

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА»
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
«ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ В ЕКОНОМІЦІ»

Кафедра інформаційних систем в економіці

галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

Форма навчання: денна

КВАЛІФІКАЦІЙНИЙ БАКАЛАВРСЬКИЙ ПРОЕКТ
на тему
ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ З НАДАННЯ
ЛІТЕРАТУРНИХ РЕКОМЕНДАЦІЙ

здобувача Гаршиної Анни Леонідівни_____

Науковий керівник:

к.е.н., доцент

_____ Козак І.А.

Кваліфікаційний бакалаврський
проект допущений до захисту в
Екзаменаційній комісії з атестації
здобувачів вищої освіти

В.о.завідувача кафедри:

к.е.н., доцент.

_____ Тішков Б.О.

Київ 2022

**Міністерство освіти і науки України
Державний вищий навчальний заклад
«Київський національний економічний університет
імені Вадима Гетьмана»**

**Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці**

галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»
денна форма навчання

Затверджую:

в.о. завідувача кафедри _____ Тішков Б.О.

“ ____ ” _____ 2022 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на кваліфікаційний бакалаврський проект

**на тему: «Проектування інформаційної системи з надання літературних
рекомендацій»**

Гаршиної Анни Леонідівни

Тему кваліфікаційного бакалаврського проекту затверджено наказом ректора від «21» грудня 2021 р. № 1916-ст.

Кваліфікаційний бакалаврський проект виконується на матеріалах ТОВ «Аксон Технолоджіс»

План кваліфікаційного бакалаврського проекту

Розділ I . Характеристика та аналіз предметної галузі

Розділ II. Розробка вимог і моделювання інформаційної системи

Розділ III. Проектування та реалізація компонентів системи

Об'єкт дослідження: Процес надання літературних рекомендацій

Предмет дослідження: Проектування та розробка ІС з надання літературних рекомендацій за допомогою сучасних технологій веб-програмування.

Мета кваліфікаційного бакалаврського проекту: Дослідження сучасних технологій веб-розробки та створення ІС з надання літературних рекомендацій.

Конкретні завдання, які студент повинен виконати для досягнення поставленої мети:

У розділі І. Надати характеристику предметної області та об'єкта дослідження. Проаналізувати практичний досвід використання ІС в предметній галузі.

У розділі ІІ. Провести аналіз та розробити специфікацію вимог до системи. Описати постановку задачі та алгоритм функціонування системи. Здійснити моделювання системи.

У розділі ІІІ. Спроектувати та реалізувати компоненти системи.

Завдання підготував

науковий керівник

Козак І.А.

«_____» _____ 2022р.

Завдання одержав

студент

Гаршина А.Л.

(прізвище, ім'я, по-батькові)

«_____» _____ 2022р.

АНОТАЦІЯ

випускного бакалаврського проекту студентки 4 курсу

Навчально-наукового інституту «Інститут інформаційних технологій в економіці»

Гаршиної Анни Леонідівни, виконаної на тему:

«Проектування інформаційної системи з надання літературних рекомендацій»

Київ: кафедра інформаційних систем в економіці, 2022 р.

Випускний бакалаврський проект присвячений актуальній темі автоматизації процесу надання літературних рекомендацій.

Випускний бакалаврський проект складається з трьох розділів, логічно пов'язаних між собою.

В першому розділі дана характеристика предметної галузі, об'єкта та предмета дослідження. Крім того, тут наведено перелік існуючих програмних засобів даної предметної галузі та доведено доцільність створення нового програмного рішення.

Другий розділ присвячений обґрунтуванню методологій проектування та моделювання системи, розробці та аналізу вимог до системи. Також тут описані постановка задачі та алгоритм функціонування системи.

У третьому розділі наведено інформаційне, технічне програмне та організаційне забезпечення для системи з надання літературних рекомендацій. Тут детально розглянута структура інформаційного забезпечення, обґрунтований вибір комплексу технічних засобів та а також програмне забезпечення, що було використано при розробці серверної та клієнтської частин системи.

У висновках містяться підсумки роботи та рекомендації щодо впровадження інформаційної системи з надання літературних рекомендацій до використання у повсякденному житті.

РЕФЕРАТ

Випускний бакалаврський проект містить 79 сторінок, 9 таблиць, 81 рисунок, список літератури з 12 найменувань, 5 додатків.

ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ З НАДАННЯ ЛІТЕРАТУРНИХ РЕКОМЕНДАЦІЙ

Перелік ключових слів: ІНФОРМАЦІЙНА СИСТЕМА, ВЕБ-ПРОГРАМУВАННЯ, БАЗА ДАНИХ, КОРИСТУВАЦЬКИЙ ІНТЕРФЕЙС, КЛІЄНТ-СЕРВЕРНА АРХІТЕКТУРА, ЛІТЕРАТУРНА РЕКОМЕНДАЦІЯ.

Предметом дослідження є проєктування та розробка ІС з надання літературних рекомендацій за допомогою сучасних технологій веб-програмування.

Об'єктом дослідження виступає процес надання літературних рекомендацій.

Мета випускного бакалаврського проекту полягає використанні сучасних технологій веб-розробки при створенні ІС з надання літературних рекомендацій.

Завданнями випускного бакалаврського проекту є: дослідження сучасних підходів та технологій веб-розробки шляхом проєктування та розробки клієнт-серверної архітектури ІС з надання літературних рекомендацій.

Апаратні та програмні засоби, що використовувались при проєктуванні та розробці: Visual Studio Code, ZSH, JavaScript, TypeScript, Node.js, база даних SQLite, Prisma, Apollo, React.js, мова запитів GraphQL, HTML, CSS.

Результати досягнуті в процесі роботи висвітлюють найсучасніші технології веб-програмування, а також актуальність автоматизації процесу надання літературних рекомендацій.

Одержані результати можуть бути використані у повсякденному житті задля отримання нових літературних рекомендацій користувачем будь-якого віку чи професії.

Рік виконання випускного бакалаврського проекту: 2022 р.

Рік захисту випускного бакалаврського проекту: 2022 р.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ ВЕЛИЧИН І ТЕРМІНІВ	7
ВСТУП	8
РОЗДІЛ 1. ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ	11
1.1. Характеристика предметної галузі та об'єкта дослідження.....	11
1.2. Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі.....	13
РОЗДІЛ 2. РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ	17
СИСТЕМИ	17
2.1. Аналіз і специфікація вимог до інформаційної системи	17
2.1.1. Бізнес-вимоги	17
2.1.2. Нефункціональні вимоги.....	18
2.1.3. Функціональні вимоги.....	19
2.2. Постановка та алгоритм розв'язання задачі.....	21
2.2.1. Характеристика задачі.....	21
2.2.2. Вихідна інформація	22
2.2.2. Вхідна інформація.....	22
2.2.4. Алгоритм розв'язання задачі	23
2.3. Моделювання інформаційної підсистеми	24
2.3.1. Моделювання поведінки системи	24
2.3.2. Модель структури системи	26
2.3.3. Прототипування користувацького інтерфейсу системи	27
РОЗДІЛ 3. ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ	29
3.1. Інформаційне забезпечення	29
3.2. Технічне забезпечення.....	39
3.3. Програмне забезпечення	40
3.4. Результати реалізації інформаційної системи.....	47
ВИСНОВКИ.....	61
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	63
ДОДАТКИ.....	64
.....	66

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ ВЕЛИЧИН І ТЕРМІНІВ

ІС – інформаційна система

ОС – операційна система

UI/UX – user interface/ user experience, інтерфейс користувача

QL – query language, мова запитів

СКБД – система керування базами даних

ACID – atomicity, consistency, isolation, durability (атомарність, узгодженість, ізольованість, надійність); опис вимог до транзакційної системи

ORM - Object-Relational Mapping, технологія, яка пов'язує бази даних із концепціями об'єктно-орієнтованих мов програмування

CRUD – create, read, update, delete (створення, читання, оновлення та видалення – основні операції баз даних)

QL – query language, мова запитів

API – application programming interface, прикладний програмний інтерфейс

REST - representational State Transfer, набір правил для написання архітектури серверу

HTTP - hypertext transfer protocol, протокол передачі гіпертексту

HTTPS – secure hypertext transfer protocol, протокол безпечної передачі гіпертексту

HTML – hypertext markup language, гіпертекстова мова розмітки

CSS – cascading style sheets, каскадні таблиці стилів

XML - eXtensible Markup Language, роширена мова розмітки

JSON - JavaScript Object Notation, текстовий формат обміну даними

JWT – JSON web tokens, веб-токени формату JSON

VCS – Version Control System

UML - Unified Modeling Language, уніфікована мова моделювання

IDE - Integrated development environment, середовище для розробки

ZSH – Z shell, командний рядок систем UNIX

ВСТУП

Тема дослідження та проектування. Література завжди була, є і буде невід’ємною частиною життя людей. За допомогою літератури люди здобувають нові знання, розвиваються, поринають у нові захоплюючі світи, знаходять однодумців та просто приємно проводять час. Читанням зацікавлені представники всіх професій та всіх вікових категорій: від керівників підприємств до студентів.

Актуальність дослідження та проектування. Сьогодні дуже важко зупинити свій вибір на певній книжці: розмаїття тем, авторів та видавництв пропонує безліч варіантів. Саме тому надзвичайно актуальним рішенням стало створення інформаційної системи, яка буде надавати користувачам рекомендації книг за обраним розділом у вигляді стислого огляду на них. Користувачі системи матимуть змогу зекономити час на пошуки книги з розділу, що їх зацікавив та перейти безпосередньо до читання. Отже, використання ІС з надання літературних рекомендацій для оптимізації пошукових та вибіркового процесів – дуже зручний варіант.

Окрім того, надзвичайно актуальним наразі вважається напрямок веб-розробки. Веб-додатки можуть охопити будь-кого та будь-де, на будь-якому пристрої. А оскільки платформою для якої призначені веб-додатки є всесвітня мережа, яка неупинно розвивається та не контролюється жодною компанією світу, веб-програмування є вкрай цікавим та релевантним напрямком у створенні якісного та доступного програмного забезпечення.

На етапі збору інформації було проаналізовано існуючі ІС з надання літературних рекомендацій. Виявилось, що деякі лідери серед діючих рішень були розроблені для роботи лише на базі мобільних пристроїв, деякі мали застарілу дизайн-систему, а більшість із них пропонували лише платну підписку.

Після аналізу існуючих ІС та особливостей їх роботи, було вирішено:

- спроектувати та реалізувати обрану ІС для роботи у браузері з метою розширення бази користувачів, які могли б користуватись можливостями системи як з мобільних пристроїв, так і з комп'ютерів без необхідності завантажування мобільного застосунку;
- розробити унікальний, зручний та актуальний дизайн для системи;
- зробити систему безкоштовною для користування.

Об'єкт дослідження. Об'єктом дослідження є процес надання літературних рекомендацій.

Предмет дослідження. Предметом дослідження є проектування та розробка ІС з надання літературних рекомендацій за допомогою сучасних технологій веб-програмування.

Мета дослідження та проектування. Метою кваліфікаційної роботи є дослідження сучасних технологій веб-розробки та створення ІС з надання літературних рекомендацій.

Завдання. Для досягнення мети поставлено такі завдання:

- Дослідити сучасні підходи та технології веб-розробки;
- Створити список функціональних, нефункціональних та бізнес-вимог до системи;
- Обрати актуальний набір технологій та мов програмування для реалізації клієнтської та серверної частин ІС;
- Розробити дизайн інтерфейсу для ІС;
- Спроектувати базу даних та заповнити деякі її таблиці тестовими даними;
- Розробити серверну частину ІС;
- Розробити клієнтську частину ІС;
- Налаштувати контакт між клієнтською та серверною частинами ІС;
- Провести тестування розробленої системи на відповідність поставленим вимогам.

Методи дослідження та відповідний інструментарій. Розробці програмного продукту передувало проектування системи, визначення основних вимог та логіки роботи веб-додатку, розробка актуального та практичного дизайну, серверної частини, клієнтської частини та написання функцій-розпізнавачів для формування запитів на сервер.

Для успішної реалізації програмного продукту було використано API-first стратегію. Вона полягає у тому, що в першу чергу було реалізовано API, що враховує усі вимоги системи, а тоді на отриманий результат надбудовувано власне сам додаток. Такий підхід значно спростив процес розробки та кількість змін, що вносились до системи.

В якості інструменту створення програмного засобу було обрано Visual Studio Code (версія 1.45.1) – безкоштовний та вільно поширюваний редактор початкового коду для створення та редагування сучасних веб-застосунків від Microsoft, командний рядок zsh. Більшість програмного коду написана мовою JavaScript (з використанням типізованої мови TypeScript, додаткових пакетів та бібліотек), а для реалізації серверної частини використовувались технології SQLite та NodeJS (Prisma). В якості мови запитів було задіяно новітню технологію GraphQL, зокрема технологію Apollo Server.

Теоретична та практична значущість роботи. Програмний продукт «BookOpinion» може широко застосовуватися користувачами будь-якого віку та професії задля отримання літературних рекомендацій по розділам або генерування випадкових рекомендацій.

РОЗДІЛ 1. ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1. Характеристика предметної галузі та об'єкта дослідження

Важливим аспектом забезпечення розвитку є можливість ознайомлюватись з різними думками, аналізувати та формувати свій власний погляд. Зазвичай люди отримують літературні рекомендації у соціумі: від членів родини, друзів, колег, знайомих тощо. Але було б зручно автоматизувати і цей процес.

Об'єктом дослідження бакалаврського проекту є процес надання літературних рекомендацій. Розробка веб-додатку з надання літературних рекомендацій (у вигляді стислих оглядів на книги) – релевантна та доцільна задача, оскільки читання книжок є актуальним у будь-який час.

Предметом дослідження є проєктування та розробка ІС з надання літературних рекомендацій за допомогою сучасних технологій веб-програмування. Сьогодні на початкових стадіях розробки програмного продукту часто постає вибір між мобільним та веб-додатком. Результат цього вибору впливає як на роботу розробників, так і на досвід кінцевого користувача. Мобільні та веб-додатки мають чимало спільного та відмінного, у кожного свої переваги та недоліки. Тому вибір зазвичай виконується на основі конкретних цілей кінцевого продукту.

Веб-додатки мають ряд переваг та недоліків.

Переваги:

- Не потребують завантаження чи встановлення. Веб-додатки функціонують у браузері
- Легкі у підтримці. Мають єдину кодову базу і не залежать від мобільної платформи
- Не потребують оновлення
- Легкі та швидкі у розробці

- Не потребують підтвердження у крамниці застосунків, а тому можуть бути запущені швидко

Недоліки:

- Не працюють офлайн
- Повільніші та мають менший спектр можливостей, ніж мобільні додатки
- Менш помітні, так як не містяться у певній базі для пошуку, такій як крамниці застосунків
- Менш безпечні.

Існує думка, що веб-додатки ніколи не зможуть конкурувати з мобільними додатками, якщо розглядати їх надійність, швидкість та залучення користувачів. Мобільні додатки завжди присутні на домашніх екранах, папках та панелях завдань. Вони працюють незалежно від підключення до мережі і не потребують браузера. Вони можуть читати та записувати файли до локальної файлової системи, отримувати доступ до апаратного забезпечення, підключеного через USB або Bluetooth, взаємодіяти з даними, що зберігаються на пристрої, як-от контакти та події календаря. Можна керувати частиною функціоналу, навіть перебуваючи в іншому додатку. Мобільні додатки відчують себе частиною пристрою, на якому вони працюють. Окрім цього, для веб-додатків важче будувати бізнес-модель. Проте веб-додатки є більш доступними: набагато простіше і швидше відвідати веб-сайт, ніж встановлювати додаток.

Та якщо говорити про ІС з надання літературних рекомендацій, то вдалим рішенням стане його реалізація у форматі веб-додатку. Для цього є ряд причин:

- Доступність та швидкість – користувачам не треба буде завантажувати додаток на свій пристрій та займати цим обсяги своєї оперативної пам'яті, бо все, що потрібно для користування ІС – це браузер та доступ до Інтернету;

- Оскільки база даних з рекомендаціями буде часто збільшуватись, користувачам не доведеться щоразу оновлювати мобільний додаток, щоб отримати доступ до актуальних даних. Варто просто оновити сторінку у веб-додатку і нові дані з'являться;
- Розробникам буде легше підтримувати існуючу кодову базу та розробляти новий функціонал. Звісно, після розростання системи, можна буде розробити мобільний додаток з таким самим або оновленим користувацьким інтерфейсом та мігрувати туди поточну базу даних.

1.2. Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі

Наразі існує досить багато інформаційних систем, які надають літературні рекомендації. Всі вони мають різне призначення, різну цільову аудиторію та контент. Розглянемо деякі з них.

IC GetAbstract [1] є одним із великих гравців у цій галузі. І не дарма: він має найширший каталог із понад 20 000 огляді на книги. Інтерфейс головної сторінки IC GetAbstract можна побачити на Рис.1.

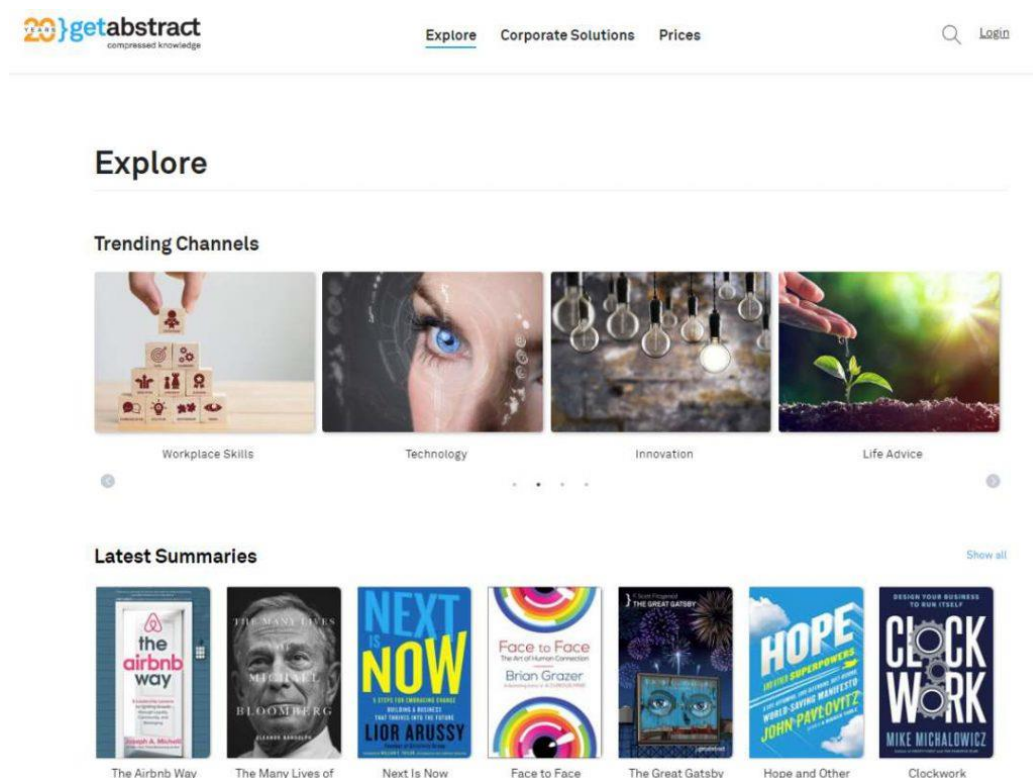


Рис.1. Інтерфейс головної сторінки ІС GetAbstract

Додаток орієнтований на корпоративний світ з такими категоріями літературних рекомендацій, як менеджмент, інновації, корпоративні ІТ. GetAbstract стверджує, що обслуговує більше однієї третини організацій зі списку Fortune 100. Огляди книг тут досить високої якості, хоча деякі з них трохи «сухі». Вони доступні сімома мовами, включаючи англійську. Кожен огляд містить рейтинг, винос, популярні цитати та біографію автора. Відмінні функції включають можливість виділяти слова, наприклад, улюблену ідею або кілька речень.

Існує 3-денна безкоштовна пробна версія, і якщо вам це подобається, ви можете підписатися за 99 євро США на рік, щоб отримувати 5000 літературних оглядів. Також існує про-підписка за 299 євро на рік, яка відкриває доступ до 20000 літературних оглядів. Повний набір існуючих підписок зображено на Рис.2.

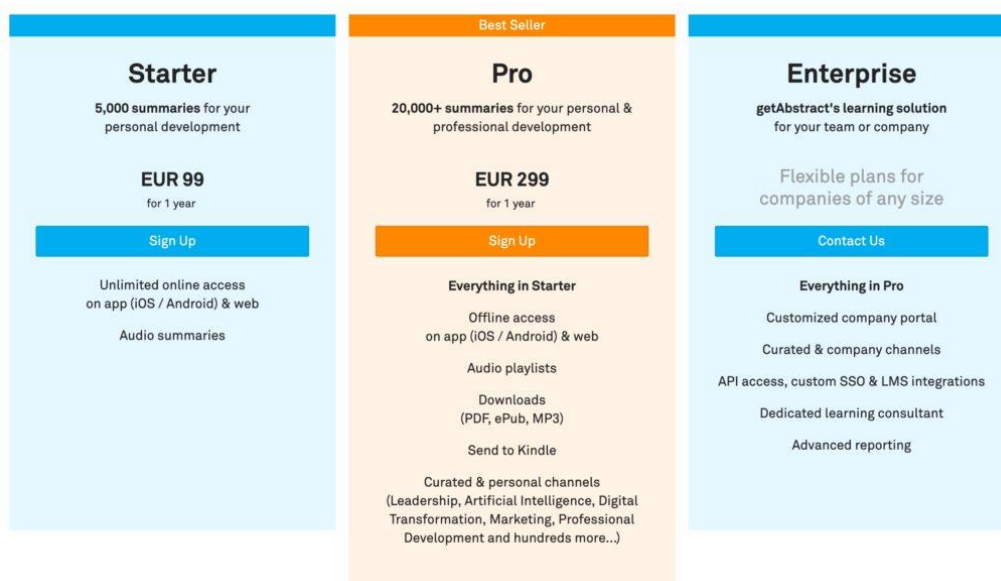


Рис.2. Плани підписок IC GetAbstract

Оновлення надає вам доступ до всіх книг, а також до кількох платформ, включаючи Kindle, а також до програм для Android та iOS.

