

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему: РОЗРОБКА ВЕБЗАСТОСУНКУ ДЛЯ ПОШУКУ ТА
РЕКОМЕНДАЦІЇ КНИЖОК
WEB APPICATION DEVELOPMENT FOR BOOK SEARCH AND
RECOMMENDATION

Виконав: студент(ка) 4 курсу, групи КНТ-128
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Сокол.Р.В.

(прізвище та ініціали)

Керівник Касьян М.М.

(прізвище та ініціали)

Рецензент Зеленьова І. Я

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 121 Інженерія програмного забезпечення
(КОД І НАЙМЕНУВАННЯ)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
“ ” 2022 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Сокола Романа Володимировича

(прізвище, ім'я, по батькові)"

1 Тема проєкту (роботи) Розробка вебзастосунку для пошуку та рекомендації книжок Web Application Development for Book Search and Recommendation

керівник проєкту (роботи) Касьян Микола Миколайович, к.т.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “27” квітня 2022 року № 102

2. Строк подання студентом проєкту (роботи) 13 червня 2022 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області та постановка завдання. 2. Опис програми. 3. Розробка програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Касьян М.М., доцент		
Нормоконтроль	Дейнега Л.Ю., ст. викладач		

7. Дата видачі завдання “28” лютого 2022 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2 – 3 тижні	Розділ 1
3	Розробка архітектури програми	4 тиждень	Розділ 2
4	Розробка програми	5 – 6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	7 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Додатки
7	Захист роботи	9 тиждень	

Студент(ка)

(підпис) Сокол Р.В.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Касьян М.М.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 96 с., 47 рис., 3 табл., 3 дод., 19 джерел.

ВЕБЗАСТОСУНОК, JAVASCRIPT, HTML, CSS, ПОШУК, КНИГИ.

Об'єкт дослідження – методи пошуку та рекомендації книг.

Предмет дослідження – вебзастосунок для пошуку та рекомендації книг.

Мета роботи – розробити вебзастосунок для пошуку та рекомендації книг.

Методи дослідження – середовище розробки Microsoft Visual Studio 2017, JavaScript, HTML, CSS, Google Books API, браузер для перевірки застосунку, система емулювання серверу MAMP, персональний комп'ютер з процесором Intel core i5, RAM: 32ГБ, жорстким диском на 512 ГБ та системою Windows 10.

Результати. Був розроблений вебзастосунок, що дозволяє шукати книги, отримувати інформацію про них та частково читати книг, а також отримувати схожі книги на ті, що цікавлять користувача.

Висновки. Розроблено програмну систему для пошуку та рекомендації книг, яка прискорює процес пошуку книг.

Галузь використання – книжкові магазини, бібліотеки, наукові репозиторії, повсякденне життя.

ABSTRACT

Explanatory note to the bachelor's thesis: 96 pages, 47 figures, 3 tables, 3 appendices, 19 sources.

WEB APPLICATION, JAVASCRIPT, HTML, CSS, SEARCH, BOOKS.

The object of research is methods of searching and recommending books.

The subject of the research is a web application for searching and recommending books.

The purpose of the work is to develop a web application for searching and recommending books.

Research methods – Microsoft Visual Studio 2017 development environment, JavaScript, HTML, CSS, Google Books API, application validation browsers, MAMP server emulation system, personal computer with Intel core i5 processor, 32GB RAM, 512GB hard drive and Windows 10.

Results. A web application has been developed that allows you to search for books, get information about them and partially read books, as well as get books similar to those of interest to the user.

Conclusions. A software system for finding and recommending books has been developed, which speeds up the process of finding books.

Scope – for any user in everyday life who has access to the Internet and the browser.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Опис предметної області	10
1.2 Аналіз аналогів	10
2 Опис програми.....	18
2.1 Огляд особливостей обраного середовища розробки	18
2.2 Вибір та обґрунтування програмних засобів реалізації вебзастосунку	19
2.3 Розробка архітектури програми.....	26
2.3.1 Опис потреб до застосунку	26
2.3.2 Робота користувача із застосунком.....	27
3 Розробка програми	32
3.1 Вимоги до інтерфейсу користувача	32
3.2 Розробка інтерфейсу користувача	36
3.2.1 Опис графічного інтерфейсу користувача.....	36
3.2.2 Структура та реалізація інтерфейсу	40
3.3 Реалізація функціоналу застосунку.....	45
3.3.1 Робота з даними та реалізація алгоритмів.....	45
3.3.2 Реалізація попереднього показу книги	51
3.3.3 Вхідні та вихідні дані.....	52
4 Керівництво програміста.....	54
4.1 Призначення та умови застосування програми	54
4.2 Характеристики програми.....	55
4.3 Звертання до програми	56
5 Керівництво оператора	57
5.1 Призначення та умови застосування програми	57

5.2 Умови виконання програми	57
5.3 Виконання програми.....	58
5.4 Повідомлення оператору	59
6 Тестування і результати.....	61
6.1 Опис методики проведення тестування.....	61
6.2 Результати використання розробленого програмного забезпечення	61
Висновки	64
Перелік джерел посилання	66
Додаток А Технічне завдання.....	68
Додаток Б Текст програми.....	72
Додаток В Слайди презентації.....	90

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API – Application Programming Interface;
CSS – Cascading Style Sheets;
HTML – HyperText Markup Language;
HTTP – HyperText Transfer Protocol;
HTTPS – HyperText Transfer Protocol Secure;
ISBN – International Standard Book Number;
JS – JavaScript;
PDF – Portable Data File;
SDK – Software Development Kit;
SEO – Search Engine Optimization;
SQL – Structured Query Language;
SVG – Scalable Vector Graphics;
XHTML – Extensible hypertext markup language;
XUL – XML User Interface Language;
БД – база даних;
ОС – операційна система;
ПЕОМ – персональна електронно-обчислювальна машина;
ПЗ – програмний засіб.

ВСТУП

Метою роботи є створення вебзастосунку який шляхом взаємодії з користувачем буде відповідати на його пошукові запити щодо книг, рекомендувати схожі книги та дозволить переглядати інформацію про них.

