Вищий навчальний заклад
«Університет економіки та права «КРОК»
Фаховий коледж

Циклова комісія з інформаційних технологій

Кваліфікаційна робота фахового молодшого бакалавра

на тему Створення інтернет-магазину для продажу книг

Виконав _____
(Підпис)

Потьомкін Нікіта Сергійович
(прізвище, ім'я, по батькові

_____

Науковий керівник
Пантєєв Роман Леонідович
(прізвище, ім'я, по батькові)

_____

_____
(Резолюція «До захисту»)

Попередній захист:

_____
(Висновок: "До захисту в екзаменаційній комісії ")

Голова циклової комісії _____ _____ _____
(Підпис) (Прізвище, ініціали) (Дата)

Київ – 2025 року

ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД

УНІВЕРСИТЕТ ЕКОНОМІКИ ТА ПРАВА «КРОК»

Фаховий коледж

Циклова комісія з інформаційних технологій

Спеціальність 121 інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Голова циклової комісії _____ Леонід УВАРОВ

(підпис)

«____» _____ 2025 року

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Здобувач освіти Потьомкін Нікіта Сергійович

1. Тема роботи створення інтернет магазину для продажу книг затверджена наказом по університету

від «___» _____ 202__ р. № _____

2. Термін здачі закінченої роботи «30» травня 2025 року

3. Вихідні дані до роботи:

1) цільова аудиторія - покупці книг, читачі різних жанрів, студенти, викладачі, любителі літератури тощо;
2) функціональність - пошук книг за назвою, автором, жанром; оформлення замовлення; відстеження статусу замовлення; рекомендації книг; сповіщення про новинки та знижки тощо;
3) технічні вимоги - веб-платформа для розробки сайту, мова програмування, база даних, фреймворки та інші технології;
4) можливість інтеграції (взаємодії) з існуючими системами управління складом, платіжними системами (LiqPay, Mono Bank, Приват24 тощо), службами доставки (Нова Пошта, УкрПошта тощо);
5) існуючі рішення та кращі практики у сфері розробки веб-сайтів для електронної комерції та книжкових магазинів.

4. Зміст пояснювальної записки

1) Розділ 1 Теоретична частина. Аналіз існуючих рішень та кращих практик у сфері розробки веб-сайтів для електронної комерції та книжкових магазинів, обґрунтування вибору платформи та технології для розроблення сайту.
2) Розділ 2 Проєктування та розробка. Створення зручного та інтуїтивно зрозумілого інтерфейсу користувача для взаємодії з сайтом, розробка алгоритму, за яким сайт буде обробляти запити користувачів щодо пошуку та замовлення книг, забезпечення можливості обміну даними між сайтом та існуючими системами управління товарами, платіжними системами та службами доставки (за необхідності).
3) Розділ 3 Експериментальна частина. Перевірка роботи сайту на різних пристроях та в різних браузерах, виявлення та виправлення помилок, аналіз того, наскільки сайт відповідає поставленим завданням та задовольняє потреби користувачів, інструкція користувачам (для технічного персоналу – опис процесу розробки сайту, його архітектури та функціональності, для звичайних користувачів – як користуватися сайтом для пошуку, замовлення книг та отримання необхідної інформації). Рекомендації щодо забезпечення безпеки персональних даних клієнтів та дотримання норм електронної комерції.

5. Перелік графічного матеріалу:

Скріншоти існуючих рішень

Скріншоти інтерфейсу продукту

Дата видачі завдання «12» лютого 2025 року

Науковий керівник _____ Пантєєв Р. Л.

(підпис) (прізвище, ім'я, по батькові)

Завдання прийняла до виконання _____ Потьомкін Н. С.

(підпис) (прізвище, ім'я, по батько

РЕФЕРАТ

Пояснювальна записка - 54 сторінок, 15 джерел**.**

Об’єкт дослідження - інтернет-торгівля книжковою продукцією.

Мета роботи - Метою даного проекту є розробка зручного та функціонального веб-сайту інтернет-магазину «Chapter One» для продажу книжкової продукції через мережу Інтернет. Наш магазин допоможе читачам будь-якого віку легко знаходити та купувати потрібні їм книги, не виходячи з дому. Сьогодні, коли все більше людей шукають можливість придбати книги онлайн, створення спеціалізованого інтернет-магазину книг є дуже актуальним. Такий сайт не тільки спростить процес пошуку та купівлі книг, але й допоможе популяризувати читання серед населення. Розроблюваний веб-сайт « Chapter One « об’єднає в одному місці широкий асортимент літератури різних жанрів та напрямків – від класичної та сучасної художньої літератури до наукових видань та підручників. Особлива увага буде приділена зручності користування сайтом та швидкості обробки замовлень, що дозволить забезпечити максимальне задоволення потреб покупців.

ABSTRACT

Explanatory note - 54 pages, 15 sources.

Object of research - online book trade.

Purpose - The purpose of this project is to develop a convenient and functional website for the online store "Chapter One" for the sale of books via the Internet. Our store will help readers of all ages easily find and buy the books they need without leaving their homes. The Chapter One website under development will bring together in one place a wide range of literature of various genres and trends - from classic and contemporary fiction to scientific publications and textbooks. Particular attention will be paid to the site's user-friendliness and speed of order processing, which will ensure maximum customer satisfaction.

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

Номер	Назва етапів проекту (роботи)	Термін виконання етапів проекту(роботи)	Примітка
1	Вибір та затвердження теми дипломного проекту	1-2 тиждень	Вибір теми та погодження з керівником
2	Ознайомлення з літературними та іншими джерелами по темі дипломного проекту	3-4 тиждень	Пошук та аналіз наукових статей і джерел
3	Розробка концепції та плану дипломного проекту	5-6 тиждень	Підготовка концепції роботи
4	Затвердження завдання на дипломний проект та графік консультацій	7 тиждень	Погодження завдання та консультацій
5	Підготовка першого розділу роботи та подання науковому керівнику	8-9 тиждень	Розробка і подання першого розділу
6	Підготовка другого розділу роботи та подання науковому керівнику	10-11 тиждень	Розробка і подання другого розділу
7	Звіт керівника про стан виконання дипломного проекту	12 тиждень	Оцінка виконання роботи
8	Підготовка третього розділу роботи та подання науковому керівнику	13-14 тиждень	Розробка і подання третього розділу
9	Підготовка повної версії дипломного проекту	15-16 тиждень	Комплексна підготовка роботи
10	Коригування роботи за зауваженнями наукового керівника	17 тиждень	Внесення правок до роботи
11	Представлення завершеної роботи науковому керівнику	17 тиждень	Подання остаточного варіанту роботи

12	Попередній захист дипломного проекту	18 тиждень	Захист проекту перед керівником
13	Перевірка роботи на плагіат	18-19 тиждень	Перевірка на плагіат
14	Підготовка супровідних документів (рецензія, відгук)	19 тиждень	Оформлення супровідної документації
15	Оформлення дипломного проекту згідно з вимогами	19-20 тиждень	Оформлення тексту та графіки
16	Подання готової роботи на захист	20 тиждень	Подання для захисту
17	Захист дипломного проекту	21 тиждень	Захист дипломної роботи
18	Оцінка роботи на засіданні комісії	21 тиждень	Оцінка проекту та підготовка висновку

Здобувач вищої освіти Потьомкін Н. С. _____
(підпис)

Керівник Пантєєв Р. Л. _____
(підпис)

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

1. **API** - Інтерфейс програмування додатків, що дозволяє взаємодіяти з іншими системами.
2. **Код** - Програмний код бота.
3. **Тестування** - Перевірка функціональності бота.
4. **Інтеграція** - Об’єднання бота з іншими медичними або платіжними системами.
5. **SSL/TLS** - Протоколи для захисту даних при їх передачі по мережі.
6. **ПІБ** - Прізвище, ім’я, по батькові.
7. **Реєстрація** - Процес створення облікового запису користувача в системі.
8. **Конфіденційність** - Захист персональних і медичних даних від несанкціонованого доступу.
9. **Шифрування** - Процес кодування даних для їх захисту.
10. **Онлайн-оплата** - Процес оплати через інтернет для медичних послуг.

ВСТУП

Актуальність завдання. В теперішній час все більше процесів переходить у цифрову площину, і книжкова справа - не виняток. Сьогодні дедалі більше людей купують книги онлайн. Це не тільки зручно, а й дозволяє значно розширити вибір. Більшість традиційних книгарень обмежені простором, натомість інтернет-магазин може запропонувати тисячі варіантів . Саме тому розробка сайту, де можна купити книги через інтернет, є актуальною і затребуваною. Особливо це важливо для регіонів, де немає великого вибору друкованої продукції. Інтернет магазин дає змогу придбати будь-яку книгу незалежно від місця розташування. Це і стало основою для обрання теми дипломної роботи.

Мета роботи. Розробка інтернет магазину книг, серверної та клієнської частини.

Завдання роботи. Завдання моєї дипломної роботи є створення інтернет-магазину книг, який буде простим у використанні та зручним як для покупців, так і для адміністраторів. Щоб досягти цієї мети, я визначив кілька ключових завдань: продумати, як має виглядати структура магазину: сторінки, розділи, навігація, розробити логіку пошуку книг за назвою, автором, жанром, категорією тощо, реалізувати реєстрацію, вхід, перегляд книг, додавання до кошика, оформлення замовлення забезпечити можливість адміністрування: додавання нових книг, редагування інформації, перегляд замовлень, протестувати функціонал на зручність і стабільність роботи.

Об’єкт дослідження. У роботі розглядається процес організації онлайн-продажу книг - починаючи з моменту вибору книги користувачем і завершуючи обробкою замовлення адміністратором. Також враховується, як зберігаються й обробляються дані, як взаємодіє користувач із системою, і які задачі стоять перед сервером.

Предмет дослідження. Предметом дослідження є сам програмний продукт — вебзастосунок, який дозволяє здійснювати продаж книг в інтернеті. Його

функціональність, структура бази даних, поведінка інтерфейсу та логіка виконання дій користувачем.

Методі дослідження. Під час роботи я використовував такі методи: аналіз аналогічних сайтів, щоб зрозуміти, як працюють популярні інтернет-магазини книг, потім - створення макету сайту, схеми бази даних, вибір інструментів для розробки. Для програмування я використовував сучасні підходи, патерни програмування, а для тестування - як ручне за допомогою додатку postman так і автоматизоване за допомогою unit тестів.

Практичне значення одержаних результатів. Інтернет-магазин книг може бути використаний як повноцінний бізнес для продажу літератури. Його можна адаптувати до будь-яких потреб: змінити зовнішній вигляд, додати нові функції, підключити різні платіжні системи. З практичного боку, проєкт є хорошим прикладом того, як із нуля створити працюючу систему електронної комерції, враховуючи як технічну реалізацію, так і потреби реальних користувачів.

РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖЕННЯ

1.1. Аналіз існуючих рішень у сфері інтернет магазинів книг

Сучасна книжкова індустрія переживає значні трансформації завдяки розвитку цифрових технологій та інтернет-торгівлі. В Україні цей процес також набирає обертів, формуючи конкурентне середовище серед онлайн-книгарень. Одним із лідерів українського ринку є Yakaboo, який вирізняється широким асортиментом, зручним інтерфейсом сайту та швидкою доставкою. Платформа пропонує не лише книги, а й створює цілу екосистему для читачів: блог із професійними оглядами, добірки за жанрами, рекомендації від експертів і навіть подкасти про літературу. Такий підхід сприяє формуванню спільноти читачів, що робить Yakaboo більше, ніж просто магазином - це культурний хаб для поціновувачів книг. Ще один помітний гравець - Book24, який робить акцент на простоті та доступності. Їхній сайт мінімалістичний, з інтуїтивно зрозумілим пошуком, що дозволяє швидко знайти потрібну книгу. Book24 активно використовує маркетингові інструменти, такі як знижки, акції та програми лояльності, що особливо приваблює студентів і читачів із обмеженим бюджетом. Окремо варто відзначити їхню активність у соціальних мережах, зокрема в TikTok, де короткі відеоогляди та трендові челенджі залучають молодшу аудиторію. Це демонструє розуміння сучасних тенденцій і потреб покоління Z, яке все частіше обирає книги через соціальні медіа.
Паралельно розвиваються платформи самвидаву, такі як LitRes (переважно російськомовний сегмент) та AuthorToday, де автори можуть самостійно публікувати й продавати свої твори. Ці сервіси дають читачам доступ до унікального контенту, який часто не потрапляє до традиційних видавництв, наприклад, нішеві жанри, фанфіки чи експериментальна література. Для авторів це можливість напряму взаємодіяти з аудиторією, отримуючи більший відсоток прибутку без посередників. Проте такі платформи мають і свої виклики: якість

контенту може бути нерівномірною, а читачам доводиться самостійно фільтрувати матеріал.

Міжнародний досвід також впливає на український ринок. Наприклад, Amazon залишається глобальним лідером завдяки своїй логістиці, персоналізованим рекомендаціям і широкому вибору як фізичних, так і електронних книг. В Україні подібні сервіси, як-от Kobo чи Google Books, менш популярні, але поступово завойовують аудиторію завдяки доступу до англомовної літератури. Загалом, онлайн-торгівля книгами в Україні стрімко розвивається, відповідаючи глобальним трендам, і повне повернення до офлайн-книгарень виглядає малоймовірним. Цифрові платформи не лише розширюють доступ до літератури, а й формують нові моделі взаємодії між читачами, авторами та видавцями.

1.2. Визначення переваг і недоліків

Онлайн-книгарні мають низку переваг, які роблять їх привабливими для сучасних читачів, але також супроводжуються певними недоліками. Розглянемо їх детальніше.

Переваги:

1. **Економія часу та зручність.** Інтернет-магазини дозволяють купувати книги, не виходячи з дому. Наприклад, жителям невеликих міст, де книгарні обмежені десятком полиць, онлайн-платформи відкривають доступ до тисяч найменувань. Пошук за ключовими словами, фільтри за жанром чи автором дозволяють знайти потрібну книгу за лічені хвилини.
2. **Широкий асортимент.** Онлайн-магазини пропонують значно більший вибір порівняно з фізичними книгарнями. Наприклад, на Yakaboo можна знайти як бестселери, так і рідкісні видання, переклади малотиражної літератури чи книги іноземними мовами. Платформи самвидаву, такі як LitRes, додають до асортименту унікальні твори, які не потрапляють до традиційних видавництв.
3. **Доступні ціни.** Завдяки відсутності витрат на оренду торгових площ і меншій кількості персоналу, онлайн-магазини можуть пропонувати конкурентні ціни. Наприклад, Book24 регулярно проводить розпродажі, де книги можна придбати

зі знижками до 50%. Програми лояльності, подарункові сертифікати та акції, як-от "купи три - четверта в подарунок", також сприяють економії.

4. **Порівняння та відгуки.** Онлайн-платформи дозволяють легко порівнювати ціни на різних сайтах, читати відгуки інших покупців і переглядати рейтинги. Наприклад, на Yakaboo користувачі можуть оцінити книгу за п’ятибальною шкалою, що допомагає зробити усвідомлений вибір. У фізичних магазинах такої можливості зазвичай немає.
5. **Додаткові сервіси.** Багато платформ пропонують не лише продаж книг, а й додаткові функції: електронні книги, аудіокниги, передзамовлення новинок. Наприклад, Yakaboo інтегрує функцію прослуховування аудіокниг, що зручно для тих, хто віддає перевагу цьому формату.

Недоліки:

1. **Відсутність тактильного досвіду.** Для багатьох читачів важливо фізично оцінити книгу перед покупкою - відчути якість паперу, палітурки, розмір шрифту. В онлайн-магазинах це неможливо, що особливо відчутно для колекціонерів чи поціновувачів подарункових видань.
2. **Терміни та умови доставки.** Незважаючи на прогрес у логістиці, доставка може займати від кількох днів до кількох тижнів, особливо якщо книга імпортується чи відсутня на складі. Наприклад, замовлення рідкісних видань із-за кордону через Yakaboo може тривати до місяця. Крім того, доставка не завжди безкоштовна, що може збільшувати кінцеву вартість покупки.
3. **Ризик пошкодження.** Хоча більшість магазинів пакують книги надійно, трапляються випадки, коли книги приходять із дефектами: погнуті сторінки, подряпини на обкладинці чи навіть намокання під час транспортування. Це створює незручності, особливо якщо книга призначалася для подарунка.
4. **Відсутність атмосфери книгарні.** Традиційні книгарні створюють особливий настрій: запах нових книг, можливість гортати сторінки, спілкування з консультантами, які можуть порекомендувати щось несподіване. Онлайн-

магазини цього досвіду не відтворюють, що для багатьох читачів є суттєвим недоліком.

5. **Складнощі з поверненням.** Повернення книги, яка не відповідає очікуванням, може бути проблематичним. Наприклад, деякі магазини вимагають заповнення спеціальних форм, оплати зворотної доставки або мають обмеження за термінами повернення. У фізичних книгарнях цей процес зазвичай простіший.
6. **Технічні проблеми.** Сайти онлайн-магазинів можуть стикатися з технічними збоями, як-от помилки під час оформлення замовлення чи проблеми з оплатою. Крім того, у невеликих містах чи селах доступ до швидкого інтернету може ускладнювати покупки.

Незважаючи на ці недоліки, сучасні читачі, особливо молодь, дедалі частіше обирають онлайн-магазини. Зручність, широкий вибір і економія часу переважають над мінусами для більшості користувачів. Особисто я також віддаю перевагу онлайн-покупкам, оскільки вони дозволяють швидко знайти потрібну книгу, порівняти ціни та отримати замовлення прямо додому. Проте для особливих випадків, як-от вибір подарункового видання, я іноді відвідую фізичні книгарні, щоб відчути ту саму "магію" книжкового світу.

РОЗДІЛ 2 ПРАКТИЧНА РЕАЛІЗАЦІЯ

2.1. Опис розробленої системи

Розроблена система являє собою сучасний веб-додаток повного циклу, що охоплює всі аспекти електронної комерції у сфері книжкової торгівлі. Архітектура побудована за принципом розділення клієнтської та серверної частин, що забезпечує гнучкість розробки, легкість масштабування та можливість незалежного розвитку різних компонентів системи**.**

Система надає користувачам комплексний інструментарій для пошуку, вибору та придбання книг. Каталог організований з багаторівневою системою категорій, розширеними фільтрами та інтелектуальним пошуком, що дозволяє швидко знаходити потрібну літературу серед тисяч позицій. Процес покупки максимально спрощено - від додавання товару до кошика до завершення оплати проходить мінімальна кількість кроків. Система підтримує різні способи оплати через інтеграцію з LiqPay, автоматично розраховує вартість доставки та надає користувачам можливість відстежувати статус замовлення в реальному часі.

Адміністративна частина системи забезпечує повний контроль над усіма аспектами роботи магазину. Менеджери можуть керувати каталогом книг, завантажувати та редагувати інформацію про товари, встановлювати ціни та знижки, модерувати відгуки користувачів.

Автоматизація процесів мінімізує ручну роботу - система автоматично оновлює статуси замовлень, відправляє повідомлення користувачам, синхронізує залишки товарів з постачальниками через API інтеграції.

2.2. Опис архітектури, функціональності та реалізації розробленого рішення

Фронтенд (React)

Клієнтська частина побудована на React 18 з використанням функціональних компонентів та хуків. Це забезпечує сучасний підхід до розробки з акцентом на переносимість коду та продуктивність.

Для управління станом використовується комбінація Context API та Redux Toolkit - Context для глобальних даних типу інформації про користувача, Redux для складних станів кошика та каталогу книг. React Router Dom забезпечує клієнтський роутинг без перезавантаження сторінок.

UI компоненти побудовані з використанням styled-components для стилізації та Material-UI для базових елементів форм. Це дозволяє швидко розробляти консистентний інтерфейс з можливістю кастомізації.

Для роботи з API використовується Axios з інтерцепторами для автоматичного додавання токенів автентифікації та обробки помилок. Реалізовано retry логіку для запитів, які можуть тимчасово не працювати.

Серверна частина (Node.js/Express)

Сервер побудований на архітектурі MVC з чіткою структурою папок. Контролери обробляють HTTP запити, сервіси містять бізнес-логіку, а репозиторії відповідають за взаємодію з базою даних.

Express.js налаштований з необхідними middleware: cors для крос-доменних запитів, helmet для безпеки, compression для стиснення відповідей, та власними middleware для логування та обробки помилок.

Автентифікація реалізована через JWT токени з refresh/access патерном. Access токени мають короткий термін дії (15 хвилин), refresh токени - довгий (7 днів). Додатково підключена Google OAuth 2.0 для швидкої реєстрації через Google акаунт.

Валідація даних відбувається на рівні middleware за допомогою class - validator . Це гарантує, що в базу даних потрапляють тільки правильно структуровані дані.

База даних (PostgreSQL + TypeORM).

PostgreSQL обрана як основна база даних завдяки її надійності та підтримці складних запитів. Структура включає таблиці для користувачів, книг, категорій, замовлень, відгуків та кошиків.

TypeORM використовується як ORM для взаємодії з базою. Це дозволяє писати запити на TypeScript, автоматично генерувати міграції та валідувати дані на рівні сутностей.

Індекси створені на найбільш часто використовуваних полях: назва книги, автор, категорія, email користувача. Це значно прискорює пошук і фільтрацію. Кешування (Redis)

Redis використовується для кешування часто запитуваних даних: популярні книги, результати пошуку, інформація про користувачів. TTL налаштований індивідуально для кожного типу даних.

Також Redis зберігає сесії користувачів та тимчасові дані типу кодів підтвердження для реєстрації.

