МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ

ННІ Економіки та бізнес освіти

Кафедра Економіки та цифрового бізнесу

Спеціальність Комп'ютерні науки

Форма навчання Денна

КВАЛІФІКАЦІЙНА РОБОТА

Мироненка Тимура Ігоровича
(прізвище, ім'я, по батькові здобувача)

на тему «Проєктування та розробка онлайн-платформи для продажу товарів»
(повна назва теми)

за матеріалами _____

(повна назва бази дослідження)

науковий керівник - _____ Бєлінський А.О.
(наук. ступінь, вчене звання) *(підпис)* *(прізвище, ініціали)*

Робота допущена до захисту в ЕК
Протокол засідання кафедри
від 09.06 20 25 р. №12

Завідувач кафедри _____
(підпис)
к.е.н., доцент Радько В.М.
Наук. ступінь, вчене звання *Ініціали, прізвище*

Кривий Ріг – 2025

ЗАТВЕРДЖЕНО
Наказ Міністерства освіти і науки,
молоді та спорту України
29 березня 2012 року № 384

Форма № Н-9.01

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЕКОНОМІКИ ТА БІЗНЕС-ОСВІТИ
(повне найменування вищого навчального закладу)

Кафедра Економіки та цифрового бізнесу
Освітній ступінь Бакалавр
Спеціальність Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри _____ В.М. Радько

"07" квітня 2025 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА ЗДОБУВАЧУ

Мироненку Тимуру Ігоровичу

1. Тема роботи «Проєктування та розробка онлайн-платформи для продажу товарів»

науковий керівник роботи Бєлінський Андрій Олександрович
затверджені наказом вищого навчального закладу від «04» квітня 2025 р. № 224-ст (д/ф)

2. Строк подання здобувачем роботи 31.05.2025р.

3. Зміст кваліфікаційної роботи бакалавра, об'єкт, предмет та мета дослідження:

Розділ 1 Теоретичні аспекти веб-розробки та електронної комерції

Розділ 2 Вибір засобів реалізації та проєктування електронного магазину

Розділ 3 Реалізація та тестування електронного магазину

Об'єкт дослідження – процеси електронної комерції та функціонування онлайн торгових платформ
Предмет дослідження – процеси, методи та інструменти розробки онлайн платформи для продажу товарів з використанням сучасних рішень.
Мета кваліфікаційної роботи бакалавра – спроєктувати та розробити веб-додаток – онлайн-платформу для продажу товарів, яка забезпечує зручну взаємодію між продавцем і покупцем, а також ефективне управління товарами, замовленнями та користувачами
4. Дата видачі завдання 04.04.2025р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Відмітка керівника про виконання етапів (дата, підпис)
1	Підготовка розділу 1	до 28.04.2025р.	25.04.2025
2	Підготовка розділу 2	до 16.05.2025р.	15.05.2025
3	Підготовка розділу 3	до 30.05.2025р.	29.05.2025
4	Реєстрація завершеної дипломної роботи	до 31.05.2025р.	30.05.2025
5	Отримання відгуку від наукового керівника	03-04.06.2025р.	04.06.2025
6	Отримання зовнішньої рецензії	05-06.06.2025р.	06.06.2025
7	Перевірка кваліфікаційної роботи на плагіат	02-09.06.2025р.	04.06.2025
8	Попередній захист кваліфікаційної роботи на кафедрі	03.06.2025р.	03.06.2025
9	Допуск кафедрою кваліфікаційної роботи до захисту	09.06.2025р.	09.06.2025
10	Підготовка студента до захисту в ЕК	до 17.06.2025р.	17.06.2025

Завдання підготував науковий керівник _____ Бєлінський А.О.

Завдання одержав здобувач _____ Мироненко Т.І.

(підпис) (прізвище та ініціали)

РЕФЕРАТ

Кваліфікаційна робота бакалавра: 71 стор., 19 рисунків, 12 джерел, 4 додатка.

Об’єкт дослідження: процеси електронної комерції та функціонування онлайн-торгових платформ.

Мета: спроєктувати та розробити веб-додаток – онлайн-платформу для продажу товарів, яка забезпечує зручну взаємодію між продавцем і покупцем, а також ефективне управління товарами, замовленнями та користувачами.

У результаті дослідження було проаналізовано особливості існуючих рішень в галузі електронної комерції, визначено функціональні та нефункціональні вимоги до онлайн-платформи, сформовано архітектуру системи з урахуванням принципів масштабованості та безпеки. реалізовано функціонал для управління товарами, категоріями, обробки зображень та виконання CRUD-операцій. Також інтегровано систему автентифікації та авторизації користувачів. Онлайн-платформа має адаптивний дизайн та зручну адміністративну панель.

Область застосування: результати цієї роботи можуть бути використані для створення онлайн-магазинів, торгових майданчиків малого та середнього бізнесу, а також у навчальних цілях для ознайомлення з принципами побудови повноцінних веб-застосунків у сфері e-commerce.

Ключові слова: ЕЛЕКТРОННА КОМЕРЦІЯ, ВЕБ-ДОДАТОК, ОНЛАЙН-ПЛАТФОРМА, БАЗА ДАНИХ, ІНТЕРФЕЙС КОРИСТУВАЧА, ІНТЕРНЕТ-МАГАЗИН, АРХІТЕКТУРА ВЕБ-СИСТЕМИ.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

API – Application Programming Interface – Інтерфейс прикладного програмування

CRUD – Create, Read, Update, Delete – Створення, читання, оновлення, видалення

HTTP – HyperText Transfer Protocol – Протокол передачі гіпертексту

JSON – JavaScript Object Notation – Текстовий формат обміну даними

ORM – Object-Relational Mapping – Об’єктно-реляційне відображення

CSS – Cascading Style Sheets

DOM – Document Object Model

ER – Entity-Relationship

HTML – HyperText Markup Language

SQL – Structured Query Language – Мова структурованих запитів

UI – User Interface – Користувацький інтерфейс

UX – User Experience – Досвід користувача

URL – Uniform Resource Locator – Уніфікований локатор ресурсу

JWT – JSON Web Token – Веб-токен у форматі JSON

SPA – Single Page Application – Односторінковий застосунок

ВСТУП

У сучасних умовах інформаційного суспільства інформаційні технології відіграють визначальну роль у трансформації бізнес-процесів, а електронна комерція поступово стає ключовим інструментом у сфері купівлі-продажу товарів і послуг. Завдяки стрімкому розвитку інтернету онлайн-торгівля забезпечує глобальний доступ до продукції, незалежно від місця розташування користувачів, що суттєво розширює можливості як для споживачів, так і для бізнесу.

Актуальність теми обумовлена динамічним зростанням цифрової економіки, що вимагає від компаній швидкої адаптації до змін ринку та впровадження інноваційних технологій у свої бізнес-моделі. Веб-додатки в електронній комерції дозволяють автоматизувати взаємодію з клієнтами, підвищити ефективність управління товарними запасами та оптимізувати процеси продажу.

Метою даної роботи є розробка електронного магазину з використанням сучасних веб-технологій. У межах дослідження проаналізовано специфіку електронної комерції, вивчено існуючі рішення, а також обґрунтовано вибір технологій для реалізації клієнтської та серверної частин системи, а також структури бази даних.

Для досягнення поставленої мети визначено такі завдання дослідження:

- проаналізувати тенденції розвитку електронної комерції та перспективи її подальшого зростання;
- дослідити сучасні веб-технології, які використовуються у створенні онлайн-магазинів;
- здійснити порівняльний аналіз засобів реалізації клієнтської та серверної частини веб-додатків;
- спроєктувати архітектуру веб-застосунку та структуру бази даних;
- реалізувати основні функціональні компоненти системи та провести їх тестування.

Об’єктом дослідження є процес створення веб-додатку для електронної комерції.

Предметом дослідження виступають технології та методи розробки, які забезпечують побудову ефективної, функціонально повної та зручної платформи для онлайн-продажів.

У процесі виконання роботи було використано такі методи дослідження, як: аналіз наукових джерел і практичних кейсів, моделювання, проєктування баз даних, а також програмування клієнтської та серверної частин веб-додатку. Інформаційною базою дослідження слугували наукові публікації, технічна документація, відкриті інструкції та приклади реалізації реальних інтернет-магазинів.

Результатом дослідження є розроблений електронний магазин, який може бути використаний як готове рішення для комерційної діяльності або слугувати основою для подальшого розширення функціональності в межах конкретних бізнес-потреб.

РОЗДІЛ 1
ТЕОРЕТИЧНІ АСПЕКТИ ВЕБ-РОЗРОБКИ ТА ЕЛЕКТРОННОЇ КОМЕРЦІЇ

1.1. Розвиток електронної комерції: історія, тренди, перспективи

Розвиток електронної комерції бере свій початок приблизно з кінця 1970-х років, коли було впроваджено системи електронного обміну даними (EDI), які дозволили компаніям передавати комерційну інформацію, таку як замовлення та рахунки-фактури, в електронному вигляді. Це стало першим кроком до автоматизації бізнес-процесів між підприємствами. Важливою віхою також стала поява перших онлайн-опцій оплати, які заклали основу для подальшого розвитку інтернет-торгівлі.

Справжній прорив у сфері електронної комерції стався у 1994 році зі створенням онлайн-платформи Amazon, яка спочатку продавала лише книги, але з часом перетворилася на глобальний маркетплейс. Amazon стала своєрідним прототипом сучасних інтернет-магазинів, що використовують масштабовані платформи, автоматизовані склади та інноваційні підходи до доставки.

У 1995 році з’явився eBay – перший великий онлайн-аукціон, який дозволив користувачам самостійно виставляти товари на продаж і брати участь у торгах. Це стало новим етапом в еволюції електронної комерції, оскільки вперше покупці та продавці могли безпосередньо взаємодіяти між собою через інтернет.

На початку 2000-х років електронна комерція зазнала стрімкого зростання. Це було зумовлено збільшенням кількості користувачів Інтернету, розвитком широкосмугового доступу, вдосконаленням систем електронних платежів (PayPal, кредитні картки), а також зростанням довіри споживачів до онлайн-покупок. У цей період з’явилися численні інтернет-магазини, що спеціалізувалися на продажу різних товарів – від електроніки до одягу та продуктів харчування.

Сьогодні електронна комерція є одним із найдинамічніших секторів світової економіки. Онлайн-магазини розвиваються з вражаючою швидкістю, інтегруючи новітні технології, такі як штучний інтелект, персоналізація покупок, чат-боти, віртуальна реальність для віртуальних примірок товарів, а також мобільні додатки для зручного шопінгу. Загальний прибуток e-commerce у світі продовжує зростати більш ніж у два рази швидше, ніж у традиційних фізичних магазинах [1]. Інформацію про зростання обсягів продажу за останні роки та прогноз на найближче майбутнє можна побачити на рис. 1.1.

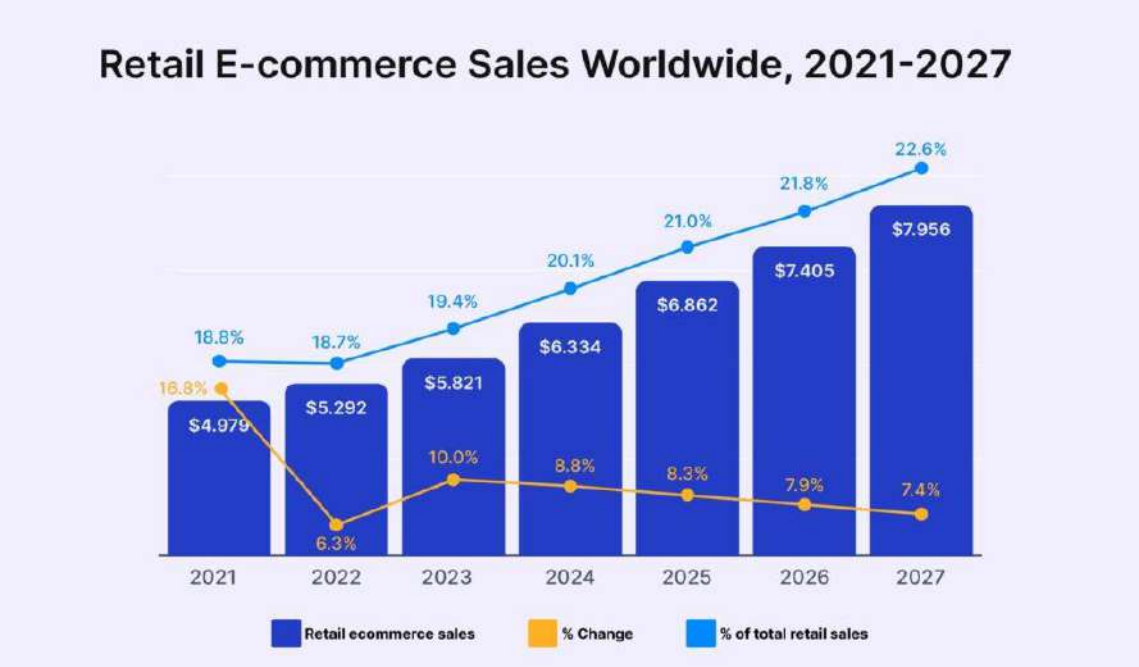


Рис. 1.1. Зростання роздрібних продажів електронної комерції в усьому світі

Примітка. Джерело: розроблено з використанням [1]

Електронна комерція сьогодні є одним із важливих елементів економіки країн, і з розвитком технологій з’являються нові тенденції. Однією з найактуальніших тенденцій є мобільна комерція або m-комерція, яка дозволяє споживачам здійснювати покупки через смартфони й планшети. Мобільна комерція залишається ключовим фактором зростання українського ринку e-commerce. У 2023 році понад 60% онлайн-покупок було здійснено зі смартфонів,

і ця тенденція лише посилюється завдяки поширенню мобільного інтернету та бюджетних смартфонів. Українські споживачі все частіше обирають покупки через мобільні додатки та мобільні версії вебсайтів. Також зростає популярність соціальної комерції, особливо продажів через соціальні мережі, такі як Instagram або Facebook, які реалізують функції покупок на своїх платформах [2].

Нова, але дуже потужна тенденція пов’язана з персоналізацією реклами та акцентує увагу на сегментації користувачів та їхніх потреб. У цьому контексті, завдяки використанню великих даних (big data) і штучного інтелекту, електронні рітейлери налаштовують пропозиції згідно з уподобаннями кожного користувача.

