МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ

ННІ/факультет	Інформаційних технологій
Кафедра	Інформатики і прикладного програмного забезпечення
Спеціальність	Інженерія програмного забезпечення
Форма навчання	Денна

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

Косяк Володимир Олександрович
(прізвище, ім'я, по батькові здобувача)

на тему **Розробка програмного забезпечення продажу книг**
(повна назва теми)

за матеріалами праць провідних спеціалістів з розробки ПЗ та проектування БД
(повна назва бази дослідження)

науковий керівник **к.е.н., доцент** _____ **Баран С.В**
(наук. ступінь, вчене звання) *(підпис)* *(прізвище, ініціали)*

Робота допущена до захисту в ЕК
Протокол засідання кафедри
від 11.06.2025 № 12
Завідувач кафедри _____
(підпис)
д.т.н., професор Зеленський О.С.
Наук. ступень, вчене звання *Ініціали, прізвище*

Кривий Ріг – 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ

ННІ/факультет	Інформаційних технологій
Кафедра	Інформатики і прикладного програмного забезпечення
Спеціальність	Інженерія програмного забезпечення
Форма навчання	Денна

«ЗАТВЕРДЖУЮ»
Завідувач кафедри _____ Зеленський О.С.
(підпис) (Прізвище, ініціали)
«11» червня 2025 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ БАКАЛАВРСЬКУ РОБОТУ

1. Тема роботи Розробка програмного забезпечення продажу книг

Керівник роботи _к.е.н_Баран С.В_____
затверджені наказом закладу вищої освіти від «04» квітня 2025 р. № 222-ст

2. Строк подання здобувачем роботи до «09» червня 2025 р.

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

Розділ 1. Постановка задачі

Розділ 2. Проєктування web-сайту

Розділ 3. Проєктування бази даних web-сайту

Розділ 4. Розробка web-сайту та додатку

Об’єкт дослідження: програмне забезпечення, орієнтоване на купівлю та ознойомлення з книгами у двох форматах – паперовому й електронному

Предмет дослідження: структура, логіка та інтерфейс веб-застосунку та десктопного *3D-модулю, а також способи подання інформації, що дозволяють полегшити пошук, вибір і читання книг для кінцевого користувача*

Мета кваліфікаційної роботи: створити інтегроване рішення, яке допомагає користувачам швидко знаходити та замовляти книги, отримувати миттєвий доступ до електронної версії під час очікування доставки паперового примірника, а також забезпечує наочний перегляд обкладинок через 3D-модель

5. Дата видачі завдання «04» квітня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів МДР	Строк виконання етапів роботи	Відмітка керівника про виконання етапів (дата, підпис)
1	Підготовка розділу 1	04.04.2025-13.04.2025	
2	Підготовка розділу 2	14.04.2025-26.04.2025	
3.	Підготовка розділу 3	27.04.2025-15.05.2025	
4	Підготовка розділу 4	16.05.2025-08.06.2025	
5	Реєстрація завершеної кваліфікаційної роботи	09.06.2025	Реєстраційний №___ «09»червня 2025 р.
6	Отримання відгуку від наукового керівника	10.06.2025	
7	Подання кваліфікаційної роботи на перегляд завідувачу кафедри	11.06.2025	
8	Отримання зовнішньої рецензії	12.06.2025	
9	Попередній захист кваліфікаційної роботи на кафедрі	13.06.2025	
10	Підготовка до захисту в ЕК	16.06.2025-21.06.2025	

Завдання підготував науковий керівник _____ ___Баран С.В____

(підпис) (прізвище та ініціали)

Завдання одержав _____ __Косяк В.О._____

(підпис) (прізвище та ініціали)

ЗГОДА здобувача вищої освіти

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

Я, _____Косяк Володимир Олександрович_____, підтримую політику Державного університету економіки і технологій з академічної доброчесності і відкритого доступу.

Засвідчую, що кваліфікаційна бакалаврська робота Розробка програмного забезпечення продажу книг виконана самостійно та не містить академічного плагіату. Я не надавав і не одержував недозволену допомогу під час підготовки цієї роботи. Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

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

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

Дата підпис ініціали, прізвище (власноруч)

АНОТАЦІЯ

на бакалаврську кваліфікаційну роботу

«Розробка програмного забезпечення продажу книг»

Косяк Володимир Олександрович

Кваліфікаційна бакалаврська робота на здобуття освітньо-кваліфікаційного рівня бакалавра зі спеціальності 121 «Інженерія програмного забезпечення» – Державний університет економіки і технологій – Кривий Ріг, 2025.

У дипломній роботі створено два програмні продукти: веб-сайт для продажу книг та настільний 3D-переглядач обкладинок. Веб-сайт реалізовано з використанням PHP і MySQL - він дозволяє переглядати каталог книг, фільтрувати за категоріями та ціною, оформлювати замовлення й читати придбані електронні копії онлайн. Інтерфейс адаптивний, побудований за принципом багатосторінкового додатку з AJAX-елементам.

Десктопний переглядач розроблено на C# (Windows Forms) із використанням OpenGL через OpenTK. Програма завантажує список книг безпосередньо з тієї самої бази даних сайту та відтворює їх обкладинки і корінці у форматі 3D. Користувач може обертати модель мишею і переглядати дизайн книги з різних боків. Алгоритм автоматично підвантажує текстури обкладинок та їхньої корінної частини й відображає їх у вікні рендерингу.

Ключові слова: ВЕБ-САЙТ, PHP, MySQL, AJAX, WINDOWS FORMS, OPENGL, 3D-ПЕРЕГЛЯДАЧ.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД(база даних)	Впорядкований набір логічно взаємопов’язаних даних, що використовується спільно і призначений для задоволення інформаційних потреб веб-застосунку та десктопу.
СУБД	Система управління базами даних.
ПЗ	Програмне забезпечення.
AJAX	Технологія асинхронного обміну даними між браузером і сервером.
MySQL	Реляційна система управління базами даних, у якій зберігаються всі сутності.
WinForms	Фреймворк від Microsoft для створення десктопних застосунків із графічним інтерфейсом (Windows Forms).

ЗМІСТ

ВСТУП

У сучасному світі все більше людей віддає перевагу поєднанню традиційного і цифрового форматів читання: з одного боку, хочеться отримати друковану книгу, яка зберігає тактильний досвід справжнього читання, а з іншого - мати змогу негайно ознайомитися з електронним варіантом, не чекаючи поки видання надійде поштою. Саме така гнучкість у виборі формату робить процес купівлі книг більш привабливим та зручним.

У багатьох виникає потреба знайти єдиний ресурс, де можна не лише придбати паперове видання, а й миттєво отримати доступ до електронної версії для читання «тут і зараз». Це особливо актуально у разі, коли доставка займає кілька днів або навіть тижнів. Можливість почати читати базові глави онлайн допомагає заощадити час і одразу зануритися в текст, не відкладаючи дозвілля.

Ідея створити інтуїтивний веб-магазин із функцією онлайн-читалки стала основою цієї роботи. Веб-сайт повинен забезпечити користувачу швидкий та зручний шлях від пошуку книги до оформлення замовлення: можливість фільтрації за жанрами і ціною, детальний опис видань, простий процес реєстрації й оплати, а також доступ до електронної версії ще під час очікування доставки паперового примірника. Завдання адміністратора - оперативно обробляти замовлення, підтримувати актуальність асортименту і оперативно реагувати на запити покупців.

Додатково розроблено десктопний 3D-додаток, який у візуальному форматі демонструє обкладинки й корінці книг. Це дозволяє покупцям отримати уявлення про реальний вигляд видання до його отримання.

Завдання, що ставилися під час реалізації:

1. Забезпечити швидкий і зручний пошук видань за назвою, жанром і ціною;
2. Реалізувати гнучкий механізм фільтрації та сортування книг у каталозі;

3. Надати користувачам можливість отримувати електронний доступ до книги одразу після оформлення замовлення;
4. Створити адміністративну панель для керування асортиментом, замовленнями та користувачами;
5. Розробити десктопний модуль із 3D-візуалізацією обкладинок і корінців для наочного перегляду асортименту.

Актуальність проєкту обумовлена зростаючою популярністю онлайн-покупок і потребою у миттєвому доступі до улюблених видань. Багато читачів не готові чекати на паперову книгу і шукають можливість почати читати одразу, незалежно від локації чи формату. Комплексне рішення, яке поєднує веб-магазин з онлайн-читалкою та 3D-переглядом обкладинок, задовольняє ці очікування й робить процес купівлі й читання більш комфортним та ефективним.

РОЗДІЛ І
ПОСТАНОВКА ЗАДАЧІ

1.1. Характеристика задачі

У дипломній роботі розглядається завдання створення програмного комплексу, що спрощує пошук, купівлю й читання художніх та навчальних книг. Ідея полягає у поєднанні двох платформ: сучасного веб-магазину й десктопного 3D-переглядача. Обидві частини працюють із єдиною базою даних, орієнтовані на одну цільову аудиторію, але розв’язують різні задачі.

Веб-сайт - це зручне торговельне середовище, де відвідувач може швидко знайти потрібну книгу, ознайомитися з її описом і негайно розпочати читання електронної версії після покупки. Завдяки фільтрам за жанром і ціною користувач отримує можливість за лічені секунди звузити асортимент до цікавих йому видань. На детальній сторінці книги представлено повний опис, що дає змогу прийняти обгрунтоване рішення. Після підтвердження оплати електронної копії - покупець одразу отримує доступ до рідера без необхідності чекати доставки паперового примірника.

Десктопний додаток перетворює книгу на інтерактивний 3D-об’єкт: користувач бачить передню й задню обкладинки, корінець, може вільно обертати модель, ніби тримає книгу в руках. Це особливо корисно тим, хто цінує візуальне сприйняття реального видання перед покупкою: можна роздивитися фактуру обкладинки, шрифт заголовка, стиль оформлення корінця. Такий підхід підсилює емоційну складову - адже для багатьох читачів важливо «відчути» книгу ще до фактичного отримання паперового примірника.

Потреби та проблеми, що вирішуються:

1. Швидкий вибір та порівняння. Користувач часто вагається між кількома книгами, особливо якщо жанри та оформлення схожі. Фільтри за жанром, ціною, видавництвом і автором дають змогу миттєво знайти

потрібні варіанти. Крім того, 3D-відображення дає більше візуальної й фактичної інформації про вигляд обкладинки та корінця ще до покупки.

2. Миттєвий доступ до вмісту. Після підтвердження замовлення електронні книги стають доступними для читання прямо на сайті. Це особливо важливо, коли фізичне видання потребує кількох днів на доставку. Зі столиці чи навіть під час відрядження користувач може почати знайомство з текстом, а потім завершити читання в зручному для себе темпі.
3. Простота адміністрування. Адміністратор має власну панель керування, у якій може додавати нові книги та категорії, контролювати залишки на складі та обробляти замовлення без залучення технічного фахівця. Це знижує операційні витрати й допомагає оперативно оновлювати асортимент.
4. Візуальне залучення. 3D-переглядач підвищує зацікавленість: крутити «живу» книгу приємніше, ніж просто дивитися статичне зображення, яке є на сайті.
5. Універсальність формату. Поєднання «паперової» та «цифрової» версій задовольняє різні потреби: у когось є буденна звичка читати на планшеті чи смартфоні, а іншим комфортніше тримати в руках паперовий примірник. Наявність обох варіантів одночасно робить послугу більш привабливою - поки книга знаходиться в дорозі до покупця, електронний формат вже готовий до перегляду.

Ключові функції веб-частини

- Каталог книг із фільтрами за жанром і ціною.
- Легка інтуїтивність для звичайного користувача.
- Розширений пошук та сторінка детального опису книги.
- Кошик і оформлення замовлення з відстеженням статусу.
- Можливість читати книги після замовлення.
- Адміністративний модуль дозволяючий легко і швидко створювати нові категорії, книги, а також перевіряти замовлення.

Функції десктопного додатку

- Отримання даних з веб-сайту: програма завантажує список доступних книг та шляхи до зображень обкладинок із віддаленої бази, синхронізуючись зі змінами в магазині.
- 3D-візуалізація видання: передня й задня обкладинки, корінець і блок сторінок відтворюються засобами OpenGL.
- Інтерактивність: користувач може обертати модель мишею й швидко перемикатися між книгами списком.

Цільова аудиторія

- Читачі, що надають перевагу електронним або друкованим виданням, але хочуть переглянути книгу «вживу» перед покупкою;
- Користувачі з базовими цифровими навичками, що цінують простий інтерфейс і швидкий доступ до контенту.
- Користувачі, які потребують миттєвого доступу до змісту книги під час очікування доставки паперового примірника.

Таким чином, проєкт поєднує класичний e-commerce із елементами 3D-візуалізації, розширюючи досвід користувача та спрощуючи шлях від пошуку книги до її читання. Окрім того, він задовольняє актуальну потребу поєднати «паперовий» і «цифровий» формати: покупець отримує друковане видання, яке приємно тримати в руках, і водночас миттєвий доступ до електронної копії. Це дає можливість не носити книгу всюди з собою, а читати її на смартфоні чи планшеті в дорозі, а також почати ознайомлення з першими розділами ще під час доставки фізичного примірника. Таким чином, рішення поєднує зручність онлайн-покупок, наочність 3D-моделі та гнучкість мультиформатного читання, що робить його особливо привабливим для сучасної аудиторії.

1.2. Огляд існуючих web-сайтів

На старті проєкту було вивчено досвід двох українських книжкових майданчиків. Аналіз дозволив зрозуміти, як організовано подачу контенту, які

інтерфейсні рішення підтримують (або, навпаки, ускладнюють) шлях покупця та що саме варто удосконалити у власному рішенні.

Одним із найпопулярніших сайтів із книгами є «Книгарня-Є» (Рис. 1.1.). Перевага цього магазину - великий каталог і детально розписані бонусні програми, знижки та умови доставки. Проте інтерфейс важко назвати дружнім: у десктопній версії ліворуч постійно висить громіздке меню з усіма категоріями й акціями. Надлишок елементів відволікає від головної дії - вибору книги - і створює відчуття хаосу на екрані.

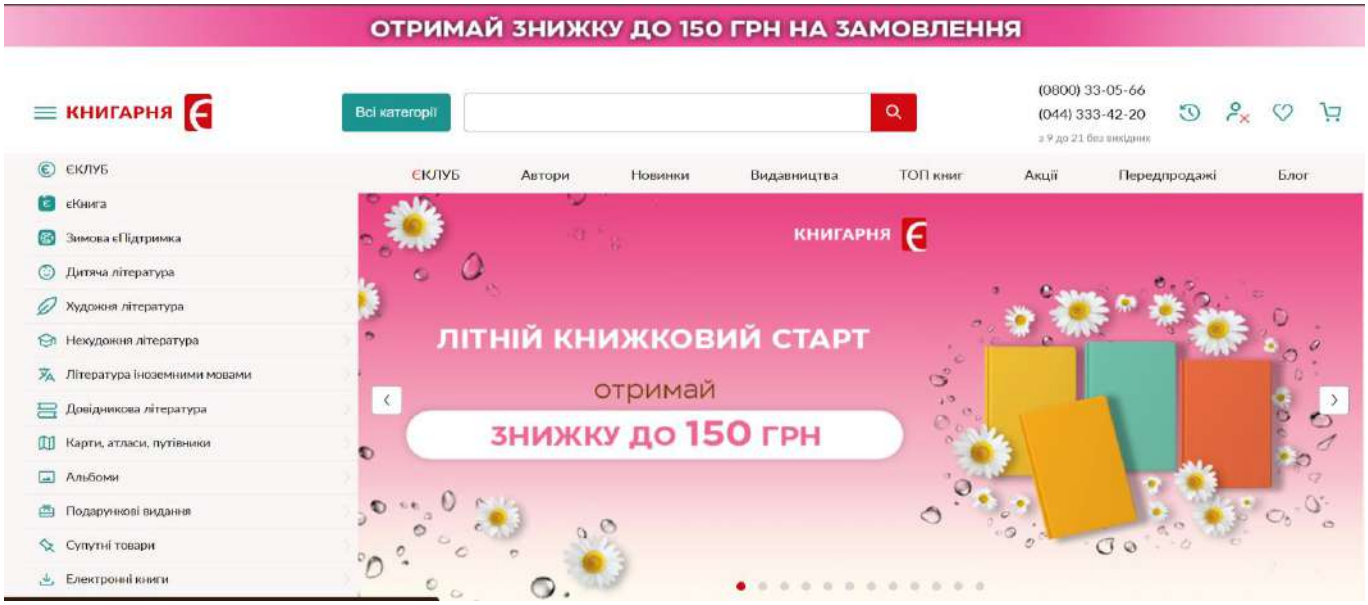

Рис. 1.1. Головна сторінка сайту «Книгарня-Є»

Переваги «Є-книгарні»

- Широкий асортимент - великий вибір друкованих та електронних видань різних жанрів.
- Розгалужені бонусні програми - детальна система знижок, промокодів і кешбеку.
- Багато фільтрів - за жанром, ціновим діапазоном, мовою.

Malopus обирає протилежну стратегію - мінімалізм. Навігація інтуїтивна: користувач одразу розуміє, куди натискатии, щоби переглянути товар чи оформити замовлення. У картці товару подано ключові дані, а для позицій у передзамовленні - очікувану дату виходу. Додатково виводяться

пов’язані новини й бонуси, що стимулює інтерес і полегшує ухвалення рішення (Рис. 1.2.).

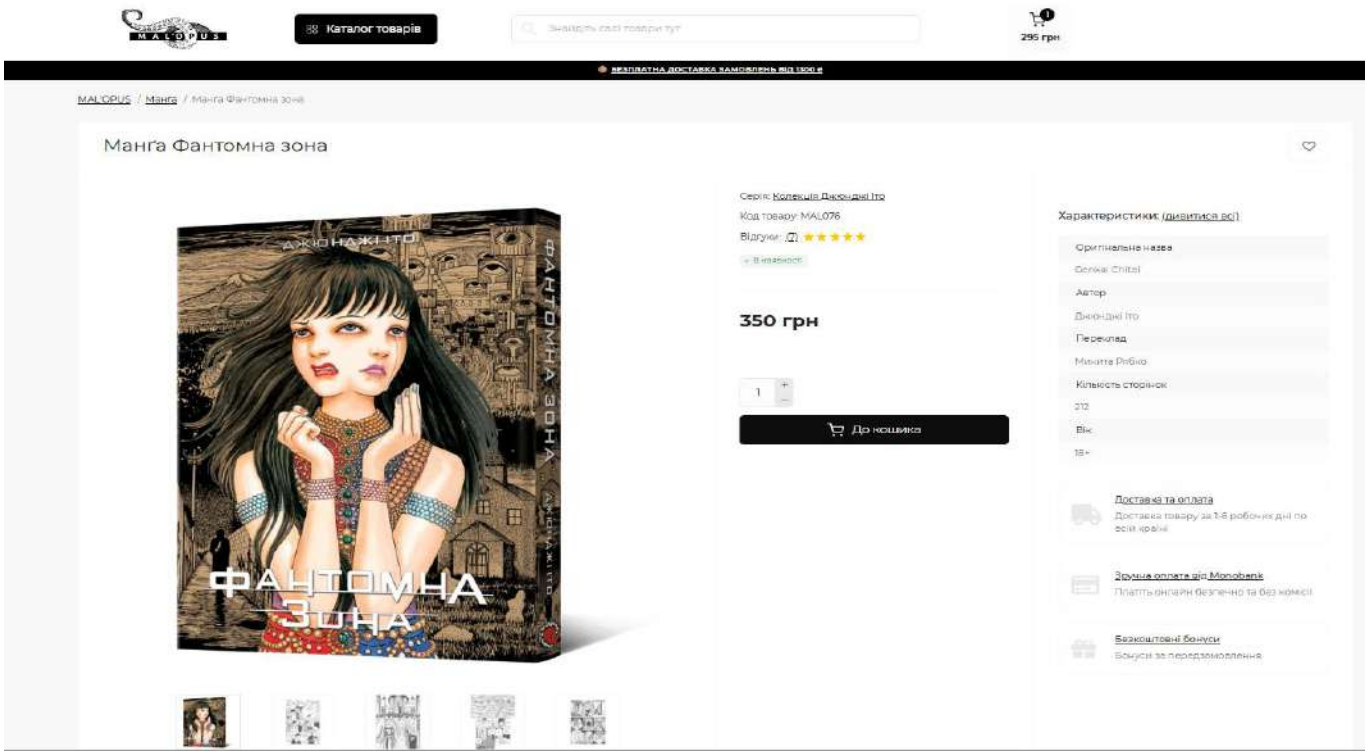

Рис. 1.2. Сторінка картки товара на сайті «Malopus»

Одним із вагомих позитивних аспектів цього ресурсу є зрозуміла та продумана фільтрація товарів. Користувач спочатку вибирає головну категорію (наприклад, комікси), після чого потрапляє на сторінку, де вже представлений список підкатегорій, які визначаються окремими тайтлами. Завдяки цьому покупець має змогу побачити повний перелік коміксів без змішування їх з іншими жанрами. На цій же сторінці можна застосувати додаткові фільтри за назвою тайтлу чи ціновим діапазоном (Рис. 1.3.).

Проте варто зазначити, що цей підхід має певні недоліки в довгостроковій перспективі. Якщо кількість товарних позицій на сайті суттєво зросте, наприклад, до кількох сотень чи навіть тисяч, то користувач може зіткнутися з труднощами через перевантаження переліку тайтлів. За такого сценарію потрібне буде створення додаткових піджанрів або внутрішніх класифікацій, що може ускладнити структуру сайту та погіршити швидкість пошуку. У власному проєкті планується уникнути таких ускладнень і зберегти

більш загальну категоризацію з простими, але ефективними фільтрами, що дозволять користувачеві швидко орієнтуватися навіть при великому обсязі асортименту. Таким чином, ми бачимо, що підхід, який використовує сайт Malopus, хоч і є надзвичайно зручним для поточного масштабу асортименту, має свої обмеження при масштабуванні.

