Державний торговельно-економічний університет

Кафедра комп'ютерних наук та інформаційних систем

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Web-система управління діяльністю он-лайн бібліотеки з рекомендаційним модулем»

Студента 2 курсу, 4м групи
спеціальності
122 «Комп'ютерні науки» _____ *підпис студента* Гонгало Вадим Віталійович

Науковий керівник
доктор фізико-математичних наук, професор _____ *підпис керівника* Пурський Олег Іванович

Гарант освітньої програми
доктор фізико-математичних наук, професор _____ *підпис керівника* Пурський Олег Іванович

Київ 2023

Державний торговельно-економічний університет

Факультет інформаційних технологій
Кафедра комп'ютерних наук та інформаційних систем
Спеціальність 122 «Комп'ютерні науки»
Освітня програма «Комп'ютерні науки»

Затверджую
Зав. кафедри _____Пурський О.І.
«9 » грудня 2022 р.

Завдання
на випускну кваліфікаційну роботу студенту

Гонгало Вадиму Віталійовичу
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи

«Web-система управління діяльністю он-лайн бібліотеки з рекомендаційним модулем»
Затверджена наказом ректора від *«06» грудня 2022 р. № 3284*

2. Строк здачі студентом закінченої роботи *24 листопада 2023 року*

3. Цільова установка та вихідні дані до роботи
Мета роботи: збільшення ефективності роботи он-лайн бібліотеки за рахунок розробки рекомендаційного модуля.
Об'єкт дослідження: процес рекомендації книжок..
Предмет дослідження: методи надання рекомендацій у он-лайн бібліотеці.

4. Перелік графічного матеріалу_____
_____
_____
_____

5. Консультанти по роботі із зазначенням розділів, за якими здійснюється консультування:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Пурський О.І.	09.12.2022 р.	09.12.2022 р.
2	Пурський О.І..	09.12.2022 р.	09.12.2022 р.
3	Пурський О.І.	09.12.2021 р.	09.12.2022 р.

6. Зміст випускного кваліфікаційної роботи (перелік питань за кожним розділом)

Вступ

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз наукових джерел за тематикою досліджень

1.2 Аналіз існуючих рекомендаційних систем

1.3 Аналіз основних процесів предметного середовища

1.4 Постановка задачі на розробку рекомендаційної системи

1.5 Висновки до першого розділу

РОЗДІЛ 2. ПРОЕКТУВАННЯ АРХІТЕКТУРИ ОНЛАЙН-БІБЛІОТЕКИ З РЕКОМЕНДАЦІЙНИМ МОДУЛЕМ

2.1 Функціональний аналіз

2.1.1 IDEF0 процесу роботи з онлайн-бібліотекою

2.1.2 Архітектура інформаційної системи

2.2 Математичне забезпечення рекомендаційного модуля

2.3 Інформаційне забезпечення

2.4 Висновки до другого розділу

РОЗДІЛ 3. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ОНЛАЙН БІБЛІОТЕКИ З РЕКОМЕНДАЦІЙНИМ МОДУЛЕМ

3.1 Обґрунтування вибору програмних засобів

3.2 Структура програмного забезпечення

3.3 Керівництво користувача

Висновки

Список використаних джерел

7. Календарний план виконання роботи

№ Пор.	Назва етапів випускної кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	фактично
1	2	3	4
1	*Вибір теми випускної кваліфікаційної роботи*	*01.11.2022*	*01.11.2022*
2	*Розробка та затвердження завдання на випускну кваліфікаційну роботу*	*09.12.2022*	*09.12.2022*
3	*Вступ*	*01.05.2023*	*01.05.2023*
4	*РОЗДІЛ 1. Дослідження предметної області*	*14.06.2023*	*14.06.2023*
5	*Підготовка статті у збірник наукових статей магістрів*	*20.06.2023*	*20.06.2023*
6	*РОЗДІЛ 2. Проектування архітектури он-лайн бібліотеки з рекомендаційним модулем*	*08.09.2023*	*08.09.2023*
7	*РОЗДІЛ 3.Програмне забезпечення он-лайн бібліотеки з рекомендаційним модулем*	*20.10.2023*	*20.10.2023*
8	*Висновки*	*02.11.2023*	*02.11.2023*
9	*Здача випускної кваліфікаційної роботи на кафедру науковому керівнику*	*22.11.2023*	*22.11.2023*
10	*Попередній захист випускної кваліфікаційної роботи*	*29.11.2023*	*29.11.2023*
11	*Виправлення зауважень, зовнішнє рецензування випускної кваліфікаційної роботи*	*04.12.2023*	*04.12.2023*
12	*Представлення готової зшитої випускної кваліфікаційної роботи на кафедру*	*06.12.2023*	*06.12.2023*
13	*Публічний захист випускної кваліфікаційної роботи*	*За розкладом роботи ЕК*	

8. Дата видачі завдання «9» грудня 2022 р.

9. Керівник випускного кваліфікаційного проєкту Пурський О.І.
(прізвище, ініціали, підпис)

10. Гарант освітньої програми Пурський О.І.
(прізвище, ініціали, підпис)

11. Завдання прийняв до виконання студент Гонгало В.В.
(прізвище, ініціали, підпис)

12. Відгук керівника випускної кваліфікаційної роботи

____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____

Керівник випускної кваліфікаційної роботи

____
(підпис, дата)

13. Висновок про випускну кваліфікаційну роботу

Випускна кваліфікаційна робота студента ____________________
(прізвище, ініціали)
може бути допущена до захисту в екзаменаційній комісії.

Гарант освітньої програми_________________________Пурський О.І.____
(підпис, прізвище, ініціали)

Завідувач кафедри _______________________________Пурський О.І.____
(підпис, прізвище, ініціали)
«_____»_________________2023 р.

Анотація

Випускна кваліфікаційна робота присвячена темі "Web-система управління діяльністю он-лайн бібліотеки з рекомендаційним модулем", робота є результатом вивчення та дослідження актуальних питань у сфері інформаційних технологій та бібліотечних систем.

В рамках випускної кваліфікаційної роботи здійснено комплексний аналіз літератури, пов'язаної з обраною темою досліджень. Робота включає в себе огляд сучасних підходів та технологій у сфері он-лайн бібліотек та рекомендаційних систем. Одним із ключових результатів проекту є розроблена архітектура системи он-лайн бібліотеки з інтегрованим рекомендаційним модулем. Ця архітектура враховує сучасні вимоги до зручності використання, доступності та персоналізації бібліотечних послуг. Разом з цим, в рамках роботи було розроблено математичне, інформаційне та програмне забезпечення для реалізації цієї системи. Це включає в себе розробку алгоритмів рекомендацій, інтерфейс користувача та базу даних, необхідну для зберігання та організації контенту бібліотеки.

Ключові слова: онлайн бібліотека, рекомендація, книга, читач.

Anotation

Graduation qualification work is devoted to the topic "Web-system for managing online library activities with a recommendation module", the work is the result of study and research of current issues in the field of information technologies and library systems.

As part of the final qualification work, a comprehensive analysis of the literature related to the chosen topic of research was carried out. The work includes an overview of modern approaches and technologies in the field of online libraries and recommendation systems. One of the key results of the project is the developed architecture of the online library system with an integrated recommendation module. This architecture takes into account modern requirements for usability, accessibility

and personalization of library services. Along with this, as part of the work, mathematical, information and software for the implementation of this system was developed. This includes the development of recommendation algorithms, the user interface, and the database needed to store and organize library content.

Keywords: online library, recommendation, book, reader.

Зміст

ВСТУП

Розробка рекомендаційних систем - це важливе завдання, що виникло на порозі епохи онлайн-технологій та інтернет-ресурсів. Зі зростанням обсягу доступної інформації до таких масштабів, коли навіть найбільш наполеглива людина не в змозі охопити її усю, виникла потреба у засобах, які допомагали б ефективно відібрати потрібний контент. Відповідно до цього, багато передових інтернет-сервісів впроваджують рекомендаційні системи. Ці системи, базуючись на інформації про користувача, намагаються передбачити, які об'єкти, продукти або пропозиції можуть бути корисні саме йому. Об'єктами можуть бути товари, книги, музика, фільми та багато іншого. Наприклад, такі популярні сервіси, як "OLX.ua," "Megogo," "Sweet.tv" і багато інших, впровадили рекомендаційні системи для підвищення користувацького досвіду та збільшення продажів.

"OLX.ua" є одним з найбільших онлайн-бізнесів в Україні, з базою даних, яка налічує понад 13 мільйонів оголошень і щомісяця поповнюється на 4 мільйони нових пропозицій. Кожен другий користувач в Україні відвідує "OLX" принаймні раз на місяць, а мобільний додаток "OLX" займає першу позицію в категоріях "Купівля" для IOS і Android за даними SimilarWeb. На "OLX" щороку продається товарів на понад 1 мільярд доларів, що становить половину річного обороту ринку e-commerce в країні[1].

"Мегого" - це український онлайн-кінотеатр, який надає доступ до легального відеоконтенту. Заснований у 2011 році, цей сервіс став найбільшим відеосервісом для України. Його бібліотека включає понад 80 тисяч відео, включаючи художні фільми, документальні фільми, серіали, шоу, трансляції спортивних подій та багато іншого. "Мегого" можна використовувати з різних пристроїв, включаючи комп'ютери, смартфони та розумні телевізори, з будь-якої точки світу[2].

"SWEET.TV" - це національний онлайн-кінотеатр, який надає доступ до телевізійних передач і фільмів для всієї родини. Платформа пропонує понад 260 телеканалів у якості HD і 4K, а також можливість запису і перемотування телепрограм на 7 днів. Крім того, "SWEET.TV" має велику бібліотеку фільмів, включаючи більше 10 000 стрічок, прем'єри та українські та голлівудські ексклюзиви[3].

Ці приклади демонструють, як рекомендаційні системи можуть бути важливим інструментом для збільшення продажів, покращення користувацького досвіду та зростання лояльності користувачів. Розробка і вдосконалення таких систем стають **актуальними** завданнями у сферах електронної комерції, пошуку контенту та інших галузях, оскільки вони можуть значно поліпшити споживчий досвід та задовольнити потреби сучасного користувача.

Метою дослідження є розробка Web-системи управління діяльністю он-лайн бібліотеки з рекомендаційним модулем.

Для досягнення поставленої мети необхідно виконати наступні **завдання**:

1. Провести системний аналіз роботи бібліотеки.
2. Провести аналіз існуючих рекомендаційних систем.
3. Здійснити функціональний аналіз та розробити архітектуру інформаційної системи.
4. Дослідити математичне та інформаційне забезпечення.
5. Розробити структуру програмного забезпечення.
6. Здійснити програмну реалізацію Web-системи та розробити технологію використання.

Об'єктом дослідження: процеси розробки он-лайн бібліотеки з функцією рекомендацій.

Предметом дослідження: методи, моделі та інформаційні технології розробки рекомендаційних систем.

Методи дослідження:

- Загальнонауковий аналітичний метод.
- Метод функціонального моделювання IDEF0/
- Методи теорії баз даних.
- Методи концептуального моделювання.
- Метод web-програмування на основі відкритої фреймворк-бібліотеки JavaScript ReactJS .

Науковою новизною в системі управління діяльністю он-лайн бібліотеки з рекомендаційним модулем є розробка та впровадження ефективного рекомендаційного алгоритму, який враховує не лише вміст бібліотеки, але й контекст інтересів користувачів. Це вдосконалення дозволяє покращити якість рекомендацій, забезпечуючи користувачам більш персоналізований та задовільний досвід при виборі контенту в онлайн бібліотеці.

Практичне значення. Web-система управління діяльністю он-лайн бібліотеки з рекомендаційним модулем має практичне значення, оскільки вона допомагає поліпшити користувальницький досвід та зробити бібліотечний ресурс більш доступним і зручним у використанні. Завдяки рекомендаційному модулю користувачі отримують персоналізовані рекомендації, що сприяє збільшенню використання бібліотеки та покращенню їхньої задоволеності. Така система також допомагає бібліотекам ефективніше вирішувати завдання зі збільшення обсягу читачів та популяризації доступного контенту, що може призвести до зростання популярності та впливу бібліотек в онлайн-середовищі.

Публікації. Результати дослідження опубліковано у збірнику наукових статей студентів, які здобувають освітній ступінь магістра за спеціальністю «Комп'ютерні науки» ДТЕУ. Web-система управління діяльністю он-лайн бібліотеки з рекомендаційним модулем // Прикладні комп'ютерні технології : зб. наук. ст. студ. / відп. ред.— Київ : Держ. торг.-екон. ун-т, 2023. – С. 34-37

Структура та обсяг випускної кваліфікаційної роботи. Випускна кваліфікаційна робота складається із вступу, трьох розділів, висновків, списку використаних джерел із 29 найменувань, додатків і містить 50 сторінок основного тексту, 33 рисунка і 2 таблиці.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз наукових джерел за тематикою досліджень

Рекомендаційна система - це інструмент, який надає користувачеві рекомендації щодо об'єктів на основі їхніх попередніх вподобань та відповідності інших користувачів з подібними смаками і інтересами. Такі системи широко використовуються у сферах, таких як електронна комерція та розваги, для подолання проблеми інформаційного навантаження. Крім того, їх застосування розглядається в інших галузях, включаючи освіту і навчання.

Попри те, що рекомендаційні системи з'явилися відносно нещодавно, проблемі їх побудови і застосування в різних прикладних завданнях присвячена величезна кількість статей і книг. Автори використовують різні методи для поліпшення якості рекомендацій і розширюють сферу застосування рекомендаційних систем. Однією з перших публікацій на цю тему є стаття Стівена Поллока [4], датована 1988 роком. У ній автор представив систему ISCREEN, яка дозволяла користувачеві самостійно визначати деякий набір правил, по яким ISCREEN надалі фільтрувала текстові повідомлення і відбирала тільки релевантні. І хоча система не рекомендувала користувачеві повідомлення, сама робота показала актуальність завдання фільтрації і відбору релевантної інформації. У 1992 році з'явилася робота Девіда Голдберга [5], у якій автори розробили експериментальну систему електронної пошти Tapestry, яка дозволяла фільтрувати email-розсилки. На відміну від роботи Д. Поллока, ця система відбирала повідомлення повністю автоматично на основі реакцій користувачів. Автори статті представили ідею колаборативної фільтрації і фільтрації на основі змісту. Робота Д. Голдберга мала сильний відгук і викликала масу обговорень. Примітна стаття Пола Різника та Хела Варьяна [6] 1997 року, в якій автори міркують про негативні наслідки застосування рекомендаційних систем, таких як: переслідування недобросовісних цілей при

складанні рекомендацій, використання персональних даних користувачів, а також погіршення продуктивності сервісів через роботу рекомендаційних систем.

Пітер Брусиловський в 1996 році в роботі [7] представив новий напрям в області адаптивних інтерфейсів - адаптивний гіпермедіа(adaptive hypermedia). Такі системи будують модель кожного користувача і використовують її, наприклад, для підстроювання змісту сторінки сайту під знання і цілі користувача або пропозиції найцікавіших йому посилань. Автором були розглянуті методи побудови моделей користувачів, що отримали широке застосування надалі.

У роботі [8] Майкла Паззані велика увага приділяється отриманню профілю призначених для користувача інтересів для рекомендаційних систем. Проведений порівняльний аналіз двох основних методів фільтрації - колаборативної фільтрації і фільтрації на основі змісту, а також описується новий тип фільтрації - фільтрація на основі демографії, який враховує соціально-демографічні показники користувача при складанні рекомендацій. По мірі накопичення теоретичного і дослідницького матеріалу стали з'являтися статті більше прикладного характеру, в яких дослідники шукали нові застосування для рекомендаційних систем. Наприклад, Кьон Чже Ким і Хюнчул Хан в роботі [9] розробили рекомендаційну систему методом колаборативної фільтрації для показу мобільної реклами. Есма Аимер і Джилес Брассар [10] створили систему для рекомендацій товарів для електронної комерції, Дженифер Голдбек [11] представила рекомендаційну систему для фільмів. Ці роботи говорять про широкий спектр застосовності рекомендаційних систем. Завдання автоматичного проставляння тегів до постів можна розглянути як завдання мультикласової класифікації. У області машинного навчання це завдання ставиться як зіставлення деякого об'єкту одній або декільком заздалегідь визначеним категоріям. У разі, коли кількість категорій рівна двом,

говорять про завдання бінарної класифікації, якщо категорій більше - мультикласифікація. Існує декілька методів класифікації тексту. Так, Володимир Вапник і Корінна Кортес в роботі [12] представили метод машинного навчань для проблеми класифікації, відомого як метод опорних векторів. Торстен Джочимс в статті [13] описав застосування цього методу в завданні класифікації тексту. Наївний байесовский класифікатор [14] відомий з 1950 років і залишається популярним методом класифікації тексту. Дерево ухвалення рішень [15] і різні його варіанти також активно застосовуються для цього завдання. Усі ці методи були вивчені і використані в роботі, а по підсумках проведено їх порівняння. У тих випадках, коли число класів, до одного з яких вимагається визначити приналежність об'єкту, більше двох, говорять про завдання багато класової класифікації(англ., multiclass classification). У свою чергу, багато класові класифікатори можна розділити на моно тематичні(англ. singlelabel classifier) і полі тематичні(англ. multilabel classifier) - в залежності від того, чи треба однозначно визначити приналежність об'єкту до класу(яке раніше сформулювалося в постановці завдання) або вимагається віднести об'єкт потенційно декільком класам одночасно. Існує два головні методи рішення задачі полі тематичної класифікації: методи перетворення проблеми(англ. problem transformation methods) і методи адаптації алгоритму(англ. algorithm adaptation methods). Методи перетворення проблеми перетворюють завдання політематичної класифікації на безліч завдань бінарної класифікації, які можуть бути вирішені бінарними класифікаторами. Методи адаптації алгоритму перетворюють алгоритми так, щоб вони могли вирішувати задачу полі тематичної класифікації без її перетворення. Серед найбільш відомих методів перетворення проблеми можна виділити метод бінарної релевантності(англ. binary relevance) [16], ланцюговий класифікатор(англ. chain classifier) [17] і метод random k - labelsets [18]. Для визначення семантичної близькості слів використовуються алгоритми дистрибутивної семантики, які на

основі величезного масиву даних будують векторні представлення слів, що називаються контекстними векторами. Потім семантична відстань між словами обчислюється як косинусна відстань між відповідними векторами. Більш повне завдання дистрибутивної семантики і її місце в прикладній лінгвістиці обговорюється в [19] і [20]. Для обчислення контекстних векторів часто використовуються методи машинного навчання. Найбільш примітні результати показують нейронні мережі. Існує величезна кількість архітектури побудови нейронних мереж [21]. Найбільш відома в дистрибутивній семантиці модель word2vec використовує багатошарові нейронні мережі для пророцтва слова за навколишніми словами, або навколишні слова по контексту [22] і [23]. Вона стала надзвичайно популярною відразу після публікації зважаючи на високу швидкість навчання і кращої якості векторних представлень в порівнянні з іншими популярними на той момент методами. Багато дослідників досі вивчають властивості цієї моделі і можливості її застосування.

