриту площину profileadmin.php.

Адміністратор отримує розширені важелі впливу без жодних проміжних екранів чи додаткових кліків.

Візуальна концепція кабінету свідомо ігнорує зайвий інформаційний шум. Центром тяжіння виступає лаконічна картка ідентифікації (profile-card). Аватар, повне ім’я та електронна пошта не просто констатують факт реєстрації. Вони формують підсвідому психологічну прив’язку до платформи. Завдяки гнучкому каркасу profile-container та правилам зі стилістичної таблиці, інтерфейс концентрує увагу людини виключно на доступних діях, адаптуючись під будь-які дисплеї.

Навігаційний блок перетворено на компактний пульт керування. Логіка взаємодії вибудувана абсолютно лінійно. Перехід до orders.php одразу розгортає історію покупок та актуальні статуси доставки. Для отримання рекомендацій щодо стилю реалізовано систему цілу систему. Один клік на активує глибоку інтеграцію з Gemini API для генерації персональних луків, порад тощо. Може здатися дещо надлишковим паралельне існування загального чату. Втім, практичні метрики доводять гостру потребу в резервному комунікаційному каналі з живими операторами. Користувачу може знадобитися дпопмога яку ШІ-асистент надати не в змозі. Життєво важливим технічним вузлом залишається лінк на зміну профіля користувача. Саме там клієнт актуалізує своє ПІБ, що згодом гарантує безпомилкову автоматичну генерацію логістичних ТТН. Для того, щоб покупець не застряг у навігаційному вакуумі, система постійно тримає під рукою швидкі повернення до каталогу. Фінальналізує це все – скрипт logout.php. Він забезпечує чисте та безпечне знищення сеансу після натискання кнопки виходу.

Зовнішній вигляд цього прагматичного інтерфейсу зафіксовано рисунку А.4.

3.5.1 Проєктування та програмна реалізація модуля «Мої замовлення» та інтегрованої системи відгуків

Модуль історії покупок (orders.php) виступає не простим статичним довідником. Це є потужним інструментом для забезпечення тотальної прозорості логістики. Під час розробки не було класичної ідентифікації суто за внутрішнім user_id. Натомість бекенд жорстко прив'язує замовлення до електронної пошти. Система не намагається ускладнити логіку. Клієнт миттєво отримує доступ до історії навіть тих покупок, якщо він здійснював як гість задовго до повноцінної реєстрації. Єдиний спільний email автоматично склеює цей розрізнений досвід у монолітний профіль.

Дані зберігаються у гібридному форматі. При розробці багато розробників уникають денормалізації таблиць через потенційні ризики.Проте пакування складу замовлення у JSON-формат (поле order_data) заради максимальної швидкодії фронтенду. Багаторівневий парсинг одразу в процесі витягує назви, ціни та мініатюри товарів. Звісно, завжди існує ймовірність програмного пошкодження такого масиву. Тому система озброєна надійним запобіжником: у разі збою під час розбору JSON, алгоритм миттєво перемикається на читання класичної реляційної таблиці order_items. Це запобігає відображенню порожнього екрана.

Логістичний блок синхронно виводить згенеровані через API Нової Пошти номери ТТН. Тут криється неочевидний, але надважливий архітектурний нюанс. Повні координати доставки (місто, регіон, відділення) фіксуються безпосередньо у транзакційній таблиці замовлень. Це забезпечує повну ізоляцію історичних даних. Навіть якщо завтра клієнт переїде і змінить

адресу у своєму профілі, старі накладні залишаться документально недоторканими.

Інтегрована система зворотного зв'язку також працює за максимально жорсткими правилами. Відгук фізично можливо залишити лише після того, як статус зміниться на «Доставлено». Це запобігає на корені будь-які спроби накрутки рейтингу. Додатковим фільтром виступає унікальна перевірка зв'язки product_id та user_id, яка на рівні бази даних блокує дублювання коментарів. Далі підключається серверна математика. Валідатор пропускає лише ті запити, де рейтингова оцінка потрапляє в діапазон $R \in [1,5]$, а текстовий блок налічує щонайменше $L \geq 10$ символів. Лише після успішного проходження цих бар'єрів та ручної перевірки модератором (статус pending) коментар конвертується у реальний, що підвищує коефіцієнт довіри(K_{trust}), механіку якого детально препарували у Розділі 2.

Наочно інтерфейс логістичного контролю представлено на рисунку – А.5. Щойно система фіксує успішну доставку, вона розблоковує модальне вікно публікації відгуків (рисунок А.6).

Реляційним фундаментом для всього описаного масиву функцій слугують таблиці orders та reviews. Їхня розгорнута структура, що повністю відповідає фактичній імплементації бази даних, наведена у таблицях Б.2 – Б.3.

3.5.2 Проєктування та програмна реалізація модуля «Чат з ШІ-Асистентом» (ai-chat-fullscreen.php)

Розробка ШІ-асистента вимагала фокосування на комерційному успіху. Це справжня та головна конкурентна перевага платформи. Замість того, щоб утримувати роздутий штат консультантів, було делегувано тонку персоналізацію клієнтського досвіду великій мовній моделі Gemini. Вона

працює в режимі реального часу, але її налаштування вимагають тонкого балансу між якістю та економікою.

Встановлення програмного обмеження у кількості 5 запитів на одну користувацьку сесію є цілеспрямованим архітектурним рішенням, яке потребує технічного обґрунтування. Проте цей крок обґрунтований вимогами оптимізації ресурсів. По-перше, надійно купірувано ризики зловживань та неконтрольованого спалювання бюджету на виклики API. По-друге, така квота психологічно дисциплінує. Вона стимулює покупця формулювати свої побажання конкретніше, а алгоритм – видавати максимально влучні товарні рекомендації без зайвих дій.

Далі розглянуто особливості користувацького інтерфейсу чату. Інтерфейс зроблено абсолютно пластичним. Завдяки скрипту theme-manager.js він миттєво й безшовно мімікрує під системні вподобання користувача (світла чи темна тема). Відомо, що пошук оптимального луку – це часто розгорнутий опис тканин, фасонів та настрою. Тому поле вводу органічно підтримує багаторядкову генерацію тексту через комбінацію Shift+Enter. Це робить неможливим випадкове відправлення неповного повідомлення. Це підвищує загальну ергономіку інтерфейсу.

Вимога обов’язкової реєстрації користувача для доступу до основного функціоналу платформи є недоцільною. Гібридна система авторизації руйнує такий бар’єр. Модуль радо обслуговує навіть анонімних гостей. Для ідентифікації «холодного» трафіку бекенд генерує тимчасові маркери сесії, тоді як зареєстровані клієнти жорстко прив’язуються до свого user_id. Функціонал асистента стає доступною з першої ж секунди перебування на сайті.

Стратегічним вектором розвитку модуля є перехід до мультимодальної взаємодії. Теоретичне обґрунтування розширення можливостей чату за рахунок візуальної ідентифікації товарів ґрунтується на застосуванні сучасних методів комп’ютерного зору та алгоритмів класифікації зображень

[18]. Впровадження такого інструментарію дозволить асистенту аналізувати фотоматеріали користувачів для надання точних стилістичних порад.

