МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ ФІНАНСІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: Методи та засоби розробки вебсайту інтернет магазину

Здобувача вищої освіти другого (магістерського) рівня,
ІІ курсу, групи КМІ-24-1, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Інформаційні управляючі системи і технології в економіці»

_____ Шмульського Олександра Олеговича
(підпис) (прізвище, ім'я, по батькові)

Науковий керівник _____ Лаговський Володимир Вікторович, канд. екон. наук, доцент
(підпис) (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Керівник освітньо-професійної програми _____ Омельчук Антон Анатолійович, канд. техн. наук
(підпис) (прізвище, ім'я, по батькові, науковийступінь, вчене звання)

Допущено до захисту за рішенням кафедри протокол від «02» січня 2026 № 8

Завідувач кафедри _____ Філоненко Михайло Миколайович, канд. фіз.-мат. наук, доцент
(підпис) (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Ірпінь – 2026

Реєстрація № 34 22.01.2026 _____
(дата) (підпис відповідальної особи)

Результати перевірки до захисту 02.01.2026 _____
(до захисту, на доопрацювання) (дата) (підпис)

Результат захисту 3 (задовільно) 10.02.2026 _____
(оцінка) (дата) (підпис)

Голова комісії: _____ О.С. Дюбанов
(підпис)

Члени комісії:

_____ А.Ю. Горбовий
(підпис)

_____ М.М. Філоненко
(підпис)

_____ І.Д. Погорєловська
(підпис)

БЛАНК ОЦІНЮВАННЯ ДИПЛОМНОЇ РОБОТИ

Об'єкти контролю	Діапазон оцінки в балах	Фактична кількість балів	Примітки	Дата, П.І.Б.: керівника, голови комісії членів комісії,секретаря, (підписи)
ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ КОМІСІЄЮ ПО ЗАХИСТУ				
1. Зміст кваліфікаційної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	**від 0 до 30**	16		10.02.2026 Дюбанов О.С. Горбовий А.Ю. Філоненко М.М. Погорєловська І.Д. Суховецька Л.П.
2. Дотримання вимог щодо оформлення	**від 0 до 10**	7		
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	**від 0 до 5**	3		
4. Рецензент	**від 0 до 5**	4		
РАЗОМ БАЛІВ	**від 0 до 50**	30		

Оцінювання кваліфікаційної роботи науковим керівником - 32 балів.

ПЕРЕВЕДЕННЯ ОЦІНКИ ЗА 100-БАЛЬНОЮ ШКАЛОЮ ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалоюЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря, (підпис)
62	D	3	10.02.2026	Дюбанов О.С. Горбовий А.Ю. Філоненко М.М. Погорєловська І.Д. Суховецька Л.П.

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою
90-100	A	5 (відмінно)
80-89	B	4 (добре)
70-79	C	
60-69	D	3 (задовільно)
50-59	E	
35-49	F X	2 (незадовільно)

Факультет фінансів та цифрових технологій
Кафедра комп'ютерних та інформаційних технологій і систем
Галузь знань 12 «Інформаційні технології»
Спеціальність 122 «Комп'ютерні науки»
Освітня програма «Інформаційні управляючі системи і технології в економіці»

Затверджую
Завідувач кафедри _____ Філоненко М.М.
«11» грудня 2024

Завдання на кваліфікаційну роботу
здобувачу вищої освіти
Шмульському Олександру Олеговичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи
Методи та засоби розробки вебсайту інтернет магазину
Затверджена наказом ректора від «10» грудня 2024 № 1852
2. Строк здачі закінченої роботи «02» січня 2026
3. Мета роботи розробка повнофункціонального веб-сайту інтернет-магазину, призначеного для підтримки основних бізнес-процесів електронної комерції.
Об'єкт дослідження процес розробки веб-сайту інтернет-магазину як складової системи електронної комерції
Предмет дослідження методи та засоби проєктування і програмної реалізації веб-сайту інтернет-магазину.
Орієнтовна база дослідження інформаційне середовище електронної комерції та практичні підходи до створення веб-ресурсів онлайн-торгівлі

4. Вихідні дані до роботи вимоги до функціональності веб-сайту інтернет-магазину, нормативні та методичні матеріали з веб-розробки, дані предметної області електронної комерції, сучасні програмні засоби та технології розробки.

5. Перелік графічного матеріалу діаграми бізнес-процесів інтернет-магазину, схема архітектури веб-сайту, фізична модель бази даних, скріншоти основних сторінок веб-сайту.

6. Зміст кваліфікаційної роботи (перелік питань за кожним розділом):
аналіз предметної області та вибір інструментів розробки веб-сайту інтернет-магазину, проєктування веб-сайту інтернет-магазину, програмна реалізація та тестування веб-сайту інтернет-магазину.

7. Календарний план виконання роботи

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	За фактом
1	2	3	4
1	Вибір теми, підбір літературних джерел, формування структури кваліфікаційної роботи	До 15.12.2024	До 15.12.2024
2	**І розділ** (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	28.05.2025	28.05.2025
3	**ІІ розділ** (аналіз експериментальної частини про методики досліджень)	08.09.2025	08.09.2025
4	**ІІІ розділ** (проведені теоретичні і (або) експериментальні дослідження та проектні рішення)	17.11.2025	17.11.2025
5	**Висновки**	08.12.2025	08.12.2025
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	22.12.2025	22.12.2025
7	Отримання відгуку керівника	20.01.2026	20.01.2026
8	Отримання зовнішньої рецензії	23.01.2026	23.01.2026

8. Дата видачі завдання “11” грудня 2024

9. Керівник кваліфікаційної роботи _____ Лаговський В.В.

10. Керівник освітньо-професійної програми _____ Омельчук А.А.

11. Завдання до виконання отримав здобувач вищої освіти _____ Шмульський О.О.

АНОТАЦІЯ

Методи та засоби розробки веб сайту інтернет магазину

Мета роботи полягає в розробці повнофункціонального веб-сайту інтернет-магазину, призначеного для підтримки основних бізнес-процесів електронної комерції.

Завданнями роботи є:

– аналіз предметної області інтернет-магазинів;
– обґрунтування вибору методів та технологій веб-розробки;
– проєктування структури, архітектури та бази даних веб-сайту;
– реалізація і тестування функціоналу веб-сайту інтернет-магазину.

Предметом дослідження є методи та засоби проєктування і програмної реалізації веб-сайту інтернет-магазину.

Об’єктом дослідження є процес розробки веб-сайту інтернет-магазину як складової системи електронної комерції.

Базою дослідження виступає інформаційне середовище електронної комерції та практичні підходи до створення веб-ресурсів онлайн-торгівлі.

За результатами дослідження визначено основні етапи та підходи до розробки веб-сайту інтернет-магазину з урахуванням сучасних вимог до функціональності та архітектури.

Одержані результати можуть бути використані підприємствами, що здійснюють онлайн-продаж товарів, для автоматизації процесів представлення продукції, оформлення замовлень і взаємодії з клієнтами.

Кваліфікаційна робота містить 66 сторінки, 26 таблиць, 19 рисунків, список використаних джерел 40 найменувань, 2 додатки.

Ключові слова: інтернет-магазин, веб-сайт, електронна комерція, веб-розробка, архітектура, база даних, онлайн-торгівля.

Рік виконання кваліфікаційної роботи 2025

Рік захисту роботи 2026

ABSTRACT

Methods and Tools for Developing an E-Commerce Website

The aim of this work is to develop a fully functional e-commerce website intended to support the core business processes of electronic commerce.

The tasks of the work are as follows:

– analysis of the subject area of e-commerce websites;
– justification of selection of web development methods and technologies;
– design of the structure, architecture, and database of the website;
– implementation and testing of the functionality of the e-commerce website.

The subject of the research is the methods and tools for designing and programmatically implementing an e-commerce website.

The object of the research is the process of developing an e-commerce website as a component of an electronic commerce system.

The research is based on the information environment of electronic commerce and practical approaches to the development of online trading web resources.

The results of the research include the identification of the main stages and approaches to developing an e-commerce website in accordance with modern requirements for functionality and system architecture.

The obtained results can be used by enterprises engaged in online sales of goods to automate product presentation, order processing, and customer interaction processes.

The qualification work consists of 66 pages, 26 tables, 19 images, a list of 40 references, and 2 appendices.

Keywords: e-commerce website, web site, electronic commerce, web development, architecture, database, online trade.

Year of qualification work: 2025

Year of work defense: 2026

ЗМІСТ

ВСТУП 3
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ВИБІР ІНСТРУМЕНТІВ РОЗРОБКИ 6
1.1. Історія та сучасний стан інтернет магазинів 6
1.2. Аналіз веб сайтів 12
1.3. Вибір інструментів розробки веб-сайту 19
РОЗДІЛ 2. ПРОЄКТУВАННЯ ВЕБ-САЙТУ 25
2.1. Проєктування основних бізнес процесів веб-сайту 25
2.2. Проєктування бази даних веб-сайту 28
2.3. Проєктування архітектури веб-сайту 42
РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ВЕБ-САЙТУ 44
3.1. Розробка клієнтської частини веб-сайту 44
3.2. Розробка серверної частини веб-сайту 50
3.3. Тестування реалізованого функціоналу 54
ВИСНОВКИ 61
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 63
ДОДАТКИ 67

ВСТУП

Актуальність теми. У сучасних умовах глобальної цифровізації економічних процесів електронна комерція стає однією з провідних форм взаємодії між підприємствами й споживачами. Сформована цифрова екосистема сприяє трансформації традиційних ринків збуту, що відображається у постійному зростанні частки онлайн-торгівлі у світовому товарообігу. Так, за даними міжнародних досліджень, до 2025 року обсяги світових продажів через інтернет перевищать 6,8 трильйона доларів США, при цьому частка електронної комерції у роздрібних продажах становитиме майже 21 % загального обсягу торгівлі, демонструючи стабільний довгостроковий тренд зростання попиту на цифрові канали закупівель [1].

Аналіз ринку електронної комерції в Україні підтверджує локальну релевантність цього тренду: обсяг онлайн-продажів українськими споживачами у 2024 році досяг 239 мільярдів гривень, що на 25 % більше, ніж у попередньому році [2]. Також статистично фіксується значне зростання кількості активних інтернет-покупців та частоти транзакцій, що свідчить про подолання бар’єрів довіри до цифрових каналів продажу та поступову адаптацію населення до онлайн-торгівлі як базової моделі споживчої поведінки.

Розвиток електронної комерції стимулюється не лише зростанням обсягів транзакцій, а й поступовим збільшенням кількості електронних торгових платформ у світі: за останні роки їх кількість зросла з приблизно 24 мільйонів у 2022 році до понад 30,7 мільйонів у 2024 році, що є показником активної цифрової експансії підприємницьких ініціатив у сегменті електронної торгівлі та конкуренції за споживача на глобальному рівні [3].

Узагальнення вищезазначених даних дозволяє стверджувати, що сучасний ринок послуг і товарів дедалі більше орієнтується на цифрові платформи, де інтернет-магазин відіграє ключову роль як інструмент взаємодії між продавцем і покупцем. Це зумовлює потребу у систематичному дослідженні методів і засобів створення функціональних і ефективних веб-рішень для інтернет-

торгівлі, які здатні забезпечувати масштабованість, зручність для користувачів і інтеграцію з аналітичними й бізнес-процесами підприємства.

Метою кваліфікаційної роботи є розробка повнофункціонального веб-сайту інтернет-магазину, призначеного для підтримки основних бізнес-процесів електронної комерції.

Для досягнення поставленої мети у роботі необхідно розв’язати такі основні *задачі*:

– проаналізувати предметну область інтернет-магазинів і наявні програмні рішення, визначивши ключові функціональні вимоги, типові бізнес-процеси та обмеження, що впливають на проєктування веб-сайту електронної комерції;

– обґрунтувати вибір інструментів і технологій розробки веб-сайту, з урахуванням вимог до продуктивності, масштабованості, безпеки, зручності розроблення та підтримки програмного продукту;

– спроєктувати структуру та архітектуру веб-сайту, включаючи модель основних бізнес-процесів, логічну структуру бази даних і загальну архітектурну схему взаємодії компонентів системи;

– реалізувати та перевірити працездатність веб-сайту інтернет-магазину, забезпечивши коректну роботу основного функціоналу та провівши тестування реалізованих можливостей.

Об’єкт дослідження – процес розробки веб-сайту інтернет-магазину як складової системи електронної комерції.

Предмет дослідження – методи та засоби проєктування і програмної реалізації веб-сайту інтернет-магазину.

База досліджень – інформаційне середовище електронної комерції та практичні підходи до створення веб-ресурсів онлайн-торгівлі.

Методи дослідження – теоретичні: аналіз і узагальнення наукових публікацій, галузевих звітів та Інтернет-ресурсів з питань електронної комерції і веб-розробки для визначення сучасних підходів до створення інтернет-магазинів; практичні: проєктування структури веб-сайту, бази даних і

архітектури системи; емпіричні: реалізація та тестування функціоналу веб-сайту з метою перевірки коректності обраних методів і засобів розробки.

Науково-практична важливість отриманих результатів полягає в узагальненні та обґрунтуванні підходів до проєктування і програмної реалізації веб-сайтів інтернет-магазинів, а також у можливості практичного використання розробленого веб-рішення для організації та підтримки процесів електронної торгівлі, підвищення зручності взаємодії з користувачами та автоматизації основних бізнес-процесів онлайн-продажу.

Практичне значення отриманих результатів полягає у можливості безпосереднього використання розробленого веб-сайту інтернет-магазину в діяльності підприємств, що здійснюють онлайн-продаж товарів, з метою автоматизації процесів представлення каталогу, оформлення замовлень і взаємодії з клієнтами, а також у застосуванні напрацьованих рішень і підходів як основи для створення або модернізації аналогічних веб-ресурсів у сфері електронної комерції.

Апробація результатів роботи. Результати дослідження висвітлено в конференції «Цифрова трансформація фінансової системи України та Країн V- 4 в умовах євроінтеграції» (м. Дубляни, 26 квітня 2023 р.).

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ВИБІР ІНСТРУМЕНТІВ РОЗРОБКИ

1.1. Історія та сучасний стан інтернет магазинів

Перші прояви електронної комерції з’явилися ще наприкінці 20-го століття. У 1969 році компанія CompuServe почала надавати послуги електронного обміну даними між бізнесами, заклавши підґрунтя для майбутніх онлайн-покупок [4]. У 1979 році британський винахідник Майкл Олдріч продемонстрував концепцію «електронного шопінгу», з’єднавши модифікований телевізор із комп’ютером для обробки транзакцій через телефонну лінію. Цей експеримент фактично став точкою відліку для появи інтернет-магазинів. Уже в 1980-х роках з’являються перші електронні торговельні майданчики: зокрема, у 1984 році сервіс CompuServe відкрив Electronic Mall – один із перших онлайн-магазинів, що дозволив користувачам купувати товари приблизно у 100 різних продавців через мережу [5]. Таким чином, ще до появи Всесвітньої павутини були зроблені перші спроби організувати дистанційну торгівлю за допомогою комп’ютерних мереж.

По-справжньому стрімкий розвиток інтернет-магазинів почався з поширенням World Wide Web у 1990-х роках [4]. Ключовою подією стало впровадження протоколу безпечних з’єднань SSL у 1994 році, що дозволило шифрувати дані транзакцій [4]. 11 серпня 1994 року відбулася перша безпечна онлайн-купівля – придбання користувачем диску Sting через платформу NetMarket, що ознаменувало народження повноцінної онлайн-торгівлі із захистом платежів [6]. Вже у 1995 році були засновані перші великі інтернет-магазини та платформи електронної комерції: Джефф Безос запустив Amazon.com (спочатку як книжковий онлайн-магазин), а П’єр Омідьяр – аукціонний сайт AuctionWeb, відомий пізніше як eBay [4]. Поява Amazon та eBay продемонструвала потенціал інтернет-комерції для масового споживача. Незабаром додалися й інші важливі елементи екосистеми: у 1998 році стартував сервіс електронних платежів PayPal, що спростив переказ коштів через інтернет

[4]. Це дозволило малим онлайн-магазинам приймати платежі від покупців по всьому світу, не створюючи власних складних платіжних систем.

Кінець 1990-х – початок 2000-х ознаменувалися бумом доткомів: сотні підприємців відкривали онлайн-бізнеси, а обсяг глобальної електронної комерції досяг $150 млрд у 1999 році [4]. Хоча «бульбашка доткомів» луснула у 2000 році, тимчасово загальмувавши розвиток, електронна торгівля вистояла і почала нове зростання. Уже до середини 2000-х онлайн-продажі стрімко відновилися: з'явився термін «Cyber Monday» (перший понеділок після Дня подяки) для позначення піку інтернет-розпродажів у святковий сезон. У 2005 році компанія Amazon запустила сервіс Amazon Prime з безкоштовною швидкою доставкою, що підняло очікування споживачів щодо рівня сервісу і стимулювало інших рітейлерів покращувати логістику доставки. Приблизно в цей же період сформувався ринок платформ для створення інтернет-магазинів: у 2006 році було засновано Shopify – хмарну платформу, що дозволила невеликим бізнесам легко запускати власні онлайн-вітрини. Паралельно з'явилися й інші системи управління електронною комерцією, такі як Magento та BigCommerce, які надавали готовий базовий функціонал магазину «з коробки» [7].

Протягом 2010-х років інтернет-магазини проникли майже в усі сфери роздрібної торгівлі. Онлайн-торгівля вийшла за межі книг та електроніки – споживачі почали замовляти одяг, товари повсякденного вжитку і навіть продукти харчування через інтернет. Наприклад, у 2012 році з'явився сервіс Instacart, який зробив можливим масштабну доставку продуктів з місцевих супермаркетів, вирішивши проблеми зберігання та логістики для онлайн-замовлень продуктів. У цей період спостерігався бурхливий розвиток інфраструктури електронної комерції: удосконалювалися платіжні системи (впровадження технологій 3D Secure, цифрових гаманців тощо), інтегрувалися служби доставки та відстеження посилок, з'явилися послуги сторонніх логістичних операторів (3PL) для аутсорсингу виконання замовлень [8]. Також значного поширення набули мобільні пристрої, що дало початок явищу m-commerce (мобільної комерції). Ритейлери були змушені адаптувати свої сайти

для смартфонів і випускати мобільні додатки, оскільки дедалі більша частка трафіку і замовлень надходила саме з мобільних платформ.

На кінець 2010-х – початок 2020-х інтернет-торгівля досягла зрілого стану, ставши помітною часткою світового роздрібного товарообігу. Так, у США частка e-commerce у загальному обсязі роздрібних продажів у третьому кварталі 2023 року склала 15,6%, або $284 млрд [4]. Глобальні показники також вражають: у 2017 році обсяг світової електронної комерції (включно з B2B і B2C) оцінювався майже в $29 трлн [4], і ця цифра продовжує зростати двозначними темпами щороку. Особливим каталізатором стала пандемія COVID-19 у 2020 році, коли через локдауни багато споживачів перейшли на онлайн-покупки. За оцінками IBM, пандемія прискорила перехід від офлайн- до онлайн-торгівлі приблизно на 5 років. Інтернет-магазини масово впровадили безконтактну доставку, розширили асортимент товарів першої необхідності і зміцнили інфраструктуру, щоб впоратися з піковими навантаженнями.

Отже, еволюція інтернет-магазинів пройшла кілька етапів: від поодиноких експериментів з електронними транзакціями в 1970–80-х, через становлення веб-комерції в 1990-х (з основними віхами на кшталт впровадження SSL та запуску піонерських платформ Amazon і eBay), до швидкого зростання у 2000–2010-х роках з появою спеціалізованих рішень та виходом в масовий ринок [9]. Сьогодні інтернет-магазини є комплексними інформаційними системами з глобальним охопленням, інтегрованими платіжними сервісами та розвиненою логістикою, що забезпечують користувачам зручність замовлення товарів з будь-якої точки світу. Розвиток технологій і зміна споживчих вподобань продовжують стимулювати появу нових форматів онлайн-торгівлі, про які йтиметься у подальших підрозділах.

На ранніх етапах становлення веб-комерції архітектура сайтів інтернет-магазинів була переважно монолітною. Це означає, що всі функції – від інтерфейсу користувача до керування базою даних – реалізовувалися в межах єдиного цілісного веб-додатку. Монолітні системи кінця 90-х – початку 2000-х були популярними через відносну простоту розробки: вони надавали «єдине

рішення для всього», де всі компоненти магазину тісно інтегровані між собою [10]. Подібні платформи «все-в-одному» (наприклад, ранні версії OSCOMMERCE або Magento) дозволяли запустити онлайн-магазин швидко, але мали фіксовану структуру і обмежену гнучкість [11]. З часом, із зростанням бізнесу та ускладненням вимог, недоліки моноліту ставали все відчутнішими – великі системи виявилися важкими у масштабуванні та підтримці. Внесення змін вимагало обережності: навіть невелике оновлення могло вплинути на роботу всього сайту, адже всі частини системи тісно пов’язані. Крім того, висока взаємозалежність компонентів призводила до ситуацій єдиної точки збою: вихід з ладу будь-якого модуля міг порушити роботу всього магазину. Компанії стикалися з тим, що для додавання нових можливостей або індивідуальних налаштувань доводилося переписувати значну частину базового коду, а оновлення займали багато часу. Це сповільнювало впровадження нових функцій і ускладнювало реагування на зміну ринкових трендів.