Застосунок повинен забезпечувати коректний та зручний пошук і перегляд книг для користувача у сучасних браузерях та надавати користувачу необхідну йому інформацію.

Було виконано огляд HTML, CSS та JavaScript для створення вебзастосунків.

Актуальність роботи полягає у створенні вебзастосунку для простого та швидкого пошуку книг, що цікавлять користувача або книг, що можуть зацікавити користувача на основі його пошукового запиту до застосунку.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області

Предметна область пошуку книг представлена багатьма ресурсами, в основному – сайтами або вебсервісами, що дозволяють користувачу продивлятися книги за жанрами, авторами, оцінками, популярністю та надають доступ до вебверсії або pdf-версії книги. Найчастіше ці книги є піратськими версіями та надаються користувачам у вільному доступі. Окрім сайтів, що розповсюджують книги є також магазини книг, що надають користувачу інформацію про книгу та можливість придбати її у фізичному або цифровому форматі з можливістю ознайомитися з коротким описом книги та відгуками про неї.

Об'єктом дослідження є запит користувача на отримання інформації про книгу, що його цікавить або книгу з якоїсь області літератури для ознайомлення з нею та можливого подальшого придбання.

Метою роботи є розробка вебзастосунку, що дозволить користувачу шукати книги за конкретними назвами, авторами, жанрами, рекомендувати аналоги для книг, якими цікавиться користувач, надавати загальну інформацію про книгу для можливої подальшої покупки.

1.2 Аналіз аналогів

Розглядаючи пошук книг, одним з базових інструментів є «Розширений пошук книг» у пошуковій системі Google, що дозволяє шукати книги, але за має дуже специфічні критерії пошуку, такі як: розподілення на журнали, газети, книги, пошук за ISBN та не зручний для кінцевого користувача інтерфейс (рис. 1.1).

Google книги Розширений пошук книг Пошук Google

Знайти результати

з усіма словами
що містять точну фразу
хоча б з одним зі слів
що не містять слів

Пошук:

10 результатів

Тип видання: ☒ Усі книги ☐ Обмежений попередній перегляд і повний перегляд ☐ Тільки повний перегляд ☐ Лише електронні книги Google

Мова: ☒ Будь-які видання ☐ Книжки ☐ Журнали ☐ Газети

Показувати сторінки такою мовою:

Назва: Відобразити книги з назвою

Автор: Відобразити книги, написані

Видавець: Відобразити книги, видані

Дата публікації: ☒ Показати вміст, надрукований будь-коли ☐ Показати вміст, надрукований у період між або

ISBN: Показати книги з номером ISBN

ISSN: Показати журнали з номером ISSN

Рисунок 1.1 – Розширений пошук книг Google

Сервіс є лише розширенням можливостей щодо пошуку у пошуковій системі Google та після введення результатів надає користувачу вебсторінку знайдених книг для попереднього перегляду без можливостей пошуку, наприклад, за жанрами (рис. 1.2) [1].

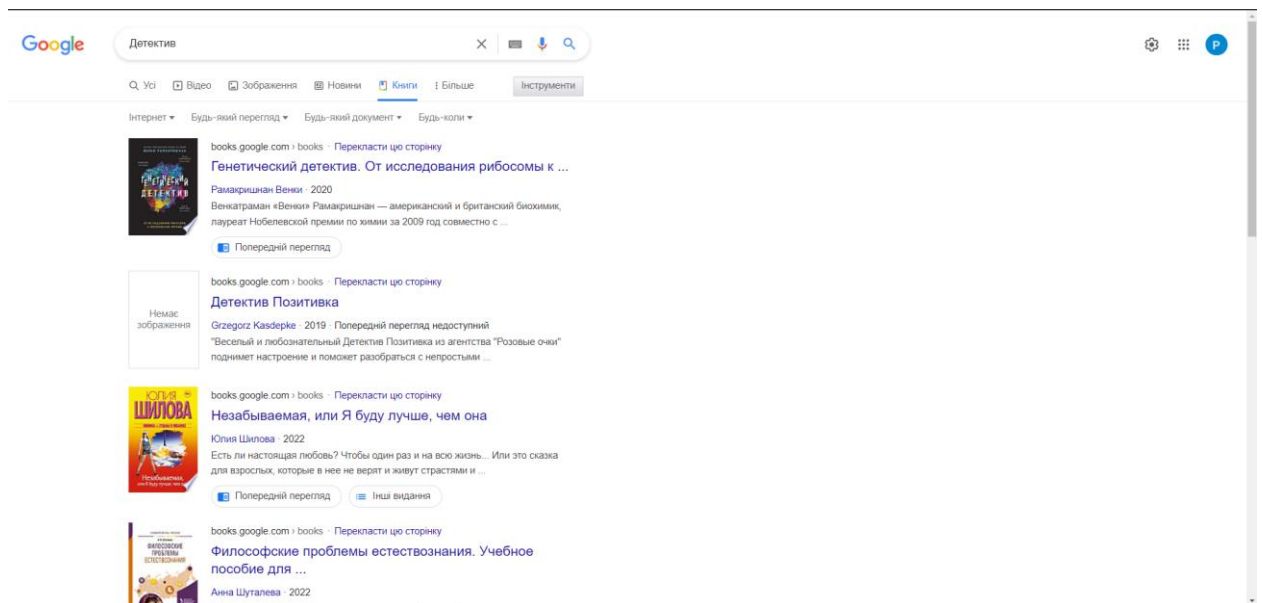


Рисунок 1.2 – Результат пошуку

Ще одним аналогом є сайт bookfinder.com.ua, що є ресурсом для пошуку книг по інтернет-магазинам країни та дозволяє обрати книгу та магазин у якому можливо на даний момент купити бажану книгу (рис. 1.3 – 1.4).