Візуальну складову цього флагманського інтерфейсу зафіксовано та представлено на рисунку – А.7.

Вся історія взаємодії, записується у спеціалізовану таблицю ai_chat_history. Завдяки цьому модель аналізує попередні репліки, «розуміє» контекст і з кожним кроком б’є все точніше в ціль. Особлива увага під час розробки ШІ-асистента приділялася якості та природності генерації відповідей для клієнтів. З цією метою під час проєктування доцільно спиратися на передові методи, що передбачають комбіноване використання кількох версій мовної моделі для оптимальної генерації тексту [19]. Більше того, цей масив логів перетворюється на фундамент для довгострокової бізнес-аналітики. Структура таблиці ai_chat_history описана та представлена у таблиці – Б.4.

Під час інтеграції інтелектуального помічника у існуючу веб-платформу критично важливим етапом є комплексна перевірка надійності програмного забезпечення. Детальний аналіз існуючих механізмів тестування веб-застосунків [20] дозволяє обрати оптимальні стратегії для виявлення дефектів інтеграції API та забезпечення безперебійної взаємодії клієнтської та серверної частин.

3.5.3 Проєктування та програмна реалізація гібридного модуля клієнтської підтримки «Smart Chat» (chat-fullscreen.php)

Проєктування комунікаційного хабу (chat-fullscreen.php) зумовило необхідність оптимізації алгоритмів маршрутизації клієнтських запитів. Недоцільним є спрямування всіх базових запитів на опрацювання важкій нейромережі. Це невиправдане та ірраціональна використання ресурсів. Тож було розгорнуто жорстку трирівневу архітектуру підтримки. Перша лінія

оборони – це так званий статичний фільтр. Блок compact-faq-buttons опрацьовує основну частину користувацьких запитів. Інтерактивні тригери доставки, оплати чи графіку роботи миттєво генерують заздалегідь «зашиті» відповіді. Практика доводить: такий банальний на перший погляд інтерфейс успішно «зрізає» до 80% типового інформаційного шуму, та дуже знижує навантаження як на адміністрацію сайту так і на ШІ-асистента. Серверні потужності API залишаються вільними. Користувач же не витрачає жодної секунди на очікування генерації тексту, чи відповіді від адміністратору.

Для обробки решти 20% нестандартних запитів застосовується механізм ескалації. Кліку на кнопку «Консультація ШІ» достатньо, щоб система безшовно перекинула клієнта до активного діалогу з Gemini. Саме тут розгортається повноцінний діалог з порадами та вільна генерація персональних луків. Проте цілком логічно припустити, що навіть найдосконаліша мовна модель фізично позбавлена доступу до специфічних бек-офісних процесів магазину а знає лише контекст. Для вирішення таких вузьких інцидентів архітектура передбачає третій, найвищий рівень – прямий лінк із живим менеджером.

Конфлікт між машинним та людським часом розв’язується через мультипанельну структуру. Завдяки системі незалежних вкладок (Tabs), клієнт вільно пермикатися між чатами для спілкування. Жодних розривів комунікації та жодних оновлень сторінки. Можна написати скаргу живому оператору (ця вкладка обладнана чесним індикатором «Онлайн/Офлайн») і, не гаючи часу на очікування відповіді, паралельно підбирати осінній гардероб з ШІ-асистентом у сусідньому вікні.

Окрім динамічної зміни візуальної теми, інтерфейс глибоко поважає приватність. Психологічний комфорт забезпечується функцією тотальної очистки історії сесій – клієнт власноруч контролює свої цифрові сліди. Топологію цього гібридного модуля наочно ілюструють рисуноки А.8 та А.9.

На рівні бази даних платформи за збереження людської комунікації відповідає ізольована реляційна таблиця admin_chat_messages. Вона створює

фундамент архітектури модуля, гарантуючи, що жоден нестандартний запит до адміністрації не зникне в нікуди навіть при раптовому закритті браузера. Опис структури таблиці admin_chat_messages надано у таблиці – Б.5.

3.6 Профіль адміністратора (Admin Profile)

3.6.1 Проєктування та програмна реалізація модуля «Адміністраційна панель керування замовленнями» (admin_orders.php)

Серцевиною бек-офісу виступає модуль admin_orders.php. Даний компонент виконує розширені функції, що виходять за межі простої фіксації кількісних показників. Це компактний та зручний командний пункт керування бізнес-процесами. Оператор тут не лише механічно перемикає статуси відправлень (з умовного «В обробці» на жадане «Доставлено»). Він отримує в руки потужний інструмент для препарування маркетингової ефективності, зокрема – для оцінки реальної конверсії від розданих промокодів. Але доступ сюди суворо регламентований. Доступ надається тільки тим користувачам хто є адміністратором(піднято прапорець isadmin).

Далі постає логічне питання швидкої навігації у величезних масивах транзакцій. Це питання треба було вирішити. Архітектура пошуку спирається на динамічну генерацію SQL-запитів. Механізм здатний одночасно обробляти шість незалежних параметрів: від номера накладної та ПІБ клієнта до конкретного використаного промокоду. Теорія проєктування баз даних може наполягають на використанні виключно точного пошуку для уникнення колізій. Проте сувора практика інтернет-торгівлі вимагають інакших правил гри. Механізм часткового збігу рядків виявляється на порядок ефективнішим, рятуючи менеджера від рутинного або навітьручного пошуку за обривками email-адрес чи номерів ТТН.

Життєвий цикл кожної транзакції контролюється буквально у кілька кліків через візуальний інтерфейс. Будь-яка зміна логістичного статусу

миттєво ініціює перезапис рядків у базі. Паралельно з цим процесом незупинно працює аналітичний дашборд. Такі процеси спрямовані на поліпшення швидкодії та банального комфорту при роботі адміністратора.

Система автоматично обчислює ключові показники ефективності (KPI), зокрема загальний обсяг замовлень та кількість нових транзакцій. Сумарний обсяг згенерованих знижок. Уся ця математика без жодних затримок лягає на екран адміністратора. Наочно інтерфейс цього аналітичного хабу представлено на рисунку А.11.

Фундамент, на якому тримається не лише ця панель, а й уся комерційна логіка платформи, виступає таблиця orders. Структура таблиці замовлень представлена на таблиці – Б.6

3.6.2 Проєктування та програмна реалізація модуля «Адміністраційна панель модерації відгуків» (admin_reviews.php)

Контроль якості користувацького контенту (UGC) вимагає жорсткої дисципліни. Залишити вітрину магазину без нагляду – прямий шлях до репутаційних втрат через накрутки або спам. Саме тому скрипт admin_reviews.php функціонує як безкомпромісний фільтр, що захищає релевантність рейтингів.