Аналізуючи стан ринку e-commerce в Україні, варто звернути увагу не лише на його зростання та перспективи, а й на виклики, з якими стикається галузь. Незважаючи на активний розвиток онлайн-торгівлі, існує низка факторів, що можуть гальмувати її подальше зростання та впливати на довіру споживачів. Серед основних проблем [3]:

- Безпека та довіра споживачів: Попри зростання популярності онлайн-шопінгу, українці все ще з обережністю ставляться до покупок в інтернеті. Основні побоювання пов’язані з ризиками шахрайства та захисту персональних даних. Незважаючи на покращення у сфері кібербезпеки, випадки платіжних афер і витоків інформації залишаються поширеними. Для підтримки довіри користувачів e-commerce компанії мають активно впроваджувати сучасні методи шифрування та автоматизовані системи виявлення шахрайських операцій.
- Війна та її вплив на ринок: Воєнні дії продовжують створювати труднощі для розвитку електронної комерції, особливо в регіонах, що зазнали руйнувань або перебувають під загрозою атак. Логістичні проблеми, перебої з енергопостачанням та нестабільність постачальницьких ланцюгів ускладнюють ведення бізнесу. Водночас компанії демонструють

гнучкість, адаптуючи свої процеси, перебудовуючи логістичні маршрути та відкриваючи нові склади у відносно безпечних регіонах.

- Нестабільне регулювання ринку: Законодавча база електронної комерції в Україні досі не дотягує до стандартів розвинених країн. Попри зусилля щодо вдосконалення правил захисту споживачів, обробки персональних даних та регулювання міжнародної торгівлі, ці процеси просуваються повільно. Чіткі та зрозумілі законодавчі норми сприятимуть стабільному розвитку ринку, а також підвищать рівень довіри користувачів.
- Посилення конкуренції та домінування маркетплейсів: Український ринок e-commerce стає дедалі більш насиченим, оскільки як локальні, так і міжнародні компанії активно борються за аудиторію. Великі бренди відкривають власні маркетплейси, витісняючи менші проєкти та перетягуючи клієнтів на свої платформи. Частка маркетплейсів у секторі e-commerce вже досягла 60% і продовжує зростати, що змушує дрібніші онлайн-магазини шукати нові стратегії для залучення покупців.
- Інфраструктурні виклики: Окрім військових факторів, складнощі створюють загальні інфраструктурні проблеми – нестабільність інтернет-з’єднання у віддалених регіонах, висока вартість доставки в невеликі населені пункти та дефіцит кваліфікованих спеціалістів у сфері логістики та IT. Інвестування в покращення логістичних маршрутів та розвиток цифрової інфраструктури допоможе мінімізувати ці проблеми.
- Зміна споживчих звичок: В умовах економічної нестабільності українці стають більш вибагливими до цінової політики та шукають вигідніші пропозиції. Знижки, кешбек-програми, швидка доставка та якісний сервіс стають ключовими факторами вибору магазину. Це змушує e-commerce компанії впроваджувати більш персоналізовані маркетингові стратегії та покращувати клієнтський досвід.

Попри всі виклики, e-commerce в Україні залишається однією з найбільш перспективних галузей, яка продовжує активно зростати. Технологічний

розвиток, адаптація бізнесів до нових умов і зміна споживчих звичок стимулюють удосконалення сервісів, покращення логістики та впровадження сучасних рішень у сфері безпеки й платежів. З огляду на динаміку ринку та зростання довіри до онлайн-покупок, електронна комерція в Україні має всі передумови для подальшого стрімкого розвитку, відкриваючи нові можливості для бізнесу та споживачів.

1.2. Значення веб-технологій у розвитку електронної комерції

Сучасний розвиток електронної комерції нерозривно пов’язаний із еволюцією веб-технологій. У цифрову епоху, коли більшість покупок здійснюється онлайн, саме веб-технології забезпечують технічну основу для створення, підтримки та вдосконалення онлайн-магазинів, маркетплейсів, торгових платформ і цифрових маркетингових інструментів. Вони виступають ключовим чинником, що визначає не лише зовнішній вигляд та функціональність електронної платформи, але й її надійність, масштабованість, здатність до інтеграції з іншими сервісами та безпеку користувацьких даних. Саме завдяки цим технологіям електронна комерція змогла перетворитися з допоміжного каналу продажу на повноцінну галузь глобальної економіки.

Веб-технології визначають рівень зручності, швидкодії та безпеки онлайн-торгівлі, що безпосередньо впливає на користувацький досвід і довіру покупців до інтернет-продажів. Сучасний споживач очікує максимально швидкого завантаження сторінки, зручного інтерфейсу, різноманітних способів оплати, можливості відстеження замовлення в реальному часі, а також ефективної взаємодії з технічною підтримкою. Усі ці очікування реалізуються саме за допомогою прогресивних технологічних рішень.

Основні веб-технології, що сприяють розвитку електронної комерції:

- HTML, CSS і JavaScript: Основними технологіями для створення інтерфейсів веб-магазинів є HTML (HyperText Markup Language), CSS (Cascading Style Sheets) та JavaScript. HTML визначає структуру сторінки, CSS забезпечує

стильове оформлення, а JavaScript додає інтерактивність, що сприяє зручності користування сайтом.

- Серверні технології та бази даних: Веб-сервери та технології серверного програмування (Node.js, PHP, Python, Ruby, Java) відповідають за обробку запитів, управління даними та виконання бізнес-логіки. Бази даних (MySQL, PostgreSQL, MongoDB, Firebase) дозволяють зберігати інформацію про товари, замовлення та користувачів, забезпечуючи ефективне керування даними.
- CMS та фреймворки для електронної комерції: Готові платформи для розробки інтернет-магазинів, такі як Shopify, WooCommerce, Magento, OpenCart, значно спрощують створення та управління онлайн-торгівлею. Для індивідуальної розробки використовують фреймворки, наприклад, Next.js, React, Angular або Vue.js, що підвищує продуктивність і розширюваність веб-додатків.
- Технології безпеки: Успішна електронна комерція неможлива без захисту особистих і платіжних даних користувачів. HTTPS-протокол, SSL/TLS сертифікати, методи аутентифікації (OAuth, JWT), а також сучасні засоби шифрування гарантують безпечні онлайн-транзакції та збереження довіри клієнтів.
- Інтеграція платіжних систем: Веб-технології дозволяють реалізовувати різноманітні способи оплати через платіжні шлюзи (Stripe, PayPal, LiqPay, Fondy). Інтеграція API платіжних систем забезпечує зручність і швидкість здійснення покупок.
- Хмарні технології та масштабованість: Використання хмарних обчислень (AWS, Google Cloud, Azure) дозволяє масштабувати онлайн-платформи, автоматизувати резервне копіювання даних та зменшити витрати на інфраструктуру.
- Штучний інтелект і машинне навчання: Алгоритми штучного інтелекту допомагають аналізувати поведінку покупців, персоналізувати контент,

оптимізувати пошукові запити та автоматизувати підтримку клієнтів через чат-боти та голосових асистентів.

Впровадження сучасних веб-технологій сприяє зростанню ринку електронної комерції, підвищенню конкуренції між онлайн-магазинами та покращенню досвіду користувачів. Висока швидкість завантаження сторінок, адаптивний дизайн, інтерактивні можливості та персоналізовані пропозиції стають ключовими факторами успішності онлайн-бізнесу. Крім того, сучасні веб-технології забезпечують гнучкість у масштабуванні проєктів – від малих інтернет-магазинів до великих торгових платформ з мільйонами користувачів.

Завдяки інноваціям у веб-розробці електронна комерція продовжує динамічно розвиватися, відкриваючи нові можливості для підприємців, підвищуючи ефективність торгівлі та вдосконалюючи якість обслуговування клієнтів. Це не лише стимулює економічний розвиток, а й формує нові моделі споживання, де комфорт, швидкість і безпека стають пріоритетами. У найближчому майбутньому можна очікувати ще більшої інтеграції з технологіями доповненої реальності, блокчейном, IoT та іншими інноваційними рішеннями, що ще більше розширить можливості електронної комерції й зміцнить її позиції у світовій економіці.

1.3. Аналіз існуючих рішень

Розробка сучасного онлайн-магазину потребує ретельно спланованого підходу до вибору архітектури, технологічного стеку та функціонального наповнення. Одним із важливих етапів цього процесу є аналіз існуючих рішень, що дозволяє виявити найкращі практики, оцінити ефективність різних підходів та визначити потенційні недоліки, яких доцільно уникати на етапі реалізації власного проєкту.

Дослідження популярних платформ електронної комерції дає змогу з’ясувати, які функції є найбільш затребуваними серед користувачів, які UX/UI-рішення сприяють покращенню зручності користування, а також які технології

забезпечують високу продуктивність, стабільність та масштабованість системи. Крім того, аналіз конкурентного середовища дає змогу виявити унікальні можливості для вдосконалення майбутнього проєкту, що є запорукою формування конкурентоспроможного та технологічно ефективного веб-додатку.

У цьому підрозділі здійснюється огляд трьох найбільших інтернет-магазинів в Україні станом на сьогодні: Rozetka, Comfy та Allo.

Rozetka – один із найбільших і найвідоміших українських інтернет-магазинів, який пропонує широкий асортимент товарів, включаючи електроніку, побутову техніку, товари для дому, одяг, косметику, продукти харчування тощо. Завдяки продуманому дизайну та функціональним можливостям, платформа забезпечує зручний процес покупки для користувачів з різним рівнем цифрової грамотності.

Сайт підтримує ефективну систему пошуку з фільтрацією за параметрами, зручну навігацію, функцію порівняння товарів, а також особисті кабінети, які дозволяють зберігати обрані товари, відстежувати замовлення та швидко здійснювати повторні покупки.

Однією з ключових особливостей Rozetka є інтеграція маркетплейсу, що дає змогу стороннім продавцям розміщувати свої товари на платформі, суттєво розширюючи асортимент. Крім веб-версії, існує мобільний застосунок, адаптований до різних пристроїв, який забезпечує швидкий доступ до каталогу та оформлення замовлень у кілька кліків.

Сервіс доставки включає кілька варіантів: кур’єрська доставка, самовивіз із точок видачі та доставка через партнерські поштомати.

Основні переваги Rozetka:

1. Інтуїтивно зрозумілий інтерфейс – навігація спрощує пошук товарів.
2. Потужна система фільтрації – користувач може сортувати товари за брендом, ціною, рейтингом тощо.
3. Можливість порівняння товарів – допомагає у прийнятті раціонального рішення.
4. Система відгуків і рейтингів – забезпечує прозорість і довіру до продавця.

5. Адаптивність – сайт і застосунок оптимізовані для всіх типів пристроїв.
6. Формат маркетплейсу – можливість купівлі товарів як від Rozetka, так і від партнерських продавців.
7. Гнучкі варіанти доставки – вибір зручного способу отримання замовлення.

Недоліки Rozetka:

1. Перевантаження системи в періоди акцій – можливі затримки з обробкою замовлень.
2. Якість товарів на маркетплейсі – не всі сторонні продавці дотримуються стандартів якості.
3. Надмірна кількість супутніх товарів і рекламних блоків – може відволікати користувача.
4. Обмежений доступ до міжнародної доставки – більшість товарів доступна лише для українського ринку.

Rozetka заслужено вважається технологічно розвиненою платформою вітчизняного ринку e-commerce, яка поєднує зручність, функціональність та широкий вибір продукції. За загальним обсягом трафіку сайт значно випереджає своїх конкурентів, що свідчить про високу довіру з боку споживачів.

Зразок головної сторінки Rozetka представлено на рис. 1.2.

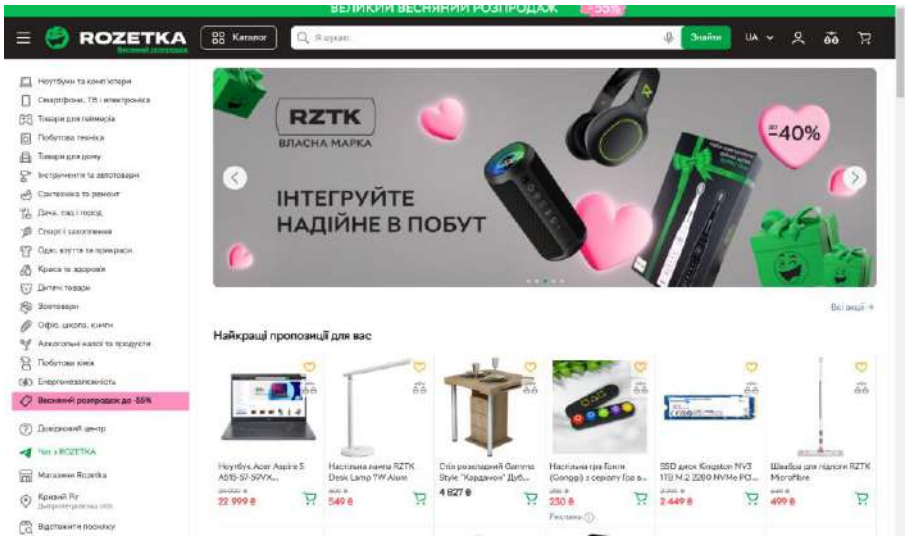


Рис. 1.2. Головна сторінка rozetka.com.ua

Примітка. Джерело: розроблено автором

Allo – український інтернет-магазин, який спеціалізується на реалізації смартфонів, електроніки, смарт-гаджетів та побутової техніки. У процесі розвитку платформа поступово розширює свій асортимент, включаючи товари для дому, особистої гігієни та краси. Основна конкурентна перевага ресурсу полягає в фокусі на мобільні пристрої та інноваційні електронні продукти, зокрема новинки від провідних світових брендів.

Інтерфейс магазину представлений на рис. 1.3. Він характеризується сучасним дизайном та орієнтацією на мобільного користувача, що відповідає основному напрямку діяльності платформи.

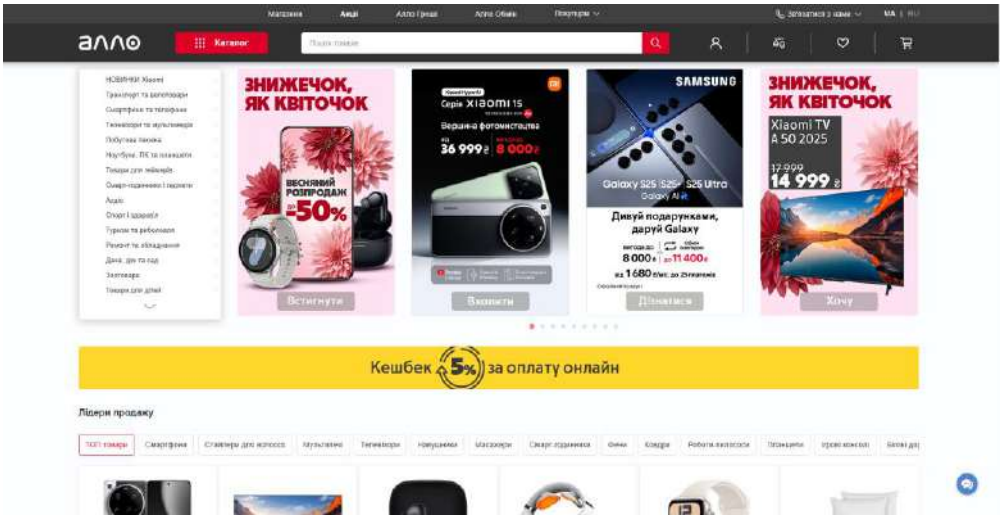


Рис. 1.3. Головна сторінка allo.ua

Примітка. Джерело: розроблено автором

Переваги Allo:

1. Широкий асортимент мобільної техніки та гаджетів – один із найкращих виборів смартфонів в Україні, включаючи останні моделі та популярні бренди.
2. Вигідні фінансові умови покупки – підтримка покупки в розстрочку, акційні програми, спеціальні цінові пропозиції.