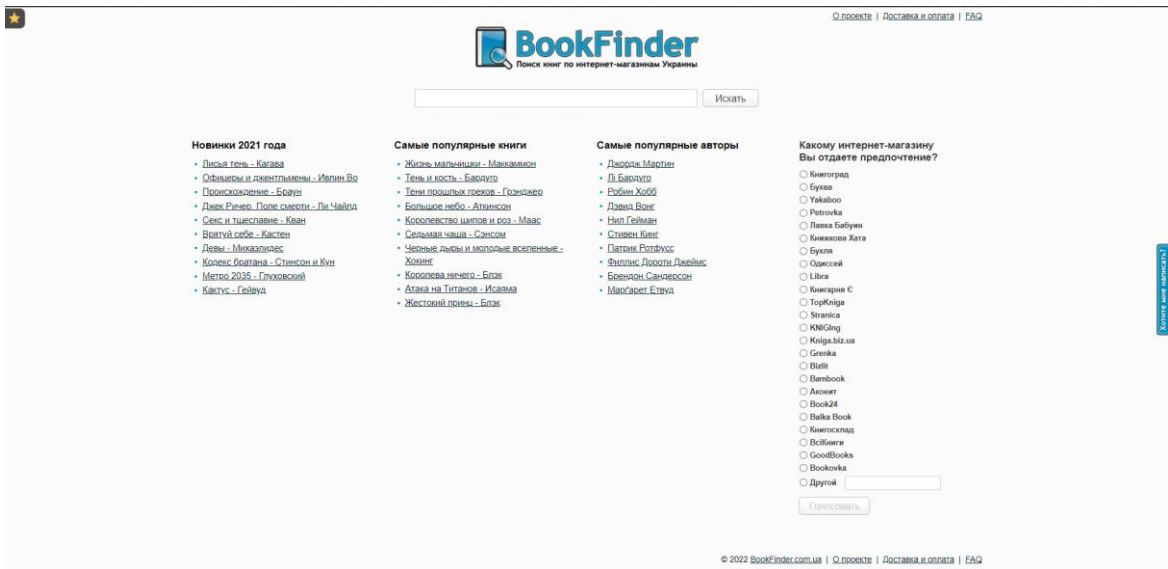


Рисунок 1.3 – Пошуковий екран bookfinder.com.ua

Сервіс надає можливість шукати книгу за фразою у пошуковому рядку, обрати магазин у якому шукати книгу або обирати з популярних категорій, таких як: новинки 2021 року, найпопулярніші книги та найпопулярніші автори, що розширює пошукові можливості користувачів. Також сервіс надає можливість сортувати книги та обирати лише ті книги, що є у наявності (рис. 1.4) [2].

Але інтерфейс все ще застарілий та не повністю дозволяє користувачу більш зручно шукати книги, що його цікавлять.

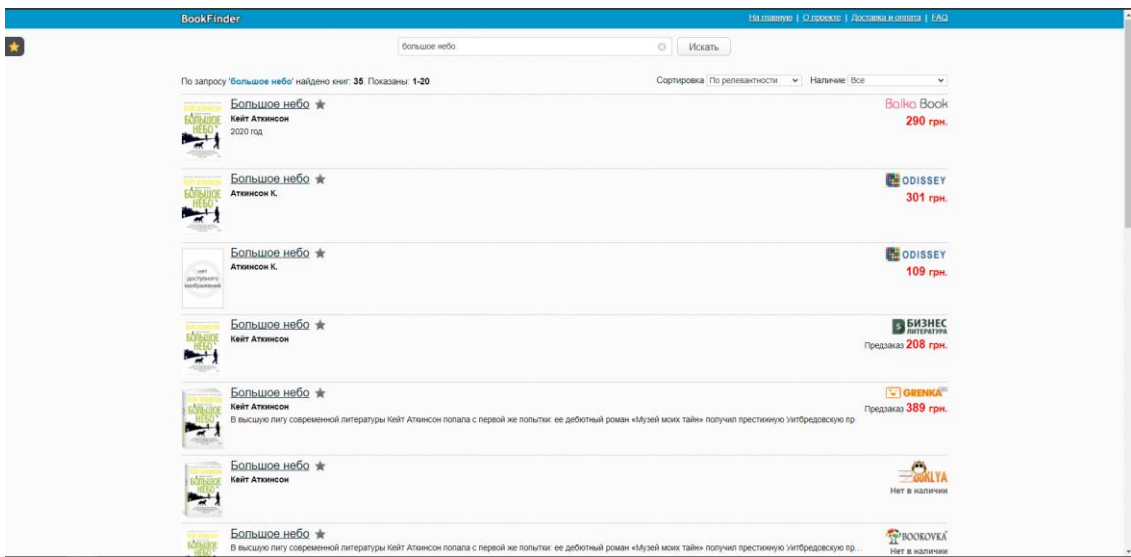


Рисунок 1.4 – Результат пошуку bookfinder.com.ua

Результатом пошуку є список книг у інтернет магазинах та їх ціни, відповідно, з можливістю перейти на сайт магазину для подальшого знайомства з книгою (рис. 1.5).