В цілому, можна підсумувати, що IC призначена в першу чергу для співробітників та керівників, які працюють на великих підприємствах.

Ще одна IC, на яку варто звернути увагу – Summaries [2]. Вона надає стислу інформацію та інструменти, що спеціалізується на бізнес-книгах.

Інтерфейс головної сторінки IC Summaries можна побачити на Рис.3.

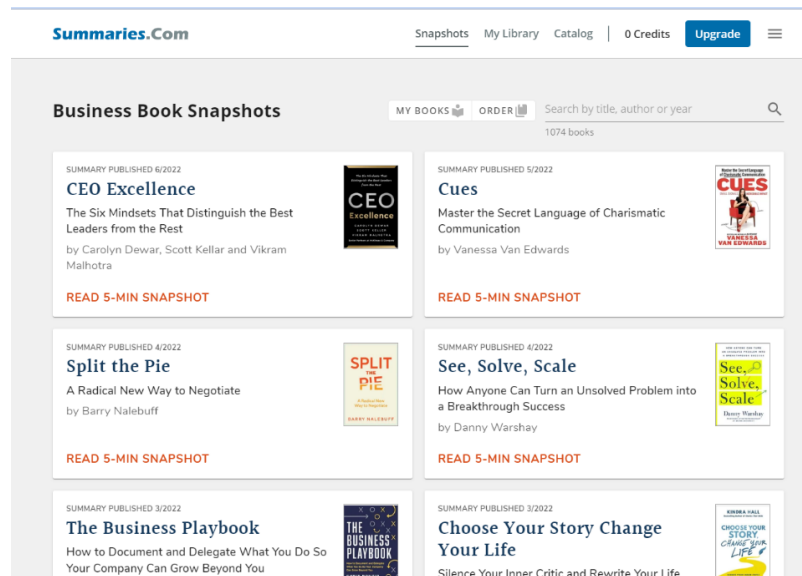


Рис.3. Інтерфейс головної сторінки IC Summaries

Однією з переваг Summaries.com є те, що її редактори можуть вмістити головні ідеї книги на 300 та більше сторінок у 8 сторінок огляду. Компанія каже, що їхній досвід показує, що вісім сторінок є ідеальною довжиною для того, щоб хтось міг зрозуміти основні ідеї.

Підписка від 12 доларів на місяць дає вам новий огляд щотижня. Або купуйте поштучно, ціни від 15 доларів за п'ять оглядів. Веб-сайт також пропонує безкоштовний план із скороченими 5-хвилинними рекомендаціями в публікаціях блогу (їх також можна завантажити у форматі PDF). Набір існуючих підписок на IC Summaries зображено на Рис.4.

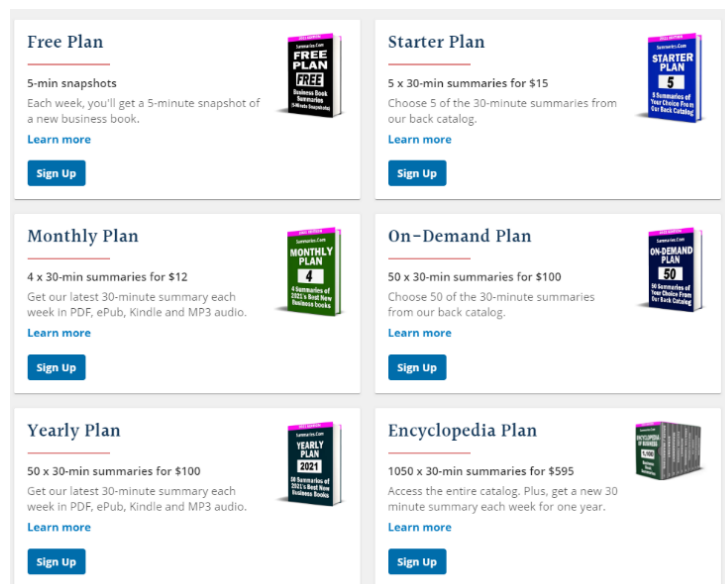


Рис.4. Плани підписок IC Summaries

Проаналізувавши існуючі ІС було прийнято рішення створити нову ІС з надання літературних рекомендацій. Серед основних змін: у новій системі було використано спеціально розроблену дизайн-систему для користувача, програмний код був написаний з використанням актуальних бібліотек та фреймворків; інформація у системі тепер завантажується швидше, а робота запитів на сервер та виконання програмного коду стали більш оптимізованими. А одним з головних нововведень у порівнянні з існуючими ІС є оновлена та актуальна UI/UX-складова. Крім того, користування розробленою системою є безкоштовним.

РОЗДІЛ 2. РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1. Аналіз і специфікація вимог до інформаційної системи

2.1.1. Бізнес-вимоги

Бізнес-вимоги – це високорівневі цілі та потреби організації, які зумовлюють вимоги до системи. Говорячи про бізнес-вимоги до системи варто згадати поняття стейкхолдерів. Стейкхолдери – сторони, які зацікавлені в функціонуванні ІС. Розглянемо стейкхолдерів ІС «BookOpinion»:

- Власник ІС;
- Інтернет-магазин книг;
- Потенційний користувач.

Кожен стейкхолдер має власні бізнес-цілі, розглянемо їх:

- Власник ІС - взаємодія з інтернет-магазинами книг задля підвищення обсягів їхніх продажів. Оскільки було вирішено зробити користування ІС безкоштовним, власник ІС має знайти інший спосіб отримувати прибуток від роботи системи. Було прийнято рішення співпрацювати з інтернет-магазинами книг шляхом розміщення посилання на сторінку їх магазину, де можна придбати книгу з рекомендації під оглядом на книгу. Таким чином від інтернет-магазину надходить прибуток за послуги реклами.
- Інтернет-магазин книг - підвищення обсягів продаж за рахунок додаткового каналу реалізації товарів.
- Користувач – зручне, швидке та безкоштовне рішення для отримання книжкової рекомендації.

Згідно бізнес-цілей кожного стейкхолдера було складено бізнес-вимоги до ІС. ІС з надання книжкових рекомендацій має надавати можливість:

- Власнику ІС
 - пропонувати інтернет-магазинам книг додатковий майданчик для їх продажів;

- Інтернет-магазину книг
 - використати додатковий канал продажів книг;
- Користувачу
 - авторизуватись в системі
 - переглянути список рекомендацій
 - згенерувати випадкову рекомендацію
 - переглянути деталі рекомендації
 - додати рекомендацію в обране
 - видалити рекомендацію з обраного
 - перейти до інтернет-магазину, який продає книгу, для якої написана рекомендації
 - залишити коментар під рекомендацією
 - оновити свій коментар під рекомендацією
 - видалити свій коментар під рекомендацією
 - оновити персональні дані в особистому кабінеті

Проаналізувавши бізнес-вимоги, було побудовано діаграму бізнес-вимог (Рис.1).

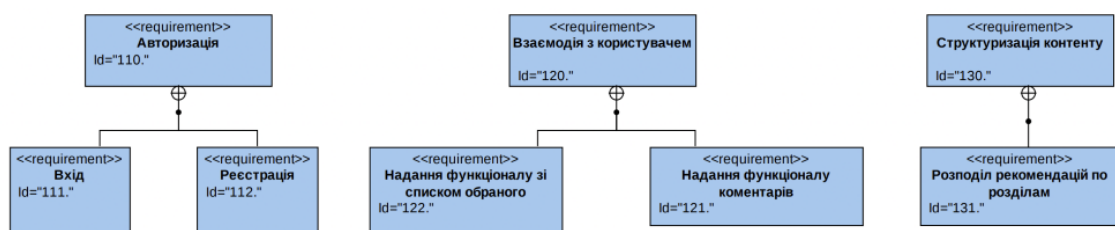


Рис.1. Діаграма бізнес-вимог

2.1.2. Нефункціональні вимоги

Нефункціональні вимоги — це вимоги до програмного забезпечення, які задають критерії для оцінки якості його роботи. Нефункціональні вимоги відповідають на питання якою система повинна бути.

На етапі аналізу вимог було виділено такі нефункціональні вимоги:

- Адаптивність – система має бути адаптованою під різні пристрої та середовища.
 - Браузерна адаптивність – система має коректно працювати в усіх основних браузерах;
 - Адаптивність відносно пристроїв – користувацький інтерфейс системи має коректно відображатись на всіх можливих розширеннях екранів.
- Швидкодія – система має швидко реагувати на дії користувача.
 - Швидкодія спілкування з БД – час реакції системи на запити користувача не повинен перевищувати 3 с.
 - Швидкодія роутінгу – перехід між екранами користувацького інтерфейсу системи має бути миттєвим.

Діаграму нефункціональних вимог зображено на Рис. 2.



Рис.2. Діаграма нефункціональних вимог

2.1.3. Функціональні вимоги

Функціональні вимоги — це вимоги до програмного забезпечення, які описують внутрішню роботу системи, її поведінку: калькулювання даних, маніпулювання даними, обробку даних та інші специфічні функції, які має виконувати система. На відміну від нефункціональних вимог, які визначають якою система повинна бути, функціональні вимоги визначають, що система повинна робити.

Для ІС «BookOpinion» було розроблено ряд функціональних вимог:

- Якщо у користувача немає акаунту, він має можливість зареєструватись
- Якщо у користувача є акаунт, він має можливість авторизуватись
- Система забезпечує автентифікацію користувача, поки він самотійно не вийде з акаунту
- У неавторизованого користувача має бути можливість переглянути список літературних розділів, за якими у системі наявні рекомендації
- У неавторизованого користувача має бути можливість згенерувати випадкову літературну рекомендацію

- У неавторизованого користувача має бути можливість переглянути список літературних рекомендацій по розділам
- У неавторизованого користувача має бути можливість переглянути деталі літературної рекомендації
- У неавторизованого користувача має бути можливість перейти до сторінки інтернет-магазину, де продається книга, про яку була написана рекомендація
- У неавторизованого користувача має бути можливість переглянути список коментарів під літературною рекомендацією
- У авторизованого користувача має бути можливість додати літературну рекомендацію в список "Обране"
- У авторизованого користувача має бути можливість створювати під літературною рекомендацією довільну кількість коментарів
- У авторизованого користувача має бути можливість оновлювати свої коментарі під літературною рекомендацією
- У авторизованого користувача має бути можливість видаляти свої коментарі під літературною рекомендацією
- У авторизованого користувача має бути можливість оновити особисту інформацію у профілі
- У авторизованого користувача має бути можливість побачити свій список "Обране" у профілі

Діаграму функціональних вимог зображено на Рис. 3.

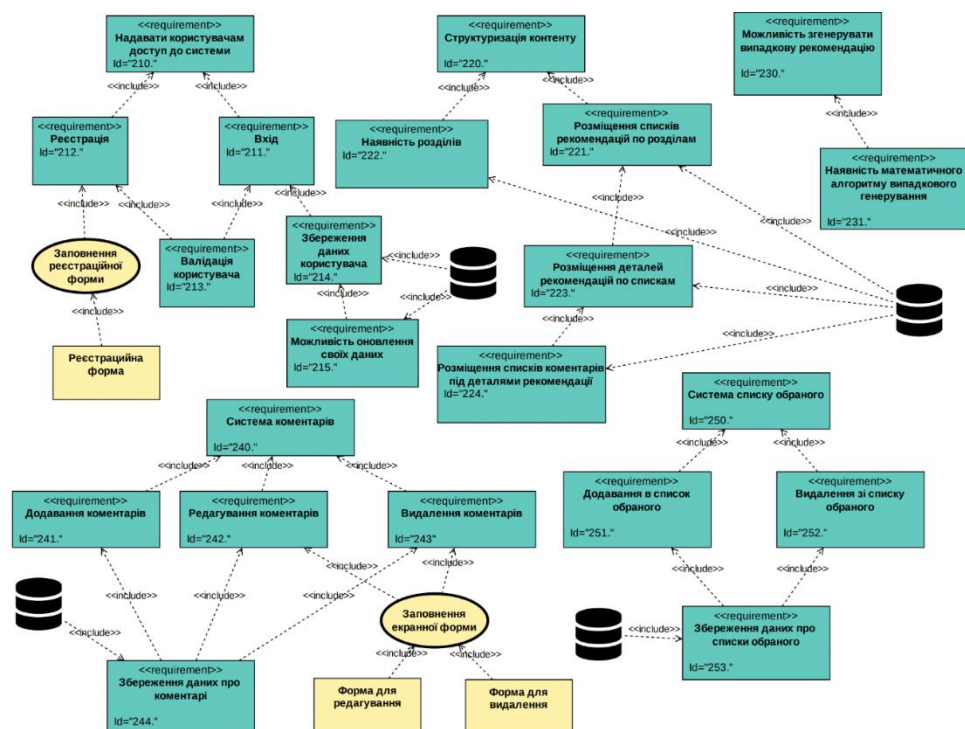


Рис.3. Діаграма функціональних вимог

Деталізовані специфікації вимог наведено у додатку 1.

2.2. Постановка та алгоритм розв'язання задачі

2.2.1. Характеристика задачі

Призначення ІС «BookOpinion» полягає у наданні літературних рекомендацій в рамках веб-браузеру.

Потреба у автоматизованому рішенні для реалізації даної ІС обгрунтована зручністю та швидкістю користування веб-браузером як платформою для роботи програмного продукту, а також актуальністю використання засобів веб-програмування. Рішенням задачі з надання літературних рекомендацій є отримання користувачем рекомендації (огляду) на книгу.

Задача розв'язується за наявності у користувача пристрою, що має доступ до мережі Інтернет. Задача вирішується щоразу, як відправляється відповідний запит на сервер й вихідна інформація може бути отримана одразу після успішного виконання сервером запиту та повернення відповіді. Автоматизоване розв'язання задачі припиняється за умови закриття відповідного вікна з користувацьким інтерфейсом ІС у браузері.

Інформаційна модель задачі зображена на Рис.4.



Рис.4. Інформаційна модель задачі

Як бачимо з інформаційної моделі вище – на вхід до ІС надходить код рекомендації, яку треба згенерувати, а на вихід – електронна форма самої рекомендації.

2.2.2. Вихідна інформація

Вихідна інформація ІС – це інформація, яка видається на об'єкт управління, персоналу чи в інші системи управління, у вигляді документів, відображень, даних і сигналів і отримується в результаті виконання функцій інформаційної системи.

Форми носіїв вихідної інформації, оброблюваної за допомогою обчислювальної техніки, залежать від експлуатаційних можливостей технічних засобів, які застосовуються, варіанта і повноти обробки цієї інформації, її призначення і методів використання.

Основним вихідним повідомленням буде літературна рекомендація. У Таблиці 1 наведено вихідну інформацію у системі.

Таблиця 1. Вихідні повідомлення.

№ з/п	Назва вихідного повідомлення	Ідентифікатор	Форма подання	Термін частота надходження	Джерело
1	Рекомендація	recommendation	Екранна форма	Протягом дня	ІС

2.2.2. Вхідна інформація

До вхідних повідомлень можна віднести дані, які використовуються для формування оперативних файлів бази даних; файли, що потрапляють на вхід задачі чи комплексу з інших задач та їх комплексів; документи та файли, які носять довідковий характер та можуть бути віднесені до умовно-постійної інформації. Формою подання як для вхідних, так і для вихідних повідомлень може бути або файл, або екранна форма, або паперовий документ.

Вхідними даними для функціонування бази даних ІС з надання літературних рекомендацій можна вважати наступну поточну інформацію: від користувача надходить запит з кодом рекомендації, яку треба відобразити в екранній формі. У Таблиці 2 наведено вхідну інформацію у системі.

Таблиця 2. Вхідні повідомлення.

№ з/п	Назва вхідного повідомлення	Ідентифікатор	Форма подання	Термін частота надходження	Джерело
1	Код рекомендації	Recommendation_id	Файл БД	Протягом дня	Користувач
2	Інформація про рекомендацію	Recommendation	Файл БД	Протягом дня	ІС
3	Інформація про користувача	User	Файл БД	Протягом дня	ІС
4	Інформація про коментарі	Comment	Файл БД	Протягом дня	ІС
5	Інформація про обране	Favorite	Файл БД	Протягом дня	ІС
6	Інформація про розділ	Section	Файл БД	Протягом дня	ІС

2.2.4. Алгоритм розв'язання задачі

Математичний опис. У Таблиці 3 наведено список атрибутів предметної області.

Таблиця 3. Список атрибутів предметної області

Назва атрибута	Ідентифікатор	Умовне позначення	Характеристика атрибута
Код рекомендації	<i>Kod_Rek</i>	k	якісний , груповий
Код розділу	<i>Kod_Roz</i>	r	якісний , груповий
Код першої рекомендації в таблиці БД	<i>Kod_Per</i>	P	якісний , груповий
Код останньої рекомендації в таблиці БД	<i>Kod_Ost</i>	O	якісний , груповий
Випадкове число з діапазону від 0 до 1	<i>RNDM</i>	V	Кількісний, розрахунковий
Код згенерованої рекомендації	<i>KZR</i>	KR	Кількісний, розрахунковий

Основні математичні розрахунки в системі передбачають калькуляцію коду випадкової рекомендації відповідно до вказаних параметрів та генерування коду рекомендації після вибору розділу та рекомендації вручну.

Нижче приведено формули, які було використано в дипломному проєкті:

- KR_r – код k-ої рекомендації з r-ого розділу
- $KR_r = V * (O_r - P_r) + P_r$ - код випадкової k-ої рекомендації з r-розділу.

У програмній реалізації атрибут V обчислюється за допомогою javascript-методу $Math.random()$, який імпортується з бібліотеки для математичних обчислень $math$.

Алгоритм розв’язання задачі. Для того, щоб зрозуміти як і коли будуть відбуватися математичні обчислення та генеруватиметься код рекомендації, було створено алгоритм розв’язання задачі (Рис.5).

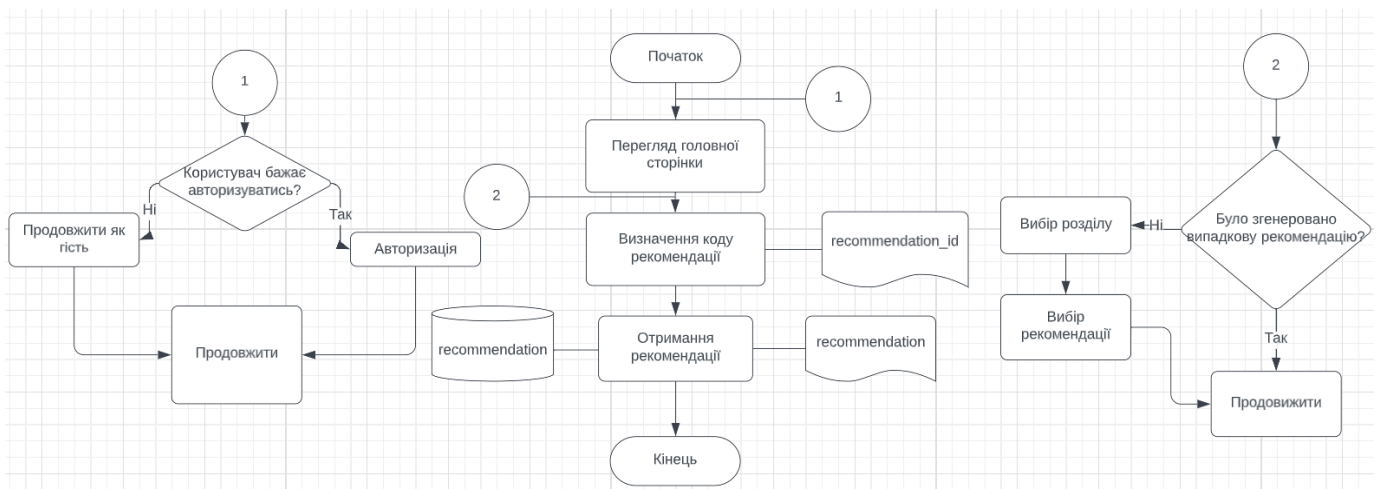


Рис.5. Алгоритм розв’язання задачі.