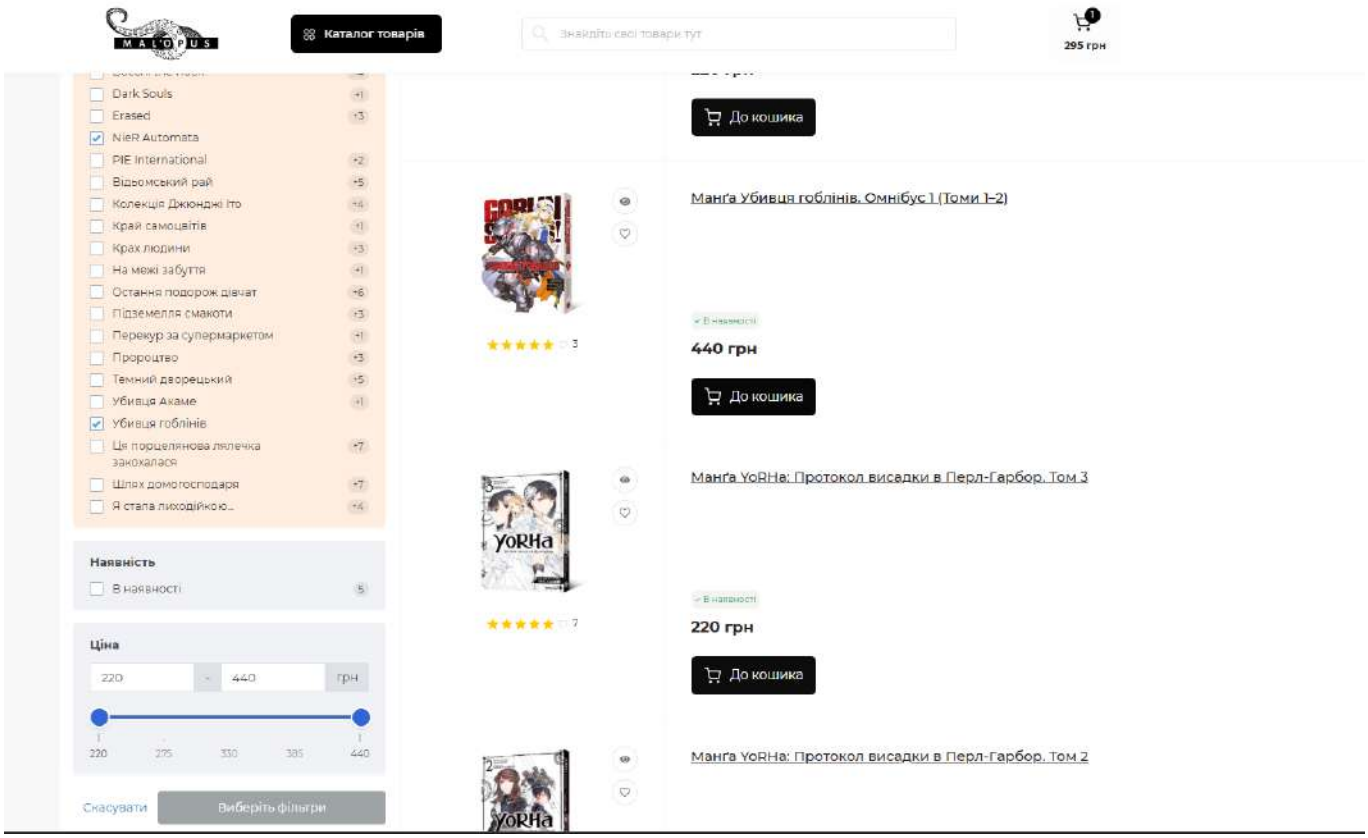

Рис. 1.3. Сторінка фільтрації товарів на сайті «Malopus»

Переваги «Malopus»

- Мінімалістичний дизайн - інтерфейс без зайвих елементів, швидко зрозуміло, куди натискати.
- Інформативна картка товару - короткий опис, статус наявності, дата релізу для передзамовлень.
- Новинні блоки поруч із товаром - пов’язані статті та оголошення про бонуси за передзамовлення.
- Чітка структура каталогу - зрозумілі категорії й навігація без «схованих» підменю.

- Швидкість завантаження сторінок - легка верстка й оптимізовані зображення забезпечують комфортний перегляд.

Після огляду стало зрозуміло, що більшість книжкових майданчиків стикаються з труднощами пошуку балансу між функціональністю й зручністю. Є-книгарня демонструє багатий сервіс, але страждає від перевантаженого інтерфейсу, тоді як Malopus виграє ергономікою, та може поступатися гнучкістю акційних механізмів. У власному проєкті планується поєднати сильні сторони обох підходів: забезпечити повний набір торгових можливостей і водночас залишити інтерфейс мінімалістичним та інтуїтивно зрозумілим.

1.3. Аналіз вимог до веб-сайту та додатку

Після визначення мети проєкту й аналізу існуючих книжкових онлайн-майданчиків було сформовано комплексний перелік вимог до майбутнього програмного рішення, що складається з двох компонентів: веб-магазину книг та інтерактивного десктопного 3D-переглядача.

Головна задача - створити сервіс, який дозволяє користувачеві легко знаходити книги, оформлювати замовлення та миттєво отримувати доступ до електронних копій, поки друкований примірник знаходиться на етапі доставки.

Користувач має отримати максимально комфортний досвід, тому особлива увага приділяється продуманості навігації та мінімалізму інтерфейсу, де немає зайвих елементів чи складних кроків, які тільки заважають користувачеві. З цією метою були сформовані такі основні вимоги до функціоналу веб-частини.

Вимоги до веб-сайту:

- Каталог із ефективними фільтрами: користувач може швидко знайти книгу за жанром і ціновим діапазоном, що дозволяє суттєво скоротити час пошуку серед великої кількості пропозицій.

- Сторінка книги: обкладинка, коротка анотація, ціна, кнопки «Додати в кошик» і «Читати онлайн» (для придбаних електронних копій).
- Кошик та оформлення замовлення: користувач може легко керувати обраними книгами, змінюючи кількість товарів, обираючи спосіб доставки й оплати, а також переглядаючи підсумкову інформацію про замовлення.
- Онлайн-рідер для читання придбаних книг без завантажень.
- Особистий кабінет зі списком замовлень, повторним завантаженням електронних файлів.
- Адмін-панель для створення, редагування та видалення книг, жанрів і замовлень.
- Зручний інтерфейс для користувачів.
- можливість масштабування проекту, збільшуючи функціонал і дані для бази даних.

Такий набір вимог закладає основу для ресурсу, що поєднує візуальний мінімалізм із найнеобхіднішими можливостями. Головний акцент - забезпечити користувачеві простий шлях: обрати книгу, швидко оплатити та одразу після підтвердження отримати електронну версію для читання. Щоб цей шлях був справді безшовним, інтерфейс будується за принципом «нічого зайвого»: зручні фільтри, зрозуміла картка товару, кошик у два кліки й рідер, який відкриває придбане видання. У результаті користувач витрачає мінімум часу на навігацію сайтом і максимум - на читання обраної книги.

Окрім зручності використання веб-сайту, важливим аспектом стала можливість надати користувачеві наочне уявлення про вигляд книги в реальному житті. Для цього розроблено додатковий компонент - десктопний додаток для перегляду книг у форматі інтерактивної 3D-моделі. Відповідно, було сформульовано такі основні вимоги

Вимоги до десктопного додатку:

- Синхронізація: отримання актуального списку книг і шляхів до обкладинок безпосередньо з бази.

- 3D-візуалізація: кожна книга представлена у вигляді інтерактивної тривимірної моделі, яка включає передню й задню обкладинки, а також корінець, що забезпечує повне уявлення про дизайн видання.
- Інтерактивність та простота використання: користувач може обертати модель книги за допомогою миші, переглядати її з різних ракурсів та швидко перемикатися між книгами у списку за допомогою зручних елементів керування.

Додатковою перевагою такого десктопного рішення є посилення емоційної складової вибору книг. Користувач має можливість переглянути видання ще до його фізичного отримання, що може стати вирішальним чинником для ухвалення рішення про покупку.

Об'єднуючи описані вимоги до веб-сайту та десктопної програми, отримуємо цілісний програмний комплекс, що максимально відповідає очікуванням сучасних користувачів. Такий підхід дає змогу створити процес придбання книги швидким, ефективним і привабливим. Завдяки інтегрованому рішенню користувачі не тільки швидко знаходять і купують книги, а й отримують можливість візуально ознайомитися з ними за допомогою інтерактивної тривимірної моделі. Це поєднання електронної й паперової версії, простоти інтерфейсу й інтерактивної візуалізації робить проєкт конкурентним на сучасному книжковому ринку, забезпечуючи максимальну зручність і задоволення для користувачів.

Висновки до розділу 1

У першому розділі було сформульовано загальну постановку завдання зі створення програмного комплексу для продажу та перегляду книг, що складається з веб-магазину й десктопного 3D-переглядача. Проведено аналіз двох існуючих платформ («Є-книгарня» та Malopus»), який дав змогу визначити їхні сильні та слабкі сторони: «Є-книгарня» вирізняється широким функціоналом, але страждає від перевантаженого інтерфейсу, а Malopus

демонструє зручний мінімалістичний дизайн із повною інформацією в картках товарів.

На основі цієї оцінки та загальних потреб користувачів сформовано перелік ключових вимог до веб-сайту: від каталогу з фільтрами за жанром і ціною до вбудованого онлайн-рідера й адміністративної панелі для керування асортиментом і замовленнями.

Для десктопного 3D-переглядача окреслено необхідність синхронізації з базою даних сайту та серверним сховищем зображень, рендерингу обкладинок і корінця книги засобами OpenGL

Таким чином, перший розділ обґрунтував потребу в комплексному рішенні, що поєднує переваги обох платформ: багатофункціональність веб-магазину та наочність 3D-візуалізації. Сформульовані вимоги стануть підґрунтям для подальшого проєктування бази даних, архітектури та безпосередньої розробки обох компонентів системи.

РОЗДІЛ ІІ
ПРОЄКТУВАННЯ WEB-САЙТУ

2.1. Розробка алгоритму

У межах розробки веб-сайту для продажу книг було визначено набір основних сценаріїв взаємодії користувача та адміністратора. Кожний сценарій реалізовано окремим PHP-скриптом із чіткою послідовністю кроків, що забезпечують коректність роботи та зручність використання. Нижче наведено загальні тези для кожного ключового алгоритму.

1. Ініціалізація та загальна логіка сторінок

- На початку кожного запиту підключається файл config.php, який встановлює з’єднання з базою даних та запускає сесію.
- Далі підключається header.php (для звичайного користувача) або admin_header.php (для адміністратора), що відповідають за виведення меню й перевірку прав (у випадку адміна).
- За потреби скрипт аналізує GET і POST параметри, формуж SQL-запити та оброблює результати.

2. Перегляд каталогу та фільтрація

- Зчитуються опціональні параметри фільтрації: вибрана категорія (category), ціновий діапазон (price_min, price_max), порядок сортування (sort), номер сторінки пагінації (page).
- На основі цих параметрів динамічно формується умова (WHERE) у SQL-запиті до таблиці books.
- Виконується підрахунок загальної кількості записів для побудови пагінації.
- Потім відбувається вибірка частини результатів для поточної сторінки (з урахуванням LIMIT і OFFSET).

- Сформовані результати передаються у шаблон, де генеруються картки книг із базовою інформацією (зображення, назва, ціна) та посиланням на детальну сторінку.

3. Пошук по каталогу

- Користувач вводить запит у поле пошуку, що надсилається методом POST до search_page.php.
- Формується SQL-запит із використанням оператора LIKE для пошуку за назвою або описом книги.
- Отримані результати виводяться як картки, аналогічно до каталогу, за відсутності збігів показується повідомлення про відсутність результатів.

4. Перегляд детальної сторінки книги

- При переході на product.php?id={ID} відбувається валідація параметра id (перевірка наявності і коректності).
- Виконується один JOIN-запит до таблиць books і categories для отримання повних даних про книгу (назва, опис, жанр, ціна, статус наявності).
- Якщо книга знайдена, система відображає детальну інформацію та кнопки для подальших дій: «Додати в кошик» або «Читати онлайн».

5. Додавання товару до кошика

- При натисканні кнопки «Додати в кошик» формується POST-запит на shop.php.
- Система перевіряє авторизацію користувача; якщо не залогінений - виконує редирект на сторінку входу.
- Далі скрипт перевіряє, чи товар із вказаним id існує у таблиці books.
- Якщо все коректно, ідентифікатор товару та кількість додаються у таблицю cart у базі даних. У разі повторного додавання збільшується лічильник кількості.

6. Перегляд вмісту кошика

- Відкриваючи cart.php, система зчитує товари з таблиці cart у базі даних для поточного користувача.

- Якщо кошик порожній, виводиться повідомлення про відсутність товарів.
- Якщо є товари, для кожного з них виводяться назва, ціна, обкладинка, кількість, сума, опис, а також використовуються елементи керування (зміна кількості, видалення).
- Загальна сума доданих книг в кошик обчислюється автоматично.
- Кнопка «Оформити замовлення» веде на сторінку checkout.php.

7. Оформлення замовлення

- У checkout.php спочатку перевіряється, чи користувач авторизований. Якщо ні - відбувається редирект на login.php.
- При GET-запиті виводиться форма з полями: ім’я, телефон, місто, відділення, адреса доставки, email, спосіб оплати.
- При POST-запиті скрипт проводить валідацію введених даних.
- Якщо перевірка успішна, створюється запис у таблиці orders, після чого для кожної книги з кошика створюється запис у таблиці purchases.
- Після цього кошик очищується, і користувач бачить повідомлення про успішне оформлення замовлення.
- У разі помилки виводиться повідомлення про помилку.

8. Перегляд власних замовлень

- Відвідавши orders.php, користувач бачить список своїх замовлень.
- Для кожного замовлення відображаються основні поля: id замовлення, загальна сума (total_price), статус оплати (payment_status), дата створення (placed_on), а також список книг (total_books).

9. Онлайн-рідер придбаних книг

- У read_book.php?id={ID} перевіряється, чи користувач авторизований.
- Далі перевіряється, чи існує файл книги з таким ID.
- Якщо файл знайдено, користувач перенаправляється на цей файл для перегляду (відкривається у браузері).

10. Адміністративні сценарії

- Кожен скрипт адміністрування (наприклад, admin_products.php, admin_categories.php, admin_orders.php, admin_users.php, admin_contacts.php) починається з перевірки авторизації адміністратора - перевіряється наявність $_SESSION['admin_id'].
- Видалення запису здійснюється через GET-параметр. Перед видаленням використовується підтвердження через JavaScript-діалог.

Адміністративна частина реалізована окремо через набір скриптів admin_*.php, що дозволяють керувати усіма сутностями (товари, категорії, замовлення, користувачі, контакти), забезпечуючи доступ лише після перевірки ролі. Така структуризація алгоритмів гарантує модульність, легкість розширення та підтримки кожного елемента системи. Наступним кроком буде вибір загальної архітектури сайту, що узгодить ці алгоритми з технологічними рішеннями і шаблонами проєктування.

2.2. Вибір архітектури сайту

Для розробки веб-сайту продажу книг обрано традиційну клієнт-серверну архітектуру на базі PHP та MySQL, що забезпечує надійність, простоту налаштування та можливість швидкого розгортання. На локальному етапі використовується платформа OpenServer, оскільки вона дозволяє одним кліком налаштувати потрібні версії Apache, PHP і MySQL, а також включає phpMyAdmin для зручної роботи з базою даних. Важливим аргументом на користь OpenServer є те, що в межах одного інсталяційного пакету розгортаються всі необхідні сервіси, а зміна конфігурацій (порти, налаштування PHP, версії модулів) відбувається через зручний інтерфейс, без ручного редагування файлів.

У ролі веб-сервера обрано Apache, що є стандартним рішенням для запуску PHP-скриптів. До ключових переваг відносять стабільність, широку поширеність і велику кількість налаштувань через .htaccess. Саме за допомогою файлу .htaccess реалізовано маршрутизацію книг, зручні URL-

адреси (наприклад, product/123 замість product.php?id=123). Додавання правил у .htaccess дозволяє легко змінювати структуру URL, не редагуючи безпосередньо код на сервері.

Як систему керування базами даних обрано MySQL, що сумісний з PHP і є чи не найбільш поширеним рішенням у сфері e-commerce. Завдяки цьому ми працюємо з реляційною моделлю, а phpMyAdmin дає змогу розробнику переглядати структуру таблиць, виконувати запити вручну, експортувати чи імпортувати бази.

На стороні клієнта для рендерингу динамічного контенту використовується стандартний набір технологій: HTML5, CSS3 і JavaScript. Проте щоб забезпечити плавну взаємодію без перезавантаження сторінки, у окремих місцях застосовано AJAX-запити. Наприклад, при підрахунку кількості непрочитаних повідомлень чи нових замовлень у шапці адмін-панелі реалізовано AJAX-опитування (polling) кожні кілька секунд, щоб відображати свіжу цифру поруч із іконкою дзвінка.

Першопричиною відмови від використання веб-сокетів (Socket.IO на базі Node.js) стала відносно невелика складність системи сповіщень. Оскільки необхідно лише повідомляти адміністратора про нові замовлення або зміни статусів, використання двостороннього з’єднання зі збереженням відкритого сокета було визнано надмірним. Натомість AJAX-технології, які реалізують періодичні запити (наприклад, кожні 10–15 секунд), повністю задовольняють вимогу своєчасного відображення повідомлень, одночасно не ускладнюючи архітектуру проєкту встановленням окремого Node-сервера. Такий підхід дозволяє зекономити ресурси сервера та спростити розробку на хостинг, оскільки достатньо підтримувати лише стандартне PHP-оточення.

Маршрутизація в межах проєкту побудована так, щоб кожна функціональна одиниця мала власний PHP-файл, а зрозумілі URL формувалися через правила .htaccess. Наприклад: RewriteRule ^product/([0-9]+)$ product.php?id=$1 [L,QSA]

Це дає змогу показувати користувачеві «красиві шляхи» та одночасно передавати необхідні GET-параметри всередині PHP-скриптів.

Для управління залежностями та спрощення роботи з бібліотеками було вирішено не вдаватися до складних фреймворків типу Laravel або Symfony, оскільки завдання проєкту порівняно вузьке, а використання чистого PHP забезпечує більшу прозорість коду. Проте при необхідності підключати сторонні класи (наприклад, для валідації форм чи роботи з PDF-файлами) можна використовувати Composer.

Особливу увагу приділено безпеці: весь вхід у файли адмін-панелі (admin_*.php) захищено через admin_header.php, який передає тільки тим користувачам, у яких встановлена сесійна змінна $_SESSION['admin_id'].

Такий вибір архітектури гарантує простоту розгортання, зрозумілу підтримку коду та можливість поступового розширення функціоналу (додавання платіжного шлюзу, впровадження багатомовності, інтеграція з зовнішніми API). Завдяки цьому ми отримуємо оптимальне поєднання швидкої розробки, безпеки, зручності адміністрування та задоволення користувацьких вимог щодо продуктивності й інтерфейсу.

Даний сайт можна покращити низкою додаткових функцій. По-перше, варто інтегрувати API «Нової Пошти» та інших поштових операторів, оскільки наразі використовується спрощена шаблонна форма введення адреси доставки. Додавши автоматичний розрахунок вартості та вибір найближчого відділення через API, ми значно полегшимо користувачеві процес оформлення замовлення та зменшимо кількість помилок при введенні адреси.

По-друге, бракує інтеграції платіжних систем, наприклад, LiqPay або Monobank API. Наразі оплата відбувається за класичною схемою «накладений платіж» або банківський переказ, що потребує ручного підтвердження коштів адміністратором. Впровадження LiqPay або Monobank API дозволить клієнту здійснювати оплату безпосередньо на сайті, отримувати підтвердження платежу в реальному часі й відразу після успішної транзакції автоматично розблоковувати доступ до електронної версії книги. Таким чином, користувач

зможе негайно почати читати придбану книгу, не очікуючи ручного підтвердження від адміністратора.

З огляду на відсутність цих механізмів, зараз замість API використовується проста текстова форма «Адреса доставки» без перевірки коректності. Інтеграція платіжного шлюзу і сервісів кур'єрської доставки значно підвищить рівень довіри та зручності, адже користувачі зможуть бачити остаточні тарифи на доставку, відслідковувати статус оплати й отримувати електронні копії книг одразу після підтвердження транзакції.

Наведені доповнення сприятимуть не лише зростанню кількості автоматизованих процесів, а й зниженню навантаження на адміністратора: замість ручної обробки платежів і оброблення помилок у введенні адреси, система сама перевірятиме дані та передаватиме інформацію до відповідних служб.

Висновки до розділу 2

У другому розділі було детально розглянуто ключові алгоритми роботи веб-сайту та обґрунтовано вибір його архітектури. Алгоритмічна частина продемонструвала, що кожен сценарій - від перегляду каталогу до оформлення замовлення і керування адмін-панеллю - розділено на окремі PHP-скрипти з чіткою послідовністю кроків. Такий підхід забезпечує прозорість роботи системи, дозволяє швидко відстежити та виправити помилки в різних функціональних блоках і полегшує подальше масштабування: за необхідності можна додати нові режими фільтрації, альтернативні способи оплати або розширити модуль блогу без втручання в основну бізнес-логіку.

Архітектурне рішення на базі PHP і MySQL (під керуванням OpenServer у локальному середовищі та Apache у продакшні) підтвердило свою доцільність: воно поєднує простоту налаштування, стабільність і можливість використання інструментів типу phpMyAdmin. Використання файлу .htaccess для налаштування «гарних» URL і маршрутизації гарантує, що навіть при зміні

структури шляхи залишаться інтуїтивно зрозумілими для користувачів і пошукових систем. Асинхронні AJAX-запити замість веб-сокетів у частині повідомлень (наприклад, оновлення статусу замовлень в адмінці) довели ефективність для задачі невеликої частоти оновлень, зберігаючи при цьому ресурси сервера.

Нарешті, архітектура закладає основу для подальшого розвитку: у майбутньому можливо легко інтегрувати API кур’єрських служб (Нова Пошта, Justin тощо), додати онлайн-платежі через LiqPay чи Monobank, а також упровадити багатомовність. Поява таких функцій знизить навантаження на адміністратора, автоматизує обробку адрес та оплату і поліпшить досвід кінцевого користувача, який зможе одразу після успішного платежу розпочати читання електронної книги. Таким чином, вибрана структура сайту та застосовані технології забезпечують баланс між простотою реалізації, безпекою й гнучкістю для майбутніх доповнень.

РОЗДІЛ ІІІ
ПРОЄКТУВАННЯ БАЗИ ДАНИХ WEB-САЙТУ

3.1. Обгрунтування вибору СУБД

Для реалізації веб-сайту продажу книг обрано реляційну систему керування базами даних MySQL (версія 8.0) з механізмом зберігання InnoDB. Даний вибір базується на детальному аналізі потреб та особливостей проєкту, а також характеристик і переваг обраного рішення. Нижче детально наведено обґрунтування цього рішення з урахуванням технічних, практичних та експлуатаційних факторів:

1. Відповідність реляційній моделі:

Проєкт складається з чітко структурованих, логічно взаємопов’язаних сутностей: книги (таблиця books), категорії (categories), користувачі (users), кошик (cart), замовлення (orders) та покупки (purchases). Для ефективної роботи з цими сутностями необхідна чітка модель взаємодії та зв’язків між ними. MySQL як реляційна СУБД надає зручні інструменти для моделювання складних зв’язків «один-до-багатьох», таких як: одна категорія може включати багато книг, один користувач може здійснювати багато замовлень. Використання зовнішніх ключів у MySQL з механізмом InnoDB гарантує автоматичну підтримку цілісності зв’язків та захист від випадкового порушення логічної структури бази даних. Наприклад, при видаленні категорії книги відповідні записи про книги можуть автоматично отримувати пусті значення (SET NULL), що зберігає цілісність інформації.

2. Підтримка транзакцій і цілісності даних:

Однією з ключових вимог до веб-магазину є гарантія цілісності та достовірності даних під час здійснення замовлень. Складні операції, такі як оформлення замовлення, потребують атомарності виконання. InnoDB забезпечує механізм транзакцій, завдяки чому вся група операцій (створення замовлення у таблиці orders, додавання відповідних записів у таблицю

purchases, оновлення статусу книги в кошик) або повністю виконується, або, у разі виникнення помилки чи збою, повністю скасовується (ROLLBACK). Таким чином, гарантована послідовність і повнота виконання операцій суттєво знижує ризик пошкодження або втрати даних.

3. Продуктивність і масштабованість:

MySQL із механізмом зберігання InnoDB демонструє високу продуктивність у типових сценаріях e-commerce, де основним навантаженням є пошук товарів, додавання в кошик, оформлення замовлень, читання та оновлення інформації про товари й користувачів. Система підтримує ефективну обробку запитів SELECT, INSERT, UPDATE, DELETE навіть при значних обсягах даних. Використання автоінкрементних первинних ключів (наприклад, id у таблицях books та orders) дозволяє швидко та безпечно генерувати унікальні ідентифікатори нових записів, забезпечуючи стабільність роботи бази. Крім того, MySQL дає змогу додавати індекси за полями, які часто використовуються у вибірках, що значно прискорює пошук необхідних даних у великих таблицях.

4. Простота адміністрування:

Одним із суттєвих факторів вибору MySQL стала його простота використання та адміністрування. Платформа OpenServer, яка застосовується для локального розгортання проєкту, містить вбудовану підтримку phpMyAdmin. Завдяки цьому адміністратор може швидко й зручно переглядати структуру бази, додавати нові поля, таблиці, здійснювати імпорт та експорт даних, створювати резервні копії, а також оперативно відновлювати інформацію у разі збою. Інтуїтивно зрозумілий інтерфейс phpMyAdmin дозволяє швидко адаптуватися навіть користувачам-початківцям, що знижує поріг входження та зменшує ймовірність помилок під час роботи з даними.