3. Ексклюзивні товари та передзамовлення – можливість раннього доступу до нових пристроїв, що підвищує привабливість платформи серед технологічно обізнаних користувачів.
4. Програма лояльності – система кешбеку, накопичення бонусних балів та індивідуальні пропозиції, що сприяють утриманню постійних клієнтів.

COMFY — один із провідних українських ритейлерів, що спеціалізується на продажу електроніки та побутової техніки, поєднуючи онлайн- та офлайн-формати торгівлі. Асортимент компанії охоплює широкий спектр продукції, зокрема смартфони, ноутбуки, телевізори, дрібну та велику побутову техніку, а також різноманітні аксесуари.

Однією з ключових стратегічних переваг COMFY є розвиток омніканальної моделі обслуговування, яка дозволяє покупцям обирати зручний спосіб взаємодії з брендом – через мережу фізичних магазинів або через онлайн-платформу. Такий підхід забезпечує високий рівень клієнтського досвіду та сприяє гнучкому обслуговуванню різних категорій споживачів.

Інтерфейс інтернет-магазину COMFY представлено на рис. 1.4. Він вирізняється простотою навігації, адаптивністю до різних пристроїв та інтуїтивно зрозумілою структурою, що дозволяє швидко знайти необхідний товар.

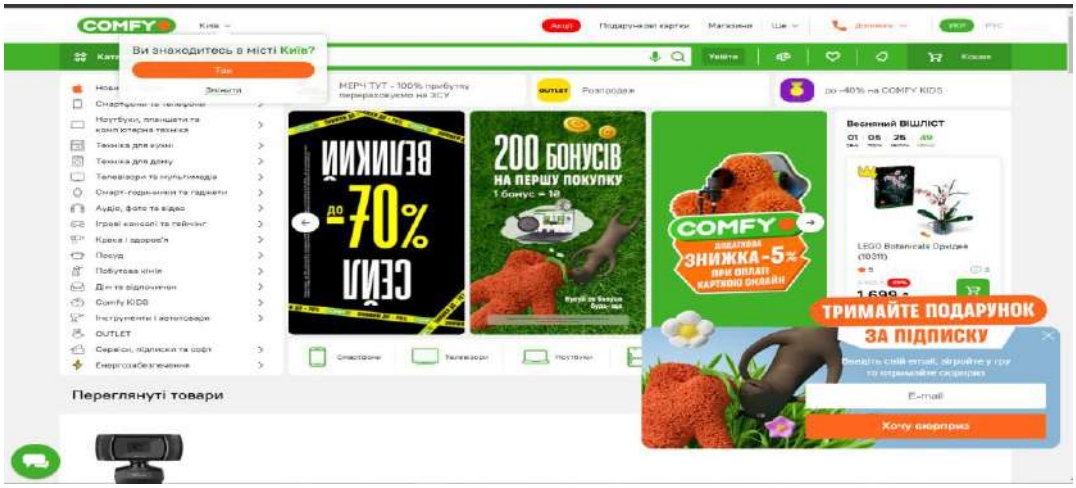


Рис. 1.4. Головна сторінка comfy.ua

Примітка. Джерело: розроблено автором

COMFY демонструє стійке позиціонування на ринку завдяки поєднанню традиційної та цифрової моделей торгівлі. Основними перевагами платформи є:

1. Омніканальний формат взаємодії – інтеграція інтернет-магазину з мережею фізичних точок продажу забезпечує гнучкість вибору каналу покупки та оперативність отримання товарів.
2. Інноваційність – впровадження сучасних цифрових технологій, таких як інтерактивні зони у магазинах, електронні каталоги, інтелектуальні рекомендаційні системи.
3. Функціональний онлайн-магазин – зручний вебсайт та мобільний застосунок з широкими можливостями фільтрації, пошуку та персоналізації інтерфейсу.
4. Гнучкі фінансові інструменти – доступність купівлі в кредит або розстрочку, участь у програмах лояльності з бонусними балами та кешбеком.
5. Різноманітні опції доставки – можливість вибору між самовивозом, стандартною кур’єрською доставкою та експрес-доставкою, що забезпечує комфорт та адаптацію до потреб клієнта.

COMFY акцентує увагу на впровадженні технологічних рішень, що покращують користувацький досвід, роблячи покупки швидкими, інформативними та персоналізованими.

Процес вибору онлайн-магазину з боку споживача залежить від комплексу факторів, серед яких найважливішими є економічна доцільність та користувацький комфорт. Споживачі орієнтуються на:

- доступність цін;
- акційні пропозиції;
- можливість придбання товарів у кредит чи розстрочку на вигідних умовах;
- вартість та швидкість доставки.

Підвищена конкуренція між e-commerce платформами стимулює впровадження знижок, бонусів та персоналізованих програм лояльності, що формує додаткову цінність для кінцевого користувача.

Водночас зручність користування веб-ресурсом відіграє не менш важливу роль. Інтуїтивно зрозумілий інтерфейс, наявність мобільної версії сайту, зручна навігація, розширена система пошуку та фільтрації – усе це значною мірою впливає на якість користувацького досвіду. Додатковими факторами виступають:

- швидкий доступ до детальної інформації про товар;
- наявність перевірених відгуків користувачів;
- можливість оформлення замовлення в декілька кліків.

Таким чином, успішна e-commerce платформа має не лише забезпечити конкурентні ціни, а й створити максимально комфортне середовище для здійснення покупок, що сприятиме лояльності клієнтів та їхній готовності повторно використовувати сервіс у майбутньому.

Висновки до розділу 1

У першому розділі було здійснено комплексний аналіз теоретичних засад електронної комерції та сучасних підходів до веб-розробки, що дозволило сформувати цілісне уявлення про тенденції та технологічні аспекти розвитку галузі. Розглянуто історичну еволюцію e-commerce – від базових електронних транзакцій до складних цифрових екосистем, які поєднують персоналізоване обслуговування, широкі функціональні можливості та зручність для кінцевого користувача.

Особливу увагу приділено трансформації традиційної роздрібної торгівлі, яка активно переходить до омніканальних моделей, забезпечуючи присутність як в онлайн-, так і в офлайн-просторах. Такий підхід дозволяє брендам

підтримувати тіснішу взаємодію з клієнтами та оперативно реагувати на їхні потреби.

У ході дослідження було виявлено низку сучасних трендів, які визначають подальший розвиток електронної комерції, зокрема:

- активне використання мобільних пристроїв для здійснення покупок;
- автоматизація процесів і впровадження елементів штучного інтелекту для підвищення персоналізації;
- висока швидкість технологічних змін, що потребує від компаній гнучкості, масштабованості та інноваційного підходу до розробки рішень.

Окрему увагу приділено значенню веб-технологій у сфері e-commerce. Було проаналізовано роль адаптивного дизайну, хмарних рішень, API-інтеграцій та прогресивних веб-додатків (PWA), що забезпечують зручність, продуктивність і безпеку користування онлайн-платформами. Також розглянуто сучасні практики DevOps і CI/CD, які дозволяють ефективно впроваджувати оновлення, підтримувати стабільну роботу систем і забезпечувати гнучке масштабування під навантаженням.

Кібербезпека окреслена як критично важливий компонент довіри користувачів, що вимагає застосування сучасних методів захисту персональних даних та забезпечення безпеки транзакцій.

У завершальній частині розділу проведено аналіз трьох провідних українських інтернет-магазинів – Rozetka, Comfy та Allo. Оцінено їхні переваги та недоліки, визначено функціональні особливості, що формують користувацький досвід: зручна навігація, потужна система пошуку, інтерфейс мобільних застосунків, персоналізовані пропозиції, гнучкі умови оплати та доставки.

Результати аналізу підтвердили, що успішні e-commerce платформи поєднують високотехнологічні рішення з орієнтацією на користувача, що забезпечує їхню конкурентоспроможність та лояльність аудиторії.

Таким чином, у межах першого розділу було:

- сформовано цілісне бачення розвитку електронної комерції;

- систематизовано ключові технології веб-розробки;
- виокремлено сучасні вимоги до онлайн-платформ, зокрема: адаптивність, безпека, масштабованість, продуктивність і гнучкість налаштувань.

Здобуті знання та результати дослідження слугуватимуть методологічною основою для подальших етапів – проєктування та реалізації власного веб-додатку для електронної комерції.

РОЗДІЛ 2
ВИБІР ЗАСОБІВ РЕАЛІЗАЦІЇ ТА ПРОЄКТУВАННЯ ЕЛЕКТРОННОГО МАГАЗИНУ

2.1. Аналіз технологій для розробки клієнтської частини

Клієнтська частина веб-додатку (frontend) – це частина вебсайту, з якою безпосередньо взаємодіє користувач. Вона відповідає за відображення інформації, логіку користувацького інтерфейсу та обробку введених даних. Головною метою клієнтської частини є забезпечення зручного, швидкого та інтуїтивно зрозумілого користувацького досвіду.

Для створення клієнтської частини веб-застосунку використовуються три основні технології: HTML, CSS і JavaScript.

HTML (HyperText Markup Language) є стандартною мовою розмітки, що визначає структуру та зміст веб-документів. Він використовує систему тегів для формування елементів сторінки, таких як заголовки, абзаци, списки, гіперпосилання, зображення та форми введення. HTML також підтримує атрибути, що дозволяють задавати додаткові параметри елементів, та сучасні API, наприклад, Web Components, які сприяють модульності інтерфейсу.

CSS (Cascading Style Sheets) є мовою стилізації, що відповідає за візуальне представлення веб-документів. Вона визначає зовнішній вигляд елементів HTML шляхом застосування властивостей, таких як колір, шрифт, розташування, відступи, тіні та анімації. Завдяки каскадному принципу та механізму успадкування, CSS дозволяє ефективно керувати стилями та забезпечує гнучкість у розробці інтерфейсу. Сучасні технології, такі як CSS Grid, Flexbox та медіа-запити, дають змогу створювати адаптивні та динамічні макети, що коректно відображаються на різних пристроях.

JavaScript – це мова програмування, яка реалізує динамічну поведінку веб-додатків. Вона підтримує об’єктно-орієнтовану, функціональну та подієву парадигми програмування, що робить її універсальним інструментом для

розробки фронтенду. JavaScript взаємодіє з DOM (Document Object Model), дозволяючи змінювати структуру, вміст і стиль елементів сторінки в реальному часі. Крім того, він забезпечує роботу з асинхронними запитами (AJAX, Fetch API), що значно покращує продуктивність та користувацький досвід. За допомогою сучасних можливостей, таких як ECMAScript-модулі (ES6+), Promise, async/await та Web API (наприклад, Local Storage, WebSockets, Service Workers), JavaScript є основою інтерактивних веб-додатків.

Сучасні веб-сайти та додатки, окрім чистого JavaScript, зазвичай використовують його фреймворки та бібліотеки, що дозволяють швидше впроваджувати функціональність, покращувати продуктивність і забезпечувати зручну організацію коду. Фреймворк – це набір готових компонентів, бібліотек і правил, що полегшують розробку програмного забезпечення, зокрема веб-застосунків. Вони надають розробникам готові рішення для управління компонентами інтерфейсу, маршрутизації, роботи з даними та станом застосунку, що значно прискорює процес розробки. Використання цих інструментів дає змогу мінімізувати рутинний код, підвищити масштабованість проєкту та забезпечити підтримку надійної архітектури. Найпопулярнішими з них є React, Vue.js та Angular.

1. React – це бібліотека для створення інтерфейсів, розроблена Meta (Facebook). Вона використовує концепцію компонентного підходу, що дозволяє створювати масштабовані та повторно використовувані елементи. React має високу продуктивність завдяки віртуальному DOM, що оптимізує процес оновлення сторінки. Крім того, React підтримує Next.js, який забезпечує серверний рендеринг та покращує SEO.

2. Vue.js – це прогресивний фреймворк, який поєднує простоту та потужність. Його головні переваги – легка інтеграція, зручний синтаксис та ефективність у створенні динамічних інтерфейсів. Vue.js добре підходить для невеликих і середніх проєктів, а також підтримує Nuxt.js, що додає можливості серверного рендерингу.

3. Angular – це повноцінний фреймворк від Google, який використовує TypeScript і пропонує вбудовані рішення для роботи з формами, HTTP-запитами, маршрутизацією та керуванням станом. Angular ідеально підходить для великих корпоративних проєктів, але має вищий поріг входу через складну архітектуру.

У комплексних веб-додатках ефективне управління станом відіграє критичну роль. Стан додатка містить усі дані, що впливають на інтерфейс та взаємодію користувача, такі як поточні налаштування, статус авторизації, вміст кошика в інтернет-магазині або отримані з сервера дані. Без належної організації керування станом код може стати неструктурованим і важким для підтримки.

Різні бібліотеки керування станом дозволяють централізовано зберігати дані, синхронізувати їх між компонентами та забезпечувати прогнозованість поведінки додатка. Серед найпоширеніших рішень можна виокремити:

1. Redux – популярна бібліотека керування станом, яка використовує централізоване сховище та принцип односпрямованого потоку даних. Стан змінюється через чисті функції-редуктори, що забезпечує передбачуваність. Redux Toolkit спрощує використання, зменшуючи кількість шаблонного коду.

2. MobX – реактивна бібліотека, яка автоматично оновлює залежні компоненти при зміні стану. Використовує обсервабельні об’єкти, що зменшує необхідність у явному виклику дій, роблячи код простішим. Однак може бути менш передбачуваною у великих проєктах.

3. Zustand – легковагова альтернатива Redux, що пропонує простий API без зайвого коду. Підтримує глобальний стан без складної конфігурації та запобігає зайвим перерендерам. Чудово підходить для невеликих і середніх додатків.