2.3. Моделювання інформаційної підсистеми

2.3.1. Моделювання поведінки системи

У ході моделювання поведінки системи було розроблено діаграму прецедентів (Рис.6)

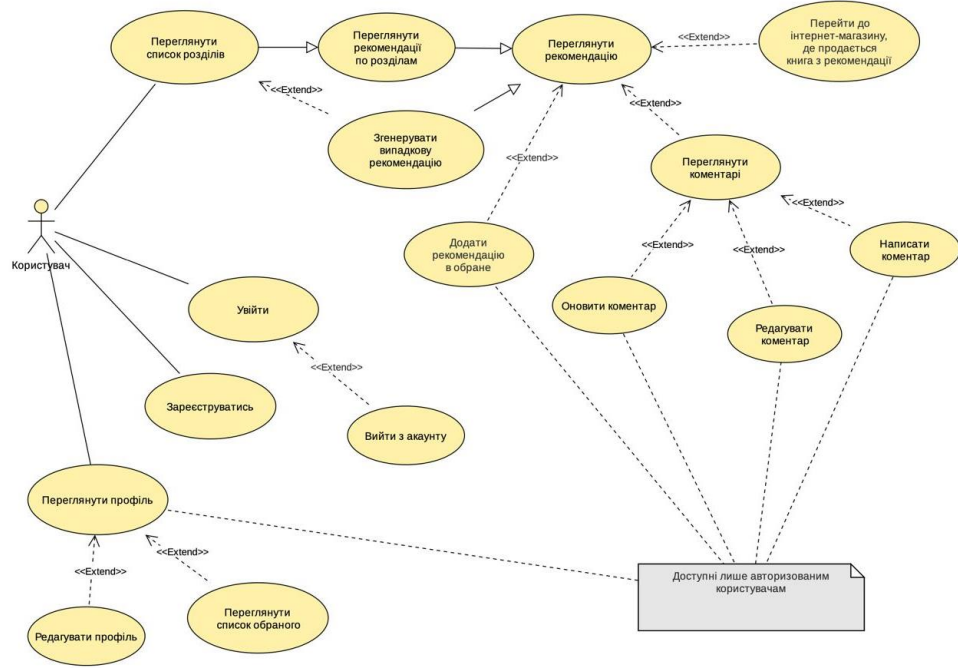


Рис.6. Діаграма прецедентів

Крім того, було розроблено діаграми послідовності для кожного прецеденту у відповідності до сценаріїв варіантів використання в обсязі, достатньому для виокремлення всіх об'єктів (компонентів) проектованої системи.

Було розроблено такі діаграми послідовності:

1. Послідовність викликів класів неавторизованим користувачем (Рис.7).

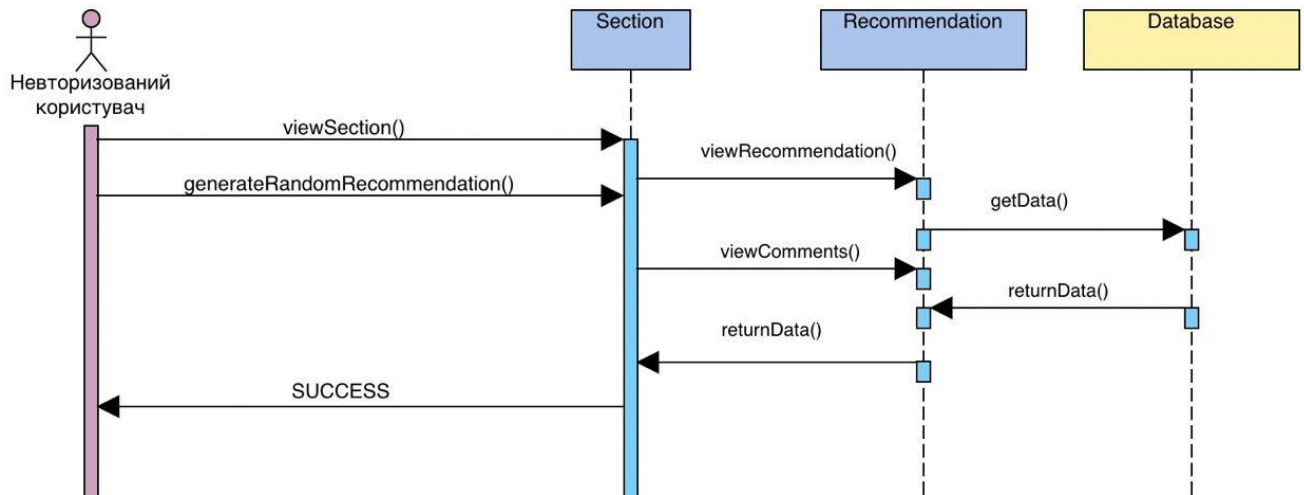


Рис.7. Діаграма послідовності для процесу авторизації

2. Послідовність виклику класів авторизованим користувачем (Рис.8).

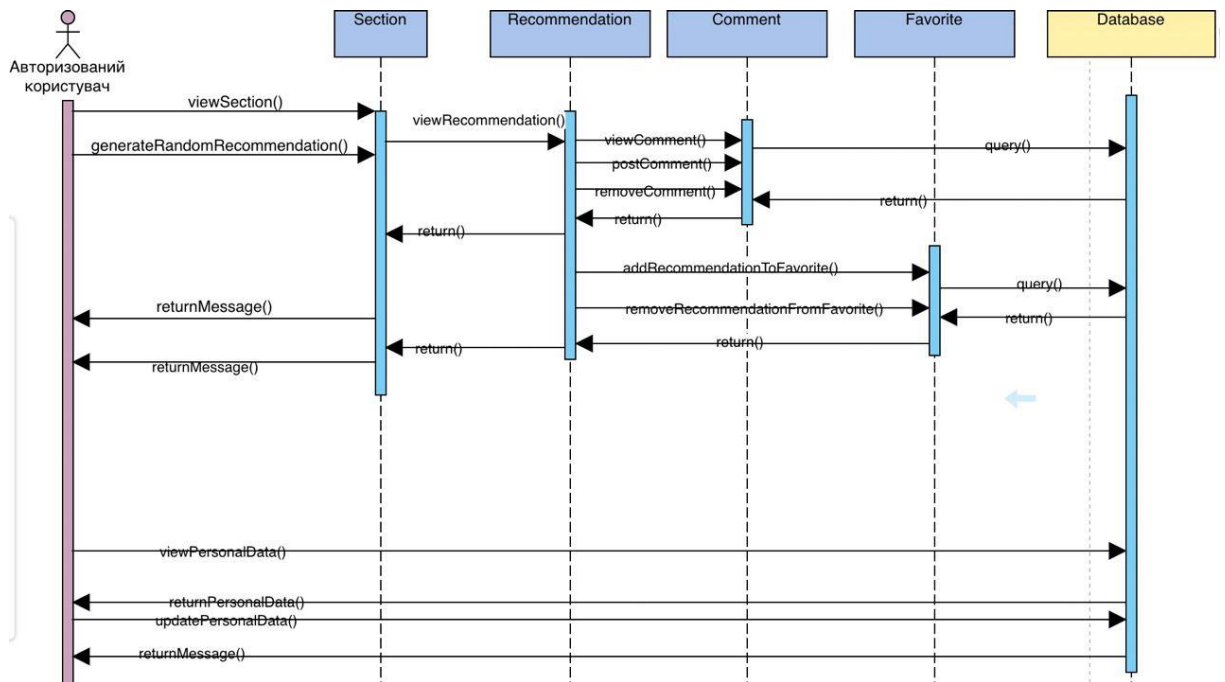


Рис.8. Діаграма послідовності для процесу перегляду розділів та рекомендацій

2.3.2. Модель структури системи

З метою відображення статичної структури класів системи і зв'язків між ними, було розроблено діаграму класів (Рис.9). Неавторизований клас наслідує авторизований клас і деякі його методи й може лише переглядати вміст інших класів. Авторизований користувач може переглядати вміст інших класів та вносити до них зміни.

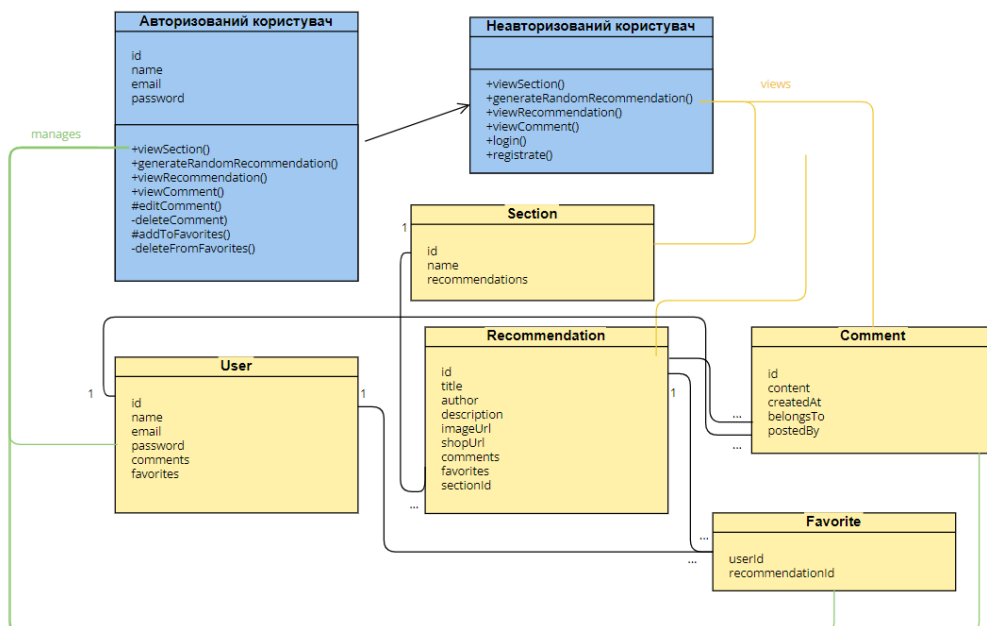


Рис.9. Діаграма класів

2.3.3. Прототипування користувацького інтерфейсу системи

Прототип системи було реалізовано за допомогою редактора [Figma](#). Figma — це хмарний векторний графічний редактор і інструмент для створення прототипів, який переважно працює в Інтернеті з додатковими автономними функціями. Являє собою пропрієтарну систему.

Figma працює на будь-якій операційній системі, яка запускає веб-браузер. З Figma можна використовувати комп'ютери Mac, комп'ютери з Windows, комп'ютери з Linux і навіть Chromebook. Користувачі різних операційних систем, браузерів та пристроїв можуть ділитися, відкривати та редагувати файли Figma. У багатьох організаціях дизайнери використовують комп'ютери Mac, а розробники комп'ютери з ОС Linux. Figma допомагає об'єднати ці групи.

Спільна робота з файлами в режимі реального часу допомагає пом'якшити «зміщення дизайну», яке визначається як неправильне тлумачення або відхилення від узгодженого дизайну. Зміна дизайну зазвичай відбувається, коли ідея задумана та швидко реалізована під час виконання проекту. На жаль, це часто призводить до відхилення від встановленої конструкції, що викликає тертя і повторну роботу.

На базі Figma наявна надзвичайно велика кількість плагінів та бібліотек, що містять найрізноманітніші елементи - від іконок до ілюстрацій. Слід зазначити, що серед них наявні як безкоштовні, так і платні пакети. Важливим є і те, що у Figma можна обрати екран будь-якого пристрою для створення дизайну під конкретне розширення.

На Рис.10 зображено логотип розробленої ІС.



Рис.10. Логотип ІС «BookOpinion»

На Рис.11 можна побачити всі компоненти та сторінки, розроблені для ІС «BookOpinion» у середовищі Figma. Дизайн-систему можна переглянути за [посиланням](#).

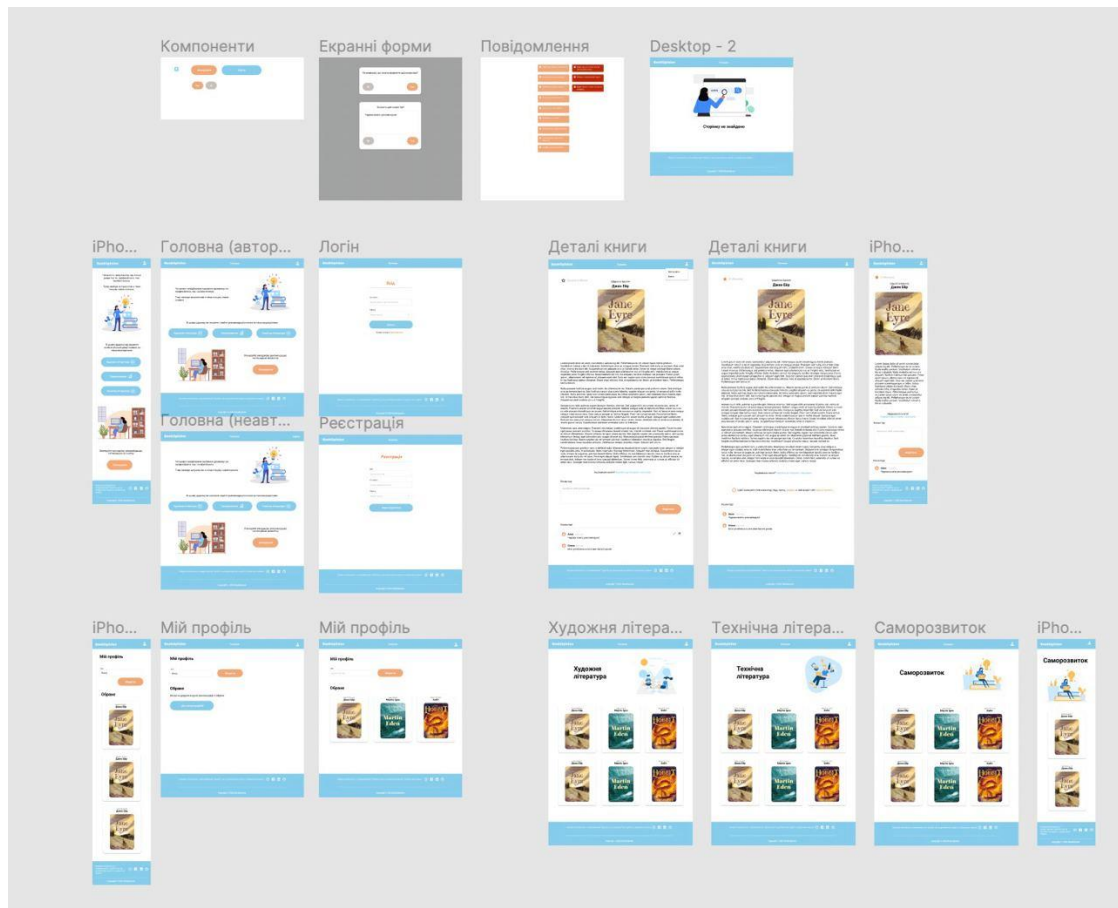


Рис.11. Розроблена дизайн-система для ІС «BookOpinion»

Рисунки прототипів всіх можливих екранів та компонентів системи наведено у додатку 2.

РОЗДІЛ 3. ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

3.1. Інформаційне забезпечення

Загальна характеристика інформаційного забезпечення. Одними з ключових пунктів у питанні функціонування будь-якої інформаційної системи є організація та проектування її інформаційного забезпечення (ІЗ). Інформаційне забезпечення являє собою єдину систему класифікації та кодування інформації, уніфіковану систему документації, схеми інформаційних потоків, що циркулюють в інформаційній системі, а також методологію побудови, склад та зміст баз даних.

По структурі інформаційне забезпечення ІС складається з двох частин:

- Позамашинне ІЗ, яке призначене для обслуговування системи управління об'єктом у вигляді, що сприймається людиною без використання технічних засобів.
- Внутрішньомашинне ІЗ, що включає всі види спеціально організованої інформації, представленої у вигляді, зручному для сприйняття технічними засобами: файли (масиви), бази даних, бази знань, вітрини даних та інформаційні сховища.

На Рис.1 представлена загальна схема ІЗ, задіяна для функціонування ІС “BookOpinion”.

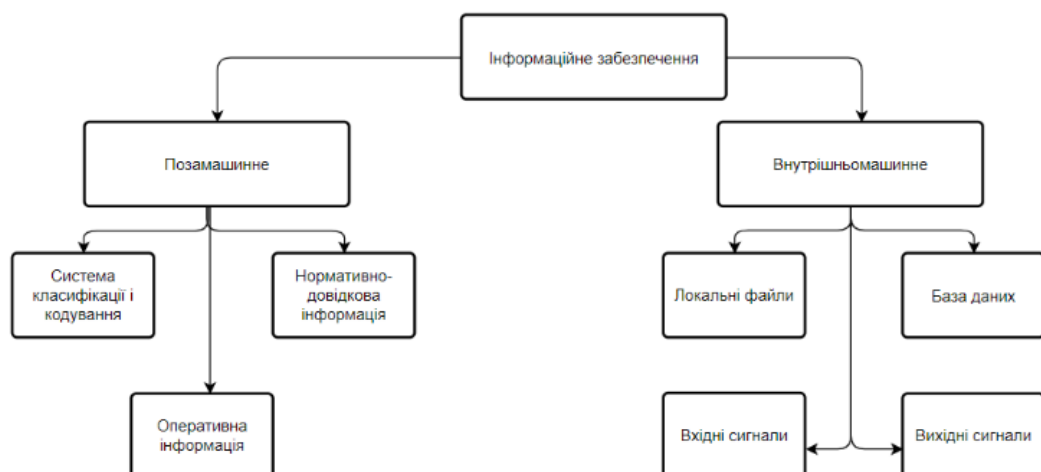


Рис.1. Загальна схема ІЗ, задіяна для функціонування ІС “BookOpinion”

Вибір СКБД. Під час вибору бази даних для реалізації певного проекту дуже важливо вибрати таку БД, яка найбільше відповідає вимогам інформаційної системи. Крім того, необхідно визначитися яка модель автоматизації реалізується. При виборі СКБД необхідно взяти до уваги наступні фактори:

1. максимальна кількість користувачів, які одночасно звертаються до бази даних;
2. характеристики клієнтської предметної області;
3. апаратні компоненти сервера;
4. серверна операційна система;
5. рівень знань обслуговуючого персоналу.

У якості СКБД для ІС по темі бакалаврського проекту було обрано технології SQLite [3] та Prisma [4]. SQLite — це бібліотека на мові С, яка реалізує невеликий, швидкий, автономний, високонадійний, повнофункціональний механізм баз даних SQL. SQLite є найбільш використовуваним механізмом баз даних у світі. Формат файлу SQLite є стабільним, міжплатформним і зворотно сумісним, і розробники обіцяють зберегти його таким до 2050 року. Файли бази даних SQLite зазвичай використовуються як контейнери для передачі багатого вмісту між системами і як довгостроковий архівний формат для даних. Наразі у світі активно використовується понад 1 трильйон баз даних SQLite.

Серед переваг баз даних SQLite можна виділити:

- Транзакції є атомарними, послідовними, ізольованими та тривалими (ACID) навіть після системних збоїв і збоїв живлення.
- Нульова конфігурація - не потрібно налаштування або адміністрування.
- Повна база даних зберігається в одному кросплатформному дисковому файлі. Чудово підходить для використання як формат файлу програми.
- Підтримує терабайтні бази даних і гігабайтні рядки та великі об'єкти.
- Простий, легкий у використанні API.

- Швидко: у деяких випадках SQLite швидше, ніж прямий ввід-вивід файлової системи
- Написано в ANSI-C. Прив'язки TCL включені. Прив'язки для десятків інших мов доступні окремо.
- Добре прокоментований вихідний код із 100% покриттям тестуванням гілок.
- Доступний у вигляді одного файлу вихідного коду ANSI-C, який легко компілювати і, отже, легко додати до більшого проекту.
- Автономність: немає зовнішніх залежностей.
- Кросплатформність: Android, *BSD, iOS, Linux, Mac, Solaris, VxWorks і Windows (Win32, WinCE, WinRT). Легко переносити на інші системи.
- Постачається з автономним клієнтом інтерфейсу командного рядка (CLI), який можна використовувати для адміністрування баз даних SQLite.

Для реалізації IC по темі дипломного проекту було вирішено обрати саме базу даних SQLite оскільки вона не вимагає конфігурації та зберігає інформацію у звичайних дискових файлах, а отже є популярним вибором як база даних для підтримки малих і середніх веб-додатків.

Оскільки IC було реалізовано з використанням новітньої технології GraphQL [5], було прийнято рішення використати технологію Prisma в якості набору інструментів для спілкування з базою даних.

Prisma — це набір інструментів баз даних з відкритим вихідним кодом, який дозволяє розробникам легко отримати доступ до баз даних, надаючи чистий і безпечний API для надсилання запитів бази даних.

Prisma складається з трьох інструментів:

- Клієнт Prisma: автоматично згенерований і безпечний для типів конструктор запитів для Node.js і TypeScript.
- Prisma Migrate: декларативна система моделювання та міграції даних.

- Prisma Studio: графічний інтерфейс для перегляду та редагування даних у базі даних.

Налаштування бази даних відбулося наступним чином. Було створено каталог `server`, який містить каталог `prisma`, де знаходиться головний файл бази даних - `schema.prisma`. Крім того, тут знаходиться каталог `migrations`, де зберігаються SQL-запити до бази даних для оновлення моделей (таблиць) бази даних. На Рис. 2 наведено структуру каталогу, який містить серверний код.

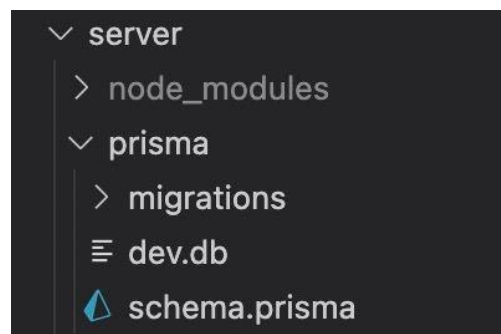


Рис.2. Структура каталогу `server`

У даному файлі вказано, що джерелом бази даних буде SQLite:

```

datasource db {
  provider = "sqlite"
  url      = "file:./dev.db"
}

```

Рис. 15. Фрагмент коду для встановлення зв'язку з базою даних

Поля, що передаються в блок джерела даних:

- Provider (постачальник): визначає конектор джерела даних `sqlite`.
- url: вказує URL-адресу підключення до бази даних SQLite. URL-адреса підключення завжди починається з файлу префікса і містить шлях до файлу, що вказує на файл бази даних SQLite. У цьому випадку файл знаходиться в тому ж каталозі і називається `dev.db`.