Зовнішні інтеграції

AWS S3 для зберігання обкладинок книг. Файли завантажуються через presigned URLs для безпеки, автоматично оптимізуються за розміром та форматом. Налаштовано CDN для швидкої доставки зображень користувачам.

LiqPay як основний платіжний шлюз. Інтеграція включає створення платежів, обробку callback'ів про успішні транзакції, повернення коштів. Всі платіжні дані зберігаються в зашифрованому вигляді.

MailJet для відправки email повідомлень: підтвердження реєстрації, сповіщення про зміну статусу замовлення, розсилка новинок. Використовуються готові шаблони з можливістю персоналізації.

Безпека та продуктивність - сервер захищений від основних типів атак: SQL injection (через ORM), XSS (валідація даних), CSRF, брутфорс (rate limiting). Всі паролі хешуються через bcrypt з індивідуальними солями.

Для продуктивності реалізовано pagination для великих списків, lazy loading для зображень, compression для API відповідей. Моніторинг здійснюється через логування помилок та метрики продуктивності.

Також створенна UML діаграма всіх модулів на серверній частині сайту, знайти можна в додатках:

Структура бази даних

Концептуальна модель даних системи "Chapter one" відображана як UML діаграма основні сутності та зв'язки між ними, зроблена за допомогою сервісу draw.io, знайти можна в додатках.

Таблиці які присутні в проекті:

1.Users
2.Books
3.Languages
4.Publishers
5.Reset_passwords
6.Publishers
7.Comments
8.Orders
9.Refresh_sessions
10.Genres
11.Categories
12.Authors

База даних розроблена з урахуванням вимог системи та необхідності ефективного зберігання і доступу до інформації. Структура бази даних представляє собою набір взаємопов'язаних таблиць, які забезпечують збереження всіх необхідних даних у нормалізованому вигляді, що дозволяє уникнути дублювання інформації та забезпечити цілісність даних.

Основний функціонал системи реалізований через наступні модулі:

Модуль користувача. Вся логіка яка пов’язана з користувачем виконується тут.

Авторизація - реалізовано з використанням JWT-токенів для безпечної авторизації користувачів. Підтримка ролей (користувач, адміністратор) для розмежування доступу до різних функцій системи. Для реєстрації в системі, користувач повинен ввести в форму такі дані: ім’я, пошту, пароль. Після того як користувач ввів необхідні дані, і натиснув кнопку реєстрації, на пошту буде надіслано повідомлення з підтвердженням на пошту, користувач його відкриває та отримає пару JWT токенів (access,refresh).Також в нас реалізована авторизація за допомогою Google OAuth. Якщо користувач не пам'ятає пароль, він має змогу змінити його.

Адмін панель. Дозволяє адміністраторам додавати, редагувати та видаляти товари. Реалізовано завантаження зображень через хмарне сховище Amazon S3. Адміністратори можуть видаляти коментарі користувачів на сайті, додавати та редагувати категорії, жанри,мови,видавництва.Також адміністратори мають змогу переглядати історію замовлень.

Модуль оформлення замовлень. Включає кошик покупок, форму оформлення замовлення, де користувач вказує свої контактні дані, вибирає спосіб доставки і оплати товару. На нашому сайті реалізована оплата за допомогою платіжної системи LiqPay. В Україна ця платіжна система займає перше місце за кількістю транзакцій завдяки зрозумілій документації та швидкому розгортанню.

Модуль книг. Цей модуль є основную частиною нашого сайту. Завдяки цьому модулю користувачі можуть бачити книги на сайті, робити замовлення, писати відгуки, додавати в обране тощо… Для проектування цього модуля було витрачено найбільше часу, в нас стояла задача зробити так щоб код не тільки працював, а був напасаний з урахуванням патернів проектування та максимально оптимізований.

Модуль коментарів. Цей модуль відповідальний за додавання,оновлення,видалення коментарів користувача. Користувач може ставити лайки на коментарі інших користувачив та відповідати.Обмежень на кількість коментарів для користувача немає.

Модуль оплати. Цьому моделі реалізована логіка оплати за допомогою LiqPay, з клієнтської частини сайту передається в параметри id замовлення, після чого генерується форма оплати, якщо оплата пройшла успішно – платіжна система LiqPay відправляє webhook і я змінюю статус замовлення.У випадку коли користувач обрав спосіб оплати “при отриманні”, менеджер отримає повідомлення з контактними даними користувача, якщо користувач підтвердив замовлення, менеджер змінює статус замовлення.

Модуль промокодів. Цей модуль відповідає за управління промокодами на сайті. Виконує такі операції: створення, видалення,оновлення,перевірка,отримання всіх промокодів.Адміністратор має змогу вказувати строк дії промокоду, мінімальну суму, відсоток знижки.

Модуль категорій,авторів,мов,видавництв,жанрів. Завдяки цим модулямадміністратори сайту можуть зручно створювати нові книги в база даних. Доступні такі CRUD операції: створення,оновлення, видалення,отримання всіх.

Реалізація існуючого рішення на практиці

1. Реєстрація

Для повноцінного використання всіх функцій сайту рекомендуємо зареєструватися:

1. Натисніть кнопку "Реєстрація" у верхньому меню сайту.
2. Заповніть форму реєстрації, вказавши:
    - Ім'я
    - Електронну пошту
    - Пароль (не менше 8 символів)
3. Натисніть кнопку "Зареєструватися".
4. На вказану вами електронну пошту буде надіслано лист із посиланням для підтвердження реєстрації.
5. Перейдіть за посиланням у листі для активації облікового запису.

2. Авторизація

Для входу на сайт ви можете скористатися двома способами:

Стандартна авторизація:

1. Натисніть кнопку "Увійти" у верхньому меню сайту.
2. Введіть вашу електронну пошту та пароль.
3. Натисніть кнопку "Увійти".

Авторизація через Google:

1. Натисніть кнопку "Увійти" у верхньому меню сайту.
2. Натисніть кнопку "Увійти через Google".
3. У відкритому вікні виберіть потрібний акаунт Google або введіть дані для входу.

3. Відновлення паролю

Якщо ви забули пароль:

1. Натисніть "Забули пароль?" на сторінці авторизації.
2. Введіть електронну пошту, пов'язану з вашим обліковим записом.
3. Натисніть кнопку "Відновити пароль".
4. На вашу електронну пошту буде надіслано лист із посиланням для створення нового паролю.
5. Перейдіть за посиланням та встановіть новий пароль.

4. Перегляд каталогу

Для перегляду доступного асортименту книг:

1. Перейдіть на головну сторінку сайту.
2. Натисніть на розділ "Каталог" у верхньому меню.
3. Виберіть категорію книг, яка вас цікавить (Художня література, Наукова література, Підручники тощо).
4. Перегляньте список доступних книг у вибраній категорії.

5. Пошук та фільтрація

Для пошуку потрібної книги:

1. Скористайтеся полем пошуку у верхній частині сайту.
2. Введіть назву книги, автора або ключове слово.
3. Натисніть кнопку "Пошук" або клавішу Enter.

Для фільтрації результатів:

1. На сторінці каталогу або результатів пошуку використовуйте панель фільтрів зліва.
2. Фільтрація доступна за такими параметрами:
    - Жанр
    - Автор
    - Видавництво
    - Мова видання
    - Ціна (мінімальна та максимальна)
    - Рік видання
3. Після вибору потрібних фільтрів натисніть кнопку .

6. Детальна інформація про книгу

Для перегляду детальної інформації про книгу:

1. Натисніть на зображення або назву книги в каталозі.
2. На сторінці книги ви знайдете:
    - Зображення обкладинки
    - Назву
    - Автора
    - Видавництво
    - Рік видання
    - Жанр
    - Кількість сторінок
    - Тип обкладинки
    - ISBN
    - Мову видання
    - Анотацію
    - Ціну та наявність знижки
    - Відгуки інших користувачів

7. Додавання до кошика

Щоб додати книгу до кошика:

1. На сторінці книги натисніть кнопку "Додати до кошика".
2. Вибрана книга буде додана до вашого кошика.
3. Повідомлення про успішне додавання з'явиться у верхній частині сторінки.
4. Кількість товарів у кошику буде відображатись на іконці кошика у верхньому меню.

8. Додавання до обраного

Для додавання книги до списку обраних:

1. На сторінці книги натисніть на іконку "Серце" біля назви книги.
2. Книга буде додана до вашого списку обраних товарів.
3. Доступ до списку обраних книг можна отримати через особистий кабінет.

9. Робота з кошиком

Для перегляду вмісту кошика:

1. Натисніть на іконку кошика у верхньому меню сайту.
2. У кошику ви побачите всі додані товари з вказаною кількістю та ціною.
3. Для зміни кількості книг використовуйте кнопки "+" та "-" біля кожної позиції.
4. Для видалення книги з кошика натисніть кнопку "Видалити" біля відповідної позиції.
5. Загальна сума замовлення буде відображатися внизу сторінки.

10. Промокоди

Для застосування промокоду:

1. На сторінці кошика знайдіть поле "Промокод".
2. Введіть код у відповідне поле.
3. Натисніть кнопку "Застосувати".
4. Якщо промокод дійсний, знижка буде автоматично застосована до загальної суми замовлення.
5. Інформація про застосований промокод та розмір знижки буде відображена під загальною сумою.

11. Вибір доставки

Для вибору способу доставки:

1. На сторінці оформлення замовлення виберіть бажаний спосіб доставки:
    - Поштові служби (Нова Пошта)
    - Самовивіз із пункту видачі
2. Залежно від вибраного способу, заповніть необхідні поля:
    - Для поштових служб: адреса, місто, номер телефону, пошту, ПІБ, номер відділення

12. Вибір оплати

Для вибору способу оплати:

1. На сторінці оформлення замовлення виберіть бажаний спосіб оплати:
    - Онлайн-оплата через LiqPay
    - Оплата при отриманні
2. У разі вибору LiqPay, після підтвердження замовлення ви будете перенаправлені на сторінку оплати.
3. У разі вибору оплати при отриманні, з вами зв'яжеться менеджер для підтвердження замовлення.

13. Особиста інформація

Для перегляду та редагування особистої інформації:

1. Увійдіть у свій обліковий запис.
2. Натисніть на своє ім'я у верхньому меню та виберіть "Особистий кабінет".
3. На вкладці "Особиста інформація" ви можете:
    - Переглянути та змінити своє ім'я або email
    - Змінити пароль

14. Управління обраним

Для управління списком обраних книг:

1. В особистому кабінеті перейдіть на вкладку "Обране".
2. На цій сторінці ви побачите всі книги, які ви додали до обраного.
3. Для видалення книги зі списку обраних натисніть кнопку "Видалити".
4. Для додавання книги з обраного до кошика натисніть кнопку "Додати до кошика".

Як змінити або скасувати замовлення?