Щоб подолати обмеження монолітів, у архітектурі веб-додатків почали застосовувати багаторівневий підхід. Спочатку набув поширення двохрівневий варіант архітектури: клієнтський рівень (веб-інтерфейс) напряму взаємодіє з сервером баз даних. Така 2-рівнева архітектура характерна для невеликих і простих інтернет-магазинів, де браузер користувача відправляє запити безпосередньо до серверу, що одночасно обробляє бізнес-логіку і звертається до БД [12]. Перевагою було спрощення структури, але при розширенні функціоналу це рішення ставало неповоротким: будь-які зміни у логіці вимагали модифікації всього серверного додатку, а навантаження на один сервер зростало непропорційно. Тому поступово індустрія перейшла до трирівневої архітектури, де чітко розділені: презентаційний рівень (інтерфейс користувача), прикладний рівень (бізнес-логіка) та рівень даних (бази даних). У такій моделі фронтенд (наприклад, веб-сторінки HTML/CSS/JS) відповідає лише за відображення і взаємодію з користувачем, бізнес-логіка (серверне застосування) опрацьовує запити – виконує операції з товарами, замовленнями, застосовує правила ціноутворення тощо – а окремий сервер бази даних зберігає інформацію про

товари, акаунти клієнтів, історію замовлень тощо. Трирівнева архітектура підвищила гнучкість і масштабованість: можна незалежно оновлювати дизайн сайту, оптимізувати базу даних або розширювати бізнес-логіку без повного перегляду всієї системи. Крім того, з'явилася можливість розподілити навантаження – наприклад, використовувати декілька веб-серверів для обслуговування користувачів, балансувати запити до БД тощо.

Наступним кроком в еволюції архітектур інтернет-магазинів стало впровадження сервіс-орієнтованих підходів. З розвитком хмарних технологій та зростанням вимог бізнесу, великі платформи почали переходити від моноліту до мікросервісної архітектури. Суть підходу полягає в розділенні великого застосунку на набір дрібніших автономних сервісів, кожен з яких відповідає за окрему функцію або бізнес-можливість [13]. Така декомпозиція нагадує конструктор LEGO – окремі сервісні «цеглинки» можуть поєднуватися для побудови цілісної системи. Наприклад, незалежними сервісами можуть бути каталог товарів, сервіс кошика, модуль оформлення замовлень, сервіс обробки платежів, модуль управління обліковими записами користувачів тощо. Кожен мікросервіс розгортається, масштабується та оновлюється автономно, що розв'язує проблему «єдиної точки відмови» і прискорює цикл розробки: зміни в одному компоненті мінімально впливають на інші і можуть впроваджуватися без простою всього сайту. На практиці це забезпечує значно більшу гнучкість бізнесу – компанія може швидше додавати нові функції або адаптуватися до ринкових вимог, не боячись зламати наявний функціонал. ак, багато сучасних високонавантажених платформ (наприклад, Amazon) перейшли на мікросервіси, щоб впоратися з масштабом операцій і різноплановістю своїх сервісів [14]. Варто зазначити, що розвитком ідей сервіс-орієнтованості стало також поняття headless commerce – «безголової» архітектури, в якій фронтенд (вітрина магазину) відокремлений від бекенд-сервісів. Headless-підхід дозволяє використовувати різні фронтенд-технології або навіть декілька каналів (веб-сайт, мобільний додаток, сторонні платформи) над спільним набором бекенд-сервісів магазину через API. Це ще більше підвищує гнучкість: можна оновлювати або змінювати

користувацький інтерфейс незалежно від бекенду, адаптуючись до трендів дизайну або особливостей пристроїв, якими користуються покупці.

У результаті еволюції архітектур сьогодні сформувалася певна стандартна структура функціональних модулів інтернет-магазину. Незалежно від того, монолітна чи мікросервісна реалізація, типовий веб-магазин включає декілька базових компонентів:

– каталог товарів – центральний модуль, що містить інформацію про продукти (описи, ціни, фото, категорії, залишки на складі тощо). Каталог забезпечує навігацію по асортименту і є джерелом даних для всіх інших частин системи. Сучасні каталоги повинні підтримувати зручне керування контентом (CMS/PIM) для адміністраторів, щоб легко додавати/редагувати товарні позиції, а також можливості фільтрації та пошуку для користувачів;

– кошик покупця – механізм, що дозволяє користувачу відкладати вибрані товари перед покупкою. Функціонал кошика передбачає додавання і видалення товарів, зміну їх кількості, розрахунок проміжної суми замовлення, врахування знижок чи промокодів. онлайн-кошик є аналогом реального візка в супермаркеті і відіграє критичну роль у зручності процесу замовлення [15]. Від його реалізації залежить конверсія: наприклад, можливість зберегти вміст кошика між сесіями чи синхронізувати його між пристроями покращує користувацький досвід;

– управління замовленнями – бекенд-модуль (іноді званий OMS – Order Management System), що обробляє отримані замовлення: генерує унікальні номери, фіксує склад замовлення, відстежує його статус (нове, оплачено, передано на доставку, виконано тощо) [16]. Цей компонент інтегрується з системами складу для резервування товарів, з логістичними сервісами для відправки посилок і з бухгалтерськими системами для обліку продажів;

– облікові записи користувачів – модуль реєстрації та автентифікації покупців, що дозволяє створювати особисті кабінети. У профілі користувача зберігаються контактні дані, історія замовлень, списки бажань, налаштування (наприклад, адреси доставки, платіжні методи), а також можуть бути реалізовані

програми лояльності. Наявність аккаунта не завжди обов’язкова для здійснення покупки (можливий гостьовий checkout), але особистий кабінет підвищує залученість клієнтів і відкриває можливості для персоналізації сервісу;

– платіжна інфраструктура – набір компонентів, що забезпечують прийом онлайн-платежів. Як правило, включає інтеграцію з одним або кількома платіжними шлюзами (банківськими еквайринговими системами, сервісами на кшталт PayPal, Stripe, LiqPay тощо). Основні вимоги – безпека передачі даних (шифрування, відповідність стандартам PCI DSS) та підтримка різних методів оплати (платіжні картки, онлайн-банкінг, електронні гаманці, післяплата). Сучасні платіжні модулі використовують перенаправлення на захищені сторони платіжних провайдерів або токенізацію даних карт, щоб інтернет-магазин не зберігав чутливі реквізити клієнтів у себе.

Перераховані функціональні модулі присутні фактично у кожному інтернет-магазині – від невеликого місцевого до світового маркетплейсу. У традиційній монолітній архітектурі всі ці підсистеми зібрані в єдиному застосунку, тоді як у сучасних підходах їх розділяють на окремі сервіси, що взаємодіють через чіткі інтерфейси. Наприклад, у мікросервісній реалізації може існувати сервіс каталогу, сервіс управління замовленнями, сервіс платежів тощо, які обмінюються даними через API. Такий підхід відповідає бізнес-логіці: окремі команди розробників можуть відповідати за свій напрям (продуктовий каталог, процесинг платежів і т.д.), масштабувати його за потреби і впроваджувати зміни незалежно від інших частин.

1.2. Аналіз веб сайтів

Сучасний сегмент електронної комерції представлений значною кількістю веб-сайтів інтернет-магазинів, які відрізняються за функціональним наповненням, структурою, зручністю користування та рівнем технічної реалізації. Такі веб-сайти виступають основним інтерфейсом взаємодії між продавцем і покупцем та безпосередньо впливають на сприйняття сервісу, ефективність продажів і лояльність користувачів. Різноманітність підходів до

організації каталогу товарів, процесу оформлення замовлення, навігації, візуального представлення інформації та інтеграції з платіжними й логістичними сервісами зумовлює необхідність системного аналізу існуючих веб-сайтів інтернет-магазинів. Такий аналіз дозволяє виявити характерні особливості реалізації ключових елементів електронної торгівлі, оцінити рівень відповідності сучасним вимогам користувацького досвіду, безпеки та продуктивності, а також сформувати практичні орієнтири для подальшого проєктування й розробки власного веб-сайту інтернет-магазину.

Amazon є масштабним веб-сайтом електронної комерції, призначеним для організації онлайн-продажу широкого спектра товарів і послуг, а також для взаємодії між продавцями та покупцями у межах єдиної цифрової платформи (рис. 1.1). Основне призначення веб-сайту полягає у забезпеченні повного циклу електронної торгівлі, що включає пошук і перегляд товарів, персоналізовані рекомендації, оформлення замовлень, онлайн-оплату, управління доставкою та післяпродажне обслуговування користувачів.

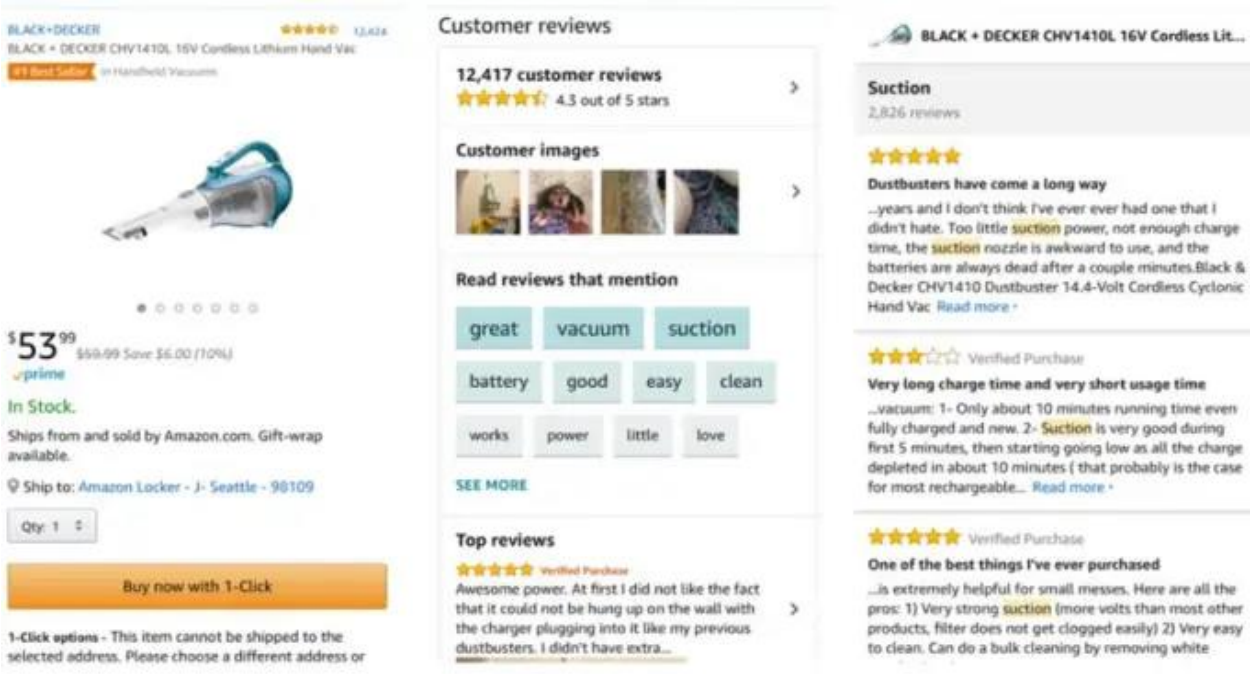


Рис. 1.1. Приклад інтерфейсу веб-сайту «Amazon» [17]

З архітектурної точки зору веб-сайт Amazon побудований на основі розподіленої багаторівневої архітектури, що поєднує клієнтську частину, серверну бізнес-логіку та масштабовану інфраструктуру зберігання даних.

Клієнтський рівень реалізовано у вигляді веб-інтерфейсу, доступного через браузер, який забезпечує взаємодію користувачів із системою [18]. Серверна частина реалізує обробку бізнес-процесів, управління каталогом, замовленнями та користувачами, використовуючи мікросервісний підхід і асинхронну взаємодію між компонентами. Дані зберігаються у розподілених базах даних та сховищах, що забезпечує високу продуктивність, відмовостійкість і здатність обробляти великі обсяги транзакцій у режимі реального часу.

Переваги веб-сайту:

– висока функціональна насиченість, що забезпечує повний цикл електронної комерції – від пошуку товарів до післяпродажного обслуговування [19];

– масштабована та відмовостійка архітектура, здатна обробляти великі обсяги трафіку і транзакцій у режимі реального часу;

– розвинені механізми персоналізації, рекомендацій та аналітики, що підвищують зручність користування і стимулюють продажі [20];

– інтеграція з власними логістичними, платіжними та хмарними сервісами, що забезпечує швидку доставку та стабільну роботу платформи.

Недоліки веб-сайту:

– складний і перевантажений інтерфейс, який може ускладнювати навігацію для нових користувачів;

– обмежені можливості індивідуальної взаємодії з продавцем у порівнянні з нішевими інтернет-магазинами [21];

– високі вимоги до технічних ресурсів клієнтської частини, що може впливати на швидкодію на слабких пристроях або повільних мережах [22];

– складність адаптації архітектурних рішень для використання у невеликих або локальних інтернет-магазинах через масштаб і специфіку платформи.

Отже, Amazon є прикладом високонавантаженого веб-сайту інтернет-магазину з максимальною функціональністю та складною архітектурою, що

ефективно реалізує процеси електронної комерції на глобальному рівні, однак його підходи та технічні рішення доцільні насамперед для великих платформ і не завжди є оптимальними для малих і середніх проєктів.

eBay є веб-сайтом електронної комерції, призначеним для організації онлайн-торгівлі за моделлю маркетплейсу, у межах якого здійснюється взаємодія між продавцями та покупцями з різних країн (рис. 1.2). Основне призначення веб-сайту полягає у забезпеченні платформи для продажу нових і вживаних товарів із використанням механізмів фіксованої ціни та аукціонів, а також у підтримці процесів пошуку товарів, оформлення замовлень, онлайн-оплати та управління транзакціями між сторонами.

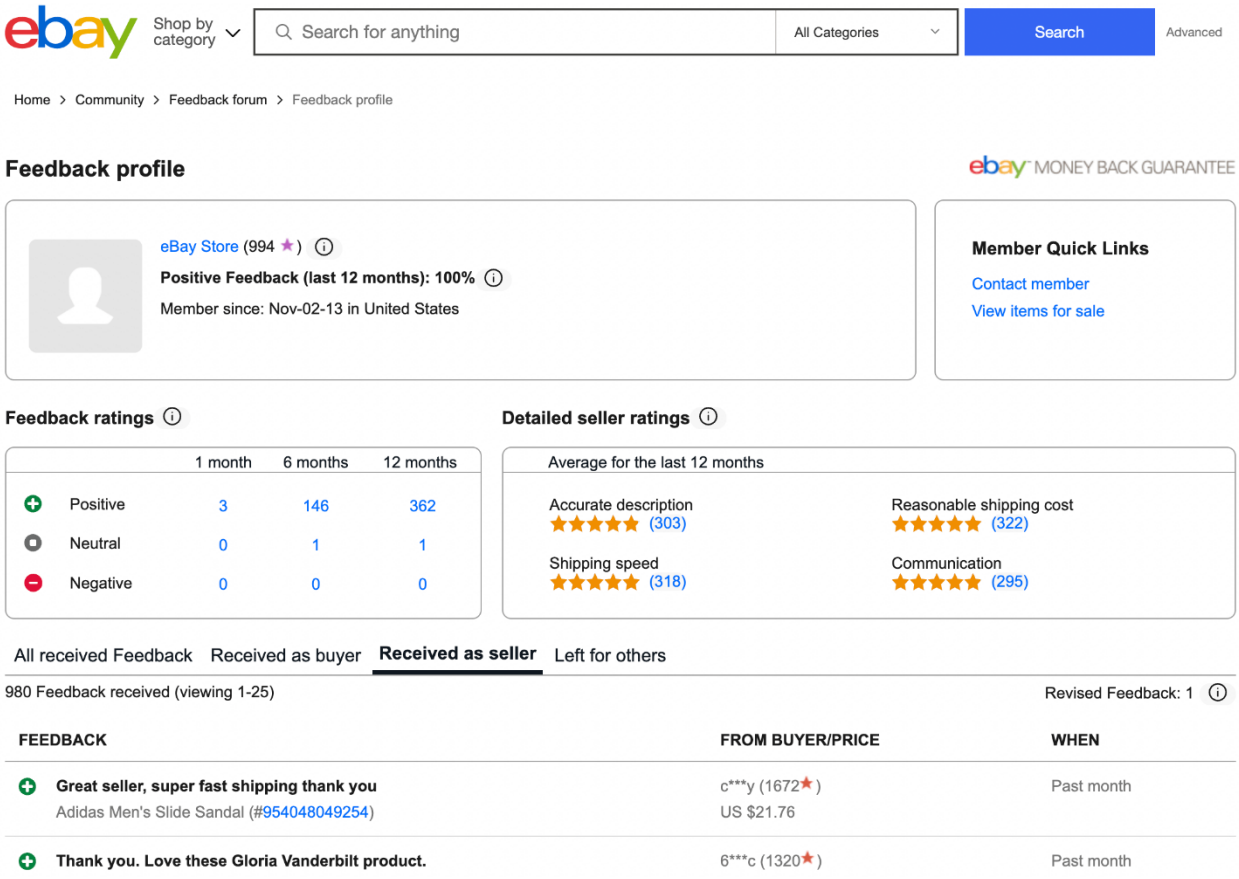


Рис. 1.2. Приклад інтерфейсу веб-сайту «eBay» [24]

З архітектурної точки зору веб-сайт eBay побудований на основі розподіленої багаторівневої архітектури, що поєднує клієнтський веб-інтерфейс, серверну бізнес-логіку та масштабовану інфраструктуру зберігання даних [23]. Клієнтський рівень забезпечує доступ користувачів до функціоналу платформи через браузер і реалізує інтерфейси перегляду лотів, участі в аукціонах та

керування обліковими записами. Серверна частина відповідає за обробку бізнес-процесів маркетплейсу, синхронізацію ставок, управління замовленнями та фінансовими операціями, використовуючи сервісно-орієнтований і мікросервісний підходи. Зберігання даних здійснюється у розподілених базах даних і кешуючих системах, що забезпечує високу доступність, продуктивність і стійкість платформи до великих навантажень.

Переваги веб-сайту:

– реалізація моделі маркетплейсу з підтримкою аукціонів і продажу за фіксованою ціною, що розширює можливості взаємодії між продавцями та покупцями [25];

– широке охоплення міжнародної аудиторії та підтримка транскордонної електронної комерції [26];

– масштабована архітектура, здатна обробляти великі обсяги користувацьких запитів і транзакцій у режимі реального часу;

– розвинені механізми рейтингу та відгуків, які підвищують рівень довіри між учасниками торговельного процесу.

Недоліки веб-сайту:

– складна навігація та перевантажений інтерфейс, що може ускладнювати пошук товарів для нових користувачів;

– залежність якості обслуговування від конкретних продавців, що впливає на загальний користувацький досвід [27];

– обмежені можливості кастомізації процесу продажу для окремих продавців у межах платформи [28];

– комісійні збори та додаткові платежі, які можуть знижувати економічну привабливість платформи для малого бізнесу.

Отже, eBay є ефективним веб-сайтом інтернет-торгівлі, орієнтованим на модель маркетплейсу та аукціонних продажів, який забезпечує масштабовану й надійну інфраструктуру для глобальної електронної комерції, однак складність

інтерфейсу та обмеження кастомізації зумовлюють доцільність його використання переважно для масового та міжнародного ринку.

Best Buy є корпоративним веб-сайтом роздрібної торгівлі, призначеним для онлайн-продажу електроніки, побутової техніки та супутніх сервісів, а також для інтеграції цифрових каналів із фізичною мережею магазинів (рис. 1.3). Основне призначення веб-сайту полягає у забезпеченні повного циклу взаємодії з клієнтом – від ознайомлення з асортиментом і порівняння товарів до оформлення замовлення, вибору способу отримання (доставка або самовивіз), післяпродажного обслуговування та підтримки сервісних послуг.