2.2. Аналіз технологій для реалізації серверної частини

Серверна частина веб-застосунку (backend) відповідає за обробку запитів клієнтів, управління бізнес-логікою, взаємодію з базою даних і забезпечення безпеки. Вибір технологій для серверної розробки визначає продуктивність,

масштабованість і гнучкість майбутнього онлайн-магазину. Серверна частина може бути написана за допомогою більшості сучасних мов програмування, серед найпоширеніших:

- JavaScript (Node.js) – одна з найпоширеніших мов для бекенду, що дозволяє використовувати одну мову як на клієнтській, так і на серверній частині. JavaScript добре підходить для високонавантажених асинхронних застосунків, але може поступатися іншим мовам у продуктивності при обробці складних обчислювальних задач.
- Python – універсальна мова, що вирізняється простотою синтаксису та широкими можливостями для роботи з даними, штучним інтелектом та автоматизацією. Python часто використовується для швидкої розробки серверних додатків, проте може бути менш ефективним у порівнянні з компільованими мовами для ресурсомістких операцій.
- Java – потужна мова програмування, яка відзначається високою продуктивністю, надійністю та кросплатформеністю. Вона широко застосовується у великих корпоративних рішеннях, але може мати складніший вхідний поріг для розробників через необхідність писати більш розгорнутий код.
- PHP – одна з найстаріших мов для серверної розробки, що залишається популярною завдяки простоті та гнучкості. PHP добре підходить для створення веб-сайтів і контент-орієнтованих платформ, проте його продуктивність і безпека можуть вимагати додаткових оптимізацій.
- C# – мова програмування від Microsoft, що добре інтегрується з екосистемою .NET. Вона забезпечує високу продуктивність і безпеку, що робить її відмінним вибором для корпоративних рішень та хмарних застосунків, проте її основне використання обмежене платформою Windows та Azure.

При розробці бекенда, окрім мови програмування, часто використовується фреймворк для прискорення роботи та легшої імплементації тих чи інших

функцій. Вибір фреймворку залежить від вибраної мови програмування та потреб проекту. Серед найпопулярніших фреймворків різних мов програмування:

- Express.js – мінімалістичний фреймворк для Node.js, який спрощує створення REST API. Підходить для гнучких рішень та мікросервісної архітектури.
- NestJS – фреймворк на основі TypeScript, який використовує модульний підхід і забезпечує високу підтримуваність коду.
- Django – фреймворк для Python із вбудованими механізмами безпеки та ORM для зручної роботи з базами даних.
- Spring Boot – потужний фреймворк для Java, який дозволяє створювати масштабовані серверні рішення з мікросервісною архітектурою.
- ASP.NET Core – сучасний кросплатформенний фреймворк для C#, що забезпечує високу продуктивність та підтримку контейнеризації.

Більшості сучасних web-додатків необхідно зберігати ті чи інші дані користувачів. Тому для звернення до бази даних необхідні спеціальні бібліотеки. ORM (Object-Relational Mapping) – це технологія, що дозволяє працювати з реляційними базами даних через об’єктно-орієнтовану модель. Вона спрощує взаємодію з базою даних, замінюючи написання SQL-запитів на використання об’єктів і методів у коді. Завдяки ORM розробники можуть маніпулювати даними у базі так, ніби працюють зі звичайними об’єктами в коді, що підвищує зручність і безпеку роботи з базою. Популярні ORM для різних мов програмування:

- Sequelize (JavaScript, Node.js) – ORM для роботи з SQL-базами (PostgreSQL, MySQL, MariaDB, SQLite, MSSQL).
- TypeORM (TypeScript, Node.js) – потужне ORM з підтримкою роботи з класами та декораторами.
- Django ORM (Python) – вбудоване рішення у Django, що дозволяє легко працювати з базами.

- SQLAlchemy (Python) – більш гнучкий варіант ORM, який підтримує як декларативний, так і імперативний стиль.
- Entity Framework (C#) – ORM від Microsoft для роботи з базами даних у середовищі .NET.
- Hibernate (Java) – одна з найпопулярніших ORM для Java, що реалізує JPA (Java Persistence API).

ORM значно спрощує роботу з базами даних, підвищує продуктивність розробки та робить код більш підтримуваним. Однак у деяких випадках використання ORM може мати певні недоліки, зокрема зниження продуктивності при складних запитах, тому у високонавантажених системах інколи доводиться комбінувати ORM із «чистими» SQL-запитами.

2.3. Аналіз технологій для реалізації бази даних

База даних (БД) – це організована сукупність даних, що зберігається в електронному вигляді та керується за допомогою спеціального програмного забезпечення – системи управління базами даних (СУБД). Бази даних відіграють важливу роль у веб-розробці, забезпечуючи зберігання, обробку та отримання інформації, необхідної для роботи онлайн-платформ.

Існує два основних типи баз даних:

- Реляційні бази даних (SQL) зберігають інформацію у вигляді таблиць із чітко визначеними зв’язками між ними. Вони використовують мову SQL (Structured Query Language) для управління даними та забезпечують високу узгодженість, підтримку транзакцій і цілісність даних.
- Нереляційні бази даних (NoSQL) пропонують гнучкіші моделі зберігання, такі як документи, ключ-значення, графи або колонки. Вони підходять для високонавантажених систем і роботи з великими обсягами даних, що динамічно змінюються.

Обидва типи мають свої переваги та використовуються в залежності від специфіки веб-застосунку.

Реляційні бази даних ідеально підходять для веб-застосунків, що потребують високого рівня узгодженості та транзакційної цілісності, наприклад, фінансових систем, інтернет-магазинів, корпоративного програмного забезпечення та CRM-систем. Завдяки підтримці ACID-транзакцій реляційні бази даних гарантують коректність і надійність операцій з даними, що є критично важливим у бізнес-застосунках.

Нереляційні бази даних застосовуються у високонавантажених системах, таких як соціальні мережі, стрімінгові платформи, мобільні додатки та сервіси реального часу (чати, трекери). Документо-орієнтовані БД (наприклад, MongoDB) дозволяють легко зберігати неструктуровані або напівструктуровані дані, що робить їх ідеальними для контент-орієнтованих платформ. Колонкові БД (наприклад, Cassandra) використовуються у великих розподілених системах, оскільки добре масштабуються горизонтально. Бази даних типу «ключ-значення» (наприклад, Redis) забезпечують швидкий доступ до кешованих даних, що критично для продуктивності веб-застосунків.

Таким чином, вибір між SQL і NoSQL базами залежить від архітектури проєкту, потреб у масштабуванні, швидкості доступу до даних і рівня їхньої структурованості. У сучасних системах нерідко комбінують обидва підходи, використовуючи реляційні БД для обробки критично важливих транзакційних даних, а NoSQL — для швидкого зберігання та доступу до динамічного контенту.

Серед популярних SQL та NoSQL баз даних знаходяться:

- MySQL – одна з найпопулярніших реляційних СУБД, яка використовується в численних веб-проєктах, зокрема в CMS (WordPress, Joomla, Drupal). Відзначається високою продуктивністю, підтримкою ACID-транзакцій (InnoDB), масштабованістю та великою спільнотою користувачів, що забезпечує активний розвиток і підтримку.
- PostgreSQL – потужна об’єктно-реляційна СУБД, що забезпечує високу надійність, підтримку складних запитів і можливість роботи з JSON-даними. Використовується у фінансових, наукових та високонавантажених

проєктах завдяки розширеній функціональності, масштабованості та відповідності сучасним вимогам безпеки.

- Microsoft SQL Server – корпоративна СУБД від Microsoft, що забезпечує потужні можливості для аналітики, високий рівень безпеки та інтеграцію з іншими продуктами Microsoft. Використовується у великих комерційних проєктах, зокрема у фінансових системах та корпоративних рішеннях. Основним недоліком є висока вартість ліцензії.
- SQLite – легка, вбудована СУБД, яка не потребує окремого серверного процесу. Ідеально підходить для мобільних застосунків, тестування або невеликих веб-проєктів. Має обмежену підтримку багатокористувацьких режимів, що робить її менш придатною для високонавантажених веб-систем.
- MongoDB – документо-орієнтована NoSQL СУБД, що зберігає дані у форматі JSON-подібних документів. Використовується у динамічних веб-застосунках, які потребують швидкої роботи з неструктурованими даними, таких як каталоги товарів, блоги та соціальні мережі. Забезпечує високу гнучкість та горизонтальне масштабування.
- Redis – NoSQL база даних типу ключ-значення, яка використовується як кеш або для зберігання швидкодоступних даних. Має дуже високу швидкість операцій, тому застосовується для обробки сесій користувачів, зберігання проміжних результатів та інших високонавантажених завдань. Менш підходить для складних реляційних зв’язків між даними.
- Cassandra – розподілена колонкова СУБД, створена для роботи з великими масивами даних у масштабованих системах. Використовується у сервісах із глобальним навантаженням, таких як Facebook, Instagram, Netflix. Відзначається високою стійкістю до збоїв і можливістю горизонтального масштабування без втрати продуктивності.
- Firebase Realtime Database – хмарне рішення від Google для мобільних та веб-застосунків. Забезпечує збереження даних у режимі реального часу,

що робить його ідеальним для чатів, стрімінгових сервісів, онлайн-ігор та інших інтерактивних застосунків. Використання хмарних сервісів може мати свої недоліки, зокрема залежність від провайдера та потенційні витрати на масштабування.

2.4. Обґрунтування вибору засобів реалізації

PostgreSQL здобула чудову репутацію як потужний, багатофункціональний інструмент для реляційних даних. Завдяки своїй стабільності, функціональності і відповідності стандартам, PostgreSQL є ідеальним вибором для багатьох проектів. Також PostgreSQL віда тим, що пропонує чудову реалізацію основних реляційних функцій, не обмежуючись рамками традиційних СУБД. Хоча жодна база даних не може задовольнити всі потреби, PostgreSQL є чудовим варіантом, достатньо універсальним, щоб задовольнити багато випадків використання [4].

Для ефективної роботи з PostgreSQL важливе значення має вибір зручного інструменту керування, найкращим варіантом є pgAdmin – офіційний графічний інтерфейс для адміністрування цієї СУБД. Інтерфейс pgAdmin 4 можна побачити на рис. 2.1.

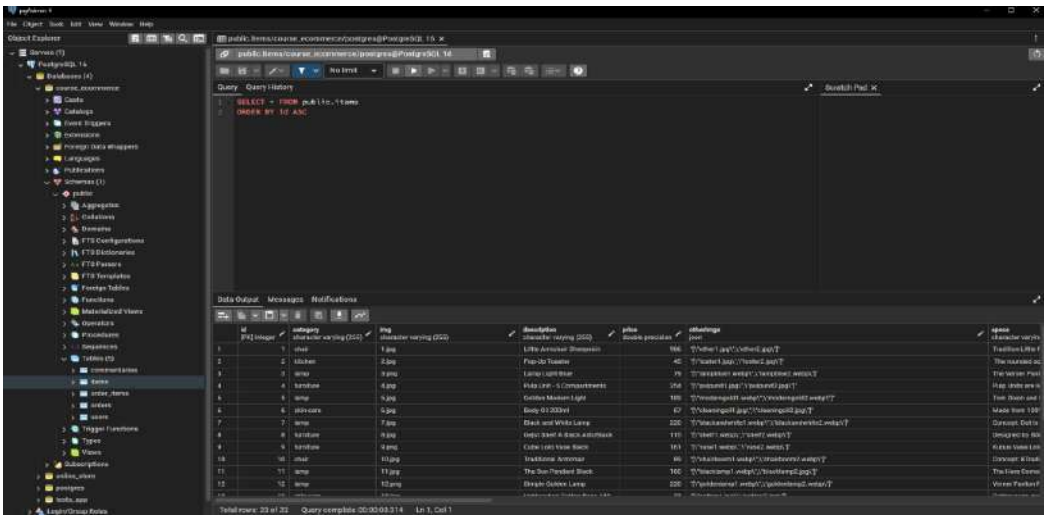


Рис 2.1. Графічний інтерфейс програми pgAdmin

Примітка. Джерело: розроблено автором

У процесі розробки веб-додатку для електронної комерції було використано низку сучасних технологій, що забезпечують високу продуктивність, масштабованість та зручність підтримки системи. Обґрунтований вибір інструментів на всіх рівнях архітектури – від бази даних до інтерфейсу користувача – став запорукою створення ефективного та надійного програмного продукту.

Для роботи з базою даних було обрано pgAdmin – інструмент адміністрування PostgreSQL, який надає широкий набір функцій у візуально зручному та інтуїтивно зрозумілому інтерфейсі. Цей інструмент дозволяє не лише виконувати стандартні SQL-запити, але й забезпечує повноцінний контроль над структурою, продуктивністю та вмістом бази даних, підтримуючи роботу з кількома серверами, розширеннями та моніторингом у реальному часі. Його функціональні можливості є корисними як для новачків, так і для досвідчених фахівців.

Як засіб об’єктно-реляційного відображення (ORM) було обрано Sequelize. Цей фреймворк забезпечує гнучкий об’єктно-орієнтований інтерфейс для роботи з реляційними базами даних, що значно спрощує інтеграцію з базою без потреби у великій кількості SQL-запитів. Sequelize підтримує основні типи зв’язків між таблицями (один-до-одного, один-до-багатьох, багато-до-багатьох), що є критично важливим для коректного моделювання бізнес-логіки електронного магазину — включаючи взаємодію між товарами, замовленнями, користувачами та транзакціями.

На стороні клієнта використано React – популярний JavaScript-фреймворк для створення інтерфейсів користувача. React забезпечує високу швидкодію за рахунок віртуального DOM, а також дозволяє створювати компонентно-орієнтовану структуру, яка легко підтримується та масштабовується. Такий підхід особливо ефективний для реалізації динамічних інтерфейсів, таких як каталог товарів, кошик, особистий кабінет, сторінка замовлень тощо [5].

З метою підвищення надійності коду та мінімізації типових помилок у процесі розробки було використано TypeScript – надбудову над JavaScript, що додає статичну типізацію. Це забезпечує вищу стабільність і прогнозованість поведінки системи, що є критично важливим для складних веб-застосунків [6].

Для стилізації інтерфейсу використано Sass – препроцесор CSS, який підтримує змінні, вкладеність, міксини та інші розширення, що дозволяють ефективно організовувати та повторно використовувати стилі. Це полегшує підтримку масштабних проєктів і сприяє формуванню уніфікованого візуального стилю платформи.

Для централізованого управління станом додатку було інтегровано React Redux, що забезпечує єдине джерело правди для всіх компонентів. Це особливо актуально для таких модулів, як кошик, історія замовлень, особистий кабінет, де важливо зберігати узгодженість даних між різними частинами системи [7].

Для організації навігації між сторінками застосовано React-Router-Dom, що дозволяє реалізувати SPA (Single Page Application) – односторінковий додаток із можливістю динамічного оновлення контенту без повного перезавантаження сторінки. Це забезпечує плавну та зручну навігацію, підвищуючи якість користувацького досвіду.

Крім базового стеку, було використано додаткові бібліотеки, зокрема для реалізації функцій шифрування даних, створення HTTP-запитів, обробки помилок, інтеграції з бекендом та підвищення рівня безпеки. Це розширює функціональність додатку, дозволяючи реалізувати захист персональних даних і підтримку безпечної взаємодії з сервером.

Таким чином, застосований технологічний стек (PostgreSQL, pgAdmin, Sequelize, React, TypeScript, Sass, Redux, React-Router) дозволяє забезпечити:

- надійність і стабільність роботи додатку;
- гнучкість та масштабованість архітектури;
- зручність підтримки і подальшого розширення функціоналу;
- високу якість користувацького досвіду.