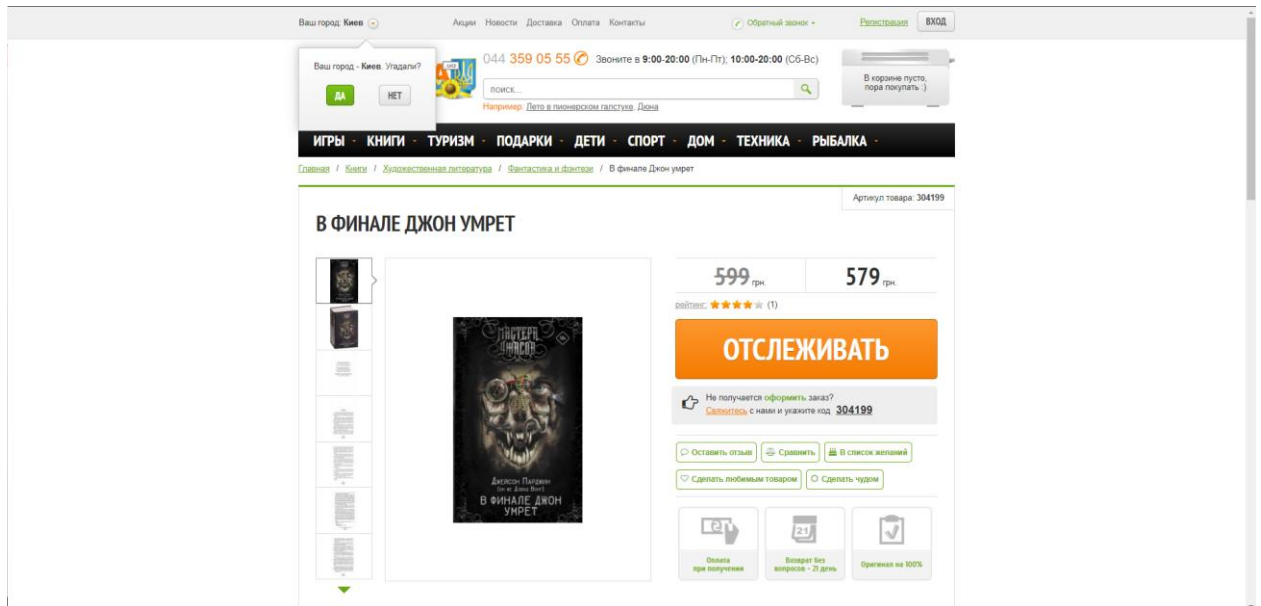


Рисунок 1.5 – Книга на сторінці одного з інтернет-магазинів

У порівнянні з Google пошуком bookfinder.com.ua має більш зручний інтерфейс користувача та надає можливість шукати книги, що цікавлять користувача, у конкретних магазинах для подальшої покупки на сайті обраного магазину, але він більше спрямований на пошук конкретної, книги, що цікавить користувача і не дуже допомагає у виборі книги, якщо користувач не повністю знає, що саме він шукає для покупки.

Також існують аналоги застосунків, такі як Goodreads, Bookmojis, Libby та інші мобільні застосунки, що виконують аналогічні цілі, але є доступні лише на пристроях з операційними системами Android та IOS (рис. 1.6-1.7).

Вони мають більш розширений функціонал ніж Google пошук книг, але дозволяють також додавати книги до списку бажаних до читання (рис. 1.6). та вже поділяють книги на категорії і мають рекомендації (рис. 1.7).