5. Широка підтримка та інтеграція

Ще однією важливою перевагою MySQL є його широка поширеність і підтримка у веб-розробці. СУБД має безліч готових бібліотек, драйверів та інструментів, що забезпечують безпроблемну інтеграцію з PHP та іншими

технологіями, які застосовуються у веб-проєкті. Така інтеграція дозволяє ефективно реалізовувати бізнес-логіку, пов’язану з роботою веб-ресурсу, і спрощує процес розробки та підтримки коду.

6. Висока надійність та стабільність

InnoDB у складі MySQL зарекомендувала себе як надійний механізм зберігання даних завдяки розвиненим засобам контролю цілісності, механізму блокувань рядків (row-level locking) та підтримці багатопоточності, що мінімізує затримки при одночасному доступі багатьох користувачів до даних. Ці характеристики надзвичайно важливі для проєктів, що очікують зростання кількості користувачів і транзакцій з часом.

7. Безпека та захист інформації

MySQL містить гнучку систему привілеїв, що дозволяє чітко визначати рівні доступу для різних категорій користувачів. Це забезпечує належний захист конфіденційної інформації користувачів (дані замовлень, особисті дані), а також мінімізує ризики несанкціонованого доступу до важливих ресурсів бази.

Таким чином, MySQL з механізмом зберігання InnoDB повністю задовольняє всі ключові вимоги веб-сайту продажу книг: ефективне моделювання зв’язків, забезпечення цілісності та безпеки даних, висока продуктивність і масштабованість, простота адміністрування та широка інтеграція з існуючими технологіями. Це робить його оптимальним вибором для розробки та подальшого розвитку онлайн-магазину книг, що відповідає сучасним стандартам і здатен витримувати зростаючі навантаження.

3.2. Структура таблиць бази даних

Для забезпечення стабільної роботи веб-сайту продажу книг необхідно створити централізований простір зберігання даних, який відповідає за зберігання всіх основних сутностей системи: від інформації про користувачів до даних про замовлення й товари. Ядро цього простору – реляційна база

даних MySQL/InnoDB, де кожна таблиця виконує чітко визначену роль та дозволяє розподілити логіку бізнес-процесів між окремими сутностями.

Загальний принцип побудови полягає в розподілі даних на таблиці зі зрозумілими атрибутами: наприклад, окрема таблиця зберігає користувачів, інша – книги, ще одна – жанри. Це дає змогу уникнути дублювання інформації, підтримувати консистентність та легко виконувати запити для відображення даних на відповідних сторінках сайту. Таким чином, при завантаженні каталогу використовується тільки одна таблиця з описом книг, а для формування фільтрів із жанрів звернення йде до окремої таблиці категорій.

Ключовим аспекти при проектуванні є забезпечення контролю доступу: таблиця користувачів містить не тільки дані для автентифікації, а й визначає роль (звичайний клієнт або адміністратор), що обмежує та розширює доступ до адмін-панелі. При цьому інформація про замовлення рознесена в дві таблиці: одна – загальні дані про самі замовлення, інша – деталізація придбаних позицій, що дозволяє легко вирахувати кількість кожної книги в замовленні й умови доступу до електронних копій.

Окремою групою йдуть таблиці, які тимчасово зберігають дані – наприклад, кошик, де фіксуються товари, призначені до оформлення, але ще не оплачені. Такий підхід дає можливість зберігати стан кошика між сесіями користувача та відновлювати його навіть після виходу з сайту.

Завдяки чіткому розподілу на таблиці можна гарантовано працювати з великими обсягами даних, легко розширювати функціонал (додавати рейтинги, знижки, нові атрибути товарів) та забезпечувати прозору аналітику. У результаті база даних стає не лише сховищем, а й основою для гнучкої логіки взаємодії між користувачем і системою, адаптованої до змін і зростання проєкту.

Таблиця books (табл. 3.1) зберігає інформацію про всі книги, доступні для продажу: назву, короткий опис, ціну, шляхи до зображень обкладинок (передньої, задньої, корінця) та, за наявності, до електронного файлу. На рівні

бізнес-логіки ці дані формують каталог і детальні сторінки товарів, дозволяють відображати книги у вигляді карток із базовими атрибутами.

Таблиця 3.1

Опис таблиці books

Поле	Тип даних	NULL	Опис
id	INT	NOT NULL	Унікальний ідентифікатор книги (PK, AUTO_INCREMENT)
name	VARCHAR(100)	NOT NULL	Назва книги
price	INT	NOT NULL	Ціна в гривнях
image	VARCHAR(255)	NULL	Ім’я файлу з обкладинкою(Для сайту)
file	VARCHAR(255)	NULL	Електронна версія книги
description	TEXT	NOT NULL	Опис книги
category_id	INT	NULL	Зовнішній ключ - categories.id (жанр);
cover_front	VARCHAR(255)	NULL	Передня обкладинка
cover_back	VARCHAR(255)	NULL	Задня обкладинка
spine	VARCHAR(255)	NULL	Корінець книги

Таблиця categories (табл. 3.2) містить перелік жанрів або категорій, до яких можуть належати книги. У загальному вигляді вона служить для фільтрації каталогу: на сайті користувач обирає жанр із випадаючого списку, а система відбирає записи з books, позначені відповідним category_id. Таким чином, categories задає ієрархію тем, роблячи пошук і навігацію зручнішими.

Таблиця 3.2

Опис таблиці categories

Поле	Тип даних	NULL	Опис
id	INT	NOT NULL	Унікальний ідентифікатор категорії (PK, AUTO_INCREMENT)
name	VARCHAR(100)	NOT NULL	Назва жанру

Таблиця users (табл. 3.3) містить облікові дані зареєстрованих відвідувачів і адміністраторів: ім’я (або нікнейм), email, хеш пароля та роль користувача (user чи admin). Вона є центром ідентифікації - у ній фіксується, хто здійснює замовлення, у кого зберігся кошик, і хто має доступ до панелі керування. Зміна ролі на адміна відкриває доступ до розширених можливостей з управління контентом і замовленнями.

Таблиця 3.3

Опис таблиці users

Поле	Тип даних	NULL	Опис
id	INT	NOT NULL	Унікальний ідентифікатор юзера (PK, AUTO_INCREMENT)
name	VARCHAR(100)	NOT NULL	Ім’я користувача
email	VARCHAR(100)	NOT NULL	Електронна адреса
password	VARCHAR(255)	NOT NULL	Хешований пароль (збережено через PHP password_hash)
user_type	VARCHAR(20)	NOT NULL	Роль користувача: 'user' або 'admin' (за замовчуванням 'user')

Таблиця cart (табл. 3.4) служить для тимчасового зберігання вмісту кошика кожного користувача протягом сесії і між сеансами. Коли клієнт додає книгу до кошика, система фіксує в cart його user_id, назву книги на момент додавання, актуальну ціну, кількість примірників і шлях до зображення обкладинки для відображення мініатюри. Завдяки цьому навіть якщо браузер буде закрито або користувач перейде на інший пристрій під тим самим обліковим записом, вміст кошика залишиться незмінним і доступним після повторного входу. При перегляді вмісту кошика на сторінці виводяться всі записи з цієї таблиці, включно з деталями кожної позиції (назва, зображення, кількість, ціна за одиницю), а також автоматично обчислюється й відображається загальна сума замовлення. Це забезпечує зручність

повернення до покупки й мінімізує ризик втрати обраних товарів у процесі оформлення.

Таблиця 3.4

Опис таблиці cart

Поле	Тип даних	NULL	Опис
id	INT	NOT NULL	Унікальний ідентифікатор рядка в кошику (PK, AUTO_INCREMENT)
user_id	INT	NOT NULL	Ідентифікатор користувача, що додав товар (FK - users.id)
name	VARCHAR(100)	NOT NULL	Назва книги у кошику
price	INT	NOT NULL	Ціна книги на момент додавання
quantity	INT	NOT NULL	Кількість примірників у кошику
image	VARCHAR(255)	NOT NULL	Файл обкладинки

Таблиця orders (табл. 3.5) акумулює дані про всі завершені замовлення в системі, зберігаючи унікальний номер кожного замовлення, ідентифікатор користувача, а також повні контактні дані отримувача - ім’я, телефон, email і детальну адресу доставки. Крім цього, фіксується обраний спосіб оплати (наприклад, «накладений платіж» або «онлайн»), загальна кількість книг у замовленні, їх назви у вигляді структурованого списку та підсумкова сума. Поле payment_status дає змогу відстежувати стан оплати: від «pending» до «completed». Завдяки інформації з цієї таблиці веб-сайт може демонструвати кожному користувачеві історію покупок з поточним статусом та деталями доставки, а адміністратор отримує можливість оперативно реагувати на нові замовлення та оновлювати їхній статус у режимі реального часу. Це також створює основу для подальшої аналітики продажів та оптимізації логістичних процесів.

Таблиця 3.5

Опис таблиці orders

Поле	Тип даних	NULL	Опис
id	INT	NOT NULL	Унікальний ідентифікатор замовлення (PK, AUTO_INCREMENT)
user_id	INT	NOT NULL	Ідентифікатор клієнта
name	VARCHAR(100)	NOT NULL	Ім’я отримувача
number	VARCHAR(12)	NOT NULL	Номер телефону отримувача
email	VARCHAR(100)	NOT NULL	Email отримувача
method	VARCHAR(50)	NOT NULL	Метод оплати
address	VARCHAR(500)	NOT NULL	Адреса доставки
total_books	TEXT	NOT NULL	Список назв книг і кількості
total_price	INT	NOT NULL	Загальна сума замовлення
placed_on	VARCHAR(50)	NOT NULL	Дата та час оформлення
payment_status	VARCHAR(20)	NOT NULL	Статус оплати

Таблиця purchases (табл. 3.6) деталізує кожне окреме замовлення, зберігаючи інформацію про кожну придбану позицію: тут фіксуються унікальний ідентифікатор користувача, ідентифікатор книги, номер відповідного замовлення, а також точна дата й час покупки. Завдяки цим даним легко визначити, які саме видання входили до конкретного замовлення, відстежити хронологію придбань і структуру багатопозиційних замовлень. Для електронних книг ця таблиця виконує ще й функцію перевірки прав на читання: наявність запису в purchases слугує сигналом для системи, що

користувач отримав дозвіл на відкриття та перегляд відповідного файлу онлайн.

Таблиця 3.6

Опис таблиці purchases

Поле	Тип даних	NULL	Опис
id	INT	NOT NULL	Унікальний ідентифікатор запису (PK, AUTO_INCREMENT)
user_id	INT	NOT NULL	Ідентифікатор користувача, який придбав книгу
book_id	INT	NOT NULL	Ідентифікатор книги (FK - books.id)
order_id	INT	NOT NULL	Ідентифікатор замовлення (FK - orders.id)
purchase_date	DATETIME	NULL	Дата й час придбання

Таблиця message (табл. 3.7) акумулює звернення з форми «Контакти»: ім’я й email автора повідомлення, номер телефону, й текст самого звернення. Адміністратор може переглядати ці записи в зручному інтерфейсі, сортувати їх, відповідати на запити клієнтів, видаляти оброблені повідомлення та зберігати повну історію звернень для подальшого аналізу і якісної підтримки сайту.

Таблиця 3.7

Опис таблиці message

Поле	Тип даних	NULL	Опис
id	INT	NOT NULL	Унікальний ідентифікатор повідомлення (PK, AUTO_INCREMENT)
user_id	INT	NOT NULL	Ідентифікатор авторизованого користувача (FK → users.id)
name	VARCHAR(100)	NOT NULL	Ім’я відправника
email	VARCHAR(100)	NOT NULL	Email відправника
number	VARCHAR(12)	NOT NULL	Номер телефону відправника
message	VARCHAR(500)	NOT NULL	Текст повідомлення

Усі згадані таблиці побудовані з урахуванням нормалізації: кожна відповідає за свій тип даних і сприяє мінімізації дублювання. Завдяки цьому структура бази даних залишається гнучкою та легко піддається розширенню - наприклад, за потреби можна додати поля для рейтингу книг, багатомовних описів чи зберігання промокодів. Кожна таблиця бере участь у життєвому циклі замовлення або взаємодії з сайтом: books і categories формують інвентар і фільтрацію, users і cart опікуються особистими даними та кошиком, orders і purchases забезпечують фіксацію купівель, а message - комунікацію з клієнтом. Усе це разом забезпечує послідовну та зрозумілу роботу інтерфейсу та бізнес-логіки веб-сайту.

3.3. Реляційний зв’язок між таблицями

У базі даних усі сутності пов’язані між собою через чітко визначені реляційні зв’язки, що забезпечує не лише цілісність даних, а й дозволяє гнучко й ефективно виконувати навіть доволі складні SQL-запити з об’єднаннями та фільтрацією. Для наочності та полегшення розуміння було створено детальну схему зв’язків таблиць (Рис. 3.1.), яка демонструє ключові відношення між основними сутностями.

Основні зв’язки:

- Між таблицями books і categories існує зв’язок «багато до одного»(many-to-one): кожна книга (books.category_id) належить одній категорії (categories.id), а в одній категорії може бути багато книг. При цьому у полі category_id у таблиці books може зберігатися значення NULL, що означає відсутність категорії для певної книги (наприклад, якщо жанр видалено). У разі видалення категорії відповідні category_id у книгах автоматично обнуляються (ON DELETE SET NULL), зберігаючи записи про самі книги.
- Зв’язок між users та cart «один до багатьох» (one-to-many): кожен користувач(users.id) може мати кілька рядків у cart (по одному для

кожної книги або оновлення кількості), а в кожному рядку кошика (cart.user_id) зберігається ID того, кому належить корзина. Це дозволяє при вході користувача підвантажувати всі товари, що він додав раніше, та оновлювати їх кількість незалежно для кожного продукту.

- Між users і orders діє зв’язок «один до багатьох»: один користувач (orders.user_id) може оформити будь-яку кількість замовлень, причому кожне замовлення належить лише цьому конкретному користувачеві. Це забезпечує чітку історію транзакцій і дає змогу швидко відфільтрувати всі замовлення конкретної особи.
- Таблиця purchases є зв’язуючою ланкою між orders, users і books: кожен запис із purchases.order_id належить певному замовленню (orders.id), кожен purchases.user_id – конкретному користувачеві, а кожен purchases.book_id – певній книзі. Таким чином формується зв’язок «багато до одного» у кожному напрямку: одне замовлення може складатися з декількох придбаних книг, одне придбання належить лише одному замовленню та одному користувачу; і одна книга може бути придбана в різних замовленнях різними користувачами. Завдяки цим зв’язкам сайт може за необхідності показати детальні дані, наприклад, всі користувачі, які купили певну книгу, або всі книги в конкретному замовленні.
- У таблиці message поле user_id пов’язане із users.id, що дозволяє зберігати історію звернень від зареєстрованих клієнтів. Оскільки повідомлення створюються під час відправлення контактної форми, user_id завжди вказує на існуючий обліковий запис. Завдяки цьому адміністратор може побачити, хто писав запит, а система може використовувати email/ім’я з users для швидшого встановлення контакту.

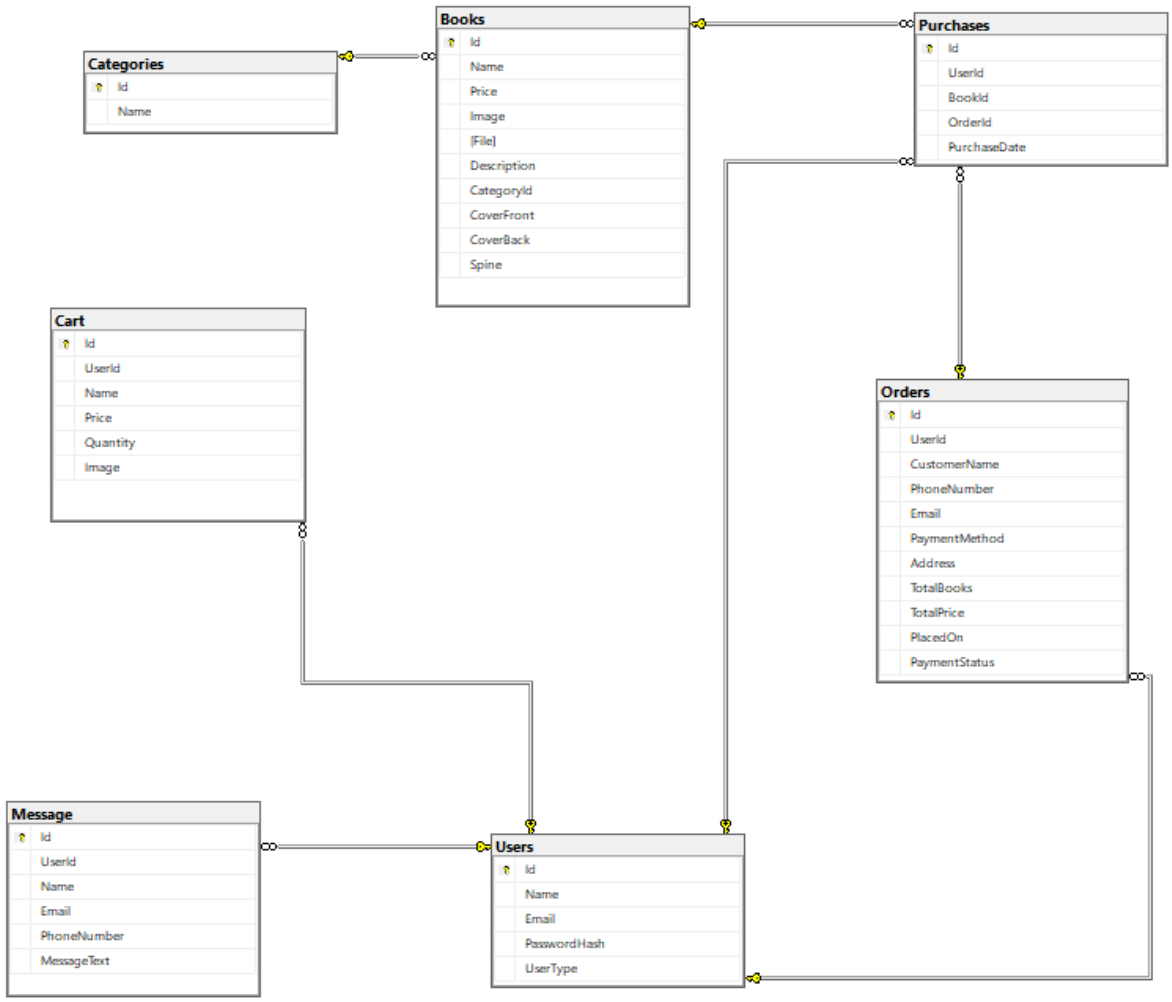

Рис. 3.1. Схема бази даних

Завдяки продуманим реляційним зв’язкам між таблицями (books–categories, users–cart, users–orders, purchases та message) досягається цілісність даних, що суттєво спрощує формування складних запитів і підтримку бізнес-логіки системи. Це забезпечує надійну основу для масштабування та подальшого розвитку проєкту.

Висновки до розділу 3

У третьому розділі було обґрунтовано вибір реляційної СУБД MySQL/InnoDB, розкрито структуру основних таблиць та описано реляційні зв’язки між ними. З’ясовано, що саме реляційна модель найкраще відповідає

вимогам проєкту: вона дозволяє чітко розмежувати сутності-книги, категорії, користувачів, замовлення та повідомлення-забезпечуючи цілісність і консистентність даних.

Аналіз структури таблиць показав, що кожна з них виконує свою роль: books формує основу каталогу, categories забезпечує фільтрацію, users-позначує права доступу та автентифікацію, cart і orders гарантують коректне зберігання тимчасових і завершених покупок, а purchases деталізує кожен набір придбаних позицій. Окрема таблиця message покриває комунікацію з клієнтами через форму «Контакти». Така нормалізована структура уможливлює просте та прозоре масштабування: у майбутньому можна додати рейтинги, промокоди чи багатомовні назви, не порушивши існуючої логіки.

Реляційні зв’язки гарантують узгодженість даних під час видалення чи оновлення записів. З допомогою зовнішніх ключів із налаштуваннями ON DELETE та ON UPDATE система уникає «сирітських» записів і підтримує цілісність.

Отже, побудована база даних слугує надійним фундаментом для роботи веб-сайту продажу книг, забезпечуючи коректну взаємодію між клієнтською частиною, логікою кошика й оформлення замовлень, а також адміністративними модулями. Згодом цю структуру легко можна розширити відповідно до нових бізнес-вимог.

РОЗДІЛ IV
РОЗРОБКА WEB-САЙТУ ТА ДОДАТКУ

4.1. Обгрунтування вибору технологій розробки

У проєкті поєднується дві основні платформи: веб-сайт для продажу книг і десктопний 3D-переглядач на основі WinForms + OpenGL. Для кожної платформи обрано набір інструментів і технологій, які найкраще відповідають обмеженням, вимогам до продуктивності й швидкому розгортанню. Нижче наведено мотиви вибору ключових компонентів.

1. Серверна частина (Back-end)

- Мова PHP (версія 8). Має вбудовану підтримку роботи з MySQL/MariaDB, що дозволяє виконувати підготовлені вирази. Велика спільнота й обширна документація дають змогу швидко знайти рішення поширених проблем. Для невеликих і середніх завдань не потрібно використовувати важкий фреймворк - достатньо чистого PHP, що забезпечує гнучкість і прозорість коду
- Apache HTTP Server. Здатність гнучко налаштовувати маршрутизацію через файл .htaccess. Стабільність та можливість використовувати на будь-якому хостингу, що підтримує Linux/Windows.
- MySQL(версія 8.0). Підтримує реляційні зв’язки. Має прості інструменти резервного копіювання і можливість адміністрування через phpMyAdmin. Добре масштабується під стандартні e-commerce-навантаження (SELECT, INSERT, UPDATE над сотнями тисяч записів).
- OpenServer. Містить одночасно Apache, PHP, MySQL (або MariaDB) і phpMyAdmin у єдиному пакетному інсталяторі. Дозволяє легко перемикатися між різними версіями PHP та MySQL за кілька кліків, що важливо для тестування сумісності на різних середовищах. Спрощує процес налаштування: не потрібно окремо інсталювати кожен компонент або налаштовувати права доступу вручну.

2. Клієнтська частина (Front-end)

- HTML5 та CSS3. Підтримуються всі сучасні браузери (Chrome, Firefox, Edge, Safari). Flexbox і CSS Grid дають змогу створювати адаптивний дизайн без великих витрат часу: макети легко розбиваються на колонки й ряди, елементи динамічно масштабуються під різні ширини екрану.
- JavaScript + AJAX. використовуються для підрахунку нових сповіщень у шапці адмін-панелі (наприклад, нові замовлення), що дозволяє оновлювати цю інформацію без повного перезавантаження сторінки . DOM-маніпуляції - застосовуються для відображення підказок, повідомлень про успіх/помилку, а також для роботи з формами (наприклад, валідація на клієнтській стороні, автозакриття повідомлень, слайдери тощо).

3. Десктопна частина (WinForms + OpenGL)

- WinForms. Дає змогу для елементів керування (ListBox, Buttons, GLControl), що дозволяє швидко створити UI без написання великого об’єму розміткового коду.
- OpenTK. OpenTK є обгорткою для OpenGL Клас GLControl із простим API (Load, Resize, Paint) забезпечує інтеграцію OpenGL-контексту в WinForms-форму. OpenTK містить утиліти для роботи зі векторами та матрицями (OpenTK.Mathematics), що спрощує обчислення.
- MySqlConnector (.NET-бібліотека). Легко підключається до MySQL-бази нашого сервера, де розгорнуто веб-сайт, за використання однакової рядкової константи підключення. Підтримує асинхронні методи ExecuteReaderAsync(). Список книг у BookRepository.GetAll() читається один раз при старті форми, після чого передається до ListBox для вибору. Це дозволяє десктоп-додатку працювати автономно до тих пір, поки сервер доступний.
- WinForms Designer + Visual Studio. Керувати залежностями проекту через BookViewer3D.csproj (налаштування .NET 8, пакети NuGet OpenTK.WinForms, MySqlConnector). Мати вбудовані інструменти

налагодження (breakpoints) для швидкого виявлення проблем із рендерингом чи підключенням до БД.