Життєвий цикл будь-якої оцінки є чітко та суворо регламентованим. Жоден коментар не потрапляє на сторінку товару автоматично. База даних примусово присвоює свіжим дописам статус pending, повністю приховуючи їх від сторонніх очей. Лише після візуального контролю менеджер активує стан approved, легалізуючи публікацію. Недоречний контент просто не пройде модерацію та отримає статус rejected і вилучається з публічного доступу.

Звісно, існує потужний тренд на повністю автоматизовану модерацію через алгоритми обробки природної мови (NLP). Проте на поточному етапі розгортання проєкту було обрано надійність а не трендовість.

Візуальну хронологію цього процесу послідовно фіксують графічні матеріали. Базовий інтерфейс панелі контролю наведено на рисунку А.12. Щойно оператор затверджує відгук (рисунок А.13), він миттєво матеріалізується на клієнтській сторінці відповідного товару (рисунок А.14).

Увесь цей механізм спирається на реляційну таблицю reviews, яка через зовнішні ключі намертво пов’язана з масивом користувачів для безпомилкової ідентифікації авторів. Таблиця – Б.7 описує структуру бази даних з відгуками.

3.6.3 Проєктування та програмна реалізація модуля «Адміністраційна панель конструктор оголошень» (add_product.php)

Наповнення електронної вітрини – це далеко не тривіальна рутина. Спроєктований модуль add_product.php функціонує як гнучкий інтерактивний конструктор, який фізично неможливо запустити без відповідних привілеїв. Серверна логіка повністю обмежує будь-які спроби несанкціонованого редагування комерційного контенту. Скрипт не просто шукає активну сесію, а прискіпливо валідує прапорець адміністратору.

Значна частина популярних CMS характеризується надмірною універсальністю інтерфейсів, що ускладнює навігацію адміністратора через надмірну кількість полів. Концепція базується на тотальній адаптивності (Dynamic UI). Щойно оператор обирає специфічну категорію товару, JavaScript-тригер updateSizes() миттєво перезбирає доступну сітку розмірів. Для взуття бекенд генерує суворий числовий діапазон (36-46). Натомість для одягу автоматично підтягується класична літерна шкала (від XS до XXXL). Оператор фізично не має шансу зробити логічну помилку.

Цілком зрозуміло, що «сліпий» імпорт графіки катастрофічно уповільнює роботу контент-менеджера. Тому механізм завантаження мультимедіа від самого початку проєктувався під пакетну обробку фотографій через атрибут multiple accept=image/*. Для того, щоб уникнути публікації помилкового візуалу, фронтенд-функція previewFiles() рендерить локальні мініатюри зображень ще до моменту фактичної відправки форми на сервер. Менеджер повністю контролює ситуацію.

Архітектура бази даних передбачає розділення полів актуальної та попередньої вартості товару. Такий підхід дозволяє клієнтській частині застосунку автономно розраховувати відсоток знижки та генерувати відповідні графічні індикатори (бейджі) в каталозі без додаткового адміністрування чи зміни програмного коду.

Візуал та ергономіку цього адміністративного блоку детально зафіксовано на рисунках А.15 та А.16.

Завершувальний етап – це зібраний масив даних, який перехоплює транзитний скрипт add_product_handler.php. Саме він відповідає за безпечну ін’єкцію нових параметрів до бази. Нижче наведено розгорнуту специфікацію полів таблиці products, яка де-факто утримує на собі всю товарну номенклатуру платформи. Цю структуру зображено у таблиці – Б.8.

3.6.4 Проєктування та програмна реалізація модуля «Адміністраційна панель редактор оголошень» (admin_products.php)

Управління тисячами товарних позицій з часом перетворюється на операційний хаос без єдиного центру контролю. Саме таку роль координаційного хабу бере на себе скрипт admin_products. Менеджер отримує не просту статичну таблицю, а надзвичайно маневровий інструмент для моніторингу всього асортименту.

Існує поширена ілюзія, що стандартного посторінкового виведення (пагінації) цілком достатньо для зручної навігації. Сувора реальність e-commerce доводить повністю протилежне. Коли рахунок номенклатури йде на тисячі, критичним параметром стає швидкість доступу до конкретної одиниці. Тому ядро модуля покладається на інтелектуальну систему багатофакторної фільтрації. Алгоритм одразу вихоплює потрібний товар за брендом, унікальним складським артикулом (art), категорією чи навіть фрагментом назви. Жодних затримок у пошуку.

Далі під’днується продумана візуальна ергономіка. Розробники бек-офісів часто ігнорують проблему «порожніх полиць», змушуючи персонал вчитуватися в цифри залишків. Тут інтерфейс працює інакше. Розпродані позиції автоматично «підсвічуються» контрастними графічними індикаторами. Адміністратор оперативно отримує інформацію про відсутні товарні позиції, які вимагають термінового поповнення.

Паралельно з логістикою відбувається жорсткий фінансовий моніторинг. Виведення поруч актуального та старого прайсів у загальному реєстрі.

Також, цей дашборд слугує головним стартовим майданчиком для роботи з контентом. Один клік – і застаріла позиція назавжди видаляється з каталогу. Інший клік запускає модуль глибокого коригування edit_product.php. Технічно цей скрипт є ідентичним клоном уже знайомого конструктора add_product.php. Він дозволяє миттєво переписувати описи, оновлювати фотографії та змінювати статуси наявності.

Робочий простір цього редакторського блоку наочно демонструє рисунок А.17 та А.18

Немає сенсу вдруге дублювати опис реляційних зв’язків. Уся розгорнута специфікація масиву products вже була детально була детально описана раніше (Таблиця Б.8).

3.6.5 Проєктування та програмна реалізація модуля «Адміністраційна панель управління користувачами» (admin_users.php)

Модуль admin_users.php виступає ядром рольового доступу (RBAC). Це головний пульт бек-офісу, де вирішується, хто саме керуватиме платформою.

У цьому проєкті архітектура безапеляційно відкидає такий сценарій коли адміністратор додав ще купу інших адмін-акаунтів. Для цього створено акаунт «захищеного адміністратора» який є абсолютно недоторканним на програмному рівні. Його фізично неможливо понизити у статусі. Додатково впроваджено превентивний механізм від самоблокування: система просто не дозволить поточному оператору зняти привілеї із самого себе. Жодних випадкових втрат контролю над панеллю. Тільки жорстка ієрархія.

Шукати потрібну людину вручну у великій базі – є доволі не ефективним. Тому гібридний пошуковий алгоритм миттєво «просіює» таблицю одночасно за трьома векторами: ім’ям, логіном та email-адресою.

Зовнішній вигляд цього захищеного інтерфейсу детально зафіксовано на рисунку – А.19.

Апаратним фундаментом для описаних алгоритмів слугує таблиця users. Її структуру наведено у таблиці – Б.9

3.6.6 Проєктування та програмна реалізація модуля «Адміністраційна панель керування чатами» (admin-chat-fullscreen.php)

Організація клієнтської підтримки вимагала створення повноцінного цифрового пульта для оператора (admin-chat-fullscreen.php). Будь-яка взаємодія тут починається з безжального фільтра на статус адміністратора.