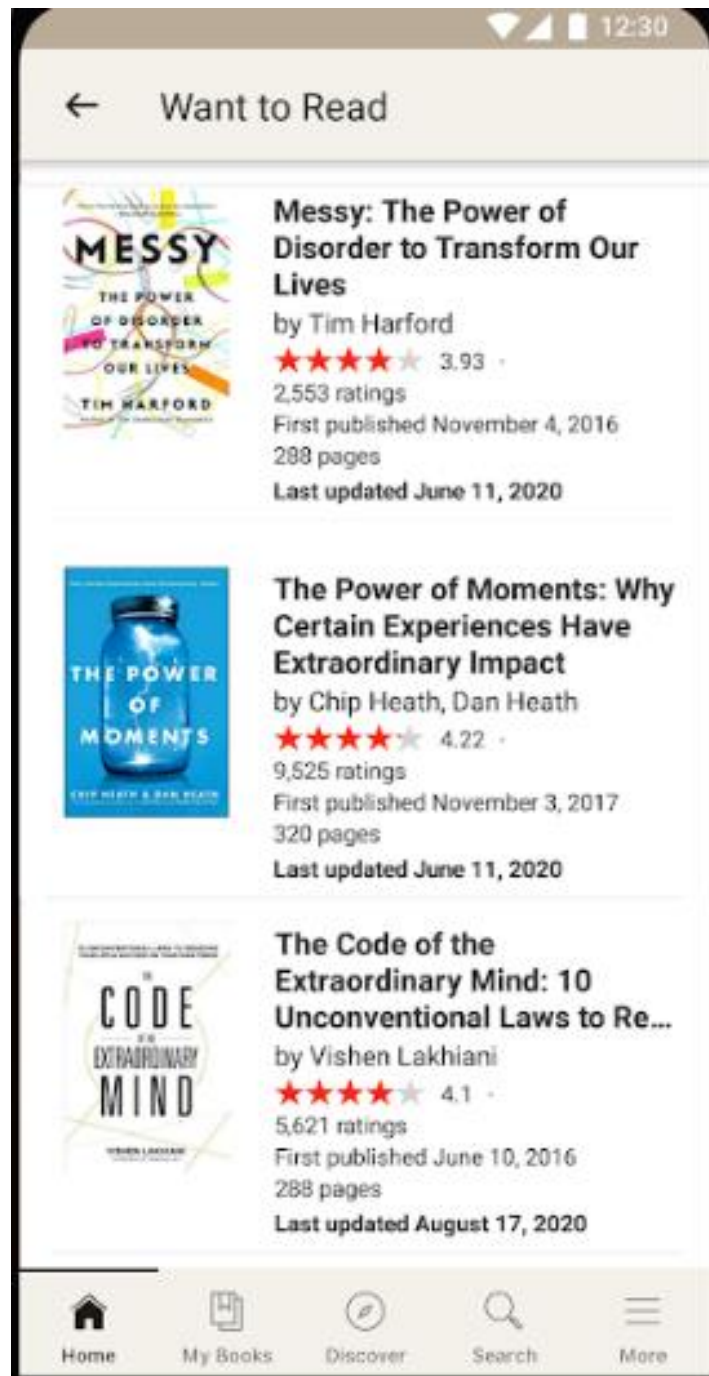


Рисунок 1.6 – Goodreads книги бажані до читання

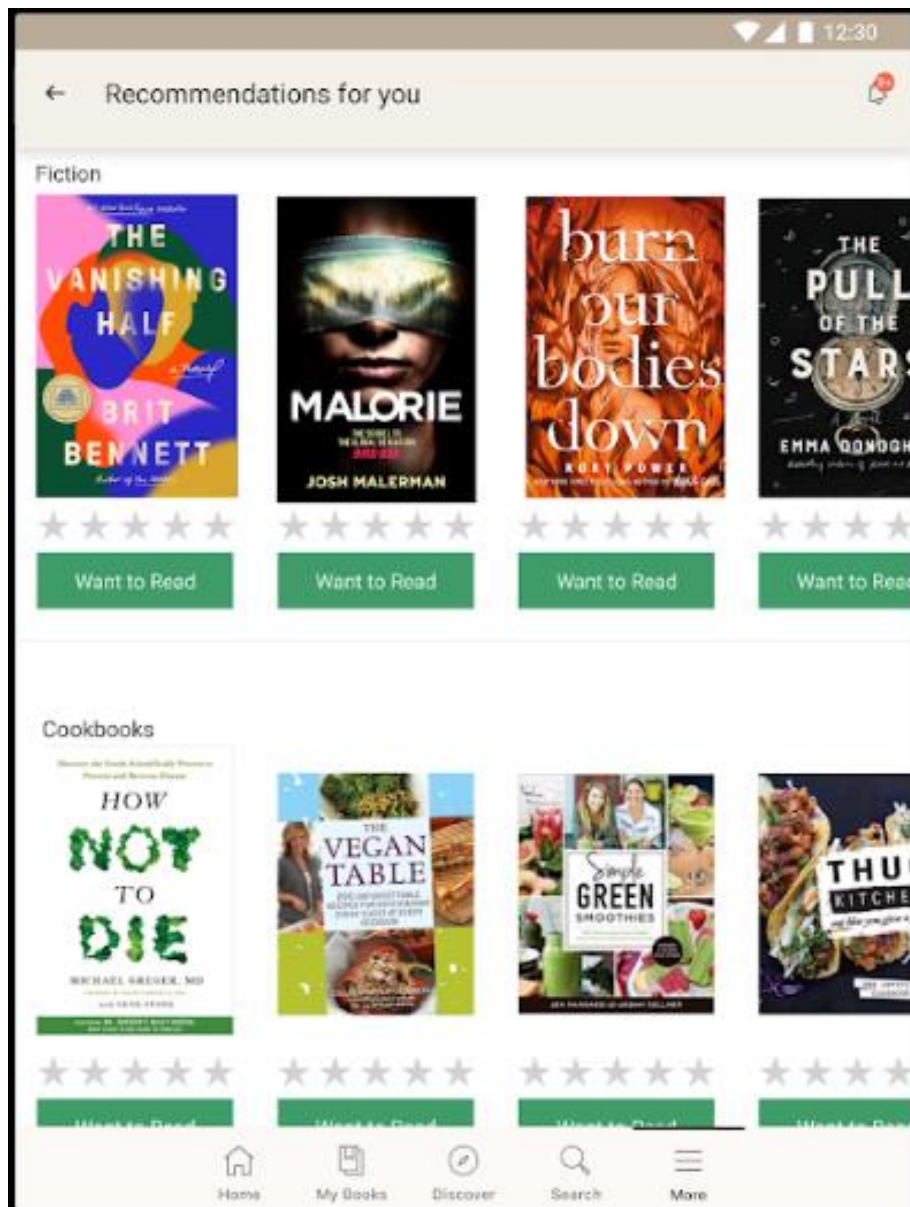


Рисунок 1.7 – Goodreads книги за жанрами у рекомендаціях користувача

Також, за допомогою цих застосунків користувач може:

- переглядати цифровий каталог книг;
- брати в оренду електронні книги, аудіокниги та журнали;
- завантажувати книги для офлайн-читання;
- слухати аудіокниги через Android Auto;
- автоматично синхронізувати положення читання на всіх пристроях, що мають встановлений застосунок.

А тому ці застосунки правильніше буде віднести не до застосунків пошуку книг, а до повноцінних бібліотек з розширеними можливостями, що доступні при покупці додаткових функції та без них є достатньо обмеженими у функціоналі, а також обмежені операційною та технічною платформами, тобто, є доступними лише на мобільних пристроях під керуванням операційних систем Android або IOS [3].

Дивлячись на те, що кінцевим результатом має бути розробка вебзастосунку для пошуку та рекомендації книг, найбільш наближеним до цього є сервіс bookfinder.com.ua, що має недоліки у вигляді застарілого інтерфейсу та недостатніх можливостей з пошуку і рекомендацій. А саме, застарілість інтерфейсу полягає у дизайні сайту у стилі початку 2000-их років, не зручною для користувача головною сторінкою з пропозиціями для користувача, що не привертають до себе уваги, бо є лише текстом, що переводить на відповідні посилання, ніяк не виділяється та не заохочує користувача звертати на нього увагу (рис. 1.8), в іншому ж дизайн сайту – це просто білий фон на якому розташовані назви книг, що цікавлять користувача, з дуже яскравими логотипами магазинів та яскраво-червоними цінниками на книги, що не є гарним рішенням через м'які кольори іншої частини інтерфейсу та не сприяють комфортній роботі користувача із сайтом (рис. 1.9).