- Для зміни або скасування замовлення зв'яжіться зі службою підтримки за номером, вказаним у розділі "Контакти".
- Зміни можливі лише для замовлень у статусі "Оформлено" або "Оплачено", якщо відправка ще не відбулась.

Чи можна повернути книгу?

- Так, книгу можна повернути протягом 14 днів з моменту отримання, якщо вона не має слідів використання.
- Для оформлення повернення зв'яжіться зі службою підтримки.

Як довго діють промокоди?

- Термін дії промокоду зазначений у його описі.
- Зазвичай промокоди діють на обмежений час або до вичерпання ліміту використань.

3. Тестування та єксперементи

3.1. Результати експериментів і тестування

Методологія тестування. Коли я розпочав тестування онлайн-магазину книг, то вирішив застосувати досить широкий підхід. Включили функціональні перевірки, тести на навантаження, перевірку безпеки та зручності використання. Головна ідея полягала в тому, аби переконатися - система справді відповідає технічним потребам і тому, чого очікують користувачі. Частину тестів автоматизували для регресійних перевірок, а от користувацький досвід перевіряли вручну. Намагалися створити тестове середовище максимально схоже на те, яке буде в реальній роботі.

Функціональне тестування.

1. **Як працює вхід і права доступу**

Система входу показала досить стабільні результати - JWT токени відповідають приблизно за 120 мілісекунд. Інтеграція з Google OAuth теж працює без проблем, хоча авторизація через Google займає близько 1,5 секунди. Регестрація за допомогою email та паролю виконується за 600мс.

2. **Каталог і пошук**

Пошукова система є досить точною - близько 97% результатів релевантні, навіть коли люди шукають звичайною мовою. При роботі з базою понад 5 тисяч книг пошукові запити обробляються максимум за 500мс мілісекунд завдяки пагінації на сайті. Фільтри працюють як треба по всіх параметрах.

3. **Процес замовлення**

Процес замовлення працює дуже швидку при натисканні на кошик користувача не перекидає на стронній сайт, а відкривається форма з заповненням необхідних даних,в залежності від вибраного користувачем способом доставки і оплати формується замовлення і форма закривається.Заголом користувач витрачає декілька хвилин на формування замовлення. Для здійснення оплати на сайті не обов’язково мати картку якогось окремого банку, платіжна система працює з усіма банками, також є можливіст здійснити олату за допомогою google або apple pay.

Перевірка безпеки

1. **Тести на стійкість до атак**

Перевіряли основні загрози такі як: SQL-ін'єкції, XSS, CSRF, підбір паролів. Система витримала всі спроби. Rate limiting спрацьовує як треба - після сотні запитів за хвилину IP блокується на певний час.

2. **Шифрування та захист даних**

Вся критична інформація йде через HTTPS з TLS 1.3. Паролі хешуються через bcrypt з рівнем складності 12.Я викоритовую для авторизації JWT токени, access та refresh ,такий підхід забезпечує максимальну надійність та безпеку, завдяки тому що access зберігається на клієнті з таймером в 30 хвилин, а refresh токен зберігається в кукі протягом неділі з флагом httpOnly, що забезпечує максимальну надійність та безпеку.

Unit тести

1. **Тестування модуля книг**

Для модуля книг написали повний набір unit тестів, які покривають всі основні сценарії роботи. Тестуємо створення, редагування, видалення книг, а також всі валідації даних. Покриття коду становить 97% - пропустили лише кілька edge cases, які практично неможливі в реальних умовах.

Особливо ретельно протестували логіку пошуку книг. Перевірили роботу різними типами запитів: пошук за назвою, автором, ISBN, категорією. Також протестували складні сценарії - коли користувач вводить частково неправильні дані або використовує спеціальні символи. Unit тести допомогли виявити кілька помилок у логіці, які могли б проявитися тільки при специфічних запитах.

Тестування CRUD операцій показало стабільну роботу всіх методів.Створення нової книги проходить валідацію на всіх рівнях - перевіряємо обов'язкові поля, формат ISBN, коректність цін. Оновлення книги правильно обробляє часткові зміни

і зберігає цілісність даних. Видалення працює з урахуванням зв'язків - якщо книга є в чиїйсь корзині або замовленні, система блокує операцію.

3. Валідація та бізнес-логіка

Unit тести для валідації охоплюють всі можливі сценарії некоректних даних. Перевіряємо роботу з порожніми полями, надто довгими рядками, неправильними форматами дат та цін. Особливу увагу приділили тестуванню ISBN - система правильно розпізнає як старий 10-значний, так і новий 13-значний формати. Бізнес-логіка для розрахунку знижок та акційних цін також покрита тестами. Перевіряємо правильність застосування різних типів знижок, їх комбінацій, обмежень по датах. Unit тести допомогли знайти помилку в логіці накопичувальних знижок, яка могла б коштувати чималих грошей.

Тестування через postman

Postman використовував для комплексного тестування всіх методів сайту. Створив детальні колекції для кожного модуля - книги, користувачі. В результаті протестував всі endpoints системи з різними сценаріями використання.

1. **Модуль книг**

Почав з базових операцій над книгами. Запит GET /books для отримання каталогу книг на головну сторінку, там є три типи: бестселери, нещодавно додані(інтервал 7 днів), зі знижкою. Запит GET /books/:id повертає повну інформацію про книгу . Запит GET /books/category/:name повертає всі книги заданої категорії, також можна додати фільтрацію завдяки query параметрам, наприклад: author, price , genre , language, publisher, publicationYear, new. Запити виконуються досить швидку, навіть якщо користувач одразу обере всі фільтри, якщо жодної книги не знайдено повертається пустий масив. Запити PUT /books/:id виконує оновлення книги, DELETE /books/:id виконує видалення книги, доступ до цих запитів є тільки у адміністратору магазину, перевірка відбувається шляхом порівняння ролі користувача. Якщо запит виконує не адмістратор сайту - повертається помилка з кодом 401 та повідомдення “You do not have the necessary permission.” , якщо книги з таким айді не існує в базі даних - повертається помилка з кодом 404 та

повідомлення “Book doesn't exist.” . Запит POST /books для створення нових книг, доступ до нього має лише адміністратор, для забезпечення валадноісті даних викоритовується Data transfer object, він перевірає чи співпадає тип даних відправлений з клієнта з типом на сервері, якщо ні - повертає відповідну помилку. Обов’язковою умовою для створення книги є фото обкладинки, з клієнта передається в бінарному форматі, для збереження на сервері використовується сервіс S3 Amazon, який повертає посилання на зображення. Оброка помилок відбувається наступним чином: назва книги вже присутня в базі даних - повертається помилка з кодом 403 та повідомленням “Book title already exists, please select another one.” , запит виконує не адмістратор сайту - повертається помилка з кодом 401 та повідомдення “You do not have the necessary permission.” . Запит GET /books/search відповідає за пошук книг на сайті, коли коритувач вводить текс в пошукову стрічку відправляється запит до сервера, пошук відбувається по назві книги та ПІБ автора, пошук налаштований так, що достатньо ввести літери , які є в ПІБ автора, або в назві книги. Для оптимізації пошука на поля по яким здійснюється пошук додані btree індекси - це значно прискорює пошук та прискорює роботу сайту. Також можна додавати фільтри для уже знайдених книг. Якщо жодної книги не знайдено повертається порожній масив. Запит POST / books/:id/favorite надає змогу користовачу додавати книги в обране, так само запит POST /:id/unfavorite видаляти книги з обраного. Обробка помилок відбувається наступним чином: запит виконує не адмістратор сайту - повертається помилка з кодом 401 та повідомдення “You do not have the necessary permission.” . Запит GET /books/liked/all повертає всі обрані книги користувача.

2. Модуль користувача

Модуль користувачів забезпечує повний цикл управління обліковими записами у системі. Запит POST /users/register виконує реєстрацію нових користувачів у системі, для забезпечення валідності даних використовується Data transfer object (CreateUserDto), який перевіряє чи співпадає тип даних відправлений з клієнта з типом на сервері, якщо ні - повертає відповідну помилку. Обробка помилок

відбувається наступним чином: паролі не співпадають при реєстрацію - повертається помилка з кодом 422 та повідомленням “Password didn't match”. Користувач в системі вже існує з таким ім’ям чи поштою. Запит POST /confirm-email відповідає за підтвердження електронної пошти під час реєстрації. Користувач отримує на пошту токен підтвердження, який необхідно передати в цьому запиті. Якщо токен недійсний або прострочений - повертається помилка з кодом 403 та повідомленням “Invalid confirmation token”. Запит POST /users/login забезпечує автентифікацію користувачів у системі, використовується validation з LoginUserDto для перевірки введених даних. При успішній автентифікації повертаються токени access , a refresh записується в cookie та об’єкт користувача. Обробка помилок: неправильний пароль - повертається помилка з кодом 422 та повідомленням "User is unfound", акаунт не підтверджений - повертається помилка з кодом 403 та повідомленням "Email is not confirmed". Запит POST /users/logout виконує вихід користувача з системи. При виході токени користувача стають недійсними. Якщо користувач не автентифікований - повертається помилка з кодом 401 та повідомленням “Not authorized”. Запит POST /users/refresh відновлює access токен за допомогою refresh токена. Це дозволяє підтримувати сесію користувача активною без повторного введення облікових даних. Якщо refresh токен недійсний - повертається помилка з кодом 403 та повідомленням “Forbiden”, якщо токена немає в бд, тоді код 401 з повідомленням “Not authorized”. Запит GET /users повертає повну інформацію користувача, налаштування та іншу персональну інформацію. Якщо користувач не автентифікований – повертається помилка з кодом 401 та повідомленням “Not authorized”. Запит PUT /users виконує оновлення даних користувача. Використовується validation з UpdateUserDto для забезпечення валідності оновлених даних. Обробка помилок: користувач не автентифікований - повертається помилка з кодом 401 та повідомленням “Not authorized”, якщо некоректні дані для оновлення - повертається відповідна помилка валідації.Запит DELETE /users виконує видалення облікового запису користувача. Користувач може видалити лише свій власний обліковий запис. При видаленні всі пов'язані дані

користувача також видаляються з системи. Якщо користувач не автентифікований – повертається помилка з кодом 401 та повідомленням “ Not authorized”. Запити POST /users/request-password-reset та POST /reset-password забезпечують функціональність відновлення паролю. Перший запит ініціює процес відновлення, використовуючи validation з PasswordResetRequestDTO, і відправляє на електронну пошту користувача токен для скидання паролю. Другий запит виконує фактичну зміну паролю за допомогою отриманого токена, використовуючи validation з PasswordResetDTO. Обробка помилок: email не знайдений в системі - повертається помилка з кодом 422 та повідомленням "User not found.", недійсний або прострочений токен - повертається помилка з кодом 403 та повідомленням “Invalid or expired token”. Система також підтримує автентифікацію через Google OAuth. Запит GET /users/google ініціює процес автентифікації через Google, перенаправляючи користувача на сторінку авторизації Google з необхідними правами доступу до профілю та електронної пошти. Запит GET /users/google/callback обробляє відповідь від Google після успішної автентифікації. Запит POST /success-google-auth завершує процес автентифікації через Google, використовуючи validation з CreateUserGoogleDto для обробки даних отриманих від Google. При першому вході користувача через Google автоматично створюється обліковий запис у системі. Обробка помилок: ім’я коритувача в системі не існує, повертається з кодом 422 та повідомлення “Name is taken”. Всі операції з користувачами логуються для забезпечення безпеки та можливості аудиту. Паролі зберігаються у зашифрованому вигляді з використанням алгоритму argon2 . Система також включає захист від брутфорс атак та інших загроз безпеки.