Потрапивши всередину, менеджер отримує сегментований робочий простір. Свіжі запити не змішуються з уже вирішеними проблемами тому, що – це гарантований шлях до операційного хаосу. З огляду на це, інтерфейс

автономно розщеплює потік діалогів на дві ізольовані черги: «Активні», та архівні «Завершені». Концентрація уваги фахівця стає максимальною. Завдяки глибокій прив’язці до фронтенд-модуля admin-chat-fullscreen.js, нові репліки прилітають на екран абсолютно синхронно, без необхідності повного перезавантаження сторінки.

Окремий нюанс – це питання життєвого циклу комунікацій. За для вирішення цього питання та оптимізації серверної пам’яті адміністратор озброєний інструментом примусового переривання сесії. Клікнувши «Завершити чат», він неменуче скидає розв’язану проблему до архіву, паралельно вивільняючи обчислювальні ресурси платформи.

Інтерфейс цього комунікаційного вузла демонструє рисунок А.20 та рисунок А.21.

Архітектурним фундаментом для маршрутизації всього масиву повідомлень виступає реляційна таблиця admin_chats. Опис структури наведено у таблиці – Б.10

3.6.7 Проєктування та програмна реалізація модуля «Адміністраційна панель керування промокодами» (admin_promo.php)

Попри простоту, генерація знижок вимагає математичної логіки. Скрипт admin_promo надає менеджеру розширені можливості для управління. Жодної рутини. Конструктор дозволяє швидко налаштувати тіло самого промокоду, відсоток дисконту, максимальну кількість активацій та жорсткі часові рамки. Щойно новий запис відправляється до сервера, алгоритм миттєво виконує превентивну перевірку на наявність дублікатів.

Далі бекенд повністю перебирає керування на себе. Життєві цикли акцій перемикаються абсолютно автономно: від запланованого upcoming до робочого active чи вже недійсного expired. Маркетологу залишається лише

аналізувати вбудовану панель статистики, спостерігаючи за реальним фінансовим вихлопом від рекламної кампанії.

Візуально цей маркетинговий пульт зафіксовано на рисунку А.22 та А.23.

Вся архітектура лояльності надійно запакована у таблицю promocodes, яка виступає критично важливим вузлом загальної підсистеми білінгу. Структуру описаної таблиці наведено у додатку Б (таблиця – Б.11)

3.7 Проєктування та програмна реалізація блоку: «Каталог товарів» (catalog.php)

Головна вітрина платформи (catalog.php) – це територія гібридної архітектури. Існує технічна можливість перенести весь рендеринг виключно на серверну частину. Проте такий архаїчний підхід миттєво погіршує динаміку шопінгу. Тому обов’язки чітко розділено. PHP бере на себе важку бекенд-логіку: ініціалізацію сесій, контроль прав доступу та генерацію первинних списків категорій безпосередньо з бази. Натомість фронтенд повністю віддано у частину JavaScript. Асинхронне завантаження товарних карток та миттєве застосування фільтрів відбуваються приховано, на серверному рівні. І ніяких перезавантажень сторінки. Покупець отримує абсолютно плавний, інтерактивний досвід. Такий підхід прямо впливає на комфорт та позитивно впливає на рівень коефіцієнта довіри (K_{trust}).

Багаторівнева система фільтрації працює чітко та безвідмовно. Клієнт не обмежений єдиним критерієм. Він вільно комбінує незалежні вектори пошуку: повнотекстовий запит за моделлю, жорстку сегрегацію за категорією одягу чи взуття, вибір улюбленого бренду та специфічну економічну палітру.

Доволі часто розробники-початківці нехтують складними алгоритмами ранжування, залишаючи лише базові опції. У цьому випадку тратегія кардинально інакша. Інтелектуальне сортування за часом додавання,

динамікою прайсу чи алфавітом дозволяє в один клік отримати найсвіжіші новинки або найглибші знижки каталогу. Таким чином підвищується якість та точність пошуку зменшуючи $K_{friction}$ – коефіцієнт технічного тертя.

Окрім текстового аналізу, алгоритми машинного навчання демонструють високу ефективність у задачах класифікації візуальних об’єктів, що відкриває перспективи для автоматичного тегування товарів у каталозі при майбутньому маштабуванні [21].

Для безпечної передачі авторизаційного контексту із серверної частини на клієнтську застосовано глобальний об’єкт window та спеціальні метатеги. Сервер «прошиває» ці маркери ще на етапі генерації документа. Завдяки цьому фронтенд набуває здатності до самоадаптації. Інтерфейс на при ініціалізації розпізнає ролі: адміністратору автоматично надається доступ до елементів управління, тоді як гість бачить лише чисту вітрину без зайвого візуального шуму. Своєю чергою, динамічний заголовочний блок або Header формує випадаючі списки виключно на основі реального вмісту бази. Базовий SQL-запит із параметром DISTINCT гарантує залізобетонну актуальність. Щойно адміністратор завантажить товар нового бренду, він миттєво матеріалізується в навігації без жодного втручання в код та витрат часу.

Окремий технічний, не менш важливий, виклик полягав в інтеграції кошика та модулів ШІ-підтримки. Серцем цього процесу виступає об’єкт CartManager. Алгоритм забезпечує керування модальними вікнами вибору розмірів і через надійний JSON-міст асинхронно синхронізує стан покупок із PHP-сесією. Плаваючі тригери інтелектуального чату та бокові панелі зберігають повноцінну ергономіку навіть на найменших смартфонах.

Останнім невирішеним моментом при моделюванні сторінки каталогу залишалася її візуальна цілісність. Сучасна технологія MutationObserver безперервно відстежує зміну світлої чи темної теми на основному сайті, синхронно перефарбовуючи вкладені модальні вікна чату. Жодного

розсинхрону в дизайні. На рисунку А.24 зображено інтерфейс блоку каталог товарів.

Коректна робота каталогу забезпечується злагодженою роботою декількох програмних блоків які наведені у таблиці – Б.12.

3.7.1 Проєктування та програмна реалізація блоку: «Сторінка товару»

Фінальна частина будь-якої воронки продажів лунає саме тут – на сторінці детального огляду (product.php). Це найважливіша сторінка сайту. Це територія остаточного рішення клієнта. Алгоритмічний сценарій стартує з банального, на перший погляд, перехоплення унікального ID через GET-запит. Утім, далі сервер викликє дійсно важкі запити. Замість ризикованих прямих запитів, система ініціює складну SQL-вибірку виключно через підготовлені вирази (prepared statements).

Бекенд бездоганно витягує з бази повний спектр атрибутів: від базової назви та бренду до багатовимірних масивів доступних розмірів, кольорів і, звісно ж, поточного складського статусу.