1.2 Аналіз існуючих рекомендаційних систем

Сьогодні, в сучасному світі, існує безліч технологічних гігантів, які активно використовують системи рекомендацій для покращення своїх продуктів та послуг. Ці компанії пропонують різні види контенту, такі як аудіо та відео, реалізують різноманітні товари та функціонують як посередники у торгівлі цими товарами, а також надають інформацію про новини та багато інше. Нижче ми розглянемо деякі з найпопулярніших з них:

SoundCloud - це всесвітньо відома платформа для аудіо контенту, яка відображає справжню силу співпраці між творцями, слухачами і кураторами, які завжди в курсі щодо актуального звучання та майбутніх тенденцій у світі музики. Заснована у 2007 році, SoundCloud надає аудіо-творцям з усього світу найкращі інструменти, сервіси та ресурси для розвитку їхньої кар'єри. Понад

200 мільйонів треків від понад 20 мільйонів авторів прозвучали у 190 країнах, і SoundCloud залишається лідером в світі музичної індустрії.

YouTube - це безкоштовна онлайн-платформа для обміну відеоматеріалами, яка дозволяє легко переглядати та навіть завантажувати власні відео для обміну ними з іншими користувачами. Почавши свою діяльність у 2005 році, YouTube на сьогоднішній день є однією з найпопулярніших платформ в Інтернеті, на якій щомісяця користувачі дивляться приблизно 6 мільярдів годин відео. Сервіс пропонує два типи рекомендацій: перший - це рекомендації відео, які наразі є найбільш популярними, а другий - рекомендації, засновані на відео, переглянутих користувачем.

eBay - це найбільший онлайн-ринок у світі з понад 100 мільйонами активних користувачів по всьому світу, де можна купувати і продавати різні товари. Платформа була заснована у 1995 році. Компанія eBay Inc. є лідером у глобальній торгівлі, об'єднуючи мільйони покупців і продавців з усього світу. В їхньому портфелі брендів входять eBay Marketplace і eBay Classifieds Group, які працюють на 190 ринках світу. eBay надає продавцям можливість розвивати свій бізнес незалежно від його масштабу, походження або географічного розташування. Громада користувачів eBay впливає на світовий ринок електронної комерції, і загальна вартість проданих товарів на eBay становила понад 68,6 мільярдів доларів на кінець 2011 року. Це означає понад 2100 доларів продажів кожну секунду. eBay надає рекомендації на основі покупок, переглядів, оцінок та відгуків користувачів.

Apple News - це додаток, який об'єднує різноманітні джерела новин, блоги, інформаційні матеріали, відео публікації, журнали та газети в одній зручній стрічці. Користувачі можуть відстежувати різні видання та відображати їх у стрічці новин, яка сортується за темою та видавцем. Додаток також є платформою для Apple News+, підписки, що надає доступ до преміального

новинного контенту. Apple News була вперше представлена разом з iOS 9 і, використовуючи канали RSS та публікації, збирає статті з усього Інтернету, щоб подати їх користувачам у зручному форматі. Додаток доступний для iOS, iPadOS та macOS, і користувачі можуть підписуватися на свої улюблені видання, ставити лайк або дизлайк статтям та зберігати історії для офлайн-читання.

Щодо рекомендаційних систем, пов'язаних з книгами, варто зазначити, що на ринку цей сегмент поки що досить малопопулярний і має значний потенціал для розвитку. Нижче розглянемо найпопулярніші онлайн-бібліотеки:

Yakaboo.ua - це одна з найбільших книжкових платформ в Україні, яка пропонує як електронні, так і друковані книги. Останнім часом компанія провела успішний ребрендінг і наразі має базу з понад 3 мільйонів клієнтів.

Nashformat.ua - Ця бібліотека діє на ринку вже понад 15 років і виділяється серед інших онлайн-магазинів тим, що принципово не продає російські книжки. Незважаючи на це, бібліотека нараховує понад 30 тисяч найменувань книг від українських видавництв.

Ukrlib.com.ua - Це найбільша українська онлайн-бібліотека, яка успішно функціонує з 2000 року і абсолютно безкоштовна та вільна для всіх користувачів. Основною метою цього проекту є зробити українську літературу доступною для всіх, хто бажає її читати. Каталог бібліотеки налічує нескінченну кількість художньої, пізнавальної та навчальної інформації, включаючи українську літературу та літературу інших народів світу.

Книгарня "Є" - Це найбільший інтернет-магазин книг в Україні. Перший магазин був відкритий 21 грудня 2007 року в місті Київ, і вже в 2008 році цю бібліотеку визнано найкращою серед книгарень України.

Нижче представлено таблицю з порівнянням цих чотирьох сервісів:

Таблиця 1.1. Порівняльна таблиця онлайн-бібліотек

	Yakaboo.ua	nashformat.ua	Ukrlib.com.ua	Книгарня “Є”
Наявність IOS та Android додатку	+	-	-	-
Можливість оцінювання книг	+	+	-	+
Доставка друкованих книг до дому	+	+	-	+
Наявність преміум підписки	-	+	-	-
Підтримка різних мов у системах	+	-	+	-
Наявність персонального кабінету	+	+	-	+
Офлайн режим	-	+	-	-

Вже зараз можна зробити висновок, що ринок рекомендаційних систем для книг є практично неосвоєним. Саме через цю можливість, створення нової рекомендаційної системи вважається вкрай обіцяючою і доцільною, особливо з урахуванням сучасних тенденцій у розвитку цієї сфери протягом останніх років.

1.3 Аналіз основних процесів предметного середовища

У цьому розділі ми проведемо аналіз основних процесів та побудуємо відповідні діаграми, фокусуючись на предметному середовищі без застосування рекомендаційної онлайн системи для вибору книг.

Спочатку створимо контекстну діаграму "ЯК Є" та подальшу декомпозицію процесів вибору та читання книги без використання рекомендаційної системи (рисунок 1.1).

На початку діаграми ми маємо вибір бібліотеки, яку обрав наш клієнт, та обрані жанри книг, які цікавлять його в цій бібліотеці. Процес вибору книг включає пошук відповідного розділу в бібліотеці та можливу консультацію з бібліотекарем щодо вибору книги. На виході з усіх цих процесів отримуємо перелік прочитаних книг.

Елементами управління цим процесом є відповідні закони про авторське право, оскільки в бібліотеці мають бути лише ліцензійні книги, і в жодному разі не можна розповсюджувати піратські копії. Механізмами для цього процесу є сам клієнт та бібліотекар, з яким він взаємодіє, а також доступ до Інтернету, завдяки якому наш клієнт може знайти бібліотеку та необхідну інформацію.

Рис. 1.1. Контекстна діаграма

Подальшим кроком буде створення декомпозиції основних функцій (рисунок 1.2). Коли користувач планує читання книг, завжди першим кроком є пошук бібліотеки у своєму місті в мережі інтернет. Після вибору однієї з бібліотек він приходить до неї та входить в її приміщення, де починає пошук розділу з бажаним жанром книг. Для вибору конкретних книг, які йому цікаві, користувач взаємодіє з бібліотекарем та слухає рекомендації щодо кожної книги, після чого розпочинає їх читати.

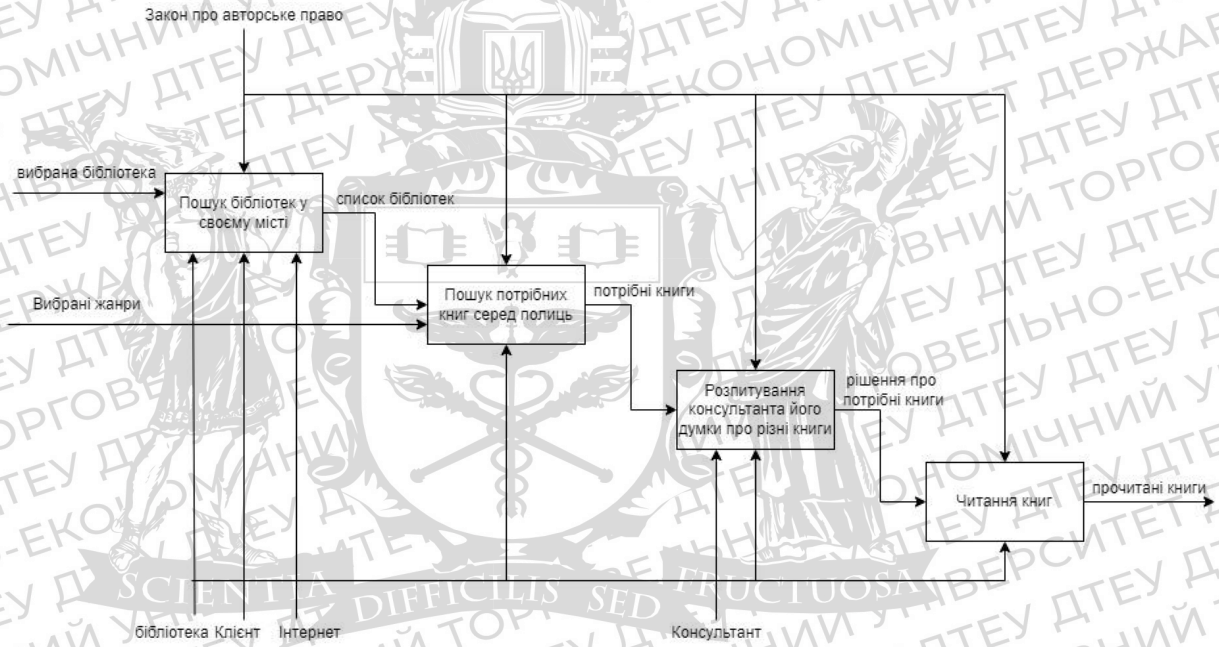

Рис. 1.2. Декомпозиція основних функцій

Отже, процес вибору та читання книг є досить часо- та енерговитратним завданням. Пошук відповідної бібліотеки, перехід між безліччю відділів з різними жанрами та недостатність відгуків про більшість книг можуть значно ускладнити завдання для потенційних читачів. Завдяки розробленій рекомендаційній системі, цей процес значно спроститься: користувачам доведеться лише відвідати веб-сайт та обрати книги, які система рекомендує, і почати їх читати.

1.4 Постановка задачі на розробку рекомендаційної системи

Задача розробки – розробка рекомендаційної системи для сортування найбільш релевантних книг для користувачів.

Мета створення – створення комфортної та зрозумілої системи задля задоволення потреб користувачів в плані контенту.

Задачі системи:

1. Реалізація адміністративної панелі задля:
   - Додавання книг;
   - Видалення книг;
   - Редагування інформації про книги;
2. Надання персональних рекомендацій користувачеві сформованих на основі його діяльності у системі
3. Можливість оцінювання(5-зіркова шкала) книг у системі користувачем
4. Можливість написання відгуку від користувача для кожної книги у системі
5. Можливість перегляду каталогу книг у системі користувачем
6. Можливість створити власний кабінету для входу у систему задля отримання персональних рекомендацій та доступу до закладок користувача
7. Можливість пошуку книг за назвою користувачем системи
8. Можливість сортування книг за рейтингом/жанрами/за алфавітом
9. Створення преміум розділу для користувачів з підпискою, в якому буде відкриватися повний доступ до всього каталогу книг.

І на останок подивимось на інформаційну архітектуру сайту (рисунок 1.3), який вміщає в собі особистий кабінет, розділ “Про нас”, каталог книг та адміністративні функції. В особистому кабінеті користувач може побачити список прочитаних книг та улюблених жанрів і отримати рекомендовану книгу. В розділі ”Про нас” він може прочитати інформацію про бібліотеку, а у

каталозі книг побачити список книг та здійснити пошук по ним або відфільтрувати їх за жанрами.

Рис. 1.3. Інформаційна архітектура сайту

1.5 Висновки до першого розділу

В ході аналізу літератури та порівняльного аналізу існуючих рекомендаційних систем було виявлено актуальність розробки нової рекомендаційної системи. Результати цього аналізу вказують на необхідність створення і впровадження відмінної від існуючих систем рекомендаційної системи для книг, що відповідає потребам сучасних користувачів.

Подальший аналіз основних процесів предметного середовища, включаючи контекстну діаграму та декомпозицію, надав можливість кращого розуміння складності завдання та структури системи. Це допомогло визначити ключові етапи реалізації рекомендаційної системи книг.

У результаті постановки задачі на розробку рекомендаційної системи та розгляду інформаційної архітектури сайту були визначені основні цілі та завдання проекту. Висновки цього розділу служать основою для подальшої розробки та впровадження рекомендаційної системи для книг, спрямованої на підвищення задоволення користувачів та покращення їхнього досвіду використання веб-сайту.

РОЗДІЛ 2. ПРОЕКТУВАННЯ АРХІТЕКТУРИ ОН-ЛАЙН БІБЛІОТЕКИ З РЕКОМЕНДАЦІЙНИМ МОДУЛЕМ

2.1 Функціональний аналіз он-лайн бібліотеки

Спершу розглянемо функціональну структуру он-лайн бібліотеки. Розділимо її на дві основні категорії: клієнтські та адміністративні функції. Клієнтська частина включає можливість створення особистого облікового запису, перегляд доступного каталогу книг та пошук книг за назвою або жанром. Користувач також може здійснити реєстрацію у системі. У своєму особистому кабінеті клієнт має можливість отримати персональні рекомендації, залишити відгук про книги та переглянути свій список улюблених книг.

Адміністративна частина включає в себе роботу з клієнтами, включаючи управління підписками та формування списку клієнтів. Також вона охоплює адміністрування книг, включаючи додавання, вилучення та редагування книг.

Рис. 2.1. Дерево функцій

2.1.1 IDEF0 процесу роботи з он-лайн бібліотекою

У цьому розділі проведемо аналіз основних процесів та побудуємо відповідні діаграми. Спочатку створимо контекстну діаграму "ЯК БУДЕ" та її декомпозицію для ілюстрації процесів у предметному середовищі онлайн-бібліотеки з рекомендаційним модулем.

На контекстній діаграмі (рисунок 2.2) представлено процес роботи з он-лайн бібліотекою. На початку цього процесу ми маємо вхідні дані про користувачів, такі як імена, електронні адреси або улюблені книги. Інший важливий вхідний процес полягає в зборі інформації про книги, що включає в себе дані про автора книги, її назву та опис. Після виконання всіх необхідних процесів користувач отримує персональну рекомендацію щодо книги, яку він може прочитати. Елементами управління цього процесу є закон про авторське право, оскільки бібліотека повинна пропонувати лише легальні книги з відповідними ліцензіями та не поширювати піратські копії. Механізмами цієї бібліотеки є рекомендаційний модуль, який надає рекомендації щодо цікавих книг, та взаємодія користувача з веб-сайтом.

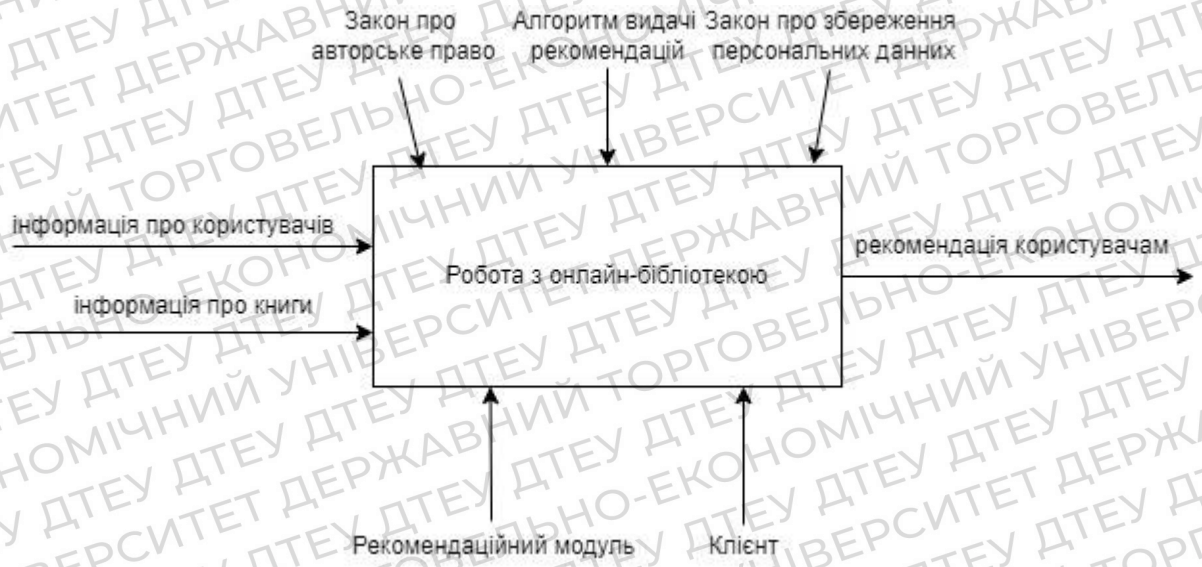

Рис. 2.2. Контекстна діаграма IDEF0

Потім я створив декомпозицію основних функцій (рис. 2.3). Коли користувач бажає знайти книгу, яка зацікавила б його, він входить до свого особистого кабінету. Після входу в особистий кабінет він переходить до каталогу книг і починає пошук книг, що його зацікавили. Обравши книги, які його цікавлять, користувач повертається до свого особистого кабінету, де отримує персональну рекомендацію.