3. Модуль користувача

Модуль коментарів забезпечує повну функціональність для управління коментарями до книг. Запит POST /comments/:bookId створює новий коментар для конкретної книги. Для забезпечення валідності даних використовується validation з CommentDto, який перевіряє чи співпадає тип даних відправлений з клієнта з типом на сервері, якщо ні - повертає відповідну помилку. Обробка помилок: користувач не

автентифікований - повертається помилка з кодом 401 та повідомленням “Not authorized”, книга з таким ID не існує - повертається помилка з кодом 404 та повідомленням “Book doesn`t exist.” Запит GET /comments повертає всі коментарі у системі. Коментарі повертатюся з пагінацією та сортуванням для оптимізації завантаження. Якщо користувач не автентифікований - повертається помилка з кодом 401 та повідомленням “Not authorized”. Запит PUT /comments/:id виконує оновлення існуючого коментаря. Користувач може редагувати лише свої власні коментарі. Використовується validation з CommentDto для перевірки оновлених даних. Обробка помилок: користувач не автентифікований - повертається помилка з кодом 401 та повідомленням "Authentication required.", коментар з таким ID не існує - повертається помилка з кодом 404 та повідомленням “Comment doesn't exist.”, користувач намагається редагувати чужий коментар - повертається помилка з кодом 403 та повідомленням "You aren't authhor this comment". Запит DELETE /comments/:id виконує видалення коментаря. Користувач може видаляти лише свої власні коментарі, адміністратори мають право видаляти будь-які коментарі. При видаленні коментаря також видаляються всі відповіді до нього. Обробка помилок: користувач не автентифікований - повертається помилка з кодом 401 та повідомленням “Not authorized”. коментар з таким ID не існує - повертається помилка з кодом 404 та повідомленням “Comment doesn't exist.”, користувач намагається видалити чужий коментар без відповідних прав - повертається помилка з кодом 403 та повідомленням “You aren't author this comment.” Запит POST /comments/:id/add-reply/:bookId додає відповідь до існуючого коментаря, створюючи ієрархічну структуру обговорення. Використовується validation з CommentDto для перевірки даних відповіді. Параметр bookId необхідний для прив'язки відповіді до контексту книги. Обробка помилок: користувач не автентифікований - повертається помилка з кодом 401 та повідомленням “Not authorized” , батьківський коментар не існує - повертається помилка з кодом 404 та повідомленням “Comment doesn't exist.”, книга не існує - повертається помилка з кодом 404 та повідомленням “Book doesn't exist.” .Запит POST /comments/:id/favorite дозволяє користувачу ставити вподобайку

на коментарі та /:id/unfavorite прибирає вподобайку з коментаря. Обробка помилок для обох запитів: користувач не автентифікований - повертається помилка з кодом 401 та повідомленням “Not authorized” , коментар з таким ID не існує - повертається помилка з кодом 404 та повідомленням “Comment doesn’t exist”.

Система коментарів підтримує модерацію контенту та захист від спаму. Неприйеятні коментарі можуть бути видалені адміністратором. Всі операції з коментарями логуються для забезпечення прозорості та можливості аудиту взаємодій користувачів.

4. Модуль замовлень

Модуль замовлень забезпечує обробки покупок користувачів від створення замовлення до його підтвердження та управління. Запит POST /orders/checkout створює нове замовлення в системі. Для забезпечення валідності даних використовується validation з CreateOrderDto, який перевіряє чи співпадає тип даних відправлений з клієнта з типом на сервері, якщо ні - повертає відповідну помилку. При створенні замовлення система перевіряє наявність товарів, розраховує загальну вартість з урахуванням знижок, некоректні дані замовлення - повертається відповідна помилка валідації. Запит GET /orders повертає всі замовлення в системі, доступ має лише адміністратор магазину, перевірка відбувається через authMiddleware та chechRoleGuard. Звичайні користувачі не можуть переглядати чужі замовлення з міркувань приватності та безпеки. Замовлення повертаються з повною інформацією включаючи статус, дату створення, список товарів та деталі оплати. Якщо запит виконує не адміністратор сайту - повертається помилка з кодом 401 та повідомленням “Not authorized”. Запит PUT /orders/:id виконує оновлення замовлення. Використовується validation з UpdateBookDto для перевірки оновлених даних замовлення. Адміністратор може змінювати статус замовлення, додавати примітки, оновлювати інформацію про доставку. Обробка помилок: запит виконує не адміністратор сайту - повертається помилка з кодом 401 та повідомленням “Not authorized”, замовлення з таким ID не існує - повертається помилка з кодом 404 та повідомленням "Order not found.", некоректні дані для оновлення - повертається

відповідна помилка валідації. Запит DELETE /orders/:id виконує видалення замовлення. Видалення замовлення може бути необхідним у випадках скасування, помилкового створення або інших адміністративних потреб. При видаленні замовлення також оновлюється інформація про наявність товарів на складі. Обробка помилок: запит виконує не адміністратор сайту - повертається помилка з кодом 401 та повідомленням “Not authorized” , замовлення з таким ID не існує - повертається помилка з кодом 404 та повідомленням "Order not found." Запит POST /orders/confirm/:token підтверджує замовлення за допомогою унікального токена підтвердження uuid. Цей токен генерується при створенні замовлення та відправляється на пошту адміністратора. Підтвердження замовлення змінює його статус. Обробка помилок: токен підтвердження недійсний - повертається помилка з кодом 403 та повідомленням “Invalid confirmation token”. Система замовлень інтегрована з платіжними системами для обробки онлайн-платежів та автоматично генерує чеки після успішного завершення транзакції. Всі операції з замовленнями логуються для забезпечення прозорості, відстеження статусу та можливості аудиту фінансових операцій.

5. Модуль оплати онлайн

Модуль платежів забезпечує безпечну обробку фінансових транзакцій та інтеграцію з платіжною системою LiqPay для здійснення онлайн-платежів. Запит GET /payments/pay/:orderId генерує платіжну форму для конкретного замовлення, створюючи необхідні параметри для ініціації платежу через платіжну систему. При генерації форми система перевіряє існування замовлення. Формується унікальний платіжний запит з усіма необхідними параметрами включаючи суму, валюту, опис товарів та callback URL користувач отримує готову html сторінку для оплати замовлення. Обробка помилок відбувається наступним чином: замовлення з таким ID не існує - повертається помилка з кодом 404 та повідомленням "Order doesn't exist ."". Запит POST /payments/handle-webhook обробляє webhook сповіщення від платіжної системи LiqPay про зміну статусу платежів. Цей ендпоінт отримує автоматичні сповіщення про успішні платежі, відхилені транзакції та інші статуси

оплати. При отриманні webhook система перевіряє підпис запиту для забезпечення автентичності даних, декодує отримані параметри та оновлює статус відповідного замовлення в базі даних. Успішна обробка webhook ініціює автоматичну відправку підтвердження оплати користувачу та запускає процес виконання замовлення. Обробка помилок: якщо підпис webhook не збігається на сервері оновлення статусу замолення не відбудеться.

6. Модуль промокодів

Модуль промокодів забезпечує створення та управління системою знижок для стимулювання продажів та лояльності клієнтів. Запит POST /promocodes/ створює новий промокод у системі. Для забезпечення валідності даних використовується validation з CreatePromoCodeDto, який перевіряє чи співпадає тип даних відправлений з клієнта з типом на сервері, якщо ні - повертає відповідну помилку. При створенні промокоду система генерує унікальний код, встановлює параметри знижки, терміни дії та умови використання. Обробка помилок відбувається наступним чином: запит виконує не адміністратор сайту - повертається помилка з кодом 401 та повідомленням "You do not have the necessary permission.", промокод з такою назвою вже існує - повертається помилка з кодом 409 та повідомленням “Promo code is taken”. Запит GET /promocodes повертає список всіх промокодів у системі, доступний тільки для адмістратора. Промокоди повертаються з основною інформацією включаючи назву, розмір знижки та термін дії. Якщо в системі немає активних промокодів - повертається порожній масив.Запит PUT /:id виконує оновлення існуючого промокоду, доступ має лише адміністратор .Використовується validation з UpdatePromoCodeDto для перевірки оновлених даних. Адміністратор може змінювати параметри знижки, продовжувати термін дії, встановлювати лімити використання та модифікувати умови застосування. Обробка помилок: запит виконує не адміністратор сайту - повертається помилка з кодом 401 та повідомленням “You do not have the necessary permission.”, промокод з таким ID не існує - повертається помилка з кодом 404 та повідомленням “Promo code doesn’t exist .”. Запит DELETE /:id виконує видалення промокоду, доступ має лише

адміністратор магазину . При видаленні система перевіряє чи не використовується промокод в активних замовленнях. Обробка помилок: запит виконує не адміністратор сайту - повертається помилка з кодом 401 та повідомленням “You do not have the necessary permission.”, промокод з таким ID не існує - повертається помилка з кодом 404 та повідомленням “Promo code doesn’t exist.”. Запит POST /check-promo-code перевіряє валідність та застосовність промокоду при оформленні замовлення. Використовується validation з CheckPromoCode для перевірки формату коду. Обробка помилок: промокод не існує або неактивний - повертається помилка з кодом 403 та повідомленням “Promo code is unvalid.”, промокод прострочений - повертається помилка з кодом 403 та повідомленням “Promo code has expired” досягнуто ліміт використання - повертається помилка з кодом 403 та повідомленням “Minimum order amount for this promotional code:” . Система промокодів підтримує різні типи знижок включаючи фіксовані суми, відсоткові знижки та спеціальні пропозиції. Всі операції з промокодами логуються для аналізу ефективності маркетингових кампаній та відстеження використання.

7. Модулі: автори, жанри, мови, категорії, видавництва

Ці модулі використовуються в панелі для адміністраторів, Мають однокові методи: POST / для створення запису в таблиці, PUT /:id для оновлення , DELETE /:id для видалення, GET /:id для отримання одного запису, GET / для отримання всіх. До цих методів доступ має тільки адміністратор. Обробка помилок відбувається наступним чином: запист з такою назвою вже існує в базі даних - повертається помилка з кодом 403 та повідомленням “name is taken”, якщо запису з такім id не існує – повертається помилка з кодом 404 та повідомленням “doesn’t exist”.