Загалом, обраний набір технологій для веб та десктопної частин проєкту створює збалансовану екосистему, де кожен компонент працює над своєю ділянкою функціоналу: PHP + MySQL керують даними та бізнес-логікою сайту, а C# + OpenTK відповідають за візуальну взаємодію з 3D-книгою в десктопному додатку. Це дозволяє поєднати переваги обох світів - негайну доступність у браузері та інтерактивність у віконній версії.

4.2. Інтерфейс-користувача сайту

Першим кроком для користувача або адміністратора є реєстрація (bookstore/register) та авторизація (bookstore/login). Для реєстрації необхідно вказати базові дані: ім’я, email, пароль та підтвердження пароля (Рис. 4.1.). Після успішної реєстрації користувач може одразу увійти в систему, ввівши email і пароль (Рис. 4.2.).

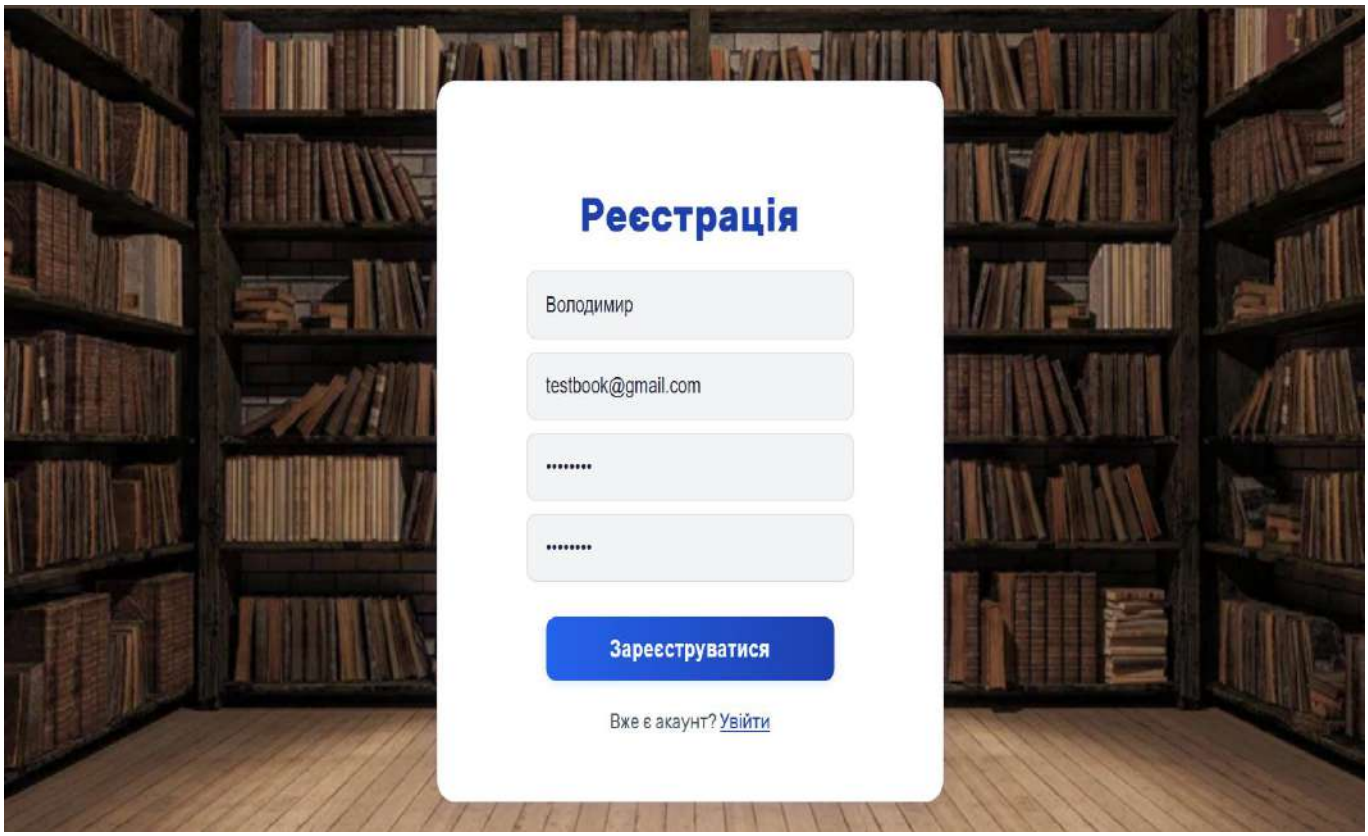

Рис. 4.1. Сторінка реєстрації

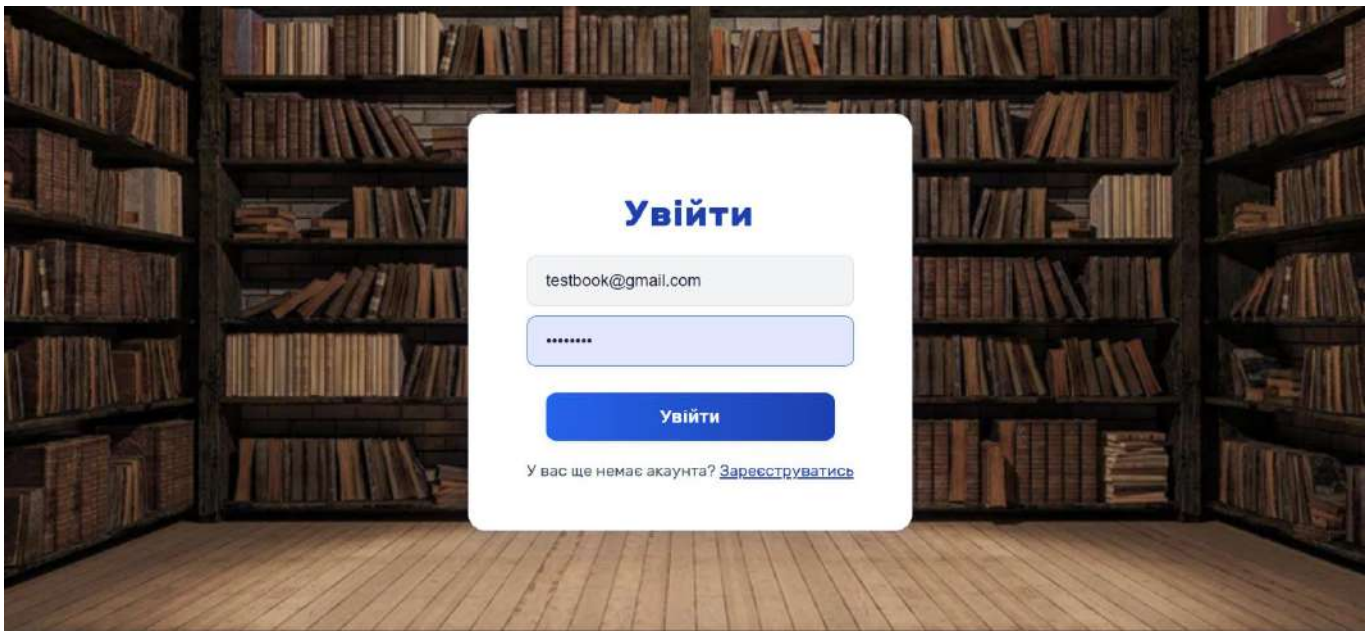

Рис. 4.2. Вхід до акаунту

Після авторизації відкривається головна сторінка (bookstore/home). На ній розміщено невеликий блок з рекомендаціями товарів у вигляді слайдера: він автоматично перемикає товари або дозволяє користувачеві самостійно гортати пропозиції (Рис. 4.3).

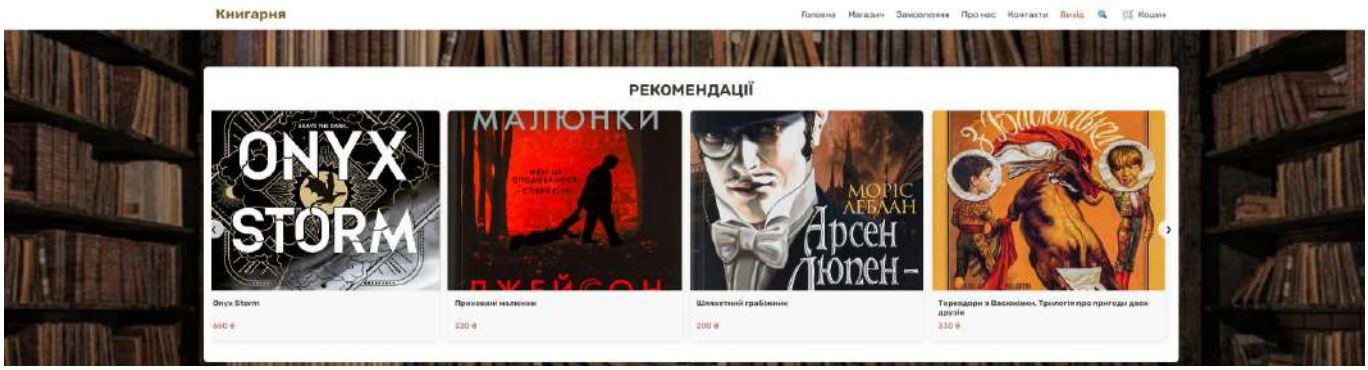

Рис. 4.3. Головна сторінка сайту

У шапці (header.php) розташоване основне меню навігації - посилання на різні розділи сайту. Якщо користувач додає товар у кошик, у правому верхньому куті відображається кількість позицій у кошику (Рис. 4.4).

Рис. 4.4. Навігація для користувача

Сторінка «Про нас» (bookstore/about) містить коротку інформацію про магазин, що дає змогу користувачу швидко ознайомитися з метою та перевагами проєкту (Рис. 4.5).

Рис. 4.5. Сторінка сайту About

На сторінці «Магазин» (bookstore/shop) представлено каталог книг із вказанням ціни, назви, категорії та опису (Рис. 4.6). Користувач може натиснути на будь-яку книгу, щоб перейти до її детальної сторінки. Також на рівні каталогу доступна фільтрація: можна обрати категорію та вказати ціновий діапазон для звуження результатів пошуку. Якщо користувач уже придбав електронну версію книги, на сторінці книги відобразиться кнопка «Читати», що дозволяє читати книгу онлайн (Рис. 4.7).

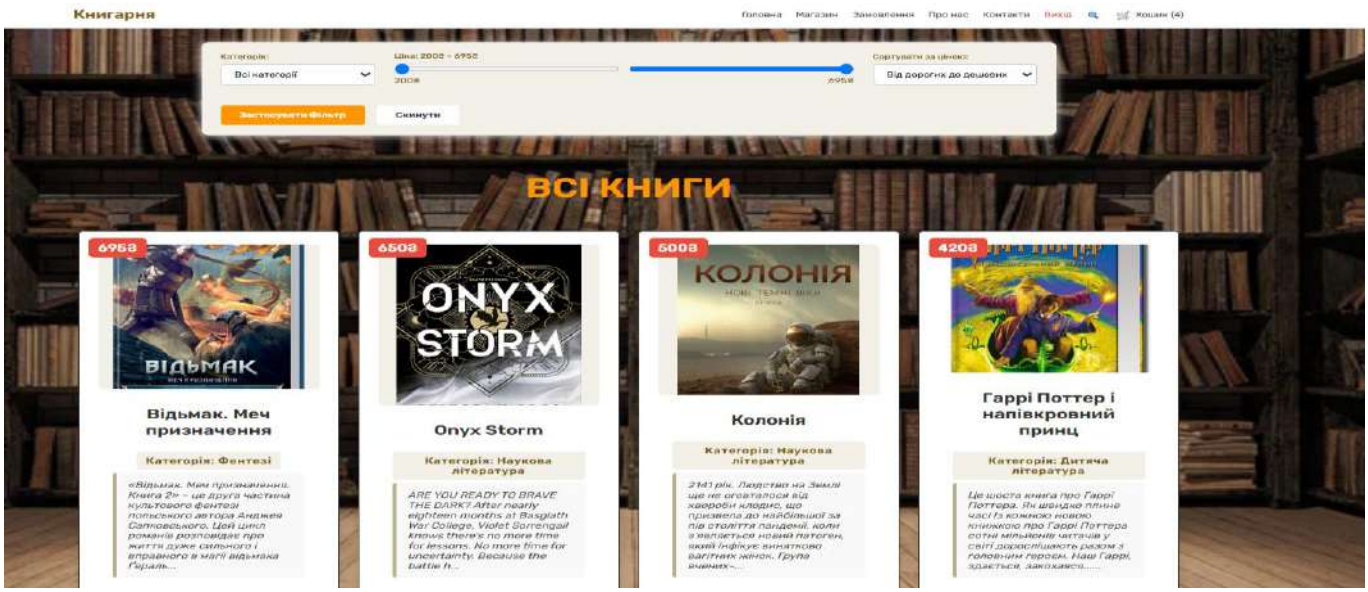

Рис. 4.6. Сторінка сайту Shop

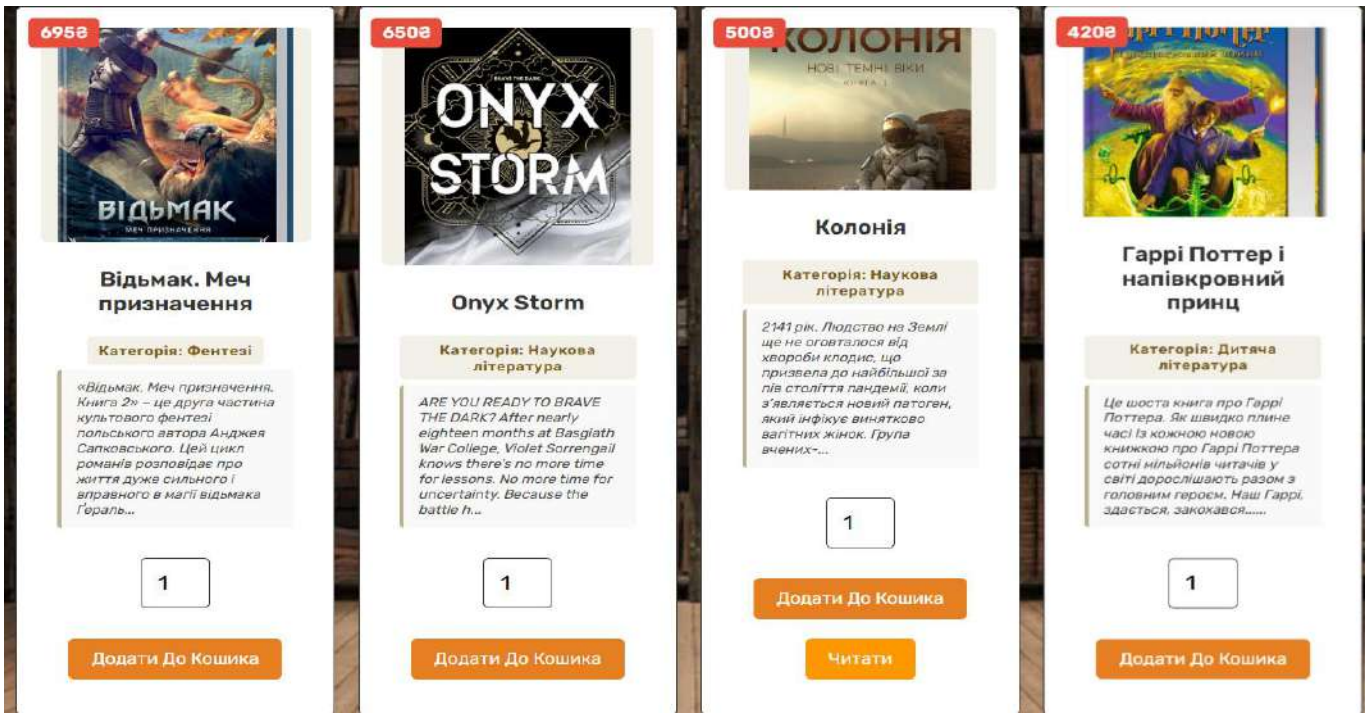

Рис. 4.7. Сторінка Shop після того, як користувач купив «Колонія»

У картці товару (bookstore/product/id) відображається докладна інформація: велика обкладинка, повний опис, ціна та можливість вказати кількість для додавання в кошик. Під основним описом розташований блок рекомендованих книг того самого жанру, що сприяє знайомству користувача з подібними позиціями (Рис. 4.8).

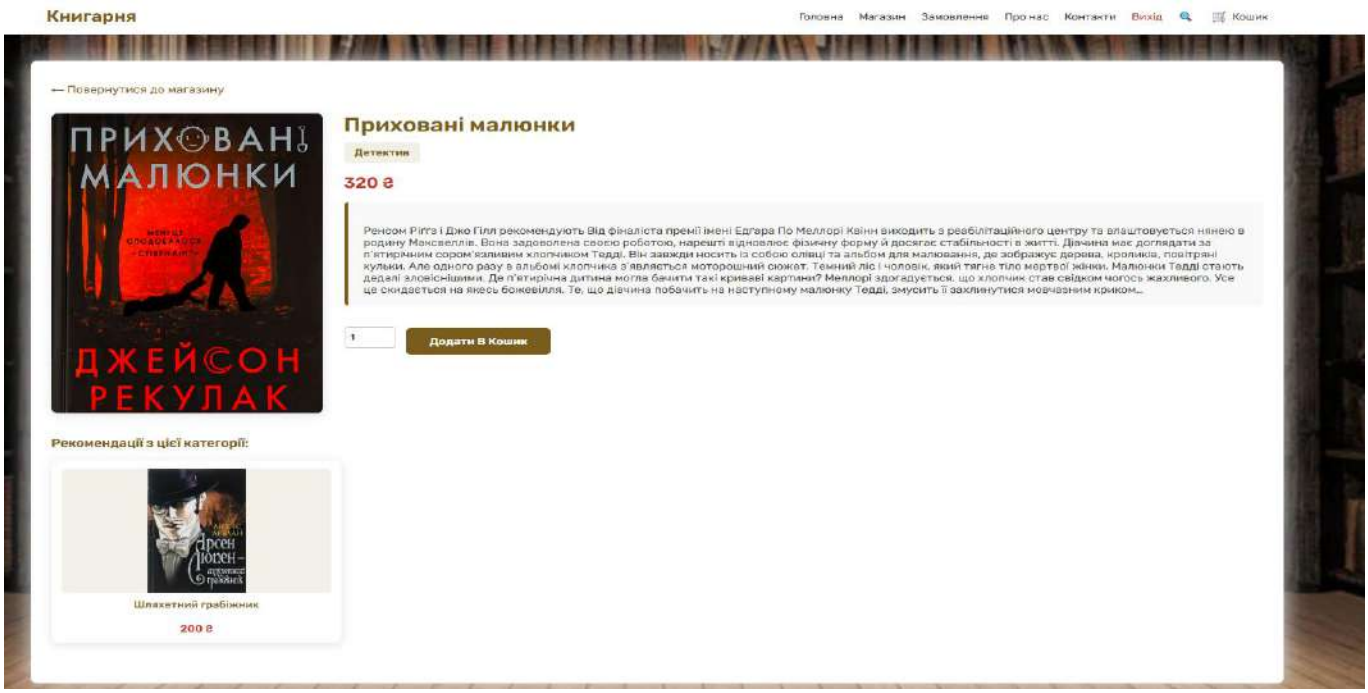

Рис. 4.8. Сторінка книги

На сторінці пошуку (bookstore/search) користувач може знайти книгу за назвою. Пошук відбувається через одне поле вводу, а результати виводяться у вигляді карток, аналогічних до каталогу (Рис. 4.9).

Рис. 4.9. Сторінка сайту Search

Коли користувач додає товари до кошика, він може перейти на сторінку кошика (bookstore/cart). Тут відображаються всі позиції з інформацією про назву, ціну за одиницю, кількість та підсумкову суму для кожного товару. Користувач може збільшити або зменшити кількість примірників, видалити товари або очистити увесь кошик. Під таблицею кошика відображається загальна сума за всі книги та кнопка «Оформити замовлення» (Рис. 4.10).

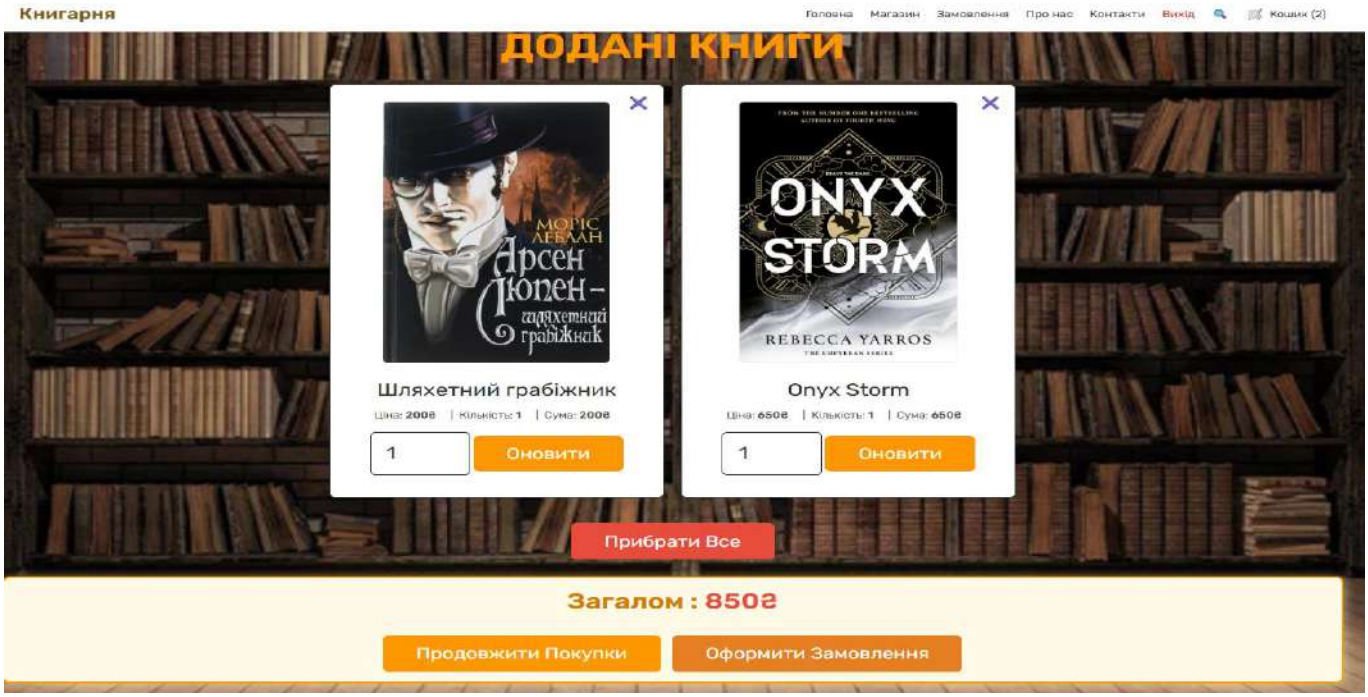

Рис. 4.10. Сторінка кошику товарів

На сторінці оформлення замовлення (bookstore/checkout) користувач заповнює форму з контактними даними: ім’я, телефон, email, місто доставки, відділення пошти, додаткову адресу та обирає спосіб оплати. Після підтвердження форми замовлення створюється в базі, і користувач отримує повідомлення про успішне здійснення покупки (Рис. 4.11).