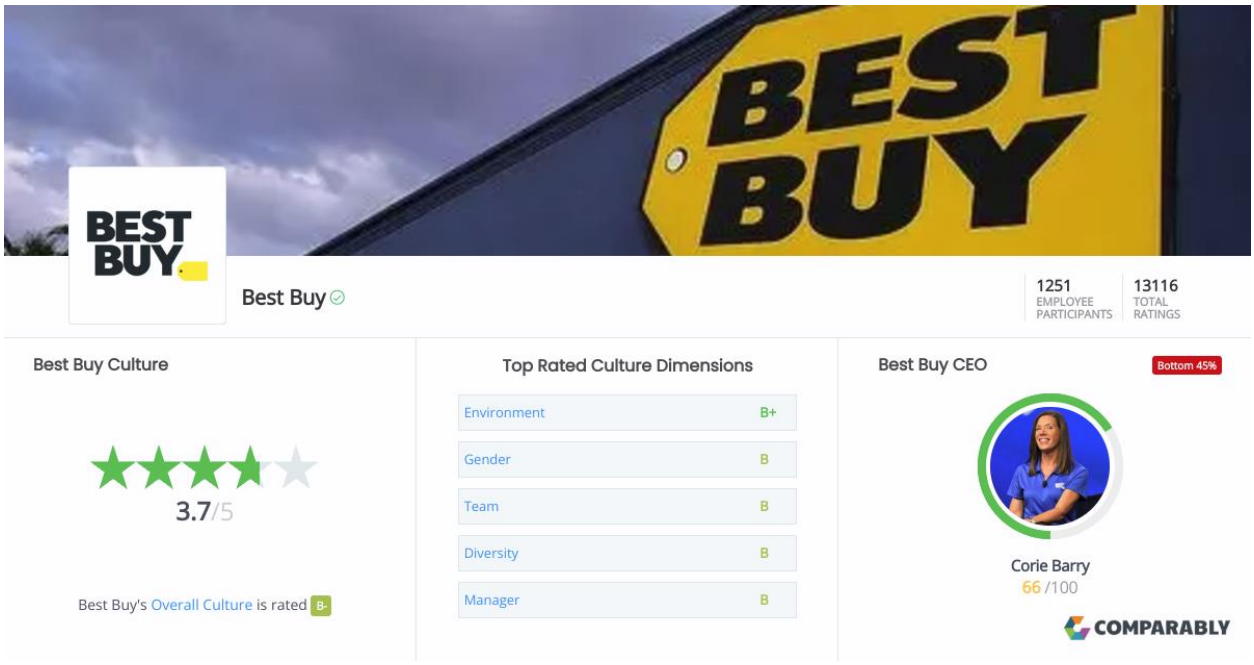


Рис. 1.3. Приклад інтерфейсу веб-сайту «Best Buy» [29]

З архітектурної точки зору веб-сайт Best Buy реалізовано на основі багаторівневої, сервісно-орієнтованої архітектури з чітким розмежуванням клієнтського інтерфейсу, серверної бізнес-логіки та рівня даних. Клієнтська частина забезпечує адаптивний веб-інтерфейс для різних типів пристроїв і реалізує сценарії перегляду товарів, персоналізованих рекомендацій та керування обліковим записом [30]. Серверний рівень відповідає за обробку замовлень, управління каталогом, ціноутворенням і запасами, а також за інтеграцію з внутрішніми корпоративними системами та зовнішніми платіжними й логістичними сервісами. Зберігання даних і обмін між компонентами

здійснюються з використанням масштабованих баз даних, кешування та асинхронних механізмів, що забезпечує стабільну роботу веб-сайту за умов високого навантаження.

Переваги веб-сайту:

– тісна інтеграція онлайн-платформи з фізичною мережею магазинів, що підтримує сценарії доставки та самовивозу [31];

– чітко структурований каталог електроніки з можливістю порівняння товарів і доступом до детальних технічних характеристик [32];

– масштабована архітектура, орієнтована на стабільну роботу під високими навантаженнями та піковим трафіком;

– інтеграція сервісних послуг і післяпродажної підтримки, що підвищує цінність платформи для користувачів.

Недоліки веб-сайту:

– відносно складний інтерфейс для користувачів без досвіду онлайн-покупок через велику кількість функціональних елементів;

– орієнтація переважно на локальний ринок, що обмежує можливості використання платформи за межами регіону присутності [33];

– значна залежність від внутрішніх корпоративних систем, що ускладнює швидку модифікацію бізнес-логіки [34];

– обмежена гнучкість дизайну та користувацьких сценаріїв у порівнянні з нішевими інтернет-магазинами.

Отже, Best Buy є прикладом корпоративного веб-сайту інтернет-магазину з високим рівнем інтеграції цифрових і офлайн-каналів продажу, який ефективно підтримує складні бізнес-процеси роздрібної торгівлі, проте його архітектурні та функціональні рішення доцільні насамперед для великих торговельних мереж і менш придатні для малих або спеціалізованих онлайн-проєктів.

Проведений аналіз існуючих програмних рішень і веб-сайтів інтернет-магазинів свідчить, що більшість популярних платформ орієнтовані на масштабні комерційні проєкти та характеризуються складною архітектурою,

надмірною функціональністю й залежністю від платних сервісів. Використання таких рішень є економічно та технічно недоцільним для невеликих онлайн-проєктів, які потребують базового набору функцій для представлення товарів і оформлення замовлень. У зв’язку з цим обґрунтованою є розробка нового веб-сайту інтернет-магазину з простішим функціоналом, зрозумілою структурою та мінімальними вимогами до інфраструктури, що забезпечує легкість впровадження і супроводу.

1.3. Вибір інструментів розробки веб-сайту

Вибір інструментів розробки веб-сайту є одним із ключових етапів створення інтернет-магазину, оскільки від обраних технологій залежить функціональність, продуктивність, масштабованість і зручність подальшого супроводу веб-ресурсу. Сучасна веб-розробка ґрунтується на використанні мов програмування загального призначення, які мають розвинені екосистеми, підтримку серверної логіки та інтеграцію з базами даних і веб-фреймворками. До найбільш поширених мов, що застосовуються для розробки серверної частини веб-сайтів інтернет-магазинів, належать Python, Java та PHP.

Python є високорівневою мовою програмування, орієнтованою на читабельність коду та швидкість розробки програмних рішень [35, с. 27]. Вона широко використовується у веб-розробці завдяки наявності фреймворків, які спрощують створення серверної логіки та роботу з базами даних. Python доцільно застосовувати для проєктів, де важливими є швидке прототипування та подальше розширення функціоналу.

Java є об’єктно-орієнтованою мовою програмування, що забезпечує високу надійність, масштабованість і стабільність веб-застосунків [36, с. 12]. Вона активно використовується у корпоративних інформаційних системах та великих інтернет-проєктах завдяки строгій типізації та розвиненим інструментам для побудови багаторівневих архітектур. Java доцільна для реалізації складних і високонавантажених веб-сайтів інтернет-магазинів.

PHP є скриптовою мовою програмування, спеціально орієнтованою на створення веб-застосунків і серверну обробку HTTP-запитів [37, с. 21]. Вона має простий синтаксис, широке поширення та тісну інтеграцію з веб-серверами і реляційними базами даних. PHP часто використовується для розробки інтернет-магазинів малого та середнього масштабу завдяки доступності хостингу й великій кількості готових рішень.

Для обґрунтованого вибору мови програмування доцільно виконати порівняльний аналіз найбільш поширених рішень, що застосовуються для серверної розробки веб-сайтів інтернет-магазинів (табл. 1.1).

Таблиця 1.1

Порівняння мов програмування для розробки веб-сайту інтернет-магазину

Характеристика	Python	Java	PHP
Тип виконання	Інтерпретована	Компілюється у байткод	Інтерпретована
Складність початкового впровадження	Середня	Висока	Низька
Типова вартість хостингу	Середня	Висока	Низька
Наявність масового shared-хостингу	Обмежена	Відсутня	Широка
Час розгортання простого проєкту	Середній	Тривалий	Короткий
Інтеграція з веб-серверами	Через проміжні компоненти	Через сервери застосунків	Нативна
Поріг входження для розробника	Середній	Високий	Низький
Кількість готових веб-рішень і CMS	Середня	Невелика	Дуже велика
Вимоги до апаратних ресурсів	Середні	Високі	Низькі

Проведений порівняльний аналіз мов програмування свідчить, що PHP є доцільним вибором для розробки веб-сайту інтернет-магазину з огляду на його безпосередню орієнтацію на серверну веб-розробку та простоту інтеграції з веб-серверами і базами даних. Низька складність початкового впровадження та короткий час розгортання проєкту забезпечують швидкий перехід від етапу проєктування до практичної реалізації веб-сайту. Широка підтримка з боку

доступних хостинг-провайдерів дозволяє мінімізувати витрати на інфраструктуру та спрощує експлуатацію системи. Значна кількість готових бібліотек, фреймворків і прикладних рішень сприяє ефективній реалізації типових функцій інтернет-магазину. Сукупність зазначених характеристик робить PHP раціональним вибором для створення веб-сайту з базовим і розширюваним функціоналом за умов обмежених ресурсів і потреби у простоті супроводу.

Вибір системи управління базами даних для розробки веб-сайту є важливим етапом проєктування інтернет-магазину, оскільки СУБД визначає надійність зберігання даних, швидкодію доступу до інформації, можливості масштабування та зручність супроводу системи. Серед найбільш поширених реляційних СУБД, що застосовуються у веб-розробці, доцільно розглянути MS SQL Server, MySQL та Firebird.

MS SQL Server є потужною реляційною системою управління базами даних, орієнтованою на корпоративні інформаційні системи та високонавантажені застосунки [38, с. 17]. Вона забезпечує розширені механізми транзакцій, безпеки та аналітичної обробки даних. Використання MS SQL Server доцільне для великих проєктів, однак потребує значних апаратних ресурсів і ліцензійних витрат.

MySQL є однією з найпоширеніших реляційних СУБД у сфері веб-розробки, що відзначається простотою використання та високою продуктивністю при роботі з веб-застосунками [39, с. 22]. Вона добре інтегрується з популярними мовами програмування та веб-серверами і широко підтримується хостинг-провайдерами. MySQL часто використовується для інтернет-магазинів малого та середнього масштабу завдяки поєднанню функціональності та доступності.

Firebird є компактною реляційною СУБД з відкритим вихідним кодом, орієнтованою на автономні та вбудовані інформаційні системи [40, с. 15]. Вона не потребує складного адміністрування та може працювати з мінімальними вимогами до ресурсів. Firebird доцільно застосовувати у невеликих веб-проєктах

або локальних рішеннях, де важливими є простота розгортання та низькі витрати на підтримку.

Для обґрунтованого вибору системи управління базами даних доцільно виконати порівняльний аналіз найбільш поширених реляційних СУБД, що застосовуються у веб-розробці інтернет-магазинів. Узагальнені результати порівняння СУБД MS SQL Server, MySQL та Firebird за основними якісними й кількісними характеристиками наведено в табл. 1.2.

Таблиця 1.2

Порівняння систем управління базами даних для веб-сайту інтернет-магазину

Характеристика	MS SQL Server	MySQL	Firebird
Тип СУБД	Реляційна	Реляційна	Реляційна
Орієнтація на веб-застосунки	Часткова	Безпосередня	Обмежена
Підтримка масового веб-хостингу	Обмежена	Широка	Обмежена
Складність адміністрування	Висока	Низька	Низька
Потреба у серверних ресурсах	Висока	Середня	Низька
Вартість використання	Платна (є обмежені безкоштовні версії)	Безкоштовна	Безкоштовна
Продуктивність для веб-запитів	Висока	Висока	Середня
Підтримка великої кількості одночасних підключень	Висока	Висока	Обмежена
Інтеграція з PHP	Опосередкована	Нативна	Обмежена
Поширеність у веб-проєктах, %	~20	~60	~10
Документація та спільнота	Велика	Дуже велика	Середня
Простота розгортання	Середня	Висока	Висока

З огляду на обґрунтований вибір мови програмування PHP доцільним є застосування сучасного серверного фреймворку, який забезпечує структуровану організацію коду, підтримку шаблонів проєктування та засобів безпеки. У межах даної роботи розробка веб-сайту інтернет-магазину здійснювалася на базі фреймворку Laravel, який є одним із найбільш поширених і функціонально насичених PHP-фреймворків. Laravel реалізує архітектурний шаблон Model–

View–Controller, що сприяє розділенню логіки обробки даних, бізнес-правил та представлення інформації, а також підвищує зрозумілість і підтримуваність програмного коду.

Використання фреймворку Laravel дозволяє значною мірою спростити реалізацію типових функцій інтернет-магазину, зокрема механізмів автентифікації та авторизації користувачів, маршрутизації HTTP-запитів, роботи з формами, валідації вхідних даних і захисту від поширених веб-загроз [41]. Наявність вбудованого ORM-механізму Eloquent забезпечує зручну та безпечну взаємодію з реляційними базами даних, зменшуючи залежність прикладної логіки від конкретної реалізації СУБД. Крім того, розвинена екосистема пакетів і активна спільнота розробників створюють передумови для швидкого розширення функціональних можливостей веб-застосунку без суттєвого ускладнення архітектури.

Таким чином, поєднання мови програмування PHP із фреймворком Laravel забезпечує раціональний баланс між швидкістю розробки, структурованістю програмного коду та можливостями подальшого розвитку веб-сайту інтернет-магазину. Обраний підхід дозволяє реалізувати серверну частину системи з урахуванням вимог до надійності, безпеки та зручності супроводу.

Проведений порівняльний аналіз систем управління базами даних показує доцільність використання MySQL для розробки веб-сайту інтернет-магазину з огляду на її безпосередню орієнтацію на веб-застосунки та широку підтримку у середовищі хостинг-провайдерів. Низька складність адміністрування та простота розгортання забезпечують швидке створення і підтримку бази даних без значних витрат апаратних ресурсів. Нативна інтеграція MySQL з мовою програмування PHP та фреймворком Laravel дозволяє ефективно реалізувати серверну логіку й доступ до даних з використанням сучасних ORM-механізмів. Висока продуктивність при обробці типових веб-запитів і підтримка значної кількості одночасних підключень роблять MySQL придатною для використання в інтернет-магазинах малого та середнього масштабу.

Водночас обрана архітектура веб-застосунку та використання фреймворку Laravel створюють передумови для подальшої масштабованості системи. У разі зростання обсягів каталогу товарів, кількості користувачів або потреби в обробці великих масивів напівструктурованих даних можливим є поетапний перехід до використання NoSQL-сховищ для окремих підсистем, зокрема каталогів продукції, історії переглядів або логів активності. Такий підхід дозволяє зберегти реляційну базу даних для транзакційно критичних операцій і водночас підвищити гнучкість та масштабованість інформаційної системи без повної зміни її технологічної основи.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ВЕБ-САЙТУ

2.1. Проєктування основних бізнес процесів веб-сайту

Проєктування основних бізнес-процесів веб-сайту інтернет-магазину є необхідним етапом розробки, оскільки саме на цьому рівні визначається логіка взаємодії користувачів із системою, порядок обробки даних та взаємозв'язок між програмними компонентами. Чітке формалізоване представлення бізнес-процесів дозволяє забезпечити коректну реалізацію функціональних вимог, узгодженість дій клієнтської і серверної частин, а також цілісність даних, що зберігаються в базі даних. Для відображення послідовності дій та обміну повідомленнями між основними компонентами веб-сайту доцільно використовувати UML-діаграми послідовності.

На рис. 2.1 наведено діаграму послідовності процесу реєстрації користувача на веб-сайті інтернет-магазину, яка відображає взаємодію між користувачем, веб-інтерфейсом сайту, серверною частиною, базою даних та сервісом електронної пошти.

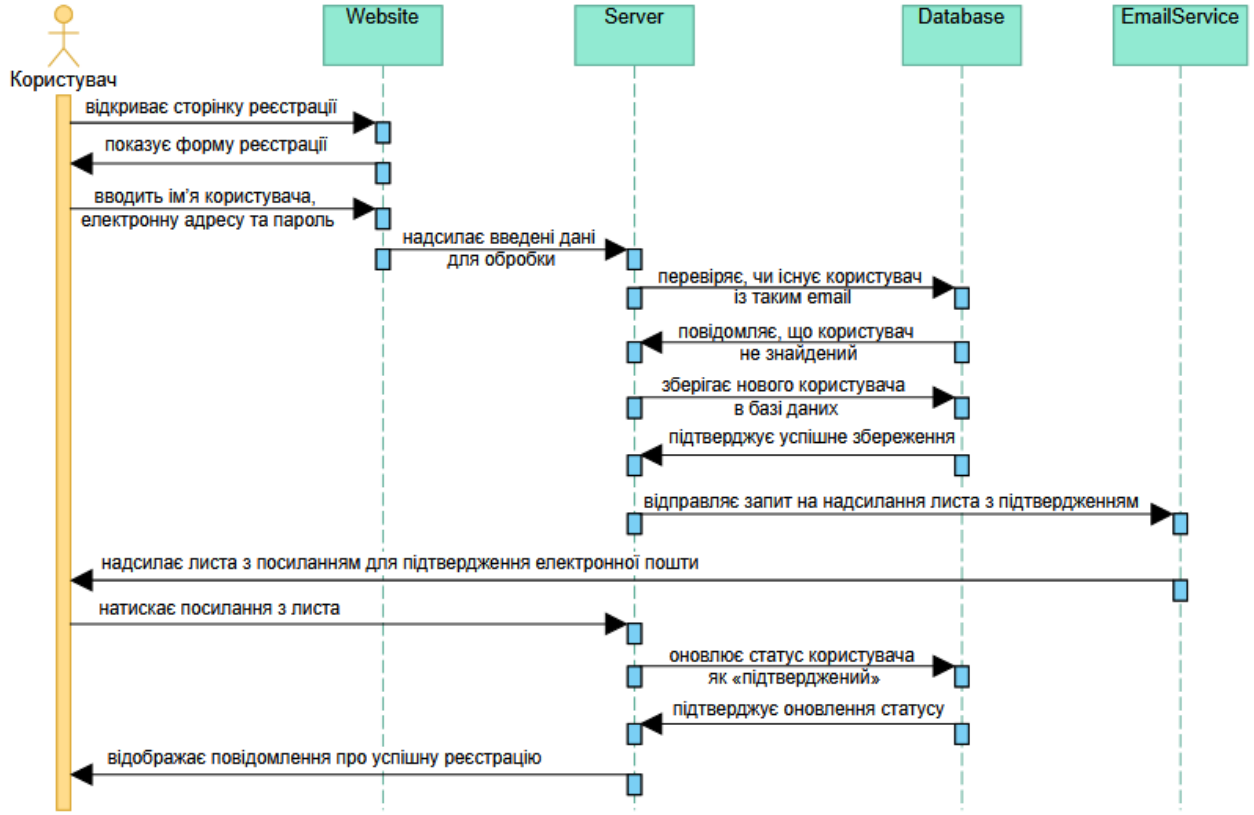


Рис. 2.1. Діаграма послідовності процесу реєстрації користувача на сайті

Процес розпочинається з ініціативи користувача, який відкриває сторінку реєстрації, після чого веб-сайт відображає форму для введення реєстраційних даних. Користувач вводить ім’я, адресу електронної пошти та пароль, а веб-інтерфейс передає ці дані на сервер для подальшої обробки. Серверна частина виконує перевірку наявності користувача з таким самим електронним адресом у базі даних і, у разі відсутності відповідного запису, ініціює створення нового користувача, при цьому додатково здійснюється валідація коректності введених даних та відповідності пароля встановленим вимогам безпеки. За результатами перевірки формується внутрішній службовий запис, що визначає можливість продовження процедури реєстрації.

Після збереження даних у базі сервер отримує підтвердження успішної операції та формує запит до сервісу електронної пошти для надсилання листа з посиланням підтвердження реєстрації. Користувач отримує електронний лист і переходить за наданим посиланням, що призводить до повторного звернення до серверної частини веб-сайту. У відповідь сервер оновлює статус облікового запису користувача як підтверджений, фіксує зміни в базі даних і отримує підтвердження успішного оновлення, а також може реєструвати відповідну подію в журналі системи для забезпечення аудиту та контролю доступу.

Завершальним етапом процесу є відображення веб-сайтом повідомлення про успішну реєстрацію, що свідчить про коректне виконання всіх етапів бізнес-процесу та готовність користувача до подальшої роботи з функціоналом інтернет-магазину. Одночасно система надає можливість негайної авторизації або переходу до персонального кабінету, що забезпечує безперервність взаємодії користувача з веб-ресурсом.

На рис. 2.2 наведено діаграму послідовності процесу додавання товару до кошика та оформлення замовлення, яка відображає взаємодію між користувачем, веб-сайтом, серверною частиною, базою даних та платіжним шлюзом у межах одного з ключових бізнес-процесів інтернет-магазину. Даний процес є центральним для функціонування веб-сайту електронної комерції, оскільки

поєднує етапи перегляду товарів, формування кошика, створення замовлення та виконання платіжної операції.

Процес починається з того, що користувач відкриває головну сторінку інтернет-магазину, після чого веб-сайт ініціює запит до серверної частини на отримання списку актуальних товарів для відображення у відповідному розділі. Сервер звертається до бази даних, отримує інформацію про товари, включаючи назву, зображення, ціну, розмір знижки та наявність, і передає ці дані веб-інтерфейсу. Отримавши інформацію, веб-сайт відображає картки товарів, забезпечуючи користувачу можливість перегляду асортименту.