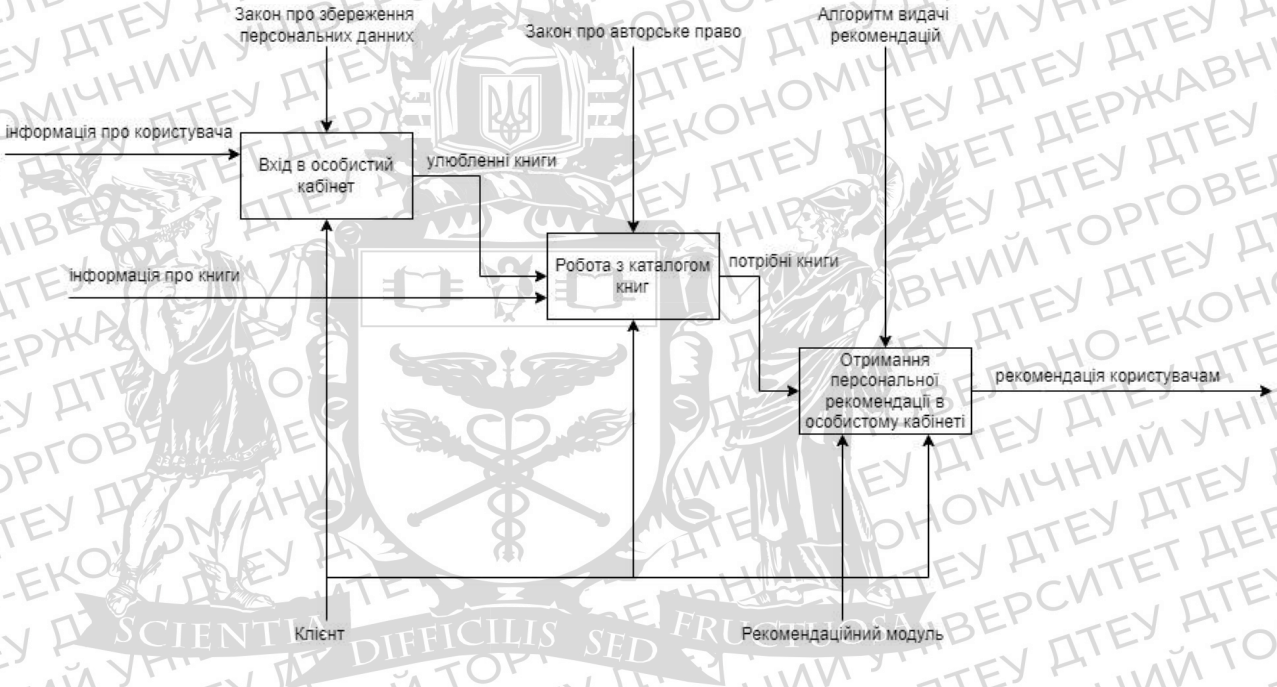

Рис. 2.3. Декомпозиція основних функцій IDEF0

2.1.2 Архітектура інформаційної системи

Давайте розглянемо надану інформаційну архітектуру нашої системи (рис. 2.4). На цьому зображенні представлена он-лайн бібліотека книг, яка складається з двох основних частин: модуль роботи з клієнтами і адміністративний модуль. Модуль роботи з клієнтами розділений на три інших модулі: модуль роботи з каталогом книг, модуль реєстрації та модуль роботи з особистим кабінетом, де ми можемо знайти рекомендаційний модуль та модуль перегляду улюблених книг та жанрів. Крім того, у нас є адміністративний модуль, який включає в себе підсистему роботи з базою даних і модуль

звітності. Зокрема, модуль підсистеми роботи з базою даних поділяється на модуль роботи з підбазою підписок, роботи з підбазою клієнтів та підбазою книг.

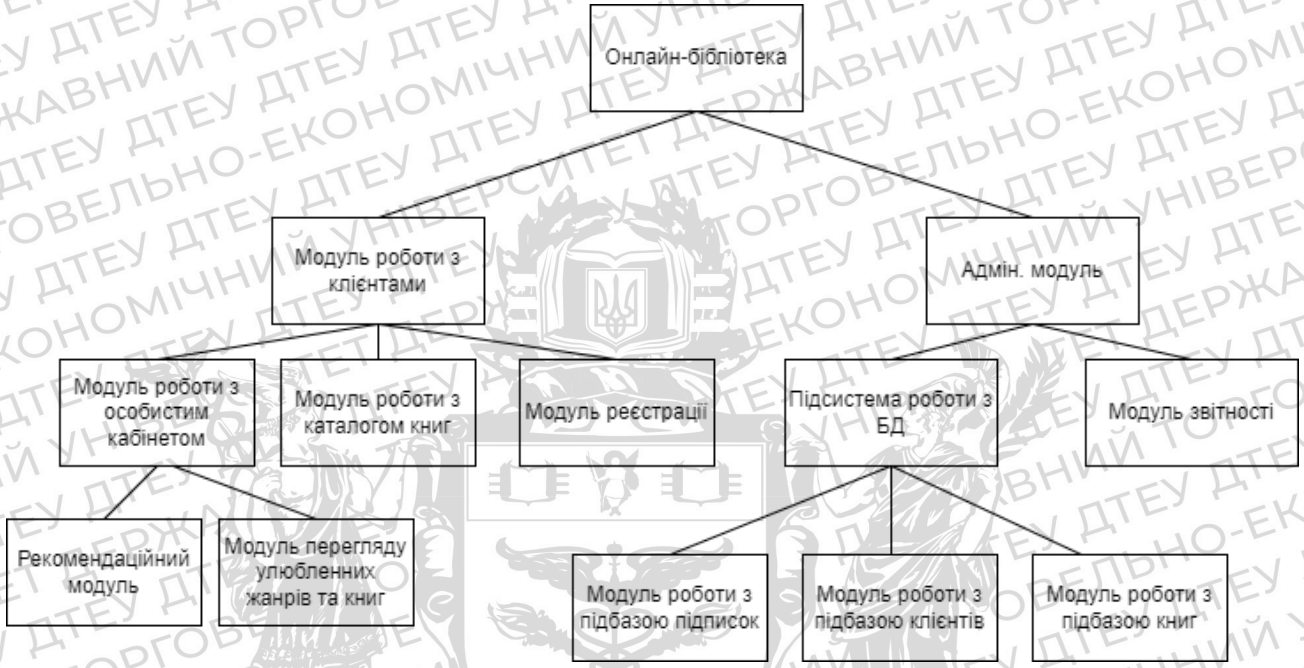

Рис. 2.4. Архітектура системи

2.2 Математичне забезпечення рекомендаційного модуля

Математично, задачу знаходження найбільш релевантного об'єкта можна описати наступним чином[23, 24]:

$$\forall u \in U, s'_u = argmax_{s \in S} h(u, s), \quad (1)$$

де U – множина користувачів, u – вибраний користувач, S – множина елементів системи, які потенційно можуть бути рекомендовані користувачу, s – вибраний елемент, h - функція, яка показує ступінь зіставлення деякого елементу s з деяким користувачем u. Таким чином, задача зводиться до знаходження такого елементу $s' \in S$, для якого ступінь зіставлення з користувачем $u \in U$ є максимальним (у випадку рекомендації одного елементу). При цьому на точність рекомендації суттєво впливає не лише інформація всередині системи, а і вибір функції [24].

Розглянемо більш детально метод рекомендацій. Я використовую у своїй випускній кваліфікаційній роботі метод фільтрації на основі вмісту. Як можна зрозуміти з назви підходу, фільтрація на основи вмісту перш за все опирається на інформацію про елементи системи, а не на інформацію о користувачах. Тобто рекомендації базуються на знаходженні схожих елементів, до тих, що користувач вже побачив у минулому.

Переваги методу фільтрації на основі вмісту включають:

- Незалежність від взаємодії користувачів: Метод не потребує даних про інших користувачів для генерації рекомендацій, що робить його привабливим для нових юзерів або у випадках, коли даних про взаємодію недостатньо.
- Розширення рекомендацій: Метод може легко розширювати рекомендації на новий контент, оскільки він аналізує характеристики самого контенту, а не залежить від даних про попередню взаємодію користувачів з ним.
- Інтерпретованість рекомендацій: Оскільки метод використовує характеристики контенту для рекомендацій, його результати можуть бути легше пояснені користувачам. Користувачі можуть бачити, які характеристики використовуються для порівняння та рекомендацій.

Для цього методу необхідно створити профіль читача та профіль елемента системи. Після цього, на основі параметрів елементів, можна зробити висновок щодо відповідності конкретного елементу конкретному користувачу. Для опису елементів системи та створення їх профілю рекомендаційні системи ставлять у відповідність кожному елементу певний набір ключових слів. Фактично роботу рекомендаційної системи на основі фільтрації вмісту можна зобразити наступною діаграмою(рисунок 2.5):

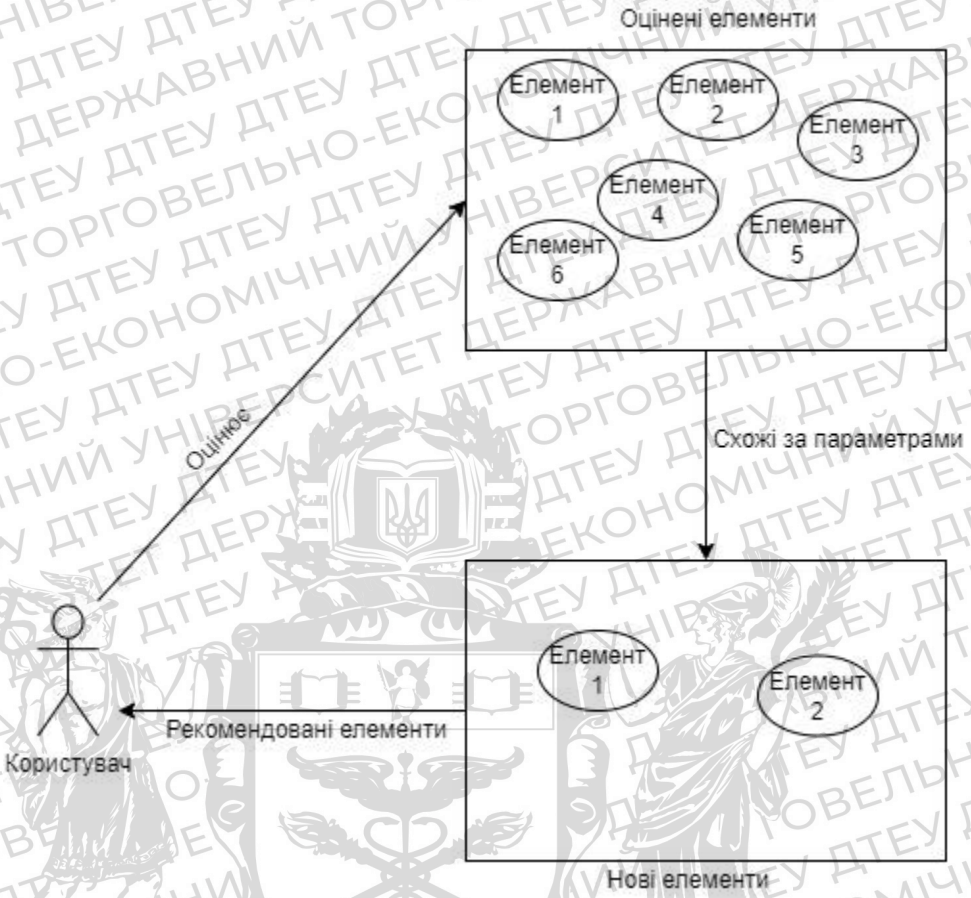

Рис. 2.5. Принцип роботи рекомендаційної системи на основі фільтрації вмісту

Іншими словами, щоб надавати користувачу рекомендації щодо нових елементів, система повинна проаналізувати елементи, які користувач вже оцінив у минулому, знайти параметри, що їх об'єднують, і на основі цих параметрів рекомендувати нові елементи, які відповідають цим параметрам і які користувач ще не переглядав, тобто вони є новими для нього.

2.3 Інформаційне забезпечення

У розділі про інформаційне забезпечення ми детально розглянемо різні моделі баз даних, включаючи концептуальні та фізичні моделі. Давайте розпочнемо з концептуальної моделі.

Ми почнемо з користувача, який має особисті дані, такі як ім'я, логін і пароль для авторизації в особистому кабінеті. Крім того, якщо користувач має права адміністратора, він має доступ до адміністративної панелі, де може додавати нові книги. У каталозі книг, який розташований в окремому розділі, доступній кожній книзі присвоюються такі атрибути, як назва та короткий опис, які допомагають користувачам визначити, чи цікаво читати дану книгу. Кожна книга також призначена до певного жанру, і жанр характеризується своєю назвою. Крім того, кожна книга має одного автора, який її написав.

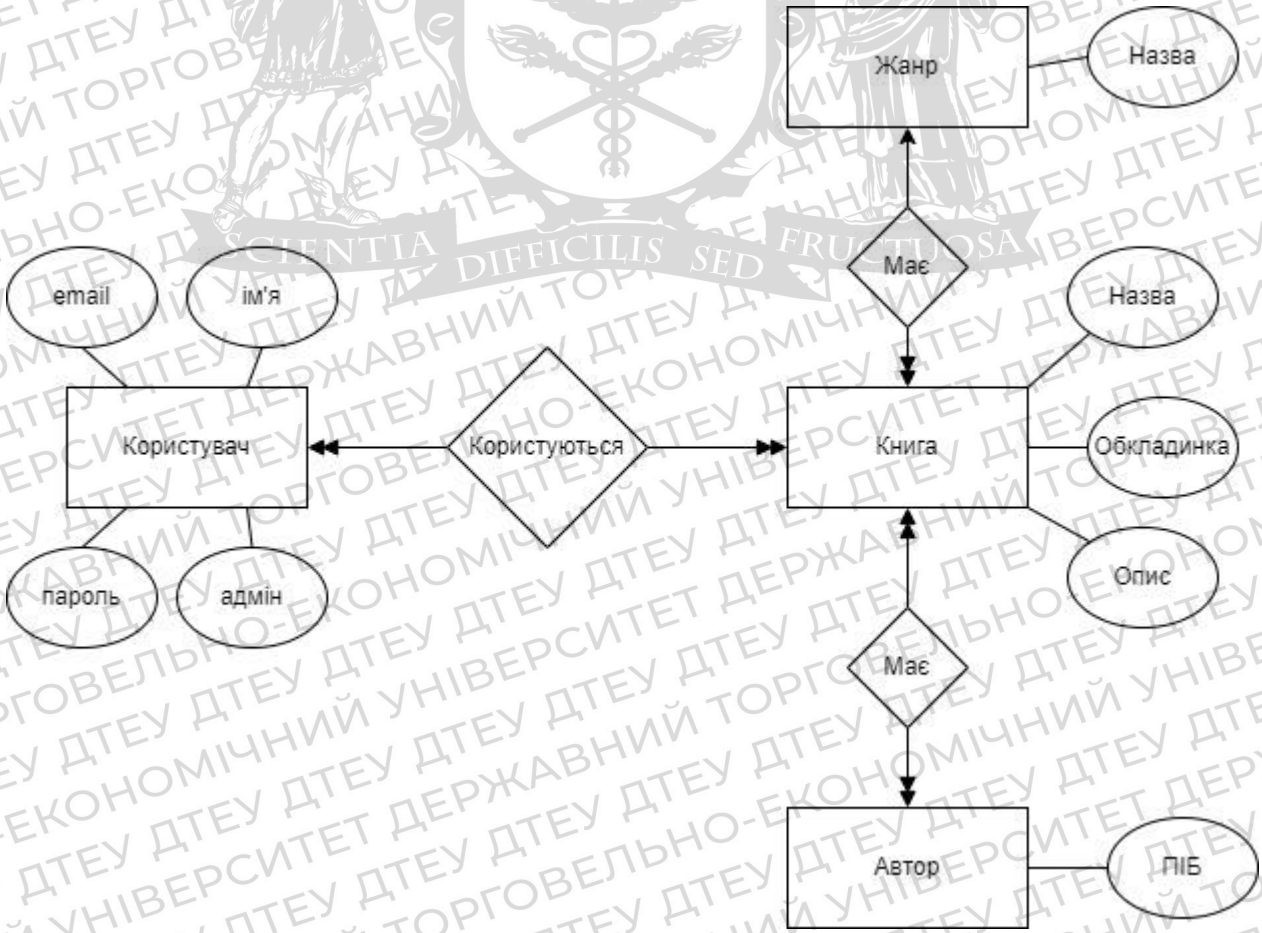

Рис. 2.6. концептуальна модель бази даних

Для розробки NoSQL бази даних біла використана система управління базою даних MongoDB:

MongoDB - Це база даних NoSQL з відкритим кодом, розроблена для високої продуктивності, високої доступності та простої масштабованості. Колекція та документ - це два в основному використовувані терміни / концепції в MongoDB.

Далі, розглянемо процес розробки даних таблиць бази даних у NoSQL СУБД MongoDB. На рисунку 2.7 можна побачити створену базу даних, де можна виявити п’ять колекцій баз даних, під назвами “Users”, “Books” та “favorites”, “recommendations” та “reviews”. Нижче буде наведений опис і рисунок кожної таблиці окремо.