Інфологічна модель бази даних. Метою інфологічного проектування є створення структурованої інформаційної моделі предметної області, для якої буде розроблятися база даних.

Сутність інфологічного проектування полягає у визначенні інформаційних об'єктів (сутностей) предметної області, які підлягають зберігання в БД, визначенню характеристик об'єктів та структурних зв'язків між ними. Результатом інфологічного моделювання є інформаційно-логічна модель предметної області та опис її складових елементів. Інфологічна модель має задовольняти всі вимоги користувачів та прикладних програм.

Основними складовими елементами інфологічної моделі є:

- інформаційний об'єкт;
- атрибут;
- інформаційний запит;
- запитувальний зв'язок;
- структурний зв'язок.

Інфологічне проектування здійснюється за наступними етапами:

- вилучення омонімії та синонімії;
 - кожен атрибут повинен мати унікальне ім'я та унікальну семантику для однозначної його ідентифікації.
 - ні синонімії, ні омонімії в даному разі немає.
- агрегація атрибутів в інформаційні об'єкти. Тобто усі атрибути агрегуються в об'єкти, яким присвоюються унікальні імена;
- перевірка на відповідність умовам нормалізації; приведення до 3 чи 4 нормальної форми.
- виділення інформаційних запитів та їх опис;
- представлення інформаційних запитів у структурованому вигляді.
- перевірка запитальних зв'язків на відповідність умові канонічності;
- побудова структурних зв'язків між об'єктами.

На Рис. 3 наведено інфологічну модель бази даних.

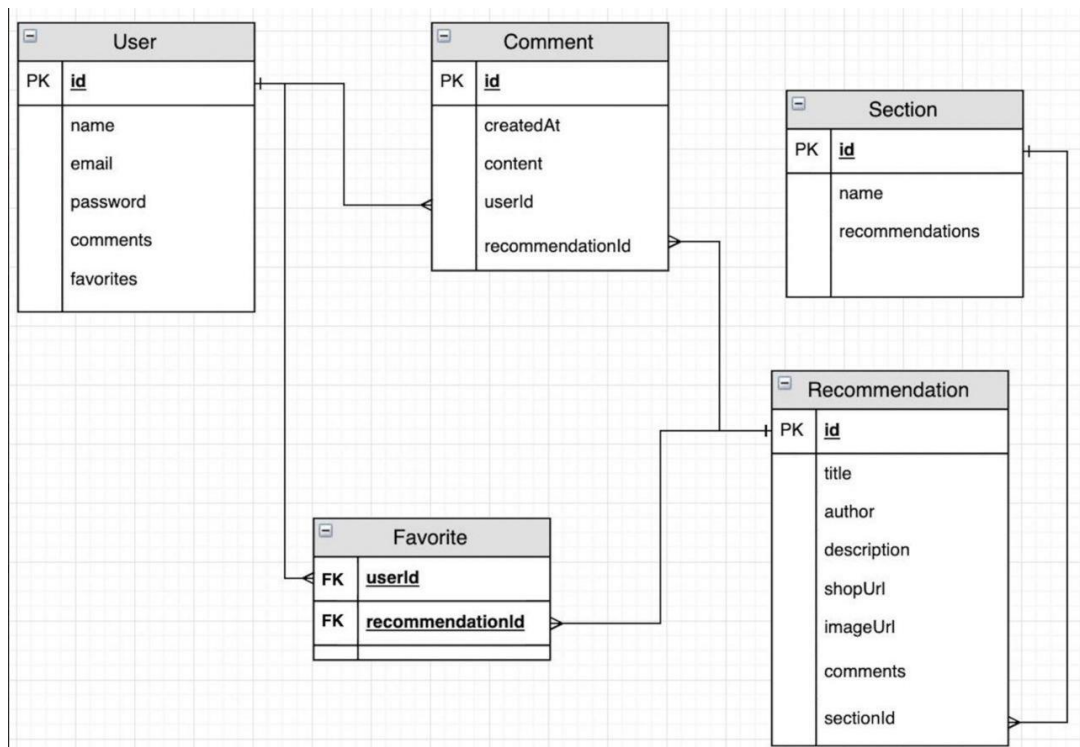


Рис.3. Інфологічна модель бази даних

Даталогічна модель бази даних. Важливий крок проектування системи бази даних - відображення особливостей інфологічної моделі предметної області. Цю стадію називають логічним (або даталогічним) проектуванням бази даних, а її результатом є схема бази даних, що включає визначення всіх інформаційних елементів і зв'язків, у тому числі завдання типів, характеристик та імен.

Дана база даних є реляційною. У ній об'єкти й зв'язки між ними представляються у вигляді таблиць (відносин), що складаються з рядків і стовпців. Стовпець – це поле, рядок – це запис. Кожне поле має ім'я й тип. Імена полів – це атрибути (вони визначаються властивостями об'єкта). Тип задає спосіб подання атрибута.

Головна властивість таблиці в реляційною базі даних - відсутність у ній однакових записів. В таблиці повинні бути один або кілька атрибутів, які забезпечують унікальність кожного рядка. Такі атрибути називаються ключем. Ключів у таблиці може бути кілька. З них вибирається один, котрий буде надалі представляти (заміняти) кожен запис таблиці. Такий ключ називається первинним.

У реляційній БД використовується кілька таблиць, між якими встановлюються зв'язки. Таким чином, інформація, уведена в одну таблицю, може бути пов'язана з однією або декількома записами з іншої таблиці.

Технологія Prisma пропонує поняття «моделі» для таблиць реляційної бази даних. Моделі прописуються розробником вручну у файлі `schema.prisma`, а потім за допомогою командного рядку модель мігрує у базу даних та на її основі створюється нова таблиця.

Нижче приведено фрагмент коду (Рис.4), на якому представлено 5 моделей (таблиць) бази даних для ІС “BookOpinion” та типи їх полів.

```
model Comment {
  id          Int          @id @default(autoincrement())
  createdAt   DateTime @default(now())
  content     String
  postedBy    User?        @relation(fields: [postedById], references: [id])
  postedById  Int?
  belongsTo   Recommendation? @relation(fields: [belongsToId], references: [id])
  belongsToId Int?
}

model User {
  id          Int          @id @default(autoincrement())
  name        String
  email       String       @unique
  password    String
  comments    Comment[]
  favorites   Favorite[]
}

model Recommendation {
  id          Int          @id @default(autoincrement())
  title       String
  author      String
  description  String
  shopUrl     String
  imageUrl    String
  comments    Comment[]
  section     Section?     @relation(fields: [sectionId], references: [id])
  sectionId   Int?
  favorites   Favorite[]
}

model Section {
  id          Int          @id @default(autoincrement())
  name        String
  recommendations Recommendation[]
}

model Favorite {
  id          Int          @id @default(autoincrement())
  addedBy     User?        @relation(fields: [addedById], references: [id])
  addedById   Int?
  addedRecommendation Recommendation? @relation(fields: [addedRecommendationId], references: [id])
  addedRecommendationId Int?
}
```

Рис.4. Даталогічна модель бази даних

Дамо коротку характеристику кожному із зв'язків між таблицями у даталогічній моделі даних.

- Розділ: Рекомендація (1: ∞) – у одному розділі може бути багато рекомендацій.
- Рекомендація: Коментар (1: ∞) – у одній рекомендації може бути багато коментарів.
- Рекомендація: Обране (1: ∞) – одна рекомендація може бути у багатьох в обраному.
- Користувач: Коментар (1: ∞) – у одного користувача може бути багато коментарів.
- Користувач: Обране (1: ∞) – у одного користувача може бути багато рекомендацій в обраному.

Зупинимося детальніше на структурі кожної таблиці. Структура бази для ІС з надання літературних рекомендацій подана в Таблиці 3.

Таблиця 3. Структура бази даних

Назва поля	Назва у БД	Тип поля	Допустимі значення
Користувач (User)			
*Код користувача	id	Integer	Генерується автоматично
Ім'я	name	VarChar	Текст від 3 до 50 символів
Пошта	email	VarChar	Текст до 50 символів
Пароль	password	VarChar	Текст до від 8 до 90 символів
Коментарі	comments	Comment[]	Необмежений масив значень
Обране	favorites	Favorite[]	Необмежений масив значень
Коментар (Comment)			

*Код коментаря	id	Integer	Генерується автоматично
Дата створення	createdAt	VarChar	Генерується автоматично
Коментар	content	VarChar	Текст від 3 до 200 символів
Код користувача	userId	Integer	Підставляється автоматично
Код рекомендації	recommendationId	Integer	Підставляється автоматично
Розділ (Section)			
*Код розділу	id	Integer	Генерується автоматично
Назва розділу	name	VarChar	Текст від 3 до 50 символів
Рекомендації	recommendations	Recommendation[]	Необмежений масив значень
Рекомендація (Recommendation)			
*Код рекомендації	id	Integer	Генерується автоматично
Назва рекомендації	title	VarChar	Текст до 50 символів
Автор	author	VarChar	Текст до 50 символів
Опис	description	VarChar	Текст до 3000 символів
Посилання на інтернет-магазин	shopUrl	VarChar	Текст до 255 символів
Посилання на зображення	imageUrl	VarChar	Текст до 255 символів

Коментарі	comments	Comment[]	Необмежений масив значень
Код розділу	sectionId	Integer	Підставляється автоматично
Обране (Favorite)			
*Код користувача	userId	Integer	Підставляється автоматично
*Код рекомендації	recommendationId	Integer	Підставляється автоматично

Як видно з таблиці, більшість полів БД мають тип даних Integer, діапазон значень якого від -2147483648 до 2147483647.

Ключові поля таблиць генеруються автоматично, а ідентифікатори зв'язкових відношень - проставляються автоматично після вибору користувачем їх значень.

Задля контролю введених користувачем значень було використано регулярні вирази – спеціальні синтаксичні правила. На Рис.5 наведено фрагмент коду з регулярними виразами для перевірки введених значень паролю, пошти та імені користувача.

```
const passwordRegex = /^(?=.*[A-Z])(?=.*[a-z])(?=.*[0-9]).{8,90}$/

const emailRegex = /\S+@\S+\.\S+\/

const nameRegex = /^[A-Za-z-!@?#"'%&;() *,;\-=[\]\_{}<>\u0400-\u04FF]*$/g
```

Рис.5. Регулярні вирази

Якщо запустити Prisma Studio - графічний інтерфейс для перегляду та редагування даних у базі даних – то можна побачити створені моделі (таблиці) та кількість записів у них (Рис.6).

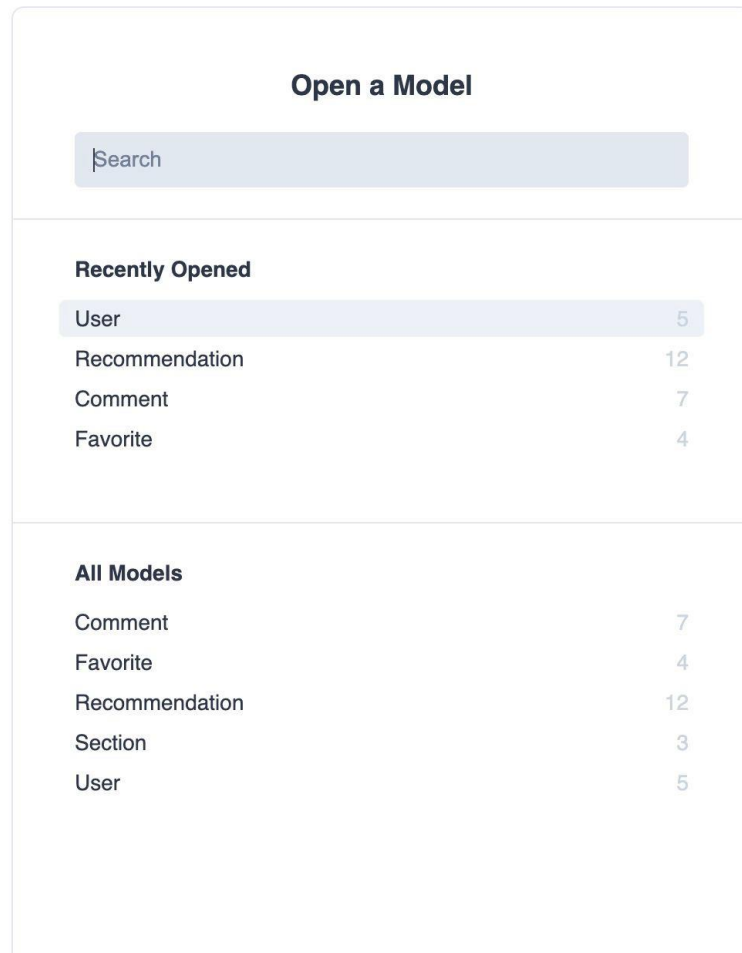


Рис.6. Моделі (таблиці) бази даних

В додатку 3 наведено тестові дані, завантажені в базу даних.

3.2. Технічне забезпечення

Загальні положення та схема автоматизації. Інформаційна система для з надання літературних рекомендацій передбачає наявність віддаленого серверу на якому буде розташовано ядро системи, база даних та необхідні файли. Доступ клієнтів до системи буде здійснюватися через використання веб-додатку у браузері з будь-якого пристрою, що має доступ до мережі Інтернет.

На Рис.7 подано структуру КТЗ у вигляді діаграми розгортання.

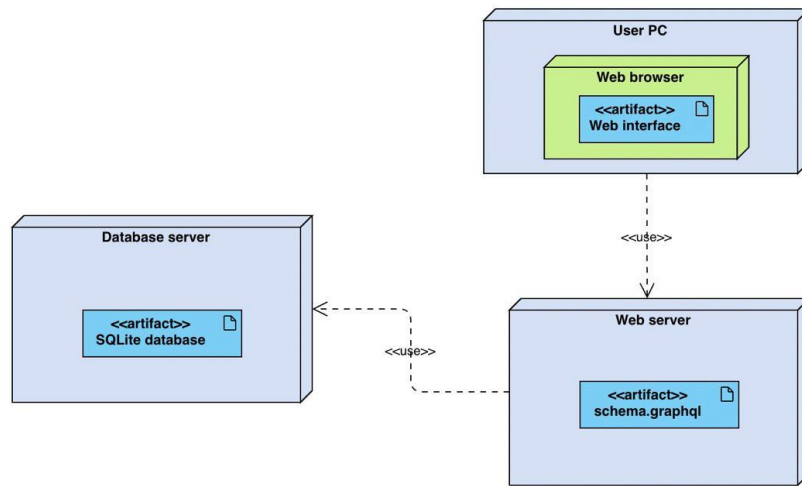


Рис.7. Діаграма розгортання

3.3. Програмне забезпечення

Структура програмного забезпечення. На Рис. 8 Зображено схему укрупненої взаємодії ПЗ

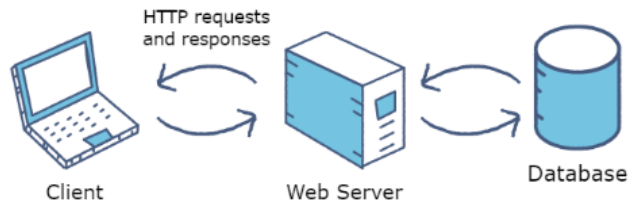


Рис.8. Схема укрупненої взаємодії ПЗ

На Рис.9 Зображено деталізовану схему взаємодії ПЗ з уточненням технологій, що використовувались.

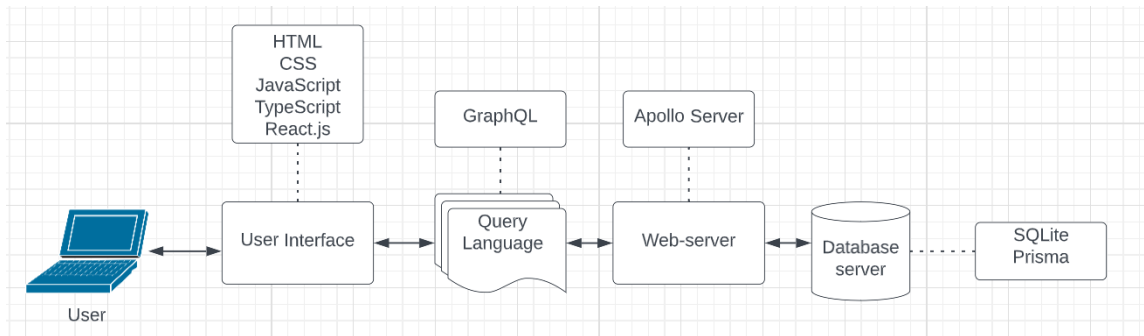


Рис.9. Деталізована схема взаємодії ПЗ

Системне програмне забезпечення. ІС була реалізована на базі можливостей ОС Mac OS. В якості інструменту для розробки й оформлення графічних схем було використано веб-інструменти [Lucid](#) та [Visual Paradigm](#). Для інсталяції та конфігурації залежностей проекту використано npm - менеджер пакетів для Node.js. Для контролю над версіями проекту було використано VCS Git та її інтерфейс GitHub. Проект було розроблено в IDE Visual Studio Code.

Розглянемо більш детально технології, що були використані для розробки клієнтської та серверної частини, а також мову запитів до сервера.

У таблиці 4 описано які технології було використано в ході розробки клієнтської частини ІС.

Таблиця 4. Технології клієнтської частини ІС

Назва	Тлумачення
JavaScript	Мова програмування, призначена для використання як на клієнтській, так і на серверній частинах. Ключова технологія Всесвітньої мережі.
TypeScript	Строго типізована мова програмування, яка базується на мові JavaScript
React.js	JavaScript-бібліотека з відкритим вихідним кодом для розробки інтерфейсів користувача.
Apollo Client	GraphQL-клієнт

HTML	Мова розмітки документів для перегляду веб-сторінок у браузері.
CSS	Мова ієрархічних правил (таблиць стилів), що використовується для представлення зовнішнього вигляду документа,
Material-UI	Бібліотека компонентів
react-hook-form	Бібліотека для роботи з формами
react-router-dom	Бібліотека для роботи з роутінгом
notistack	Бібліотека для роботи з повідомленнями
Jwt-decode	Бібліотека для роботи з jwt-токенами

Клієнтську частину проекту було реалізовано за допомогою React.js бібліотеки, що використовується для побудови інтерфейсів користувача [6]. Сьогодні React.js широко застосовується при створенні різноманітних веб-застосунків і залишається лідером серед подібних бібліотек та фреймворків, таких як Angular.js та Vue.js. Для конструювання React компонентів було використано сучасну технологію React Hooks.

Проект було ініціалізовано за допомогою creat-react-app, що створює початковий базис з усіма необхідними залежностями та налаштуваннями.

У таблиці 5 наведено технології було використано в ході розробки серверної частини ІС.

Таблиця 5. Технології серверної частини ІС

Назва	Тлумачення
SQLite	Механізм баз даних SQL
Prisma	Набір інструментів для роботи з базами даних, сучасна альтернатива для ORM
Apollo Server	GraphQL-сервер

В якості мови запитів було використану новітню технологію побудови API – GraphQL.

GraphQL – це мова запитів для API. Це також середовище виконання на сервері для обробки цих запитів, що використовує систему типів, створену для певних даних. GraphQL надає чіткий і зрозумілий опис даних для API. Це новий стандарт, який надає більш ефективну, потужну і гнучку альтернативу архітектурі REST. GraphQL формує додатковий шар між клієнтом та сервером описує структуру даних та способи їх отримання. На Рис. 25 приведено схему взаємодії клієнтської частини ІС з серверною з використанням мови запитів GraphQL.

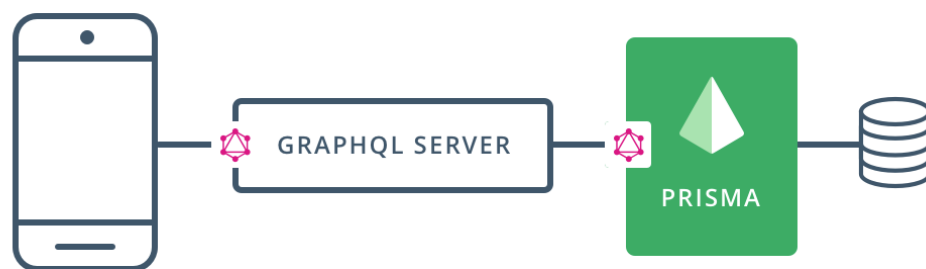


Рис.10. Схема роботи мови запитів GraphQL

GraphQL представляє дані у вигляді графу, де вершини – це об’єкти нашої моделі даних, а ребра – відношення між ними. Запити GraphQL дозволяють діставати довільні піддерева з цього графу.

В загальному GraphQL можна розділити на 3 основних блоки:

- Схема визначень (*definition schema*)
- Запити (*queries*)
- Функції-розпізнавачі (*resolvers*)

В таблиці 6 можна побачити порівняльну характеристику технології GraphQL та традиційного підходу REST.

Таблиця 6. Порівняння GraphQL та REST підходів

	REST	GraphQL
Тип	Архітектурний стиль	Мова запитів
Доступні формати даних	XML, HTML, JSON	JSON
Протокол комунікації	HTTP	HTTP
Типізація даних	Слабка	Сильна
Визначення даних	На сервері	На клієнті
Кінцеві точки	Декілька	Одна
Кешування	Так	Ні

Основні переваги GraphQL:

- Декларативний підхід запитів. Форма запиту відповідає формі результату. Клієнт отримує те, що він визначає у запиті, відповідь не містить не більше, не менше. Це робить результат строго

передбачуваним і дозволяє розробникам сконцентруватися на продукті, а не реалізації. Очевидно, що це вирішує великі проблеми архітектури REST, а саме надлишок та нестача запитуваних даних (*overfetching / underfetching*)

- Деревоподібна структура представлення даних дозволяє отримувати багато ресурсів в одному запиті. Це беззаперечно скорочує кількість запитів до серверу, що значно пришвидшує роботу системи.
- Строго описана система типів спрощує розробку та використання API. Дані описуються у вигляді типів та полів і доступні за допомогою однієї кінцевої точки (*endpoint*). Розробники знають про помилки уже під час компіляції коду (можна навіть налаштувати автозаповнення при написанні запитів).