Окремий виклик для фронтенду – це робота з медіаконтентом. База віддає лише «сирий» серіалізований рядок шляхів до файлів. Програмний парсер розщеплює його, миттєво формуючи гнучку галерею: масивне головне фото та сітку мініатюр. Динамічне перемикання ракурсів відбувається абсолютно безшовно. Жодних підвисань сторінки. Чимало інтернет-магазинів досі змушують контент-менеджерів вручну вираховувати акційні відсотки. У сучасних умовах розробки такий підхід є справжнім рудиментом. Алгоритм робить це математично та автоматично на основі старої ціни, одразу генеруючи яскраві графічні маркери знижок. Щодо варіативності товару, то було впроваджено жорсткий UX-контроль. Кнопка додавання до кошика залишається неактивною, доки покупець свідомо не клікне на конкретний розмір.

Важливою особливістю проєктування інтерфейсу є вміщення величезного масиву технічної інформації у вузький екран смартфона, не спровокувавши у клієнта когнітивне перевантаження. Вирішенням стала класична, але безвідмовна система вкладок (Tabs). Вона елегантно ізолює розширений опис, таблицю характеристик та соціальні докази один від одного. Модуль відгуків тісно сплетений із рейтинговим рушієм: кожна оцінка конвертується у графічні зірки, а алгоритм паралельно вираховує загальний середній бал. Щоб додатково розігнати показник середнього чека, сторінка автоматично підтягує блок крос-селу (схожих позицій), орієнтуючись виключно на ідентичну товарну категорію.

Технічна монолітність усього цього ансамблю фіксується одним чітким маневром. Бекенд інкапсулює зібраний об’єкт товару і напряму передає його в глобальний контекст JavaScript. Цей інформаційний міст гарантує, що обрані параметри синхронізуються з менеджером кошика за долі секунди. Екранний інтерфейс цього конверсійного вузла представлено на рисунку Б.25.

Коректна робота каталогу та картки товару забезпечується злагодженою роботою наступних програмних блоків, які наведено у таблиці Б.13.

3.8 Проєктування та програмна реалізація блоку «Оформлення замовлення»

Екран оформлення замовлення (checkout.php) – це найвужче місце будь-якої комерційної воронки. Саме тут клієнт найчастіше перериває процес замовлення через технічні збої. Згідно з аналізу, безперебійна робота та загальна оптимізація процесу онлайн-замовлення виступають ключовою детермінантою успіху в електронній комерції, оскільки безпосередньо впливають на лояльність покупця та фінансові метрики [22]. Алгоритм не

покладається на ідеальну поведінку браузера чи стабільність мережі. Спочатку скрипт намагається десеріалізувати масив $_POST, переданий із фронтенду. Якщо ж, щось піде не так або клієнт натисне F5 – то система не зламається. Коли абсолютна більшість платформ просто скинуть кошик, змушуючи покупця починати все з нуля. Система діє інакше. Вона миттєво звертається до резервної копії у глобальній PHP-сесії. Жодної втрати конверсії через технічні форс-мажори.

На стороні бекенду кожен елемент має свої чіткі фільтри типізації. Ідентифікатори товарів та їхня кількість конвертуються в цілі числа. В свою чергу ціни обов'язково перетворюються на формат із плаваючою комою. Будь-які текстові назви чи шляхи до фотографій очищуються від потенційно небезпечного коду. Це повністю нейтралізує спроби несанкціонованої підмінити вартість товару через інспектор браузера. Паралельно з цією зачисткою алгоритм непомітно підтягує контакти з таблиці users для авторизованих покупців. Водночас «холодний» трафік безперешкодно проходить через гостьове оформлення. Відсутність примусової реєстрації перед етапом оплати дозволяє зберегти цілісність воронки продажів.

З метою запобігання несанкціонованому використанню акційних пропозицій розроблено відповідну алгоритмічну логіку. Модуль обробки промокодів використовує SQL-запити для синхронної верифікації часових міток (таймстампів) акції із системним часом сервера. Проте лише часових меж недостатньо. Скрипт суворо контролює загальні ліміти (max_uses) та особисто перевіряє історію конкретного покупця через таблицю used_promocodes. Алгоритм миттєво блокує будь-яку спробу повторно надати системі одноразовий дисконтний код. Тільки після успішної валідації всіх вище описаних пунктів – математичний рушій перераховує фінансові показники. Фінальні значення discount_amount та final_amount надійно фіксуються у прихованих полях форми. У випадках коли знижка не використовується а просто записується у поточну ціну у поле final_amount.

PHP-скрипт безпечно ін’єктує ключ доступу безпосередньо у глобальний об’єкт window. Цей тригер пробуджує JavaScript-модуль nova_poshta_autocomplete.js. Тут починається динамічний пошук. Система інтерактивно спілкується з серверами логістичного оператора(Нова пошта), моментально підвантажуючи населені пункти та актуальні відділення. Це не просто красива фішка для покращення UX. Це критичний інструмент боротьби з друкарськими помилками в адресах, які неминуче призводять до повернень товару.

Фінал усієї сесії – пакування зібраного масиву даних у чистий JSON. Цей пакет і відправляється до обробника add_order.php, який виконує останній фінальний аудит і завантажує дані зібрані від клієнта в реляційній таблиці замовлень. Інтерфейс блоку офрмлення замовлень зображено на рисунку – А.26

3.8.1 Проєктування та програмна реалізація блоку «Чек замовлення»

Сторінка order_success.php поєднує функції безпеки та інформування клієнта. Для запобігання несанкціонованому доступу через підміну ID, вхідний параметр $_GET примусово приводиться до цілочисельного типу та проходить перевірку через підготовлені вирази mysqli_prepare. У разі відсутності запиту в базі даних користувач миттєво перенаправляється на головну сторінку.

Після успішної верифікації клієнт бачить номер замовлення та реквізити ТТН. Фінансові дані структурується за допомогою функції number_format, при цьому умовний рендеринг приховує поля, що не стосуються конкретного замовлення, зокрема при ігноруванні промокодів.

Для підвищення довіри та виключення ризику втрати даних, система автоматично надсилає цифровий чек на e-mail клієнта одразу після запису замовлення в БД. Такий підхід гарантує наявність документального

підтвердження транзакції поза межами браузера. Візуалізацію інтерфейсу та структуру електронного листа-квитанції наведено на рисунках А.27 та А.28 відповідно.

3.9 Проєктування та програмна реалізація блоку кошика

3.9.1 Реалізація інтерфейсу кошика та системи вибору параметрів товару

Відривати покупця від вітрини заради переходу до кошика – гарантоване падіння конверсії. Саме тому модуль cartModal функціонує виключно як динамічний напівпрозорий оверлей. Людина фокусується на обраних позиціях, але візуально залишається в каталозі. Жодних перезавантажень сторінки.

Уся, доволі складна, внутрішня логіка підпорядкована механізму умовного рендерингу. Порожня сесія миттєво генерує лаконічну заглушку із закликом продовжити шопінг (загальний вигляд цього стану демонструє рисунок А.29). Щойно лунає клік додавання, інтерфейс автоматично перезбирається. З’являються інформативні рядки з мініатюрами, а нижня панель cartFooter виходить із тіні, агрегуючи фінальний чек у реальному часі. Лаконічниість та простота цієї зони не є випадковою: кнопка оформлення суворо прорахована під сліпе натискання великим пальцем на мобільних дисплеях.