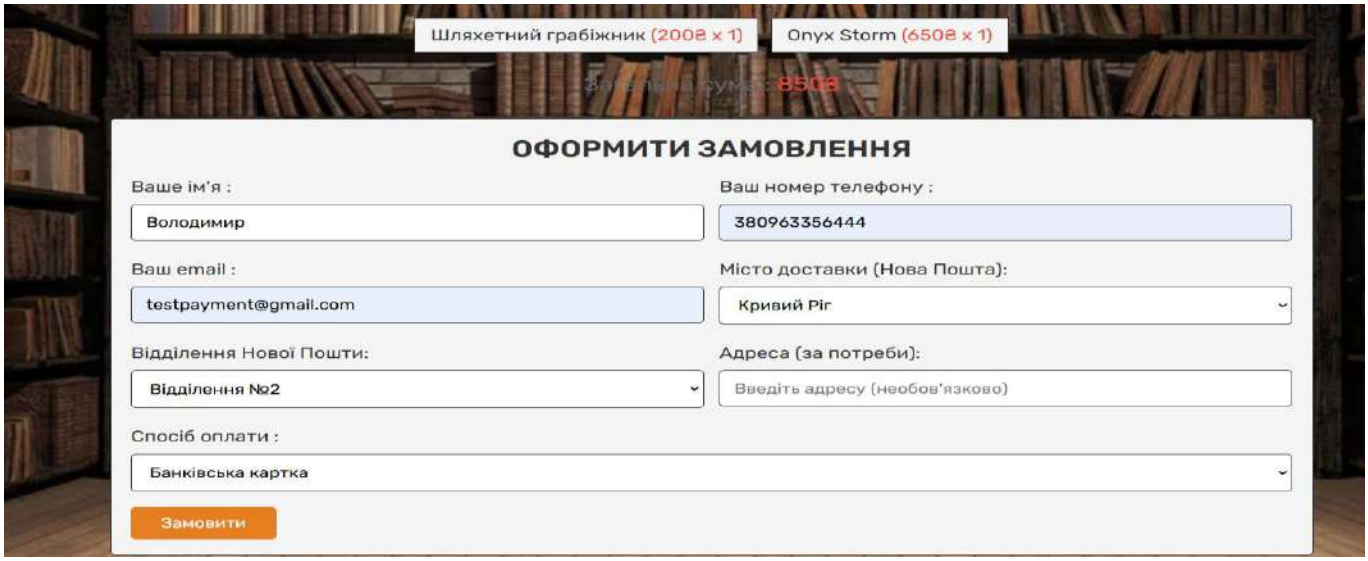

Рис. 4.11. Сторінка оформлення замовлення

У розділі «Мої замовлення» (bookstore/orders) відображається перелік усіх оформлених замовлень: номер замовлення, контактні дані, статус оплати та загальна сума. Якщо статус «Очікує оплати», доступ до електронної версії книги заблокований, а після зміни статусу на «Оплачено» користувач отримує кнопку «Читати», яка дозволяє відкрити електронну книгу (Рис. 4.12).

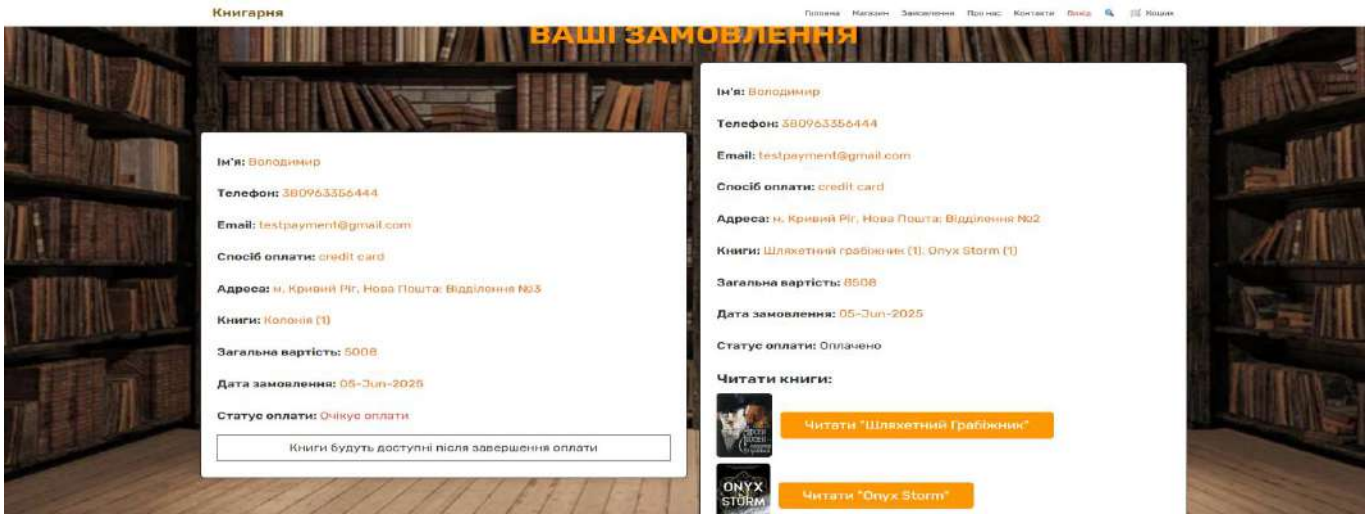

Рис. 4.12. Сторінка замовлень користувача

Якщо у користувача виникають запитання чи побажання, він може перейти до сторінки «Контакти» (bookstore/contact). Тут достатньо вказати ім’я, email, номер телефону та текст повідомлення. Після відправлення запит надходить до таблиці повідомлень у базі, і адміністратор може оперативно відповісти користувачу (Рис. 4.13).

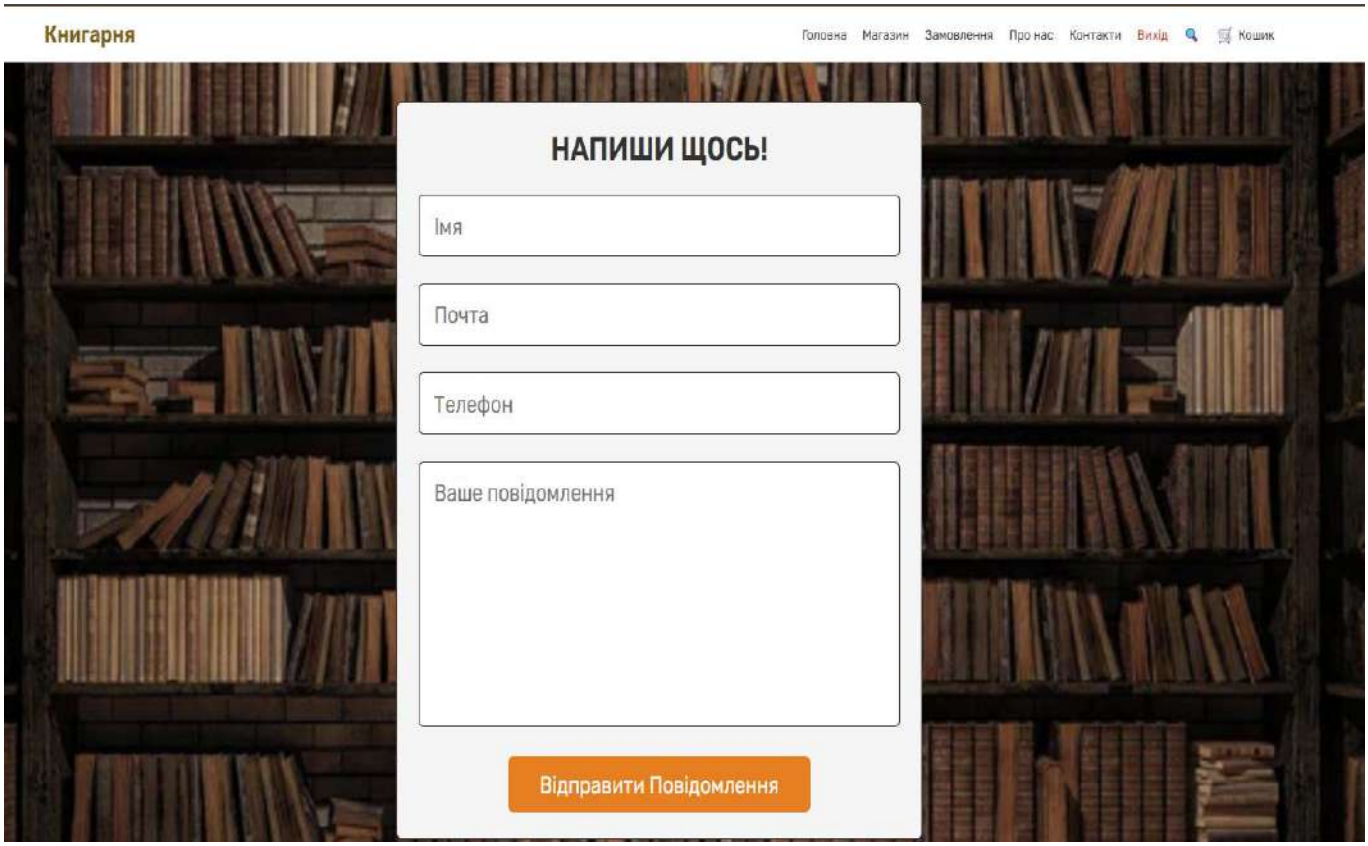

Рис. 4.13. Сторінка сайту contact

Інтерфейс користувача побудовано таким чином, щоб забезпечити інтуїтивно зрозумілу навігацію та мінімізувати кількість кроків для здійснення основних дій: реєстрації, перегляду каталогу, пошуку та оформлення замовлення. Використання єдиного шаблону шапки, чітке відображення стану кошика та доступності електронної версії після оплати створюють зручний досвід для покупця. Крім того, функціональні можливості сторінок «Кошик», «Оформлення» та «Мої замовлення» забезпечують повний цикл покупки від вибору книги до завершення транзакції. Таким чином, інтерфейс поєднує простоту, логічну послідовність і необхідний набір інструментів для комфортного користування сайтом.

4.3. Адміністративна частина сайту

Завдання адміністратора полягає у перевірці вхідних сповіщень про замовлення, аналізі статистики продажів, а також у створенні та редагуванні книг і категорій. Для входу в адмін-панель використовується та ж форма авторизації, що й для звичайних користувачів (bookstore/login). Якщо введені email та пароль належать запису з полем user_type = 'admin', користувача перенаправляє на сторінку адміністратора замість стандартної головної.

Першою у панелі адміністратора є сторінка «Дашборд» (bookstore/admin), де відображається ключова інформація:

- сума замовлень, що ще очікують оплати («Очікується оплата»).
- сума вже оплачених замовлень («Оплачено»).
- загальна кількість оформлених замовлень.
- кількість книг, що є в каталозі.
- кількість звичайних користувачів.
- Кількість адміністраторів.
- кількість нових повідомлень від клієнтів.

Ці показники допомагають швидко оцінити фінансовий стан магазину й активність користувачів (Рис. 4.14).

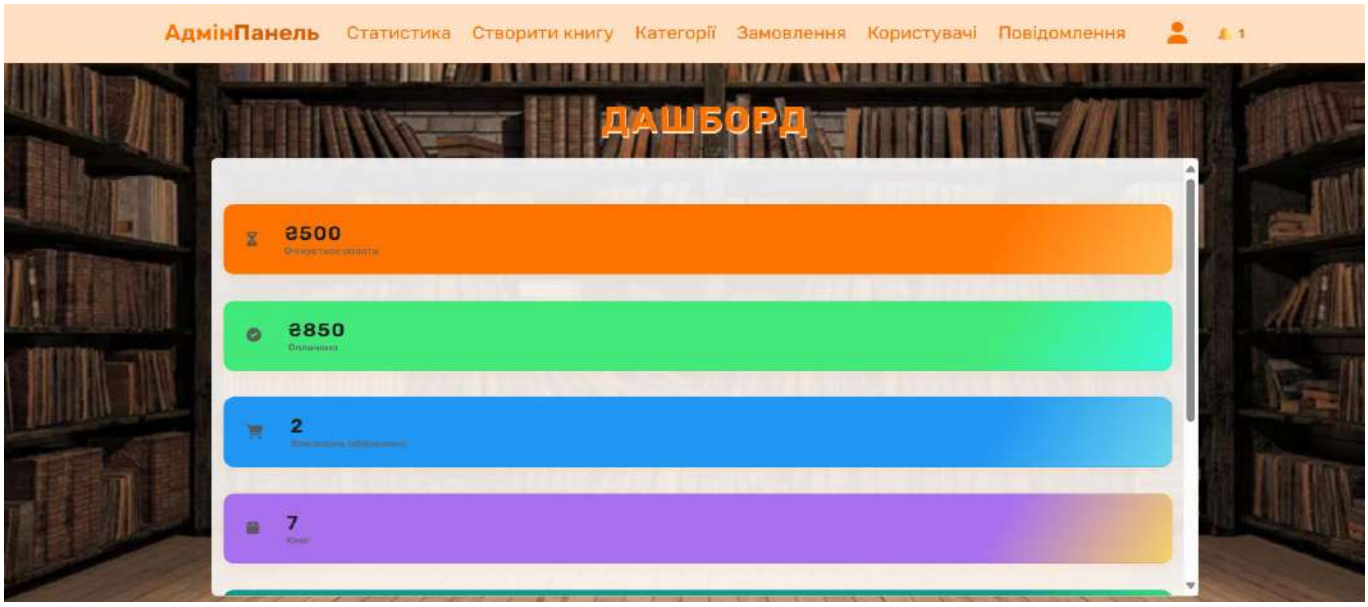

Рис. 4.14. Сторінка адміна для статистики

У навігації адміністратора розміщено посилання на основні розділи: створення й редагування книг, керування категоріями, перегляд замовлень, список користувачів та повідомлень від клієнтів. Також поряд знаходиться іконка дзвінка, яка вказує кількість необроблених замовлень (Рис. 4.15).

Рис. 4.15. Навігація адміна

На сторінці «Створити книгу» (bookstore/admin/products) адміністратор вводить назву видання, ціну, опис і обирає категорію. Далі завантажує основну обкладинку, файл електронної версії (зазвичай PDF) і три зображення для 3D-перегляду: передню обкладинку, задню обкладинку та корінець. Після збереження книга з’являється в базі даних і стає доступною в каталозі магазина (Рис. 4.16).

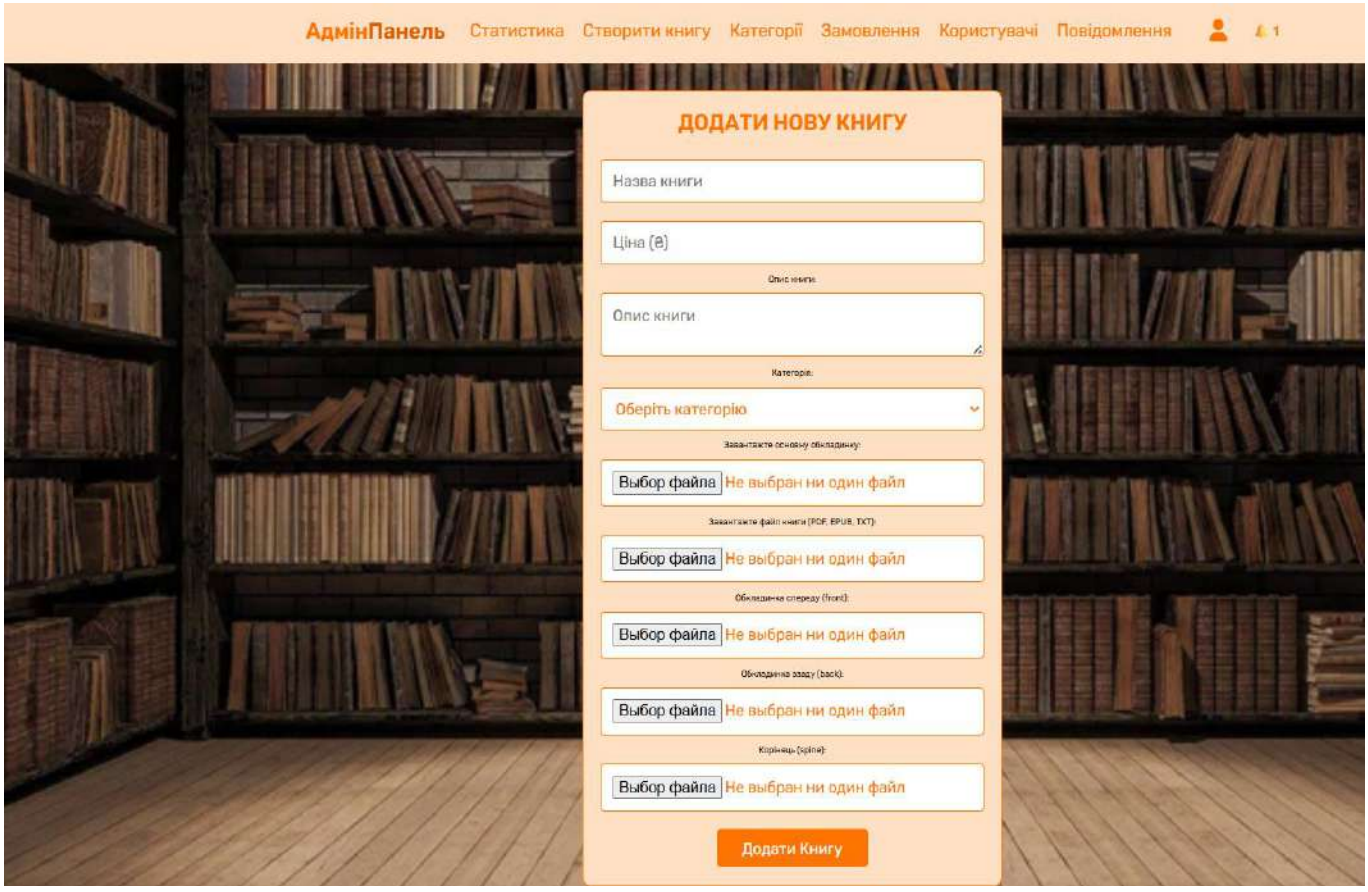

Рис. 4.16. Сторінка створення книг

У розділі «Категорії» (bookstore/admin/categories) можна додати новий жанр, просто вказавши його назву. Цей жанр відразу ж відображається у фільтрі каталогу на сайті (Рис. 4.17).

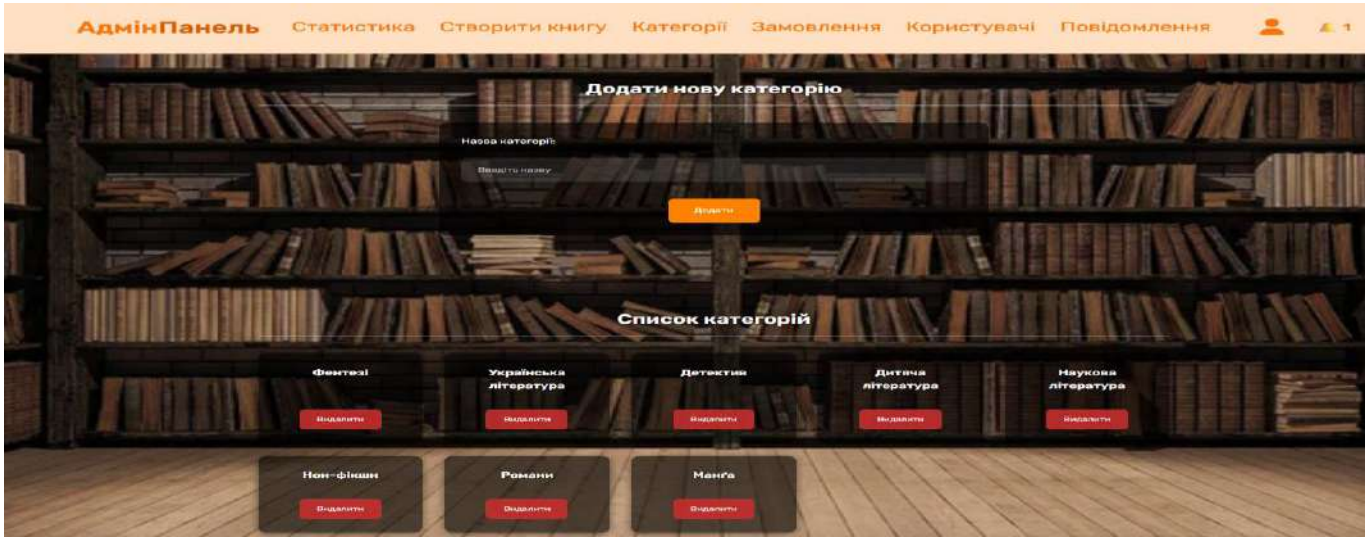

Рис. 4.17. Сторінка створення категорій

На сторінці «Користувачі» (bookstore/admin/users) відображаються всі зареєстровані акаунти, і адміністратори, і звичайні користувачі. Адміністратор може переглянути дані будь–якого користувача та видалити обліковий запис у разі потреби (Рис. 4.18).

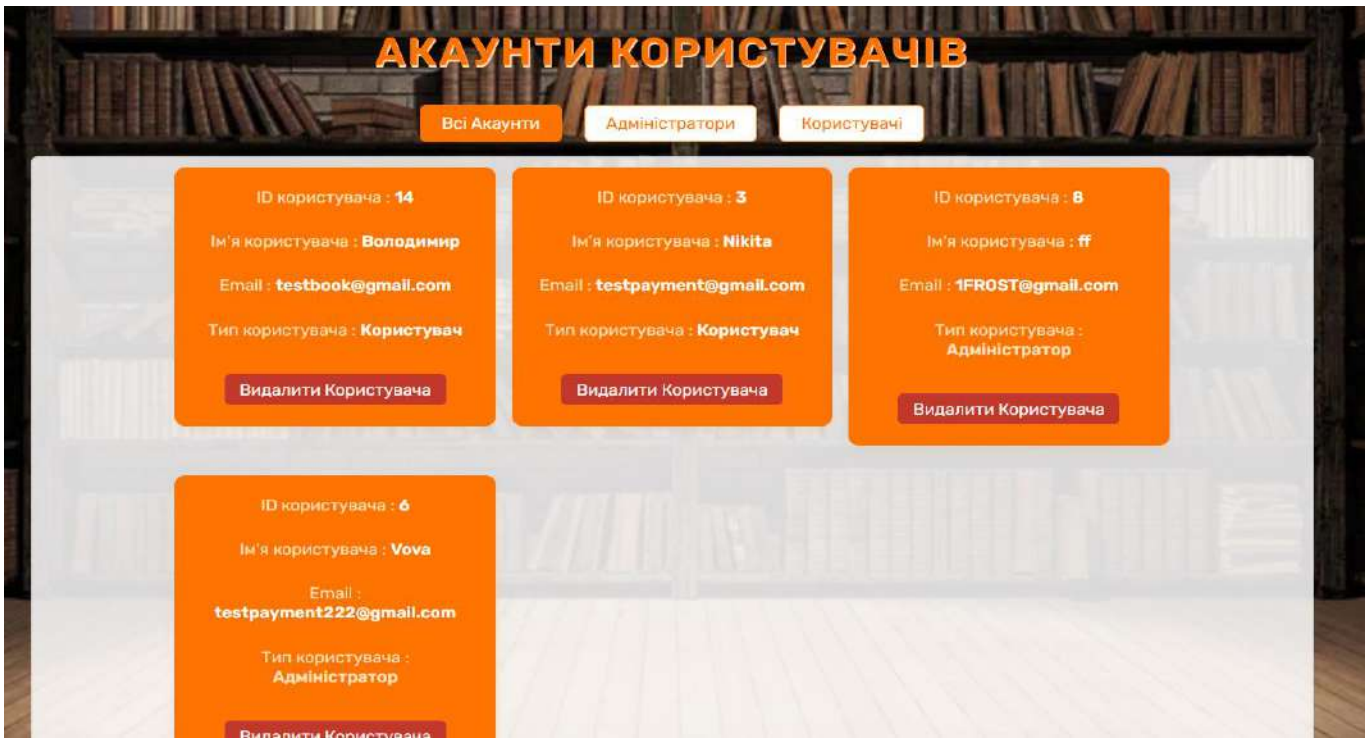

Рис. 4.18. Сторінка перегляду користувачів

Якщо користувач залишає звернення через форму «Контакти» (bookstore/contact), воно з’являється в адмінці на сторінці «Повідомлення» (bookstore/admin/contacts). Тут зберігаються ім’я автора, email, телефон та текст повідомлення. Адміністратор може переглянути звернення або видалити його після обробки (Рис. 4.19).