Обрані технології повністю відповідають сучасним вимогам до веб-застосунків у сфері електронної комерції та можуть слугувати основою для розгортання повноцінного комерційного продукту.

2.5. Вимоги до електронного магазину: функціональні та нефункціональні

Заздалегідь визначені функціональні та нефункціональні вимоги відіграють ключову роль у комфортній розробці та досягненні необхідного результату. Функціональні потреби визначають основні функції, необхідні для реалізації базових потреб користувачів. З іншого боку, нефункціональні вимоги встановлюють стандарти продуктивності, безпеки, сумісності та інших технічних характеристик, які забезпечують стабільну роботу системи. Чітке формулювання вимог дозволяє створити ефективну архітектуру додатку, уникнути непередбачених проблем під час розробки і зменшити витрати на подальше доопрацювання.

Функціональні вимоги описують основні можливості системи, які вона має реалізовувати. Основні функціональні вимоги до створюваного електронного магазину включають:

а) Реєстрація та авторизація користувачів:
   1) Можливість створення облікового запису користувача.
   2) Авторизація через електронну пошту та пароль.

б) Перегляд каталогу товарів:
   1) Відображення списку товарів із ключовою інформацією (назва, зображення, ціна).
   2) Пошук товарів за категоріями.

в) Кошик покупок:
   1) Додавання товарів до кошика.
   2) Зміна кількості товарів у кошику.

3) Видалення товарів із кошика.

4) Підрахунок загальної суми замовлення.

г) Оформлення замовлення:

1) Заповнення форми з контактною інформацією.

ґ) Адміністрування:

1) Додавання товарів.

д) Перегляд інформації про виконані замовлення

Нефункціональні вимоги визначають технічні та операційні характеристики системи, які впливають на її продуктивність, безпеку та зручність у використанні. Основні нефункціональні вимоги до електронного магазину включають:

а) Продуктивність:

1) Час завантаження сторінок не повинен перевищувати 2-3 секунд.

2) Система повинна підтримувати одночасну роботу великої кількості користувачів (масштабованість).

б) Безпека:

1) Захист облікових записів користувачів через хешування паролів.

2) Захист від атак, таких як SQL-ін’єкції та XSS.

в) Масштабованість:

1) Система повинна легко адаптуватися до збільшення кількості товарів і користувачів.

г) Доступність і сумісність:

1) Адаптивний дизайн для коректного відображення на мобільних і настільних пристроях.

2) Підтримка основних браузерів (Chrome, Firefox, Safari, Edge).

ґ) Зручність використання:

1) Простий і зрозумілий інтерфейс для користувачів усіх рівнів.

2) Легкий доступ до ключових функцій, таких як пошук, кошик, оформлення замовлення.

2.6. Архітектура веб-додатку

Коректно спроєктована архітектура веб-сайту є ключовим чинником досягнення ефективності, масштабованості та зручності супроводу веб-додатків. Вона визначає, яким чином компоненти системи взаємодіють між собою, і формує надійну основу для подальшого розвитку програмного продукту. Продумана архітектура дозволяє уникнути численних проблем, пов’язаних із низькою продуктивністю, складністю підтримки та вразливістю до зовнішніх загроз.

Однією з головних переваг дотримання архітектурних принципів є забезпечення можливості масштабування. У разі зростання кількості користувачів або збільшення обсягів даних, які обробляються системою, архітектура повинна дозволяти безболісне додавання нових серверів, оптимізацію обробки запитів та інтеграцію із зовнішніми сервісами без негативного впливу на функціонування додатку. Це має особливе значення для інтернет-магазинів, у яких в періоди високої активності (наприклад, під час сезонних розпродажів) навантаження на сервери може зростати в рази.

Ще одним важливим аспектом є зручність обслуговування та модернізації. Модульна архітектура дає змогу локалізувати зміни у певних частинах системи - наприклад, оновити інтерфейс користувача або змінити логіку обробки замовлень – без необхідності втручання в інші компоненти додатку. Це значно спрощує процес впровадження нових функціональностей, тестування, усунення помилок і зменшує загальний час розробки.

Крім того, належна архітектура підвищує рівень інформаційної безпеки. Зокрема, чітке розділення між клієнтською та серверною частинами дозволяє ізолювати доступ до бази даних, що мінімізує ризики несанкціонованого втручання з боку користувача.

Для реалізації простого інтернет-магазину доцільно використовувати класичну клієнт-серверну архітектуру, яка є стандартом для більшості сучасних

веб-додатків. Така модель включає три основні рівні [8]: Клієнтський рівень (frontend), Серверний рівень (backend) та Рівень збереження даних (база даних).

Ця архітектура дозволяє ефективно розподіляти навантаження між компонентами, підтримує гнучкість у виборі технологій на кожному рівні та створює умови для подальшого масштабування проєкту. Схематичне зображення описаної архітектурної моделі подано на рис. 2.2.

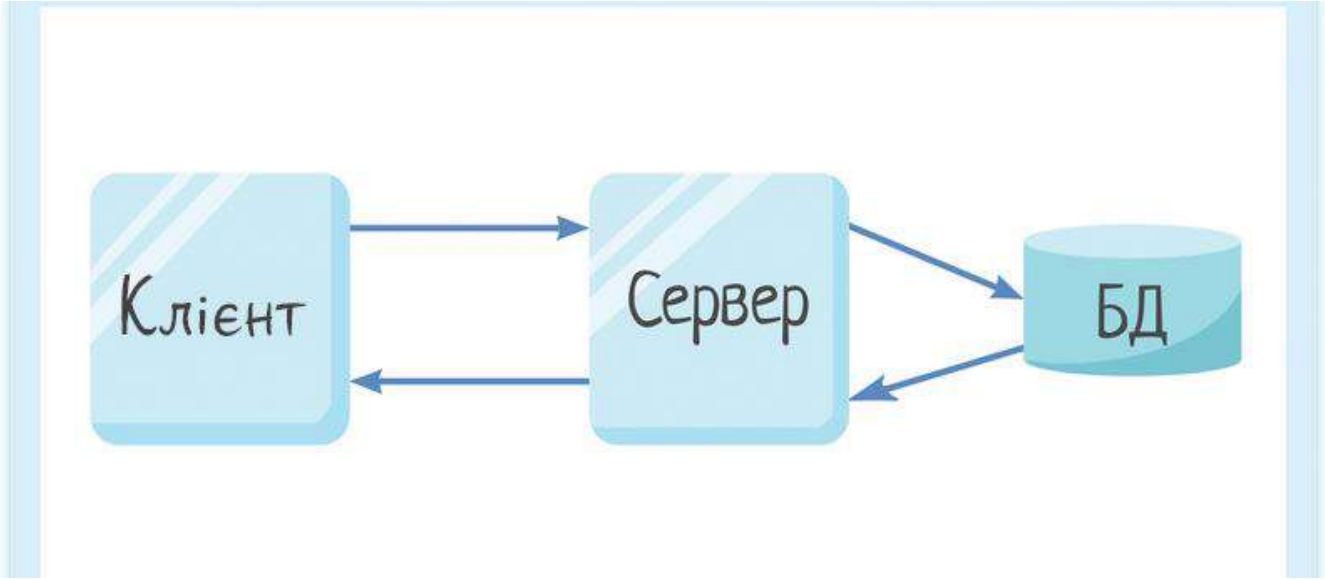


Рис. 2.2. Клієнт-Серверна Архітектура

Примітка. Джерело: розроблено із використанням [8]

Клієнтський рівень: цей рівень представляє інтерфейс користувача програми та відповідає за представлення даних користувачеві та отримання введених даних. Він включає компоненти графічного інтерфейсу користувача (GUI), такі як веб-сторінки, форми та віджети. Цей рівень також може містити логіку валідації на стороні клієнта, що дозволяє перевіряти введені дані до їх надсилання на сервер. Його основна мета – забезпечити зручний та інтуїтивно зрозумілий доступ до функціоналу програми.

Серверний рівень: цей рівень містить бізнес-логіку програми та виконує основну обробку даних. Він відповідає за обробку запитів користувачів, отримання та обробку даних, а також виконання складних операцій. Він включає

такі частини, як контролери, служби та API. Серверний рівень функціонує як посередник між клієнтом і базою даних, забезпечуючи обробку даних відповідно до заданих правил і логіки системи. Він також реалізує механізми безпеки, такі як автентифікація, авторизація та захист від шкідливих запитів.

Рівень даних: цей рівень відповідає за керування зберіганням і пошуком даних у системі. Він включає базу даних або файлову систему, де зберігаються дані, а також рівень доступу до даних, який взаємодіє з базою даних для читання та запису даних. Рівень даних забезпечує цілісність і узгодженість інформації, дозволяє ефективно масштабувати систему та підтримує механізми резервного копіювання для запобігання втраті важливих даних.

У контексті створюваного проекту клієнтський рівень представлений додатком на React, Серверний рівень представлений додатком на Express, а рівень даних представлений базою даних на PostgreSQL. Користувач кінцевого продукту зможе виконувати різні операції за допомогою графічного інтерфейсу та надсилати запити на сервер, де вони будуть оброблятися та виконувати подальші маніпуляції з базою даних

2.7. Опис структури бази даних

База даних – одна з важливіших частин будь-якого веб-додатку. Вся головна інформація зберігається саме в ній, тому важливо витратити час на вибір правильної СУБД та її налаштування під потреби проєкту. Ці кроки дозволять уникнути проблем в майбутньому, коли проєкт стане більше і з’явиться потреба зберігати, зчитувати, видаляти і редагувати великі об’єми даних щоденно. Однією з найкращих СУБД для SQL-баз даних є PostgreSQL, це універсальне і сильне рішення для більшості проєктів. За допомогою серверної частини проєкту, записи будуть вноситися до сховища. Структура бази даних представлена на рис. 2.3, та може бути представлена наступними сутностями:

1. User (Користувач)
2. Order (Замовлення)

3. Order_item (Товар в замовленні)
4. Item (Товар)
5. Commentary (Коментар)

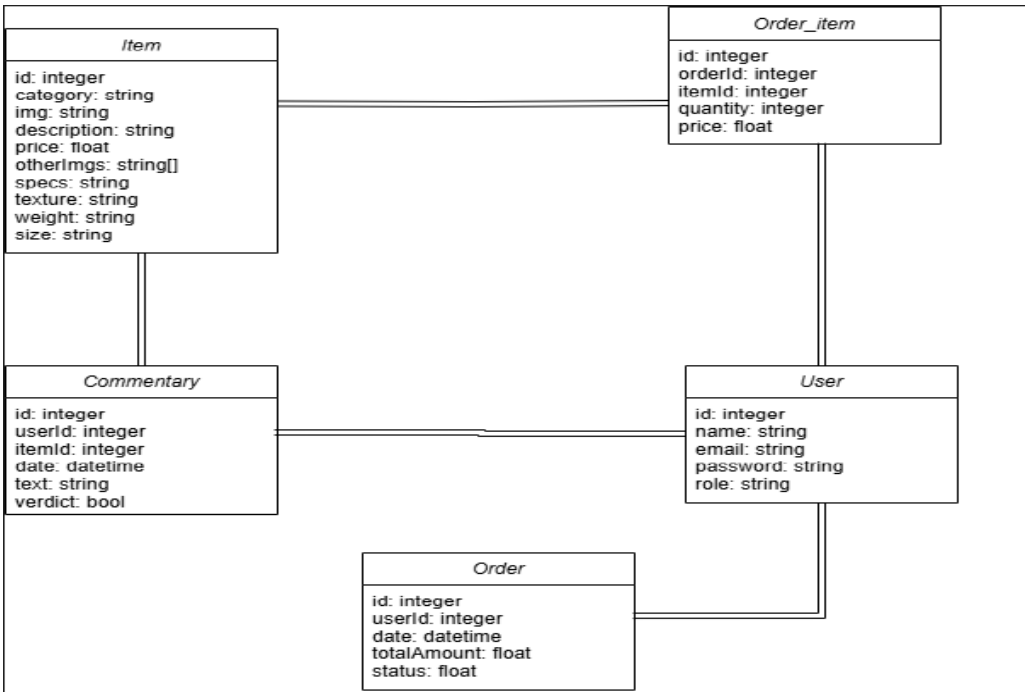


Рис. 2.3. Діаграма структури бази даних

Примітка. Джерело: розроблено автором

Нижче наведено детальний опис таблиць бази даних, їх полів та зв’язків між таблицями.

1. Таблиця User (Користувач)

Опис: Таблиця зберігає інформацію про користувачів системи.

Поля:

- id: Унікальний ідентифікатор користувача.
- email: Електронна пошта користувача.
- password: Пароль для входу.
- role: Роль користувача (наприклад, адміністратор або клієнт, строка).

2. Таблиця Item (Товар)

Опис: Таблиця містить дані про товари, які доступні для покупки.

Поля:

- id: Унікальний ідентифікатор товару.
- category: Категорія товару.
- img: Головне зображення товару.
- description: Опис товару.
- price: Ціна товару.
- otherImgs: Додаткові зображення товару.
- specs: Характеристики товару.
- texture: Текстура товару.
- weight: Вага товару.
- size: Розмір товару.

3. Таблиця Order (Замовлення)

Опис: Таблиця зберігає інформацію про замовлення, оформлені користувачами.

Поля:

- id: Унікальний ідентифікатор замовлення.
- userId: Ідентифікатор користувача, який оформив замовлення.
- date: Дата створення замовлення.
- totalAmount: Загальна сума замовлення.
- status: Статус замовлення.

4. Таблиця Order_item (Позиції замовлення)

Опис: Таблиця зберігає окремі товари, які входять до складу замовлення.

Поля:

- id: Унікальний ідентифікатор позиції.
- orderId: Ідентифікатор замовлення, до якого належить позиція.
- itemId: Ідентифікатор товару.
- quantity: Кількість товару в цій позиції.
- price: Ціна товару на момент покупки.

5. Таблиця Commentary (Коментарі)

Опис: Таблиця містить коментарі користувачів до товарів

Поля:

- id: Унікальний ідентифікатор коментаря.
- userId: Ідентифікатор користувача, який залишив коментар.
- itemId: Ідентифікатор товару, до якого залишено коментар.
- date: Дата створення коментаря.
- text: Текст коментаря.
- verdict: Оцінка товару (може приймати булеві значення)

Зв’язки між таблицями в базі даних:

User ↔ Order:

Зв’язок: «Один до багатьох».

Опис: Один користувач може створити кілька замовлень.

Поле userId у таблиці Order є зовнішнім ключем, що посилається на поле id у таблиці User.

User ↔ Commentary:

Зв’язок: «Один до багатьох».

Опис: Один користувач може мати багато коментарів.

Поле userId у таблиці Commentary є зовнішнім ключем, що посилається на поле id у таблиці User.

Item ↔ Commentary:

Зв’язок: «Один до багатьох».

Опис: Один товар може мати багато коментарів.

Поле itemId у таблиці Commentary є зовнішним ключем, що посилається на поле у таблиці Item.