Окремим архітектурним викликом стала проблема вибору габаритів. Деякі інженери намагаються втиснути цю логіку безпосередньо в загальний каталог, перевантажуючи дизайн. Було обрано шляхом ізоляції.

Запуск незалежного модуля sizeModal (рисунок А.30) діє як жорсткий фільтр помилок. Цей контейнер динамічно підтягує з бази лише ті розміри, які фізично наявні на складі для конкретної моделі. Замовлення «фантомної» позиції стає технічно неможливим. За лаштунками цих процесів безупинно

працює точковий JavaScript-менеджер. Він легко маніпулює глобальними класами станів за їхніми унікальними ідентифікаторами. Анімації плавні. Перерахунок кількості миттєвий. Увага сучасного користувача занадто крихка, щоб змушувати його чекати на рендеринг хоча б зайву мілісекунду. Інтерфейс простий задля покащення продуктивності.

3.9.2 Архітектурна реалізація та алгоритми функціонування кошика(cart.js)

Основним елементом досвіду керує вузькоспеціалізований клас CartManager. Було обрано прототипно-орієнтований підхід для його архітектури. Вибір такого підходу обґрунтовано необхідністю. Це радикально економить оперативну пам’ять клієнта під час обробки сторінок зі складним життєвим циклом.

Процес ініціалізації стартує, як завжди, з перевірки прапорців стану. Подвійне навішування обробників подій – це гарантований шлях до витоків пам’яті та збоїв інтерфейсу. Тому алгоритм превентивно блокує повторні активації. Далі коли менеджер завантажує динамічні стилі, витягує ідентифікатори користувача і проводить первинне «зшивання» локальних даних із сесією PHP.

Ідентифікація покупця була довгим та складним процесом та вимагала створення справжнього каскаду перевірок. Скрипт не покладається наосліп на єдине джерело. Спочатку алгоритм аналізує мета-теги та атрибути документа. Якщо там порожньо, система обов’язково ініціює фоновий запит до бекенд-вузла get_user_id.php. Це дозволяє намертво прив’язати поточний кошик до конкретного облікового запису. На етапі тестування минулий етап потребував багато часу. Але найскладнішим інженерним викликом стала проблема визначення «джерела істини». Для синхронізації даних між локальним сховищем браузера та серверною сесією використовується

спеціальний ідентифікатор у localStorage. Наявність цього маркера дозволяє системі диференціювати втрату мережевого з'єднання від цілеспрямованого видалення товарів із кошика користувачем. Відповідно до цього маркера, система динамічно передає право першості серверу або клієнту.

Архітектура збереження спирається на динамічну генерацію ключів. Анонімні гості отримують статичний версіонований хеш. Він надійно утримує вибрані товари навіть після жорсткого закриття вкладки, перезавантаження сторінки, та навіть жорсткого перезавантаження кешу. Для авторизованих клієнтів ключ формується виключно на базі їхнього унікального ID з бази.

Перед кожним записом у локальну пам'ять масив проходить крізь монументальну базовану процедуру валідації. Ціни примусово кастуються у числа з плаваючою комою, а ідентифікатори обов'яково – у рядки. Такий підхід є базисом. Це формує непробивний бар'єр проти пошкодження структури JSON, рятуючи скрипт від фатальних помилок при подальшій десеріалізації.

Додавання кожної нової позиції працює за принципом інтелектуального злиття. Якщо ідентифікатор та обраний розмір збігаються з уже наявними у кошику, система не плодить нові рядки, а просто інкрементує лічильник. Жодних дублікатів-клонів у списку. Кожна така транзакція миттєво викликає метод syncCartToServer. Він пакує актуальний стан і відправляє його POST-запитом прямо в PHP-сесію. Візуальний фідбек відпрацьовує синхронно. Лічильник у хедері оновлюється. Виринає спливаюче повідомлення. Активується CSS-анімація іконки. Конверсія залежить від динаміки, і фізично привертається погляд покупця до кожної успішної дії.

Взаємодія менеджера з вітриною побудована через систему гнучких адаптерів. Метод getProductDataFromPage займається класичним витягуванням даних безпосередньо з DOM-дерева. Він зчитує назву, ціну та лінки на фотографії прямо з HTML-селекторів поточної сторінки. Дехто

вважає такий підхід менш надійним, вимагаючи постійних запитів до серверного API. Проте обрана стратегія економить сотні мілісекунд і суттєво розвантажує базу при кожному кліку. Для загального каталогу працює функція-міст addToCartWithSize, яка безпомилково ідентифікує товарну картку.

Сама ж логіка вибору габаритів інкапсульована у проміс-орієнтований метод loadAvailableSizes. Цей алгоритм асинхронно опитує склади через бекенд і моментально блокує в модальному вікні ті розміри, які вже розпродані. Жодних хибних очікувань у клієнта.

3.10 Проєктування та програмна реалізація інтелектуального модуля на базі API Gemini

Інтеграція мовної моделі Gemini спонукала нас запровадити ізольованого бекенд-посередника.

Спочатку маршрутизатор конфігурує CORS-заголовки, легалізуючи крос-доменні запити, та фіксує UTF-8 кодування для JSON-пакетів. Відтепер кирилиця не перетвориться «ієрогліфи». Далі алгоритм починає прослуховувати потік php://input. Десеріалізація вхідного об’єкта миттєво запускає багаторівневий аудит. Система не може просто валідувати наявність тексту. Ще вона вираховує квоти сесії. Економіка проєкту не бескінечна. Варто клієнту вичерпати встановлений ліміт запитів, як обробник перериває транзакцію, повертаючи у фронтенд структурований код помилки.

Окремим моментом є внутрішня механіка prompt-інжинірингу. Змусити нейромережу коректно працювати в жорстких межах конкретного бізнесу вкрай важко без правильного бекграунду. Тому система створює приховану системну інструкцію (system prompt). Модель отримує чітку роль приязного консультанта магазину, її цілі та функції. Але текст відповідей є суто

українською мовою. А весь цей фізичний контакт із сервісами ШІ забезпечує функція callGeminiAPI.

Для реалізації мережевих запитів використовується тут працює класична бібліотека cURL. Певна частина інженерів залишає параметри LLM за замовчуванням, проте такий інертний підхід часто призводить до фактологічних галюцинацій. Було обрано інший вектор. Точкове калібрування температури для ідеального балансу між креативністю порад та математичною точністю, паралельно обрізаючи максимальну кількість вихідних токенів. Обов’язково активуються нативні safety-фільтри Google.

Архітектура скрипта передбачає глибоке ешелонування обробки винятків, миттєво ідентифікуючи HTTP-статуси 429 (Too Many Requests) чи раптові збої авторизації.

Якщо базовий кластер не відповідає, система не викидає користувача з діалогу. Замість цього швидко активується резервний алгоритм callGeminiAPIWithFallback. Він починає послідовно відправляти запити до альтернативних моделей Gemini із заздалегідь підготовленого пулу. І ніяких втрат для UX. Клієнт навіть і не знає, що у бекенді відбуваються складні обчислювальні процеси. Згодом, відпрацьований JSON-пакет із готовою реплікою ШІ та оновленими лімітами тихо повертається в інтерфейс покупця.