3.2. Порівняння з існуючими рішеннями

1. Аналіз конкурентів

Для порівняння було обрано наступні популярні інтернет-магазини книг: Books24

Yakaboo.

2. Конкурентні переваги

Переваги нашого рішення:

1. **Локалізація**: Повна підтримка української мови та культурних особливостей
2. **Ціноутворення**: Конкурентні ціни та система знижок
3. **Доставка**: Швидка доставка по Україні (1-2 дні), безоплатна від 800 грн.
4. **Підтримка**: Цілодобова підтримка українською мовою

Унікальні функції:

- Попередній перегляд перших розділів
- Інтеграція з українськими видавництвами
- Програма лояльності з накопичувальними балами

3. Області для покращення

На основі порівняльного аналізу виявлено наступні напрямки для розвитку:

1. **Розширення асортименту**: Збільшення кількості книг
2. **Рекомендації**: Впровадження алгоритмів рекомендацій
3. **Мобільний додаток**: Розробка нативних додатків для смартфони
4. **Міжнародна доставка**: Розширення географії доставки
5. **Інтеграція з е-книгами**: Додавання цифрових версій книг

4. Висновки тестування

Проведене тестування показало, що розроблений інтернет-магазин книг:

1. **Відповідає технічним вимогам**: Всі ключові функції працюють стабільно
2. **Забезпечує хороший користувацький досвід**: Високі оцінки юзабіліті
3. **Конкурентоспроможний**: За багатьма показниками не поступається лідерам ринку
4. **Має потенціал для розвитку**: Виявлені чіткі напрямки для покращення

ВИСНОВКИ

1. Підсумки роботи

У рамках дипломного проекту нами було успішно розроблено та впроваджено інтернет-магазин "Chapter One", який повністю відповідає поставленим завданням. Процес розробки включав детальний аналіз вимог, проектування архітектури системи, створення бази даних та реалізацію всіх необхідних модулів.

Під час розробки проекту було успішно виконано такі завдання:

1. Проведено аналіз ринку продажу книг в Україні та вивчено роботу існуючих інтернет-магазинів книг для визначення їхніх переваг та недоліків.
2. Створено базу даних на основі PostgreSQL, що включає 12 взаємопов'язаних таблиць (Users, Books, Languages, Publishers, Reset_passwords, Comments, Orders, Refresh_sessions, Genres, Categories, Authors), які забезпечують ефективне зберігання та доступ до всієї необхідної інформації.
3. Розроблено серверну частину (back-end) з використанням Node.js та Express.js, що забезпечує надійну роботу всіх компонентів системи.Клієнтська частина сайта написана на React.
4. Реалізовано модульну структуру проекту, що включає: модуль користувача, адмін-панель, модуль оформлення замовлень, модуль книг, модуль коментарів, модуль оплати та модуль промокодів.
5. Впроваджено систему авторизації з використанням JWT-токенів та Google OAuth, що підвищує безпеку та спрощує доступ користувачів до системи.
6. Інтегровано платіжну систему LiqPay, яка забезпечує зручний та безпечний спосіб оплати онлайн.
7. Розроблено інтуїтивно зрозумілий інтерфейс користувача.

Розроблений веб-сайт "Chapter One" дозволяє користувачам легко знаходити та купувати книги, залишати відгуки, відстежувати статус замовлень та взаємодіяти з іншими читачами. Для адміністраторів система надає потужні інструменти управління каталогом товарів, обробки замовлень та взаємодії з користувачами.

2. Можливі шляхи покращення проекту

На основі аналізу функціональності розробленої системи та сучасних тенденцій розвитку електронної комерції, ми визначили такі напрямки подальшого вдосконалення проекту "Chapter One":

1. Впровадження системи персоналізованих рекомендацій - розробка алгоритмів, які аналізуватимуть поведінку користувачів (історію переглядів, покупок, вподобань) та пропонуватимуть книги, які можуть їх зацікавити. Це збільшить середній чек та підвищить задоволеність користувачів.
2. Розробка мобільних додатків - створення нативних мобільних додатків для iOS та Android, що забезпечить більш зручний досвід користування та збільшить залученість аудиторії.
3. Програма лояльності - розробка системи накопичення балів за покупки, відгуки та активність на сайті з можливістю їх використання для отримання знижок, спеціальних пропозицій або ексклюзивного контенту.
4. Розширення асортименту цифрового контенту - додавання електронних книг та аудіокниг з відповідними системами захисту контенту та онлайн-читання/прослуховування.
5. Розширені інструменти аналітики - впровадження детальних звітів та інформаційних панелей для аналізу продажів, поведінки користувачів та ефективності маркетингових кампаній.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

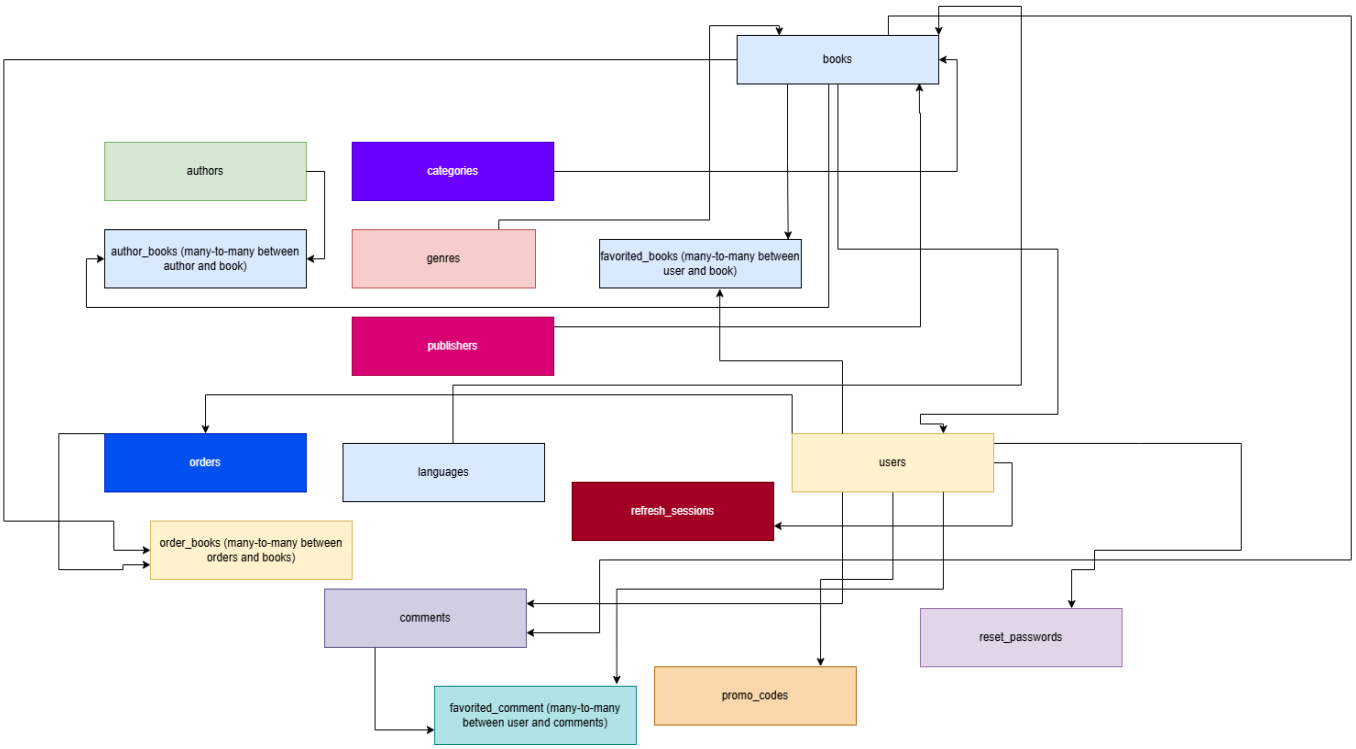
1. Node.js documentation URL: https://nodejs.org/docs/latest/api/
2. Express framework documentation URL: https://expressjs.com/
3. Type orm documentation URL: https://typeorm.io/
4. React framework documentation URL: https://react.dev/reference/react
5. Postgres documentation URL: https://www.postgresql.org/docs/
6. Кантор, Ілья. Сучасний туторіал JavaScript. URL: https://javascript.info/
7. Gomez, Felipe. Database Query Optimization: Advanced Strategies and Techniques. URL: https://medium.com/@EuFelipegomes__/database-query-optimization-advanced-strategies-and-techniques-12c11e537d63
8. Ors, Mikel. Clean Code — A practical approach.URL: https://medium.com/clarityai-engineering/clean-code-a-practical-approach-896546435235
9. Mohamed, Ibrahim. Best Practices for Structuring an Express.js Project. URL: https://dev.to/moibra/best-practices-for-structuring-an-expressjs-project-148i
10. Derakhshan, Hossein. Things that make you far better Node.js developer (part 1). URL: https://medium.com/@derakhshanfar.hossein/things-that-make-you-far-better-node-js-developer-part-1-60f585323bfc
11. Lauret, Arnauld. The Design of Web APIs. [c.4-157]
12. Браун, Ітан. Веб-розробка із застосуванням Node і Express. Повноцінне використання стека JavaScript. 2-е видання.[с.10-340]
13. Кархунова Катя. Не всі книги змінюють життя, але наші — змінюють: інтерв'ю з власником інтернет-магазину “У Фрейда”. URL: https://horoshop.ua/ua/blog/intervu-s-vladeltsem-internet-magazina-u-freyda/
14. Денисюк Микола. Як відкрити свій книжковий інтернет-магазин в Україні з нуля. URL: https://pro.lviv.ua/yak-vidkryty-sviy-knyzhkovyy-internet-mahazyn-v-ukraini-z-nulya

15. Georgiev Petar. How to setup automatic migrations with TypeORM on deployment. URL: https://medium.com/@peturgeorgievv/how-to-setup-automatic-migrations-with-typeorm-on-deployment-2224544a4838