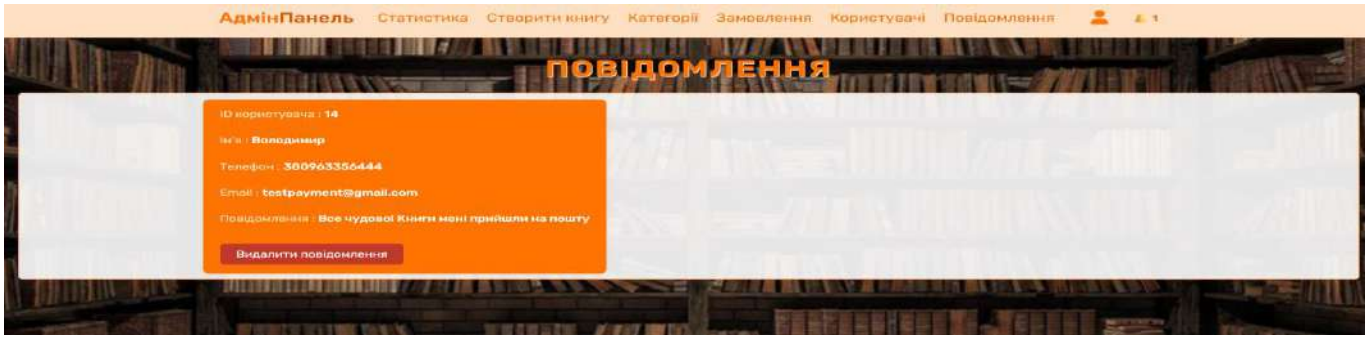

Рис. 4.19. Сторінка повідомлень

Головна сторінка «Замовлення» (bookstore/admin/orders) містить список усіх оформлених замовлень з детальною інформацією: номер замовлення, ідентифікатор клієнта, час замовлення, ім’я, телефон та email отримувача, адреса доставки, перелік замовлених книг, загальна сума та спосіб оплати. Якщо статус оплати вказаний як «Очікує оплати», доступ до електронної версії книги буде заблокований. Після підтвердження оплати адміністратор змінює статус замовлення на «Оплачено», і тоді клієнт отримує можливість почати читати книгу онлайн (Рис. 4.20).

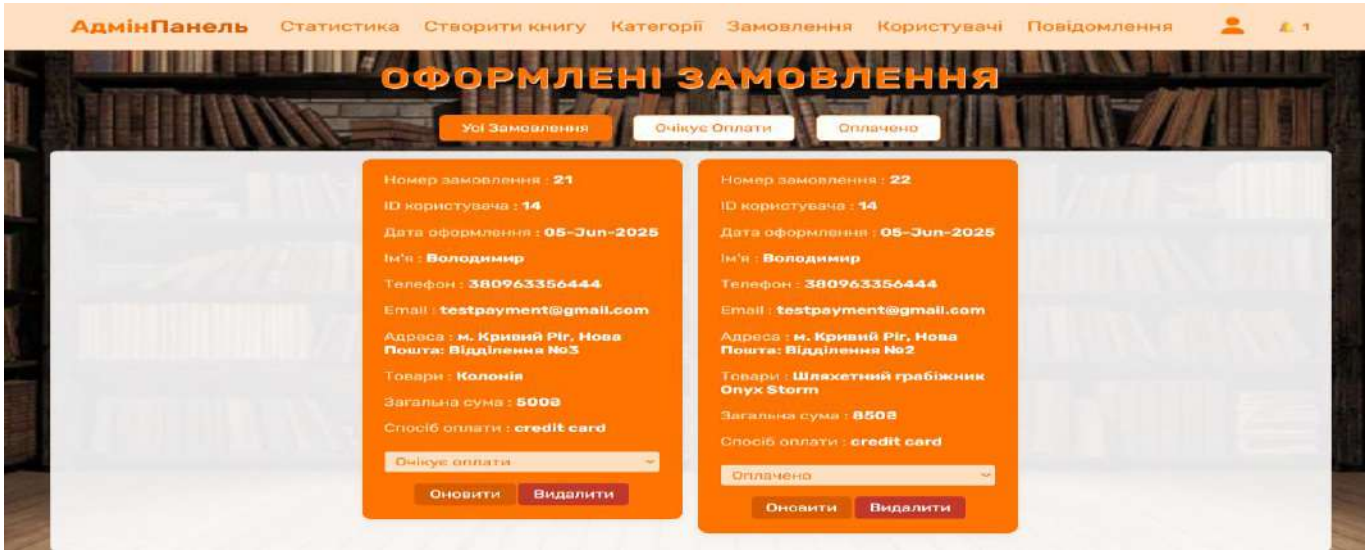

Рис. 4.20. Сторінка замовлень

Як уже згадувалося раніше, якщо користувач здійснив замовлення, адміністратор повинен перевіряти ті, що перебувають в обробці. Для спрощення цього є сповіщення про замовлення: біля хедера розміщена іконка дзвінка, яка при наявності необроблених замовлень відображає їх кількість. Натиснувши на цю іконку, ми побачимо коротку інформацію: номер замовлення, ім’я замовника, суму до сплати та дату оформлення (Рис. 4.21). Натиснувши на кнопку «Детальніше», ми потрапляємо на сторінку замовлень. Якщо оплата пройшла, адміністратор може одразу змінити статус замовлення.

Рис. 4.21. Повідомлення про замовлення

Таким чином, адміністративна частина забезпечує повний цикл керування: від перегляду статистики й обробки замовлень до управління каталогом і користувачами. Інтуїтивно зрозумілі інструменти дають змогу оперативно реагувати на нові замовлення, своєчасно додавати нові книги та контролювати якість обслуговування клієнтів.

4.4. Десктопний додаток на C#

Десктопний додаток реалізовано у вигляді Windows Forms–проєкту під .NET 8 із використанням OpenTK для рендерингу тривимірних моделей книг. Головна мета цього модуля - надати користувачеві можливість переглянути

обкладинку з переднього й заднього боку та корінець книги в інтерактивному режимі, отримуючи дані безпосередньо з тієї самої бази, що й веб-сайт.

При запуску програми відкривається стандартне вікно Windows Forms, яке розділене на дві основні ділянки (Рис. 4.22):

- Лівая панель - ListBox - перелік доступних книг (назви). Список автоматично наповнюється при запиті до бази й відображає всі наявні позиції.
- Правая панель-GLControl - область для рендерингу 3D-моделі. Вона займає приблизно 75 % ширини вікна.

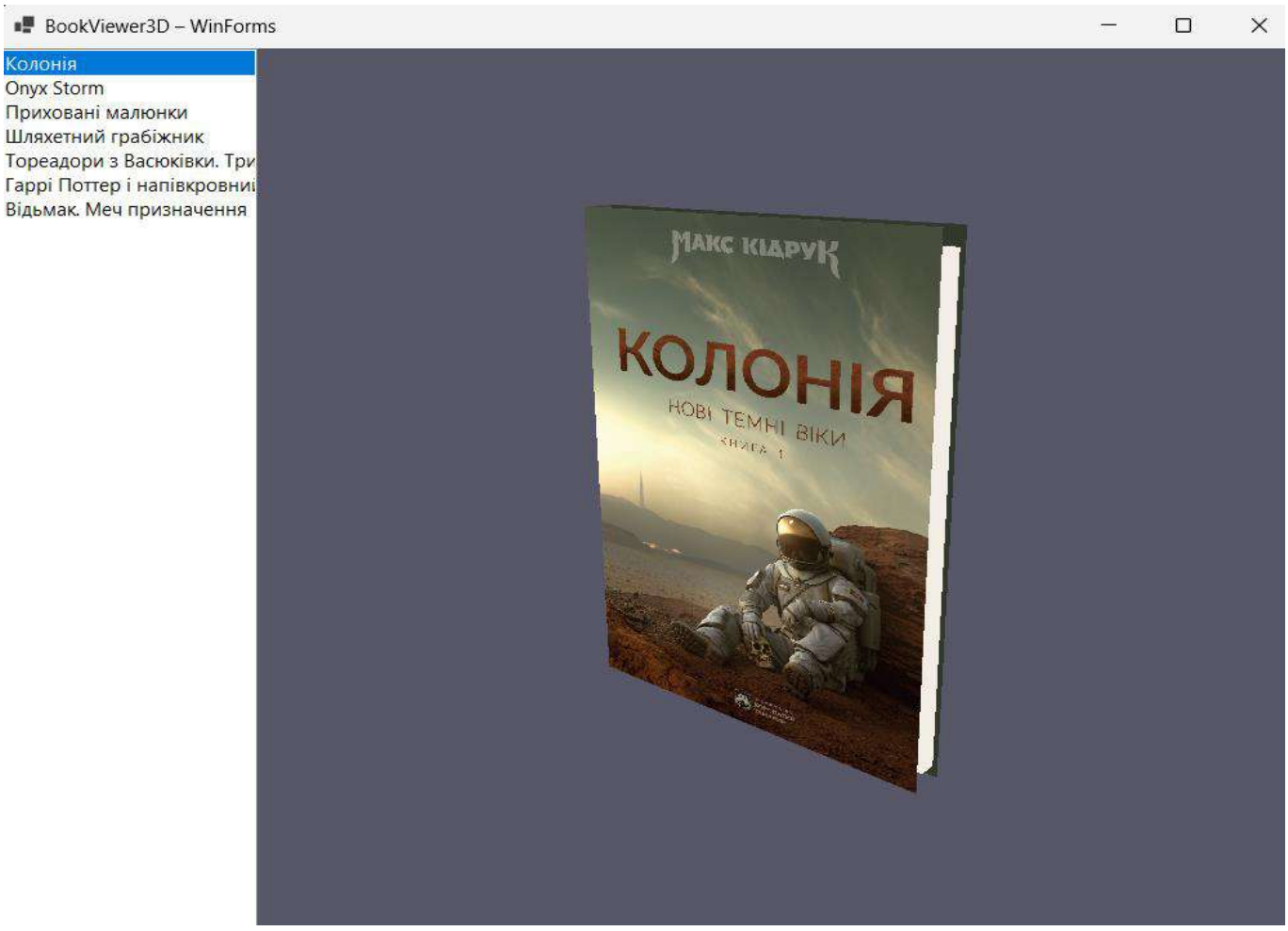

Рис. 4.22. Список книг та 3D модель обраної книги

Клас BookRepository відповідає за підключення до MySQL (через MySqlConnector) з тими самими параметрами, що й веб-додаток. У методі

GetAll () виконується SQL-запит: SELECT id, name, cover_front, cover_back, spine FROM books";

Клас TextureLoader забезпечує читання PNG/JPEG-файлів обкладинок із папки BookStore\uploaded_img/. Якщо файл відсутній або пошкоджений, завантажується білий квадрат (Рис. 4.23).

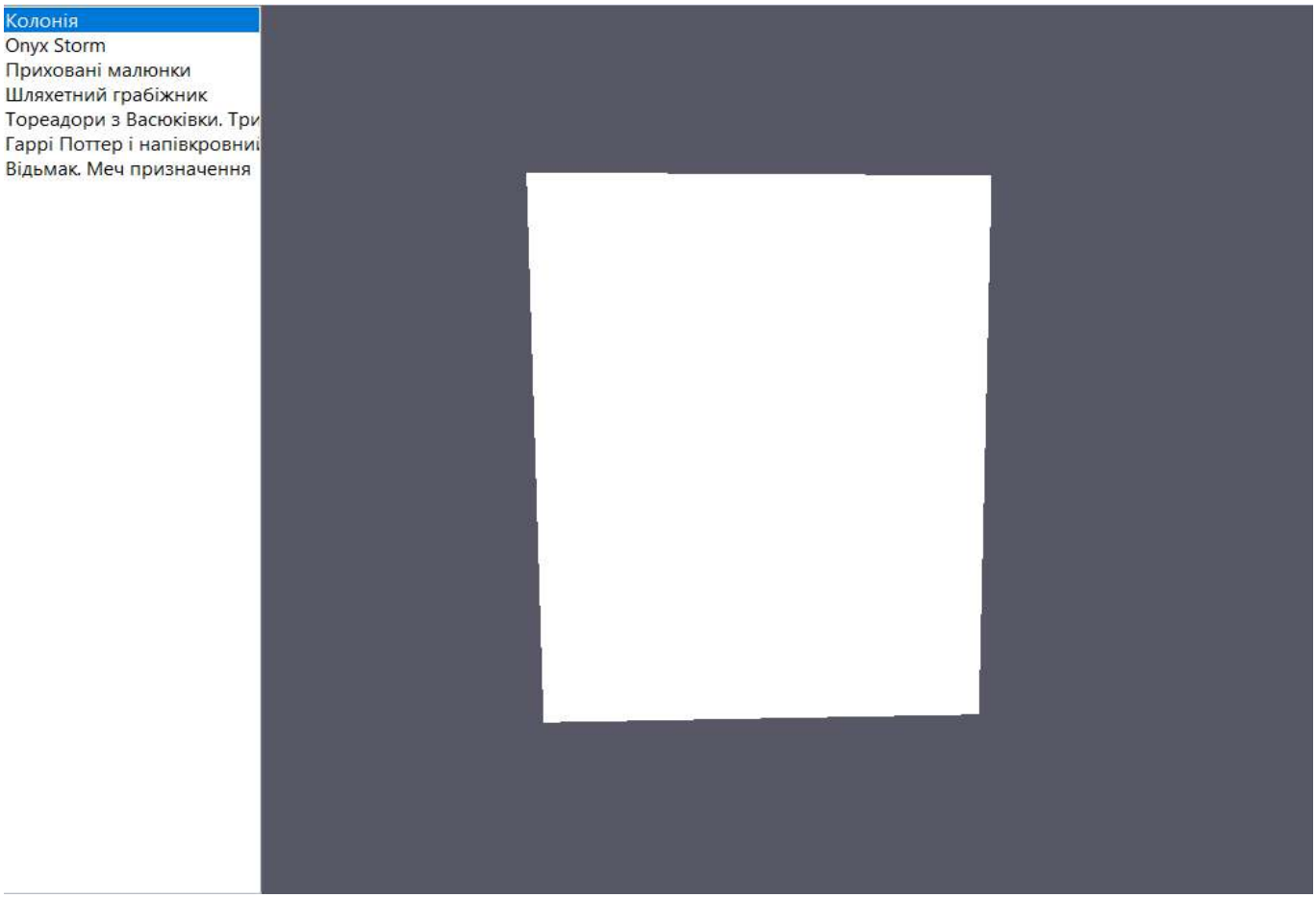

Рис. 4.23. 3D модель книги в разі відсутності зображення

Після відкриття через Bitmap відбувається формування двовимірної текстури OpenGL (формат RGBA) із автоматичним масштабуванням до максимальної роздільної здатності 2048×2048.

Клас BookRenderer ініціалізує інтерпретацію текстур як шести граней моделі:

- Передня обкладинка (front) і задня (back) - дві площини з текстурами.
- Корінець (spine) - окрема невелика площина між front і back.
- Блок сторінок - простий одноколірний прямокутник між обкладинками.

Обертання моделі: натискання та перетягування миші змінює кути повороту навколо осей X і Y. У події MouseMove відстежується зміщення курсора, а в MouseDown/MouseUp - фіксується стан “drag”.

Архітектура файлів:

- Program.cs - налаштування хосту WinForms, виклик Application.Run(new Form1()).
- Form1.cs - основна форма з ініціалізацією компонентів: створення GLControl і ListBox. У конструкторі відбувається виклик BookRepository.GetAll() та заповнення ListBox.
- BookRepository.cs - містить метод для роботи з базою: відкриття з’єднання, запит до таблиці books, повернення списку моделей book.
- Book.cs - запис з полями Id, Title, CoverFront, CoverBack, Spine.
- TextureLoader.cs - статичний клас із методом Load.
- BookRenderer.cs - інкапсулює код, пов’язаний із малюванням.

Десктопний додаток на C# із WinForms та OpenTK доповнює веб-магазин, надаючи користувачу наочну 3D-візуалізацію обкладинок книг та легкий доступ до електронних версій. Використання знайомих технологій (.NET 8, MySqlConnector, OpenGL) забезпечує стабільну роботу, швидке завантаження даних і простоту супроводу.

Висновки до розділу 4

У четвертому розділі було детально описано користувацький інтерфейс веб-сайту, адміністративну панель та десктопний 3D-застосунок. Інтерфейс сайту побудовано так, щоб будь-який користувач міг швидко зареєструватися, знайти потрібну книгу, відфільтрувати каталог і оформити замовлення. Форма «Кошик - Оформлення - Замовлення» забезпечує правильну послідовність дій.

Адміністративна частина реалізована окремими модулями: «Дашборд» для загальної статистики, «Книги» і «Категорії» для керування асортиментом,

«Замовлення» з сповіщеннями та «Користувачі» й «Повідомлення» для модерації й підтримки клієнтів.

Десктопний додаток на C# із WinForms та OpenTK доповнює веб, дозволяючи в інтерактивному режимі переглядати 3D-моделі обкладинок і корінців книг. Завантаження текстур напряму з тієї самої MySQL-бази, що й сайт, гарантує актуальність даних. Простий і зрозумілий інтерфейс зліва (ListBox) і велика область рендерингу справа дають змогу швидко орієнтуватися й взаємодіяти з моделями.

Загалом, поєднання веб-інтерфейсу, адмін-панелі й десктопного модуля створює цілісний комплекс, який покриває всі етапи взаємодії: від реєстрації до візуалізації продукту у 3D. Вибрані технології забезпечують необхідну гнучкість, продуктивність і можливість подальшого розвитку функціоналу.

ВИСНОВКИ

В результаті виконання даної кваліфікаційної роботи було досягнуто поставленої мети - створено програмне забезпечення для продажу книг, яке об’єднує веб-магазин та десктопний 3D-переглядач. Проєкт охоплює розробку клієнтської частини у вигляді веб-сайту з каталогом, кошиком і адміністративною панеллю, а також створення Windows Forms-застосунку для наочного перегляду обкладинок книг.

В результаті проведеної роботи можна зробити такі висновки:

1. На веб-сайті реалізовано весь необхідний цикл взаємодії, реєстрація та авторизація користувачів, каталог із фільтрацією за жанрами та ціною, детальна картка книги з описом і кнопками «Додати в кошик» і «Читати онлайн», кошик із можливістю змінювати кількість товарів та оформлювати замовлення. Після підтвердження замовлення клієнт бачить історію покупок зі статусами і, у разі оплати, отримує доступ до електронної версії книги.

2. Адміністративний інтерфейс дозволяє модератору переглядати всю статистику - кількість неоплачених і оплачених замовлень, загальну кількість книг у каталозі, кількість користувачів і нових повідомлень. У панелі адміністратора реалізовано можливість додавати і видаляти книги та категорії, керувати замовленнями (переглядати їх деталі, змінювати статус оплати).

3. Десктопний додаток забезпечує інтерактивний перегляд обкладинок у форматі 3D. Користувач може обертати модель книги й перемикатися між різними виданнями, ознайомлюючись із передньою й задньою обкладинками та корінцем. Завантаження даних напряму з тієї ж бази, що й на сайті, гарантує актуальність інформації.

Таким чином, виконана робота забезпечила злагоджену взаємодію веб-інструментів і десктопного компонента, створивши цілісний сервіс для продажу та перегляду книг. Користувачі отримали простий інтерфейс для покупки й читання, а адміністратори - ефективний набір інструментів для управління контентом і обробки замовлень.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Коноваленко І. В., Марущак П. О. Платформа .NET та мова програмування C# 8.0 (навчальний посібник). 2020, ФОП Паляниця В. А.
2. Балик Н. Р., Мандзюк В. І. Бази даних MySQL (навчальний посібник). 2010, Навчальна книга – Богдан, 2010.— 160 с.
3. Шилдт Г. C# 10 і .NET 6: повний довідник для професіоналів / Г. Шилдт. — Київ : Діалектика, 2022. — 880 с.
4. Troelsen A., Japikse P. Pro C# 10 with .NET 6 / Andrew Troelsen, Philip Japikse. — Apress, 2022. — 1500 с.
5. Шевченко В. Графічне програмування засобами OpenGL. Навчальний посібник. — Київ : НТУУ «КПІ», 2020. — 124 с.
6. Хом’як В. Основи побудови графічних інтерфейсів користувача на C#. — Львів : Видавництво ЛНУ, 2019. — 102 с.
7. McKesson D. OpenGL Insights / D. McKesson. — CRC Press, 2012. — 712с.
8. Бабич А., Кирилюк Т. Алгоритми та структури даних у графіці. — Харків : ХНУРЕ, 2021. — 168с.
9. Dave Shreiner, Graham Sellers, John Kessenich – OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.5 with SPIR-V. — Pearson Education (Addison-Wesley), 2016. — 976 с.
10. Graham Sellers, Richard S. Wright Jr., Nicholas Haemel – OpenGL SuperBible: Comprehensive Tutorial and Reference (7-th ed.). — Addison-Wesley Professional, 2016. — 880 с.
11. М. Ф. Пічугін, І. О. Канкін, В. В. Воротніков – Комп’ютерна графіка: навчальний посібник. — Центр учбової літератури, 2013. — 346 с.
12. М. І. Безменов, О. М. Безменова, Д. В. Калінін – Основи візуального програмування мовою C#. — ФОП Панов А. М. (НТУ «ХПІ»), 2023. — 648 с.
13. Надія Балик, Віктор Мандзюк – Бази даних MySQL: навчальний посібник. 2010. авчальний посібник. — Навчальна книга — Богдан, 2010. — 160 с

14. Eric A. Meyer, Estelle Weyl – CSS: The Definitive Guide (4th Edition). 2017. O'Reilly Media. 2017. -1088 с.
15. Daniel Nichter – Efficient MySQL Performance: Best Practices and Techniques. — O'Reilly Media, 2022. — 335 с.
16. Jon Duckett – HTML and CSS: Design and Build Websites. — John Wiley & Sons, 2011. — 512 с.
17. Dan Ginsburg, Budirijanto Purnomo – OpenGL ES 3.0 Programming Guide (2-nd ed.). — Addison-Wesley Professional, 2014. — 560 с.
18. Jennifer Niederst Robbins – Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics (5-th ed.). — O'Reilly Media, 2018. — 808 с.
19. Девід Вітні – Програмування для дітей. HTML, CSS та JavaScript. — Віват, 2019. — 208 с.
20. Paul DuBois – MySQL book (3-rd ed.). — O'Reilly Media, 2014. — 866 с.
21. Rick Silva – MySQL Crash Course: A Hands-On Introduction to Database Development. — No Starch Press, 2023. — 352 с.
22. Сидоренко В., Константинова Л., Смірнов С. — Організація баз даних: навчальний посібник. — Кропивницький : ЦНТУ, 2018. — 274 с.
23. Гайдаржи В. І., Дацюк О. А. — Основи проектування та використання баз даних: навчальний посібник. — Київ : ІВЦ «Видавництво Політехніка», 2004. — 256 с.
24. Мулеса О. Ю., Варга Я. В. — Інформаційні системи та реляційні бази даних: навчальний посібник. — Ужгород : ДВНЗ «УжНУ», 2018. — 118 с
25. Бугаєва І. Г., Новікова Н. О., Панченко Т. Д. — Основи PHP: навчальний посібник. — Одеса : ОНМУ, 2018. — 83 с.