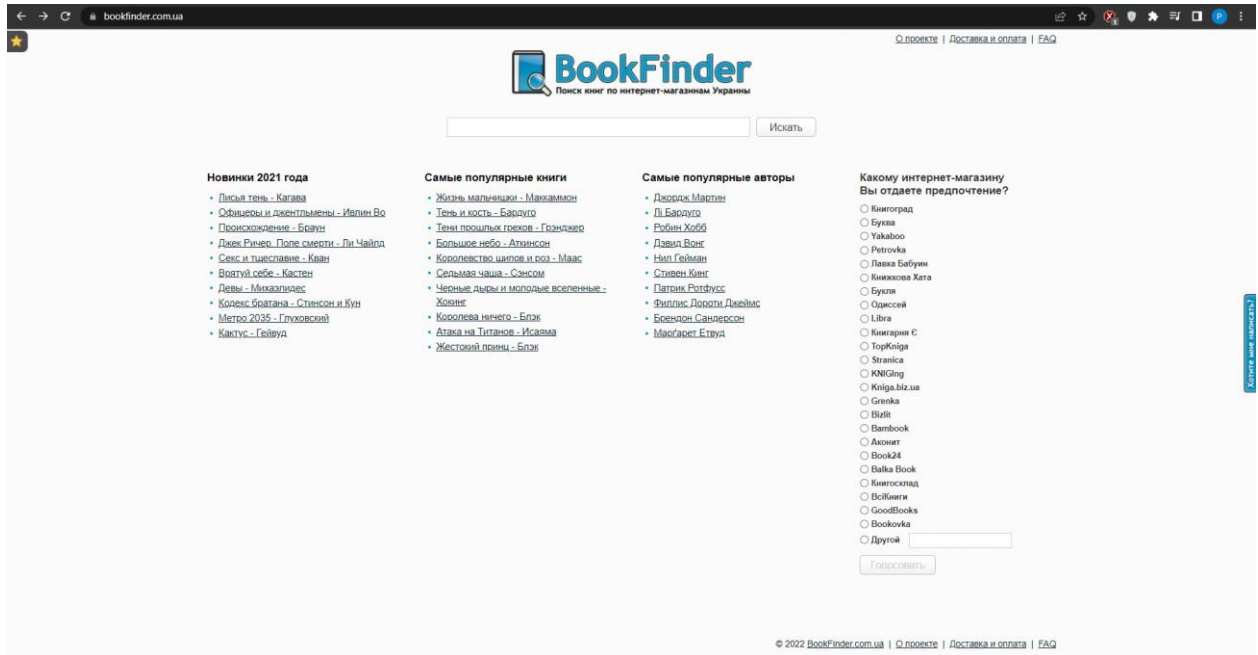


Рисунок 1.8 – Головна сторінка сайту bookfinder.com.ua

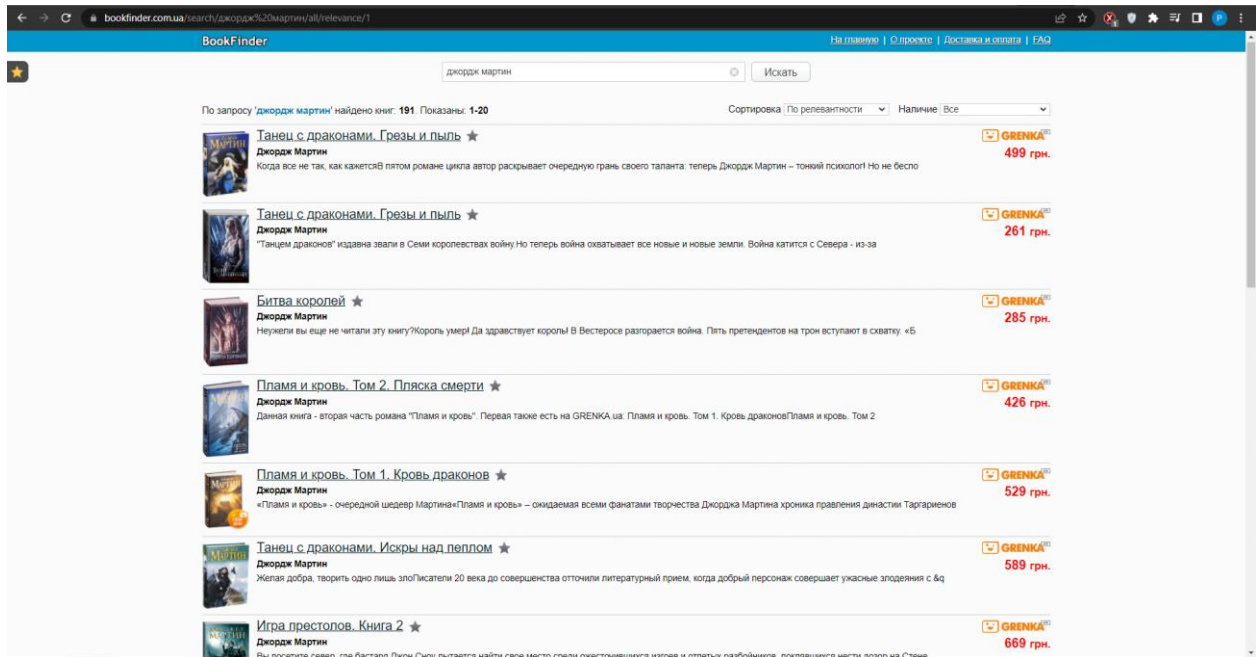


Рисунок 1.9 – Сторінка пошукового запиту користувача на сайті bookfinder.com.ua

Відповідно, використовуючи за основу ідею пошуку та рекомендації книг користувачу для подальшого ознайомлення з ними має бути розроблений вебзастосунок, що буде надавати безкоштовний функціонал користувачу з пошуку та рекомендацій книг для подальшого ознайомлення з ними.

2 ОПИС ПРОГРАМИ

2.1 Огляд особливостей обраного середовища розробки

Основними критеріями для вибору середовища розробки була підтримки мови JS та наявність засобів для веброзробки, а також досвід користування середовищем розробки.

За результатами порівняльної таблиці 2.1 середовищ розробки було обрано Microsoft Visual Studio.

Таблиця 2.1 – Порівняння середовищ розробки

Критерій	Webstorm	Atom	Microsoft Visual Studio
Багатоплатформність	+	+	+
Підтримка репозиторіїв	+	+	+
Автоматичне доповнення коду	+	+	+
Інтегрований відладчик	+	+	+
Плагіни, що надаються середовищем	+	+	++
Безкоштовна версія	+	+	+

Microsoft Visual Studio — це потужна IDE, яка дуже ресурсомістка і в основному використовується для серйозної розробки програмного забезпечення на C, C++, .NET і C#. Вона має чудові функції, як-от одночасне планування та редагування. А також визначає найкращий у галузі Code Intel і має вбудовані відладники. Іншими словами, це належна IDE галузевого рівня [4].

Microsoft Visual Studio було обраний через те, що вже є досвід користування цією IDE, а також статистика показує, що при розробці на JS Visual Studio є найбільш використовуваним IDE.

2.2 Вибір та обґрунтування програмних засобів реалізації вебзастосунку

Основними засобами, що були використані для розробки вебзастосунку є: JavaScript, HTML та CSS.

Основою для верстки сторінок використовувався HTML (мова гіпертекстової розмітки) та CSS, що представляє собою формальну мову для опису зовнішнього вигляду вебсторінку, що написаний за допомогою мови розмітки, тобто HTML або XHTML.