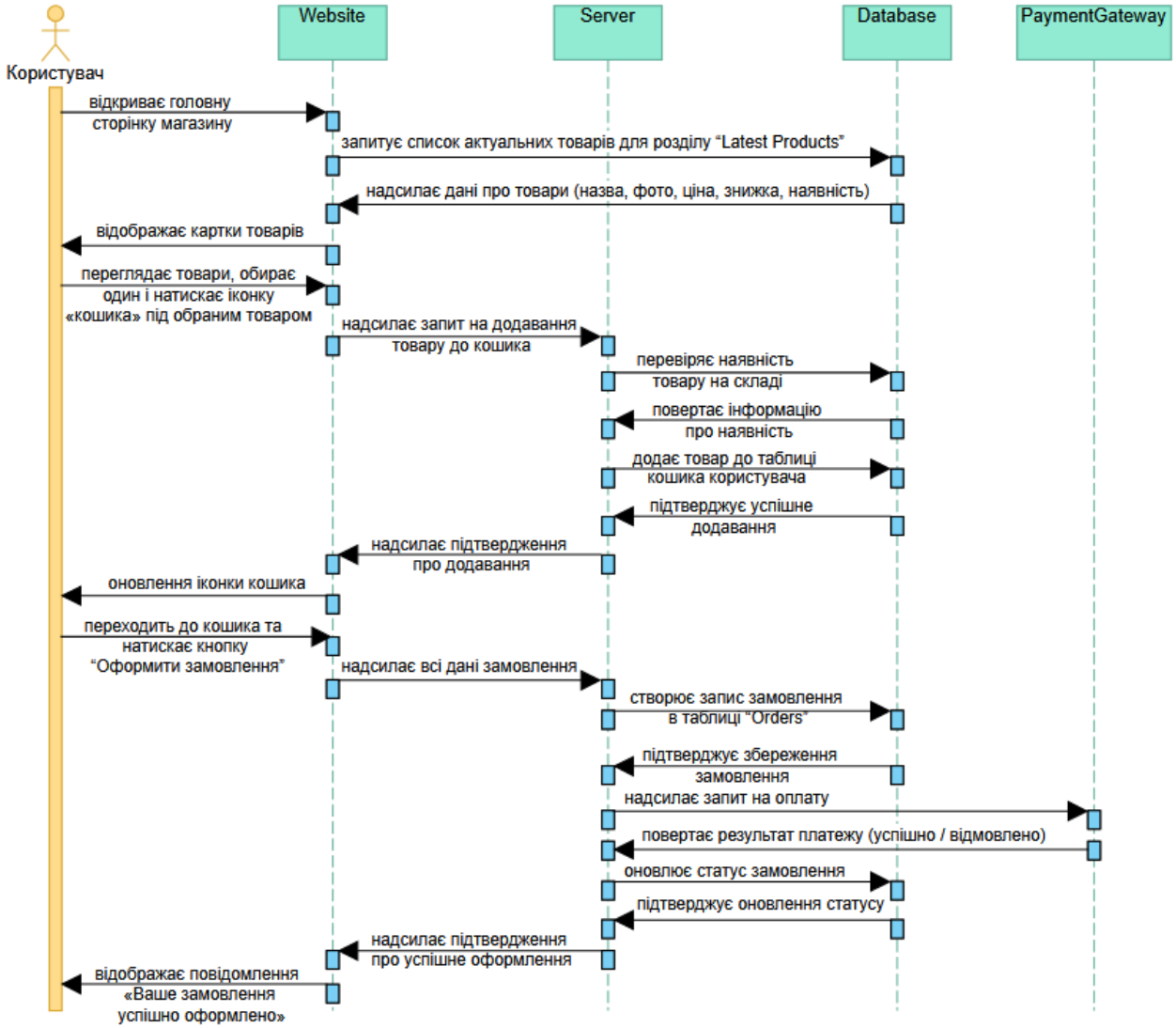


Рис. 2.2. Діаграма послідовності процесу додавання товару та оформлення замовлення

Після вибору товару користувач ініціює дію додавання до кошика, натискаючи відповідну іконку під обраним товаром. Веб-сайт передає серверу запит на додавання товару до кошика користувача, після чого серверна частина виконує перевірку наявності товару на складі шляхом звернення до бази даних. У разі підтвердження доступності товару сервер додає відповідний запис до таблиці кошика користувача та отримує підтвердження успішного збереження даних. Далі сервер надсилає веб-сайту повідомлення про успішне додавання товару, що призводить до оновлення індикатора кошика у користувацькому інтерфейсі.

На наступному етапі користувач переходить до сторінки кошика та ініціює оформлення замовлення, натискаючи відповідну кнопку. Веб-сайт формує та передає серверу повний набір даних замовлення, включаючи інформацію про обрані товари та користувача. Серверна частина створює новий запис замовлення у таблиці Orders бази даних і отримує підтвердження його успішного збереження. Після цього сервер ініціює взаємодію з платіжним шлюзом, надсилаючи запит на виконання платіжної операції.

Платіжний шлюз обробляє запит і повертає результат виконання платежу, який може бути як успішним, так і відхиленим. На основі отриманої відповіді сервер оновлює статус замовлення у базі даних та отримує підтвердження зміни стану. Завершальним етапом процесу є передавання веб-сайту повідомлення про успішне оформлення замовлення, після чого користувачу відображається відповідне інформаційне повідомлення.

2.2. Проєктування бази даних веб-сайту

Проєктування бази даних веб-сайту інтернет-магазину є складовою процесу розробки інформаційної системи, у межах якої формується структура зберігання та обробки даних, необхідних для функціонування прикладної частини веб-ресурсу. Структура бази даних визначає спосіб організації інформації, характер зв’язків між сутностями та механізми виконання запитів у процесі роботи системи. Реляційна модель даних дає змогу формалізувати

основні об’єкти предметної області, зокрема користувачів, товари, замовлення та довідкові сутності, забезпечуючи логічну узгодженість і можливість подальшого розширення функціональності веб-сайту. У процесі проєктування бази даних враховуються принципи нормалізації таблиць, визначення первинних і зовнішніх ключів, а також забезпечення транзакційної цілісності даних під час виконання бізнес-процесів електронної комерції. У межах даної роботи база даних веб-сайту інтернет-магазину спроєктована з орієнтацією на підтримку роботи з каталогом товарів, кошиком, замовленнями та обліковими записами користувачів, що зумовлює використання таких основних таблиць:

Таблиця Users призначена для зберігання облікових даних користувачів веб-сайту інтернет-магазину (табл. 2.1). Вона використовується для ідентифікації та автентифікації користувачів у системі, а також для фіксації базової інформації про момент створення та оновлення облікового запису. Дані цієї таблиці застосовуються під час виконання операцій реєстрації, входу в систему, оформлення замовлень і керування користувацькими профілями.

Таблиця 2.1

Атрибути сутності «Users»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор користувача
2	name	VARCHAR(255)	Ім’я користувача, що відображається у системі
3	email	VARCHAR(255)	Адреса електронної пошти користувача, яка використовується для авторизації та є унікальною
4	password	VARCHAR(255)	Хешоване значення пароля користувача
5	created_at	DATETIME	Дата та час створення облікового запису
6	updated_at	DATETIME	Дата та час останнього оновлення даних користувача

Таблиця BlogTags призначена для зберігання переліку тегів, що використовуються для додаткової тематичної класифікації блогових публікацій

(табл. 2.2). Теги застосовуються як гнучкий механізм групування контенту за ключовими поняттями, незалежно від ієрархії категорій, а також для реалізації пошуку та фільтрації матеріалів за відповідними мітками.

Таблиця 2.2

Атрибути сутності «BlogTags»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор тегу
2	name	VARCHAR(255)	Назва тегу, що відображається у веб-інтерфейсі
3	slug	VARCHAR(255)	Унікальний символьний ідентифікатор тегу для використання в URL

Таблиця BlogCategories призначена для зберігання інформації про категорії блогу веб-сайту (табл. 2.3). Вона забезпечує ієрархічну організацію тематичних розділів публікацій, що дає змогу формувати багаторівневу структуру категорій та підкатегорій. Таблиця використовується для класифікації блогових матеріалів, побудови навігації та фільтрації контенту за тематичними ознаками.

Таблиця 2.3

Атрибути сутності «BlogCategories»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор категорії блогу
2	parent_id	BIGINT UNSIGNED	Ідентифікатор батьківської категорії для формування ієрархії
3	name	VARCHAR(255)	Назва категорії, що відображається на веб-сайті
4	slug	VARCHAR(255)	Унікальний символьний ідентифікатор категорії для використання в URL
5	title	VARCHAR(255)	Заголовок категорії
6	description	TEXT	Текстовий опис категорії
7	_lft	INT	Ліва межа вузла в ієрархічній структурі категорій
8	_rgt	INT	Права межа вузла в ієрархічній структурі

Таблиця BlogPosts призначена для зберігання інформації про публікації блогу веб-сайту (табл. 2.4). Записи цієї таблиці використовуються для формування стрічки блогу, відображення окремих публікацій і керування життєвим циклом контенту.

Таблиця 2.4

Атрибути сутності «BlogPosts»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор блогової публікації
2	category_id	BIGINT UNSIGNED	Ідентифікатор категорії публікації
3	author_id	BIGINT UNSIGNED	Ідентифікатор автора публікації
4	title	VARCHAR(255)	Назва блогової публікації
5	slug	VARCHAR(255)	Символьний ідентифікатор публікації
6	content	MEDIUMTEXT	Основний текстовий вміст публікації
7	created_at	DATETIME	Дата та час створення публікації
8	updated_at	DATETIME	Дата та час останнього редагування
9	published_at	DATETIME	Дата та час оприлюднення публікації

Таблиця BlogPostTag призначена для реалізації зв’язку між публікаціями та тегами за схемою «багато-до-багатьох» (табл. 2.5). Вона забезпечує можливість призначення одній публікації кількох тегів і, навпаки, віднесення одного тегу до множини публікацій. Таке рішення використовується для підтримки механізмів тематичної індексації, формування добірок матеріалів та побудови переліків публікацій за обраним тегом.

Таблиця 2.5

Атрибути сутності «BlogPostTag»

№	Найменування	Тип даних	Призначення
1	post_id	BIGINT UNSIGNED	Ідентифікатор публікації, з якою пов’язується тег
2	tag_id	BIGINT UNSIGNED	Ідентифікатор тегу, що призначається публікації

Таблиця BlogPostComments призначена для зберігання коментарів користувачів до публікацій блогу (табл. 2.6). Вона забезпечує можливість фіксації авторства коментаря, його належності до конкретної публікації, а також підтримує ієрархічну структуру обговорень шляхом збереження відповідей на інші коментарі.

Таблиця 2.6

Атрибути сутності «BlogPostComments»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Унікальний ідентифікатор коментаря
2	author_id	BIGINT UNSIGNED	Ідентифікатор автора коментаря
3	post_id	BIGINT UNSIGNED	Ідентифікатор публікації
4	parent_id	BIGINT UNSIGNED	Ідентифікатор батьківського коментаря
5	text	TEXT	Текстовий вміст коментаря
6	active	TINYINT(1)	Ознака активності коментаря (відображається / прихований)
7	created_at	DATETIME	Дата та час створення коментаря
8	updated_at	DATETIME	Дата та час останнього редагування коментаря

Таблиця ShopBrands призначена для зберігання інформації про торгові марки товарів, представлених в інтернет-магазині (табл. 2.7). Вона використовується для групування продукції за виробниками або брендами, формування каталогу товарів, а також для реалізації фільтрації та навігації за брендовою ознакою.

Таблиця 2.7

Атрибути сутності «ShopBrands»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор бренду
2	name	VARCHAR(255)	Назва торгової марки
3	slug	VARCHAR(255)	Унікальний символьний ідентифікатор бренду для використання в URL
4	description	TEXT	Текстовий опис бренду або додаткова довідкова інформація

Таблиця ShopProducts призначена для зберігання інформації про товари, представлені в інтернет-магазині (табл. 2.8). Вона містить дані про належність товарів до відповідних категорій і брендів, їх основні характеристики, цінові показники, стан доступності та статистичні показники взаємодії користувачів. Записи цієї таблиці використовуються для формування каталогу продукції, відображення сторінок товарів і обробки замовлень.

Таблиця 2.8

Атрибути сутності «ShopProducts»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Унікальний ідентифікатор товару
2	category_id	BIGINT UNSIGNED	Ідентифікатор категорії, до якої належить товар
3	brand_id	BIGINT UNSIGNED	Ідентифікатор бренду товару
4	availability	ENUM	Статус наявності товару (в наявності / відсутній)
5	title	VARCHAR(255)	Назва товару
6	slug	VARCHAR(255)	Унікальний символьний ідентифікатор товару для формування URL
7	price	INT	Вартість товару
8	content	MEDIUMTEXT	Розширений опис товару
9	status	ENUM	Статус товару в системі (активний / чернетка)
10	reviews	INT	Кількість відгуків про товар
11	comments	INT	Кількість коментарів до товару
12	rating	DECIMAL(3,2)	Середнє значення рейтингу товару
13	weight	INT	Вага товару
14	quantity	INT	Кількість одиниць товару на складі
15	created_at	DATETIME	Дата та час створення запису про товар
16	updated_at	DATETIME	Дата та час останнього оновлення інформації про товар

Таблиця ShopCategories призначена для зберігання інформації про категорії товарів інтернет-магазину (табл. 2.9). Дані цієї таблиці використовуються під час побудови навігаційного меню, формування структури каталогу та реалізації фільтрації товарів за категоріями.

Таблиця 2.9

Атрибути сутності «ShopCategories»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор категорії товарів
2	parent_id	BIGINT UNSIGNED	Ідентифікатор батьківської категорії
3	name	VARCHAR(255)	Назва категорії, що відображається у каталозі
4	slug	VARCHAR(255)	Унікальний символьний ідентифікатор категорії для використання в URL
5	title	VARCHAR(255)	Заголовок категорії
6	description	TEXT	Опис категорії товарів
7	_lft	INT	Ліва межа вузла у вкладеній структурі
8	_rgt	INT	Права межа вузла у вкладеній структурі

Таблиця ShopProductPhotos призначена для зберігання інформації про зображення товарів інтернет-магазину (табл. 2.10). Вона забезпечує можливість прив’язки кількох фотографій до одного товару, що використовується для візуального представлення продукції в каталозі та на сторінках окремих товарів. Дані цієї таблиці застосовуються під час формування галереї зображень і підтримки мультимедійного контенту товарів.

Таблиця 2.10

Атрибути сутності «ShopProductPhotos»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор зображення товару
2	product_id	BIGINT UNSIGNED	Ідентифікатор товару, до якого належить зображення
3	photo	VARCHAR(255)	Шлях або ім’я файлу зображення товару

Таблиця ShopProductComments призначена для зберігання коментарів користувачів до товарів інтернет-магазину (табл. 2.11). Вона підтримує фіксацію авторства коментарів, їх належність до конкретного товару, а також вкладену структуру обговорень завдяки можливості відповідей на інші коментарі. Дані

цієї таблиці використовуються для організації зворотного зв’язку з користувачами та модерації коментарів у межах товарних сторінок.

Таблиця 2.11

Атрибути сутності «ShopProductComments»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор коментаря
2	author_id	BIGINT UNSIGNED	Ідентифікатор автора коментаря
3	product_id	BIGINT UNSIGNED	Ідентифікатор товару
4	parent_id	BIGINT UNSIGNED	Ідентифікатор батьківського коментаря для формування вкладених відповідей
5	text	TEXT	Текстовий вміст коментаря
6	active	TINYINT(1)	Ознака активності коментаря (відображається / прихований)
7	created_at	DATETIME	Дата та час створення коментаря
8	updated_at	DATETIME	Дата та час останнього оновлення коментаря

Таблиця ShopProductReviews призначена для зберігання відгуків користувачів про товари інтернет-магазину (табл. 2.12). Дані цієї таблиці використовуються для розрахунку рейтингу товарів, відображення відгуків на сторінках продукції та аналізу зворотного зв’язку від користувачів.

Таблиця 2.12

Атрибути сутності «ShopProductReviews»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор відгуку
2	author_id	BIGINT UNSIGNED	Ідентифікатор автора відгуку
3	product_id	BIGINT UNSIGNED	Ідентифікатор товару, до якого належить відгук
4	rating	TINYINT UNSIGNED	Оцінка товару за п’ятибальною шкалою
5	text	TEXT	Текстовий опис відгуку
6	created_at	DATETIME	Дата та час створення відгуку
7	updated_at	DATETIME	Дата та час останнього оновлення відгуку

Таблиця ShopCart призначена для зберігання інформації про вміст кошика користувачів інтернет-магазину (табл. 2.13). Вона фіксує перелік обраних товарів, їх кількість та агреговані показники вартості й ваги для кожного користувача. Дані цієї таблиці застосовуються під час формування замовлення, перерахунку підсумкової суми покупки та керування станом кошика користувача.

Таблиця 2.13

Атрибути сутності «ShopCart»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор запису кошика
2	user_id	BIGINT UNSIGNED	Ідентифікатор користувача, якому належить кошик
3	product_id	BIGINT UNSIGNED	Ідентифікатор товару, доданого до кошика
4	quantity	INT UNSIGNED	Кількість одиниць товару в кошику
5	total_price	INT UNSIGNED	Загальна вартість обраної кількості товару
6	total_weight	INT UNSIGNED	Загальна вага обраної кількості товару

Таблиця ShopCharacteristics призначена для зберігання переліку характеристик товарів інтернет-магазину (табл. 2.14). Дані цієї таблиці застосовуються під час формування карток товарів, реалізації фільтрації за параметрами та відображення технічних характеристик продукції.

Таблиця 2.14

Атрибути сутності «ShopCharacteristics»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор характеристики
2	name	VARCHAR(255)	Назва характеристики товару
3	type	ENUM	Тип значення характеристики
4	required	TINYINT(1)	Ознака обов’язковості заповнення характеристики
5	default	VARCHAR(255)	Значення характеристики за замовчуванням
6	sort	INT	Порядок відображення характеристики у переліку

Таблиця ShopProductCharacteristics призначена для зберігання значень характеристик конкретних товарів (табл. 2.15). Дані цієї таблиці використовуються для відображення параметрів товарів, порівняння продукції та реалізації механізмів фільтрації за характеристиками.

Таблиця 2.15

Атрибути сутності «ShopProductCharacteristics»

№	Найменування	Тип даних	Призначення
1	product_id	BIGINT UNSIGNED	Ідентифікатор товару, для якого задано характеристику
2	characteristic_id	BIGINT UNSIGNED	Ідентифікатор характеристики товару
3	value_int	BIGINT	Значення характеристики цілого типу
4	value_float	DECIMAL(18,6)	Значення характеристики дійсного типу
5	value_string	VARCHAR(255)	Рядкове значення характеристики

Таблиця ShopDeliveryMethods призначена для зберігання інформації про способи доставки товарів інтернет-магазину (табл. 2.16). Дані цієї таблиці використовуються для формування переліку доступних способів доставки та визначення їх застосовності залежно від параметрів замовлення.

Таблиця 2.16

Атрибути сутності «ShopDeliveryMethods»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор способу доставки
2	name	VARCHAR(255)	Назва способу доставки
3	cost	INT UNSIGNED	Вартість доставки
4	min_weight	INT UNSIGNED	Мінімальна вага замовлення для застосування способу доставки
5	max_weight	INT UNSIGNED	Максимальна вага замовлення для застосування способу доставки
6	sort	INT	Порядок відображення способів доставки

Таблиця ShopOrderCustomers призначена для зберігання контактних даних покупців, які використовуються під час оформлення замовлень в інтернет-

магазині (табл. 2.17). Вона містить узагальнену інформацію про замовника, незалежно від наявності облікового запису в системі, що дає змогу зберігати історичні дані про замовлення та забезпечувати гнучкість процесу оформлення покупки. Дані цієї таблиці застосовуються для обробки замовлень, зв'язку з клієнтами та формування супровідної документації.

Таблиця 2.17

Атрибути сутності «ShopOrderCustomers»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор даних замовника
2	full_name	VARCHAR(255)	Повне ім'я замовника
3	phone	VARCHAR(50)	Контактний номер телефону замовника
4	email	VARCHAR(255)	Адреса електронної пошти замовника
5	created_at	DATETIME	Дата та час створення запису
6	updated_at	DATETIME	Дата та час останнього оновлення даних

Таблиця ShopOrderDeliveries призначена для зберігання інформації про адресу доставки замовлень інтернет-магазину (табл. 2.18). Вона містить узагальнені дані щодо місця отримання товарів, що дає змогу відокремити логістичну інформацію від основних параметрів замовлення та забезпечити повторне використання даних у межах різних операцій. Дані цієї таблиці застосовуються під час обробки замовлень і формування доставки товарів.

Таблиця 2.18

Атрибути сутності «ShopOrderDeliveries»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор даних доставки
2	city	VARCHAR(255)	Назва населеного пункту доставки
3	address	VARCHAR(255)	Адреса доставки замовлення
4	postal_code	VARCHAR(50)	Поштовий індекс адреси доставки
5	created_at	DATETIME	Дата та час створення запису
6	updated_at	DATETIME	Дата та час останнього оновлення даних доставки

Таблиця ShopOrders призначена для зберігання інформації про замовлення, оформлені користувачами інтернет-магазину (табл. 2.19). Вона об’єднує дані про користувача, замовника, параметри доставки, спосіб оплати, загальну вартість замовлення та поточний стан його обробки. Записи цієї таблиці використовуються для керування життєвим циклом замовлення, відстеження його статусів і формування історії покупок користувачів.

Таблиця 2.19

Атрибути сутності «ShopOrders»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор замовлення
2	user_id	BIGINT UNSIGNED	Ідентифікатор користувача, який оформив замовлення
3	customer_data_id	BIGINT UNSIGNED	Ідентифікатор даних замовника
4	delivery_data_id	BIGINT UNSIGNED	Ідентифікатор даних доставки
5	delivery_method_id	BIGINT UNSIGNED	Ідентифікатор способу доставки
6	delivery_method_name	VARCHAR(255)	Назва обраного способу доставки
7	delivery_method_cost	INT UNSIGNED	Вартість доставки
8	payment_method	VARCHAR(100)	Обраний спосіб оплати
9	cost	INT UNSIGNED	Загальна вартість замовлення
10	note	TEXT	Додаткові примітки до замовлення
11	current_status	VARCHAR(100)	Поточний статус замовлення
12	cancel_reason	TEXT	Причина скасування замовлення
13	created_at	DATETIME	Дата та час створення замовлення
14	updated_at	DATETIME	Дата та час останнього оновлення замовлення

Таблиця ShopOrderItems призначена для зберігання інформації про склад замовлень інтернет-магазину (табл. 2.20). Вона містить перелік товарів, що входять до конкретного замовлення, із фіксацією їх кількості, вартості на момент оформлення та підсумкової ціни. Дані цієї таблиці використовуються для детального відображення замовлення, формування фінансових розрахунків і збереження історичних даних щодо придбаної продукції.