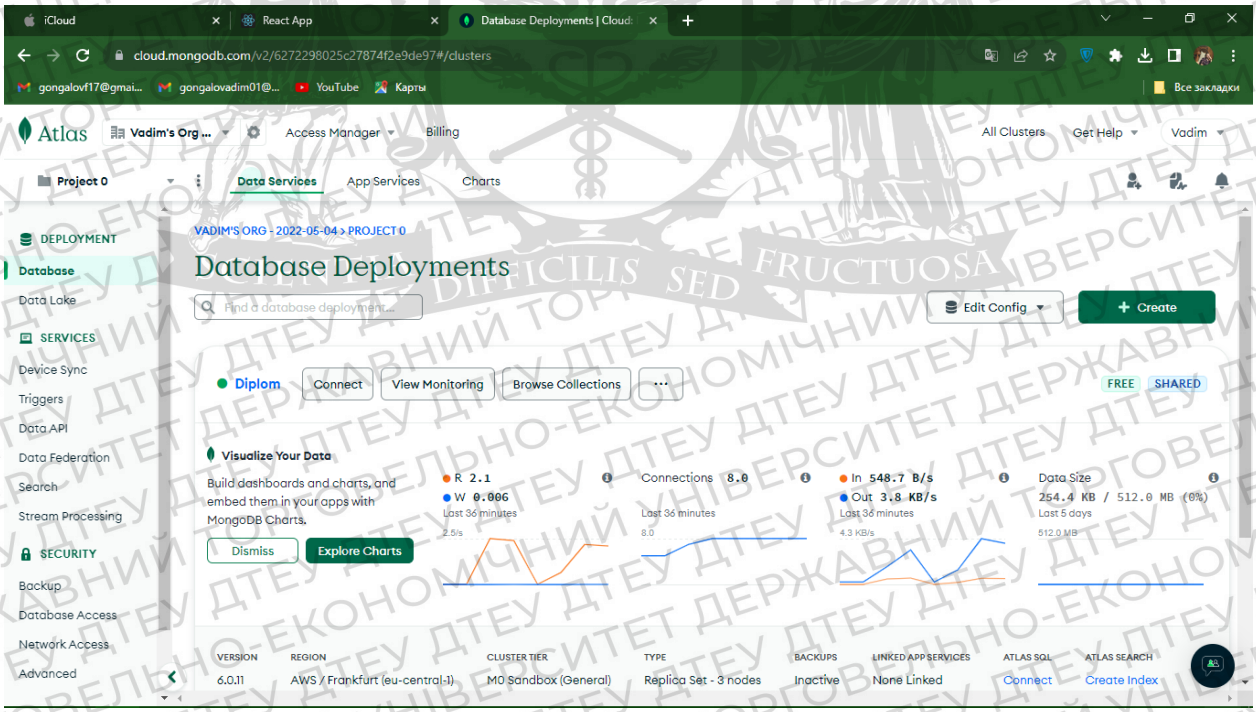

Рис. 2.7. створена база даних

Таблиця з книгами наведена на рисунку 2.8. Дана таблиця зберігає повний каталог книг разом з усіма доступними параметрами, такими як: “name”, “author”, “genre”, ”description” і “image”. Колекція відповідає за наявність книг у

каталозі та особистому кабінеті. Без цієї таблиці неможлива рекомендація читачу.

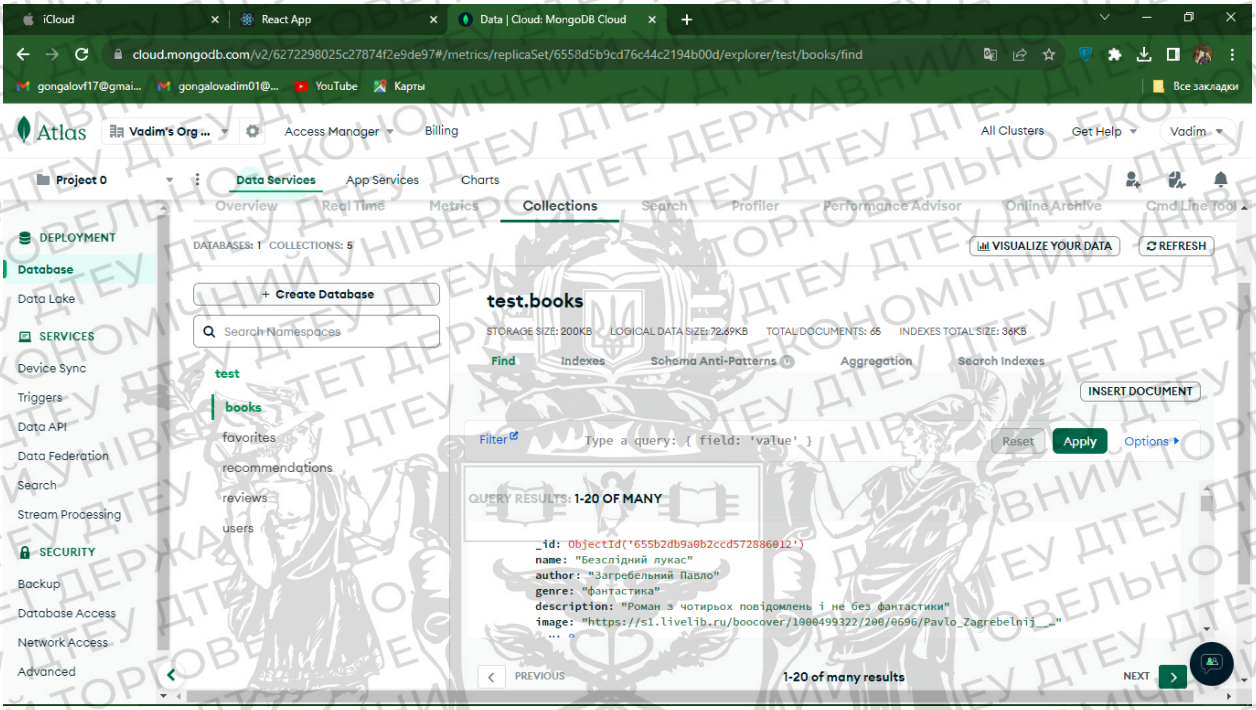

Рис. 2.8. таблиці з книгами

Нижче, на рисунку 2.9 наведений документ з улюбленими книгами користувача. Цей документ потрібен для відображення відзначених книг читачем і, у подальшому, формування рекомендації. Колекція, має документи, які зберігають такі поля, як: “id”, “BookID”, “UserID”. Під цю таблицю у системі виведене окреме поле у особистому кабінеті, відповідно якому користувач може побачити ці книги.

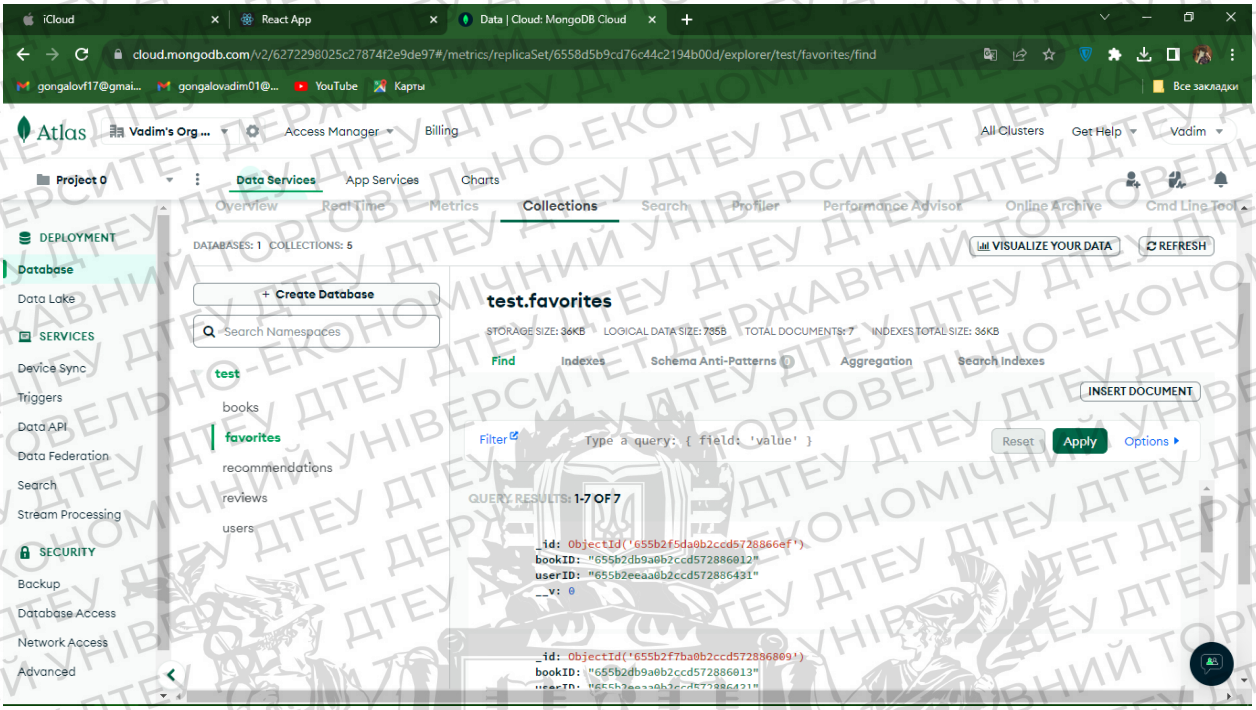

Рис. 2.9. улюблені книги користувачів

На рисунку 2.10 наведено таблицю, де зберігаються рекомендації користувачам. У таблиці є такі поля, як: “userID”,”genre”,”author”. Поле “userID” відповідає за користувача і в ньому ми можемо побачити унікальний номер користувача, якому система рекомендує елементи. А поля “genre” та “author” відповідають за жанр та автора книги.

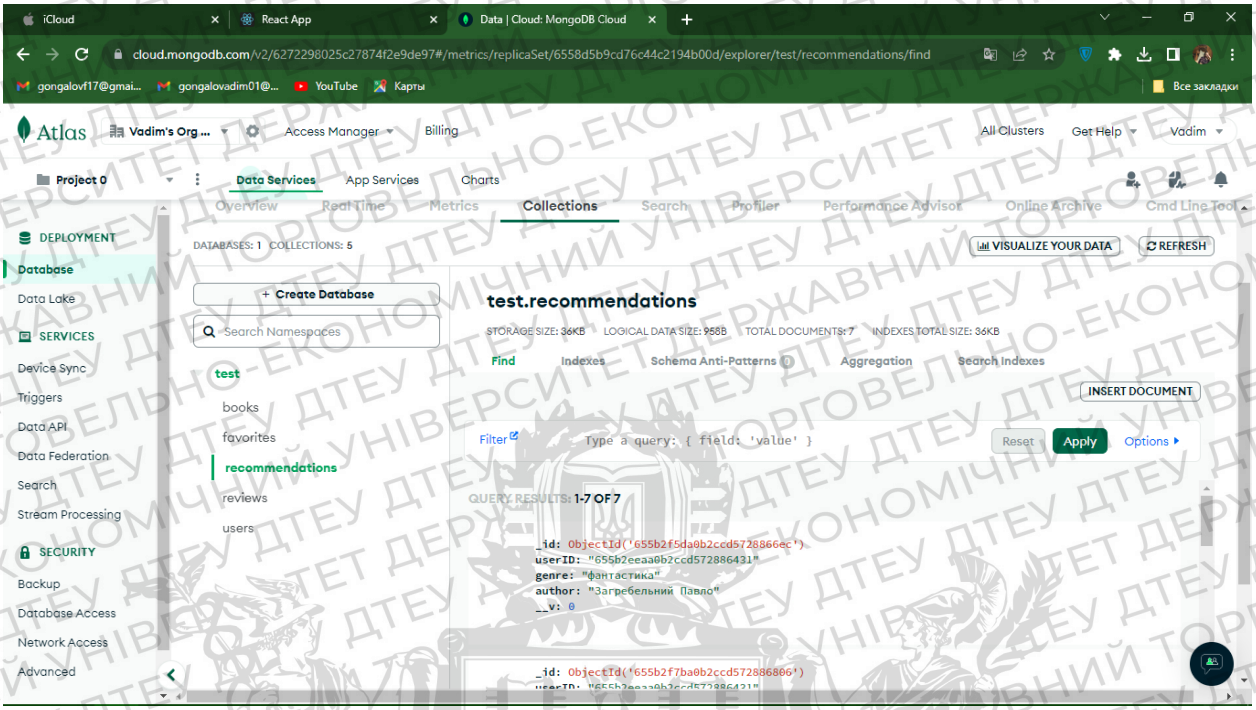

Рис. 2.10. таблиця з рекомендаціями

У цій таблиці(рисунок 2.11) можна побачити колекцію яка відповідає за відгуки. Маємо тут такі поля: “bookID”,”userID”, “review”,”grade”. Поля “bookID” та “userID” я вже описував вище. У поле “review” зберігається текстовий відгук написаний читачем. Але це не обов’язкове поле оскільки читач може тільки оцінити книги за п’яти бальною шкалою і не залишати текстовий відгук. А поле “grade” це шкала оцінювання книги. У цьому полі можуть бути значення від 1 до 5.

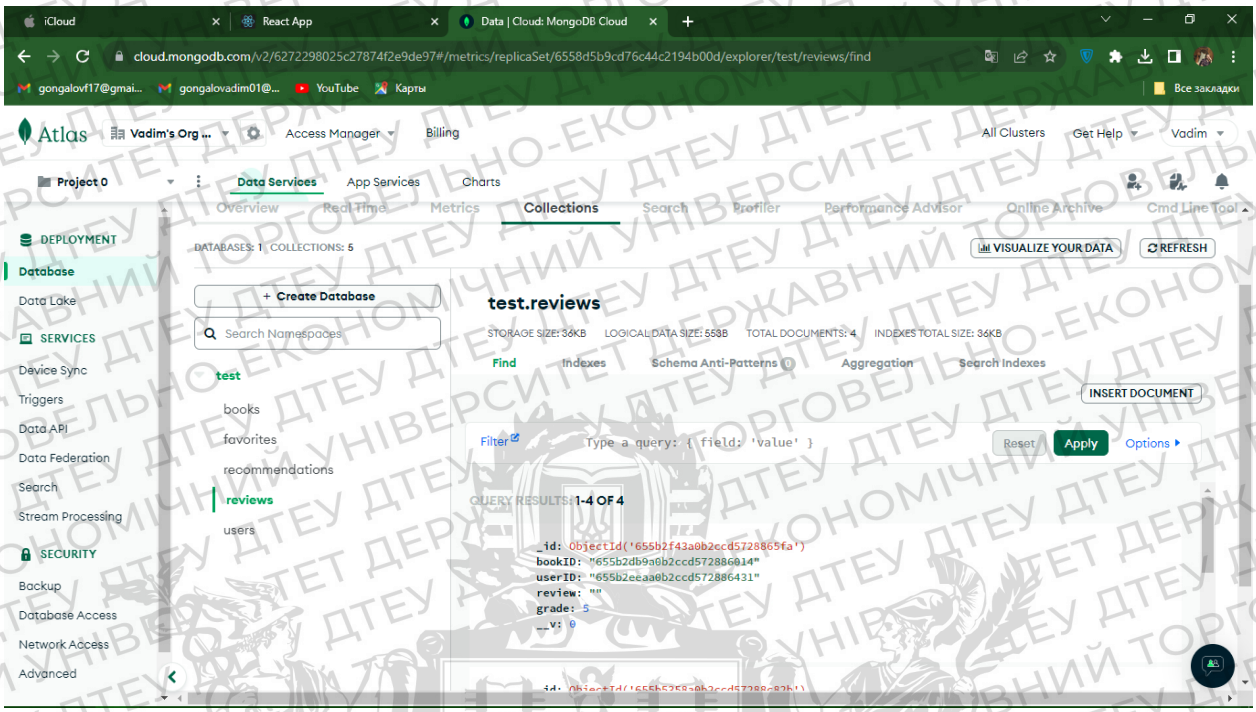

Рис.2.11. Таблиця з відгуками

І, відповідно, в останній колекції(рис. 2.12) знаходяться дані користувачів, які вже зареєструвалися і мають доступ до особистого кабінету. В даній колекції є дані, як і звичайних користувачів, так і адмінів. Без даної таблиці неможливо зайти чи зареєструватися у системі, вона зберігає всі дані реєстрації, такі як: “login”, “password” і якщо це профіль адміна, то характеристику “Admin”, яка дає право на керування адміністративною панелью.