Обробка запиту GraphQL відбувається послідовно, починаючи з найвищого рівня вкладеності. Кожному полю відповідає функція-розпізнавач, яка здійснює отримання даних з відповідної БД (у проекті це реалізовано за допомогою Prisma). Коли усі дані завантажено, GraphQL збирає результати у формат запиту і повертає клієнту.

Розглянемо приклад функції-розпізнавача типу Mutation, що відповідає за авторизацію. Авторизація користувача реалізована стандартним та поширеним методом у середовищі веб-розробки. Коли користувач надсилає запит з відповідними даними на реєстрацію (чи вхід) у системі, сервер обробляє надіслану інформацію і генерує спеціальний об'єкт (Рис. 11), який називається *jwt* (Json Web Token). Цей секретний об'єкт зашифровує у собі дані про користувача і надсилається у відповідь клієнту. Клієнт зберігає його (наприклад, у *localStorage*) і тепер при кожному наступному запиті, що потребує авторизації, надсилає цей об'єкт серверу (у заголовку "Authorization").

Таким чином, сервер розшифровує даний об'єкт, перевіряючи авторизацію та визначаючи інформацію про користувача (Рис. 12).

```

async function signup(parent, args, context, info) {
  const userExists = await context.prisma.user.findUnique({
    where: { email: args.email },
  })

  if (userExists) {
    throw new Error('Користувача з такою поштою вже зареєстровано')
  }

  const password = await bcrypt.hash(args.password, 10)
  const user = await context.prisma.user.create({ data: { ...args, password } })
  const token = jwt.sign({ userId: user.id }, APP_SECRET)

  return {
    token,
    user,
  }
}

```

Рис.11. Реєстрація користувача

```

const jwt = require('jsonwebtoken')
const APP_SECRET = 'book-opinion-application'

function getTokenPayload(token) {
  return jwt.verify(token, APP_SECRET)
}

function getUserId(req, authToken) {
  if (req) {
    const authHeader = req.headers.authorization

    if (authHeader) {
      const token = authHeader.replace('Bearer ', '')

      if (!token) {
        throw new Error('No token found')
      }

      const { userId } = getTokenPayload(token)

      return userId
    }
  } else if (authToken) {
    const { userId } = getTokenPayload(authToken)

    return userId
  }

  throw new Error('Not authenticated')
}

module.exports = {
  APP_SECRET,
  getUserId,
}

```

Рис.12. Перевірка авторизації користувача

Розглянемо приклад функції-розпізнавача типу Query, що відповідає за отримання деталей розділу (Рис.13).

```
async function getSection(parent, args, context) {
  return await context.prisma.section.findUnique({
    where: { id: Number(args.filter) },
  })
}
```

Рис.13. Функція-розпізнавач для отримання деталей розділу

GraphQL-схему наведено у додатку 4, а решту функцій-розпізнавачів можна переглянути у додатку 5.

Повний програмний код знаходиться за [посиланням](#).

Програмна документація. Програмний комплекс розрахований на функціонування при такому наборі технічних засобів для мінімальної апаратної конфігурація комп'ютера-клієнта :

- процесор з тактовою частотою 1.4 ГГц ;
- ОЗУ, обсягом не менше 1 Гб ;
- відео карта та монітор з роздільною здатністю не менше 1024x768;
- клавіатура;
- маніпулятор(mouse, touchpad, touchscreen);
- вільний дисковий простір не менш ніж 100 мб.
- доступ до мережі Інтернет (5+ Мб/с)

3.4. Результати реалізації інформаційної системи

З метою перевірки відповідності реалізованої ІС до всіх вимог, було проведено апробацію на контрольному прикладі й перевірено весь створений функціонал.

Після відкриття веб-додатку у браузері, ми потрапляємо на головну сторінку (Рис. 14).

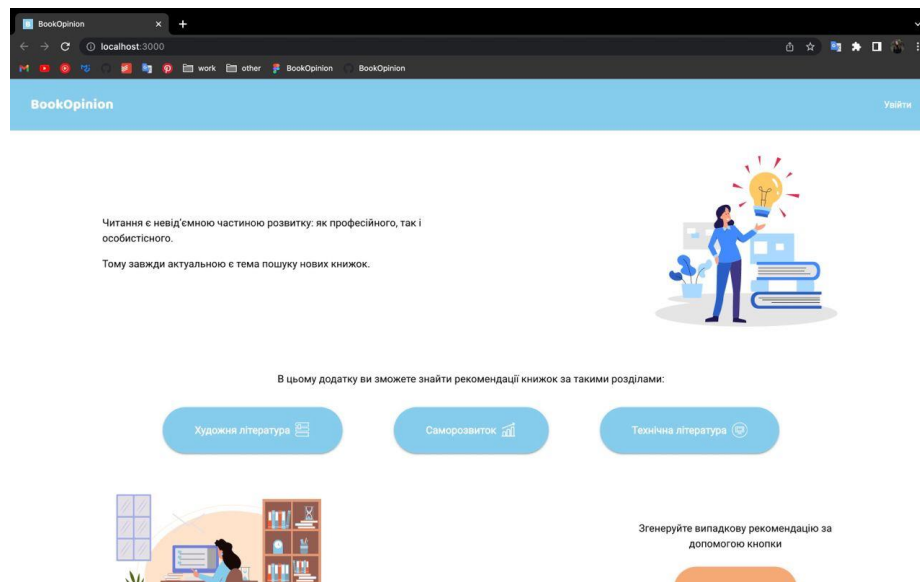


Рис.14. Головна сторінка для неавторизованого користувача

На головній сторінці зображена текстова інформація, ілюстрації та кнопки для переходу до літературних розділів та кнопка генерування випадкової рекомендації. Звернімо увагу, що справа від логотипу знаходиться кнопка «Увійти», натиснувши на яку нас буде перенаправлено на сторінку авторизації. Але відкладемо авторизацію на цьому кроці та подивимось на можливості неавторизованого користувача. Оберемо розділ, наприклад «Художня література» та натиснемо відповідну кнопку. Нас перенаправляє на сторінку цього розділу (Рис. 15).

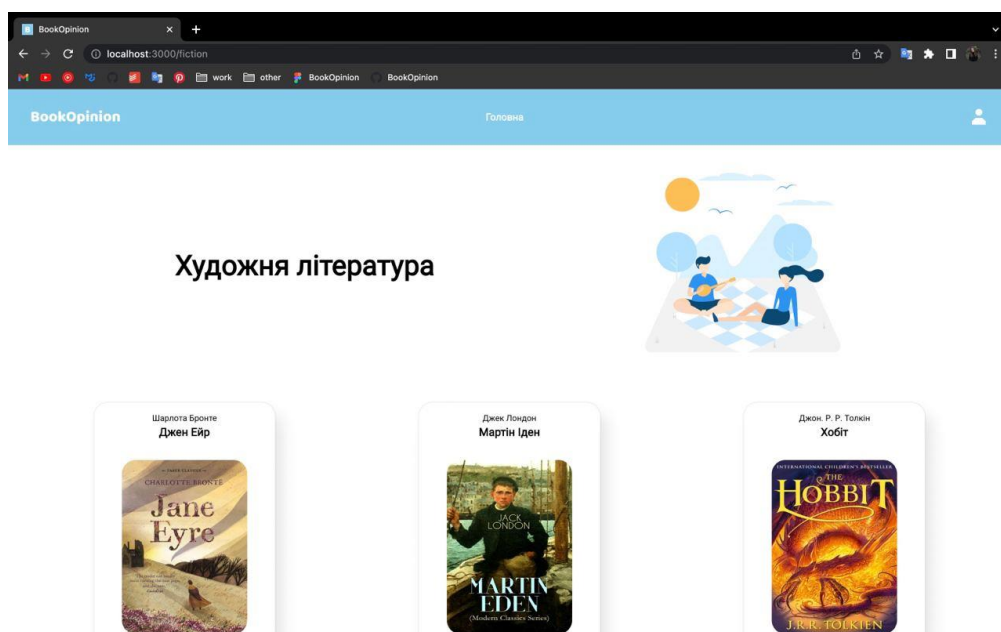


Рис.15. Сторінка розділу

Перейдемо до деталей рекомендації (Рис. 16).

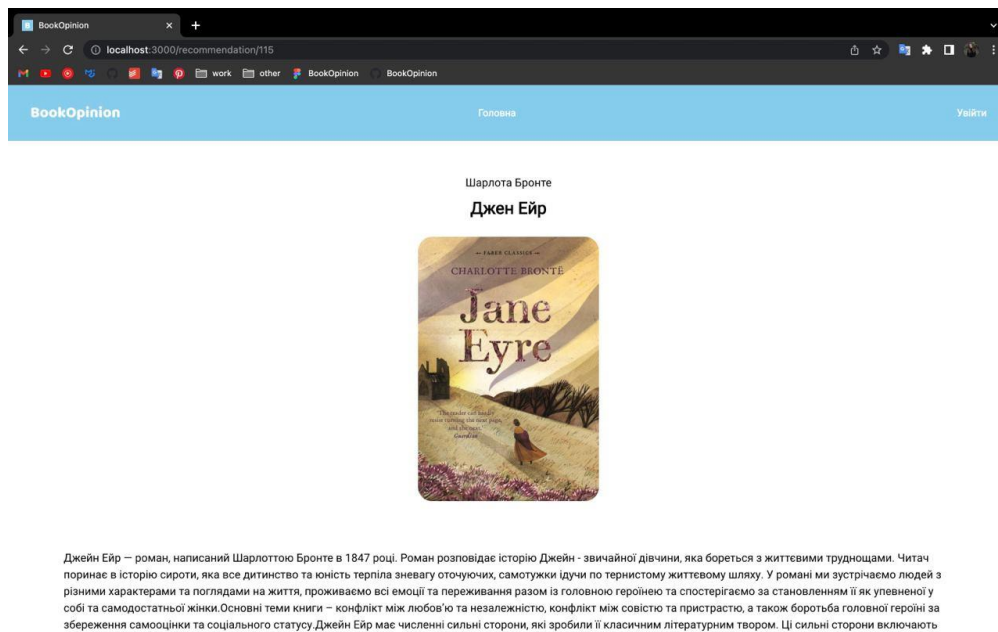


Рис.16. Сторінка рекомендації для неавторизованого користувача

Прогорнімо сторінку нижче та бачимо, що у рекомендації вже є коментарі від авторизованих користувачів. Система пропонує авторизуватись, щоб мати можливість написати коментар до рекомендації (Рис. 17).

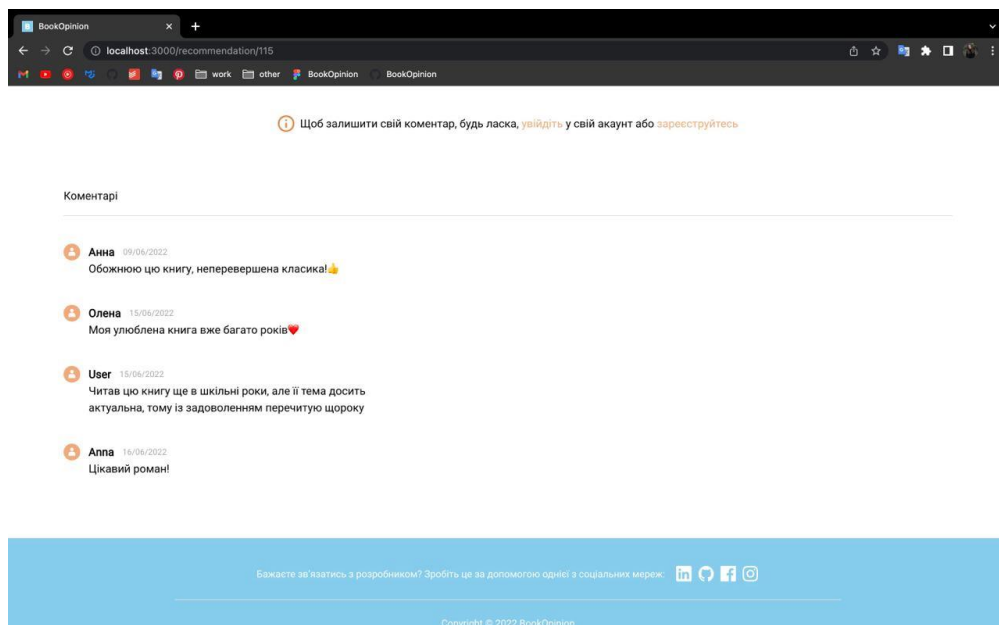


Рис.17. Блок коментарів до рекомендації для неавторизованого користувача

Якщо у рекомендації ще немає коментарів, даний блок буде виглядати наступним чином (Рис. 18).

Коментарі

У цієї рекомендації ще немає коментарів

Рис.18. Пустий блок коментарів

Спробуємо другий спосіб отримання рекомендації – через випадкову генерацію. Повертаємось на головну та натискаємо на кнопку «Згенерувати». У результаті нас так само перенаправляє на сторінку випадкової рекомендації, згенерованої системою (Рис. 19).

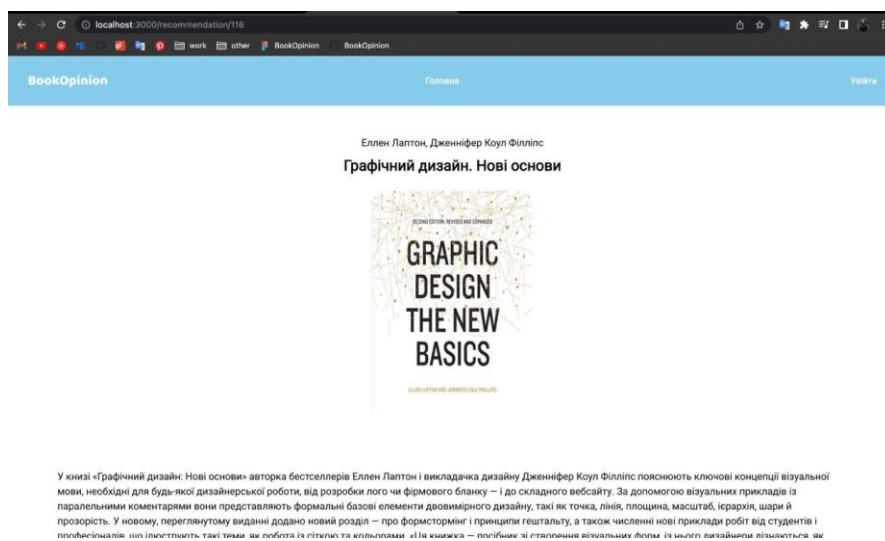


Рис.19. Сторінка випадково згенерованої рекомендації

Тепер перейдемо до авторизації, натиснувши на кнопку «Увійти» у верхньому правому кутку інтерфейсу (Рис. 20).

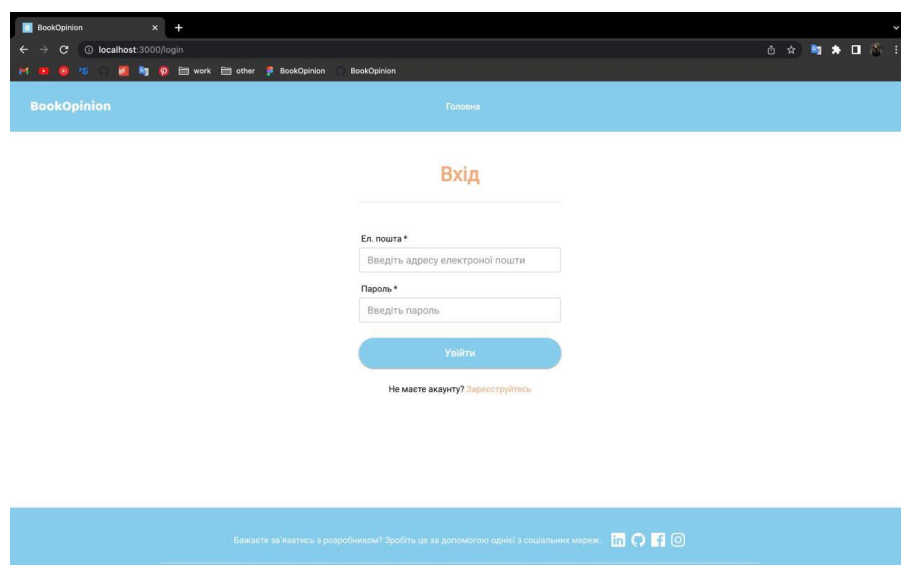


Рис.20. Сторінка входу

Звернімо увагу, що коли ми знаходимось на сторінці входу, то у верхньому правому кутку інтерфейсу зникає кнопка «Увійти», оскільки ми вже знаходимось на етапі входу в систему.

На етапі входу передбачено 2 можливих випадки появи помилок: коли було введено правильну пошту, але неправильний пароль (Рис. 21) та коли було введено неправильну пошту (Рис. 22).

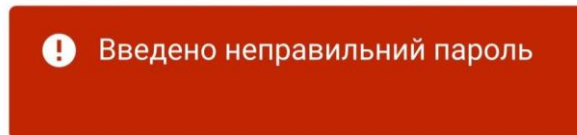


Рис.22. Повідомлення про неправильно введений пароль

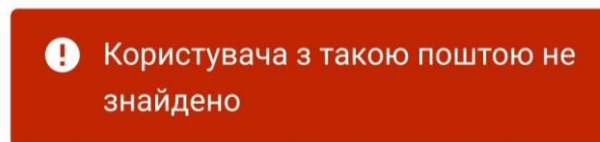


Рис.23. Повідомлення про неправильно введену пошту

Якщо ж всі дані введено вірно, то користувач увійде в систему та отримає відповідне повідомлення (Рис. 24).

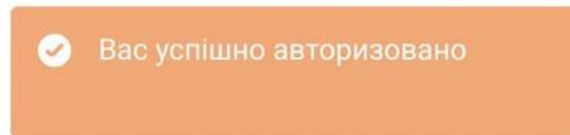
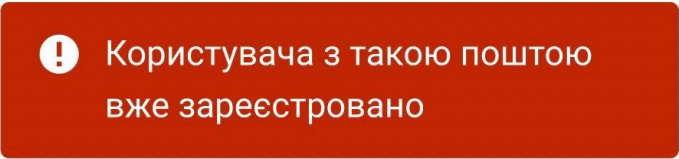


Рис.24. Повідомлення про успішний вхід в систему

Якщо у користувача ще немає акаунту, система пропонує його створити, натиснувши на посилання «Зареєструватись» під кнопкою «Увійти». Після цього нас перенаправляє на сторінку Реєстрації (Рис. 25).

Рис.25. Сторінка реєстрації

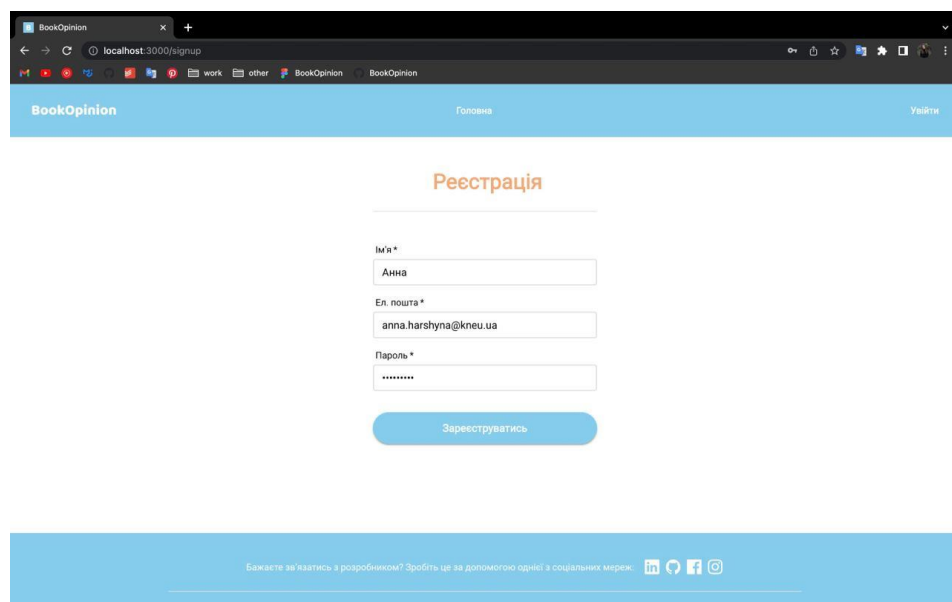
На етапі реєстрації передбачене одне повідомлення про помилку у випадку, коли користувач вводить пошту, яка вже зареєстрована в системі (Рис.26)



! Користувача з такою поштою
вже зареєстровано

Рис.26. Повідомлення про вже зареєстрованого користувача

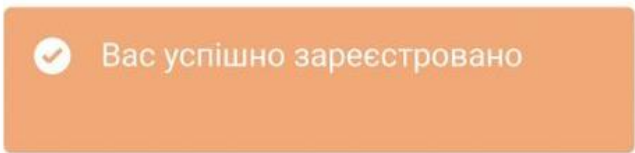
Введемо необхідні дані для реєстрації на натиснемо кнопку «Зареєструватися» (Рис. 27).