Окремої уваги заслуговує питання валідації якості роботи ШІ-моделі. Розроблена раніше комплексна методика оцінки ефективності інтелектуальних асистентів в електронній комерції [23] доводить, що економічна доцільність таких систем напряму залежить від якості вхідних інструкцій (prompt engineering). Тому архітектура модуля, відповідно до офіційної документації Gemini API [24] та досліджень патернів промптингу [25], передбачає жорстке обмеження контексту та ролі асистента для уникнення фактологічних помилок.

3.11 Проєктування та програмна реалізація логістичного модуля на базі API нової пошти

Логістика сувора до друкарських помилок. Довірити клієнту ручне введення адреси вільного формату – це свідомо приректи палтформу на втрати через нескінченні повернення посилок. Саме через це, архітектура модуля доставки працює як єдиний жорсткий конвеєр: від першої три літери в полі пошуку до генерації фінальної товарно-транспортної накладної (ТТН).

Фронтенд бере на себе роль первинного санітарного фільтра. Скрипт NovaPoshtaAutocomplete через сучасний інтерфейс fetch асинхронно опитує сервери перевізника [26, 27] (Нова-пошта). На цьому етапі значно оптимізувано процес. Контролер CheckoutAutocomplete керується інакшою логікою. Він використовує механізм дебаунсингу із затримкою у 400 мілісекунд. Мережа не перевантажується. Запити летять лише тоді, коли людина призупинила набір тексту.

Ієрархія вибору прописана лаконічно та чітко. Можливість вказати відділення розблоковується виключно після того, як система без проблем та помилок ідентифікує місто та збереже його індивідуальний Ref-ключ у прихованому дата-атрибуті city_ref. Спроба надіслати форму з простим текстом замість валідних UUID-ідентифікаторів буде жорстко блокована алгоритмом валідації. Візуально цей процес супроводжується динамічними випадаючими списками з підтримкою клавіатурної навігації (accessibility), що наочно продемонстровано на рисунках А.31 та А.32.

Взаємодія із зовнішніми логістичними сервісами реалізується через автономний клас NovaPoshtaAPI. Його основним завданням є інкапсуляція бекенд-операцій із системою нової пошти. Для запобігання блокуванню корпоративного облікового запису внаслідок надмірної кількості паралельних запитів впроваджено алгоритм waitForApiLimit. Даний метод забезпечує балансування навантаження шляхом генерації програмних затримок (мікропауз) із паралельним логуванням кожного етапу виконання.

Оскільки система здійснює безперервне логування статусів та запитів до зовнішніх API, накопичений масив технічної інформації фактично формує часові ряди даних. У перспективі це створює необхідну базу для впровадження алгоритмів онлайн-діагностики та прогнозування збоїв [28]. Такий підхід дозволить системі в майбутньому автоматично розпізнавати аномалії в навантаженні та запобігати падінню сервера. Наступним кроком обробки інформації є виклик методу createRecipient, який здійснює пошук контрагента в базі логістичного оператора за ідентифікаторами повного імені (ПІБ) та населеного пункту. За умови відсутності попередніх замовлень система автоматично реєструє нового контрагента зі статусом «Приватна особа».

Паралельно сервер нормалізує хаос вхідних даних: телефонні номери жорстко очищаються від спецсимволів, ПІБ атомізується на складові, а математичний блок самостійно вираховує об’ємно-вагові характеристики поточного кошика.

Завершальним етапом цього інженерного ансамблю стає формування складного JSON-пакета для методу save моделі InternetDocument. Приватний обробник sendRequest пакує все це у захищену cURL-сесію з контролем таймаутів. Виключається необхідність ручного введення даних або дублювання інформації. Алгоритм розбирає відповідь сервера, а головне, витягує згенерований номер ТТН разом із розрахованою вартістю доставки і намертво прив’язує їх до транзакції в базі даних. Потім ці ж дані дублюються на вказану користувачем електрону пошту. Людський фактор при підготовці відправлень мінімізовано максимально.

Технічна реалізація логістичного модуля та інтеграція з офіційним API поштового оператора [27] є фундаментальним кроком для сучасного ритейлу. Проведений аналіз впливу інтелектуальних помічників на операційну ефективність [29] підтверджує, що автоматизація рутинних процесів (таких як генерація ТТН та консультування клієнтів) створює синергетичний ефект: вона не лише знижує витрати підприємства, але й радикально покращує

клієнтський досвід. Разом із дотриманням стандартів безпеки OWASP [30] при передачі даних, це перетворює платформу на відмовостійкий конвеєр.

ВИСНОВКИ

У результаті виконання роботи розроблено та впроваджено архітектуру вебплатформи електронної комерції Shop4ik Store. Заплановані інженерні та дослідницькі цілі досягнуто в повному обсязі. Підсумком роботи став працездатний програмний комплекс, де серверна частина на базі PHP координує асинхронну взаємодію з клієнтським JavaScript-інтерфейсом. Надійна робота платформи спирається на пряму інтеграцію з хмарними сервісами через прикладні інтерфейси (API), що перетворює локальний вебмаркет на масштабовану бізнес-екосистему.

Практичний запуск підтвердив ефективність обраного технологічного стека. Реляційна база даних MySQL стабільно оперує великими масивами інформації, надійно ізолюючи персональні дані покупців та номенклатуру товарів від несанкціонованого доступу. Під час проєктування головний фокус було зміщено на архітектурну гнучкість. Модульний підхід дозволив автономізувати обробку кошика, керування візуальними темами та роботу адміністративної панелі. Як наслідок, система отримала високий рівень відмовостійкості. Подальше розширення каталогу чи модифікація бізнес-логіки не вимагатимуть структурної перебудови системного ядра.

Головним технологічним нововведенням стала інтеграція великої мовної моделі через Gemini API. Це рішення радикально трансформувало класичну модель клієнтської підтримки. Спроєктована гібридна система маршрутизації чату автономно закриває до 80% типових інформаційних запитів, звільняючи менеджерів від монотонної роботи. Завдяки алгоритмам обробки природної мови користувачі миттєво отримують персоналізовані стилістичні рекомендації безпосередньо під час сесії. Впровадження ШІ-асистента суттєво підвищило рівень довіри до платформи, забезпечивши стабільну конвертацію звичайних переглядів сторінок у реальні транзакції.

Інтеграція логістичного модуля на базі API «Нової Пошти» дозволила повністю нівелювати вплив людського фактора під час оформлення відправлень. Помилки при ручному введенні адрес виключені алгоритмічно. Імплементована система автодоповнення та автоматичної генерації товарно-транспортних накладних (ТТН) скоротила час обробки замовлення до кількох секунд. Для запобігання надлишковому навантаженню на сервери успішно застосовано механізми дебаунсингу та суворої клієнтської валідації. Це безпосередньо підвищило операційну швидкість бізнесу та прискорило цикл доставки.