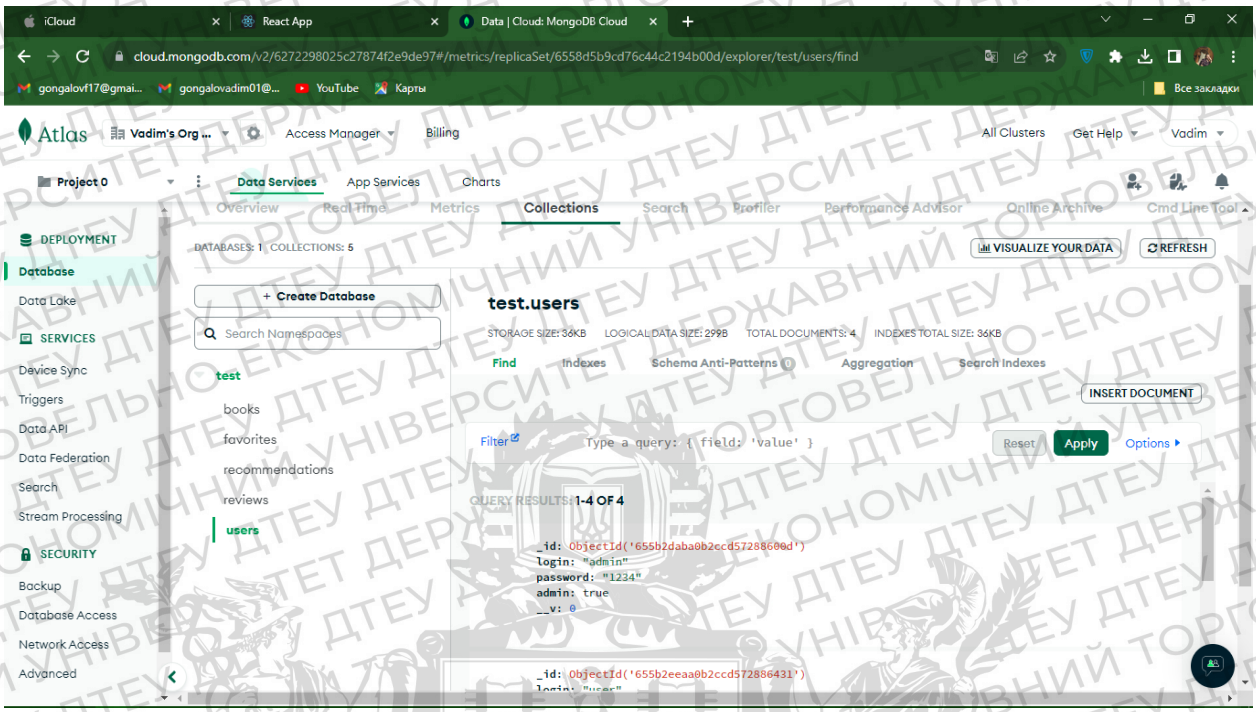

Рис. 2.12. таблиця з даними користувачів

2.4 Висновки до другого розділу

У цьому розділі ми здійснили огляд розробки архітектури он-лайн бібліотеки з рекомендаційним модулем. Ми провели функціональний аналіз системи та побудували дерево функцій, яке охоплює її основні завдання. У підрозділі 2.1.1 ми створили контекстну діаграму IDEF0 та її декомпозицію, що дозволило нам краще розібратися у взаємодії системи з її елементами. Ми також розглянули архітектуру системи та детально проаналізували математичне забезпечення проекту. В розділі, присвяченому інформаційному забезпеченню, ми розглянули дві моделі бази даних: концептуальну та фізичну, що визначають структуру та організацію даних в системі.

РОЗДІЛ 3. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ОН-ЛАЙН БІБЛІОТЕКИ З РЕКОМЕНДАЦІЙНИМ МОДУЛЕМ

3.1 Обґрунтування вибору програмних засобів

Для реалізації цього проекту використовувалися пакети засобів розробки, що складаються із:

- JS
- MongoDB
- React.JS
- Node.js
- Visual Studio Code

Я обрав ці інструменти з двох причин. По-перше, під час навчання в університеті мені вдалося вже працювати з ними. По-друге, всі ці засоби - це проекти, які постійно удосконалюються, розвиваються та користуються попитом у сучасних умовах. Кожен з цих інструментів легко та ефективно інтегрується один з одним, надаючи можливість швидкого та якісного створення інформаційного контенту та виконання поставлених завдань.

У ролі середовища програмування я вибрав Visual Studio Code з двох основних причин. По-перше, у мене вже був позитивний досвід роботи з продуктами компанії Microsoft, зокрема з Visual Studio. По-друге, Visual Studio Code підтримує всі вищезазначені засоби розробки.

Visual Studio Code — засіб для створення і редагування сучасних веб застосунків і програм для хмарних систем. Visual Studio Code розповсюджується безкоштовно і доступний у версіях для платформ Windows, Linux і OS X. VSCode розробляється в американській компанії Microsoft і успадкував ім'я від їх великовагового IDE - Visual Studio. Перший бета реліз був аж у квітні 2015 року. У 2016 році редактор став доступним для всіх бажаючих. Його головна перевага, це його швидкість та надійність. На відміну від Visual

studio, він не перевантажений усілякими розширеннями, а тому працює значно краще[25].

JavaScript – це кросплатформна об'єктно-орієнтована мова сценаріїв, що використовується для створення інтерактивних веб-сторінок (наприклад, використання складної анімації, створення кнопок, спливаючого меню тощо). Програми цією мовою називаються скриптами. Вони можуть бути написані прямо в HTML сторінки і запускатися автоматично під час її завантаження[26].

React JS — це відкритий JavaScript-фреймворк, а точніше, бібліотекою JavaScript, яка використовується для розробки інтерфейсів користувача. Він був створений компанією Facebook і швидко набув популярності серед розробників з усього світу. Реакт дозволяє ефективно створювати застосунки з високою продуктивністю і масштабованістю. Одним з ключових концепцій у React JS є компоненти. Вони представляють собою незалежні блоки коду, які відповідають за рендеринг певної частини користувацького інтерфейсу[27].

MongoDB — це база даних NoSQL, яка відрізняється від MySQL і PostgreSQL тим, що вона використовує нереляційну модель даних. Замість традиційних таблиць і стовпців, MongoDB використовує структуру, подібну до формату даних JSON, що дозволяє зберігати дані в більш гнучкому форматі. В MongoDB дані зберігаються у форматі BSON (Binary JSON), який дозволяє робити запити до бази даних за допомогою JSON-подібного синтаксису. Це може бути дуже зручно для розробників, які працюють з JavaScript, адже формат даних дуже подібний до того, з яким вони звикли працювати[28].

Node.js – середовище виконання коду JavaScript поза браузером. Ця платформа дозволяє писати серверний код для динамічних веб-сторінок та веб-застосунків, а також для програм командного рядка. За допомогою Node.js реалізується парадигма JavaScript для всього. Вона передбачає використання однієї мови програмування для розробки веб-застосунків замість застосування різних мов для роботи над фронтендом і бекендом[29].

3.2 Структура програмного забезпечення

Нижче я створив діаграму на якій схематично зображено структуру програмного забезпечення(рис. 3.1). Головна програма поділена на два розділи: app(front-частина) та server(back-частина). В кожному розділі я маю певні папки. В розділі app я маю сторінки самого проекту, які бачить користувач при взаємодії з програмою. До них відносяться сторінки з книгами, сторінка читача, розділ з описом бібліотеки та інші. А у розділі server розписана вся частина, яка включає в себе процес рекомендації книжок, створення нових користувачів, підтягування книг з бази даних, тощо.

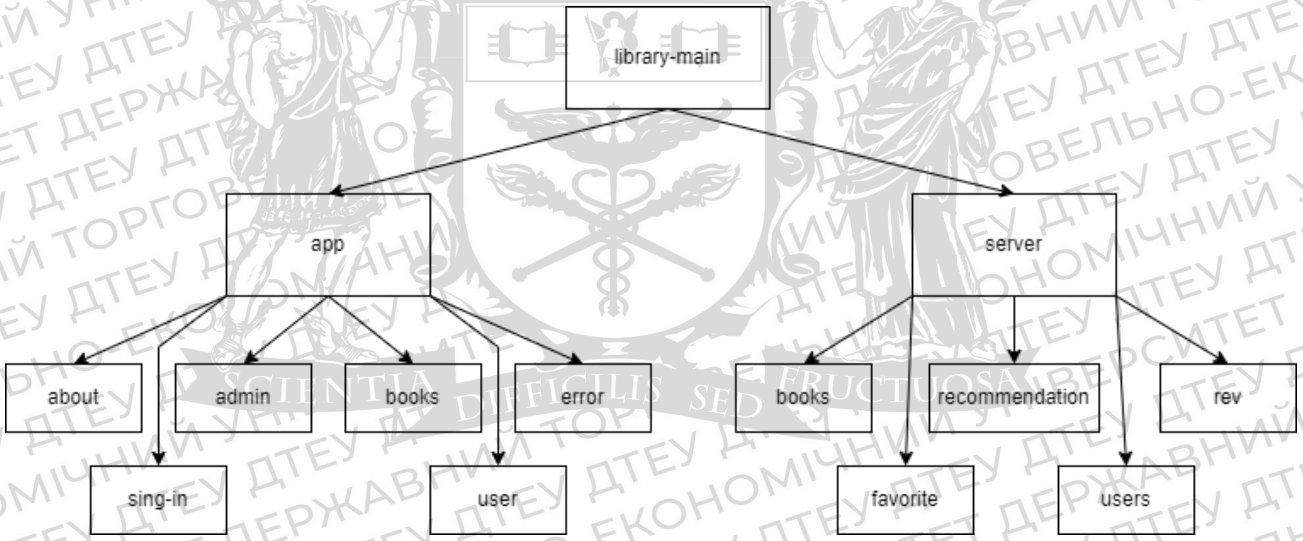

Рис. 3.1. Структура програмного забезпечення

Таблиця 3.1. *Специфікація програмного забезпечення*

Розділ	Модуль	Опис
App	about	Сторінка з інформацією про бібліотеку.
App	sing-in	Сторінка для входу в особистий кабінет
App	Admin	Сторінка адміністративної панелі, для управління бібліотекою
App	Books	Ця сторінка показує наявний каталог книг
App	User	На цій вкладці користувач може побачити свої рекомендації та улюблені книги
App	Error	Сторінка, яка виводить помилку при відсутності зв’язку з сервером
Server	Books	Частина, яка відповідає за підгрузку книг у систему
Server	Favorite	Відповідає за улюблені книги користувача
Server	recommendation	Відповідає за рекомендації користувачу
Server	users	Модуль, який відповідає за контролем реєстрації користувачів
Server	rev	Відповідає за відгуки

3.3 Керівництво користувача

При запуску проекту нас зустрічає стартова сторінка сайту(рис. 3.2). На ній користувач може зареєструватися у бібліотеці або ж, якщо він вже має особистий кабінет, увійти в нього.

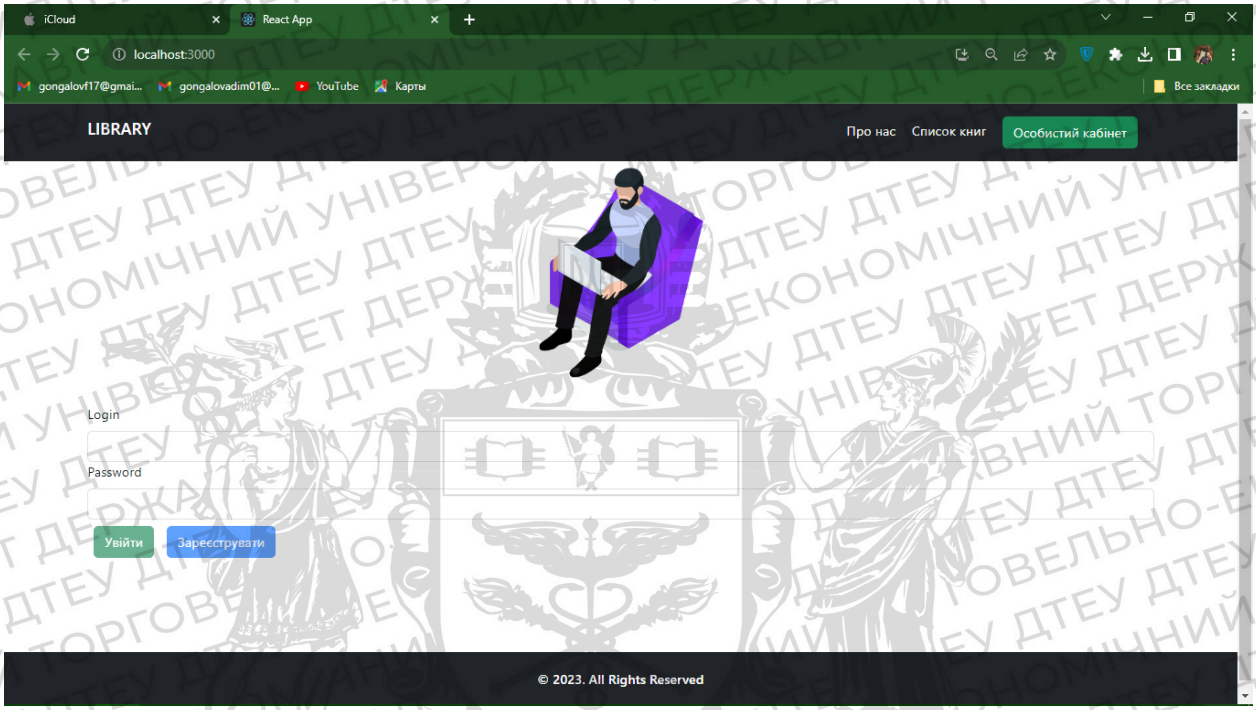

Рис. 3.2. Стартова сторінка проекту

Коли користувач, який не зареєстрований у системі, створює свій профіль йому випадає повідомлення про те, що він тепер зареєстрований у ній(рис. 3.3).

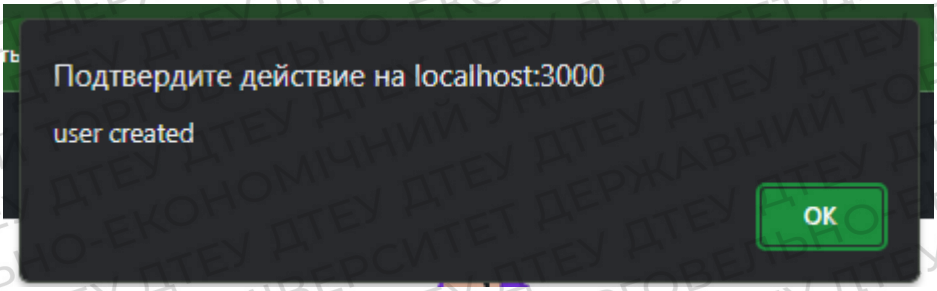

Рис. 3.3. Повідомлення про реєстрацію

Якщо ж у користувача немає особистого кабінету, але він намагається зайти, то йому буде показане відповідне повідомлення(рис. 3.4).

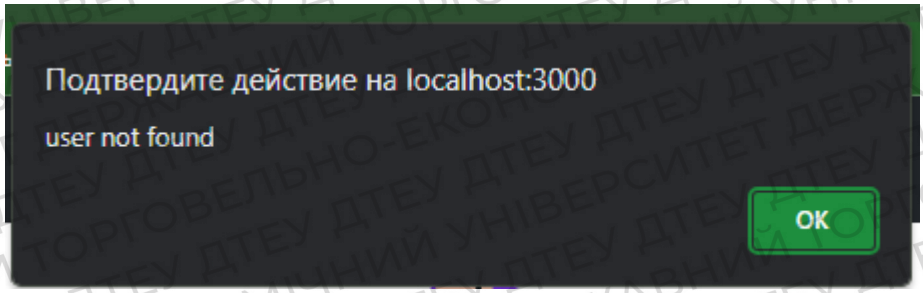

Рис. 3.4. Повідомлення про відсутність користувача у системі

Також на верхній панелі існує вкладка “про нас”, на який користувач може прочитати якусь загальну інформацію про бібліотеку(рис. 3.5).

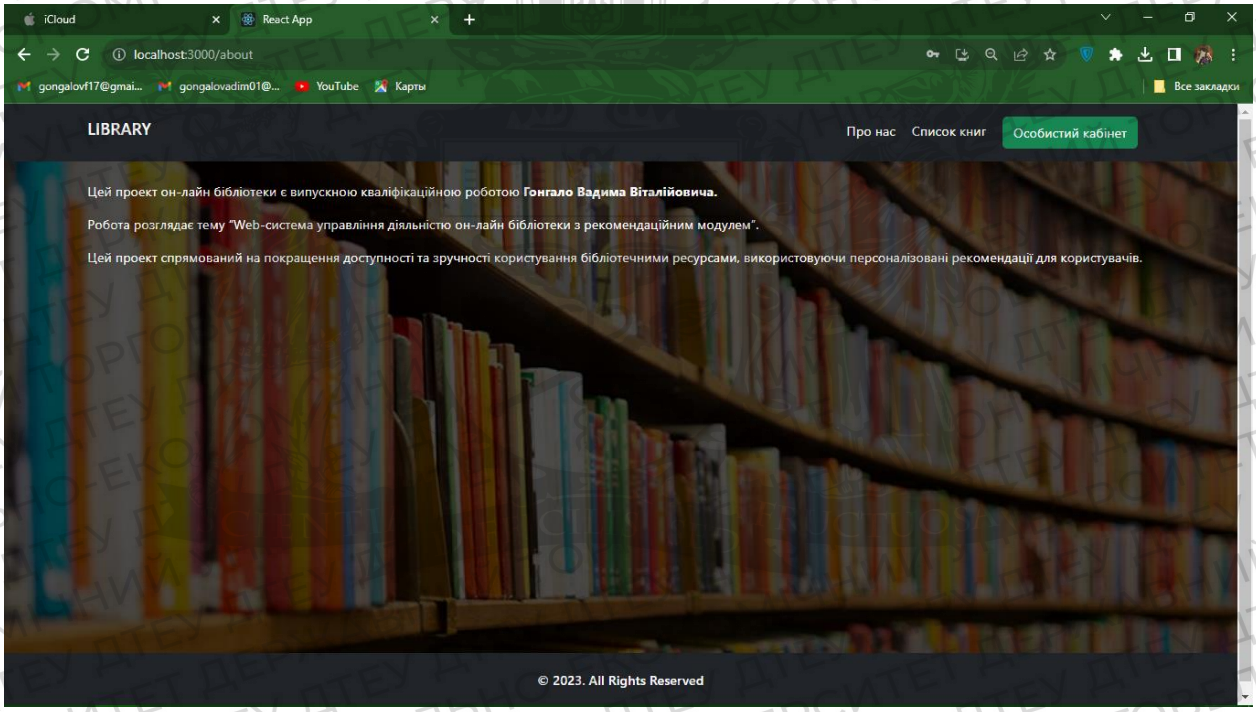

Рис 3.5. Вкладка “Про нас”

Перейдемо до основного розділу з книгами. Він відкривається при натисканні кнопки “список книг”(рис. 3.6). У цьому розділі користувач може ознайомитись з загальною інформацією про книги. Побачити їх обкладинки, прочитати короткий опис, подивитись автора. Якщо ж читача цікавить якась конкретна книга, він може скористатися пошуком на сайті(рис. 3.7)Також, при бажанні, читач може відкрити список з доступними жанрами(рис. 3.8) і вибрати потрібний йому(рис. 3.9).