Таблиця 2.20

Атрибути сутності «ShopOrderItems»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор позиції замовлення
2	order_id	BIGINT UNSIGNED	Ідентифікатор замовлення
3	product_id	BIGINT UNSIGNED	Ідентифікатор товару
4	product_name	VARCHAR(255)	Назва товару на момент оформлення замовлення
5	product_price	INT UNSIGNED	Вартість одиниці товару на момент замовлення
6	product_quantity	INT UNSIGNED	Кількість одиниць товару в замовленні
7	total_price	INT UNSIGNED	Загальна вартість позиції замовлення

Таблиця ShopOrderStatuses призначена для зберігання історії змін статусів замовлень інтернет-магазину (табл. 2.21). Вона фіксує послідовність станів, які проходить замовлення в процесі обробки, що дає змогу відстежувати його життєвий цикл і аналізувати динаміку виконання. Дані цієї таблиці застосовуються для контролю стану замовлень і формування журналу їх обробки.

Таблиця 2.21

Атрибути сутності «ShopOrderStatuses»

№	Найменування	Тип даних	Призначення
1	id	BIGINT UNSIGNED	Ідентифікатор запису статусу
2	order_id	BIGINT UNSIGNED	Ідентифікатор замовлення
3	value	VARCHAR(100)	Назва статусу замовлення
4	created_at	DATETIME	Дата та час фіксації статусу

На основі визначених сутностей та їх атрибутів було сформовано структуровану схему зберігання даних, що відображає логіку взаємодії основних компонентів веб-сайту інтернет-магазину. Узгодження таблиць між собою через первинні та зовнішні ключі забезпечує коректне представлення зв’язків між користувачами, товарами, замовленнями та допоміжними довідковими даними.

Відповідна фізична модель бази даних, побудована на основі наведених таблиць, подана на рис. 2.3 та ілюструє реалізацію структури зберігання даних на рівні конкретної СУБД.

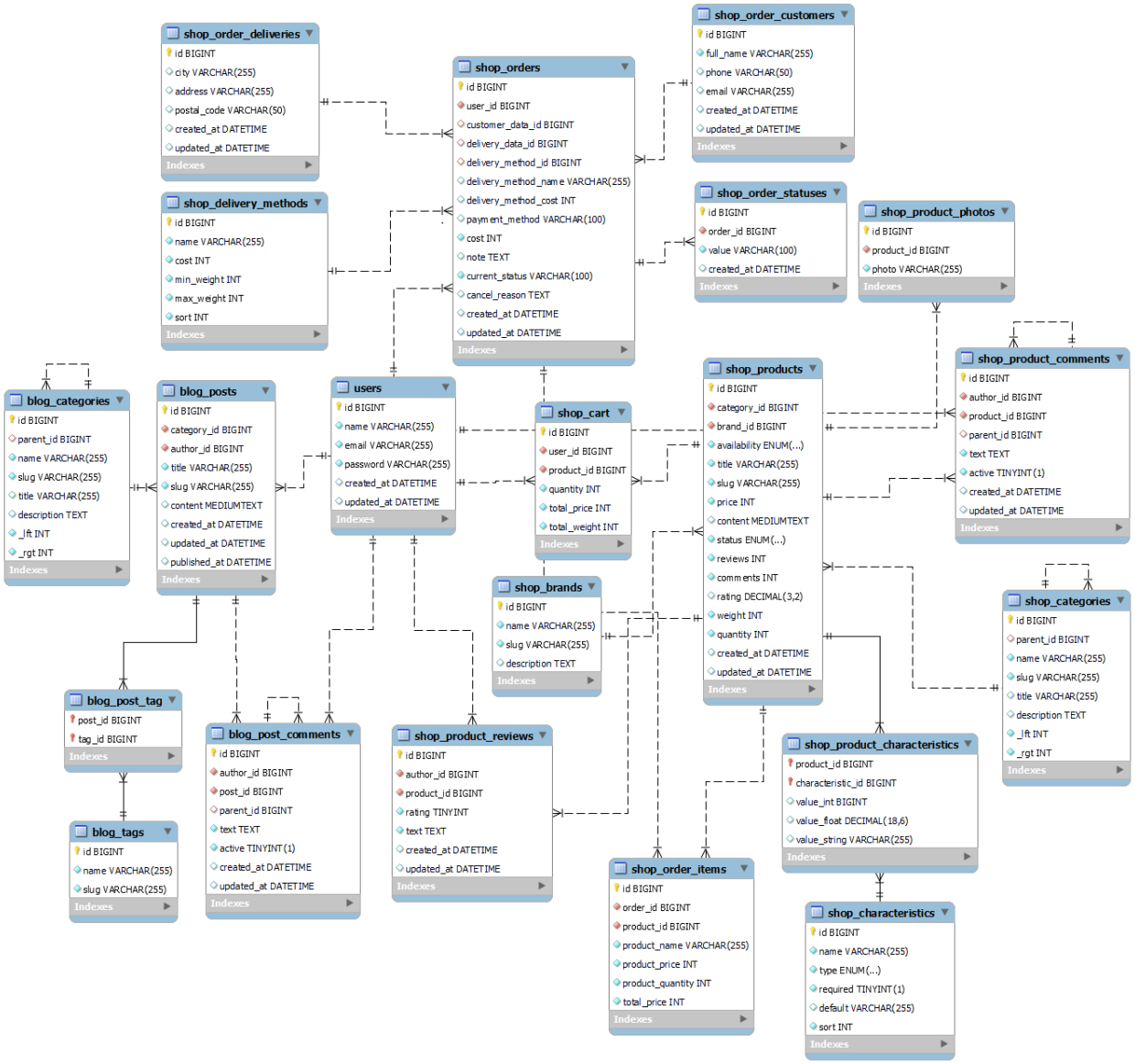


Рис. 2.3. Фізична модель бази даних

Побудова фізичної моделі бази даних дає змогу формалізувати структуру зберігання інформації на рівні конкретної СУБД з урахуванням типів даних, обмежень цілісності та механізмів зв’язку між таблицями. Така модель забезпечує узгоджену роботу прикладної частини веб-сайту з базою даних і слугує основою для реалізації операцій збереження, вибірки та оновлення даних.

Окрім того, фізична модель використовується як орієнтир під час подальшого розширення функціональності системи та супроводу програмного забезпечення.

2.3. Проєктування архітектури веб-сайту

Проєктування архітектури веб-сайту інтернет-магазину є одним із визначальних етапів розробки, оскільки саме архітектурні рішення забезпечують узгоджену роботу всіх компонентів системи, підтримку бізнес-процесів та можливість подальшого розширення функціоналу. Раціонально обрана архітектура дозволяє розділити відповідальність між окремими частинами програмного забезпечення, спростити супровід коду та підвищити надійність і масштабованість веб-сайту.

Для побудови архітектури веб-сайту обрано модель MVC, яка є поширеним і перевіреним підходом до організації серверних веб-застосунків. На рис. 2.4 наведено узагальнену схему MVC-архітектури системи, що відображає основні компоненти та напрямки взаємодії між ними.

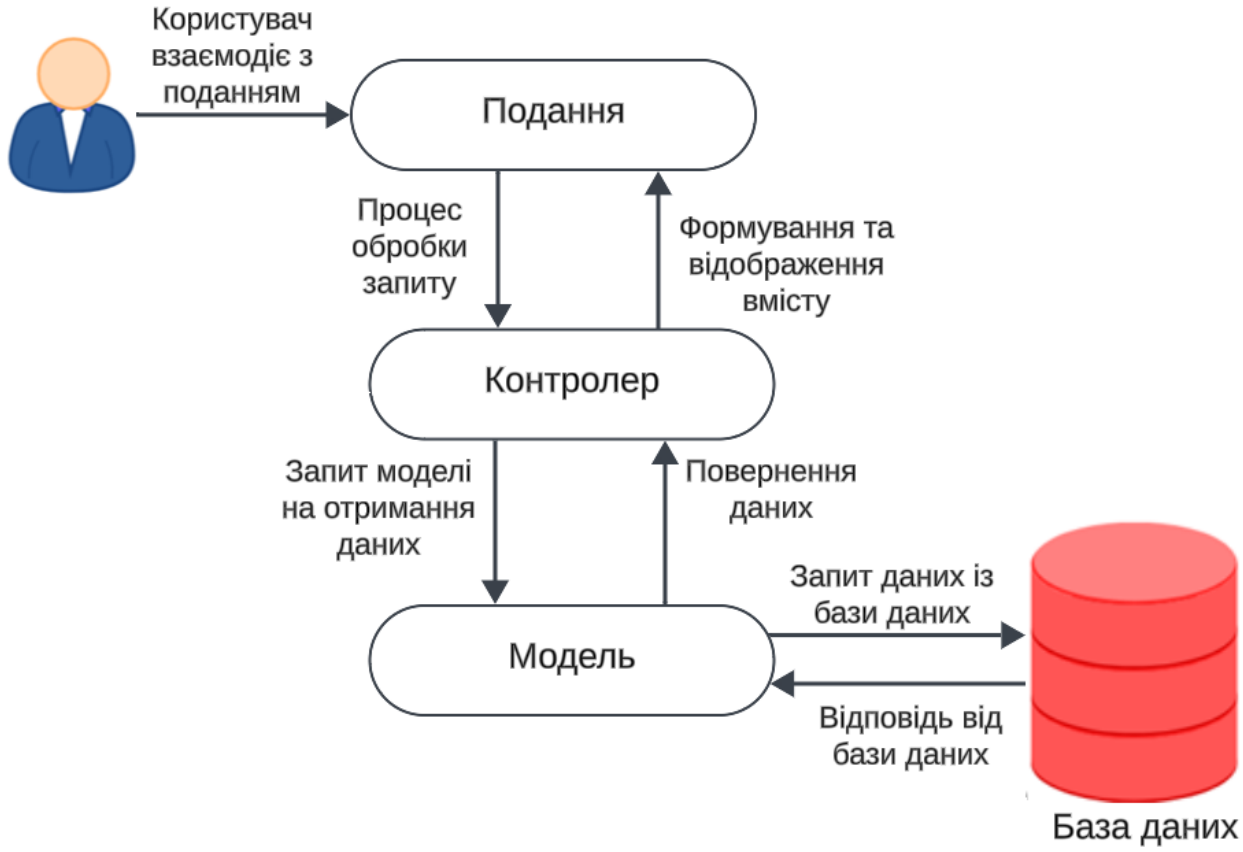


Рис. 2.4. MVC архітектура системи

Відповідно до наведеної схеми користувач взаємодіє з веб-сайтом через рівень подання, який відповідає за відображення веб-сторінок і приймання вхідних дій користувача. Подання ініціює процес обробки запиту, передаючи керування контролеру, який виступає центральною ланкою логіки застосунку. Контролер аналізує отриманий запит, визначає необхідні дії та формує звернення до моделі з метою отримання або зміни даних. Модель реалізує бізнес-логіку системи та забезпечує доступ до бази даних, надсилаючи запити на зчитування або збереження інформації та отримуючи відповідь від сховища даних. Отримані результати передаються назад контролеру, який, у свою чергу, ініціює формування подання з актуальним вмістом для користувача. Таким чином, MVC-архітектура забезпечує чітке розмежування між представленням даних, логікою обробки запитів і роботою з базою даних, що сприяє підвищенню структурованості коду, спрощенню тестування та підвищенню зручності подальшого розвитку веб-сайту інтернет-магазину.

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ВЕБ-САЙТУ

3.1. Розробка клієнтської частини веб-сайту

Розробка клієнтської частини веб-сайту інтернет-магазину спрямована на забезпечення зручної, інтуїтивно зрозумілої та візуально узгодженої взаємодії користувача з функціоналом системи. Саме клієнтська частина формує перше враження про веб-сайт, визначає зручність навігації, доступність основних дій та ефективність виконання бізнес-процесів з точки зору кінцевого користувача. Під час проєктування інтерфейсу особлива увага приділяється адаптивності сторінок, логічному розміщенню елементів керування та зрозумілості форм введення даних, що є критично важливим для процесів реєстрації та автентифікації користувачів.

Одним із елементів клієнтської частини веб-сайту є сторінка реєстрації, яка забезпечує створення облікового запису користувача та подальший доступ до персоналізованих можливостей інтернет-магазину. На рис. 3.1 наведено реалізацію сторінки реєстрації користувача веб-сайту.

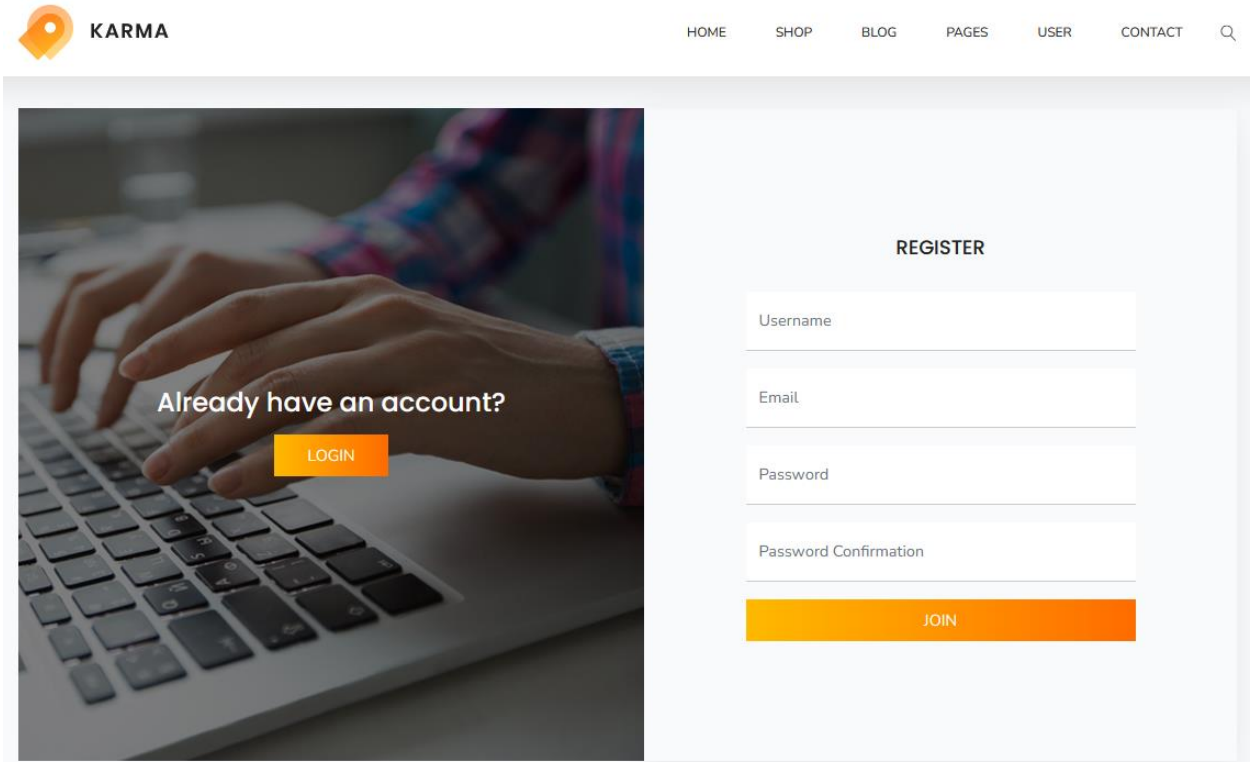


Рис. 3.1. Сторінка реєстрації користувача

Сторінка реєстрації має структурований та мінімалістичний інтерфейс, який поділено на дві функціональні області. У формі передбачені поля для введення імені користувача, адреси електронної пошти, пароля та підтвердження пароля, що забезпечує коректність введення та зменшує ймовірність помилок під час створення облікового запису. Кнопка підтвердження реєстрації розміщена внизу форми та візуально виділена, що акцентує увагу користувача на завершальному етапі процесу.

На рис. 3.2 наведено сторінку авторизації користувача веб-сайту інтернет-магазину, яка забезпечує доступ до персоналізованого функціоналу системи після успішної ідентифікації облікового запису.

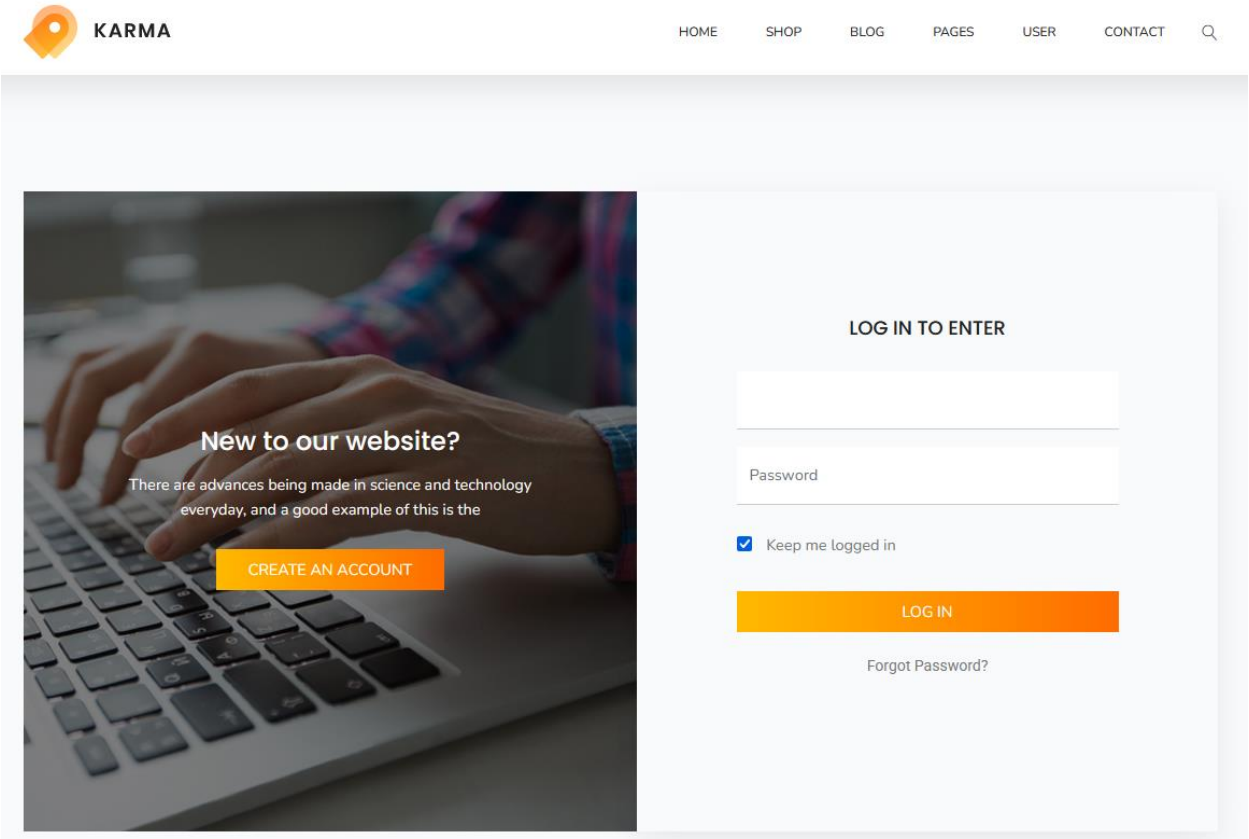


Рис. 3.2. Сторінка авторизації користувача

Форма авторизації має мінімалістичний дизайн, що зосереджує увагу користувача виключно на процесі введення даних та виконання дії входу. Кнопка підтвердження авторизації візуально виділена та розміщена у нижній частині форми, забезпечуючи інтуїтивно зрозумілий сценарій взаємодії. Також на сторінці передбачено посилання для відновлення доступу у разі втрати пароля,

що сприяє підвищенню надійності та зручності використання системи. Загалом реалізація сторінки авторизації відповідає вимогам до сучасних веб-інтерфейсів інтернет-магазинів, забезпечуючи поєднання безпеки, простоти використання та візуальної узгодженості з іншими елементами клієнтської частини веб-сайту.

Рис. 3.3 відображає головну сторінку веб-сайту інтернет-магазину, яка виконує ключову роль у представленні асортименту товарів та формуванні первинної взаємодії користувача з системою. Сторінка є точкою входу для більшості відвідувачів і безпосередньо впливає на зручність навігації, швидкість пошуку товарів і загальне сприйняття веб-сайту.