Візуальна складова платформи повністю відповідає сучасним стандартам мобільної адаптивності. Єдина дизайн-система, побудована на кастомних CSS-змінних, об’єднує клієнтську вітрину та бек-офіс у цілісний графічний простір. Оптимізація медіаконтенту та безшовна зміна тем гарантують високу плавність рендерингу на смартфонах. Отримані результати мають підтверджену практичну цінність. Розроблений програмний комплекс є готовим інструментом для діджиталізації підприємств роздрібної торгівлі, здатним забезпечити стійкі конкурентні переваги на динамічному ринку електронної комерції.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Шапошник, М. Є., & Кобилін, . М. (2025). Оцінка ефективності впровадження інтелектуальних помічників у системи електронної комерції: методологія та практичні результати. *Журнал ХНУПС*. [У друці].

2. Weizenbaum, J. (1966). ELIZA–a computer program for the study of natural language communication between man and machine. Communications of the ACM, 9(1), 36–45. https://doi.org/10.1145/365153.365168

3. Netflix's personalization powerhouse: How A/B testing at scale built a $300B streaming giant. URL: https://medium.com/@productbrief/netflixs-personalization-powerhouse-how-a-b-testing-at-scale-built-a-300b-streaming-giant-f26804e0d92c

4. Netflix Content Recommendation System – Product Analytics Case Study. URL: https://hellopm.co/netflix-content-recommendation-system-product-analytics-case-study/

5. The state of AI in 2023: Generative AI's breakout year. URL: https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai-in-2023-generative-ais-breakout-year

6. Optimize web core vitals: CSS content-visibility and GPU acceleration. URL: https://web.dev/vitals/

7. Turing, A. M. (1950). Computing machinery and intelligence. Mind, 59(236), 433–460. https://doi.org/10.1093/mind/lix.236.433

8. Боденчук-Пастухов, Є. В., & Кобилін, І. О. (2026). Оцінка моделей трансформерів машинного перекладу на низько-ресурсних, азійсько-українських мовних парах. Системи обробки інформації, 1(184), 116–123.

9. Gartner hype cycle for artificial intelligence, 2023. URL: https://www.gartner.com/en/documents/4660899

10. How Intercom achieves 86% resolution rates with Claude AI. URL: https://www.anthropic.com/customers/intercom

11. Daradkeh, Y. I., & Tvoroshenko, I. (2020). Technologies for Making Reliable Decisions on a Variety of Effective Factors using Fuzzy Logic. International Journal of Advanced Computer Science and Applications, 11(5).

12. Bodyanskiy Y., Vynokurova O., Kobylin, I., & Kobylin O. (2016). Adaptive fuzzy clustering of short time series with unevenly distributed observations in Data Stream Mining tasks. Information Technology and Management Science, 23–28.

13. Bodyanskiy, Y., Vynokurova, O., Szymański, Z., Kobylin, I., & Kobylin, O. (2016). Adaptive robust models for identification of nonstationary systems in data stream mining tasks. In 2016 IEEE First International Conference on Data Stream Mining & Processing (DSMP) (pp. 263–268). IEEE.

14. Бодянський, Є., Винокурова, О., Кобилін, І., & Мулеса, П. (2017). Адаптивна матрична нейро-фаззі самоорганізовна мережа для кластеризації багатовимірних потоків даних. Вісник Національного університету «Львівська політехніка». Комп'ютерні науки та інформаційні технології, (864), 314–319.

15. Бодянський, Є. В., Винокурова, О. А., Ізонін, І. В., Кобилін, І. О., & Мулеса, П. П. (2017). Кластеризація багатовимірних часових рядів на основі адаптивної матричної нейро-фаззі самоорганізовної мережі. Intellectual Systems For Decision Making and Problems of Computational Intelligence, 247.

16. PHP 8: Hypertext preprocessor documentation. URL: https://www.php.net/docs.php

17. MySQL 8.0 reference manual. URL: https://dev.mysql.com/doc/refman/8.0/en/

18. Кобилін, І. О., & Харченко, А. І. (2024). Classification techniques for computer vision. Системи обробки інформації, 3(178), 33–41.

19. Suprun, A., Tvoroshenko, I., Gorokhovatskyi, V., & Yakovleva, O. (2025). Development and research of a method for the combined use of large language models for text generation. International Journal of Academic and Applied Research, 9(10), 249–263.

20. Tvoroshenko, I., & Bielinskyi, Y. (2021). To the question of analysis of existing mechanisms of web application testing. In *Proceedings of the I International Scientific and Practical Conference "Problems of Modern Science and Practice"* (pp. 403–408). International Science Group. https://isg-konf.com/problems-of-modern-science-and-practice/

21. Kharchenko, A. I., & Kobylin, I. O. Performance analysis of support vector machine for vehicle classification. Ministry of Education and Science of Ukraine V. N. Karazin Kharkiv National University, 101.

22. Шапошник, М. Є., & Кобилін, І. О. (2025). Оптимізація процесу онлайн-замовлення як детермінанта успіху в електронній комерції. В *IT-простір сьогодення: тенденції, інновації та перспективи розвитку: збірник тез доповідей II Міжнародної науково-практичної студентської конференції* (с. 253–255). https://ekhnuir.karazin.ua/items/616ed73d-24cf-4327-a988-dcd100a94afe

23. Шапошник, М. Є., & Кобилін, І. О. (2025). Розробка комплексної методики оцінки ефективності інтелектуальних асистентів у електронній комерції. В *XXX Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті»* (Т. 7, с. 178–180). https://nure.ua/konferencii-ta-workshops/mizhnarodnij-molodizhnij-forum-radioelektronika-i-molod-u-hhi-stolitti/xxx-mizhnarodnyj-molodizhnyj-forum-radioelektronika-i-molod-u-khkhi-stolitti

24. Gemini API documentation: Capabilities and prompt engineering. URL: https://ai.google.dev/docs

25. White, J., Fu, Q., Hays, S., Sandborn, M., Olenick, C., & Schmidt, D. C. (2023). A prompt pattern catalog to enhance prompt engineering with ChatGPT and LLMs. arXiv. https://doi.org/10.48550/arXiv.2302.11382

26. Using Fetch API and asynchronous JavaScript. URL: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API

27. Документація API для розробників: Інтеграція логістичних процесів. URL: https://developers.novaposhta.ua/

28. Кобилін, І. О., & Ніколайчук, А. І. (2024). MONITORING AND DIAGNOSING FAULTS IN ONLINE MODE USING TIME SERIES DATA. Системи обробки інформації, 3(178), 27–32.

29. Шапошник, М. Є., & Кобилін, І. О. (2025). Аналіз впливу інтелектуальних помічників на операційну ефективність та клієнтський досвід: результати експериментального дослідження. В *Матеріали IV студентської науково-практичної конференції «Прикладні комп'ютерні науки»* (с. 116–118). https://itstep.edu.ua/scientific-measures#gsc.tab=0

30. OWASP top 10: Web application security risks and countermeasures. URL: https://owasp.org/www-project-top-ten/