The screenshot shows a web browser window with the URL `localhost:3000/signup`. The page title is "BookOpinion" and the header includes "Головна" (Home) and "Увійти" (Login). The main heading is "Реєстрація" (Registration). The form contains three input fields: "Ім'я*" (Name) with the value "Анна", "Ел. пошта*" (Email) with the value "anna.harshyna@kneu.ua", and "Пароль*" (Password) with masked characters "*****". A blue button labeled "Зареєструватися" (Register) is at the bottom of the form. Below the form, there is a blue footer bar with the text "Бажаєте зв'язатися з розробником? Зробіть це за допомогою однієї з соціальних мереж:" followed by icons for LinkedIn, Telegram, Facebook, and Instagram.

Рис.27. Введені дані в форму реєстрації

Після успішної реєстрації з'являється наступне повідомлення (Рис. 28).



✓ Вас успішно зареєстровано

Рис.28. Повідомлення про успішну реєстрацію

Нас перенаправляє на головну сторінку, де у верхньому правому кутку нам тепер доступне меню (Рис. 29).

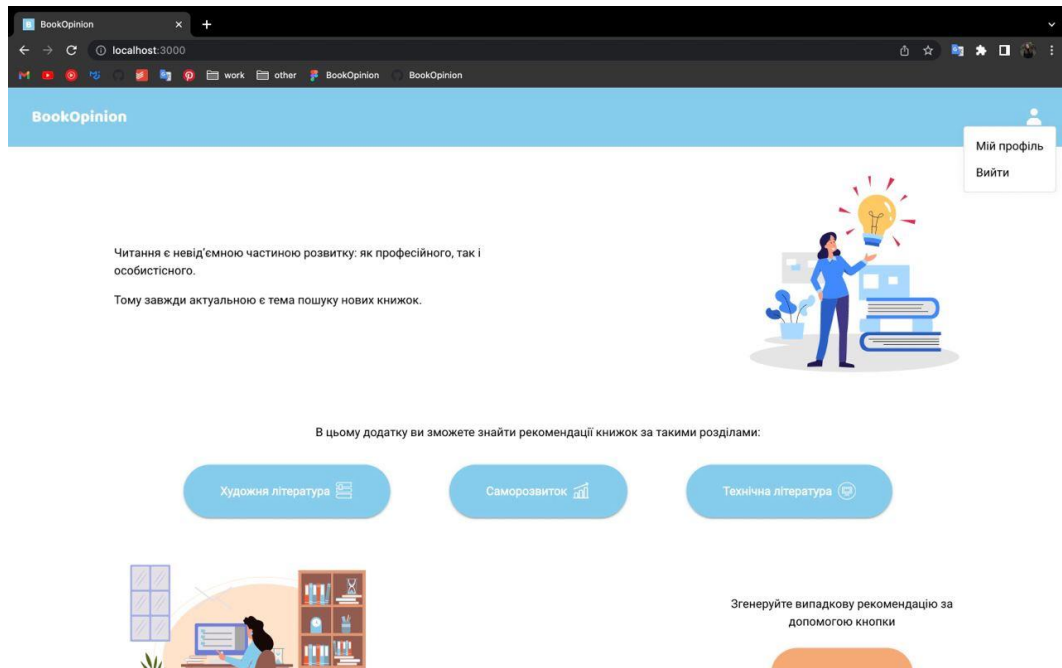


Рис.29. Головна сторінка для авторизованого користувача

Якщо натиснути на кнопку «Вихід» у меню, ми вийдемо з системи та отримаємо наступне повідомлення (Рис. 30).

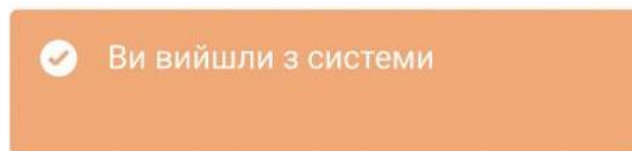


Рис.30. Повідомлення про вихід з системи

Переходимо до сторінки особистого акаунту натиснувши на кнопку «Мій профіль» у меню (Рис. 31).

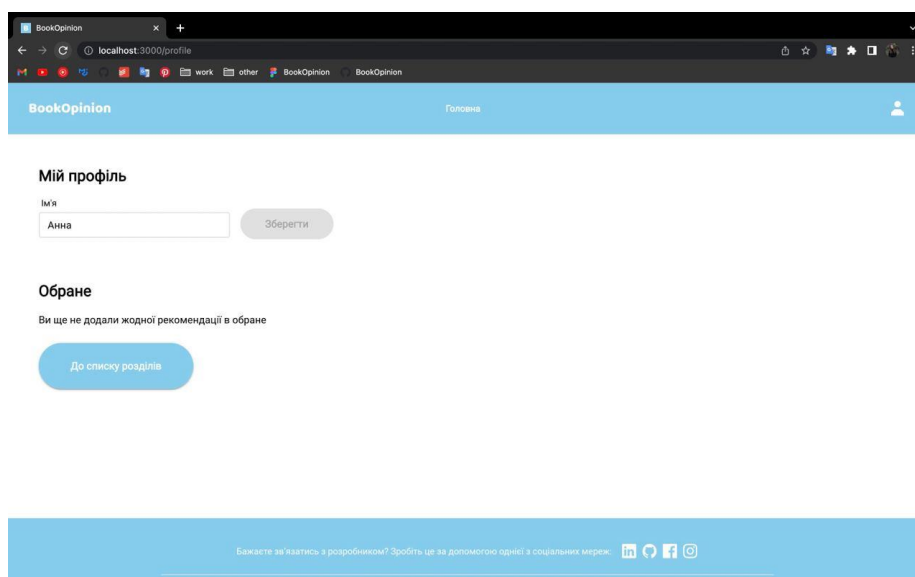


Рис.31. Сторінка особистого акаунту

Тут знаходиться форма для оновлення особистих даних. Бачимо, що кнопка «Зберегти» неактивна, оскільки ми ще не внесли змін в ім'я. Після зміни даних кнопка стає активною (Рис. 32).

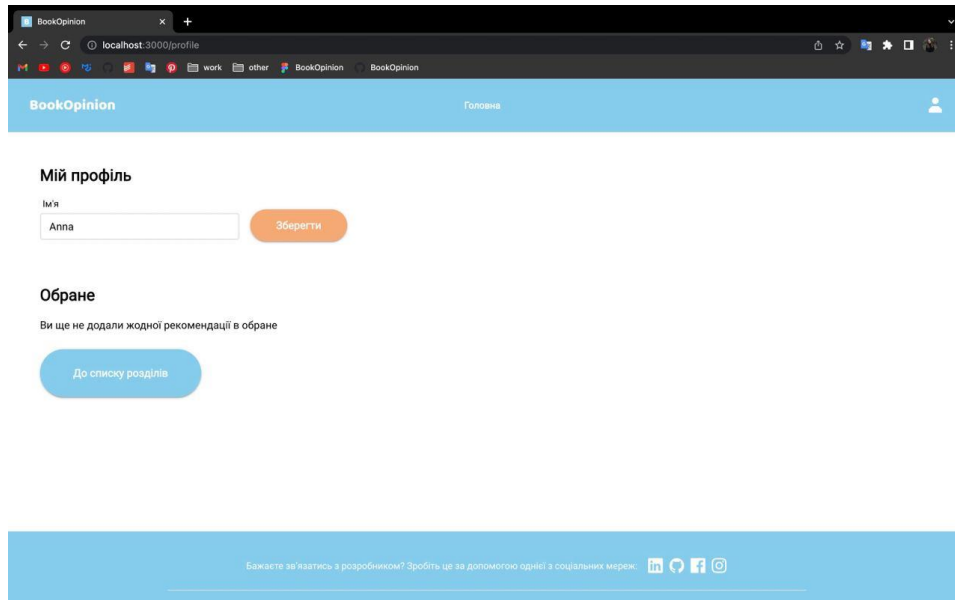


Рис.32. Введення нових даних у поле «Ім'я»

Натискаємо «Зберегти» та оновлюємо наші дані, про що свідчить відповідне повідомлення (Рис. 33).

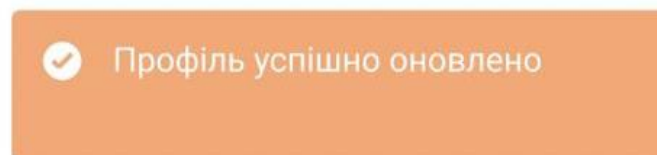


Рис.33. Повідомлення про успішне оновлення профілю

Також на сторінці профілю знаходиться список рекомендацій, які користувач додав в обране. Бачимо, що наразі список порожній. Додамо нову рекомендацію до обраного, перейшовши до списку розділів, обравши розділ, що нас цікавить та перейшовши до рекомендації (або ж згенерувавши випадкову рекомендацію на головній сторінці).

Бачимо, що коли користувач авторизований, на сторінці рекомендації з'являється кнопка для додавання рекомендації в обране у верхньому лівому кутку сторінки (Рис. 34).

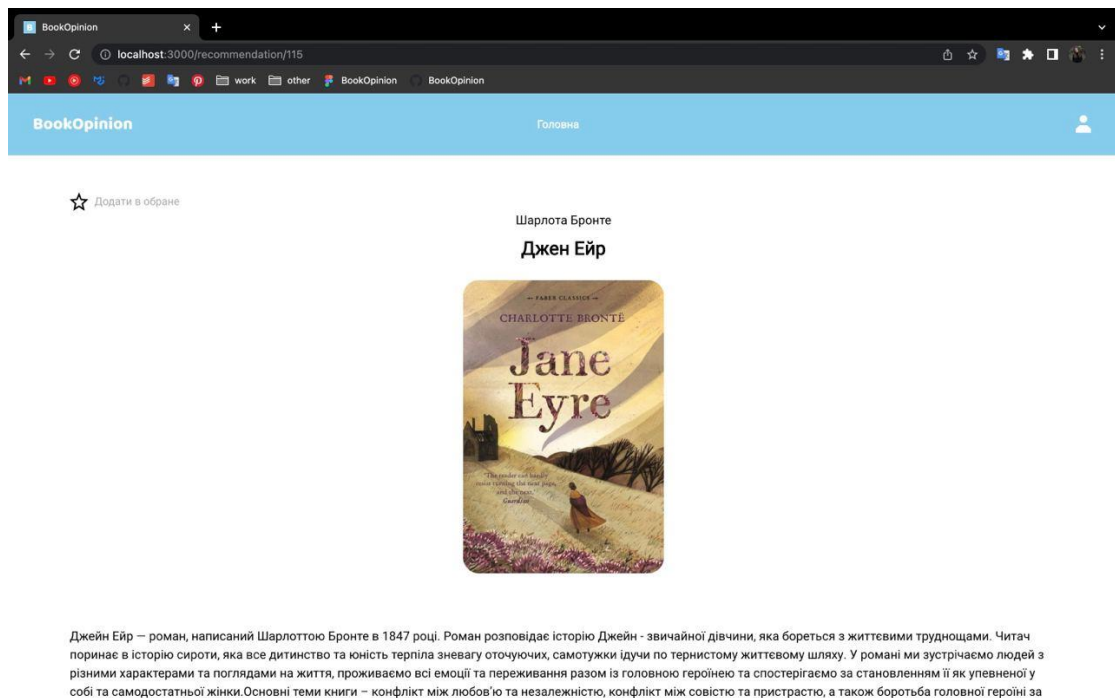


Рис.34. Сторінка рекомендації для авторизованого користувача

Після натискання на кнопку «Додати обране» з'являється відповідне повідомлення (Рис. 35) та кнопка змінюється (Рис.36). Якщо натиснути на кнопку тепер, то рекомендацію буде видалено з обраного та з'явиться повідомлення про це (Рис.37).



Рис.37. Повідомлення про успішне додавання рекомендації в обране

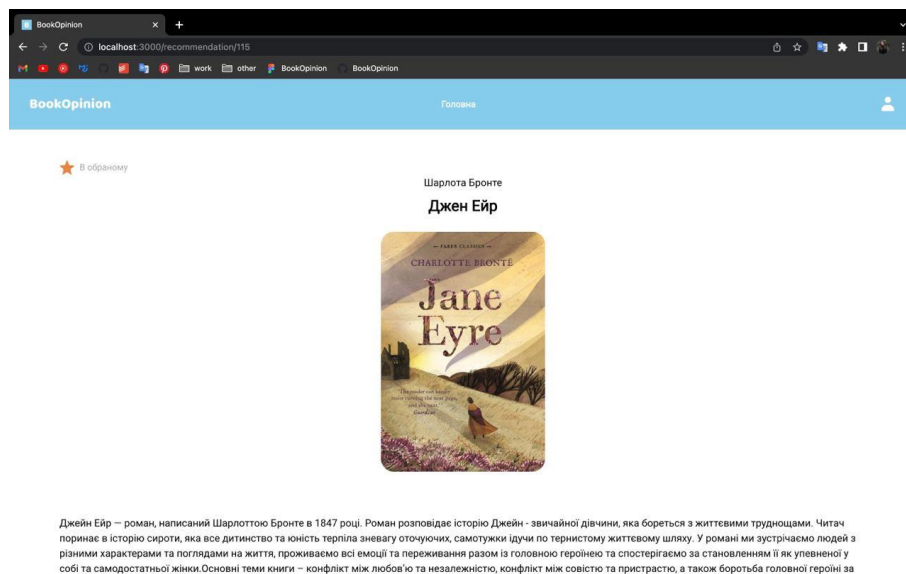


Рис.38. Сторінка рекомендації, коли рекомендацію було додано в обране



Рис.39. Повідомлення про успішне видалення рекомендації з обраного

Отже, ми додали одну рекомендацію в обране. Повертаємось до сторінки профілю та бачимо, що у нашому списку з'явилась обрана рекомендація. Це посилання, натиснувши на яке нас перенаправить на сторінку даної рекомендації (Рис. 40).

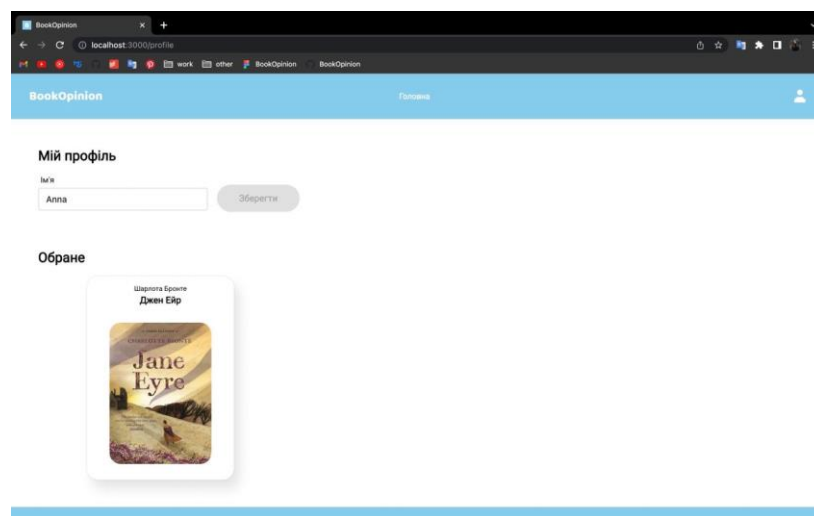


Рис.40. Сторінка особистого акаунту з непустим списком обраного

Тепер повернемося до блоку коментарів до рекомендації. Бачимо, що тепер нам доступна форма для написання коментарів, оскільки ми авторизовані в системі (Рис. 41).

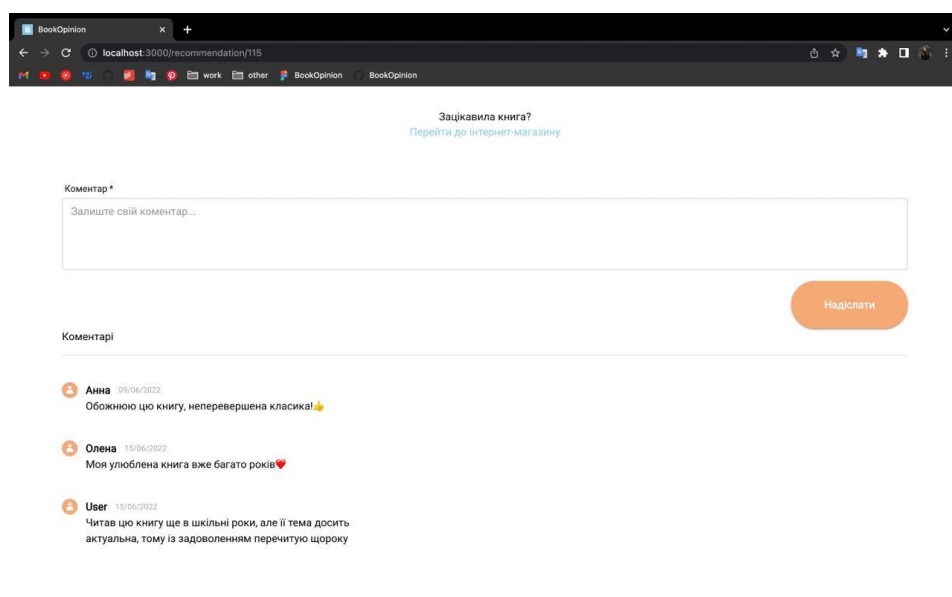
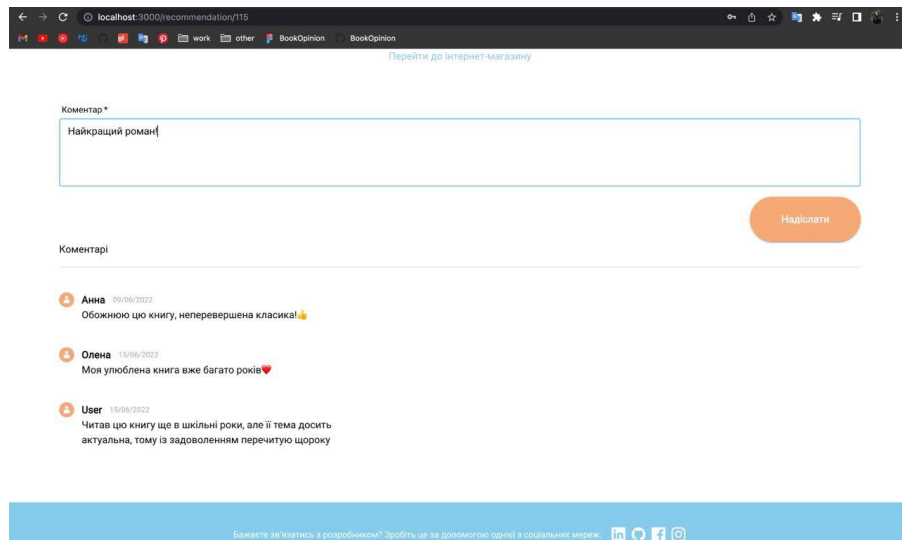


Рис.41. Блок коментарів для авторизованого користувача

Введемо свій коментар у форму (Рис.42) та натиснемо кнопку «Надіслати», після чого ми отримуємо повідомлення про успішну публікацію коментаря (Рис.43) та коментар з'являється у списку (Рис.44).



The screenshot shows a web browser window with the URL `localhost:3000/recommendation/115`. The page has a dark header with the text "Перейти до інтернет-магазину". Below the header is a form with a text input field containing "Найкращий роман!". To the right of the input field is an orange button labeled "Надіслати". Below the form is a section titled "Коментарі" containing a list of three comments:

- Анна** 09/06/2022: Обожаю цю книгу, неперевершена класика! 🌟
- Олена** 15/06/2022: Моя улюблена книга вже багато років ❤️
- User** 15/06/2022: Читав цю книгу ще в шкільні роки, але її тема досить актуальна, тому із задоволенням перечитую щороку

At the bottom of the page is a blue banner with the text: "Бажаєте зв'язатись з розробником? Зробіть це за допомогою однієї з соціальних мереж:" followed by icons for LinkedIn, Facebook, and Instagram.

Рис.42. Введення даних у форму для коментарів

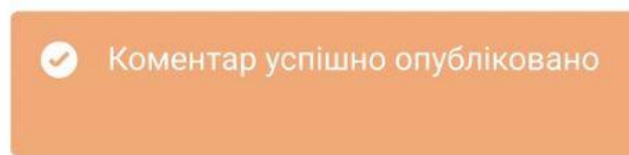
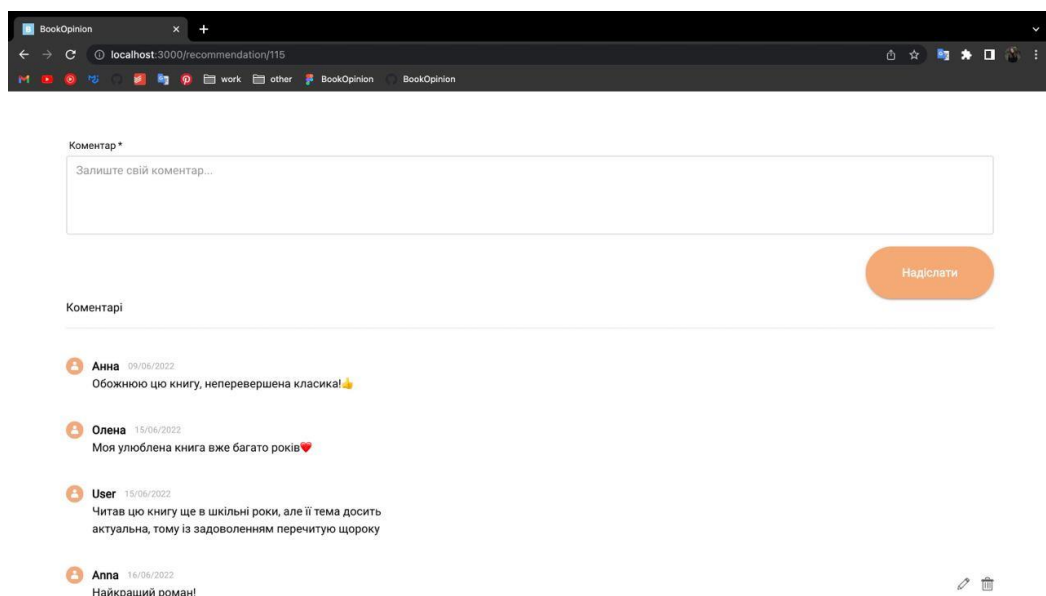


Рис.43. Повідомлення про успішну публікацію коментаря



The screenshot shows the same web browser window as in Figure 42. The text input field now contains the placeholder "Залиште свій коментар...". The "Надіслати" button is still present. The "Коментарі" section now contains four comments, with the new one at the bottom:

- Анна** 09/06/2022: Обожаю цю книгу, неперевершена класика! 🌟
- Олена** 15/06/2022: Моя улюблена книга вже багато років ❤️
- User** 15/06/2022: Читав цю книгу ще в шкільні роки, але її тема досить актуальна, тому із задоволенням перечитую щороку
- Анна** 16/06/2022: Найкращий роман! ✎ 🗑

At the bottom right of the comment list, there are icons for editing (pencil) and deleting (trash) the last comment.