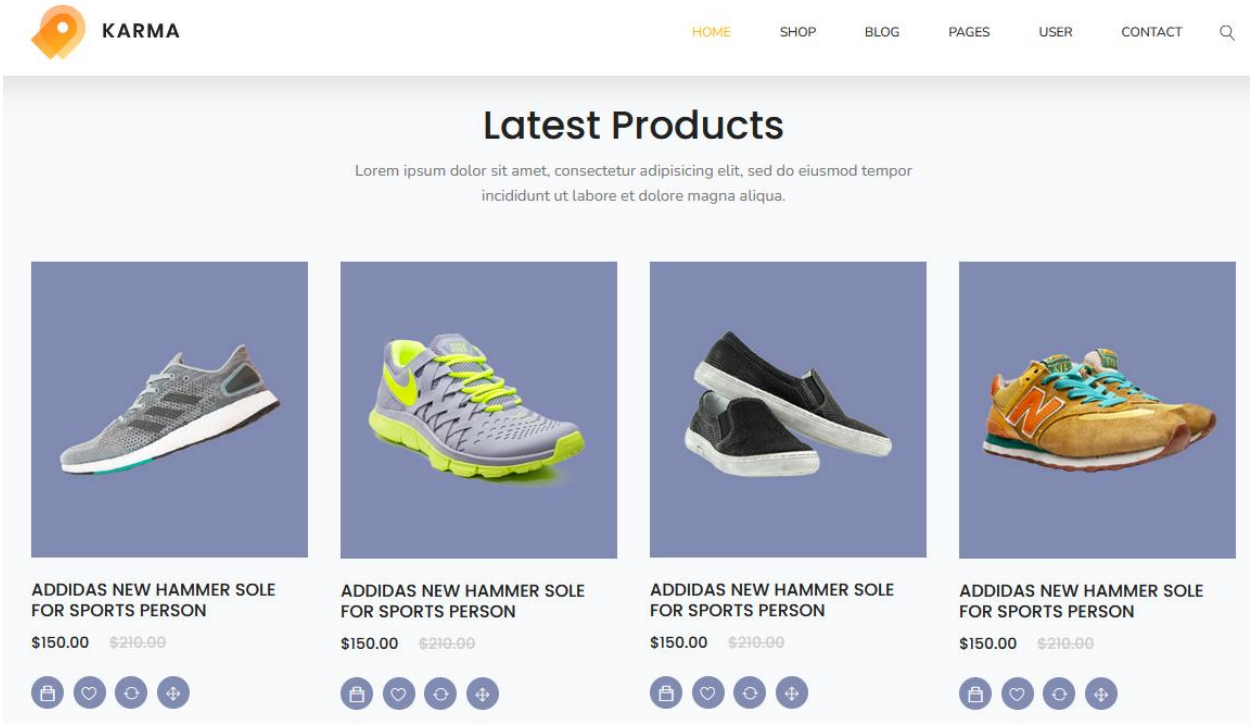


Рис. 3.3. Головна сторінка інтернет-магазину з переліком товарів

Інтерфейс головної сторінки побудований за принципом візуальної ієрархії та орієнтації на користувацький досвід. У верхній частині сторінки розміщено навігаційне меню, яке забезпечує доступ до основних розділів веб-сайту, зокрема каталогу товарів, блогу, інформаційних сторінок та користувацького профілю. Центральну частину сторінки займає блок із переліком актуальних товарів, оформлений у вигляді сітки, що дозволяє одночасно відображати кілька позицій та спрощує порівняння пропозицій. Кожна картка товару містить зображення,

назву, поточну ціну та, за наявності, попередню ціну, що дає змогу наочно продемонструвати знижки та акційні пропозиції.

На рис. 3.4 представлено вітрину інтернет-магазину, яка слугує розширенням головної сторінки та виконує функцію візуального представлення переваг сервісу й акційних пропозицій. Даний елемент клієнтської частини веб-сайту орієнтований на привернення уваги користувача, формування довіри до магазину та стимулювання подальшої навігації по каталогу товарів.

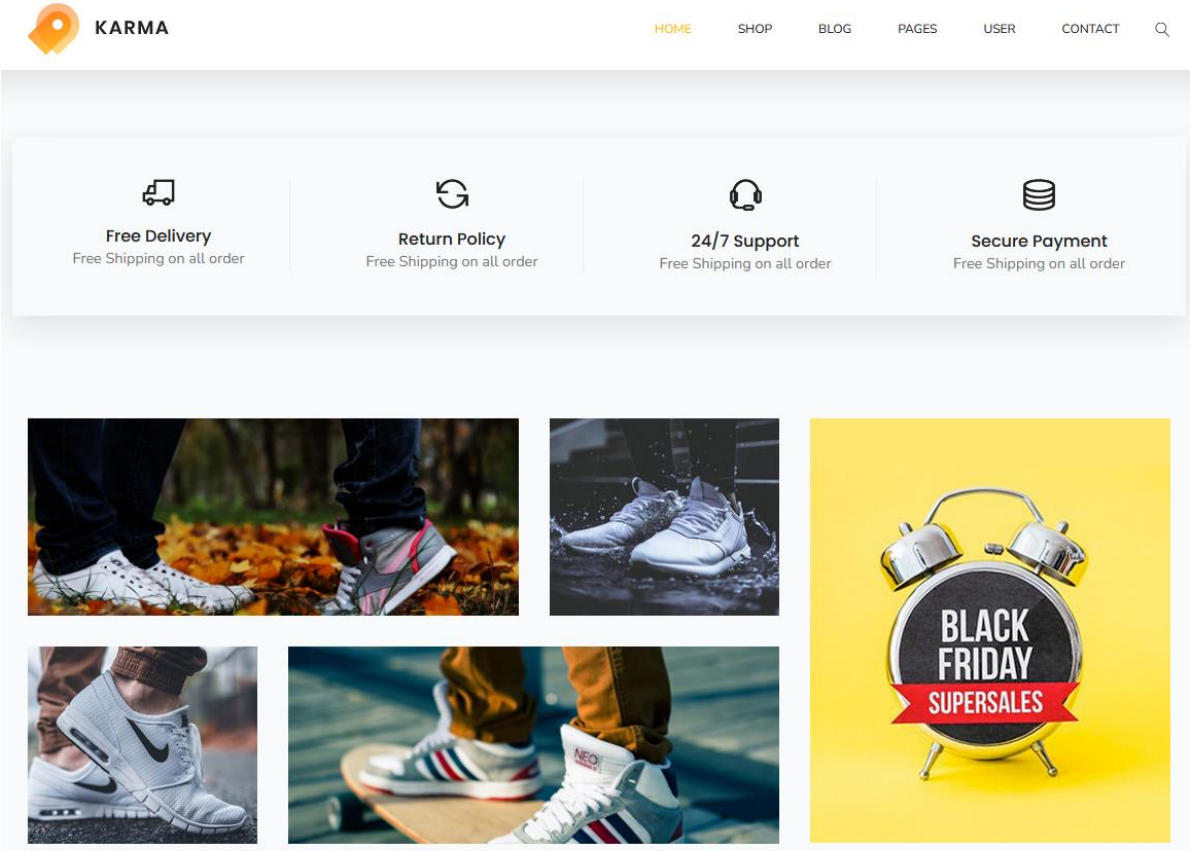


Рис. 3.4. Вітрина інтернет-магазину

У верхній частині вітрини розміщено блок із ключовими сервісними перевагами інтернет-магазину, такими як безкоштовна доставка, можливість повернення товарів, цілодобова підтримка користувачів і безпечні платіжні операції. Подання цієї інформації у вигляді іконок із короткими текстовими поясненнями дозволяє швидко донести основні переваги сервісу та підвищити рівень довіри з боку відвідувачів. Нижче розташовано візуальний блок із тематичними зображеннями товарів, який виконує роль промо-вітрини та акцентує увагу на актуальних категоріях асортименту.

Окрему увагу на вітрині приділено акційним пропозиціям, що відображаються у вигляді яскравих банерів із рекламними повідомленнями, зокрема сезонними розпродажами. Таке дизайнерське рішення дозволяє виділити спеціальні пропозиції серед загального контенту сторінки та стимулює користувача до перегляду акційних товарів. Загальна структура вітрини поєднує інформативність і візуальну привабливість, забезпечуючи логічне продовження користувацького сценарію від ознайомлення з сервісом до вибору конкретних товарів і подальшого оформлення замовлення.

Рис. 3.5 висвітлює сторінку каталогу продукції інтернет-магазину, яка забезпечує користувачеві розширені можливості перегляду та відбору товарів відповідно до заданих критеріїв. Дана сторінка є основним інструментом навігації по асортименту та відіграє ключову роль у процесі вибору товарів перед їх додаванням до кошика.

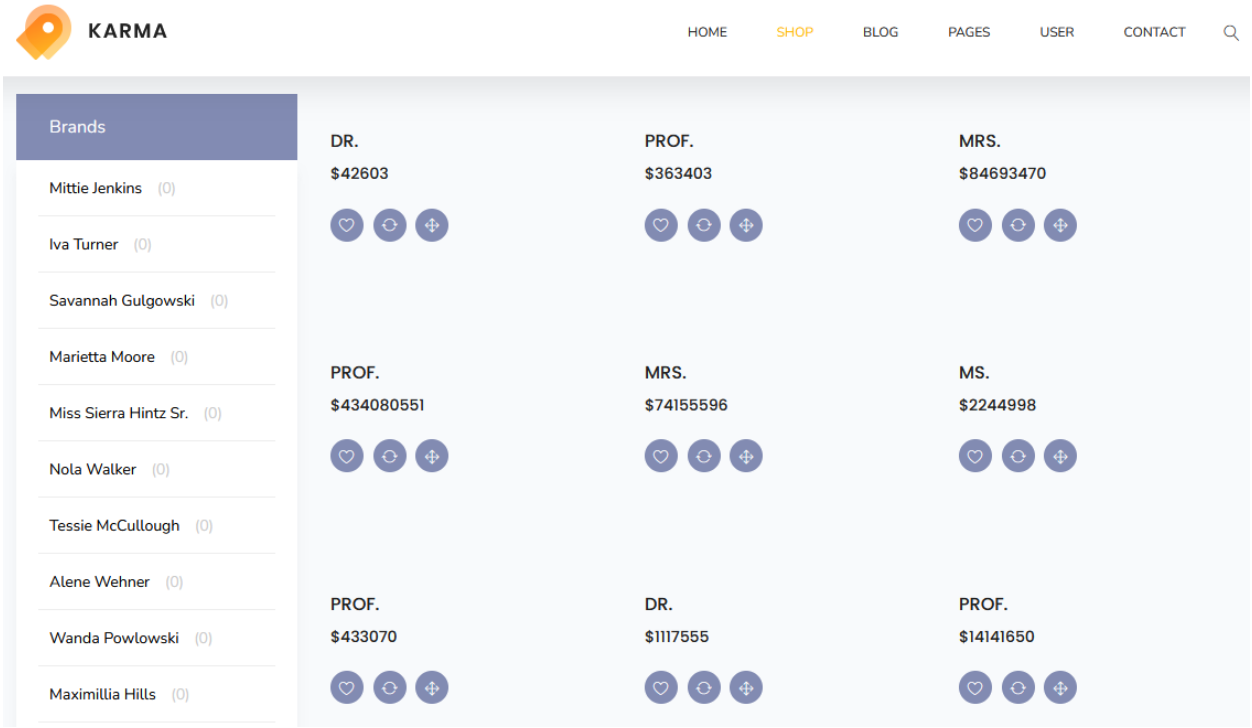


Рис. 3.5. Каталог продукції інтернет-магазину

Інтерфейс каталогу побудований з урахуванням принципів зручності та структурованості подання інформації. У лівій частині сторінки розміщено блок фільтрації, який дозволяє здійснювати відбір товарів за брендами, що спрощує пошук необхідних позицій у межах великого асортименту. Центральна частина

сторінки містить сітку товарних карток, кожна з яких відображає основну інформацію про товар, зокрема назву та вартість. Під кожною карткою передбачено інтерактивні елементи, що забезпечують швидкий доступ до типових дій, таких як додавання товару до кошика, перегляд детальної інформації або збереження до списку обраного.

На рис. 3.6 наведено сторінку публікацій веб-сайту інтернет-магазину, яка виконує інформаційну та комунікаційну функцію в межах клієнтської частини системи. Дана сторінка призначена для розміщення тематичних матеріалів, новин, оглядів та інших інформаційних публікацій, що сприяють формуванню інтересу користувачів до бренду та підвищенню рівня залученості аудиторії.

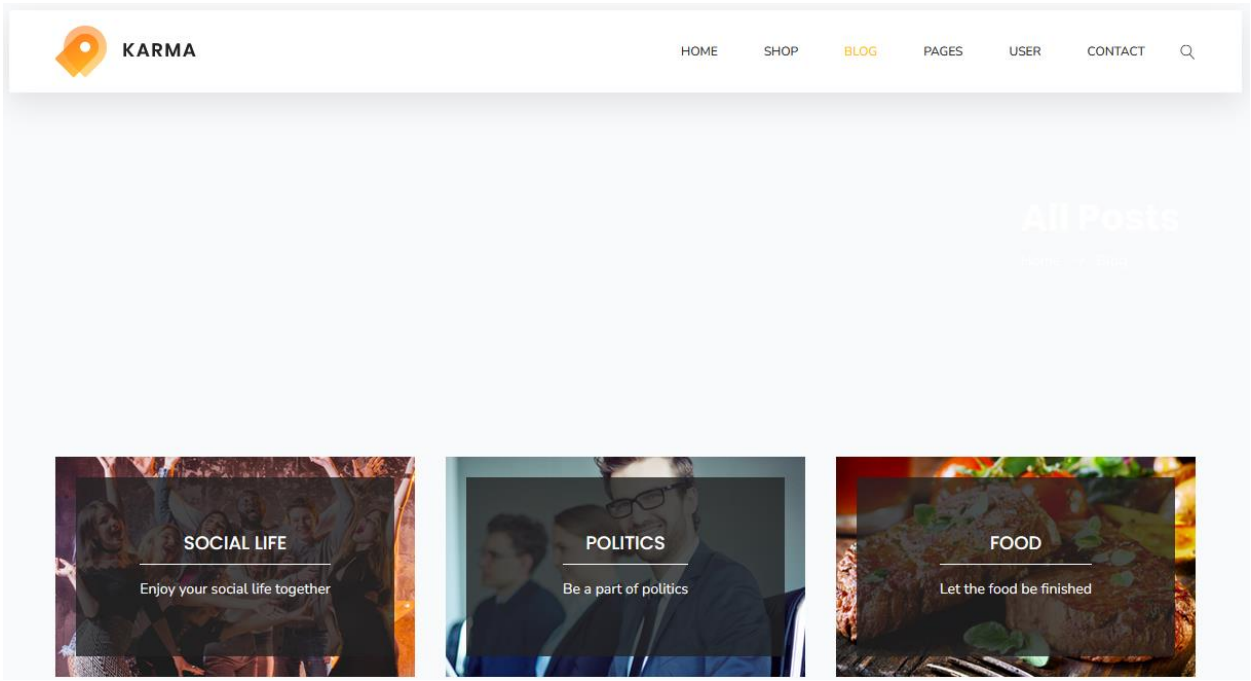


Рис. 3.6. Сторінка публікацій веб-сайту

Інтерфейс сторінки публікацій побудований за принципом візуального групування контенту та орієнтації на швидке сприйняття інформації. Основна частина сторінки представлена у вигляді тематичних блоків, кожен з яких містить зображення, заголовок та короткий опис відповідної категорії публікацій. Це дозволяє користувачеві швидко зорієнтуватися у доступному контенті та перейти до матеріалів, що відповідають його інтересам.

Навігаційне меню, розміщене у верхній частині сторінки, забезпечує зручний перехід між основними розділами веб-сайту, зберігаючи єдину

стилістику інтерфейсу. Візуальне оформлення публікацій поєднує інформативність із привабливим дизайном, що стимулює перегляд матеріалів і сприяє довшому перебуванню користувача на веб-сайті. Реалізація сторінки публікацій доповнює функціонал інтернет-магазину, забезпечуючи не лише комерційну, а й інформаційну складову взаємодії з користувачами.

3.2. Розробка серверної частини веб-сайту

Розробка серверної частини веб-сайту інтернет-магазину спрямована на реалізацію бізнес-логіки системи, обробку запитів від клієнтської частини, взаємодію з базою даних та забезпечення безпеки оброблення користувацьких даних. Саме серверний рівень відповідає за коректне виконання основних бізнес-процесів, зокрема реєстрацію та авторизацію користувачів, керування товарами, формування замовлень і контроль доступу до функціоналу веб-сайту. У межах даної роботи серверна частина реалізована з використанням об’єктно-орієнтованого підходу та архітектурної моделі MVC, що забезпечує чітке розмежування відповідальності між компонентами системи.

Для організації доступу до даних і реалізації логіки їх обробки було розроблено відповідні класи, які інкапсулюють операції створення, збереження та керування сутностями предметної області. Зокрема, для роботи з обліковими записами користувачів реалізовано методи, що забезпечують коректну реєстрацію нових користувачів із дотриманням вимог безпеки та цілісності даних. На рис. 3.6 наведено програмний код, який реалізує процес реєстрації користувача на серверній стороні веб-сайту.

```php
public static function register(string $name,
string $email, string $password): self
{
    return static::create([
        'name' => $name,
        'email' => $email,
        'password' => bcrypt($password),
        'verify_token' => null,
        'status' => self::STATUS_ACTIVE,
        'role' => self::ROLE_USER
    ]);
}
```

Рис. 3.7. Функція реєстрації користувача

Функція приймає ім’я користувача, адресу електронної пошти та пароль, після чого формує новий запис у базі даних. У процесі створення облікового запису пароль користувача обробляється за допомогою криптографічної функції хешування, що забезпечує захист конфіденційних даних у разі несанкціонованого доступу до бази даних. Додатково для нового користувача встановлюється активний статус облікового запису та стандартна роль, що визначає рівень доступу до функціоналу веб-сайту.

На рис. 3.8 наведена функція, яка забезпечує зміну ролі користувача в системі інтернет-магазину з урахуванням визначених правил доступу та обмежень. Перед виконанням оновлення здійснюється перевірка коректності переданої ролі, що дозволяє запобігти призначенню невизначених або недопустимих значень.

```php
public function changeRole($role): void
{
    if (!in_array($role, [
        self::ROLE_USER,
        self::ROLE_MANAGER,
        self::ROLE_ADMIN
    ], true)) {
        throw new InvalidArgumentException('Undefined role "' . $role . '"');
    }
    if ($this->role === $role) {
        throw new DomainException('Role is already assigned.');
    }
    $this->update([
        'role' => $role
    ]);
}
```

Рис. 3.8. Функція створення нового облікового запису користувача

Додатково контролюється ситуація, коли нова роль збігається з поточною, що унеможливлює виконання зайвих операцій та забезпечує логічну цілісність даних. У разі успішного проходження перевірок роль користувача оновлюється у сховищі даних, що дозволяє динамічно керувати правами доступу та функціональними можливостями користувачів у межах веб-сайту.

Рис. 3.9 висвітлює функцію, що призначена для створення нового запису сторінки в системі керування вмістом веб-сайту інтернет-магазину. Вона формує структурований об’єкт сторінки на основі переданих параметрів, які визначають назву, унікальний ідентифікатор для URL-адреси, відображувану назву в меню, ієрархічну належність, а також опис і основний текстовий вміст.

```php
public static function new(
    $title, $slug,
    $menuTitle, $parentId,
    $description, $content
): self
{
    return static::create([
        'title' => $title,
        'slug' => $slug,
        'menu_title' => $menuTitle,
        'parent_id' => $parentId,
        'description' => $description,
        'content' => $content,
    ]);
}
```

Рис. 3.9. Створення нового запису сторінки

Використання параметра батьківської сторінки забезпечує можливість побудови вкладеної структури навігації, що підвищує зручність організації інформаційних розділів веб-сайту. Така реалізація сприяє централізованому керуванню сторінками та забезпечує узгодженість подання контенту в межах серверної частини системи.

Функція на рис. 3.10 реалізує механізм взаємодії користувача з публікаціями веб-сайту шляхом додавання позначки вподобання та призначена для обробки асинхронних запитів з клієнтської частини. Перед виконанням основної логіки забезпечується контроль доступу, що гарантує можливість виконання дії лише автентифікованими користувачами системи.

```php
public function like(Request $request)
{
    $this->middleware('auth');

    if ($request->ajax()) {
        $postId = $request->post('id');
        $post = $this->postService->findPostById($postId);

        $likes = $this->postService->like($post->id, Auth::guard()->id());

        return $likes;
    }
}
```

Рис. 3.10. Додавання вподобання товару

У межах асинхронного запиту отримується ідентифікатор відповідної публікації, після чого серверна частина звертається до сервісного шару для пошуку об’єкта та фіксації дії користувача з урахуванням його облікового запису. Результатом виконання функції є повернення актуальної кількості вподобань, що дозволяє клієнтській частині динамічно оновлювати інтерфейс без перезавантаження сторінки та підвищує інтерактивність веб-сайту.

На рис. 3.11 наведено функцію, яка забезпечує обробку процесу додавання коментаря до публікації на веб-сайті інтернет-магазину та орієнтована на взаємодію автентифікованих користувачів із контентом. Перед виконанням основної логіки застосовується перевірка доступу, що гарантує можливість залишати коментарі лише користувачам, які увійшли до системи.

```php
public function comment(CreateRequest $request, Post $post)
{
    $this->middleware('auth');

    $this->commentService->create($request, $post, Auth::guard()->id());

    return redirect()->route('blog.posts.single', [
        'id' => $post->id,
        'slug' => $post->slug,
        '#comments'
    ]);
}
```

Рис. 3.11. Додавання коментаря до публікації на веб-сайті

У процесі виконання функції серверна частина передає перевірені дані запиту, об’єкт відповідної публікації та ідентифікатор користувача до сервісного шару, де здійснюється створення нового коментаря та його збереження. Після

успішного додавання коментаря користувач перенаправляється на сторінку перегляду публікації з автоматичним переходом до блоку коментарів, що забезпечує зручність користування та наочне підтвердження виконаної дії.