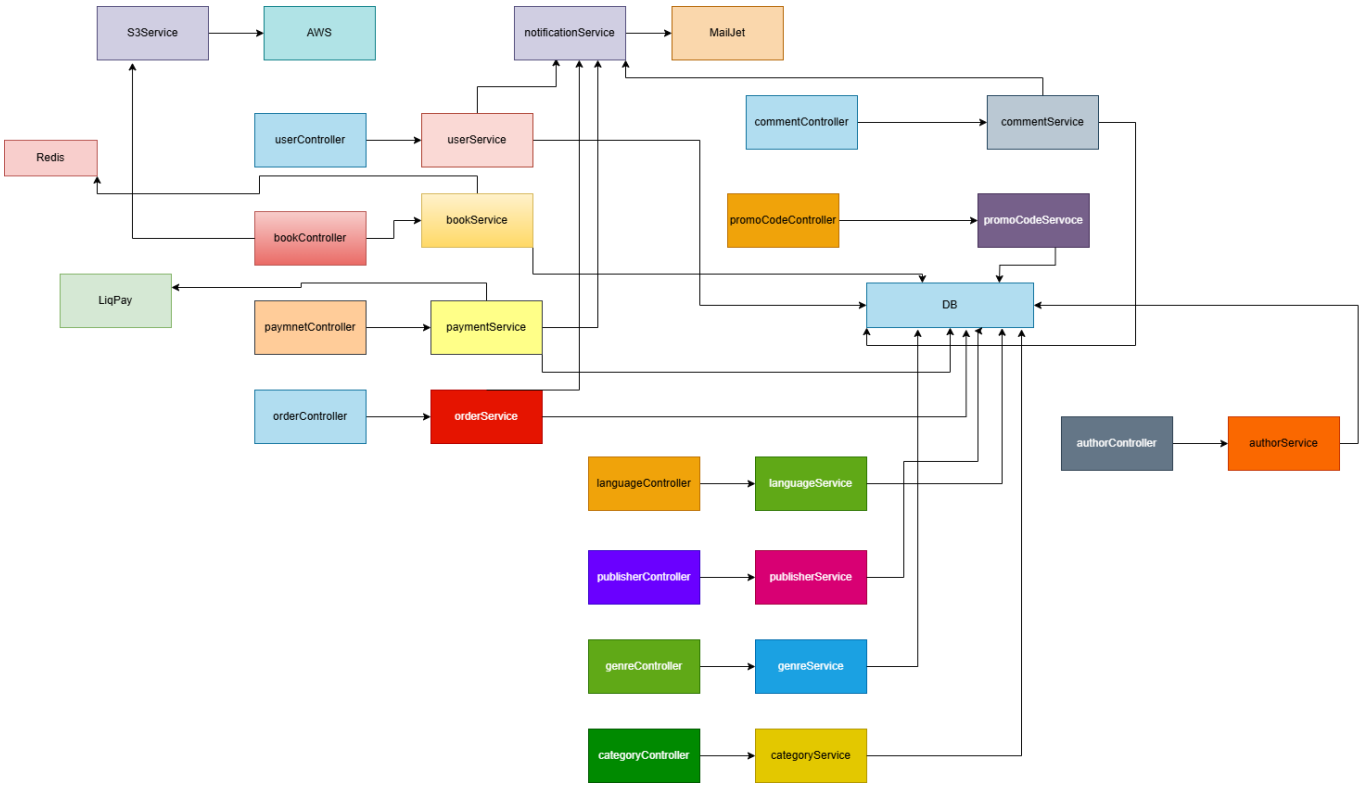
ДОДАТКИ

Додаток 1. UML діаграми

1.1. База даних



1.2. Модулі серверна частина



Додаток 2. Приклад коду

2.1 Конфігурація бази даних

```
export const dataSource = new DataSource({
  type: 'postgres',
  host: process.env.DATABASE_HOST,
  port: Number(process.env.DATABASE_PORT),
  username: process.env.DATABASE_USERNAME,
  password: process.env.DATABASE_PASSWORD,
  database: process.env.DATABASE_NAME,
  schema: process.env.DATABASE_SHEMA,
  entities: [BookEntity, RefreshSessionEntity, UserEntity, CommentEntity, LanguageEntity, CategoryEntity, Publis
  logging: false,
  synchronize: true,
});
```

2.2. Перевірка авторизації користувача

```
export async function authMiddleware(req: ExpressRequest, res: Response, next: NextFunction) {
  const accessToken = req.headers.authorization;

  if (!accessToken) {
    req.user = null;

    return next();
  }

  try {
    const token = accessToken.split(' ')?.[1];
    const decodedAccessToken = verify(token, process.env.SECRET_PHRASE_ACCESS_TOKEN) as JwtPayload;
    const id = decodedAccessToken.id as string;
    req.user = await userRepository.findOneBy({ id });
    next();
  } catch (error) {
    req.user = null;
    next();
  }
}
```

```
export async function authGuard(req: ExpressRequest, res: Response, next: NextFunction) {
  if (req.user) return next();

  res.status(401).json({ message: 'Not authorized' });
}
```

2.3 Використання планувальника завдань

```
cron.schedule('0 0 * * *', async () => {
  const expirationDate = new Date();
  expirationDate.setMinutes(expirationDate.getMinutes() - 15);

  const usersToDelete = await userRepository.find({
    where: {
      isConfirmed: false,
      createdAt: LessThan(expirationDate),
    },
  });

  if (usersToDelete.length > 0) {
    await userRepository.remove(usersToDelete);
  }
});

cron.schedule('0 0 * * *', async () => {
  const expirationDate = new Date();
  expirationDate.setDate(expirationDate.getDate() - 1);

  const ordersToDelete = await orderRepository.find({
    where: {
      status: 'pending',
      createdAt: LessThan(expirationDate),
    },
  });

  if (ordersToDelete.length > 0) {
    await orderRepository.remove(ordersToDelete);
  }
```

2.4. Data transfer object для створення користувача

```
import { IsEmail, IsNotEmpty, IsString } from 'class-validator';

export class CreateUserDto {
  @IsString()
  @IsNotEmpty()
  username: string;

  @IsEmail({}, { message: 'Email must be a valid.' })
  email: string;

  @IsString()
  @IsNotEmpty()
  password: string;

  @IsString()
  @IsNotEmpty()
  confirmedPassword: string;
}

export class CreateUserGoogleDto {
  @IsString()
  @IsNotEmpty()
  username: string;

  @IsString()
  @IsNotEmpty()
  token: string;
}
```

2.5. Опис сутності користувача в базі даних

```
@Entity({ name: 'comments' })
export class CommentEntity {
  @PrimaryGeneratedColumn('uuid')
  id: string;

  @Column()
  text: string;

  @Column({ default: 0, name: 'favorites_count' })
  favoritesCount: number;

  @CreateDateColumn({ name: 'created_at' })
  @Index('created_at_index')
  createdAt: Date;

  @UpdateDateColumn({ name: 'updated_at' })
  updateAt: Date;

  @ManyToOne(() => CommentEntity, { nullable: true, onDelete: 'CASCADE' })
  @JoinColumn({ name: 'parent_comment' })
  parentComment: CommentEntity;

  @ManyToOne(() => UserEntity, (user) => user.comments, { onDelete: 'CASCADE' })
  user: UserEntity;

  @ManyToOne(() => BookEntity, (book) => book.comments, { onDelete: 'CASCADE' })
  book: BookEntity;
}
```

2.6. Приклад створення та логування користувача(текст)

```
export class UserController {
  constructor(private readonly userService: UserService) {}

  @Log({ body: true })
  async createUser(req: Request, res: Response, next: NextFunction) {
    const fingerprint = req.fingerprint.hash;
    const createUserDto = req.body;

    await this.userService.createUser(createUserDto, fingerprint);
```

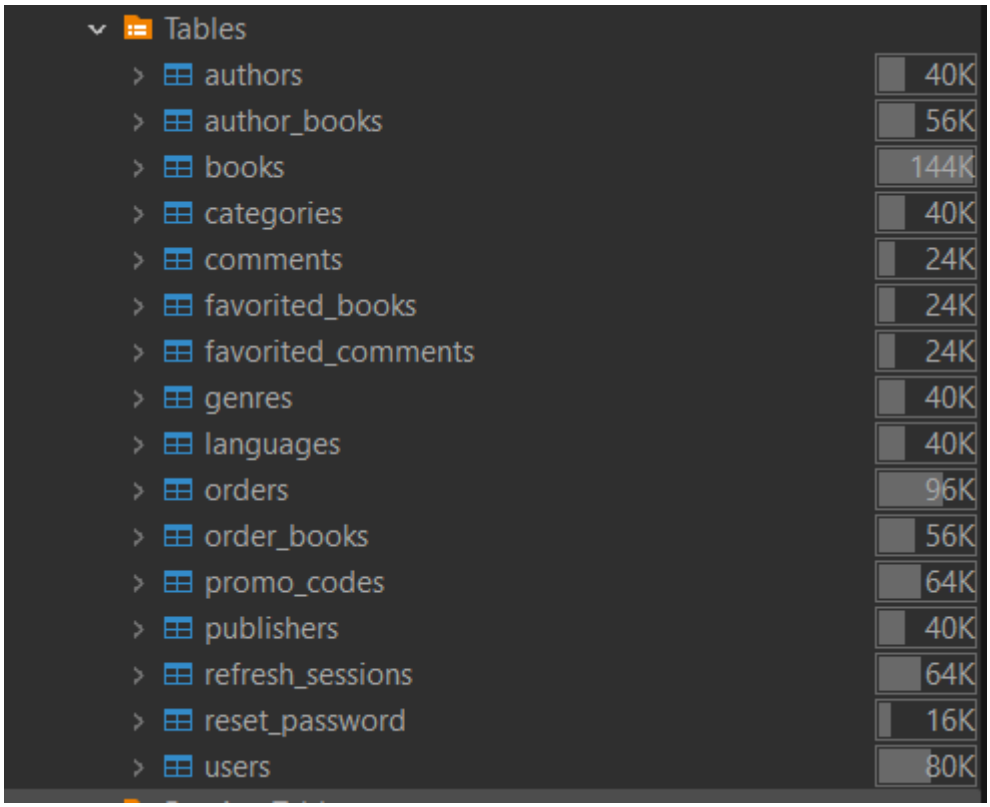
```
    res.status(201).json({ message: 'User registered. Please check your email for the confirmation code.' });
  }

  @Log({ body: true })
  async loginUser(req: Request, res: Response, next: NextFunction) {
    const fingerprint = req.fingerprint.hash;
    const loginUserDto = req.body;

    const { user, refreshToken } = await this.userService.loginUser(loginUserDto, fingerprint);
    const userResponse = this.userService.buildUserResponse(user);

    res.cookie('refreshToken', refreshToken, {
      httpOnly: true,
      sameSite: 'strict',
      maxAge: Number(process.env.REFRESH_TOKEN_EXPIRATION_15DAYS),
    });

    res.status(200).json(userResponse);
  }}
export class UserService {
  constructor(
    private readonly userRepository: Repository<UserEntity>,
```

```
    private readonly refreshSessionRepository: Repository<RefreshSessionEntity>,
    private readonly resetPasswordRepository: Repository<ResetPasswordEntity>,
    private readonly notificationService: NotificationService,
  ) {}

  async createUser(createUserDto: CreateUserDto, fingerprint: string): Promise<void> {
    if (createUserDto.confirmedPassword !== createUserDto.password) throw new
CustomError(422, "Password didn't match");

    const userByEmail = await this.userRepository.findOneBy({
      email: createUserDto.email,
    });

    const userByName = await this.userRepository.findOneBy({
      username: createUserDto.username,
    });

    if (userByEmail || userByName) throw new CustomError(422, 'Name or email is taken');

    const newUser = new UserEntity();
    Object.assign(newUser, createUserDto);

    const token = uuidv4();
```

```
    newUser.confirmationToken = token;
    newUser.password = await hash(createUserDto.password, 10);

    const user = await this.userRepository.save(newUser);
    const refreshToken = this.generateRefreshToken(user);

    await this.refreshSessionRepository.save({
      fingerprint,
      refreshToken,
      user,
    });

    await this.notificationService.sendVerificationEmail(user.email,
user.confirmationToken);
  }

  async loginUser(loginUserDto: LoginUserDto, fingerprint: string):
Promise<CreateUserResponse> {
    const user = await this.userRepository.findOne({
      where: { email: loginUserDto.email },
      select: ['id', 'username', 'password', 'email', 'role', 'isConfirmed'],
    });

    if (user.isConfirmed === false) throw new CustomError(403, 'Email is not confirmed');
```