Через те, що застосунок має бути вебзастосунком, базисом для верстки його сторінок використовувався HTML, що є найбільш поширеним способом верстки вебсторінок. Він є стандартизованою мовою для розмітки документів, що будуть продивлятися у вигляді вебсторінки у браузері.

Браузери отримують від сервера HTML документ за протоколами HTTP/HTTPS або відкривають з локального диска та далі інтерпретують код в інтерфейс, який відображатиметься у браузері [5].

Елементи HTML використовуються для будувannya блоками HTML сторінок. За допомогою HTML різні конструкції, зображення та інші об'єкти, такі як GIF, інтерактивна вебформа, можуть бути вбудовані в сторінку, що відображається. За допомогою HTML можливо створювати заголовки, списки, посилання, абзаци, цитати та інші елементи. Елементи HTML виділяються тегами, записаними з використанням кутових дужок. Такі теги, як `<img/>` та `<input/>`, безпосередньо вводять контент на сторінку. Інші теги, такі як `<p>`, оформлюють текст у собі і можуть включати інші теги як вкладені елементи. Браузери не відображають користувачу HTML-теги й використовують їх для

інтерпретації вмісту сторінки, що буде у кінцевому результаті показана користувачу (рис. 2.1).

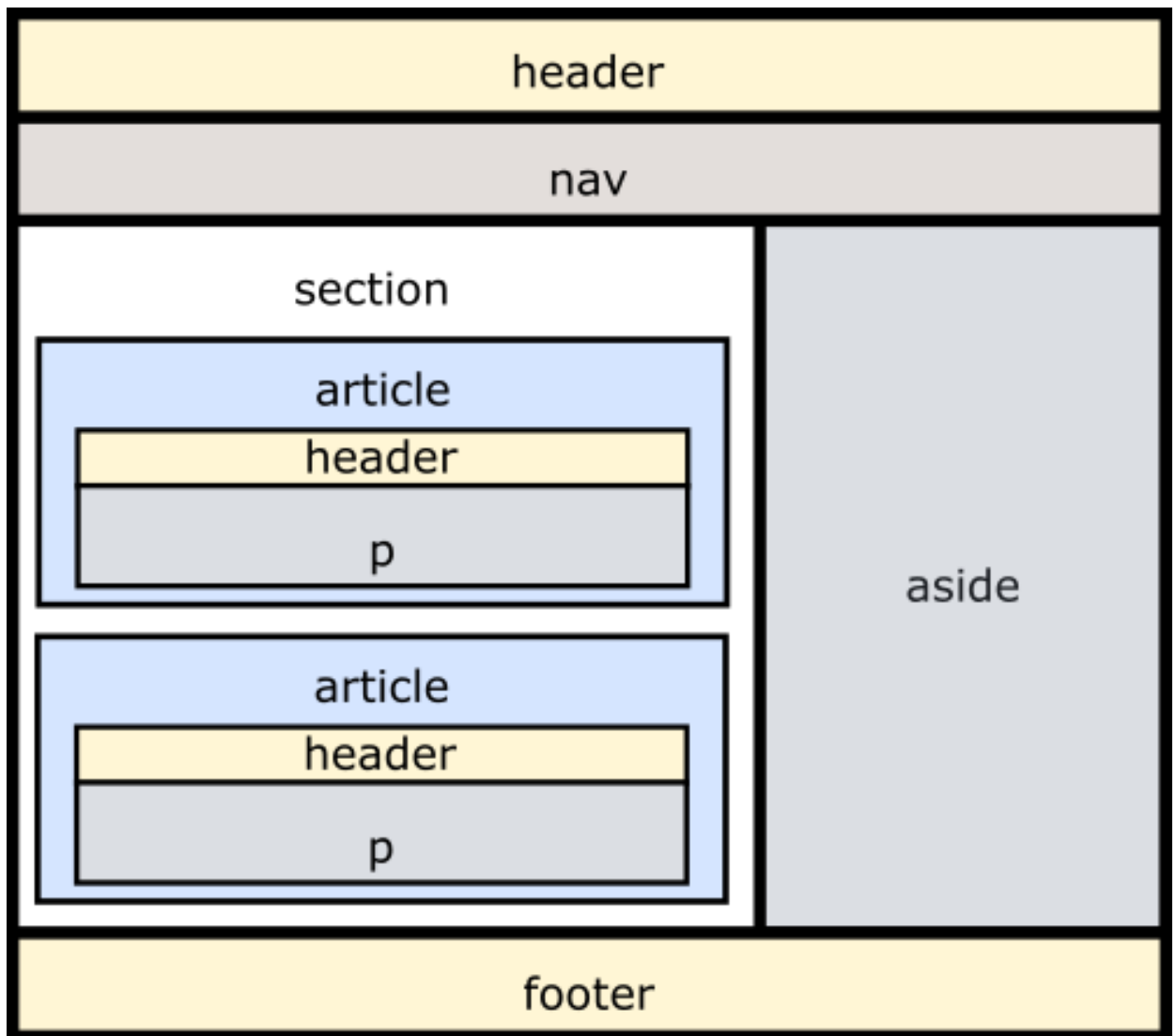


Рисунок 2.1 – Умовна структура верстки вебсторінки

У HTML можна вбудувати програмний код мовою програмування JavaScript, керувати поведінкою та змістом вебсторінок [5].

Сам по собі HTML не має усіх необхідних можливостей для створення сучасного дизайну вебсторінок і тому використовується часто лише для побудови структури вебсторінок, а задача створення візуального наповнення вебсторінки покладається на CSS (рис. 2.2).

Основною метою розробки CSS є відокремлення опису логічної структури вебсторінки від опису зовнішнього вигляду цієї вебсторінки (яке розробляється за допомогою формальної мови CSS) (рис. 2.4). Такий підхід може збільшити доступність документа, надати більшу гнучкість та можливість управління його поданням, а також зменшити складність та повторюваність у структурному вмісті, що також облегшує роботу з документом не тільки його розробнику, а й підтримку його діяльності у майбутньому іншими розробниками, що будуть звертатися до документа.