Функція на рис. 3.12 реалізує завершальний етап обробки замовлення в адміністративній частині веб-сайту інтернет-магазину та відповідає за фіксацію факту успішної оплати.

```
public function pay(Order $order)
{
    $this->service->makePaid($order);

    return redirect()->route('admin.shop.orders.show', compact('order'));
}
```

Рис. 3.12. Код функції для оплати

У процесі виконання функції серверна частина звертається до сервісного шару, який змінює стан замовлення на оплачений відповідно до внутрішніх бізнес-правил системи. Після оновлення статусу замовлення користувач перенаправляється на сторінку детального перегляду відповідного замовлення, що дозволяє одразу перевірити коректність виконаної операції та актуальний стан даних.

3.3. Тестування реалізованого функціоналу

Перевірка працездатності та коректності функціонування розробленого веб-сайту інтернет-магазину є обов’язковим етапом завершення процесу його створення, оскільки дозволяє оцінити відповідність програмної реалізації визначеним функціональним вимогам і сценаріям використання. Проведення тестування спрямоване на виявлення можливих помилок у роботі серверної та клієнтської частин, перевірку стабільності виконання основних бізнес-процесів і правильності обробки користувацьких даних у типових умовах експлуатації.

У межах даного підрозділу основна увага зосереджується на перевірці реалізованого функціоналу веб-сайту, зокрема процесів реєстрації та авторизації користувачів, перегляду каталогу товарів, додавання продукції до кошика, оформлення та обробки замовлень, а також взаємодії з адміністративною

частиною системи. Окремо аналізується коректність відображення інтерфейсу, узгодженість переходів між сторінками та відсутність критичних збоїв під час виконання основних користувацьких сценаріїв.

Результати тестування дозволяють зробити узагальнену оцінку надійності та готовності розробленого веб-сайту до практичного використання, а також сформувати підґрунтя для подальшого вдосконалення та розширення його функціональних можливостей.

Тестовий приклад: №1

Призначення: перевірити коректність виконання процесу реєстрації користувача через сторінку «Register» веб-сайту з підтвердженням правильності відображення елементів форми, валідації введених даних, створення нового облікового запису та переходу користувача до наступного логічного стану системи після успішної реєстрації.

Тест-вимоги, що перевіряються: відповідність інтерфейсу сторінки реєстрації проєктним рішенням (наявність полів Username, Email, Password, Password Confirmation та кнопки Join); коректність клієнтської та серверної валідації обов’язкових полів; забезпечення узгодженості значень Password і Password Confirmation; перевірка унікальності Email під час створення облікового запису; коректне збереження користувача в базі даних із захищеним зберіганням пароля; відсутність збоїв під час виконання операції та коректне повідомлення системи про результат.

Передумови для тесту: веб-сайт запущено у штатному режимі; встановлено коректне з’єднання з базою даних; у базі даних відсутній користувач із тестовою електронною адресою; користувач не авторизований у системі; сторінка реєстрації доступна через пункт меню USER або через пряме посилання; браузер підтримує виконання JavaScript та відображення стилів сторінки.

Критерій проходження тесту: після заповнення полів валідними даними та натискання кнопки Join система повинна успішно створити нового користувача, зберегти його у базі даних, відобразити підтвердження успішної

реєстрації або виконати логічний перехід (наприклад, перенаправлення на сторінку входу/профілю) без появи помилок; у разі некоректного введення даних система повинна блокувати реєстрацію та відобразити зрозумілі повідомлення про помилки біля відповідних полів.

Таблиця 3.1

Тестування сценарію реєстрації користувача

№ з/п	Крок сценарію	Очікуваний результат	Отриманий результат	Відмітка (Так/Ні)
1	Відкрити головну сторінку веб-сайту та перейти на сторінку реєстрації користувача	Відкривається сторінка «Register», інтерфейс відображено коректно	Сторінку відкрито, елементи інтерфейсу відображені	Так
2	Перевірити наявність та підписи елементів форми реєстрації	Відображено поля Username, Email, Password, Password Confirmation та кнопку Join	Усі поля та кнопка наявні, підписи відповідають макету	Так
3	У полі Username ввести тестове значення, наприклад TestUser01	Значення відображається у полі без спотворень, курсор працює коректно	Значення введено коректно	Так
4	У полі Email ввести валідну адресу, наприклад test.user01@gmail.com	Значення приймається полем, формат коректний	Email введено коректно	Так

Тестовий приклад: №2

Призначення: перевірити коректність відображення головної сторінки інтернет-магазину з переліком актуальних товарів у блоці «Latest Products», забезпечивши повноту виводу даних, правильність структури товарних карток, узгодженість відображення цін і знижок та коректну роботу інтерактивних елементів товарної картки.

Тест-вимоги, що перевіряються: коректне завантаження та відображення списку товарів з бази даних; наявність у картці товару основних атрибутів (зображення, назва, поточна ціна); коректне відображення акційної ціни та попередньої ціни (за наявності знижки); коректна робота іконок взаємодії з товаром (додавання до кошика, додавання до обраного/порівняння/перегляд); відсутність порушення верстки при відображенні декількох карток; стабільність роботи сторінки при повторному завантаженні.

Передумови для тесту: веб-сайт запущено у штатному режимі; база даних містить щонайменше 4 актуальні товари з коректно заповненими полями назви, ціни та зображення; для частини товарів у базі задано попередню ціну або розмір знижки; з’єднання з базою даних встановлене коректно; користувач відкриває головну сторінку з браузера; кеш браузера очищено або виконано примусове оновлення сторінки.

Критерій проходження тесту: після відкриття головної сторінки система повинна відобразити блок «Latest Products» із товарними картками, кожна з яких містить зображення, назву та ціну, а за наявності знижки – також попередню ціну; інтерактивні елементи карток мають бути доступними, сторінка не повинна містити помилок завантаження або порушень верстки.

Таблиця 3.2

Тестування сценарію відображення переліку товарів на головній сторінці

№ з/п	Крок сценарію	Очікуваний результат	Отриманий результат	Відмітка (Так/Ні)
1	2	3	4	5
1	Відкрити головну сторінку інтернет-магазину	Відкривається головна сторінка, блок «Latest Products» присутній	Сторінка відкрилася, блок відображено	Так

Продовження табл. 3.3

1	2	3	4	5

2	Перевірити наявність заголовка блоку та його розміщення	Заголовок «Latest Products» відображено коректно, розташований над переліком товарів	Заголовок відображено коректно	Так
3	Перевірити кількість відображених товарних карток	Відображено не менше 4 товарних карток у вигляді сітки	Відображено 4 картки	Так
4	Перевірити вміст товарної картки	Кожна картка містить зображення, назву товару та поточну ціну	Дані присутні та відображені коректно	Так
5	Перевірити відображення знижки (за наявності)	Для акційних товарів відображається стара ціна, що візуально відрізняється від поточної	Стара ціна відображена коректно	Так
6	Перевірити відсутність порушення верстки	Картки вирівняні, елементи не накладаються, немає «зсувів» блоків	Верстка коректна	Так
7	Натиснути іконку додавання товару до кошика на одній із карток	Система додає товар до кошика та відображає підтвердження/зміну індикатора кошика	Товар додано, індикатор оновився	Так

Тестовий приклад: №3

Призначення: перевірити коректність переходу з категорійних блоків сторінки публікацій до відповідних сторінок/фільтрів публікацій, забезпечивши правильність маршрутизації та відображення контенту після переходу.

Тест-вимоги, що перевіряються: клікабельність блоків SOCIAL LIFE, POLITICS, FOOD; коректність формування URL/маршруту; завантаження сторінки списку публікацій відповідної категорії; відсутність помилок 404/500; коректне повернення назад до сторінки «All Posts».

Передумови для тесту: сторінки категорій блогу існують та налаштовані; у категоріях є щонайменше по 1 тестовій публікації або коректно обробляється

порожній список; інтернет-з’єднання стабільне; кеш браузера очищено або виконано оновлення сторінки.

Критерій проходження тесту: після натискання на кожен блок категорії користувач має бути перенаправлений на відповідну сторінку/фільтр, де відображаються публікації цієї категорії (або коректне повідомлення про відсутність публікацій) без системних помилок.

Таблиця 3.3.

Тестування переходів за категоріями публікацій

№ з/п	Крок сценарію	Очікуваний результат	Отриманий результат	Відмітка (Так/Ні)
1	2	3	4	5
1	На сторінці «All Posts» натиснути блок «SOCIAL LIFE»	Відбувається перехід до сторінки/фільтра категорії SOCIAL LIFE	Перехід виконано	Так
2	Перевірити, що сторінка завантажилась без помилок	Відсутні 404/500, контент відображено	Помилок немає	Так
3	Повернутися назад до «All Posts»	Повернення виконується коректно	Повернення коректне	Так
4	Натиснути блок «POLITICS»	Перехід до сторінки/фільтра категорії POLITICS	Перехід виконано	Так
5	Повернутися назад до «All Posts»	Відображено сторінку з трьома блоками	Відображено коректно	Так
6	Натиснути блок «FOOD»	Перехід до сторінки/фільтра категорії FOOD	Перехід виконано	Так

Результати виконання розроблених тестових сценаріїв підтвердили коректну роботу основних функціональних елементів веб-сайту інтернет-магазину без виявлення критичних помилок або збоїв у процесі виконання користувацьких операцій. У ході тестування встановлено узгоджену взаємодію між клієнтською та серверною частинами системи, що забезпечує правильну

обробку запитів, стабільне завантаження даних із бази та коректне відображення інформації в користувацькому інтерфейсі.

Перевірка типових сценаріїв використання, зокрема реєстрації та авторизації користувачів, перегляду каталогу товарів, навігації між сторінками, взаємодії з публікаціями та відображення актуального асортименту на головній сторінці, показала відповідність реалізованого функціоналу проєктним вимогам. Динамічні елементи інтерфейсу, включно з інтерактивними товарними картками, переходами між розділами та асинхронними діями, функціонують стабільно та не призводять до порушення цілісності даних або некоректної поведінки системи.

Загальна оцінка результатів тестування свідчить про надійність та готовність розробленого веб-сайту до практичного використання в умовах реальної експлуатації. Отримані результати підтверджують доцільність обраних архітектурних і програмних рішень та створюють підґрунтя для подальшого розширення функціоналу інтернет-магазину, оптимізації окремих компонентів і адаптації системи до зростання навантаження та вимог користувачів.

ВИСНОВКИ

Дана кваліфікаційна робота присвячена розробці веб-сайту інтернет-магазину з повним циклом проєктування, програмної реалізації та перевірки коректності функціонування основних користувацьких і серверних компонентів. У межах роботи було послідовно виконано аналіз предметної області електронної комерції, обґрунтовано вибір технологічних засобів, спроєктовано архітектуру та базу даних системи, реалізовано клієнтську й серверну частини веб-застосунку, а також проведено тестування ключових сценаріїв його використання.

У першому розділі виконано ґрунтовний аналіз еволюції та сучасного стану інтернет-магазинів, що дозволило узагальнити основні етапи розвитку електронної комерції та визначити типовий набір функціональних модулів, притаманних сучасним торговельним веб-платформам. На основі аналізу існуючих рішень, зокрема веб-сайтів Amazon, eBay та Best Buy, було виявлено як сильні сторони таких систем, так і їх обмеження з погляду архітектури, зручності користування та масштабованості, що обґрунтувало доцільність розробки власного веб-сайту інтернет-магазину. Також здійснено порівняльний аналіз мов програмування та систем управління базами даних, за результатами якого обрано PHP як основу серверної частини та MySQL як СУБД, що забезпечують достатню продуктивність, гнучкість і поширеність у веб-проєктах подібного класу.

Другий розділ був присвячений проєктуванню веб-сайту, зокрема формалізації основних бізнес-процесів і побудові логічної структури системи. Було розроблено діаграми послідовності процесів реєстрації користувача, додавання товару до кошика та оформлення замовлення, що дозволило чітко визначити взаємодію між користувачем і компонентами системи. У межах проєктування бази даних виконано опис таблиць, їх атрибутів і зв’язків, а також побудовано фізичну модель бази даних, яка забезпечує цілісність, узгодженість і ефективний доступ до даних. Окрему увагу приділено проєктуванню

архітектури веб-сайту на основі шаблону MVC, що дозволило логічно розмежувати рівні подання, обробки запитів і доступу до даних та створити основу для подальшого розширення функціональних можливостей системи.

У третьому розділі реалізовано програмну частину веб-сайту та проведено тестування розробленого функціоналу. Було створено клієнтську частину веб-застосунку, яка включає сторінки реєстрації та авторизації користувачів, головну сторінку з переліком товарів, вітрину інтернет-магазину, каталог продукції та сторінку публікацій, що забезпечують зручну навігацію та наочне подання інформації. Серверна частина реалізована у вигляді класів для роботи з даними, які відповідають за створення облікових записів користувачів, керування ролями, формування сторінок, додавання вподобань і коментарів, а також обробку операцій оплати. Реалізовані функції забезпечують коректну обробку запитів, валідацію даних і взаємодію з базою даних відповідно до проєктних рішень. Для перевірки працездатності системи було розроблено та виконано тестові сценарії, що охоплюють реєстрацію користувачів, відображення товарів на головній сторінці та навігацію за категоріями публікацій, результати яких підтвердили стабільну та коректну роботу веб-сайту.

Проведена робота демонструє практичне застосування сучасних підходів до проєктування та розробки веб-застосунків у сфері електронної комерції та підтверджує можливість створення функціонального інтернет-магазину з чітко структурованою архітектурою, узгодженою базою даних і зручним користувацьким інтерфейсом. Отримані результати дозволяють використовувати розроблений веб-сайт як основу для подальшого розвитку, розширення функціональних можливостей, інтеграції платіжних сервісів і адаптації до реальних умов експлуатації, що свідчить про досягнення поставленої мети та виконання визначених у роботі завдань.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. 47 Ecommerce Statistics for Online Store Owners in 2025. URL: https://www.netguru.com/blog/ecommerce-statistics (дата звернення 02.01.2026).

2. У 2024 році українці витратили на онлайн-покупки 239 млрд грн. Це на 25% більше, ніж торік. URL: https://dev.ua/news/u-2024-mu-rotsi-ukraintsi-vytratyly-na-onlain-pokupky-na-25-bilshe-nizh-torik-1734683726 (дата звернення 02.01.2026).

3. How Many Ecommerce Sites Are There in the World? Statistics & Facts. URL: https://seo.ai/blog/how-many-ecommerce-sites-are-there-in-the-world (дата звернення 02.01.2026).

4. The History of eCommerce Timeline. URL: https://www.thefulfillmentlab.com/blog/history-of-ecommerce (дата звернення 02.01.2026).

5. Varian H. R. (2010). Computer mediated transactions. American Economic Review. 2010. Vol. 100, No 2, pp. 1-10.

6. Creating a Blue Ocean Strategy for an e-trading start-up. URL: https://krex.k-state.edu/server/api/core/bitstreams/da53056d-e60f-4ac7-a3c7-8b7e0ad6fe5a/content (дата звернення 02.01.2026).

7. Open Graph Markup -moduulin kehitys Magento-verkkokauppa-alustalla. URL: https://www.theseus.fi/bitstream/handle/10024/117276/Thesis%20Final.pdf?sequence=1 (дата звернення 02.01.2026).

8. Khando K., Islam M. S., Gao S. The emerging technologies of digital payments and associated challenges: a systematic literature review. Future Internet. 2022. Vol. 15, No 1, 21 p.

9. Chapman S. J. The Growth and Development of Online Commerce. CHOICE: Current Reviews for Academic Libraries. 2023. Vol. 60, No 10, pp. 963-971.

10. Barde K. Modular monoliths: Revolutionizing software architecture for efficient payment systems in fintech. International Journal of Computer Trends and Technology. 2023. Vol. 71, No 10, pp. 20-27.

11. Alberto Grande Sauci. Developing a complete e-commerce solution. URL: https://www.theseus.fi/bitstream/handle/10024/110509/Grande_Alberto.pdf (дата звернення 04.01.2026).

12. eCommerce Website Architecture. URL: https://webdesksolution.com/blog/ecommerce-website-architecture/ (дата звернення 04.01.2026).

13. Ecommerce Microservices and Headless: Why Distributed Systems Are the Present and Future of Online Retail. URL: https://www.bigcommerce.com/articles/ecommerce-website-development/microservices/ (дата звернення 04.01.2026).

14. Beyond Monoliths: Unveiling the Microservices Revolution. URL: https://www.expeed.com/beyond-monoliths-unveiling-the-microservices-revolution/ (дата звернення 04.01.2026).

15. 10 Basic Modules of Ecommerce Website: A Quick Guide. URL: https://easycommerce.dev/blog/basic-modules-of-ecommerce-website/ (дата звернення 04.01.2026).

16. Olumide M. L., Al Mamun Md Habib F. A., Halim G. A., Vijayan P., Rusly N. S., Fadzil L. M., Manickam S. Proposed Order Management System (OMS) for GAZI Communications. Journal of Computer Science & Computational Mathematics. 2023. Vol. 13, No 2, 157 p.

17. Amazon launches 'Themes,' a feature for filtering Customer Reviews by popular terms | TechCrunch. URL: https://techcrunch.com/2017/08/04/amazon-launches-themes-a-feature-for-filtering-customer-reviews-by-popular-terms/ (дата звернення 04.01.2026).

18. Amazon. Review summary. URL: https://www.trustpilot.com/review/www.amazon.com (дата звернення 04.01.2026).

19. How Amazon continues to improve the customer reviews experience with generative AI. URL: https://www.aboutamazon.com/news/amazon-ai/amazon-improves-customer-reviews-with-generative-ai (дата звернення 04.01.2026).

20. How to write, find, and edit Amazon reviews. URL: https://www.aboutamazon.co.uk/news/retail/how-to-write-find-and-edit-amazon-reviews (дата звернення 04.01.2026).

21. Amazon’s latest actions against fake review brokers: Amazon and BBB join forces again to combat fake reviews. URL: https://www.aboutamazon.com/news/policy-news-views/amazons-latest-actions-against-fake-review-brokers (дата звернення 04.01.2026).

22. Amazon.co.uk. Review summary. URL: https://uk.trustpilot.com/review/www.amazon.co.uk (дата звернення 02.01.2026).

23. eBay Architecture. URL: https://highscalability.com/ebay-architecture/ (дата звернення 04.01.2026).

24. 99+ eBay Feedback Examples for Buyers and Sellers. URL: https://www.3dsellers.com/blog/ebay-feedback-examples (дата звернення 04.01.2026).

25. About eBay. URL: https://www.consumeraffairs.com/online/ebay.htm (дата звернення 05.01.2026).

26. Ebay Reviews. URL: https://www.reviews.io/company-reviews/store/ebay (дата звернення 05.01.2026).

27. eBay.co.uk Reviews. URL: https://www.reviews.io/company-reviews/store/m.ebay.co.uk (дата звернення 020501.2026).

28. eBay Seller Reviews – Everything you need to know as a seller. URL: https://gominga.com/insights/ebay-seller-reviews/ (дата звернення 05.01.2026).

29. Best Buy. URL: https://www.comparably.com/companies/best-buy (дата звернення 05.01.2026).

30. Bates S., Reeve B., Trevena H. A narrative review of online food delivery in Australia: challenges and opportunities for public health nutrition policy. Public health nutrition. 2023. Vol. 26, No 1, pp. 262-272.

31. Best Buy. URL: https://www.yelp.com/brands/best-buy (дата звернення 05.01.2026).

32. About Best Buy. URL: https://www.consumeraffairs.com/retail/best_buy.htm (дата звернення 05.01.2026).

33. Best Buy Reviews. URL: https://www.reviews.io/company-reviews/store/best-buy (дата звернення 05.01.2026).

34. Best Buy. URL: https://www.trustpilot.com/review/stores.bestbuy.com (дата звернення 06.01.2026).

35. Балабанов О.І., Павленко А.П. PyCharm для розробників Python: Навчальний посібник. Львів: Видавництво Львівська політехніка, 2018. 300с.

36. Карапецький В. П. Побудова графічного контенту додатків з використанням JavaFX і Swing компонентів і даних, взятих із баз даних / В. П. Карапецький. – Науковий вісник НЛТУ. 2021. 418 с.

37. Зандстра Е. PHP об'єктно-орієнтоване програмування. Патерни і практика. Київ. Наш Формат. 2023. 456 с.

38. Петкович Д. Microsoft SQL Server 2019: Посібник для початківців / Д. Петкович. – Нью-Йорк: McGraw-Hill Education, 2020. 864 с.

39. Bell S. MySQL for Python: Interact with MySQL databases using Python. Birmingham: Packt Publishing, 2019. 440 с.

40. Петров І. Основи Firebird SQL. – Харків, видавництво “Фоліо” 2020. 350 с. Транслітерація: Ivan Petrov. Fundamentals of Firebird SQL. / Packt Publishing – Birmingham, 2018. 350 p.

41. Офіційна документація Laravel. URL: Laravel/PHP https://laravel.com/docs/12.x/installation (дата звернення 06.01.2025).