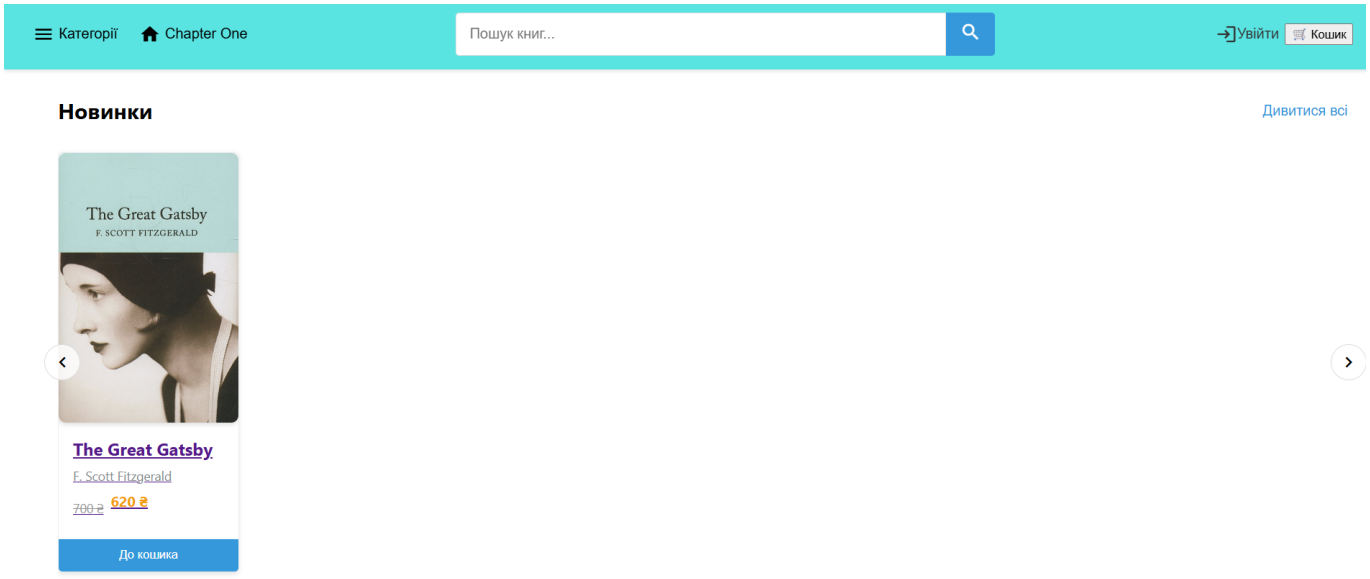
```
    if (!user) throw new CustomError(422, 'User is unfound');

    const isPassword = await compare(loginUserDto.password, user.password);

    if (!isPassword) throw new CustomError(422, 'Password is uncorrect');

    const refreshToken = this.generateRefreshToken(user);

    await this.refreshSessionRepository.save({
      fingerprint,
      refreshToken,
      user,
    });

    delete user.password;
    return { user, refreshToken };
  }
}
```

3.Таблиці в базі даних



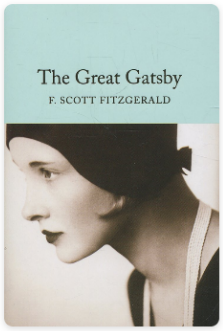
5. Інтерфейс додатку

5.1. Основна сторінка



5.2. Інформація про конкретну книгу

Категорії Chapter One Пошук книг...



The Great Gatsby

Автор: F. Scott Fitzgerald

Видавництво: Scribner

Жанр: Fiction

Категорія: Classic Literature

Мова: English

Рік публікації: 1925

ISBN: 978-3-16-148410-0

Кількість сторінок: 180

Опис: A novel set in the Jazz Age, exploring themes of wealth, love, and the American Dream.

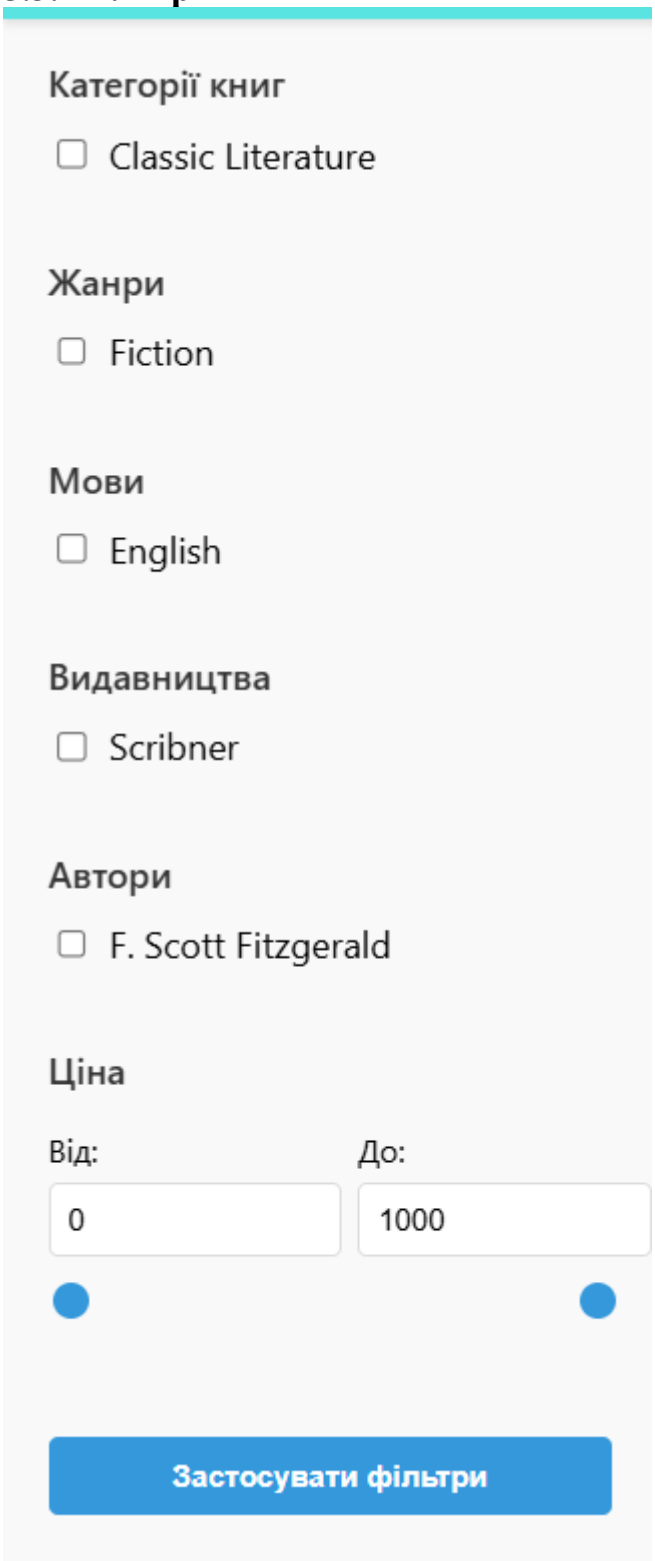
~~700 ₴~~ 620.00 ₴

До кошика

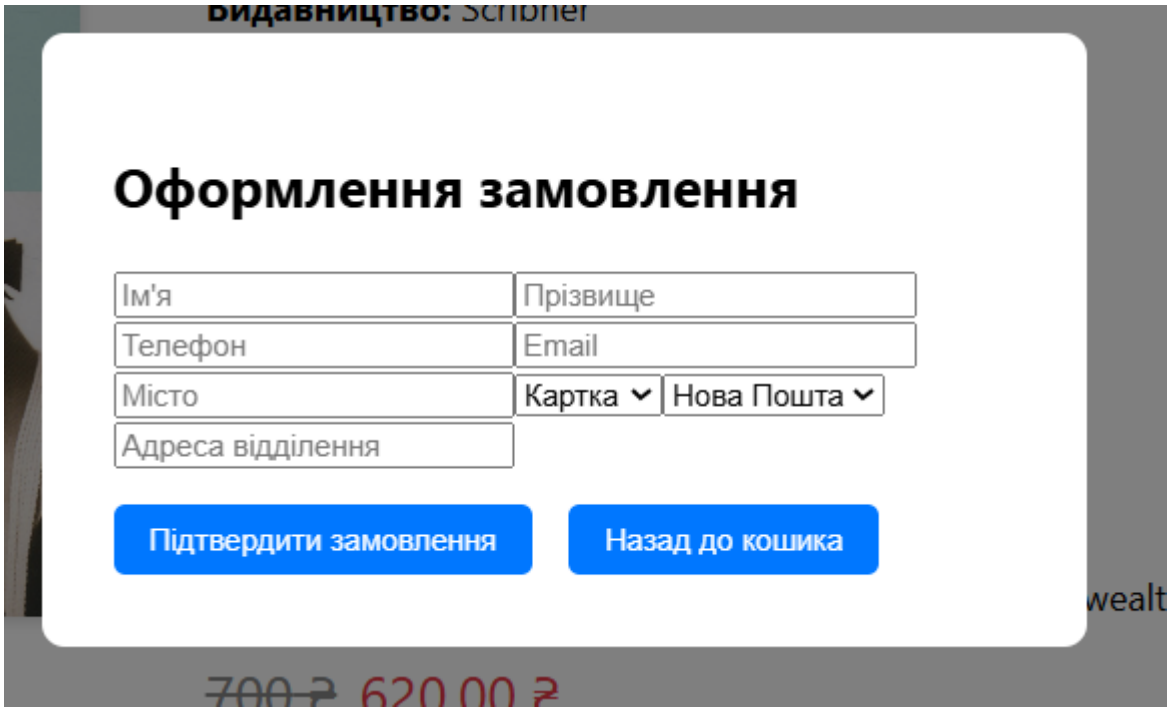
Коментарі

Коментарів ще немає.

5.3. Фільтри



5.4. Оформлення замовлення


Видавництво: Scribner
Оформлення замовлення
Ім'я
Прізвище
Телефон
Email
Місто
Картка
Нова Пошта
Адреса відділення
Підтвердити замовлення
Назад до кошика
wealt
700 ₴ 620.00 ₴