Рис.44. Оновлений список коментарів

Бачимо, що автору коментаря доступні такі дії: редагування та видалення коментаря. Якщо натиснути на кнопку редагування, то з'являється вікно з формою, де користувач може відредагувати обраний коментар (Рис.

45). Користувач може відмінити дану дію натиснувши на кнопку «Ні» або за межами вікна. Введемо нову текстову інформацію у форму та натиснемо кнопку «Так».

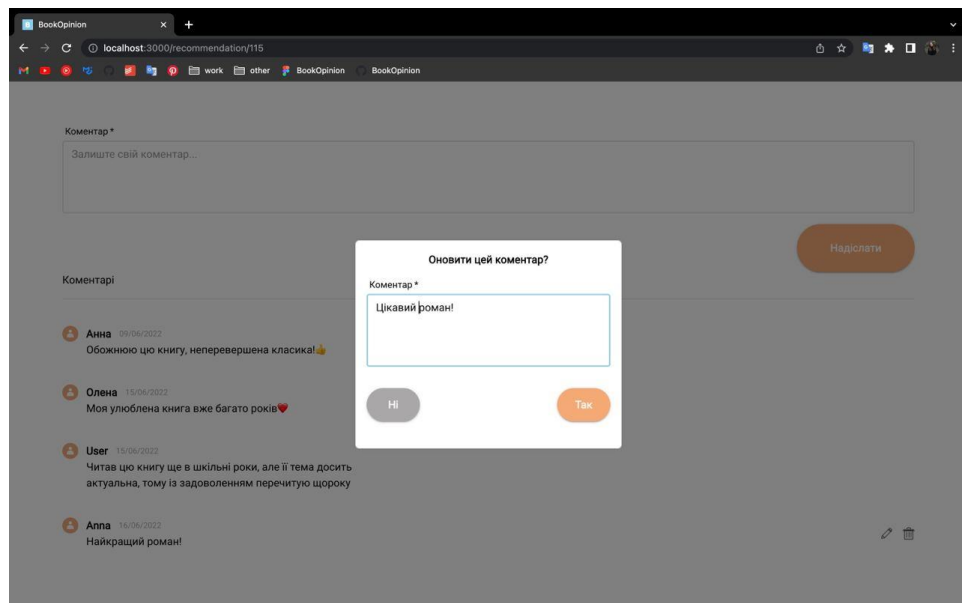


Рис.45. Вікно з формою для редагування коментаря

З'являється повідомлення про успішне оновлення коментаря (Рис.46) та бачимо оновлений коментар у списку (Рис.47)

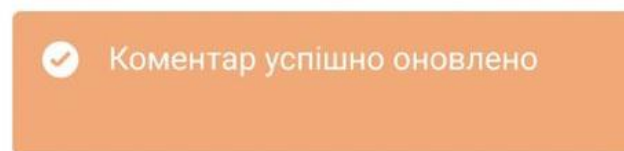


Рис.46. Повідомлення про успішне оновлення коментаря

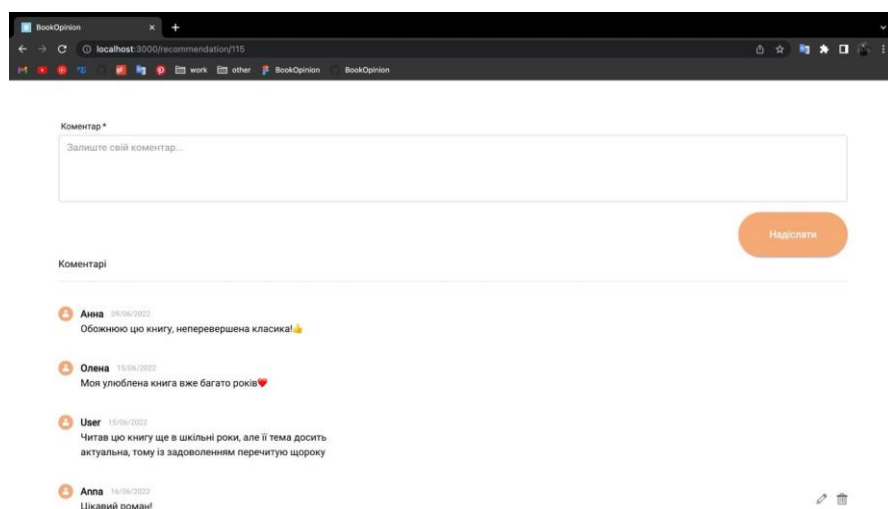


Рис.47. Оновлений список коментарів після редагування коментаря

Спробуємо видалити коментар, натиснувши на відповідну кнопку. З'являється вікно для підтвердження дії (Рис. 48). Аналогічно дії редагування

користувач може відмінити дану дію натиснувши на кнопку «Ні» або за межами вікна. Підтверджуємо дію видалення натисканням на кнопку «Так».

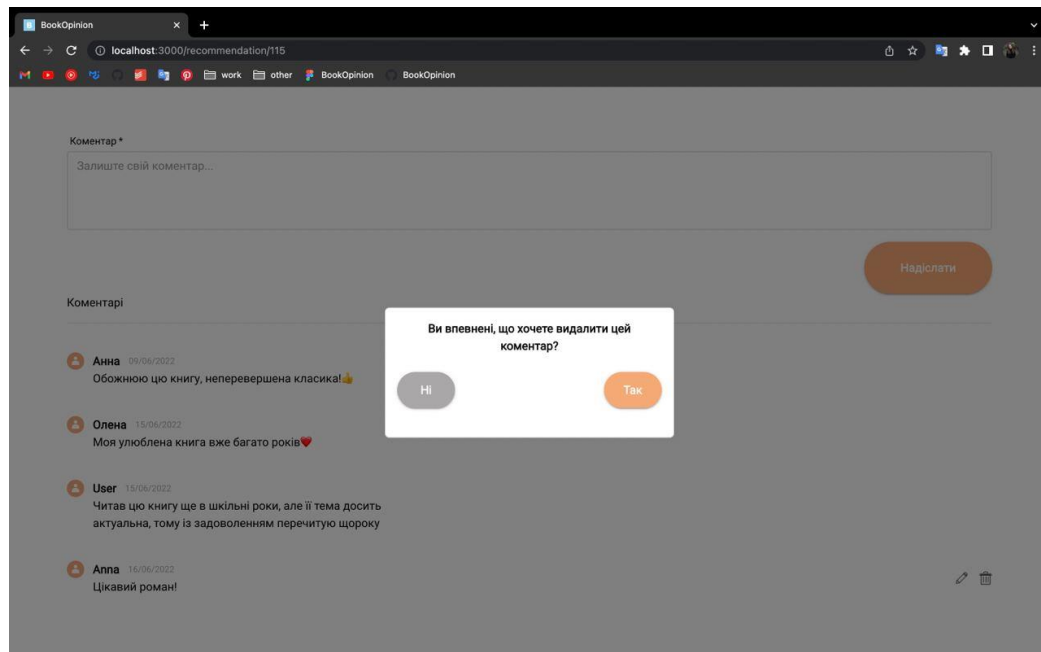


Рис.48. Вікно для підтвердження видалення коментаря

Після видалення коментаря з'являється повідомлення (Рис.49) та оновлений список коментарів, де вже немає видаленого коментаря (Рис.50)

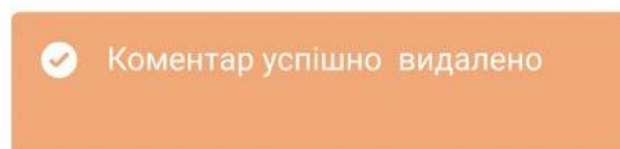


Рис.49. Повідомлення про успішне видалення коментаря

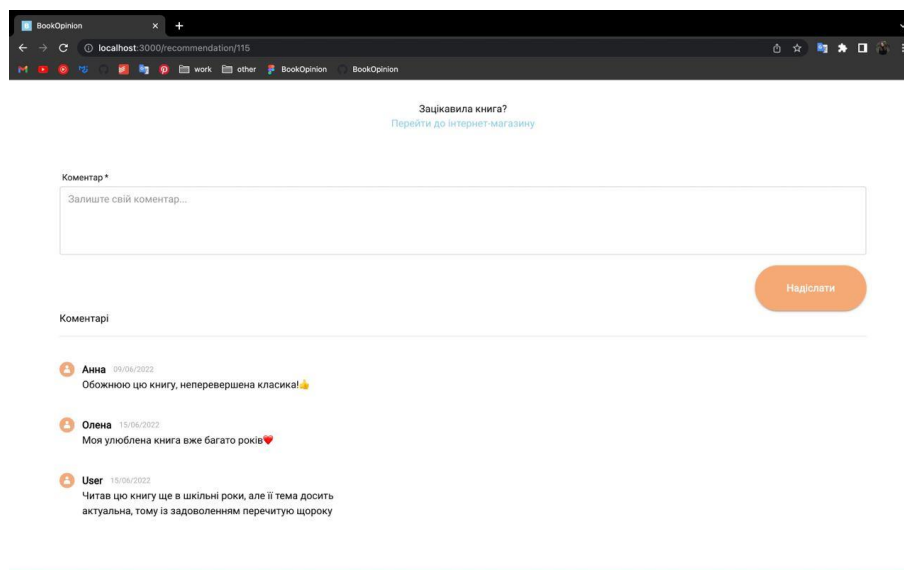


Рис.50. Оновлений список коментарів після видалення коментаря

Розглянемо випадки, коли користувач намагатиметься потрапити на сторінку, якої не існує або до якої в нього немає доступу.

- Якщо користувач спробує відкрити сторінку, яка була видалена або самостійно введе адресу неіснуючої сторінки, він побачить таку сторінку (Рис.51)

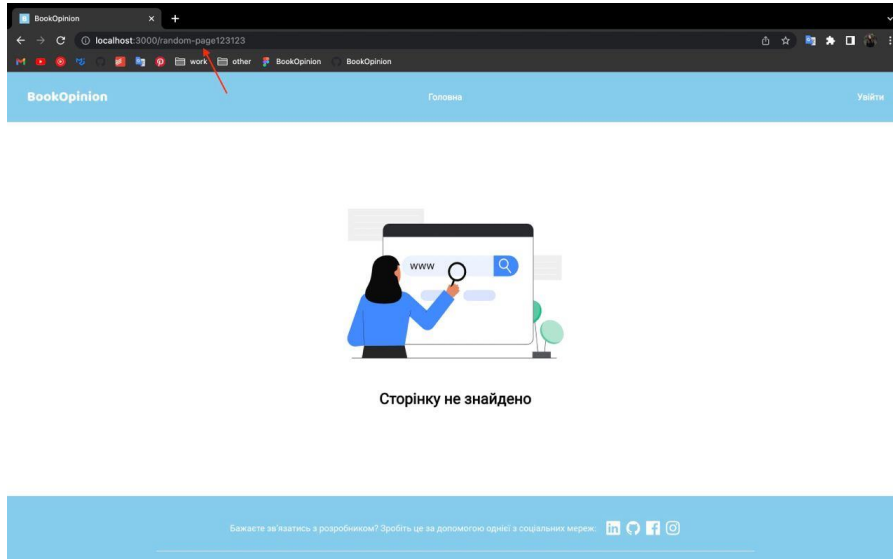


Рис.51. Вигляд сторінки, яку не знайдено

- Якщо зареєстрований користувач введе у адресному полі адресу сторінки входу або реєстрації, то його перенаправить на головну сторінку.
- Якщо користувач додав у обране браузеру сторінку профіля, вийшов з системи та через деякий час спробує відкрити сторінку профіля з обраного, його перенаправить на сторінку входу.

ВИСНОВКИ

В результаті роботи над випускним бакалаврським проектом було спроектовано та розроблено інформаційну систему з надання літературних рекомендацій «BookOpinion». Було створено:

- Специфікації вимог
- Дизайн-систему
- API
- Серверну частину
- Клієнтську частину
- Функції-розпізнавачі для з'єднання клієнтської частини з серверною

Для реалізації програмного продукту мною були обрані найсучасніші технології веб-програмування та роботи з клієнт-серверною архітектурою. Оскільки я вже працюю за профілем вибраного напрямку підготовки в ТОВ «Аксон Текнолоджіс», рішення з вибору мов програмування, технологій та бібліотек були прийняті після аналізу найактуальніших технологій, що використовує підприємство.

ІС «BookOpinion» забезпечує:

- Авторизацію та автентифікацію користувачів системи
- Зручну та інтуїтивно зрозумілу навігацію по користувацькому інтерфейсу
- Можливість переглядати контент ІС в якості неавторизованого користувача
- Зручне створення, редагування та видалення власних коментарів під рекомендаціями
- Зручний перегляд списку обраного, а також додавання та видалення рекомендацій з нього
- Можливість редагувати особисті дані

ІС «BookOpinion» може широко застосовуватися у повсякденному житті користувачами будь-якого віку та професії, які бажають отримати нові літературні рекомендації. При цьому вони самі можуть вибрати літературний розділ, який їх цікавить або ж згенерувати випадкову рекомендацію з випадкового розділу.

Існуючі можливості проекту вже є досить корисним та актуальним, проте за бажанням можна розширювати список вимог до ІС та додавати до неї новий функціонал. Водночас нові можливості системи можуть бути впроваджені дуже швидко та якісно, оскільки для її реалізації була обрані потужні та новітні технології веб-програмування, чії можливості наразі безмежні.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. GetAbstract - The most comprehensive library of compressed knowledge. [Електронний ресурс] – Режим доступу до ресурсу: <https://www.getabstract.com/>
2. Sumaries - Fresh ideas from top business books. [Електронний ресурс] – Режим доступу до ресурсу: <https://summaries.com/>
3. SQLite – The most popular database engine in the world. [Електронний ресурс] – Режим доступу до ресурсу: <https://www.sqlite.org/index.html>
4. Prisma – Next-generation ORM. [Електронний ресурс] – Режим доступу до ресурсу: <https://www.prisma.io/>
5. React – A JavaScript library for building user interfaces [Електронний ресурс] – Режим доступу до ресурсу: <https://reactjs.org/>
6. GraphQL – A query language for your API [Електронний ресурс] : Режим доступу : <https://graphql.org/>
7. М. Haverbeke. Eloquent JavaScript: A modern introduction into programming. 2018. – 435с.
8. THE ULTIMATE GUIDE TO API ARCHITECTURE: REST, SOAP OR GRAPHQL? [Електронний ресурс] // DA-14. – 2018. – Режим доступу до ресурсу: <https://da-14.com/blog/ultimate-guide-api-architecture-rest-soap-or-graphql>
9. JavaScript-Apollo tutorial - Introduction [Електронний ресурс] – Режим доступу до ресурсу: <https://www.howtographql.com/graphql-js/0-introduction/>
10. React-Apollo tutorial - Introduction [Електронний ресурс] – Режим доступу до ресурсу: <https://www.howtographql.com/react-apollo/0-introduction/>
11. Ситник Н.В. Проектування баз і сховищ даних: Нав. посіб. – К.:2004. – 348 с
12. Nicholas C. Zakas. Next-Generation JavaScript Engines // High Performance JavaScript / 2010. —ISBN 978-0-596-80279-0.

ДОДАТКИ

Додаток 1

Таблиці специфікацій вимог

Таблиця 7. Специфікація бізнес-вимог

Ідентифікатор	Стисла назва	Повна назва	Тип	Статус	Складність	Пріоритетність	Ризикованість
110	Авторизація	Наявність авторизації	Performance	Mandatory	High	High	High
111	Вхід	Вхід в систему	Performance	Mandatory	High	High	High
112	Реєстрація	Реєстрація в системі	Performance	Mandatory	High	High	High
120	Взаємодія з користувачем	Взаємодія з користувачем	Performance	Mandatory	High	High	High
121	Список обраного	Надання функціоналу зі списком обраного	Performance	Mandatory	High	High	High
122	Коментарі	Надання функціоналу коментарів	Performance	Mandatory	High	High	High
130	Структуризація контенту	Структуризація контенту	Display	Mandatory	High	High	High
131	Розділи	Розподіл рекомендацій по розділам	Display	Mandatory	High	High	High

Таблиця 8. Специфікація функціональних вимог

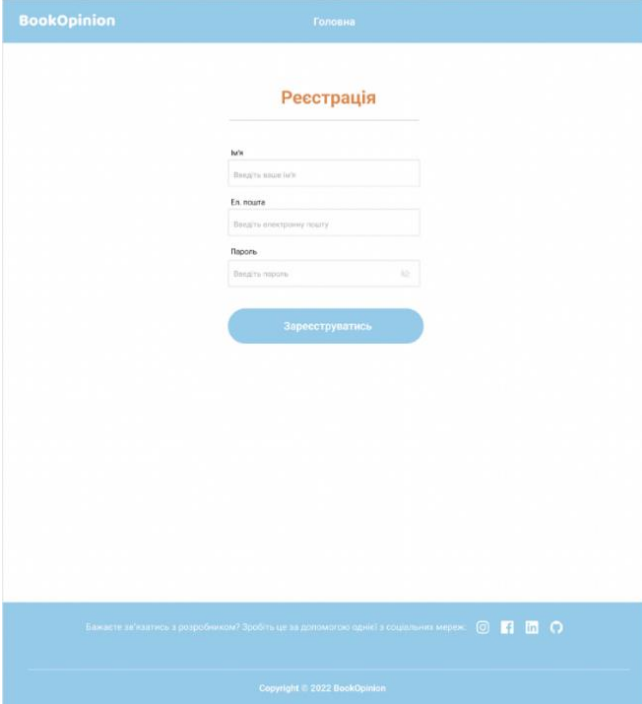
Ідентифікатор	Стисла назва	Повна назва	Тип	Статус	Складність	Пріоритетність	Ризикованість
210	Доступ	Надавати користувачам доступ до системи	Functional	Mandatory	High	High	Medium
211	Вхід	Авторизація в системі	Functional	Mandatory	High	High	Medium
212	Реєстрація	Реєстрація в системі	Functional	Mandatory	High	High	Medium
213	Валідація	Валідація користувача	Functional	Mandatory	High	High	Medium
214	Дані про користувачів	Зберігання даних про користувача	Functional	Mandatory	High	High	Medium
215	Оновлення даних	Можливість оновлення своїх даних	Functional	Mandatory	High	High	Medium
220	Структуризація	Структуризація контенту	Functional	Mandatory	Medium	High	Low
221	Розділи	Наявність розділів	Functional	Mandatory	Medium	High	Low
222	Списки	Розміщення списків рекомендацій по розділам	Functional	Mandatory	Medium	High	Low
223	Деталі	Розміщення деталей рекомендацій по спискам	Functional	Mandatory	Medium	High	Low

224	Список коментарів	Розміщення списку коментарів під деталями рекомендації	Functional	Mandatory	Medium	High	Low
230	Генерування	Можливість генерування випадкової рекомендації	Functional	Mandatory	Medium	High	Low
231	Алгоритм	Наявність математичного алгоритму випадкового генерування	Functional	Mandatory	Medium	High	Low
240	Коментарі	Система коментарів	Functional	Mandatory	Medium	High	Low
241	Додавання коментарів	Додавання коментарів	Functional	Mandatory	Medium	High	Low
242	Редагування коментарів	Редагування коментарів	Functional	Mandatory	Medium	High	Low
243	Видалення коментарів	Видалення коментарів	Functional	Mandatory	Medium	High	Low
244	Дані про коментарі	Збереження даних про коментарі	Functional	Mandatory	Medium	High	Low
250	Список обраного	Система списку обраного	Functional	Mandatory	Medium	High	Low
251	Додавання в список обраного	Додавання в список обраного	Functional	Mandatory	Medium	High	Low
252	Видалення зі списку обраного	Видалення зі списку обраного	Functional	Mandatory	Medium	High	Low
253	Дані про список обраного	Збереження даних про список обраного	Functional	Mandatory	Medium	High	Low

Таблиця 9. Специфікація нефункціональних вимог

Ідентифікатор	Стисла назва	Повна назва	Тип	Статус	Складність	Пріоритетність	Ризикованість
010	Адаптивність	Система має бути адаптивною	Performance	Mandatory	Medium	High	Low
011	Браузерна адаптивність	Система має коректно працювати в усіх основних браузерах	Performance	Mandatory	Medium	High	Low
012	Адаптивність відносно пристроїв	Система має коректно відобразитись на всіх можливих розширеннях екранів	Performance	Mandatory	Medium	High	Low
020	Швидкодія	Система має бути швидкодіююю	Performance	Mandatory	High	High	High
021	Швидкодія спілкування з БД	Час реакції системи на запити користувача е повинен перевищувати 3 сек	Performance	Mandatory	High	High	High
022	Швидкодія роутінгу	Перехід між екранами системи має бути моментальним	Performance	Mandatory	High	High	High

Прототипи екранів ІС «BookOpinion»



The registration screen features a blue header with the 'BookOpinion' logo and a 'Головна' (Home) link. The main content area is white with a light gray border. At the top, the title 'Реєстрація' (Registration) is displayed in orange. Below it, there are three input fields: 'Ім'я' (Name) with placeholder 'Введіть ваше ім'я', 'Ел. пошта' (Email) with placeholder 'Введіть електронну пошту', and 'Пароль' (Password) with placeholder 'Введіть пароль' and a visibility toggle icon. A blue 'Зареєструватись' (Register) button is positioned below the fields. The footer is blue and contains a link to the developer's portfolio, social media icons, and a copyright notice for 2022.