ДОДАТКИ

Додаток А

Рути .htacess

```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^product/([0-9]+)$ product.php?id=$1 [L,QSA]
RewriteRule ^shop$ shop.php [L,QSA]
RewriteRule ^cart$ cart.php [L,QSA]
RewriteRule ^orders$ orders.php [L,QSA]
RewriteRule ^checkout$ checkout.php [L,QSA]
RewriteRule ^contact$ contact.php [L,QSA]
RewriteRule ^about$ about.php [L,QSA]
RewriteRule ^search$ search_page.php [L,QSA]
RewriteRule ^register$ register.php [L,QSA]
RewriteRule ^login$ login.php [L,QSA]
RewriteRule ^admin/?$ admin_page.php [L,QSA]
RewriteRule ^admin/orders$ admin_orders.php [L,QSA]
RewriteRule ^admin/products$ admin_products.php [L,QSA]
RewriteRule ^admin/categories$ admin_categories.php [L,QSA]
RewriteRule ^admin/users$ admin_users.php [L,QSA]
RewriteRule ^admin/contacts$ admin_contacts.php [L,QSA]
RewriteRule ^home$ home.php [L,QSA]
RewriteRule ^$ home.php [L,QSA]
```

Додаток Б

Сторінка shop.php

```
<?php

include 'config.php';
session_start();

$user_id = $_SESSION['user_id'] ?? null;
if (!$user_id) {
    header('location:login.php');
    exit();
}

$messages = [];

$categories_list = [];
$cat_query = mysqli_query($conn, "SELECT * FROM `categories` ORDER BY
name ASC");
while ($cat = mysqli_fetch_assoc($cat_query)) {
    $categories_list[] = $cat;
}

$filter_category = $_GET['category'] ?? '';
$filter_price_min = isset($_GET['price_min']) ?
floatval($_GET['price_min']) : '';
$filter_price_max = isset($_GET['price_max']) ?
floatval($_GET['price_max']) : '';

if (isset($_POST['add_to_cart'])) {
    $prod_id       = intval($_POST['product_id']);
    $cart_quantity = intval($_POST['product_quantity']);
    $product_query = mysqli_query($conn, "SELECT * FROM `books` WHERE
id = '$prod_id'") or die('query failed');
    if (mysqli_num_rows($product_query) > 0) {
        $product = mysqli_fetch_assoc($product_query);
        $name    = mysqli_real_escape_string($conn, $product['name']);
        $price   = $product['price'];
        $image   = mysqli_real_escape_string($conn,
$product['image']);

        $check_cart = mysqli_query($conn, "SELECT * FROM `cart` WHERE
user_id = '$user_id' AND name = '$name'") or die('query failed');
        if (mysqli_num_rows($check_cart) > 0) {
            $update_cart = mysqli_query($conn, "UPDATE `cart` SET
quantity = quantity + $cart_quantity WHERE user_id = '$user_id' AND
name = '$name'") or die('query failed');
            if ($update_cart) {
                $messages[] = 'Кількість оновлено у кошику!';
            } else {
                $messages[] = 'Помилка при оновленні кількості!';
            }
        } else {
```

```
            mysqli_query($conn, "INSERT INTO `cart`(user_id, name,
price, quantity, image) VALUES('$user_id', '$name', '$price',
'$cart_quantity', '$image')") or die('query failed');
            $messages[] = 'Книгу додано до кошика!';
        }
    } else {
        $messages[] = 'Товар не знайдено!';
    }
}
$where = [];
if ($filter_category !== '' && $filter_category !== 'all') {
    $cat_id = intval($filter_category);
    $where[] = "p.category_id = '$cat_id'";
}
if ($filter_price_min !== '') {
    $where[] = "p.price >= '$filter_price_min'";
}
if ($filter_price_max !== '') {
    $where[] = "p.price <= '$filter_price_max'";
}
$where_sql = $where ? 'WHERE ' . implode(' AND ', $where) : '';
$min_price_q = mysqli_query($conn, "SELECT MIN(price) as minp,
MAX(price) as maxp FROM books");
$minmax = mysqli_fetch_assoc($min_price_q);
$min_price = floor($minmax['minp'] ?? 0);
$max_price = ceil($minmax['maxp'] ?? 1000);
$filter_price_min = isset($_GET['price_min']) ?
floatval($_GET['price_min']) : $min_price;
$filter_price_max = isset($_GET['price_max']) ?
floatval($_GET['price_max']) : $max_price;
$books_per_page = 12;
$page = isset($_GET['page']) ? max(1, intval($_GET['page'])) : 1;
$offset = ($page - 1) * $books_per_page;
$count_sql = "SELECT COUNT(*) as cnt FROM `books` p LEFT JOIN
`categories` c ON p.category_id = c.id $where_sql";
$count_result = mysqli_query($conn, $count_sql);
$total_books = mysqli_fetch_assoc($count_result)['cnt'];
$total_pages = max(1, ceil($total_books / $books_per_page));
$sort_order = ($_GET['sort'] ?? 'desc') === 'asc' ? 'ASC' : 'DESC';
$select_products = mysqli_query(
    $conn,
    "SELECT p.*, c.name as category_name FROM `books` p LEFT JOIN
`categories` c ON p.category_id = c.id $where_sql ORDER BY p.price
$sort_order, p.id DESC LIMIT $books_per_page OFFSET $offset"
) or die('query failed');
$sort_order = ($_GET['sort'] ?? 'desc') === 'asc' ? 'ASC' : 'DESC';
?>

<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
```

Продовження додатку Б

```
<title>Магазин книг</title>
<link rel="stylesheet" href="/css/style.css">

<style>
.filter-form {
    background: #f3f0e8;
    border-radius: .7rem;
    padding: 1.5rem 2rem;
    margin: 2rem auto 2.5rem auto;
    max-width: 900px;
    display: flex;
    flex-wrap: wrap;
    gap: 2rem;
    align-items: flex-end;
    box-shadow: 0 2px 10px #e0e0e0;
}
.filter-form label {
    font-weight: 500;
    margin-bottom: .5rem;
    display: block;
    color: #7a5c1d;
}
.filter-form select,
.filter-form input[type="number"] {
    padding: .7rem 1rem;
    border-radius: .4rem;
    border: 1px solid #ccc;
    font-size: 1.1rem;
    min-width: 120px;
    background: #fff;
}
.filter-form .btn {
    padding: .8rem 2rem;
    font-size: 1.1rem;
    border-radius: .4rem;
    background: var(--orange);
    color: #fff;
    border: none;
    cursor: pointer;
    font-weight: bold;
    transition: background 0.2s;
}
.filter-form .btn:hover {
    background: var(--red);
}
.filter-form .slider-labels {
    display: flex;
    justify-content: space-between;
    font-size: 1.1rem;
    margin-top: .3rem;
    color: #7a5c1d;
}
.filter-form .slider-wrap {
```

Продовження додатку Б

```
    min-width: 220px;
    flex: 1 1 220px;
}
input[type="range"] {
    width: 100%;
}

.products .box-container {
   display: grid;
   grid-template-columns: repeat(4, 1fr);
   gap: 2rem;
   padding: 2rem;
}
.products .box {
   position: relative;
   border: var(--border);
   box-shadow: var(--box-shadow);
   border-radius: .5rem;
   background: #fff;
   overflow: hidden;
   transition: transform 0.2s ease-in-out;
   display: flex;
   flex-direction: column;
   height: 700px;
}
.products .box:hover {
   transform: scale(1.03);
}
.products .box .image {
   height: 300px;
   display: flex;
   align-items: center;
   justify-content: center;
   background: #f3f0e8;
   overflow: hidden;
}
.products .box .image img {
   width: 100%;
   height: 100%;
   object-fit: contain;
   max-height: 300px;
   max-width: 100%;
   background: #f3f0e8;
   border-radius: .5rem;
   display: block;
   margin: 0 auto;
}
.products .box .price-label {
   position: absolute;
   top: 1rem;
   left: 1rem;
   background: var(--red);
   color: #fff;
   padding: .5rem 1rem;
   border-radius: .5rem;
```

```
   font-size: 1.6rem;
   font-weight: bold;
}
.products .box .content {
   padding: 1.5rem;
   text-align: center;
   display: flex;
   flex-direction: column;
   justify-content: space-between;
   flex: 1;
}
.products .box .category {
   background: #f3f0e8;
   color: #7a5c1d;
   font-weight: bold;
   padding: .6rem 1rem;
   border-radius: .4rem;
   margin: 1rem 0 .5rem 0;
   display: inline-block;
   font-size: 1.3rem;
   letter-spacing: .5px;
}
.products .box .description {
   background: #f9f9f9;
   border-left: 4px solid #b9b18c;
   color: #444;
   padding: 1rem 1.2rem;
   border-radius: .4rem;
   margin-bottom: 1rem;
   font-size: 1.25rem;

   min-height: 100px;
   text-align: left;
   overflow: hidden;
}
.products .box .actions {
   display: flex;
   justify-content: center;
   gap: 1rem;
   flex-wrap: wrap;
   margin-top: 1rem;
}
.products .box .actions .qty {
   width: 6rem;
}
.products .box .actions .btn,
.products .box .actions .option-btn {
   padding: 1rem 2rem;
   font-size: 1.6rem;
}

@media (max-width: 1024px) {
   .products .box-container {
      grid-template-columns: repeat(2, 1fr);
```

```
        }
    }
    @media (max-width: 600px) {
        .products .box-container {
            grid-template-columns: 1fr;
        }
        .products .box {
            height: auto;
        }
        .products .box .content {
            padding: 1rem;
        }
    }
    </style>
</head>
<body>

<?php include 'header.php'; ?>

<?php if (!empty($messages)): ?>
    <?php foreach ($messages as $msg): ?>
        <div class="message">
            <span><?php echo htmlspecialchars($msg); ?></span>
            <i class="fas fa-times"
onclick="this.parentElement.style.display='none';"></i>
        </div>
    <?php endforeach; ?>
<?php endif; ?>

<form method="get" class="filter-form" id="filterForm">
    <div>
        <label for="category">Категорія:</label>
        <select name="category" id="category">
            <option value="all">Всі категорії</option>
            <?php foreach ($categories_list as $cat): ?>
                <option value="<?php echo $cat['id']; ?>" <?php if
($filter_category == $cat['id']) echo 'selected'; ?>>
                    <?php echo htmlspecialchars($cat['name']); ?>
                </option>
            <?php endforeach; ?>
        </select>
    </div>
    <div class="slider-wrap">
        <label>Ціна: <span id="priceRangeLabel"><?php echo
$filter_price_min; ?>₴ – <?php echo $filter_price_max;
?>₴</span></label>
        <div style="display:flex;gap:1rem;align-items:center;">
            <input type="range" name="price_min" id="price_min"
min="<?php echo $min_price; ?>" max="<?php echo $max_price; ?>"
value="<?php echo $filter_price_min; ?>" step="1"
oninput="updatePriceLabel()">
            <input type="range" name="price_max" id="price_max"
min="<?php echo $min_price; ?>" max="<?php echo $max_price; ?>"
```

```
value="<?php echo $filter_price_max; ?>" step="1"
oninput="updatePriceLabel()">
        </div>
        <div class="slider-labels">
            <span><?php echo $min_price; ?>₴</span>
            <span><?php echo $max_price; ?>₴</span>
        </div>
    </div>
    <div>
        <label for="sort">Сортувати за ціною:</label>
        <select name="sort" id="sort">
            <option value="desc" <?php if(($_GET['sort'] ?? '') ==
'desc') echo 'selected'; ?>>Від дорогих до дешевих</option>
            <option value="asc" <?php if(($_GET['sort'] ?? '') ==
'asc') echo 'selected'; ?>>Від дешевих до дорогих</option>
        </select>
    </div>
    <div>
        <button type="submit" class="btn">Застосувати фільтр</button>
        <a href="/shop.php" class="btn" style="background:var(--
white);color:var(--black);margin-left:1rem;">Скинути</a>
    </div>
</form>

<script>
function updatePriceLabel() {
    let min = document.getElementById('price_min').value;
    let max = document.getElementById('price_max').value;
    if (parseInt(min) > parseInt(max)) {

        document.getElementById('price_min').value = max;
        min = max;
    }
    document.getElementById('priceRangeLabel').textContent = min + '₴
- ' + max + '₴';
}
document.addEventListener('DOMContentLoaded', updatePriceLabel);
</script>

<section class="products">
   <h1 class="title">Всі книги</h1>

   <div class="box-container">
      <?php
      if (mysqli_num_rows($select_products) > 0):
         while ($fetch_products =
mysqli_fetch_assoc($select_products)):
            $product_id = $fetch_products['id'];
            $can_read   = false;

            $check_purchase = mysqli_query($conn, "SELECT * FROM
`purchases` WHERE user_id = '$user_id' AND book_id = '$product_id'");
            if (mysqli_num_rows($check_purchase) > 0) {
                $can_read = true;
```

Продовження додатку Б

```
            }
      ?>
<form action="" method="post" class="box">
   <div class="image">
      <a href="product/<?php echo $product_id; ?>">
         <img src="uploaded_img/<?php echo
htmlspecialchars($fetch_products['image']); ?>" alt="cover">
      </a>
      <div class="price-label"><?php echo $fetch_products['price'];
?>₴</div>
   </div>
   <div class="content">
      <div>
         <h3 class="name"><?php echo
htmlspecialchars($fetch_products['name']); ?></h3>
         <div class="category">
            <strong>Категорія:</strong>
            <?php echo
htmlspecialchars($fetch_products['category_name'] ?? 'Без категорії');
?>
         </div>
         <div class="description">
            <em>
               <?php
                  $desc =
htmlspecialchars($fetch_products['description']);
                  if (mb_strlen($desc, 'UTF-8') > 200) {
                      echo mb_substr($desc, 0, 200, 'UTF-8') . '...';
                  } else {
                      echo $desc;
                  }
               ?>
            </em>
         </div>
      </div>
      <div class="actions">
         <input type="hidden" name="product_id" value="<?php echo
$product_id; ?>">
         <input type="number" name="product_quantity" class="qty"
min="1" value="1">
         <input type="submit" name="add_to_cart" class="btn"
value="Додати до кошика">
         <?php if ($can_read): ?>
            <a href="read_book.php?id=<?php echo $product_id; ?>"
class="option-btn">Читати</a>
         <?php endif; ?>
      </div>
   </div>
</form>
      <?php
         endwhile;
      else:
         echo '<p class="empty">Немає доступних книг</p>';
      endif;
```

```
      ?>
   </div>

   <?php if ($total_pages > 1): ?>
   <div class="pagination" style="display:flex;justify-
content:center;gap:.5rem;margin:2rem 0;">
      <?php
      $query_params = $_GET;
      for ($i = 1; $i <= $total_pages; $i++):
         $query_params['page'] = $i;
         $link = '?' . http_build_query($query_params);
      ?>
         <a href="<?php echo $link; ?>"
            class="btn"
            style="background:<?php echo $i == $page ? 'var(--orange)'
: 'var(--white)'; ?>;color:<?php echo $i == $page ? '#fff' : 'var(--
black)'; ?>;">
            <?php echo $i; ?>
         </a>
      <?php endfor; ?>
   </div>
   <?php endif; ?>
</section>

<?php include 'footer.php'; ?>

<script src="/js/script.js"></script>
<script>
document.addEventListener('DOMContentLoaded', function() {
    document.querySelectorAll('.message').forEach(function(msg) {
        setTimeout(function() {
            msg.style.opacity = '0';
            setTimeout(function() { msg.remove(); }, 300);
        }, 3000);
    });
});
</script>

</body>
</html>
```

Додаток В

Сторінка admin_orders.php

```
<?php

include 'config.php';
session_start();

$admin_id = $_SESSION['admin_id'];
if (!isset($admin_id)) {
    header('location:login.php');
    exit;
}

// 1) Обработка обновления статуса оплаты (как было)
if (isset($_POST['update_order'])) {
    $order_update_id = $_POST['order_id'];
    $update_payment = $_POST['update_payment'];
    mysqli_query($conn, "UPDATE `orders` SET payment_status =
'$update_payment' WHERE id = '$order_update_id'")
        or die('query failed');
    header('Location: /admin/orders' . (isset($_GET['filter']) ?
'?filter=' . $_GET['filter'] : ''));
    exit;
}

// 2) Обработка удаления заказа (как было)
if (isset($_GET['delete'])) {
    $delete_id = $_GET['delete'];
    mysqli_query($conn, "DELETE FROM `orders` WHERE id =
'$delete_id'") or die('query failed');
    header('Location: /admin/orders' . (isset($_GET['filter']) ?
'?filter=' . $_GET['filter'] : ''));
    exit;
}

// 3) Считываем текущий фильтр из GET-параметра
//    filter может быть 'all', 'pending' или 'completed'
$filter = 'all';
if (isset($_GET['filter'])) {
    $f = $_GET['filter'];
    if ($f === 'pending' || $f === 'completed') {
        $filter = $f;
    }
}

// 4) Формируем SQL-запрос в зависимости от фильтра
if ($filter === 'pending') {
    $sql = "SELECT * FROM `orders` WHERE payment_status = 'pending'
ORDER BY placed_on DESC";
} elseif ($filter === 'completed') {
    $sql = "SELECT * FROM `orders` WHERE payment_status = 'completed'
ORDER BY placed_on DESC";
} else {
    // 'all'
```

```
    $sql = "SELECT * FROM `orders` ORDER BY placed_on DESC";
}

$select_orders = mysqli_query($conn, $sql) or die('query failed');

?>

<!DOCTYPE html>
<html lang="uk">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-
scale=1.0">
   <title>Адмін - Замовлення</title>

   <link rel="stylesheet" href="/css/admin_style.css">
   <style>
html {
  scroll-behavior: smooth;
}
</style>
</head>
<body>

<?php include 'admin_header.php'; ?>

<section class="orders">

   <h1 class="title">Оформлені замовлення</h1>
   <div class="filter-buttons">
      <a href="/admin/orders?filter=all" class="btn <?php echo
($filter === 'all' ? 'active' : ''); ?>">
         Усі замовлення
      </a>
      <a href="/admin/orders?filter=pending" class="btn <?php echo
($filter === 'pending' ? 'active' : ''); ?>">
         Очікує оплати
      </a>
      <a href="/admin/orders?filter=completed" class="btn <?php echo
($filter === 'completed' ? 'active' : ''); ?>">
         Оплачено
      </a>
   </div>
   <div class="box-container scrollable">
      <?php if (mysqli_num_rows($select_orders) > 0): ?>
         <?php while ($fetch_orders =
mysqli_fetch_assoc($select_orders)): ?>
            <div class="box">
               <p> Номер замовлення :
                  <span><?php echo $fetch_orders['id']; ?></span>
               </p>
               <p> ID користувача :
                  <span><?php echo $fetch_orders['user_id']; ?></span>
```

```
                </p>
                <p> Дата оформлення :
                   <span><?php echo $fetch_orders['placed_on'];
?></span>
                </p>
                <p> Ім'я :
                   <span><?php echo
htmlspecialchars($fetch_orders['name']); ?></span>
                </p>
                <p> Телефон :
                   <span><?php echo
htmlspecialchars($fetch_orders['number']); ?></span>
                </p>
                <p> Email :
                   <span><?php echo
htmlspecialchars($fetch_orders['email']); ?></span>
                </p>
                <p> Адреса :
                   <span><?php echo
htmlspecialchars($fetch_orders['address']); ?></span>
                </p>
                <p> Товари :
                   <span>
                      <?php
                         $items = explode(',',
$fetch_orders['total_books']);
                         foreach ($items as $item) {
                            if (preg_match('/^(.*?)\s*\(/u',
trim($item), $m)) {
                               echo htmlspecialchars($m[1]) . ' ';
                            } else {
                               echo htmlspecialchars($item) . ' ';
                            }
                         }
                      ?>
                   </span>
                </p>
                <p> Загальна сума :
                   <span><?php echo
htmlspecialchars($fetch_orders['total_price']); ?>₴</span>
                </p>
                <p> Спосіб оплати :
                   <span><?php echo
htmlspecialchars($fetch_orders['method']); ?></span>
                </p>
                <form action="" method="post" class="update-form">
                   <input type="hidden" name="order_id" value="<?php
echo $fetch_orders['id']; ?>">
                   <select name="update_payment" class="select-
payment">
                      <option value="" selected disabled>
                         <?php
                            if ($fetch_orders['payment_status'] ===
'pending') {
```

Продовження додатку В

```
                                echo 'Очікує оплати';
                            } elseif ($fetch_orders['payment_status']
=== 'completed') {
                                echo 'Оплачено';
                            } else {
                                echo
htmlspecialchars($fetch_orders['payment_status']);
                            }
                        ?>
                    </option>
                    <option value="pending">Очікує оплати</option>
                    <option value="completed">Оплачено</option>
                </select>
                <input type="submit" value="Оновити"
name="update_order" class="option-btn">
                <a href="/admin/orders?delete=<?php echo
$fetch_orders['id']; ?>&filter=<?php echo $filter; ?>"
                    onclick="return confirm('Видалити це
замовлення?');"
                    class="delete-btn">
                    Видалити
                </a>
            </form>
        </div>
      <?php endwhile; ?>
   <?php else: ?>
      <p class="empty">Немає жодного замовлення у вибраному
фільтрі.</p>
   <?php endif; ?>
</div>

</section>

<script src="/js/admin_script.js"></script>

</body>
</html>
```

Додаток Г

Сторінка orders.php

```
<?php

include 'config.php';
session_start();

$user_id = $_SESSION['user_id'] ?? null;
$admin_id = $_SESSION['admin_id'] ?? null;

if (!$user_id && !$admin_id) {
    header('location:login.php');
    exit();
}

?>

<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Мої замовлення</title>

    <link rel="stylesheet" href="/css/style.css">
    <style>
        .placed-orders .box-container .box {
            display: flex;
            flex-direction: column;
            padding: 2rem;
            border: var(--border);
            box-shadow: var(--box-shadow);
            border-radius: .5rem;
            background: #fff;
            margin-bottom: 2rem;
        }
        .placed-orders .box-container .box p {
            font-size: 1.6rem;
            color: var(--black);
            margin: .5rem 0;
        }
        .read-links {
            margin-top: 1.5rem;
        }
        .read-links h3 {
            font-size: 2rem;
            color: var(--black);
            margin-bottom: 1rem;
        }
        .read-links .read-item {
            display: flex;
            align-items: center;
```

```
            gap: 1rem;
            margin-bottom: 1rem;
        }
        .read-links .read-item img {
            height: 10rem;
            width: auto;
            border-radius: .5rem;
            object-fit: cover;
        }
        .read-links .read-item .option-btn {
            margin: 0;
        }
    </style>
</head>
<body>

<?php include 'header.php'; ?>



<section class="placed-orders">
    <h1 class="title">Ваші замовлення</h1>
    <div class="box-container">
        <?php
        $select_orders = mysqli_query($conn, "SELECT * FROM `orders`
WHERE user_id = '$user_id' ORDER BY placed_on DESC") or die('query
failed');
        if (mysqli_num_rows($select_orders) > 0) {
            while ($order = mysqli_fetch_assoc($select_orders)) {
        ?>
        <div class="box">
            <p><strong>Ім'я:</strong> <span><?php echo
htmlspecialchars($order['name']); ?></span></p>
            <p><strong>Телефон:</strong> <span><?php echo
htmlspecialchars($order['number']); ?></span></p>
            <p><strong>Email:</strong> <span><?php echo
htmlspecialchars($order['email']); ?></span></p>
            <p><strong>Спосіб оплати:</strong> <span><?php echo
htmlspecialchars($order['method']); ?></span></p>
            <p><strong>Адреса:</strong> <span><?php echo
htmlspecialchars($order['address']); ?></span></p>
            <p><strong>Книги:</strong> <span><?php echo
htmlspecialchars($order['total_books']); ?></span></p>
            <p><strong>Загальна вартість:</strong> <span><?php echo
$order['total_price']; ?>₴</span></p>
            <p><strong>Дата замовлення:</strong> <span><?php echo
htmlspecialchars($order['placed_on']); ?></span></p>
            <p><strong>Статус оплати:</strong> <span
style="color:<?php echo $order['payment_status']=='pending'? 'var(--
red)' : 'var(--green)'; ?>;"><?php echo
$order['payment_status']=='pending' ? 'Очікує оплати' : 'Оплачено';
?></span></p>

            <?php
```

```
            if ($order['payment_status'] === 'completed') {
                $items = array_filter(array_map('trim', explode(',',
$order['total_books'])));
                if (!empty($items)) {
                    echo '<div class="read-links">';
                    echo '<h3>Читати книги:</h3>';
                    foreach ($items as $item) {
                        if (preg_match('/^(.*?)\s*\(/u', $item, $m)) {
                            $prod_name =
mysqli_real_escape_string($conn, $m[1]);
                            $res = mysqli_query($conn, "SELECT id,
image FROM `books` WHERE name = '$prod_name' LIMIT 1");
                            if ($prod = mysqli_fetch_assoc($res)) {
                                echo '<div class="read-item">';
                                echo '<img src="/uploaded_img/' .
$prod['image'] . '" alt="' . htmlspecialchars($prod_name) . '">';
                                echo '<a href="/read_book.php?id=' .
$prod['id'] . '" class="option-btn">Читати "' .
htmlspecialchars($prod_name) . '"</a>';
                                echo '</div>';
                            }
                        }
                    }
                    echo '</div>';
                }
            } else {
                echo '<p class="empty">Книги будуть доступні після
завершення оплати</p>';
            }
            ?>
        </div>
        <?php
            }
        } else {
            echo '<p class="empty">У вас немає замовлень</p>';
        }
        ?>
    </div>
</section>

<?php include 'footer.php'; ?>

<script src="/js/script.js"></script>

</body>
</html>
```