Order ↔ Order_item:

Зв’язок: «Один до багатьох».

Опис: Одне замовлення може містити кілька товарів.

Поле orderId у таблиці Order_item є зовнішнім ключем, що посилається на поле id у таблиці Order.

Order_item ↔ Item:

Зв’язок: «Багато до одного».

Опис: Кожна позиція замовлення стосується одного товару.

Поле itemId у таблиці Order_item є зовнішнім ключем, що посилається на поле id у таблиці Item.

2.8. Схема навігації та користувацький інтерфейс

Зручність та простота використання є ключовими характеристиками, що визначають успішність електронного магазину. Навіть найвищої якості товари можуть залишитися поза увагою потенційних клієнтів, якщо користувачі стикаються з труднощами при пошуку продукції або оформленні замовлення. Саме тому розробка інтуїтивно зрозумілого та функціонального користувацького інтерфейсу (UI) є одним із пріоритетних завдань у процесі створення онлайн-магазину.

Раціонально побудована навігація дозволяє користувачам легко орієнтуватися в структурі сайту, оперативно знаходити потрібну інформацію та здійснювати цільові дії з мінімальними зусиллями. Така навігація повинна бути логічно структурованою, зрозумілою як новим, так і постійним користувачам, та адаптованою до потреб цільової аудиторії.

Користувацький інтерфейс, як основна точка взаємодії між системою та користувачем, повинен відповідати сучасним стандартам дизайну та враховувати очікування споживачів. Особлива увага приділяється адаптивності інтерфейсу, що забезпечує коректне відображення та зручну навігацію як на настільних комп’ютерах, так і на мобільних пристроях.

У контексті сучасних тенденцій використання мобільного інтернету, наявність адаптивного дизайну є критично важливою. Наводимо декілька статистичних даних [9], що ілюструють актуальність цього аспекту:

- Понад 60% світового вебтрафіку надходить із мобільних пристроїв;
- Існує понад 4,32 мільярда активних користувачів мобільного інтернету;

- У регіонах, таких як Африка, мобільний трафік складає до 69,13% загального обсягу;
- До 2025 року прогнозується понад 1 мільярд з’єднань 5G у світі.

Вебсайти без адаптивного дизайну суттєво знижують якість взаємодії для мобільних користувачів, змушуючи їх масштабувати вміст вручну, що призводить до незручностей. Крім того, пошукові системи негативно ранжують ресурси, які не підтримують адаптивність, що погіршує SEO-позиції сайту.

У рамках даного проєкту користувацький інтерфейс електронного магазину було розроблено з урахуванням принципів простоти, логічності та адаптивності. Його структура включає такі ключові елементи:

- Головна сторінка – виконує функцію вітрини магазину, надаючи користувачу доступ до рекламних банерів, акційних пропозицій, популярних категорій товарів. Візуальне оформлення спрямоване на привернення уваги та сприяє швидкій навігації до інших розділів.
- Сторінка категорій – забезпечує доступ до груп товарів за тематичними напрямами. Вона дозволяє ознайомитися з асортиментом продукції та скористатися зручною фільтрацією для пошуку необхідного товару.
- Сторінка товару – містить детальний опис обраного продукту: зображення з кількох ракурсів, технічні характеристики, ціну, а також кнопку «Додати до кошика», яка забезпечує простоту оформлення замовлення.
- Панель адміністратора – доступна лише для авторизованих користувачів з правами адміністратора. У цьому розділі реалізована можливість додавання та редагування товарів, а також управління вмістом каталогу.
- Навігаційна панель – розташована у верхній частині кожної сторінки та містить посилання на основні розділи: головну сторінку, каталог, панель адміністратора, а також іконку кошика, яка відкриває перелік доданих товарів.

Таким чином, реалізований користувацький інтерфейс орієнтований на максимальну простоту, адаптивність і функціональність, що сприяє формуванню позитивного досвіду взаємодії користувача з онлайн-магазином. Такий підхід

дозволяє залучати нових клієнтів, підвищувати рівень задоволеності наявних користувачів та зміцнювати конкурентні переваги платформи.

Висновки до розділу 2

У другому розділі було здійснено ґрунтовний аналіз технологічного підґрунтя, необхідного для реалізації клієнтської, серверної частин і системи зберігання даних електронного магазину. Основною метою цього етапу стало обґрунтування вибору оптимального технологічного стеку, який забезпечує високу продуктивність, масштабованість, безпеку та зручність використання майбутнього веб-додатку.

Аналіз клієнтських технологій підтвердив доцільність використання базових інструментів – HTML, CSS та JavaScript – у поєднанні з фреймворком React, що дозволяє реалізувати динамічний, компонентно-орієнтований інтерфейс з високими показниками продуктивності. На стороні сервера було обрано Node.js із фреймворком Express, які забезпечують асинхронну обробку запитів, легкість масштабування та гнучкість у налаштуванні логіки взаємодії з клієнтом. Як систему управління базами даних було обґрунтовано використання PostgreSQL завдяки її надійності, підтримці складних зв’язків між сутностями та високому рівню узгодженості даних.

У межах розділу також було визначено функціональні вимоги до системи (реєстрація користувачів, каталог товарів, пошук, оформлення замовлень, фільтрація), а також нефункціональні вимоги, що охоплюють показники надійності, продуктивності, безпеки, адаптивності та кросбраузерної сумісності. Ці вимоги стали основою архітектурного проектування.

Архітектура веб-додатку побудована за багаторівневим принципом, що передбачає чітке розділення клієнтської частини, серверної логіки та бази даних. Такий підхід забезпечує модульність, гнучкість розширення функціоналу, простоту супроводу, а також спрощує процеси тестування та відлагодження.

Структура бази даних розроблена з урахуванням логіки предметної області, що забезпечує оптимальну організацію даних і ефективну роботу із запитами.

Окрема увага була приділена UX/UI-дизайну: розроблено структуру навігації, логіку переходів між сторінками та візуальне оформлення, яке відповідає сучасним принципам юзабіліті. Створений інтерфейс є адаптивним, інтуїтивно зрозумілим та інклюзивним, тобто доступним для користувачів з різними рівнями цифрових навичок і з обмеженими можливостями.

У результаті було сформовано цілісне бачення архітектури веб-додатку, яке охоплює всі ключові елементи системи – від зовнішнього інтерфейсу до внутрішньої логіки та бази даних. Розроблений концепт виступає надійним фундаментом для наступного етапу – безпосередньої реалізації функціонального електронного магазину відповідно до актуальних вимог ринку та очікувань користувачів.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ЕЛЕКТРОННОГО МАГАЗИНУ

3.1. Опис процесу розробки клієнтської частини веб-додатку

Розробка клієнтської частини e-commerce починається з ініціалізації проєкту. Найпоширенішим способом створення React-додатку є команда npx create-react-app [10]. Цей спосіб дає змогу не хвилюватися про створення початкової структури проєкту та налаштування збирача модулів webpack. Розробнику залишається лише встановити необхідні бібліотеки за допомогою менеджера пакетів npm.

Після початкового налаштування проєкту, для зручного розроблення було створено структуру папок і файлів, що можна побачити на рис. 3.1.

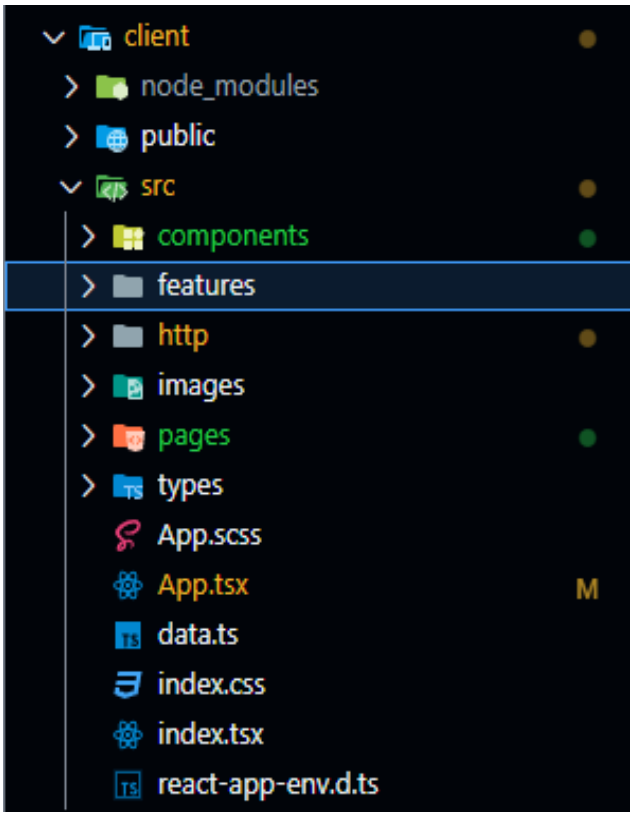


Рис. 3.1. Файлова структура клієнтської частини веб-додатку

Примітка. Джерело: розроблено автором

Папка components містить у собі головні складові будь-якого вебсайту створеного за допомогою react - компоненти. Саме тут знаходяться файли, що відповідають за навігаційну панель, картки товарів, кошик і велику кількість інших складових частин додатка.

Папка types містить у собі typescript типи, що являють собою різні складні дані з певним набором полів, наприклад: користувач, замовлення, список категорій тощо.

Папка pages містить у собі react компоненти, що представляють різні сторінки веб-додатка, описані під час розроблення графічного інтерфейсу та схеми навігації. Крім основних сторінок також було створено сторінку NotFoundPage, що відображатиметься при введенні некоректного url, яке веде на неіснуючу сторінку.

Папка features містить у собі файли, необхідні для коректної роботи Redux, бібліотеки, що дає змогу використовувати необхідні дані в будь-якому react компоненті програми. У розроблюваному e-commerce веб-додатку це дасть змогу використовувати і змінювати дані про кошик покупок і користувача в будь-якому компоненті. Це полегшить state-management і підтримку коду.

Папка http відповідає за запити до сервера. Вона містить у собі функції, що відповідають за розшифрування jwt токенів, їхнє збереження в local storage і отримання даних про замовлення і товари, даючи змогу не хвилюватися про всю цю логіку в інших частинах програми.

Будь-який інтернет-магазин складається із сотень сторінок, тому при створенні клієнтської частини необхідно вирішити проблему маршрутизації. React Router можна використовувати як просту, декларативну бібліотеку маршрутизації. Її завданням буде зіставляти URL із набором компонентів, надавати доступ до URL-даних і здійснювати навігацію в додатку. Ця стратегія популярна для сучасних SPA (односторінкові додатки), які мають власну фронтенд-інфраструктуру.

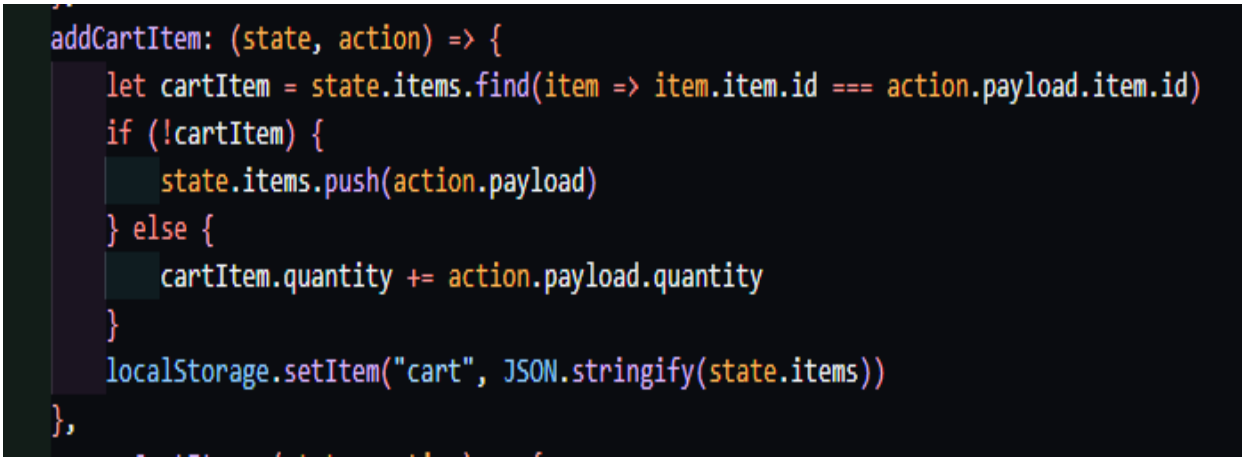
Бібліотека react-router-dom дозволяє створювати комплексні маршрути, що будуть відображатися за певної умови, або мати різні параметри. Наприклад панель адміністратора буде доступна лише користувачам з ролею адміністратора, або сторінка продукту буде змінювати свій вміст в залежності від параметра id. Фрагмент коду, що відповідає за маршрутизацію створюваного додатка, зображений на рис. 3.2.

```
return (
  <div className="app">
    <BrowserRouter>
      <ScrollToTopWrapper>
        <Navbar />
        <Cart />
        <div className="container">
          <Routes>
            <Route index path="/" element={<Home />} />
            <Route path="/categories/:category" element={<Categories />} />
            <Route path='/products/:id' element={<ProductPage />} />
            <Route path='/orders' element={isAuth ? <OrdersPage/> : <Home/>}/>
            <Route path='/admin' element={(isAuth && user.role === "ADMIN") ? <Admin/> : <Home/>} />
            <Route path='*' element={<Home />} />
          </Routes>
        </div>
        <Footer />
      </ScrollToTopWrapper>
    </BrowserRouter>
  </div>
)
```

Рис. 3.2. Фрагмент коду, що відповідає за маршрутизацію

Примітка. Джерело: розроблено автором

Іншою важливою частиною будь-якого e-commerce є кошик. Для зменшення навантаження на базу даних, було вирішено винести зберігання даних про кошик у клієнтську частину програми. Список товарів буде знаходиться в local storage, javascript-об’єкті, що дозволяє зберігати невелику кількість даних локально [11]. Для того щоб впровадити цей підхід у код, необхідно після кожної операції над списком товарів звертатися до local storage і зберігати новий стан кошика. Функція для додавання товарів у кошик зображена на рис 3.3.



```
addCartItem: (state, action) => {
    let cartItem = state.items.find(item => item.item.id === action.payload.item.id)
    if (!cartItem) {
        state.items.push(action.payload)
    } else {
        cartItem.quantity += action.payload.quantity
    }
    localStorage.setItem("cart", JSON.stringify(state.items))
},
```

Рис. 3.3. Фрагмент коду, що відповідає за додавання товарів до кошика

Примітка. Джерело: розроблено автором

Розробка клієнтської частини веб-додатку була реалізована з використанням сучасних інструментів і бібліотек, таких як React, Redux і React Router, що забезпечили ефективність, зручність і масштабованість додатку. Створена структура папок і компонентів дозволяє легко підтримувати та розширювати функціональність системи, а зберігання даних у local storage знижує навантаження на сервер. Впровадження маршрутизації та індивідуальних сторінок забезпечило гнучкість, що є важливою складовою якісного e-commerce рішення.