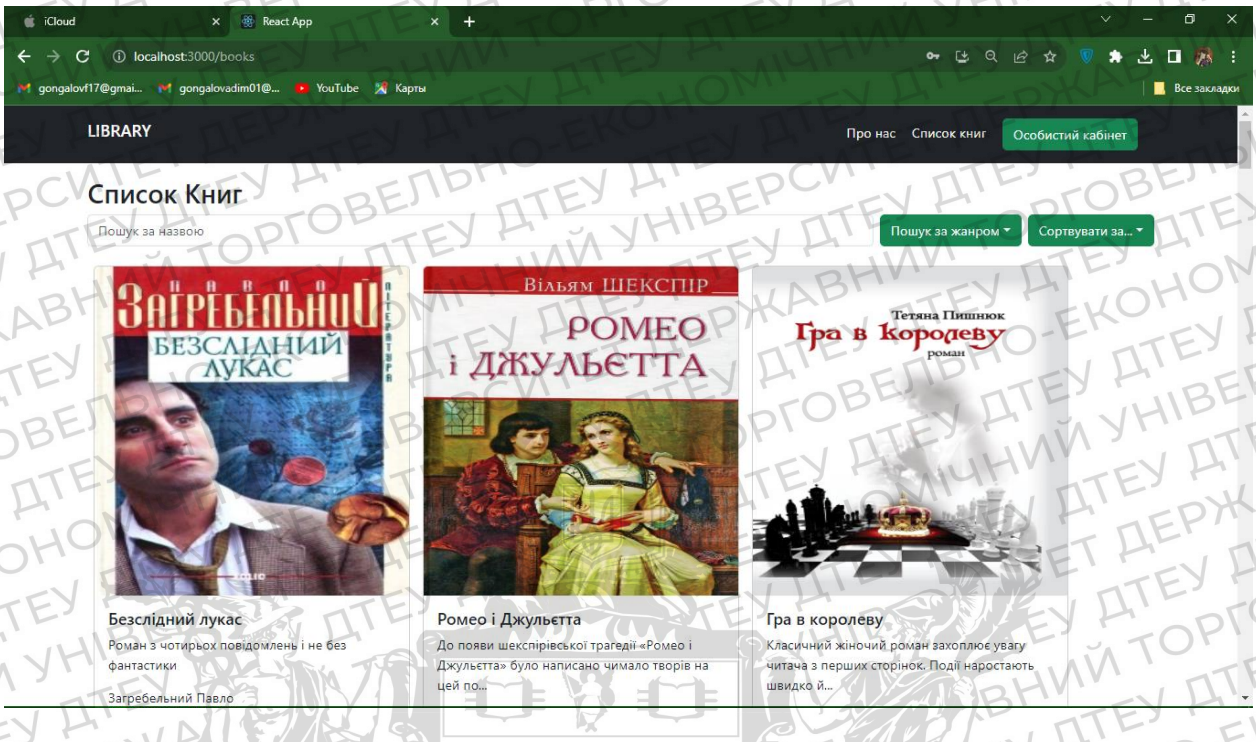

Рис. 3.6. Розділ “Список книг”

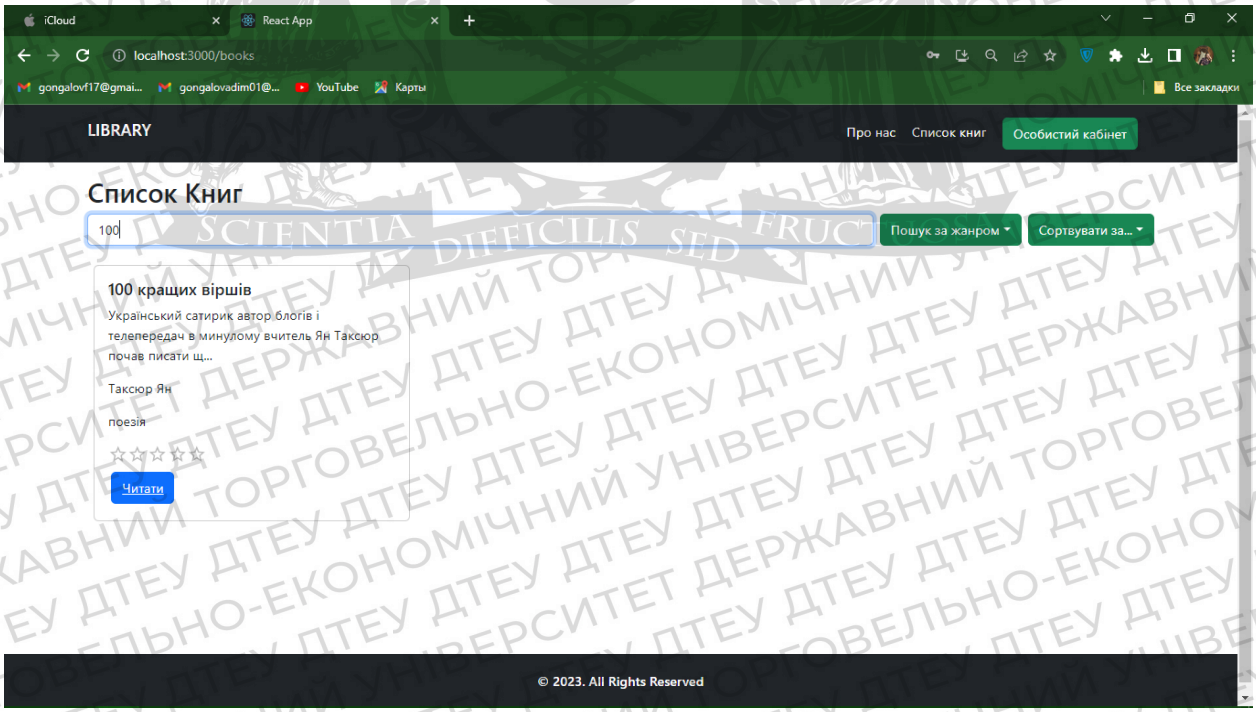

Рис.3.7. Пошук книги за назвою

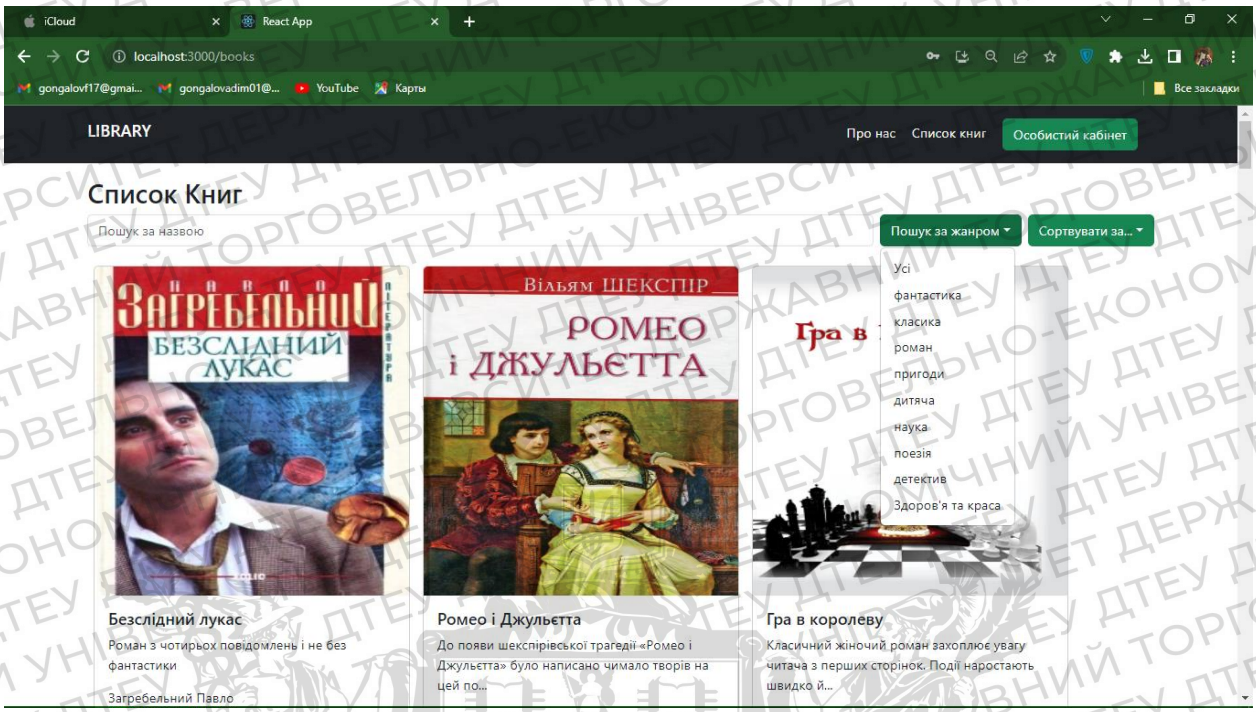

Рис.3.8. Список доступних жанрів

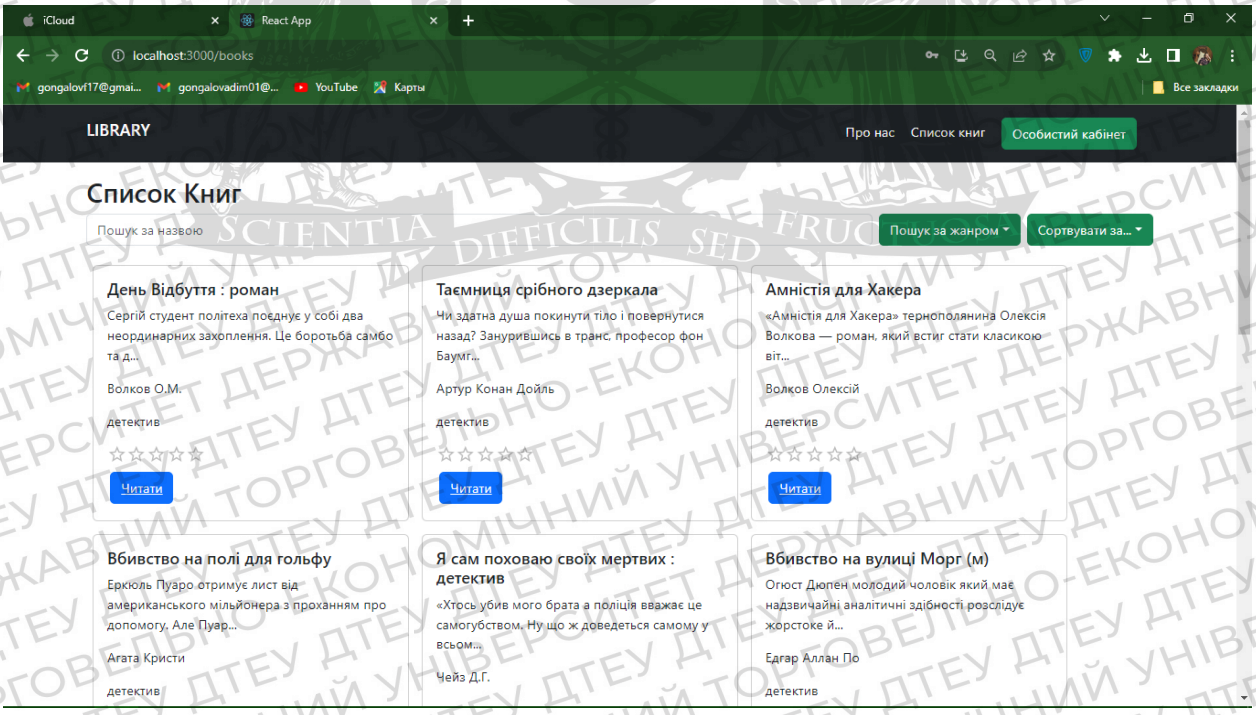

Рис.3.9. Книги жанру “детектив”

Натиснувши на потрібну користувачу книгу він може побачити більш розширений опис про неї, додати її до улюбленої або ж залишити відгук(рис. 3.10).

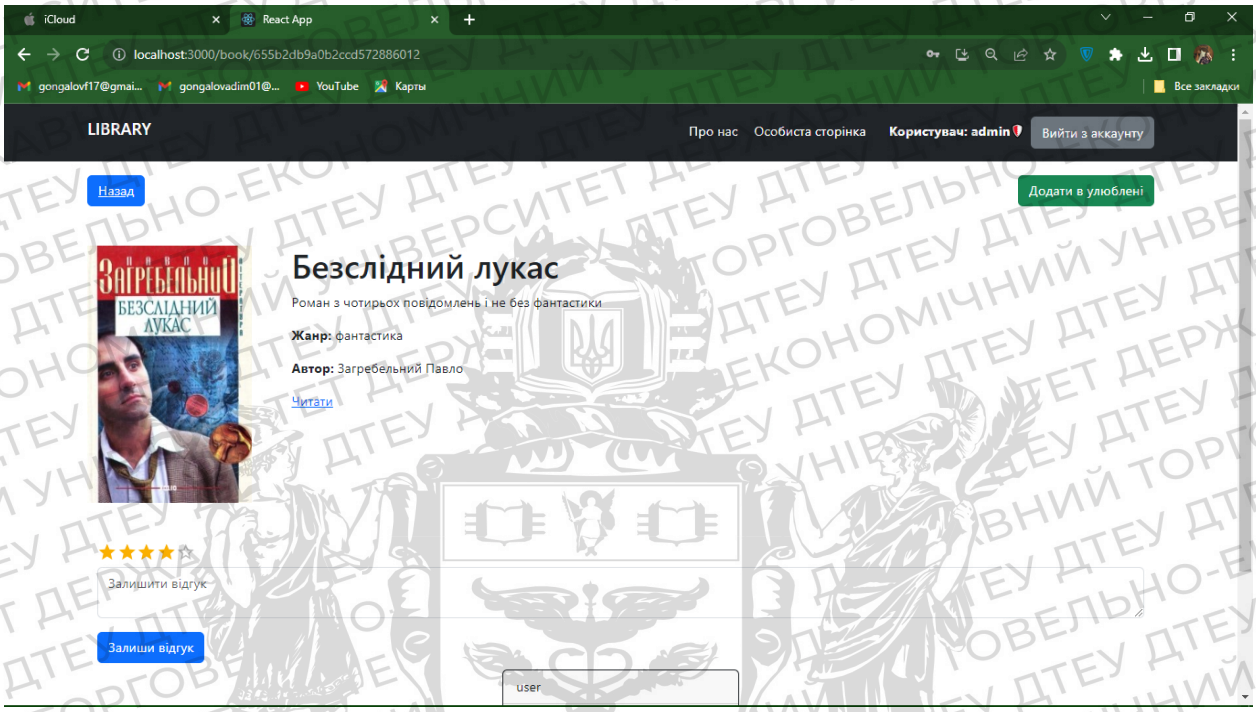

Рис.3.10. Сторінка вибраної книги

Натиснувши кнопку “читати”, система відкриє pdf-файл і читач зможе розпочати читання книги(рис.3.11).

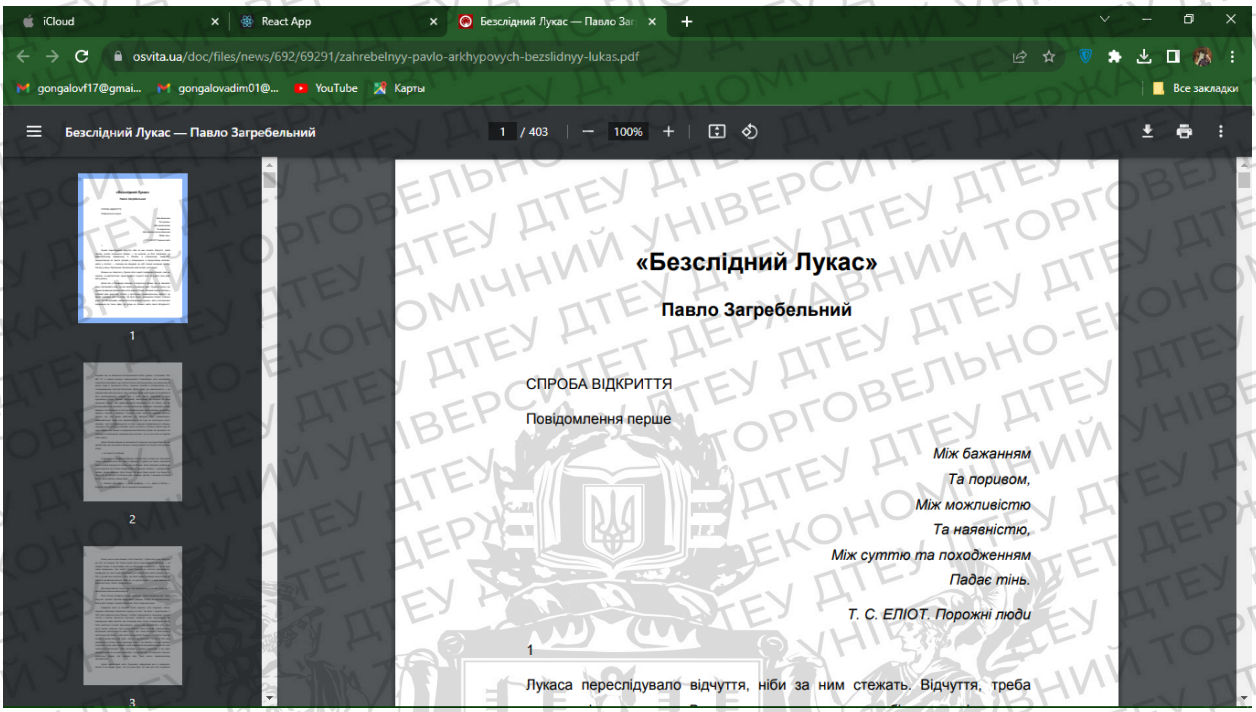

Рис.3.11. pdf-файл книги

Після того, як користувач прочитав книгу він може залишити відгук про неї та оцінити її за п’яти бальною шкалою(рис.3.12). Користувачу випаде відповідне повідомлення про це(рис.3.13).