ДОДАТКИ

Додаток А

Скрипти створення бази даних

```
SET NAMES utf8mb4;
SET FOREIGN_KEY_CHECKS = 0;

DROP DATABASE IF EXISTS `laravel_store`;
CREATE DATABASE `laravel_store`
  CHARACTER SET utf8mb4
  COLLATE utf8mb4_unicode_ci;

USE `laravel_store`;

-- ------------------------------------------------------------
-- 1) Base table: users (referenced by author_id, user_id)
-- ------------------------------------------------------------
DROP TABLE IF EXISTS `users`;
CREATE TABLE `users` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NOT NULL,
  `email` VARCHAR(255) NOT NULL,
  `password` VARCHAR(255) NOT NULL,
  `created_at` DATETIME NULL,
  `updated_at` DATETIME NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `uq_users_email` (`email`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- ------------------------------------------------------------
-- 2) BLOG: categories, posts, tags, comments
--    From: blog_categories, blog_post_comments, blog_tags
-- ------------------------------------------------------------

DROP TABLE IF EXISTS `blog_categories`;
CREATE TABLE `blog_categories` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `parent_id` BIGINT UNSIGNED NULL,
  `name` VARCHAR(255) NOT NULL,
  `slug` VARCHAR(255) NOT NULL,
  `title` VARCHAR(255) NULL,
  `description` TEXT NULL,
  -- nested set columns for NodeTrait
  `_lft` INT NOT NULL DEFAULT 0,
  `_rgt` INT NOT NULL DEFAULT 0,

  PRIMARY KEY (`id`),
```

```
  UNIQUE KEY `uq_blog_categories_slug` (`slug`),
  KEY `ix_blog_categories_parent_id` (`parent_id`),
  CONSTRAINT `fk_blog_categories_parent`
    FOREIGN KEY (`parent_id`) REFERENCES `blog_categories` (`id`)
    ON DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- Missing in snippet but required for blog_post_comments.post_id and tags linking
DROP TABLE IF EXISTS `blog_posts`;
CREATE TABLE `blog_posts` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `category_id` BIGINT UNSIGNED NOT NULL,
  `author_id` BIGINT UNSIGNED NOT NULL,

  `title` VARCHAR(255) NOT NULL,
  `slug` VARCHAR(255) NOT NULL,
  `content` MEDIUMTEXT NULL,

  `created_at` DATETIME NULL,
  `updated_at` DATETIME NULL,
  `published_at` DATETIME NULL,

  PRIMARY KEY (`id`),
  UNIQUE KEY `uq_blog_posts_slug` (`slug`),
  KEY `ix_blog_posts_category_id` (`category_id`),
  KEY `ix_blog_posts_author_id` (`author_id`),
  CONSTRAINT `fk_blog_posts_category`
    FOREIGN KEY (`category_id`) REFERENCES `blog_categories` (`id`)
    ON DELETE RESTRICT ON UPDATE CASCADE,
  CONSTRAINT `fk_blog_posts_author`
    FOREIGN KEY (`author_id`) REFERENCES `users` (`id`)
    ON DELETE RESTRICT ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

DROP TABLE IF EXISTS `blog_tags`;
CREATE TABLE `blog_tags` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NOT NULL,
  `slug` VARCHAR(255) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `uq_blog_tags_slug` (`slug`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- Pivot to connect blog_tags with the rest of schema via blog_posts
DROP TABLE IF EXISTS `blog_post_tag`;
CREATE TABLE `blog_post_tag` (
  `post_id` BIGINT UNSIGNED NOT NULL,
  `tag_id`  BIGINT UNSIGNED NOT NULL,
  PRIMARY KEY (`post_id`, `tag_id`),
  KEY `ix_blog_post_tag_tag_id` (`tag_id`),
  CONSTRAINT `fk_blog_post_tag_post`
    FOREIGN KEY (`post_id`) REFERENCES `blog_posts` (`id`)
```

```
    ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_blog_post_tag_tag`
    FOREIGN KEY (`tag_id`) REFERENCES `blog_tags` (`id`)
    ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

DROP TABLE IF EXISTS `blog_post_comments`;
CREATE TABLE `blog_post_comments` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `author_id` BIGINT UNSIGNED NOT NULL,
  `post_id` BIGINT UNSIGNED NOT NULL,
  `parent_id` BIGINT UNSIGNED NULL,
  `text` TEXT NOT NULL,
  `active` TINYINT(1) NOT NULL DEFAULT 1,

  `created_at` DATETIME NULL,
  `updated_at` DATETIME NULL,

  PRIMARY KEY (`id`),
  KEY `ix_blog_post_comments_author_id` (`author_id`),
  KEY `ix_blog_post_comments_post_id` (`post_id`),
  KEY `ix_blog_post_comments_parent_id` (`parent_id`),

  CONSTRAINT `fk_blog_post_comments_author`
    FOREIGN KEY (`author_id`) REFERENCES `users` (`id`)
    ON DELETE RESTRICT ON UPDATE CASCADE,
  CONSTRAINT `fk_blog_post_comments_post`
    FOREIGN KEY (`post_id`) REFERENCES `blog_posts` (`id`)
    ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_blog_post_comments_parent`
    FOREIGN KEY (`parent_id`) REFERENCES `blog_post_comments` (`id`)
    ON DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- ------------------------------------------------------------
-- 3) SHOP: brands, categories, products, photos, reviews, comments
--    From: shop_brands, shop_categories, shop_products, shop_product_photos,
--          shop_product_reviews, shop_product_comments
-- ------------------------------------------------------------

DROP TABLE IF EXISTS `shop_brands`;
CREATE TABLE `shop_brands` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NOT NULL,
  `slug` VARCHAR(255) NOT NULL,
  `description` TEXT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `uq_shop_brands_slug` (`slug`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

DROP TABLE IF EXISTS `shop_categories`;
CREATE TABLE `shop_categories` (
```

```
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `parent_id` BIGINT UNSIGNED NULL,
  `name` VARCHAR(255) NOT NULL,
  `slug` VARCHAR(255) NOT NULL,
  `title` VARCHAR(255) NULL,
  `description` TEXT NULL,

  -- nested set columns for NodeTrait
  `_lft` INT NOT NULL DEFAULT 0,
  `_rgt` INT NOT NULL DEFAULT 0,

  PRIMARY KEY (`id`),
  UNIQUE KEY `uq_shop_categories_slug` (`slug`),
  KEY `ix_shop_categories_parent_id` (`parent_id`),
  CONSTRAINT `fk_shop_categories_parent`
    FOREIGN KEY (`parent_id`) REFERENCES `shop_categories` (`id`)
    ON DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

DROP TABLE IF EXISTS `shop_products`;
CREATE TABLE `shop_products` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `category_id` BIGINT UNSIGNED NOT NULL,
  `brand_id` BIGINT UNSIGNED NOT NULL,

  `availability` ENUM('in-stock','out-of-stock') NOT NULL DEFAULT 'out-of-stock',
  `title` VARCHAR(255) NOT NULL,
  `slug` VARCHAR(255) NOT NULL,
  `price` INT NOT NULL,

  `content` MEDIUMTEXT NULL,

  `status` ENUM('Active','Draft') NOT NULL DEFAULT 'Draft',
  `reviews` INT NOT NULL DEFAULT 0,
  `comments` INT NOT NULL DEFAULT 0,
  `rating` DECIMAL(3,2) NULL,

  `weight` INT NOT NULL DEFAULT 0,
  `quantity` INT NOT NULL DEFAULT 0,

  `created_at` DATETIME NULL,
  `updated_at` DATETIME NULL,

  PRIMARY KEY (`id`),
  UNIQUE KEY `uq_shop_products_slug` (`slug`),
  KEY `ix_shop_products_category_id` (`category_id`),
  KEY `ix_shop_products_brand_id` (`brand_id`),
  CONSTRAINT `fk_shop_products_category`
    FOREIGN KEY (`category_id`) REFERENCES `shop_categories` (`id`)
    ON DELETE RESTRICT ON UPDATE CASCADE,
  CONSTRAINT `fk_shop_products_brand`
    FOREIGN KEY (`brand_id`) REFERENCES `shop_brands` (`id`)
```

```
    ON DELETE RESTRICT ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

DROP TABLE IF EXISTS `shop_product_photos`;
CREATE TABLE `shop_product_photos` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `product_id` BIGINT UNSIGNED NOT NULL,
  `photo` VARCHAR(255) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `ix_shop_product_photos_product_id` (`product_id`),
  CONSTRAINT `fk_shop_product_photos_product`
    FOREIGN KEY (`product_id`) REFERENCES `shop_products` (`id`)
    ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

DROP TABLE IF EXISTS `shop_product_comments`;
CREATE TABLE `shop_product_comments` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `author_id` BIGINT UNSIGNED NOT NULL,
  `product_id` BIGINT UNSIGNED NOT NULL,
  `parent_id` BIGINT UNSIGNED NULL,
  `text` TEXT NOT NULL,
  `active` TINYINT(1) NOT NULL DEFAULT 1,

  `created_at` DATETIME NULL,
  `updated_at` DATETIME NULL,

  PRIMARY KEY (`id`),
  KEY `ix_shop_product_comments_author_id` (`author_id`),
  KEY `ix_shop_product_comments_product_id` (`product_id`),
  KEY `ix_shop_product_comments_parent_id` (`parent_id`),

  CONSTRAINT `fk_shop_product_comments_author`
    FOREIGN KEY (`author_id`) REFERENCES `users` (`id`)
    ON DELETE RESTRICT ON UPDATE CASCADE,
  CONSTRAINT `fk_shop_product_comments_product`
    FOREIGN KEY (`product_id`) REFERENCES `shop_products` (`id`)
    ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_shop_product_comments_parent`
    FOREIGN KEY (`parent_id`) REFERENCES `shop_product_comments` (`id`)
    ON DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

DROP TABLE IF EXISTS `shop_product_reviews`;
CREATE TABLE `shop_product_reviews` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `author_id` BIGINT UNSIGNED NOT NULL,
  `product_id` BIGINT UNSIGNED NOT NULL,
  `rating` TINYINT UNSIGNED NOT NULL,
  `text` TEXT NOT NULL,

  `created_at` DATETIME NULL,
```

```
  `updated_at` DATETIME NULL,

  PRIMARY KEY (`id`),
  KEY `ix_shop_product_reviews_author_id` (`author_id`),
  KEY `ix_shop_product_reviews_product_id` (`product_id`),

  CONSTRAINT `fk_shop_product_reviews_author`
    FOREIGN KEY (`author_id`) REFERENCES `users` (`id`)
    ON DELETE RESTRICT ON UPDATE CASCADE,
  CONSTRAINT `fk_shop_product_reviews_product`
    FOREIGN KEY (`product_id`) REFERENCES `shop_products` (`id`)
    ON DELETE CASCADE ON UPDATE CASCADE,

  CONSTRAINT `chk_shop_product_reviews_rating`
    CHECK (`rating` BETWEEN 1 AND 5)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- ------------------------------------------------------------
-- 4) SHOP: cart
--    From: shop_cart (user_id, product_id, ...)
-- ------------------------------------------------------------
DROP TABLE IF EXISTS `shop_cart`;
CREATE TABLE `shop_cart` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `user_id` BIGINT UNSIGNED NOT NULL,
  `product_id` BIGINT UNSIGNED NOT NULL,
  `quantity` INT UNSIGNED NOT NULL,
  `total_price` INT UNSIGNED NOT NULL,
  `total_weight` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `uq_shop_cart_user_product` (`user_id`, `product_id`),
  KEY `ix_shop_cart_product_id` (`product_id`),
  CONSTRAINT `fk_shop_cart_user`
    FOREIGN KEY (`user_id`) REFERENCES `users` (`id`)
    ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_shop_cart_product`
    FOREIGN KEY (`product_id`) REFERENCES `shop_products` (`id`)
    ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- ------------------------------------------------------------
-- 5) SHOP: characteristics
--    From: shop_characteristics (name,type,required,default,sort)
--    Added: pivot table to ensure FK connectivity with products
-- ------------------------------------------------------------

DROP TABLE IF EXISTS `shop_characteristics`;
CREATE TABLE `shop_characteristics` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NOT NULL,
  `type` ENUM('Integer','Float','String') NOT NULL,
  `required` TINYINT(1) NOT NULL DEFAULT 0,
```

```
  `default` VARCHAR(255) NULL,
  `sort` INT NOT NULL DEFAULT 0,
  PRIMARY KEY (`id`),
  UNIQUE KEY `uq_shop_characteristics_name` (`name`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- Pivot to connect characteristics with products (so characteristics are not isolated)
DROP TABLE IF EXISTS `shop_product_characteristics`;
CREATE TABLE `shop_product_characteristics` (
  `product_id` BIGINT UNSIGNED NOT NULL,
  `characteristic_id` BIGINT UNSIGNED NOT NULL,

  -- values (one of them should be used depending on characteristic.type)
  `value_int` BIGINT NULL,
  `value_float` DECIMAL(18,6) NULL,
  `value_string` VARCHAR(255) NULL,

  PRIMARY KEY (`product_id`, `characteristic_id`),
  KEY `ix_spc_characteristic_id` (`characteristic_id`),

  CONSTRAINT `fk_spc_product`
    FOREIGN KEY (`product_id`) REFERENCES `shop_products` (`id`)
    ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_spc_characteristic`
    FOREIGN KEY (`characteristic_id`) REFERENCES `shop_characteristics` (`id`)
    ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- ------------------------------------------------------------
-- 6) SHOP: delivery methods
--    From: shop_delivery_methods (name,cost,min_weight,max_weight,sort)
-- ------------------------------------------------------------
DROP TABLE IF EXISTS `shop_delivery_methods`;
CREATE TABLE `shop_delivery_methods` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NOT NULL,
  `cost` INT UNSIGNED NOT NULL DEFAULT 0,
  `min_weight` INT UNSIGNED NOT NULL DEFAULT 0,
  `max_weight` INT UNSIGNED NOT NULL DEFAULT 0,
  `sort` INT NOT NULL DEFAULT 0,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- ------------------------------------------------------------
-- 7) SHOP: orders, order items, order statuses
--    From: shop_orders, shop_order_items, shop_order_statuses
--    Added: customer & delivery data tables to satisfy FK for *_data_id
-- ------------------------------------------------------------

DROP TABLE IF EXISTS `shop_order_customers`;
CREATE TABLE `shop_order_customers` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
```

```
  `full_name` VARCHAR(255) NOT NULL,
  `phone` VARCHAR(50) NULL,
  `email` VARCHAR(255) NULL,
  `created_at` DATETIME NULL,
  `updated_at` DATETIME NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

DROP TABLE IF EXISTS `shop_order_deliveries`;
CREATE TABLE `shop_order_deliveries` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `city` VARCHAR(255) NULL,
  `address` VARCHAR(255) NULL,
  `postal_code` VARCHAR(50) NULL,
  `created_at` DATETIME NULL,
  `updated_at` DATETIME NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

DROP TABLE IF EXISTS `shop_orders`;
CREATE TABLE `shop_orders` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `user_id` BIGINT UNSIGNED NOT NULL,

  `customer_data_id` BIGINT UNSIGNED NULL,
  `delivery_data_id` BIGINT UNSIGNED NULL,

  `delivery_method_id` BIGINT UNSIGNED NULL,
  `delivery_method_name` VARCHAR(255) NULL,
  `delivery_method_cost` INT UNSIGNED NULL,

  `payment_method` VARCHAR(100) NULL,
  `cost` INT UNSIGNED NOT NULL DEFAULT 0,
  `note` TEXT NULL,

  `current_status` VARCHAR(100) NOT NULL,
  `cancel_reason` TEXT NULL,

  `created_at` DATETIME NULL,
  `updated_at` DATETIME NULL,
  PRIMARY KEY (`id`),
  KEY `ix_shop_orders_user_id` (`user_id`),
  KEY `ix_shop_orders_customer_data_id` (`customer_data_id`),
  KEY `ix_shop_orders_delivery_data_id` (`delivery_data_id`),
  KEY `ix_shop_orders_delivery_method_id` (`delivery_method_id`),

  CONSTRAINT `fk_shop_orders_user`
    FOREIGN KEY (`user_id`) REFERENCES `users` (`id`)
    ON DELETE RESTRICT ON UPDATE CASCADE,

  CONSTRAINT `fk_shop_orders_customer_data`
    FOREIGN KEY (`customer_data_id`) REFERENCES `shop_order_customers` (`id`)
```

```
    ON DELETE SET NULL ON UPDATE CASCADE,

  CONSTRAINT `fk_shop_orders_delivery_data`
    FOREIGN KEY (`delivery_data_id`) REFERENCES `shop_order_deliveries` (`id`)
    ON DELETE SET NULL ON UPDATE CASCADE,

  CONSTRAINT `fk_shop_orders_delivery_method`
    FOREIGN KEY (`delivery_method_id`) REFERENCES `shop_delivery_methods` (`id`)
    ON DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

DROP TABLE IF EXISTS `shop_order_items`;
CREATE TABLE `shop_order_items` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `order_id` BIGINT UNSIGNED NOT NULL,
  `product_id` BIGINT UNSIGNED NOT NULL,
  `product_name` VARCHAR(255) NOT NULL,
  `product_price` INT UNSIGNED NOT NULL,
  `product_quantity` INT UNSIGNED NOT NULL,
  `total_price` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`id`),
  KEY `ix_shop_order_items_order_id` (`order_id`),
  KEY `ix_shop_order_items_product_id` (`product_id`),

  CONSTRAINT `fk_shop_order_items_order`
    FOREIGN KEY (`order_id`) REFERENCES `shop_orders` (`id`)
    ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_shop_order_items_product`
    FOREIGN KEY (`product_id`) REFERENCES `shop_products` (`id`)
    ON DELETE RESTRICT ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

DROP TABLE IF EXISTS `shop_order_statuses`;
CREATE TABLE `shop_order_statuses` (
  `id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `order_id` BIGINT UNSIGNED NOT NULL,
  `value` VARCHAR(100) NOT NULL,
  `created_at` DATETIME NULL,
  PRIMARY KEY (`id`),
  KEY `ix_shop_order_statuses_order_id` (`order_id`),
  CONSTRAINT `fk_shop_order_statuses_order`
    FOREIGN KEY (`order_id`) REFERENCES `shop_orders` (`id`)
    ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
SET FOREIGN_KEY_CHECKS = 1;
```

Додаток Б

Програмний код веб сайту

```php
<?php

namespace App\Http\Controllers\Admin\Blog;

use App\Entity\Blog\Tag;
use App\Http\Controllers\Controller;
use App\Http\Requests\Admin\Blog\Tag\CreateRequest;
use App\Http\Requests\Admin\Blog\Tag\UpdateRequest;
use App\UseCases\Admin\Blog\TagManageService;
use Illuminate\Database\Eloquent\Builder;
use Illuminate\Http\Request;

class TagsController extends Controller
{
    /**
     * Business logic service for managing blog tags.
     *
     * @var TagManageService
     */
    private TagManageService $service;

    /**
     * Inject the tag management service.
     */
    public function __construct(TagManageService $service)
    {
        $this->service = $service;
    }

    /**
     * Display a paginated list of tags with optional filtering.
     */
    public function index(Request $request)
    {
        // Start with a base query provided by the service layer.
        $builder = $this->service->getTags();

        // Apply filters from request parameters (id/name/slug).
        $builder = $this->applyFilters($builder, $request);

        // Paginate results to keep the list manageable in the UI.
        $tags = $builder->paginate(20);

        return view('admin.blog.tags.index', ['tags' => $tags]);
    }
```

```
/**
 * Show the tag creation form.
 */
public function create()
{
    return view('admin.blog.tags.create');
}

/**
 * Persist a new tag and redirect back to the tags list.
 */
public function store(CreateRequest $request)
{
    $this->service->create($request);

    return redirect()->route('admin.blog.tags.index');
}

/**
 * Display a single tag.
 */
public function show(Tag $tag)
{
    return view('admin.blog.tags.show', ['tag' => $tag]);
}

/**
 * Show the tag edit form.
 */
public function edit(Tag $tag)
{
    return view('admin.blog.tags.edit', ['tag' => $tag]);
}

/**
 * Update an existing tag and redirect to its details page.
 */
public function update(UpdateRequest $request, Tag $tag)
{
    $this->service->update($request, $tag);

    return redirect()->route('admin.blog.tags.show', ['tag' => $tag]);
}

/**
 * Delete a tag and redirect back to the tags list.
 */
public function destroy(Tag $tag)
{
    $this->service->remove($tag);

    return redirect()->route('admin.blog.tags.index');
```

```
    }

    /**
     * Apply request-based filters to the query builder.
     *
     * Supported filters:
     * - id   : exact match
     * - name : partial match
     * - slug : partial match
     */
    private function applyFilters(Builder $query, Request $request): Builder
    {
        // Exact filter by ID (if provided).
        $query->when($request->filled('id'), function (Builder $q) use ($request) {
            $q->where('id', (int) $request->get('id'));
        });

        // Partial filter by name (if provided).
        $query->when($request->filled('name'), function (Builder $q) use ($request) {
            $name = trim((string) $request->get('name'));
            $q->where('name', 'like', '%' . $name . '%');
        });

        // Partial filter by slug (if provided).
        $query->when($request->filled('slug'), function (Builder $q) use ($request) {
            $slug = trim((string) $request->get('slug'));
            $q->where('slug', 'like', '%' . $slug . '%');
        });

        return $query;
    }
}
```