3.2. Опис процесу розробки серверної частини веб-додатку

Початкове налаштування серверної частини застосунку не надто відрізняється від налаштування клієнтської частини. На початку було встановлено необхідні бібліотеки за допомогою npm. Другим кроком було створення основного файлу серверного додатка – index.js. Надалі саме ця частина програми об’єднає в собі логіку backend частини ecommerce-магазину. Після чого була створена файлова структура, що зображена на рис 3.4.

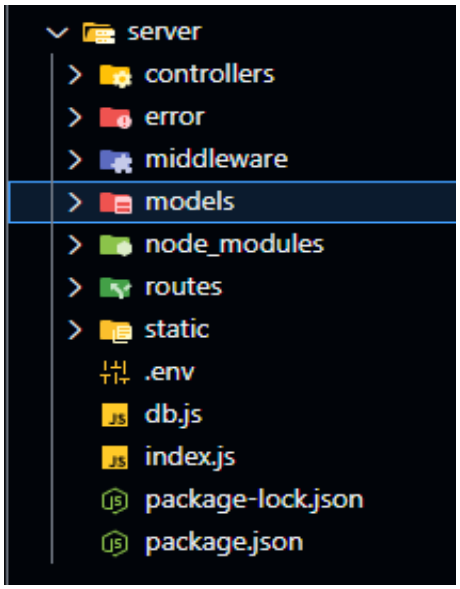


Рис. 3.4. Файлова структура серверної частини веб-додатку

Примітка. Джерело: розроблено автором

Взаємодія frontend з backend відбуватиметься за допомогою надсилання запитів із клієнта на різні кінцеві точки сервера. Наприклад, для отримання конкретного товару необхідно надіслати запит за адресою: «адреса сервера/item/id товару». За реалізацію цієї логіки і відповідають фрагменти коду в папках routes і controllers. routes приймає запити і викликає відповідні функції з controllers, після чого відбувається взаємодія з базою даних і повернення результату на клієнт.

Під час виконання операцій за кінцевими точками API, може знадобитися додаткова робота над даними або обробка виникших помилок, наприклад, перевірити роль користувача перед тим як вносити зміни в базу даних. Для таких проміжних завдань використовуються middleware (сполучне програмне забезпечення). Для цих завдань у файловій структурі є папки error і middleware.

Папка static необхідна для зберігання зображень товарів. Картинки потрапляють до неї після того, як адміністратор створює новий товар. Цей підхід дає змогу іншим користувачам легко отримати доступ до зображень на сайті. Для роботи зі статикою було використано вбудований в express middleware.

Підключення маршрутів, middleware і бази даних відбувається у файлі index.js. Фрагмент коду, що відповідає за запуск сервера, можна побачити на рис. 3.5.

```
const PORT = process.env.PORT || 5000

const app = express()
app.use(cors())
app.use(express.json())
app.use(express.static(path.resolve(__dirname, "static")))
app.use(fileUpload({}))
app.use("/api", router)
app.use(errorHandler)


const start = async () => {
    try {
        await sequelize.authenticate()
        await sequelize.sync()
        app.listen(PORT, ()=> console.log(`Server started on port ${PORT}`))
    } catch(e) {
        console.log(e)
    }
}

start()
```

Рис. 3.5. Фрагмент коду, що відповідає за сервер

Примітка. Джерело: розроблено автором

Спочатку застосовується функція use об’єкта типу Express для підключення middleware. Серед них: cors (для роботи з іншими доменами), json (для роботи з даними у форматі javascript object notation), static (для збереження зображень у вигляді статики), fileUpload (для роботи з файлами, які надсилають із клієнтської частини), підключення маршрутизатора та middleware для обробки помилок.

Після цього викликається функція start для запуску сервера. Вона є асинхронною, оскільки необхідно встановити з’єднання з базою даних за

допомогою бібліотеки sequelize і тільки після цього почати прослуховування запитів за допомогою app.listen.

3.3. Тестування функціоналу та оптимізація веб-додатку

Тестування функціоналу та оптимізація є важливими етапами розробки електронного магазину, оскільки вони гарантують стабільність та ефективність. Ці процеси охоплюють перевірку всіх ключових компонентів додатку, виявлення та усунення помилок, а також покращення продуктивності.

Процес тестування слід розпочати з перевірки основних функцій веб-додатку, включаючи реєстрацію та авторизацію користувачів. Процес введення даних для реєстрації нового користувача можно побачити на рис. 3.6.

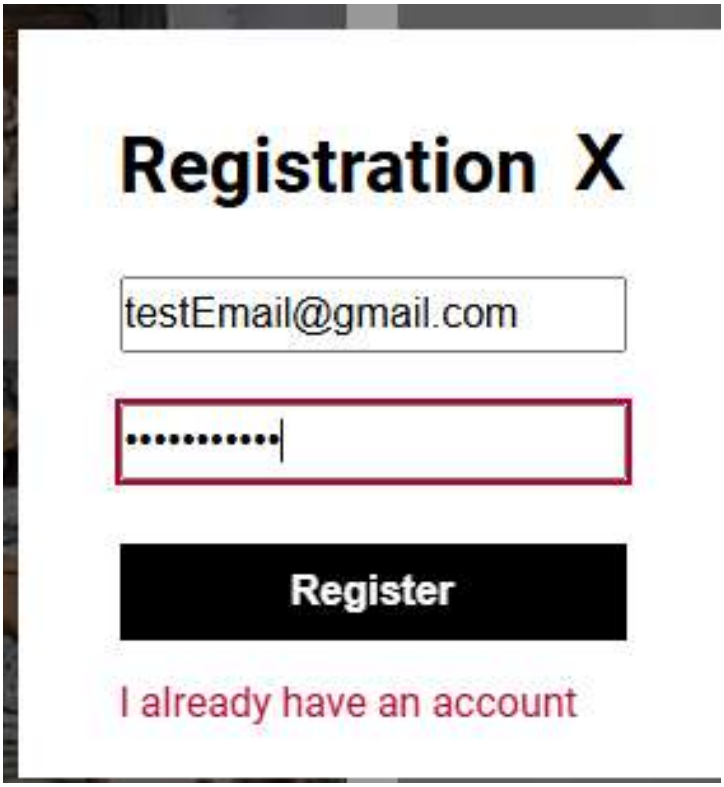


Рис. 3.6. Вікно реєстрації користувача

Примітка. Джерело: розроблено автором

Другою функцією, що була протестована, стала фільтрація категорій товарів. Ця частина додатку дуже важлива для того щоб користувачі могли

швидше знайти необхідні товари. Успішну фільтрацію за категорією ламп можна побачити на рис 3.7.

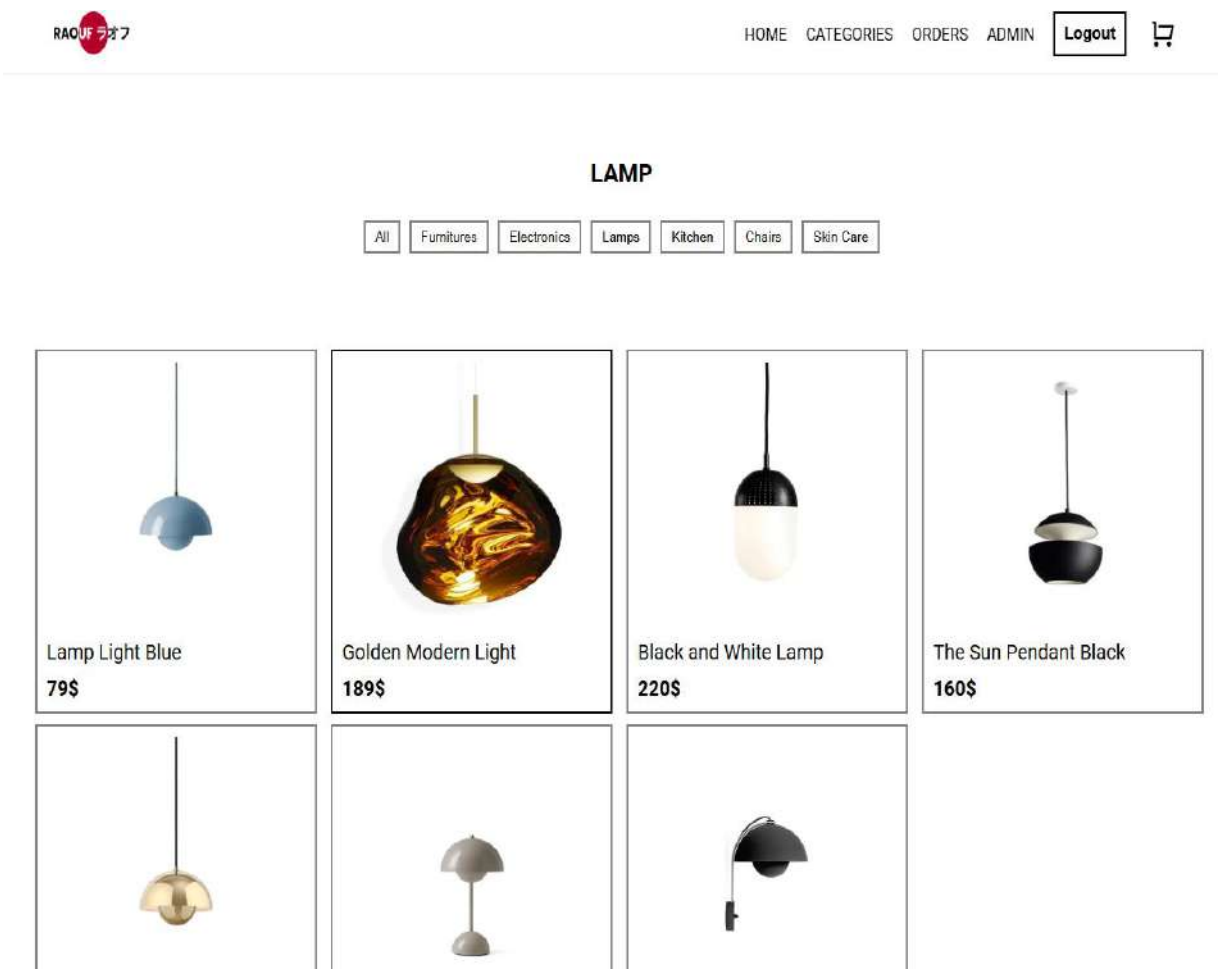


Рис. 3.7. Сторінка категорії з лампами

Примітка. Джерело: розроблено автором

Наступним кроком стане перевірка додавання товарів у кошик. Для цього знадобилося перейти на сторінку необхідного товару, вказати його кількість і натиснути на кнопку додавання в кошик. Успішний результат цих дій можна побачити на рис. 3.8.

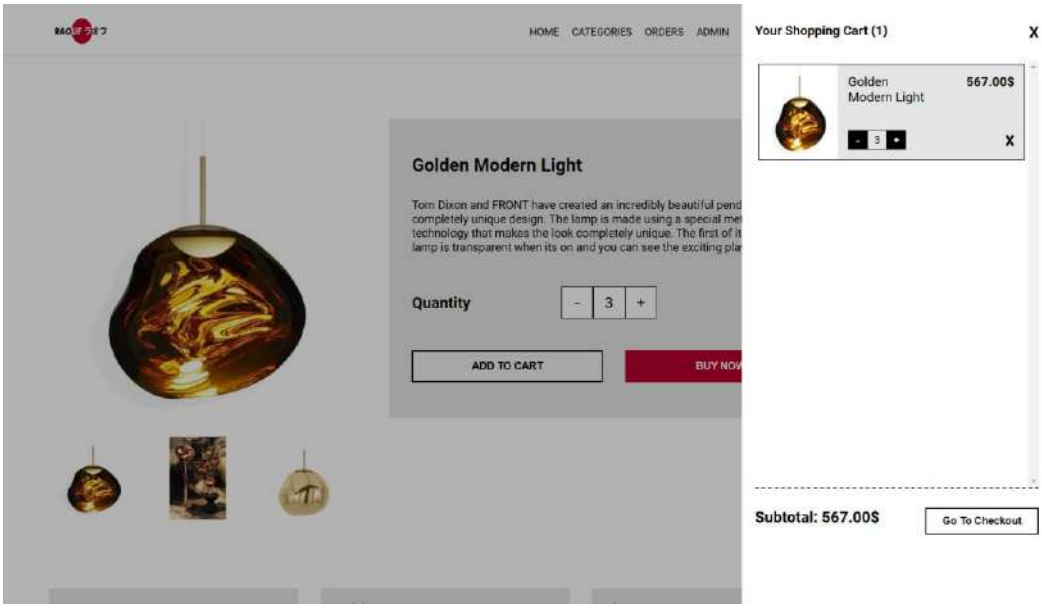


Рис. 3.8. Додавання товару до кошику

Примітка. Джерело: розроблено автором

Далі було перевірено можливість переглядати історію замовлень. Після того як замовлення здійснено, було відкрито сторінку orders. У верхній частині написано пошту користувача, що була вказана під час реєстрації, а нижче можна помітити замовлення з 3 лампами, що було здійснене раніше. Деталі замовлення можна подивитися на рис. 3.9.

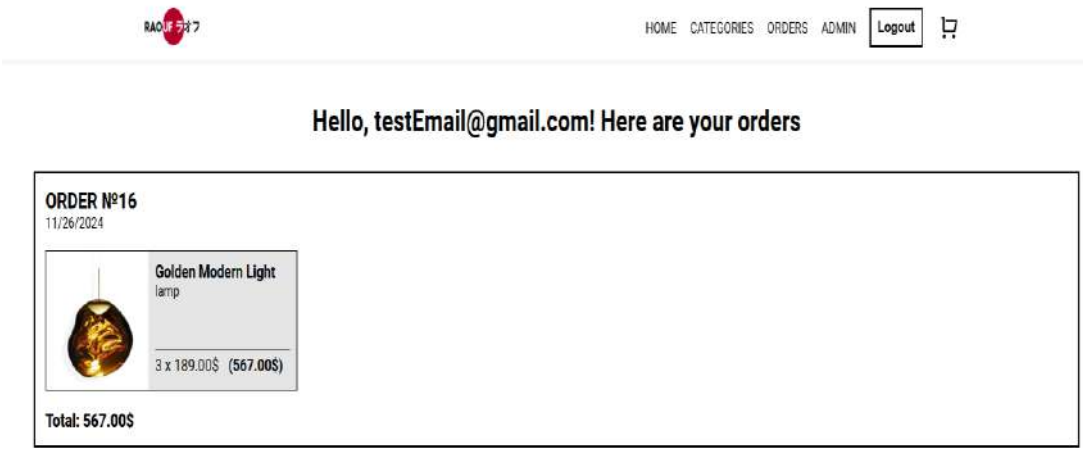


Рис. 3.9. Сторінка з історією замовлень

Примітка. Джерело: розроблено автором

Основні функції, що знадобляться простому клієнту магазину, було перевірено, і вони працюють як було задумано. Наступним кроком буде перевірка можливості адміністратора додавати товари. Для цього необхідно зайти на сторінку admin і заповнити форму для додавання товарів. Процес заповнення форми можна побачити на рис 3.10.