Додаток Д

Навігація для адміна admin_header.php

```
<?php
if (!isset($_SESSION)) {
   session_start();
}
?>

<!DOCTYPE html>
<html lang="uk">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-
scale=1.0">
   <title>Адмін-панель</title>
   <link rel="stylesheet" href="css/admin_style.css">
</head>
<body>

<header class="header">
   <div class="flex">

      <a href="/admin" class="logo">Адмін<span>Панель</span></a>

      <nav class="navbar">
         <a href="/admin">Статистика</a>
         <a href="/admin/products">Створити книгу</a>
         <a href="/admin/categories">Категорії</a>
         <a href="/admin/orders">Замовлення</a>
         <a href="/admin/users">Користувачі</a>
         <a href="/admin/contacts">Повідомлення</a>
      </nav>

      <div class="icons">
         <div id="menu-btn" class="fas fa-bars"></div>
         <div id="user-btn" class="fas fa-user"></div>
      </div>

      <div class="account-box">
         <p>Ім'я: <span><?php echo isset($_SESSION['admin_name']) ?
$_SESSION['admin_name'] : 'admin'; ?></span></p>
         <p>Email: <span><?php echo isset($_SESSION['admin_email']) ?
$_SESSION['admin_email'] : 'admin@example.com'; ?></span></p>
         <a href="logout.php" class="delete-btn">Вийти</a>
      </div>

      <span id="admin-notify" style="display:inline-block; margin-
left:1rem; color:#e67e22; font-size:1.5rem; position:relative;
cursor:pointer;">
         🔔 <span id="notify-count" style="font-weight:bold;"></span>
         <div id="notify-dropdown" style="display:none;
position:absolute; right:0; top:2.2rem; background:#fff; border:1px
solid #eee; box-shadow:0 2px 8px #ccc; min-width:260px; z-index:10;
border-radius:8px;">
```

```
            <div id="notify-list" style="max-height:300px; overflow-
y:auto;"></div>
         </div>
      </span>

   </div>
</header>

<script>
let lastOrders = [];
function checkAdminNotifications() {
    fetch('/admin_notify.php')
      .then(r => r.json())
      .then(data => {
         const count = data.pending_orders || 0;
         document.getElementById('notify-count').textContent = count >
0 ? count : '';
         document.getElementById('admin-notify').style.opacity = '1';
         lastOrders = data.orders || [];
         renderNotifyList();
      });
}
function renderNotifyList() {
   const list = document.getElementById('notify-list');
   if (!lastOrders.length) {
      list.innerHTML = '<div style="padding:1rem; color:#fff;
background:#ff9800; border-radius:6px; text-align:center; font-
weight:bold; opacity:1;">Немає нових замовлень</div>';
      return;
   }
   list.innerHTML = lastOrders.map(order =>
      `<div style="padding:.7rem 1rem; border-bottom:1px solid
#f3f3f3;">
         <strong>Замовлення #${order.id}</strong><br>
         Ім’я: ${order.name}<br>
         Сума: <span
style="color:#e67e22;">${order.total_price}₴</span><br>
         <small>${order.placed_on}</small>
         <div style="margin-top:4px;">
            <a href="/admin/orders?filter=pending#order${order.id}"
style="color:#e67e22; text-decoration:underline; font-
size:.97em;">Детальніше</a>
         </div>
      </div>`
   ).join('');
}
document.getElementById('admin-notify').onclick = function(e) {
   const dropdown = document.getElementById('notify-dropdown');
   dropdown.style.display = dropdown.style.display === 'block' ?
'none' : 'block';
   e.stopPropagation();
};
document.body.onclick = function() {
   document.getElementById('notify-dropdown').style.display = 'none';
```

```
};
setInterval(checkAdminNotifications, 15000);
checkAdminNotifications();
</script>
```

Додаток Е

Сторінка обраного товару product.php

```
<?php
   include 'config.php';
   session_start();

   if (!isset($_SESSION['user_id']) && !isset($_SESSION['admin_id']))
{
       header('Location: login.php');
       exit();
   }
   $product_id = intval($_GET['id'] ?? 0);
   $product = null;

   if ($product_id > 0) {
       $query = mysqli_query(
           $conn,
           "SELECT b.*, c.name AS category_name
            FROM `books` b
            LEFT JOIN `categories` c ON b.category_id = c.id
            WHERE b.id = '$product_id'"
       ) or die('query failed');

       if (mysqli_num_rows($query) > 0) {
           $product = mysqli_fetch_assoc($query);
       }
   }
?>

<!DOCTYPE html>
<html lang="uk">
<head>
   <meta charset="UTF-8">
   <title>Картка товару</title>
   <meta name="viewport" content="width=device-width, initial-
scale=1.0">
   <link rel="stylesheet" href="/css/style.css">
   <style>
      body {
         background: url('/images/backBook.jpg') no-repeat center
center fixed;
         background-size: cover;
      }
      .container-main {
         max-width: 1200px;
         margin: 3rem auto;
         background: var(--container-bg, #fff);
         border-radius: var(--border-radius, 0.6rem);
         box-shadow: 0 0 24px var(--card-shadow, #e0e0e0);
         padding: 2.5rem 2rem;
      }
      .back-link {
         display: inline-block;
         margin-bottom: 2rem;
         color: var(--primary-color, #7a5c1d);
```

```
    font-size: 1.2rem;
    transition: color 0.2s;
}
.back-link:hover {
    color: var(--accent-color, #e67e22);
}
.product-card {
    display: flex;
    flex-wrap: wrap;
    gap: 2rem;
    align-items: flex-start;
}
.product-image {
    flex: 1 1 220px;
    max-width: 260px;
    min-width: 180px;
    background:rgb(164, 152, 118);
    border-radius: var(--border-radius, 0.6rem);
    overflow: hidden;
    box-shadow: 0 2px 10px var(--card-shadow, #e0e0e0);
    display: flex;
    align-items: center;
    justify-content: center;
    height: 340px;
}
.product-image img {
    width: 100%;
    height: 100%;
    object-fit: cover;
}
.product-info {
    flex: 2 1 320px;
    min-width: 220px;
}
.product-title {
    font-size: 2.2rem;
    color: var(--primary-color, #7a5c1d);
    margin-bottom: 0.7rem;
    font-weight: bold;
}
.product-category {
    background: #f3f0e8;
    color: var(--primary-color, #7a5c1d);
    font-weight: bold;
    padding: 0.5rem 1rem;
    border-radius: 0.4rem;
    display: inline-block;
    font-size: 1.1rem;
    margin-bottom: 1.2rem;
}
.product-price {
    color: #c0392b;
    font-size: 1.7rem;
    font-weight: bold;
```

```
   margin-bottom: 1.2rem;
}
.product-description {
   background: #f9f9f9;
   border-left: 4px solid var(--primary-color, #7a5c1d);
   color: #444;
   padding: 1.2rem 1.5rem;
   border-radius: 0.4rem;
   margin-bottom: 1.5rem;
   font-size: 1.15rem;
   white-space: pre-line;
}
.actions {
   display: flex;
   gap: 1rem;
   flex-wrap: wrap;
   align-items: center;
}
.actions input[type="number"] {
   width: 5rem;
   padding: 0.5rem;
   font-size: 1rem;
   border: 1px solid #ccc;
   border-radius: 0.4rem;
}
.btn {
   padding: 0.9rem 2.2rem;
   font-size: 1.15rem;
   border: none;
   border-radius: 0.5rem;
   cursor: pointer;
   background: var(--primary-color, #7a5c1d);
   color: #fff;
   font-weight: bold;
   transition: background 0.2s;
}
.btn:hover {
   background: var(--accent-color, #e67e22);
}
@media (max-width: 700px) {
   .container-main {
      padding: 1rem 0.5rem;
   }
   .product-card {
      flex-direction: column;
      align-items: stretch;
   }
   .product-image {
      max-width: 100%;
      height: 220px;
   }
}
.recommend-title {
   font-size: 1.3rem;
```

```
    font-weight: bold;
    margin: 2.5rem 0 1.2rem 0;
    color: var(--primary-color, #7a5c1d);
}
.recommend-row {
    display: flex;
    flex-wrap: wrap;
    gap: 1.5rem;
    margin-bottom: 2rem;
}
.recommend-card {
    background: #fff;
    border-radius: .7rem;
    box-shadow: 0 2px 10px var(--card-shadow, #e0e0e0);
    padding: 1rem;
    width: calc(25% - 1.2rem);
    min-width: 160px;
    display: flex;
    flex-direction: column;
    align-items: center;
    transition: box-shadow 0.2s;
}
.recommend-card:hover {
    box-shadow: 0 4px 18px var(--card-shadow, #d2cfc7);
}
.recommend-img-link {
    display: block;
    width: 100%;
    height: 140px;
    margin-bottom: .7rem;
    overflow: hidden;
    border-radius: .5rem;
    background: #f3f0e8;
    text-align: center;
}
.recommend-img-link img {
    width: 100%;
    height: 100%;
    object-fit: contain;
    display: block;
}
.recommend-info {
    text-align: center;
}
.recommend-name {
    font-size: 1.08rem;
    font-weight: 500;
    margin-bottom: .3rem;
    color: var(--primary-color, #7a5c1d);
    min-height: 2.2em;
    overflow: hidden;
}
.recommend-price {
    color: #c0392b;
```

```
            font-size: 1.1rem;
            font-weight: bold;
        }
        @media (max-width: 900px) {
            .recommend-row { flex-wrap: wrap; }
            .recommend-card { width: calc(50% - 1.2rem); }
        }
        @media (max-width: 600px) {
            .recommend-row { flex-direction: column; gap: 1rem; }
            .recommend-card { width: 100%; }
            .recommend-img-link { height: 110px; }
        }
    </style>
</head>
<body>

    <?php include 'header.php'; ?>

    <div class="container-main">
        <a href="/shop" class="back-link">&larr; Повернутися до
магазину</a>

        <?php if ($product): ?>
            <div class="product-card">
                <div class="product-image">
                    <img src="/uploaded_img/<?php echo
htmlspecialchars($product['image']); ?>" alt="cover">
                </div>
                <div class="product-info">
                    <div class="product-title">
                        <?php echo htmlspecialchars($product['name']); ?>
                    </div>
                    <div class="product-category">
                        <?php echo
htmlspecialchars($product['category_name'] ?? 'Без категорії'); ?>
                    </div>
                    <div class="product-price">
                        <?php echo $product['price']; ?> ₴
                    </div>
                    <div class="product-description">
                        <?php echo
nl2br(htmlspecialchars($product['description'])); ?>
                    </div>
                    <form action="/shop" method="post" class="actions">
                        <input type="hidden" name="product_id" value="<?php
echo $product['id']; ?>">
                        <input type="number" name="product_quantity"
value="1" min="1">
                        <input type="submit" name="add_to_cart" class="btn"
value="Додати в кошик">
                    </form>
                </div>
            </div>
            <?php
```

```
            $category_id = intval($product['category_id']);
            $recommend_query = mysqli_query(
                $conn,
                "SELECT b.*, c.name AS category_name
                 FROM `books` b
                 LEFT JOIN `categories` c ON b.category_id = c.id
                 WHERE b.category_id = '$category_id' AND b.id !=
'{$product['id']}'
                 ORDER BY RAND() LIMIT 4"
            );
         ?>
         <?php if (mysqli_num_rows($recommend_query) > 0): ?>
         <div class="recommend-title">Рекомендації з цієї
категорії:</div>
         <div class="recommend-row">
            <?php while($rec = mysqli_fetch_assoc($recommend_query)):
?>
               <div class="recommend-card">
                  <a href="/product/<?php echo $rec['id']; ?>"
class="recommend-img-link">
                     <img src="/uploaded_img/<?php echo
htmlspecialchars($rec['image']); ?>" alt="<?php echo
htmlspecialchars($rec['name']); ?>">
                  </a>
                  <div class="recommend-info">
                     <div class="recommend-name"><?php echo
htmlspecialchars($rec['name']); ?></div>
                     <div class="recommend-price"><?php echo
$rec['price']; ?> ₴</div>
                  </div>
               </div>
            <?php endwhile; ?>
         </div>
         <?php endif; ?>
      <?php else: ?>
         <p>Товар не знайдено.</p>
      <?php endif; ?>
   </div>

</body>
</html>
```

Додаток Ж

Сторінка повідомлень до адміна admin_contacts.php

```
<?php

include 'config.php';
session_start();

$admin_id = $_SESSION['admin_id'];
if (!isset($admin_id)) {
   header('location:login.php');
   exit;
}
if (isset($_GET['delete'])) {
   $delete_id = $_GET['delete'];
   mysqli_query($conn, "DELETE FROM `message` WHERE id =
'$delete_id'") or die('query failed');
   header('Location: /admin/contacts');
   exit;
}

?>

<!DOCTYPE html>
<html lang="uk">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-
scale=1.0">
   <title>Повідомлення</title>

   <link rel="stylesheet" href="/css/admin_style.css">
</head>
<body>

<?php include 'admin_header.php'; ?>

<section class="messages">

   <h1 class="title">Повідомлення</h1>

   <div class="contacts-container">

      <div class="box-container scrollable">
         <?php
            $select_message = mysqli_query($conn, "SELECT * FROM
`message` ORDER BY id DESC") or die('query failed');
            if (mysqli_num_rows($select_message) > 0) {
               while ($fetch_message =
mysqli_fetch_assoc($select_message)) {
         ?>
            <div class="box">
               <p> ID користувача : <span><?php echo
$fetch_message['user_id']; ?></span> </p>
```

```
                <p> Ім'я : <span><?php echo
htmlspecialchars($fetch_message['name']); ?></span> </p>
                <p> Телефон : <span><?php echo
htmlspecialchars($fetch_message['number']); ?></span> </p>
                <p> Email : <span><?php echo
htmlspecialchars($fetch_message['email']); ?></span> </p>
                <p> Повідомлення : <span><?php echo
nl2br(htmlspecialchars($fetch_message['message'])); ?></span> </p>
                <a href="/admin/contacts?delete=<?php echo
$fetch_message['id']; ?>"
                   onclick="return confirm('Видалити це
повідомлення?');"
                   class="delete-btn">
                   Видалити повідомлення
                </a>
             </div>
          <?php
                }
             } else {
                echo '<p class="empty">У вас немає повідомлень!</p>';
             }
          ?>
       </div>

   </div>

</section>
<script src="/js/admin_script.js"></script>

</body>
</html>
```

Додаток И

Сторінка кошика cart.php

```
<?php

include 'config.php';

session_start();

$user_id = $_SESSION['user_id'];

if(!isset($user_id)){
   header('location:login.php');
}

if(isset($_POST['update_cart'])){
   $cart_id = $_POST['cart_id'];
   $cart_quantity = $_POST['cart_quantity'];
   mysqli_query($conn, "UPDATE `cart` SET quantity = '$cart_quantity'
WHERE id = '$cart_id'") or die('query failed');
   $message[] = 'корзина оновлена!';
}

if(isset($_GET['delete'])){
   $delete_id = $_GET['delete'];
   mysqli_query($conn, "DELETE FROM `cart` WHERE id = '$delete_id'")
or die('query failed');
   header('location:cart.php');
}

if(isset($_GET['delete_all'])){
   mysqli_query($conn, "DELETE FROM `cart` WHERE user_id =
'$user_id'") or die('query failed');
   header('location:cart.php');
}

?>

<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-
scale=1.0">
   <title>cart</title>
   <link rel="stylesheet" href="css/style.css">

   <style>
      .shopping-cart .box img {
         max-width: 100%;
         max-height: 300px;
         width: auto;
         height: auto;
         object-fit: contain;
```

```
         display: block;
         margin: 0 auto 1rem auto;
         border-radius: .4rem;
         background: #f8f8f8;
      }
      .cart-remove {
   position: absolute;
   top: 10px;
   right: 12px;
   color: #e74c3c;
   font-size: 1.5rem;
   text-decoration: none;
   background: none;
   border: none;
   cursor: pointer;
   z-index: 2;
   transition: color 0.2s;
}
.cart-remove:hover {
   color: #b71c1c;
   transform: scale(1.15);
}
.shopping-cart .box {
   position: relative;
}
.cart-details {
   font-size: 1.05rem;
   color: #555;
   margin-bottom: 8px;
}
.cart-details span {
   margin-right: 8px;
}
   </style>
</head>
<body>

<?php include 'header.php'; ?>


<section class="shopping-cart">

   <h1 class="title">Додані книги</h1>
   <div class="box-container">
      <?php
         $grand_total = 0;
         $select_cart = mysqli_query($conn, "SELECT * FROM `cart`
WHERE user_id = '$user_id'") or die('query failed');
         if(mysqli_num_rows($select_cart) > 0){
            while($fetch_cart = mysqli_fetch_assoc($select_cart)){
               $book_name = mysqli_real_escape_string($conn,
$fetch_cart['name']);
               $book_query = mysqli_query($conn, "SELECT * FROM
`books` WHERE name = '$book_name' LIMIT 1");
```

```
            $book = mysqli_fetch_assoc($book_query);
      ?>
      <div class="box">
         <a href="cart.php?delete=<?php echo $fetch_cart['id']; ?>"
class="cart-remove" title="Видалити" onclick="return confirm('Ви
хочете це видалити із кошика?');">✕</a>
         <img src="uploaded_img/<?php echo $fetch_cart['image']; ?>"
alt="">
         <div class="name">
            <?php if ($book): ?>
               <a href="product.php?id=<?php echo $book['id']; ?>"
style="color:inherit;text-decoration:none;">
                  <?php echo htmlspecialchars($fetch_cart['name']); ?>
               </a>
            <?php else: ?>
               <?php echo htmlspecialchars($fetch_cart['name']); ?>
            <?php endif; ?>
         </div>
         <div class="cart-details" style="font-size:1.05rem;
color:#555; margin-bottom:8px;">
            <span>Ціна: <strong><?php echo $fetch_cart['price'];
?>₴</strong></span>  | 
            <span>Кількість: <strong><?php echo
$fetch_cart['quantity']; ?></strong></span>  | 
            <span>Сума: <strong><?php echo $sub_total =
($fetch_cart['quantity'] * $fetch_cart['price']); ?>₴</strong></span>
         </div>
         <form action="" method="post">
            <input type="hidden" name="cart_id" value="<?php echo
$fetch_cart['id']; ?>">
            <input type="number" min="1" name="cart_quantity"
value="<?php echo $fetch_cart['quantity']; ?>">
            <input type="submit" name="update_cart" value="Оновити"
class="option-btn">
         </form>
      </div>
      <?php
      $grand_total += $sub_total;
         }
      }else{
         echo '<p class="empty">Ваш кошик порожній</p>';
      }
      ?>
   </div>

   <div style="margin-top: 2rem; text-align:center;">
      <a href="cart.php?delete_all" class="delete-btn <?php echo
($grand_total > 1)?'':'disabled'; ?>" onclick="return confirm('Ви
дійсно хочете видалити все з кошика?');">Прибрати все</a>
   </div>

   <div class="cart-total" style="border: 2.5px solid #ff9800; border-
radius: 0.7rem; background: #fffbe7; box-shadow: 0 2px 12px #ffecb3a0;
padding: 1.2rem 2rem; margin-top: 2rem; font-size: 1.2rem;">
```

Продовження додатку И

```
      <p style="font-weight:700; color:#d17b00;">Загалом : <span><?php
echo $grand_total; ?>₴</span></p>
      <div class="flex">
         <a href="shop.php" class="option-btn">Продовжити покупки</a>
         <a href="checkout.php" class="btn <?php echo ($grand_total >
1)?'':'disabled'; ?>">Оформити замовлення</a>
      </div>
   </div>

</section>

<?php if (!empty($message)): ?>
    <?php foreach ($message as $msg): ?>
        <div class="message">
            <span><?php echo htmlspecialchars($msg); ?></span>
            <i class="fas fa-times"
onclick="this.parentElement.style.display='none';"></i>
        </div>
    <?php endforeach; ?>
<?php endif; ?>

<?php include 'footer.php'; ?>
<script src="js/script.js"></script>
<script>
document.addEventListener('DOMContentLoaded', function() {
    document.querySelectorAll('.message').forEach(function(msg) {
        setTimeout(function() {
            msg.style.opacity = '0';
            setTimeout(function() { msg.remove(); }, 300);
        }, 3000);
    });
});
</script>

</body>
</html>
```

Додаток К

Отримання бази даних BookRepository.cs

```
using MySqlConnector;
using System.Collections.Generic;

namespace BookViewer3D;

public static class BookRepository
{
    private const string ConnStr =

"Server=localhost;Database=shop_db;Uid=root;Pwd=;Charset=utf8mb4;";

    public static List<Book> GetAll()
    {
        var list = new List<Book>();

        using var conn = new MySqlConnection(ConnStr);
        conn.Open();

        const string sql = @"
            SELECT id, name,
                   cover_front, cover_back, spine
            FROM books";

        using var cmd = new MySqlCommand(sql, conn);
        using var rdr = cmd.ExecuteReader();

        int ordId = rdr.GetOrdinal("id");
        int ordName = rdr.GetOrdinal("name");
        int ordCoverFront = rdr.GetOrdinal("cover_front");
        int ordCoverBack = rdr.GetOrdinal("cover_back");
        int ordSpine = rdr.GetOrdinal("spine");

        while (rdr.Read())
        {
            list.Add(new Book(
                rdr.GetInt32(ordId),
                rdr.GetString(ordName),
                rdr.IsDBNull(ordCoverFront) ? "" :
rdr.GetString(ordCoverFront),
                rdr.IsDBNull(ordCoverBack) ? "" :
rdr.GetString(ordCoverBack),
                rdr.IsDBNull(ordSpine) ? "" : rdr.GetString(ordSpine)
            ));
        }
        return list;
    }
}
```