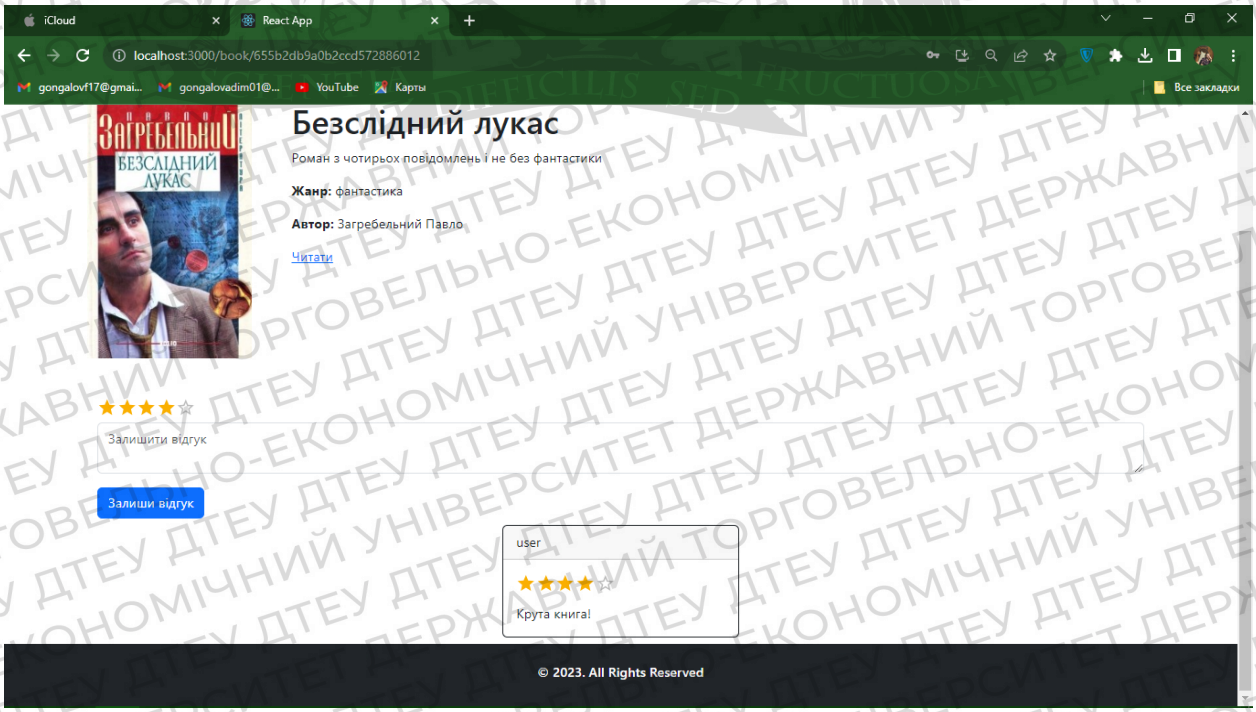

Рис.3.12. Написаний відгук про книгу

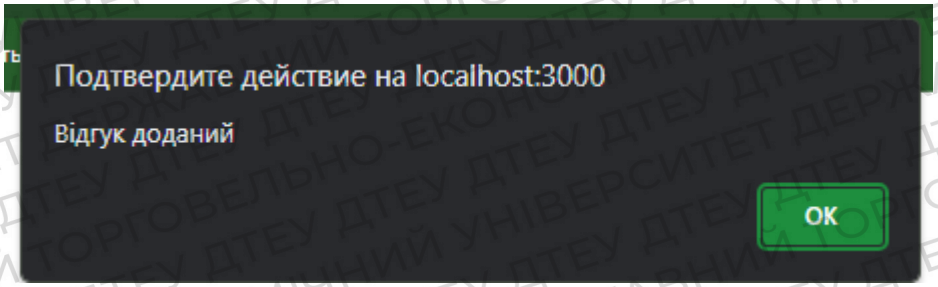

Рис 3.13. Повідомлення про доданий відгук

Тепер перейдемо до адміністративної панелі сайту(рис.3.14). У системі може бути створений тільки один профіль адміністратора. Це зроблено для того, аби звичайний користувач не зміг мати доступу до керування бібліотекою. Адміністратор може керувати книгами у бібліотеці. Додавати(рис. 3.15), видаляти або редагувати їх.

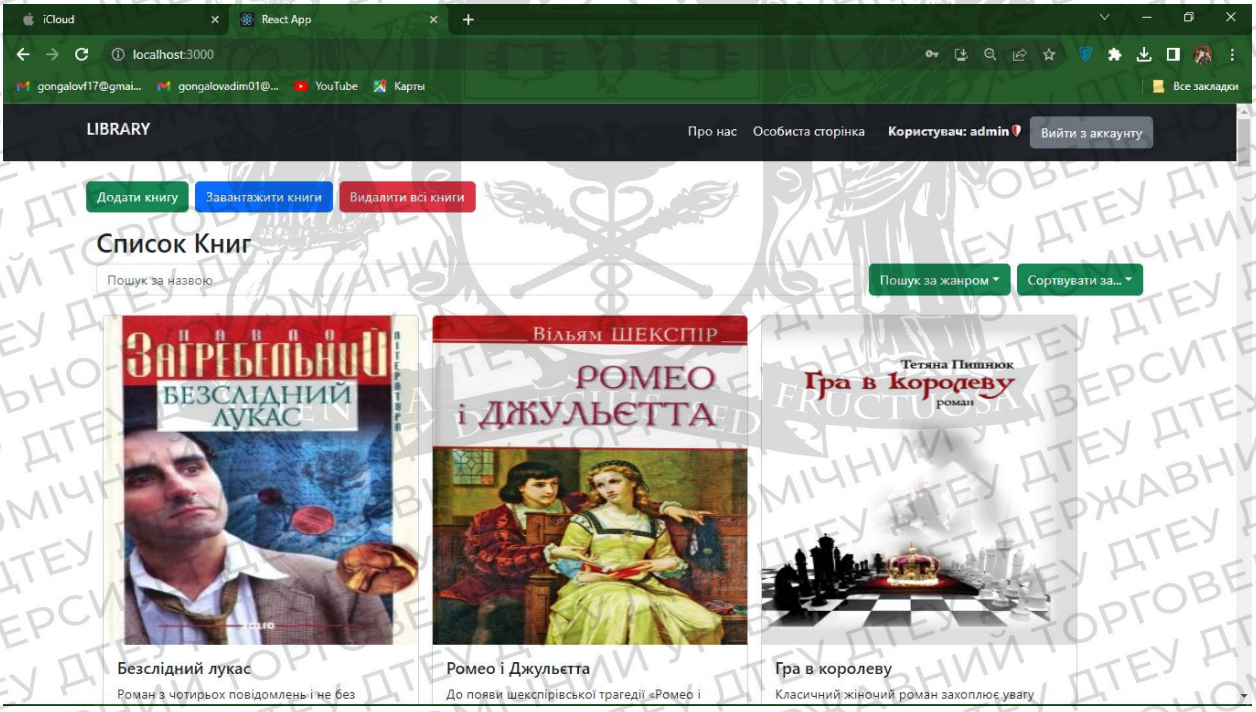

Рис. 3.14. Адміністративна панель

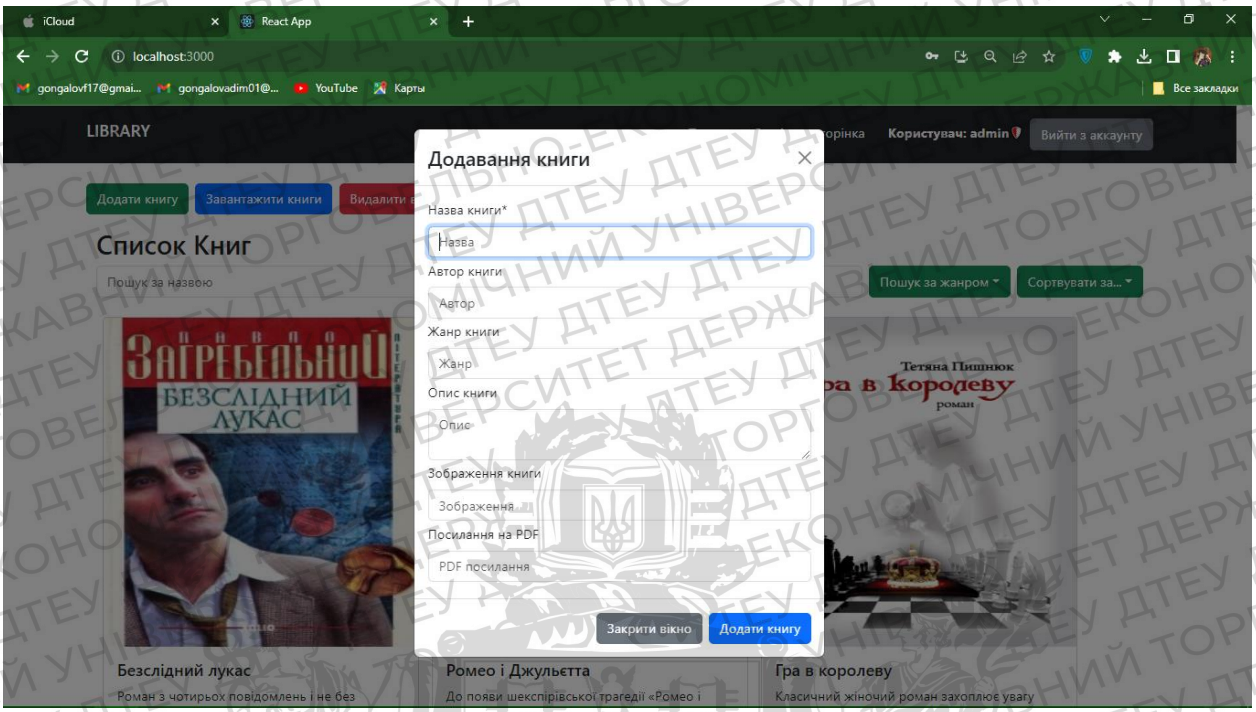

Рис.3.15. Форма для додавання книги

Також, адміністратор має право видаляти відгуки, оскільки повинна бути певна модерація на сайті(рис.3.16).

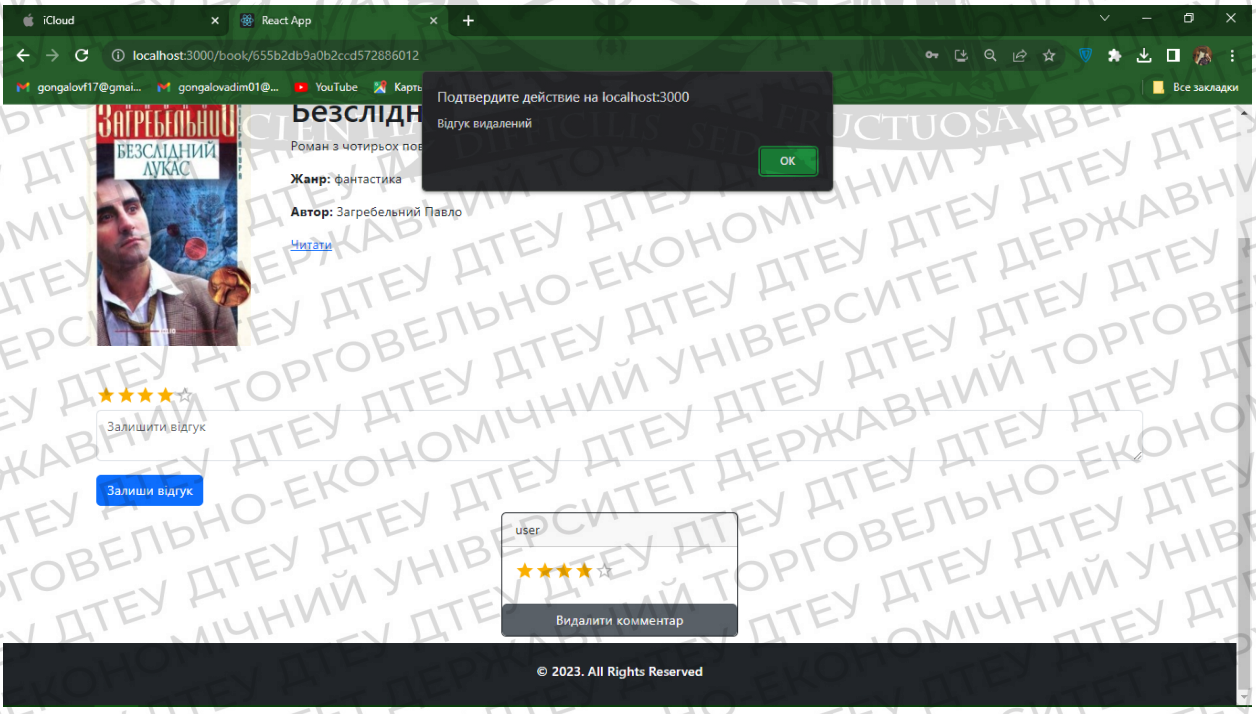

Рис. 3.16. Видалення відгуку про книгу

В особистому кабінеті користувача зустрічають персональні рекомендації та улюблені книги(рис 3.17) та історія перегляду книг(рис. 3.18).

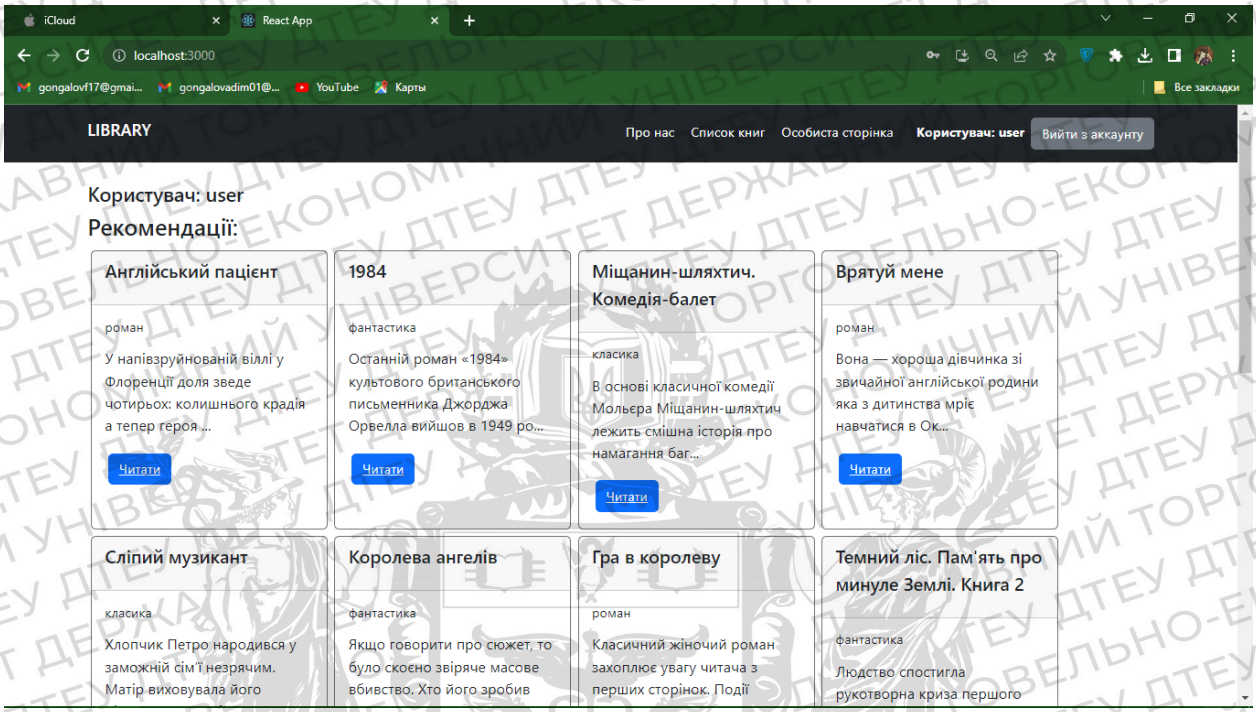

Рис.3.17. Рекомендації користувачу

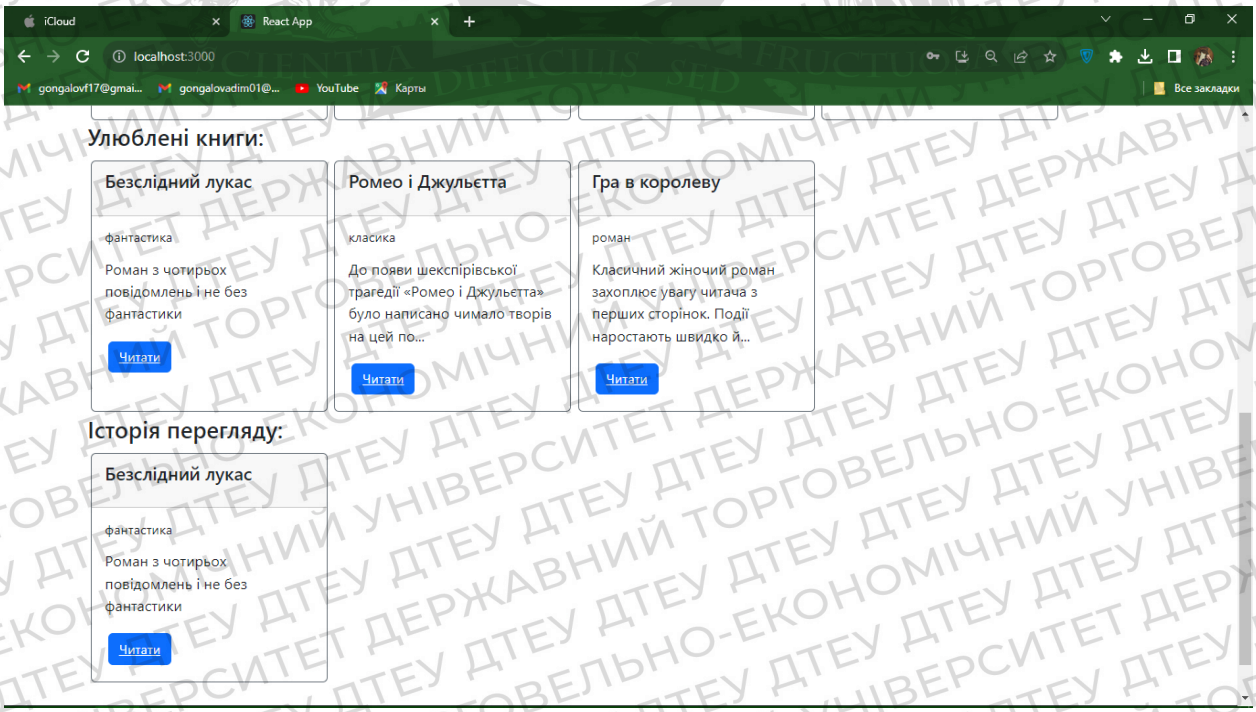

Рис.3.18. Історія перегляду

3.4 Висновки до третього розділу

У цьому розділі я висвітлив обрані програмні засоби для розробки проекту, обґрунтовуючи кожен вибір. Поклавши акцент на структуру програмного забезпечення, я також представив побудовану специфікацію програмних модулів системи.