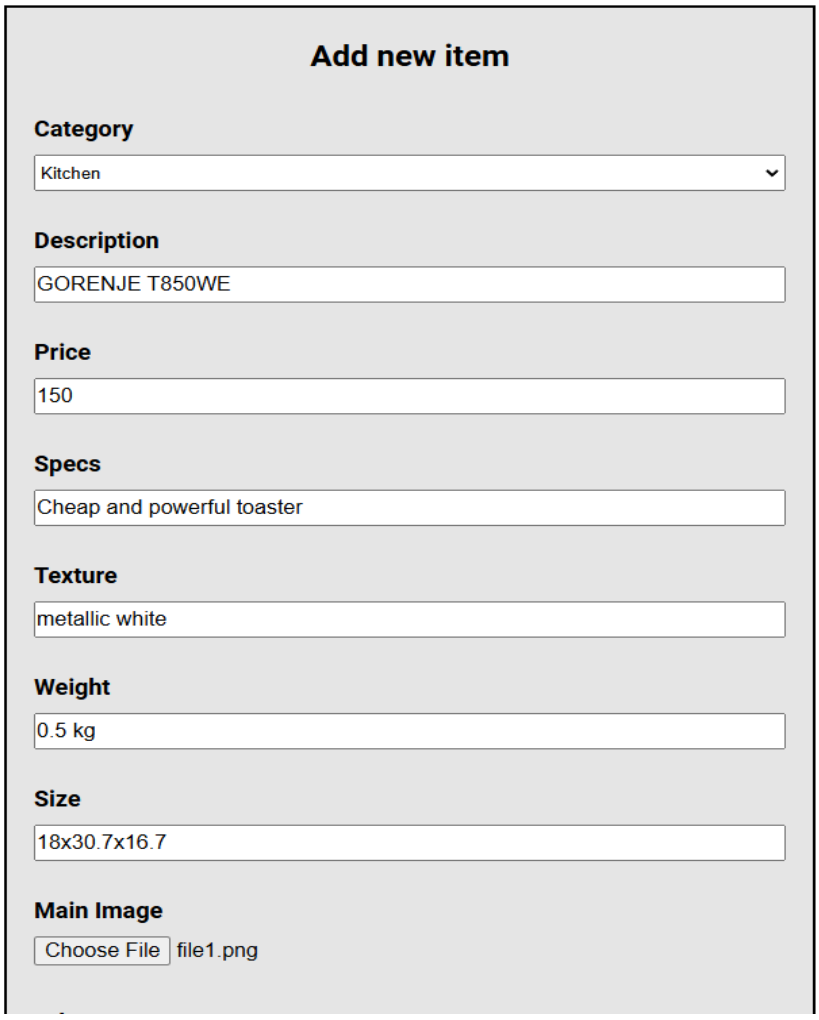


Рис. 3.10. Форма створення товару

Примітка. Джерело: розроблено автором

Після відправлення форми з новим товаром, зайшовши на сторінку категорій з фільтрацією «kitchen», можна побачити, що новий товар був успішно доданий. Назва, зображення і ціна тостера коректно відображаються. Сторінку категорій з новим товаром можна побачити на рис. 3.11.

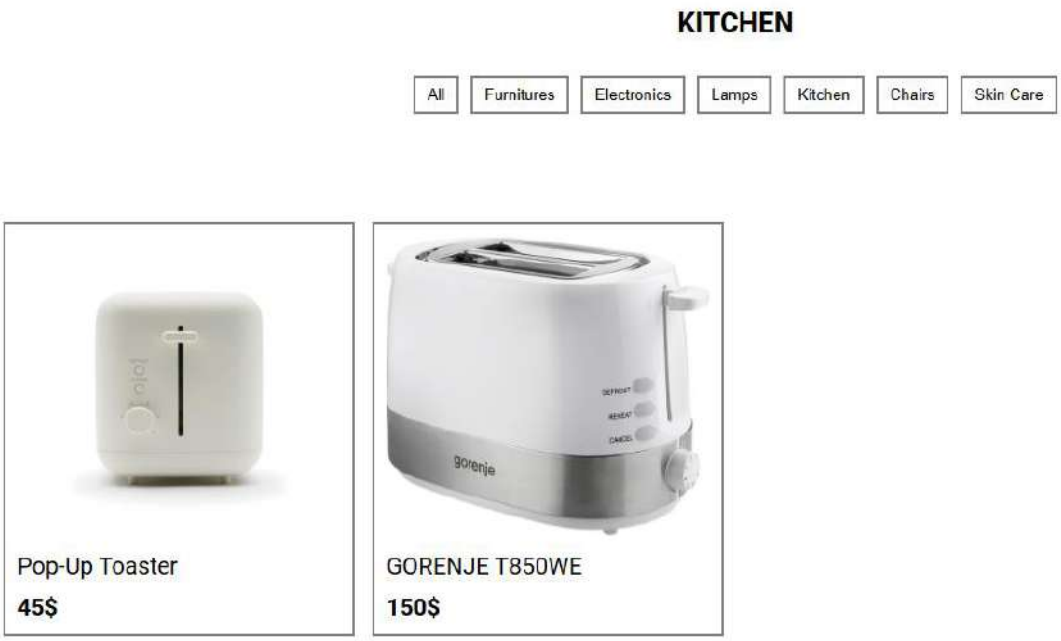


Рис. 3.11. Доданий товар на сторінці категорій

Примітка. Джерело: розроблено автором

Наступною функцією для перевірки було обрано можливість залишати коментарі. У нижній частині сторінки товару розташовано форму для введення коментаря. Було введено текст відгуку та натиснуто кнопку відправлення. Після оновлення сторінки доданий коментар відобразився у списку відгуків. Деталі відображення коментаря можна побачити на рис. 3.12.

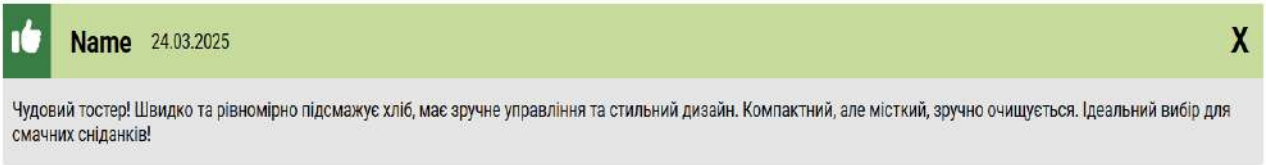


Рис. 3.12. Коментар до товару

Примітка. Джерело: розроблено автором

Проведені тестування підтвердили стабільність і високу продуктивність електронного магазину. Усі основні функції працюють коректно, а система демонструє швидке завантаження та адаптивність на різних пристроях. Ці заходи забезпечують позитивний досвід взаємодії користувачів із веб-додатком і готовність системи до використання у реальних умовах.

Висновки до розділу 3

У третьому розділі було здійснено практичну реалізацію клієнтської та серверної частин електронного магазину, а також проведено тестування та оптимізацію створеної системи. Результатом цього етапу стало повноцінне впровадження усіх функціональних компонентів, необхідних для коректного та безперебійного функціонування веб-додатку.

Клієнтська частина була реалізована із дотриманням принципів адаптивного дизайну, юзабіліті та інтуїтивної взаємодії з користувачем. Реалізовано основні сторінки електронного магазину: головну, сторінки категорій, картки товарів, кошик і панель адміністратора. Забезпечено інтеграцію з серверною частиною через REST API, що дало змогу реалізувати динамічне оновлення контенту, обробку дій користувача в реальному часі та взаємодію з базою даних.

Серверна частина додатку відповідає за обробку запитів, управління бізнес-логікою та роботу з базою даних. Забезпечено обробку CRUD-операцій (створення, читання, оновлення, видалення) для основних сутностей системи. Реалізовано механізми авторизації та автентифікації, що гарантують захист персональних даних користувачів. Структура серверної частини є масштабованою та дозволяє адаптувати систему до зростаючого навантаження без суттєвих змін архітектури.

Тестування функціональності охоплювало ключові процеси: реєстрацію та вхід користувача, пошук і фільтрацію товарів, додавання до кошика, оформлення замовлення, а також роботу адміністративного інтерфейсу. Виявлені помилки

були оперативно усунені, що дозволило досягти стабільної роботи всієї системи. Оптимізація продуктивності включала поліпшення часу відповіді API, зменшення часу завантаження сторінок і підвищення ефективності запитів до бази даних.

Загалом реалізований веб-додаток відповідає сучасним вимогам до систем електронної комерції: він є функціональним, безпечним, ефективним та готовим до подальшої експлуатації кінцевими користувачами. Створена система демонструє практичну реалізацію теоретичних і архітектурних засад, сформованих у попередніх розділах дослідження, і є конкурентоспроможною на тлі існуючих рішень у сфері онлайн-торгівлі.

ВИСНОВКИ

У процесі виконання дипломної роботи було здійснено комплексне дослідження теоретичних, технологічних та практичних аспектів створення веб-додатку електронної комерції. Робота охопила повний цикл розробки – від аналізу предметної області та вибору інструментів до реалізації, тестування та оптимізації системи.

У першому розділі розглянуто еволюцію електронної комерції, сучасні тенденції її розвитку та роль веб-технологій у цьому процесі. Проведено аналіз популярних e-commerce платформ, що дозволило виокремити їхні сильні та слабкі сторони й сформулювати вимоги до майбутнього програмного продукту.

Другий розділ було присвячено порівняльному аналізу інструментів для реалізації клієнтської, серверної частини та бази даних. На основі обґрунтованого вибору було використано стек технологій React + TypeScript на фронтенді, Node.js з Express – на бекенді, а PostgreSQL – для зберігання даних. Проведено архітектурне проектування веб-додатку з урахуванням принципів модульності, масштабованості та безпеки. Окрема увага приділена створенню адаптивного та інтуїтивного користувацького інтерфейсу відповідно до UX/UI стандартів.

У третьому розділі реалізовано повнофункціональний прототип електронного магазину: створено всі необхідні компоненти клієнтської та серверної частин, реалізовано функції реєстрації, перегляду товарів, фільтрації, роботи з кошиком і оформлення замовлень. Здійснено тестування ключових функціональних модулів, виявлено та усунуто недоліки, а також проведено базову оптимізацію продуктивності веб-додатку.

Результатом дослідження став готовий прототип системи електронної комерції, який відповідає сучасним вимогам до безпеки, продуктивності та зручності використання. Отримані знання та практичні навички можуть бути використані як основа для подальшого розвитку системи: інтеграції платіжних

сервісів, впровадження аналітичних інструментів, автоматизації логістики чи масштабування під потреби комерційного проєкту.

Крім того, дана робота зробила внесок у формування системного бачення процесу веб-розробки: від постановки завдання до створення архітектури, вибору стеку технологій, реалізації інтерфейсу й програмної логіки. Представлені підходи є універсальними й можуть бути адаптовані для широкого спектра задач, пов’язаних зі створенням сучасних веб-систем.

Таким чином, поставлені в роботі мета та завдання були повністю досягнуті, а розроблена система є не лише практично корисною, а й методологічно цінною для подальших досліджень та професійного зростання в галузі веб-розробки та цифрових технологій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. 51 ECommerce Statistics In 2025 (Global And U.S. Data) | SellersCommerce. *SellersCommerce*. URL: https://www.sellerscommerce.com/blog/ecommerce-statistics/ (дата звернення: 08.04.2025).

2. Захожий М. Тренди та виклики українського ринку eCommerce у 2024 році. *UAATEAM*. URL: https://uaateam.agency/blog/trendy-ta-vyklyky-ukrainskogo-rynku-ecommerce/ (дата звернення: 08.04.2025).

3. Дімура М. E-commerce в Україні 2024: створення інтернет-магазину, законодавство, як підвищити довіру споживачів. *BUSINESS SITE*. URL: https://www.site2b.ua/ua/web-blog-ua/e-commerce-v-ukraini-cifri-fakti-perspektivi-rozvitku-onlajn-torgivli.html (дата звернення: 12.04.2025).

4. PostgreSQL: About. *PostgreSQL: The world's most advanced open source database*. URL: https://www.postgresql.org/about/ (дата звернення: 12.04.2025).

5. Chinnathambi K. Learning React: A Hands-On Guide to Building Web Applications Using React and Redux. Addison-Wesley, 2018. 336 с.

6. Why does TypeScript exist?. *TypeScript: JavaScript With Syntax For Types*. URL: https://www.typescriptlang.org/why-create-typescript/ (дата звернення: 12.04.2025).

7. Що таке redux і в яких випадках варто його використовувати. *FoxmindEd*. URL: https://foxminded.ua/shcho-take-redux/ (дата звернення: 14.04.2025).

8. Семков Д. Розуміння Клієнт-Серверної Архітектури на прикладах. *DOU*. URL: https://dou.ua/forums/topic/44636/ (дата звернення: 14.04.2025).

9. Duarte F. Internet Traffic from Mobile Devices (Feb 2025). *Exploding Topics*. URL: https://explodingtopics.com/blog/mobile-internet-traffic (дата звернення: 18.04.2025).

10. Getting Started | Create React App. *Create React App*. URL: https://create-react-app.dev/docs/getting-started (дата звернення: 18.04.2025).

11. Window: localStorage property - Web APIs | MDN. *MDN Web Docs*. URL: https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage (дата звернення: 18.04.2025).

12. Hahn E. Express in Action: Writing, building, and testing Node.js applications. Manning Publications, 2016. 256 с.

ЗГОДА здобувача вищої освіти

Державного університету економіки і технологій про перевірку кваліфікаційної роботи на прояви академічного плагіату та розміщення в Репозитарії Університету

Я, Мироненко Тимур Ігорович, підтримую політику Державного університету економіки і технологій з академічної доброчесності і відкритого доступу.

Засвідчую, що кваліфікаційна бакалаврська робота «Проєктування та розробка онлайн-платформи для продажу товарів» виконана самостійно та не містить академічного плагіату. Я не надавав і не одержував недозволену допомогу під час підготовки цієї роботи. Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Із чинним Положенням про запобігання та виявлення академічного плагіату в роботах здобувачів вищої освіти Державного університету економіки і технологій ознайомлений. Чітко усвідомлюю, що в разі виявлення у кваліфікаційній роботі порушення норм академічної доброчесності робота не допускається до захисту або оцінюється незадовільно.

Також я поінформований, що відповідно до «Положення про Репозитарій (електронну базу даних) Державного університету економіки і технологій» зазначена робота буде розміщена в Електронному архіві Університету (Репозитарії ДУЕТ). З умовами такого розміщення ознайомлений.

10.06.2025  Мироненко Т. І.