BookOpinion Головна

Реєстрація

Ім'я
Введіть ваше ім'я

Ел. пошта
Введіть електронну пошту

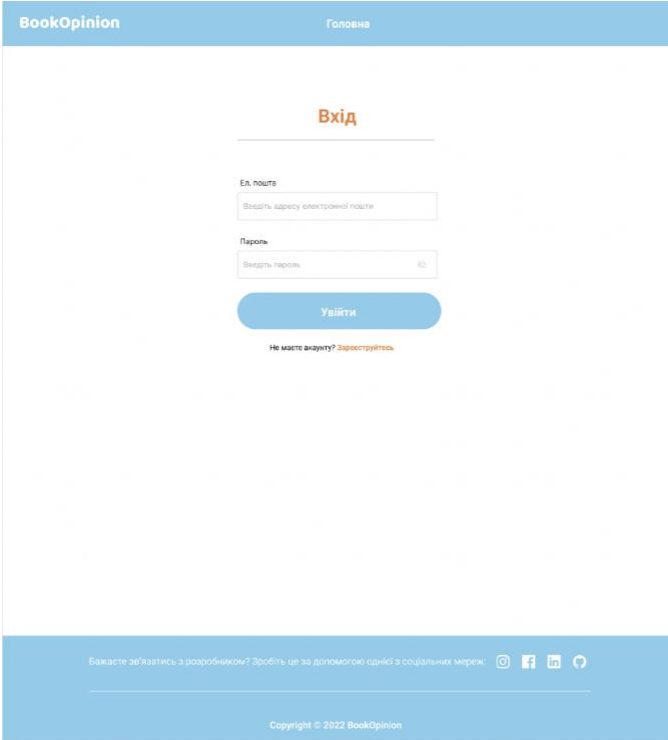
Пароль
Введіть пароль

Зареєструватись

Бажаєте зв'язатись з розробником? Зробіть це за допомогою однієї з соціальних мереж: [Instagram](#) [Facebook](#) [LinkedIn](#) [Twitter](#)

Copyright © 2022 BookOpinion

Рис.1. Прототип сторінки реєстрації



The login screen features a blue header with the 'BookOpinion' logo and a 'Головна' (Home) link. The main content area is white with a light gray border. At the top, the title 'Вхід' (Login) is displayed in orange. Below it, there are two input fields: 'Ел. пошта' (Email) with placeholder 'Введіть адресу електронної пошти' and 'Пароль' (Password) with placeholder 'Введіть пароль' and a visibility toggle icon. A blue 'Увійти' (Login) button is positioned below the fields. Below the button, there is a link 'Не маєте акаунту? Зареєструйтесь' (Don't have an account? Register). The footer is blue and contains a link to the developer's portfolio, social media icons, and a copyright notice for 2022.

BookOpinion Головна

Вхід

Ел. пошта
Введіть адресу електронної пошти

Пароль
Введіть пароль

Увійти

Не маєте акаунту? [Зареєструйтесь](#)

Бажаєте зв'язатись з розробником? Зробіть це за допомогою однієї з соціальних мереж: [Instagram](#) [Facebook](#) [LinkedIn](#) [Twitter](#)

Copyright © 2022 BookOpinion

Рис.2. Прототип сторінки входу



Рис.3. Прототип головної сторінки (для великого розширення екрану)



Рис.4. Прототип головної сторінки (для маленького розширення екрану)

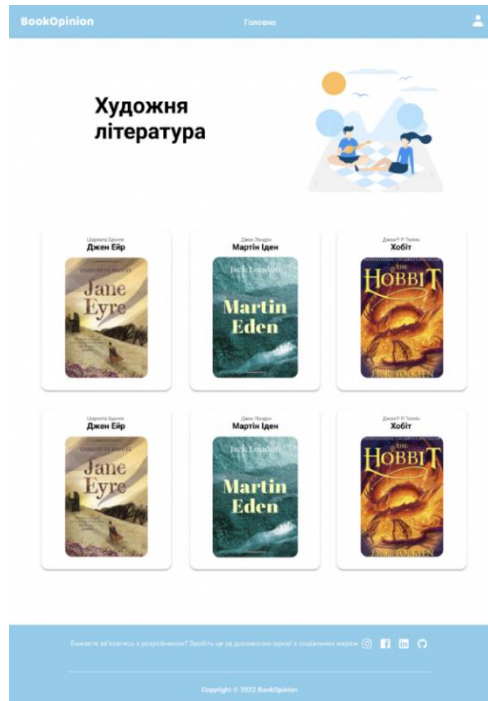


Рис.5. Прототип сторінки розділу (для великого розширення екрану)

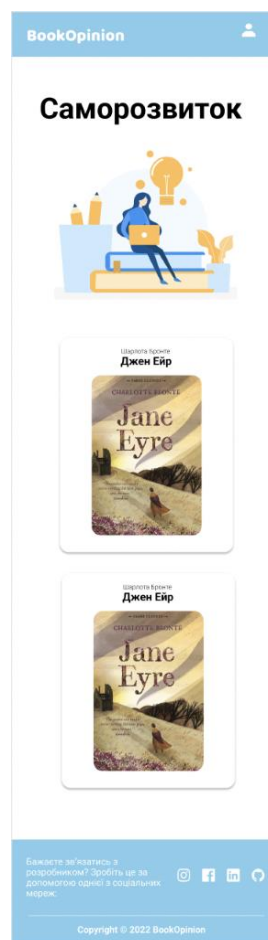


Рис.6. Прототип сторінки розділу (для маленького розширення екрану)



Рис.7. Прототип сторінки рекомендації (для великого розширення екрану)

BookOpinion

Шарлота Бронте

Джен Ейр

Lorem ipsum dolor sit amet, consectetur
 adipiscing elit. Pellentesque eu mi ornare
 ligula mattis pretium. Vestibulum rutrum a
 leo id vulputate. Duis tempor ante ac congue
 ornare. Praesent velit nulla, accumsan vitae
 ante vitae, viverra tincidunt elit. Suspendisse
 nec aliquam orci, ut blandit enim. Donec et
 neque volutpat libero rutrum rhoncus.
 Pellentesque sed porttitor tellus. Aliquam
 eget ullamcorper est, at fringilla velit.
 Vestibulum ac augue imperdiet justo fringilla
 efficitur. Nulla molestie est non dui aliquam,
 facilisis tristique nisi posuere. Fusce quam
 ipsum, ullamcorper vel egestas id, aliquam
 eget nibh. Duis nec sapien quis enim placerat
 scelerisque quis at tellus. In hac habitasse
 platea dictumst. Etiam vitae ultricies felis, id
 egestas tortor. Etiam at tincidunt libero.
 Pellentesque sed luctus ex.

Зацікавила книга?

Перейти до інтернет-магазину

Коментар

Залиште свій коментар...

Надіслати

Коментарі

Анна

09.04.2020

Чудова книга, рекомендую!

Бажаєте зв'язатись з розробником? Зробіть це за допомогою одного з соціальних мереж:

Copyright © 2022 BookOpinion

Рис.8. Прототип сторінки рекомендації (для маленького розширення екрану)

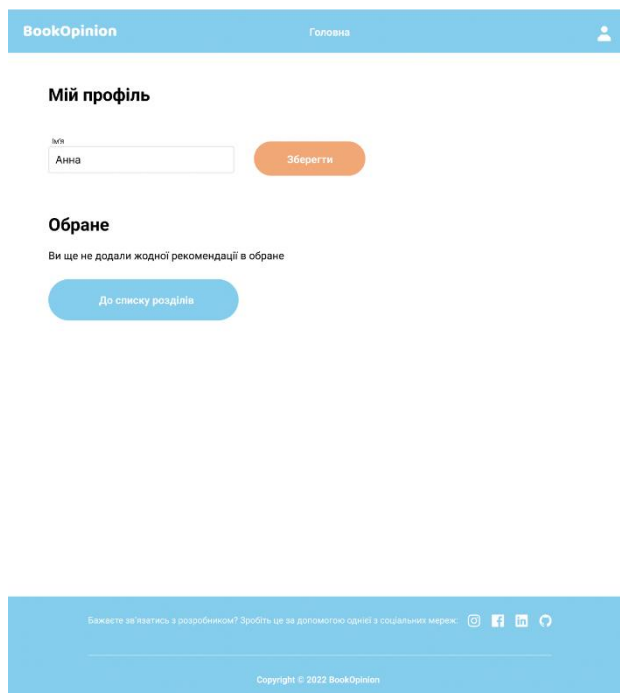


Рис.9. Прототип сторінки профілю (для великого розширення екрану)



Рис.10. Прототип сторінки профілю (для маленького розширення екрану)

Ви впевнені, що хочете видалити цей коментар?

Ні Так

Оновити цей коментар?

Чудова книга, рекомендую!

Ні Так

Рис.11. Прототип екранних форм

✓ Коментар успішно опубліковано

✗ Користувача з такою адресою пошти вже зареєстровано

✓ Коментар успішно видалено

✗ Введено неправильний пароль

✓ Коментар успішно оновлено

✓ Вас успішно зареєстровано

✓ Вас успішно авторизовано

✓ Ви вийшли з системи

✓ Рекомендацію додано в обране

✓ Рекомендацію видалено з обраного

✓ Профіль успішно оновлено

Рис.12. Список усіх повідомлень у системі

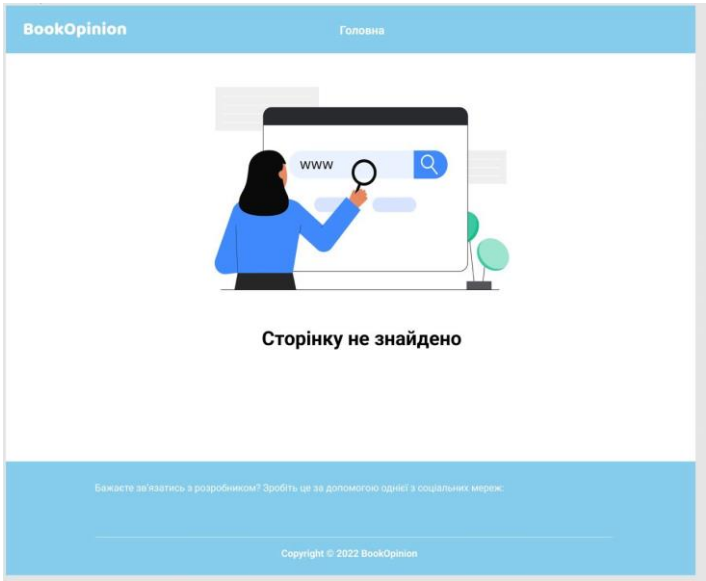


Рис.13. Прототип сторінки «Не знайдено»

Додаток 3

Тестові дані, завантажені в базу даних

User × +

C

FiltersNone

FieldsAll

Showing6 of 6

Add record

☐	id #	name A	email A	password A	comments {}	favorites {}
☐	9	Анна	anna@kneu.ua	\$2a\$10\$4C8s8S1TQH01hL...	4 Comment	3 Favorite
☐	32	Олена	olena@gmail.com	\$2a\$10\$beQ79mUyashQ06...	1 Comment	0 Favorite
☐	33	Олексій	oleksiy@gmail.com	\$2a\$10\$CH21sqHXfgQdr3...	1 Comment	1 Favorite
☐	34	User	user@gmail.com	\$2a\$10\$IhfoDBKwDk1DC...	1 Comment	0 Favorite
☐	35	Maria	maria123@gmail.com	\$2a\$10\$ZnmF17Xd1Ou6cm...	0 Comment	0 Favorite
☐	36	Anna	anna.harshyna@kneu.ua	\$2a\$10\$KMitzS3hdf5WU...	0 Comment	1 Favorite

Рис.14. Таблиця «Користувач»

Recommendation												
C FiltersNoneFieldsAllShowing 12 of 12Add record												
	id	#	title	author	description	shopUrl	imageUrl	comments				
	115		Джен Ейр	Шарлота Бронте	Джені Ейр – роман, напи...	https://www.yakaboo.u...	https://i.ibb.co/c76b...	3 Comment				
	116		Мартін Ізен	Джек Лондон	Мартін Ізен – збіжжя в...	https://www.yakaboo.u...	https://i.ibb.co/1GY6...	1 Comment				
	117		Хобіт	Джон. Р. Р. Толкін	«Хобіт» – це історія надв...	https://www.yakaboo.u...	https://i.ibb.co/TZV8...	1 Comment				
	118		Потік	Мірай Чаксентмірай	Щастя – це те, чого праг...	https://www.yakaboo.u...	https://i.ibb.co/3RVt...	1 Comment				
	119		Мистецтво відпочинку	Клаудія Гамонц	У цій книжці авторка асу...	https://www.yakaboo.u...	https://i.ibb.co/gDV3...	0 Comment				
	120		Емоційний інтелект	Деніел Гоулман	Заложність між інтелектом...	https://www.yakaboo.u...	https://i.ibb.co/y4xg...	0 Comment				
	121		Графічний дизайн. Нові ос...	Еллен Латтон, Дженніфер...	У книзі «Графічний дизаї...	https://www.yakaboo.u...	https://i.ibb.co/5203...	0 Comment				
	122		Програмування веб-застосу...	Роман Мельник	У даному підручнику викл...	https://www.yakaboo.u...	https://i.ibb.co/215d...	0 Comment				
	123		Патерни проєктування	Ерік Фрімен, Елізабет Роб...	«Патерни проєктування» –	https://www.yakaboo.u...	https://i.ibb.co/HP36...	0 Comment				
	124		Гордість і упередження	Джейн Остін	З цією книгою можна відп...	https://www.yakaboo.u...	https://i.ibb.co/7qv8...	0 Comment				
	125		Виразний JavaScript	Марейн Хавербеке	«Виразний JavaScript» –	https://bigl.ua/pl066...	https://i.ibb.co/jLc3...	1 Comment				
	126		Крихітні значки	Б. Дж. Фогт	Між тим, чого люди баж...	https://bookclub.ua/c...	https://i.ibb.co/NP75...	0 Comment				

Рис.15. Таблиця «Рекомендація»

Section

×

+

Filters

None

Fields

All

Showing 3 of 3

Add record

☐	id #	name	recommendations
☐	445	Fiction	4 Recommendation
☐	446	Technical	4 Recommendation
☐	447	Self-development	4 Recommendation

Рис.16. Таблиця «Розділ»

Comment							
Filters None Fields All Showing 7 of 7 Add record							
☐	id #	createdAt	content	postedBy {}?	postedByid #?	belongsTo {}?	belongsToId #?
☐	6	2022-06-09T10:30:19.3...	Обожнюю цю книгу, тепе...	User	9	Recommendation	115
☐	18	2022-06-09T19:01:14.0...	корисна книга! 📖	User	9	Recommendation	125
☐	19	2022-06-09T21:38:07.8...	Моя улюблена книга! 📖	User	9	Recommendation	117
☐	24	2022-06-15T01:15:33.2...	Рекомендую до прочитання!	User	9	Recommendation	118
☐	25	2022-06-15T01:16:57.0...	Захоплююча книга, однок...	User	33	Recommendation	116
☐	26	2022-06-15T01:17:51.0...	Моя улюблена книга вже ...	User	32	Recommendation	115
☐	27	2022-06-15T01:21:20.0...	Читав цю книгу ще в шк...	User	34	Recommendation	115

Рис.17. Таблиця «Коментар»

Favorite				
Filters None Fields All Showing 5 of 5 Add record				
☐	id #	addedBy {}?	addedByid #?	addedRecommendation {}?
☐	49	User	33	Recommendation
☐	50	User	9	Recommendation
☐	51	User	9	Recommendation
☐	52	User	9	Recommendation
☐	53	User	36	Recommendation

Рис.18. Таблиця «Обране»

Додаток 4

GraphQL-схема

```

type Query {
  getMyProfile: User!
  getComments: [Comment!]!
  getRecommendations(filter: ID!): [Recommendation!]!
  getSection(filter: ID!): Section!
  getRecommendationDetails(id: ID!): Recommendation!
}

type Mutation {
  getComment(commentId: ID!): Comment!
  postComment(recommendationId: ID!, content: String!):
  Comment!
  deleteComment(commentId: ID!): Comment!
  updateComment(commentId: ID!, content: String!): Comment!
  addToFavorites(recommendationId: ID!): Favorite!
  deleteFromFavorites(favoriteId: ID!): Favorite!
  updateMyProfile(id: ID!, name: String!): User!
  signup(email: String!, password: String!, name: String!):
  AuthPayload
  login(email: String!, password: String!): AuthPayload
}

```

Scalar Date

```

type AuthPayload {
  token: String
  user: User
}

```

```
}
```

```
type User {  
  id: ID!  
  name: String!  
  email: String!  
  comments: [Comment]  
  favorites: [Favorite]  
}
```

```
type Recommendation {  
  id: ID!  
  title: String!  
  author: String!  
  description: String!  
  shopUrl: String!  
  imageUrl: String!  
  comments: [Comment]  
  section: Section  
  favorites: [Favorite]  
}
```

```
type Comment {  
  id: ID!  
  createdAt: Date!  
  content: String!  
  postedBy: User  
  belongsTo: Recommendation  
}
```

```
type Section {  
  id: ID!  
  name: String!  
  recommendations: [Recommendation]  
}
```

```
type Favorite {  
  id: ID!  
  addedBy: User  
  addedRecommendation: Recommendation  
}
```

Додаток 5**Функції-розпізнавачі**

```
async function signup(parent, args, context, info) {
  const userExists = await context.prisma.user.findUnique({
    where: { email: args.email },
  })

  if (userExists) {
    throw new Error('Користувача з такою поштою вже зареєстровано')
  }

  const password = await bcrypt.hash(args.password, 10)
  const user = await context.prisma.user.create({ data: {
    ...args, password } })
  const token = jwt.sign({ userId: user.id }, APP_SECRET)

  return {
    token,
    user,
  }
}

async function login(parent, args, context, info) {
  const user = await context.prisma.user.findUnique({
    where: { email: args.email },
  })

  if (!user) {
    throw new Error('Користувача з такою поштою не знайдено')
  }

  const valid = await bcrypt.compare(args.password,
    user.password)

  if (!valid) {
    throw new Error('Введено неправильний пароль')
  }

  const token = jwt.sign({ userId: user.id }, APP_SECRET)

  return {
    token,
    user,
  }
}
```

```

async function postComment(parent, args, context, info) {
  const { userId } = context

  return await context.prisma.comment.create({
    data: {
      content: args.content,
      createdAt: new Date(),
      postedBy: { connect: { id: userId } },
      belongsTo: { connect: { id:
Number(args.recommendationId) } },
    },
  })
}

```

```

async function updateMyProfile(parent, args, context, info) {
  return await context.prisma.user.update({
    where: {
      id: Number(args.id),
    },
    data: {
      name: args.name,
    },
  })
}

```

```

async function updateComment(parent, args, context, info) {
  return await context.prisma.comment.update({
    where: {
      id: Number(args.commentId),
    },
    data: {
      content: args.content,
    },
  })
}

```

```

async function deleteComment(parent, args, context, info) {
  return await context.prisma.comment.delete({
    where: {
      id: Number(args.commentId),
    },
  })
}

```

```

async function getComment(parent, args, context) {
  return await context.prisma.comment.findUnique({
    where: {
      id: Number(args.commentId),
    },
  })
}

```

```

    },
  })
}

```

```

async function addToFavorites(parent, args, context, info) {
  const { userId } = context

```

```

    return await context.prisma.favorite.create({
      data: {
        addedBy: { connect: { id: userId } },
        addedRecommendation: { connect: { id:
Number(args.recommendationId) } },
      },
    })
  }
}

```

```

async function deleteFromFavorites(parent, args, context,
info) {
  return await context.prisma.favorite.delete({
    where: {
      id: Number(args.favoriteId),
    },
  })
}

```

```

async function getComments(parent, args, context) {
  return await context.prisma.comment.findMany()
}

```

```

async function getSection(parent, args, context) {
  return await context.prisma.section.findUnique({
    where: { id: Number(args.filter) },
  })
}

```

```

async function getRecommendationDetails(parent, args,
context){
  return await context.prisma.recommendation.findUnique({
    where: { id: Number(args.id) },
  })
}

```

```

async function getMyProfile(parent, args, context) {
  const { userId } = context
  return await context.prisma.user.findUnique({
    where: {
      id: userId,
    },
  })
}

```

```
}  
  
async function getRecommendations(parent, { filter },  
context){  
  return await context.prisma.recommendation.findMany({  
    where: {  
      section: { id: filter },  
    },  
  })  
}
```