Окрім того, докладно розглянута система в цілому, надано огляд її функціоналу та забезпечено відповідне керівництво користувача. Розгляд системи включає аналіз її ключових аспектів та особливостей, а також надається важлива інформація для зручного та ефективного використання. Ілюстрації та рисунки допомагають краще розуміти висвітлені аспекти системи та сприяють зручній навігації користувача в середовищі програми.

Висновки

У результаті випускної кваліфікаційної роботи було розроблено он-лайн бібліотеку з рекомендаційним модулем на основі функціонального аналізу предметного середовища. Під час аналітичного огляду літератури та існуючих рекомендаційних систем були сформульовані вихідні вимоги та задачі на розробку системи.

Далі, для побудови інформаційної системи було використано різноманітні інструменти, такі як функціональний аналіз та створення архітектури системи. Нарисована контекстна діаграма та її декомпозиція, а також розглянуто математичне та інформаційне забезпечення рекомендаційного модуля, які стали ключовими кроками у процесі розробки.

Вибір програмних засобів для розробки був обґрунтований на основі аналізу та врахування вимог проекту. Структура програмного забезпечення була сформульована з урахуванням оптимальної інтеграції всіх компонентів системи. Завершальним етапом було написання керівництва користувача, яке надає зрозумілі та детальні інструкції для використання розробленої он-лайн бібліотеки з рекомендаційним модулем.

Список використаних джерел

1. Olx.ua: [Електронний ресурс] // Режим доступу: https://www.facebook.com/olx.ua/
2. Megogo.ua: [Електронний ресурс] // Режим доступу: https://meta.ru/kino/wiki/chto-takoe-megogo-v-televizore-i-smartfone
3. Sweet.tv: [Електронний ресурс] // Режим доступу: https://eba.com.ua/member/sweet-tv/
4. Pollock, S. A rule-based message filtering system // ACM Transactions on Office Information Systems, 1988. Т. 6, Вип. 3. С. 232-254.
5. Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. Using collaborative filtering to weave an information tapestry. // Communications of the ACM, 1992. Т. 35, Вип. 12. С. 61-70.
6. Resnick, P., Varian, H.R. Recommender systems // Communications of the ACM, 1997. Т. 40, Вип. 3. С. 56- 58.
7. Brusilovsky, P. Methods and techniques of adaptive hypermedia // User Modeling and UserAdapted Interaction, 1996. Т. 6, Вип. 2. С. 87-129.
8. Pazzani, M.J. A framework for collaborative, content-based and demographic filtering // Artificial Intelligence Review, 1999. Т. 13. С. 393-408.
9. Ahn, H., Kim, K.J., Han, I. Mobile advertisement recommender system using collaborative filtering // Proceedings of the 2006 Conference of the Korea Society of Management Information Systems, 2006. С. 709-715.
10. A¨ımeur, E., Brassard, G., Fernandez, J.M., Onana, F.S.M. Alambic: a privacy-preserving recommender system for electronic commerce // International Journal of Infectious Diseases, 2008. Т. 7. Вип. 5. С. 307–334.
11. Golbeck, J. Generating predictive movie recommendations from trust in social networks // Trust Management, 4th International Conference, iTrust 2006, Pisa, Italy, May 16-19, 2006, Proceedings, 2006 . pp. 93–104.
12. Cortes C., Vapnik V. Support-vector networks // Machine Learning, 44

1995. Т. 20. С. 273-297.

13. Joachims T. Text categorization with support vector machines: Learning with many relevant features // Springer Berlin Heidelberg, 1998. С. 137-142.

14. Russell, S., Norvig, P. Artificial Intelligence: A Modern Approach (2nd ed.). Prentice Hall, 2003. 946 с.

15. Rokach, L., Maimon, O. Data mining with decision trees: theory and applications. World Scientific Pub Co, 2008. 305 с.

16. Tsoumakas, G.; Katakis, I. Multi-label classification: an overview. // International Journal of Data Warehousing & Mining, 2007. Т. 3. Вип. 3. С. 1-13.

17. Read, J.; Bernhard P.; Geoff H.; Eibe F. Classifier Chains for Multi-label Classification. // Machine Learning, 2011. Т. 85. Вип. 3. С. 333-359.

18. Tsoumakas, G., Vlahavas, I. Random k-labelsets: An ensemble method for multilabel classification // Machine Learning: ECML 2007, 2007. С. 406-417

19. Sinclair, J. The automatic analysis of corpora // Directions in Corpus Linguistics (Proceedings of Nobel Symposium 82), 1992. Berlin: Mouton de Gruyter, С. 379-397

20. Митрофанова О.А. Вимірювання семантичних відстаней як проблема прикладної лінгвістики // Структурна та прикладна лінгвістика. Міжвузівська збірка: журнал. Видавництво СПбДУ, 2008. Вип.7. С. 92-101. Laurene V. F., Fundamentals of neural networks. Prentice-Hall, Inc. Upper Saddle River, NJ, USA,1994. 476 с.

21. Mikolov T., Chen K., Corrado G., Dean J. Efficient Estimation of Word Representations in Vector Space, Researchgate, 2014. 12 с.

22. Levy, O., Goldberg, Y., Dagan, I. Improving Distributional Similarity with Lessons Learned from Word Embeddings // Transactions of the Association for Computational Linguistics, 2015. Т. 3. С. 211-225.

23. Adomavicius G. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions / G. Adomavicius, A.

Tuzhilin // IEEE Transactions on Knowledge and Data Engineering – 2005. – Vol. 17, No6. – P. 734- 749

24. В.В. Куриленко Розробка веб-додатку музичного аудіострімінгу з рекомендаційною системою // НД ТГУ - 2017 - Р. 7-15.
25. Visual studio Code: [Електронний ресурс] // Режим доступу: https://habr.com/ru/post/653465/
26. JavaScript: [Електронний ресурс] // Режим доступу: https://webkyrs.info/page/chto-takoe-javascript
27. React JS: [Електронний ресурс] // Режим доступу: https://cases.media/article/sho-take-react-js-yak-pochati-vivchati-reakt-navichki-dlya-react-developer
28. MongoDB: [Електронний ресурс] // Режим доступу: https://dan-it.com.ua/uk/blog/vse-shho-potribno-znati-pro-bazi-danih-dlja-pochatkivciv-mysql-postgresql-mongodb/#MongoDB
29. Node.js: [Електронний ресурс] // Режим доступу: http://blog.training.com/2016/09/about-nodejs-and-why-you-should-add.html

ДОДАТОК

Програмний код реалізації Web-додатку

BookController.js

```
import BookService from './BookService.js'
import BOOKS_LIST from './BOOKS_LIST.js'

class BookController {
  async create(req,res){
    try {
      const book = await BookService.create(req.body)
      res.json(book)
    } catch (error) {
      res.status(500).json(error.message)
    }
  }
  async createBigList(req,res){
    try {
      const book = await BookService.createBigList(BOOKS_LIST)
      res.json(book)
    } catch (error) {
      res.status(500).json(error.message)
    }
  }
  async getAll(req,res){
    try {
      const book = await BookService.getAll()
      res.json(book)
    } catch (error) {
```

```
      res.status(500).json(error.message)
    }
  }
  async getOne(req,res){
    try {
      const book = await BookService.getOne(req.params.id)
      res.json(book)
    } catch (error) {
      res.status(500).json(error.message)
    }
  }
  async update(req,res){
    try {
      const book = await BookService.update(req.body)
      res.json(book)
    } catch (error) {
      res.status(500).json(error.message)
    }
  }
  async delete(req,res){
    try {
      const book = await BookService.delete(req.params.id)
      res.json(book)
    } catch (error) {
      res.status(500).json(error.message)
    }
  }
  async deleteAll(req,res){
```

```
        try {
            const books= await BookService.deleteAll()
            res.json(books)
        } catch (error) {
            res.status(500).json(error.message)
        }
    }
}
export default new BookController()
```

BookService.js

```
import Book from './Book.js'

class BookService{
    async create(book){
        const createdBook = await Book.create(book)
        return createdBook
    }
    async createBigList(books){
        const createdBooks = await Book.create(books)
        return createdBooks
    }
    async getAll(){
        const books = await Book.find()
        return books
    }
    async getOne(id){
        if(!id) throw new Error('Problem with id')
        const book = await Book.findById(id)
```

```
        return book
    }
    async update(book){
        if(!book._id) throw new Error('Problem with id')
        const updatedBook = await Book.findByIdAndUpdate(book._id, book, {new:
true})
        return updatedBook
    }
    async delete(id){
        if(!id) throw new Error('Problem with id')
        const deletedBook = await Book.findByIdAndDelete(id)
        return deletedBook
    }
    async deleteAll(){
        const deletedBooks= await Book.deleteMany()
        return deletedBooks
    }
}

export default new BookService()
```

FavoriteController.js

```
import FavoriteService from "./FavoriteService.js"


class FavoriteController {
    async create(req,res){
        try {
            const favorite = await FavoriteService.create(req.body)
```

```
            res.json(favorite)
        } catch (error) {
            res.status(500).json(error.message)
        }
    }
    async getUsersFavorite(req,res){
        try {

            const favorite = await FavoriteService.getUsersFavorite(req.params.id)
            res.json(favorite)
        } catch (error) {
            res.status(500).json(error.message)
        }
    }

    async delete(req,res){
        try {
            const favorite = await FavoriteService.delete(req.body)
            res.json(favorite)
        } catch (error) {
            res.status(500).json(error.message)
        }
    }
}
export default new FavoriteController()
```

FavoriteService.js

```
import Favorite from './Favorite.js'
import Recommendations from '../recommendations/Recommendations.js'
```

```
import Book from '../books/Book.js'

class FavoriteService{
  async create(favorite){
    //work with recommendations - START
    //favorite = {bookID, userID}
    const book = await Book.findById(favorite.bookID)
    const recommendation = await Recommendations.findOne({
      userID: favorite.userID,
      genre: book.genre,
      author: book.author
    })
    if(!recommendation){
      Recommendations.create({
      userID: favorite.userID,
      genre: book.genre,
      author: book.author
      })
    }

    //work with recommendations - END
    const favoriteFind = await Favorite.findOne(favorite)
    if(!favoriteFind){
      const createdFavorite = await Favorite.create(favorite)
      return createdFavorite
    }

  }
```

```
  async getUsersFavorite(id){
    const favoriteList = await Favorite.find({userID:id})
    return favoriteList
  }

  async delete(favorite){
    const favoriteOne = await Favorite.findOneAndDelete({bookID:
favorite.bookID , userID: favorite.userID})
    return favoriteOne
  }

}

export default new FavoriteService()
```

RecommendationsService.js

```
import Recommendations from './Recommendations.js'
import Book from '../books/Book.js'

class RecommendationsService{
  async getRecommendations(id){
    const recommendations = await Recommendations.find({userID: id})
    if(recommendations){
      const genres = []
      recommendations.map(el=> el.genre).map(el=> genres.push({genre: el}))

      const autors = []
      recommendations.map(el=> el.author).map(el=> autors.push({author: el}))
```

```
        if(genres.length && autors.length){
          const BookListAutorAndGenre = await Book.find({$or: genres, $or: autors})
// aurot + genres
        const authorList = await Book.find({$or: autors}) // aurot
        const genreList = await Book.find({$or: genres}) // genres

        const AuthorsAndGenre_Authors_Genres =
[...[...BookListAutorAndGenre].map(el => el._id), ...[...authorList].map(el => el._id),
...[...genreList].map(el => el._id)]

        function countRepeatingElements(arr) {
         const counts = {};
         for (const el of arr) {
          counts[el] = counts[el] ? counts[el] + 1 : 1;
         }
         return Object.entries(counts).map(([el, count]) => ({
          _id: el,
          count: count
         }));
        }

        const REC_LIST =
countRepeatingElements(AuthorsAndGenre_Authors_Genres).map(el=> el._id)

        return REC_LIST
        }
      }
    }
```

```
}

export default new RecommendationsService()
```

RevController.js

```
import RevService from "./RevService.js";

class RevController{
  async create(req,res){
    try {
      const user = await RevService.create(req.body)
      res.json(user)
    } catch (e) {
      res.status(500).json(e.message)
    }
  }
  async getReviewsByBook(req,res){
    try {
      const user = await RevService.getReviewsByBook(req.params.id)
      res.json(user)
    } catch (e) {
      res.status(500).json(e.message)
    }
  }
  async getUserReviews(req,res){
    try {
      const user = await RevService.getUserReviews(req.params.id)
      res.json(user)
    } catch (e) {
```

```
            res.status(500).json(e.message)
        }
    }
    async deleteReview(req,res){
        try {
            const user = await RevService.deleteReview(req.params.id)
            res.json(user)
        } catch (e) {
            res.status(500).json(e.message)
        }
    }
}

export default new RevController()
```

UserController.js

```
import UserService from "./UserService.js";

class UserController{
    async create(req,res){
        try {
            const user = await UserService.create(req.body)
            res.json(user)
        } catch (e) {
            res.status(500).json(e.message)
        }
    }
    async login(req,res){
        try {
```

```
            const user = await UserService.login(req.params)
            res.json(user)
        } catch (e) {
            res.status(500).json(e.message)
        }
    }
    async getOne(req,res){
        try {
            const user = await UserService.getOne(req.params.id)
            res.json(user)
        } catch (e) {
            res.status(500).json(e.message)
        }
    }
}

export default new UserController()
```

UserService.js

```
import User from './User.js'


class UserService{
    async create(user){
        const existingUser = await User.findOne(user)
        if(existingUser) throw new Error("user login busy")
        let createdUser
        if(user.login == "admin") createdUser = await User.create({...user, admin: true})
        else createdUser = await User.create({...user, admin: false})
```

```
        return createdUser
    }
    async login(user){
        const findUser = await User.findOne({login: user.login})
        if(!findUser) throw new Error("user not found")
        if(findUser.password === user.password) return {...findUser,
password:"******"}
        else throw new Error("password false")
    }
    async getOne(id){
        if(!id)throw new Error("id false")
        const findUser = await User.findById(id)
        return findUser.login
    }
}

export default new UserService()
```

BooksItem.jsx

```
import React, { useEffect, useState } from 'react';
import Button from 'react-bootstrap/Button';
import Card from 'react-bootstrap/Card';
import { Link } from 'react-router-dom';
import AdminModalChangedBook from
'../../pages/admin/AdminModalChangedBook';
import { FUNC_DELETE_BOOK_BY_ID } from '../../API/API_books';
import { useDispatch, useSelector } from 'react-redux';
import { FUNC_GET_REVIEWS_BY_BOOK } from '../../API/API_reviews';
```

```
import { addHistory } from '../../store/user/userSlice'
import Rating from '@mui/material/Rating';

const BooksItem = (props) => {
  const user = useSelector(state=>state.user.user)
  const dispatch = useDispatch()
  const {_id, name, author,genre, description, image} = props
  const [grade, setGrade] = useState(0)

  const handleDeleteBook = () =>{
    FUNC_DELETE_BOOK_BY_ID(alert("DELETED"), _id)
  }

  const greadeRating = (grades) =>{
    if(grades.length > 0) {
      let gradeNumber = 0
      grades.map(g=> gradeNumber += g.grade)
      setGrade(gradeNumber / grades.length)
    }
  }

  const handleAddHistory = (bookID) => dispatch(addHistory(bookID))

  useEffect(()=>{
```

```
    FUNC_GET_REVIEWS_BY_BOOK(greadeRating, _id)
  },[])

  return (
    <Card className='m-2 custom-card' style={{ width: '24rem' }}>
      {image && <Card.Img variant="top" src={image} alt={"image:" + name}/>}
      <Card.Body>
        <Card.Title>{name}</Card.Title>
        <Card.Text>{description && description.length > 90 ? description.slice(0,
90) + "..." : description}</Card.Text>
        <Card.Text>{author}</Card.Text>
        <Card.Text>{genre}</Card.Text>
        <Rating name="read-only" value={grade} readOnly />
        <br/>
        <Button className='m-1' onClick={() => handleAddHistory(_id)}
variant="primary"><Link  className="text-white"
to={`/book/${_id}`}>Читати</Link></Button>
        {user.isAdmin && <Button className='m-1'
onClick={()=>handleDeleteBook()} variant="danger">Видалити книгу</Button>}
        {user.isAdmin && <AdminModalChangedBook book={props}/>}

      </Card.Body>
    </Card>
  );
};
```

```
export default BooksItem;
```

BooksList.jsx

```
import './bookList.style.css'
import BooksItem from './BooksItem';

const BooksList = (props) => {
  const {booksList} = props
  return (
    <div className='books-list'>
      <div className="container">
        <div className='d-flex flex-wrap'>
          {booksList.map(book => <BooksItem key={book._id}  {...book}/>)}
        </div>
      </div>
    </div>
  );
};

export default BooksList;
```