Також CSS дозволяє вебсторінці мати декілька дизайнів сторінки, що будуть варіюватися відповідно до пристроїв на яких відкривається вебсторінки. Ще CSS пришвидшує завантаження сторінок шляхом перенесення правил представлення даних до окремого файлу CSS. У цьому випадку браузер завантажує лише структуру документа та дані, які зберігаються на сторінці, а представлення цих даних завантажується браузером лише один раз і може бути закешований. Ще одним плюсом є простота подальшої зміни дизайну, тому що не потрібно редагувати кожену сторінку, а достатньо лише змінити CSS-файл. Додаткові можливості також є плюсом CSS, наприклад, за допомогою CSS-верстки можна зробити блок тексту, який решта тексту буде обтікати (наприклад для меню) або зробити так, щоб меню було завжди видно при прокручуванні сторінки [6].

```
CSS
p { color: red;}

HTML

<!DOCTYPE html>
<html>
  <head>
    <title>Привіт, світ</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <p>Цей абзац буде червоним.</p>
  </body>
</html>
```

Рисунок 2.4 – Відокремлення CSS від HTML та застосування селектору

Але за допомогою HTML та CSS неможливо створювати інтерактивні вебсторінки, а тому для реалізації інтерактивності сторінок використовувався JavaScript, що є мультипарадигмальною мовою програмування, що часто використовується у браузерах, як мова скриптів для надання інтерактивності вебсторінкам.

JavaScript – це повноцінна мова програмування, яка пожвавлює сторінки вебресурсу. З його допомогою можна зробити сайт функціональним та інтерактивним, навчити його реагувати на дії користувача, а також інші браузерні події. За допомогою JavaScript можливо обробляти різні події: завантаження сторінки, натискання кнопки, клік мишкою. JS реагує на ці та інші події, запускаючи конкретні скрипти, прописані програмістом. Таким чином, наприклад, відбувається зміна картинки у фотогалереї, закриття модального вікна, підтвердження входу в акаунт та інше (рис. 2.5). Обробка подій – основні, але не всі завдання JavaScript. З його допомогою можна змінювати вигляд сторінки, додаючи та видаляти елементи, переміщувати, перефарбовувати тощо. Також за допомогою JS здійснюється запит даних із сервера, відправлення, приймання та відображення інформації з віддалених баз [7].

```

var video = document.querySelector("#videoElement");

navigator.getUserMedia = navigator.getUserMedia || navigator.webkitGetUserMedia || navigator.
mozGetUserMedia || navigator.msGetUserMedia || navigator.oGetUserMedia;

if (navigator.getUserMedia) {
    navigator.getUserMedia({video: true}, handleVideo, videoError);
}

function handleVideo(stream) {
    video.src = window.URL.createObjectURL(stream);
}

function videoError(e) {
    // do something
}

var video = document.getElementById('videoElement');
var canvas = document.getElementById('canvas');

function snapshot()
{
    canvas.width = video.videoWidth;
    canvas.height = video.videoHeight;
    canvas.getContext('2d').drawImage(video, 0, 0);
    var img = canvas.toDataURL('image/png');
    img = img.replace(/^data:image\/\w+;base64/, "");
    var pic = document.getElementById('gambar');
    pic.value = img;
}

```

Рисунок 2.5. – Приклад функції для роботи з відео на JavaScript

Ще цікавішою є функціональність, створена поверх основної мови JavaScript. Так звані інтерфейси прикладного програмування (API), що надають додаткові можливості для використання коду JavaScript.

API – це готові набори блоків коду, які дозволяють розробнику реалізовувати програми, які інакше було б важко або неможливо реалізувати (рис. 2.6) [8].

Code Snippet NodeJS ▾

```

unirest.get("https://community-open-weather-map.p.rapidapi.com/weather?callback=test&id=2:
.header("X-RapidAPI-Host", "community-open-weather-map.p.rapidapi.com")
.header("X-RapidAPI-Key", "SIGN-UP-FOR-KEY")
.end(function(result) {
    console.log(result.status, result.headers, result.body);
});

```

Рисунок 2.6 – Приклад використання API у JavaScript для отримання інформації про погоду

За допомогою API набагато простіше брати готові рішення і комбінувати їх разом, щоб розробити застосунок, ніж самому розробляти дизайн, та функціонал, а потім зібрати їх разом.

API-інтерфейси браузера вбудовані у нього і можуть відображати дані з навколишнього комп'ютерного оточення або робити корисні складні речі [8]. Наприклад:

- API-інтерфейс DOM (Document Object Model) дозволяє маніпулювати HTML і CSS, створювати, видаляти та змінювати HTML, динамічно застосовувати нові стилі до сторінки тощо;
- API геолокації отримує географічну інформацію. Так Google Maps може знайти ваше місцеперебування та нанести його на карту;
- API Canvas та WebGL дозволяють створювати анімовані 2D та 3D-графіки. Люди роблять деякі дивовижні речі, використовуючи ці вебтехнології;
- аудіо та відео API, такі як HTMLMediaElement та WebRTC, дозволяють робити дійсно цікаві речі з мультимедіа, такі як програвання аудіо та відео прямо на вебсторінці, або захоплювати відео з вебкамери та відображати його на іншому пристрою.

У випадку із застосунком для пошуку та рекомендації книг буде використано API Google Books, використовуючи який програма може виконувати пошук і отримувати інформацію про книги, можливість перегляду та доступність електронних книг, а також надає можливість керувати своїми особистими книжковими полицями [9].

API Embedded Viewer дозволяє вставляти попередні перегляди Google Books на вебсторінку і програмно керувати цими попередніми переглядами.

2.3 Розробка архітектури програми

2.3.1 Опис потреб до застосунку

Виходячи із задач, які має виконувати вебзастосунок, має бути розроблений вебзастосунок, який виконує функції щодо пошуку та рекомендації книжок користувачу, відповідно до його пошукових запитів, надає можливості попереднього ознайомлення з бажаною книгою, яку він отримає, як відповідь, від застосунку на пошуковий запит користувача. Попереднє ознайомлення має утримувати у собі описову інформацію про книгу (назву, автора, жанр та короткий опис книги). Також користувач повинен мати можливість ознайомитись з безплатним фрагментом книги у *.pdf форматі, що буде розміщений на тій самій сторінці де знаходиться стиснута інформація про книгу.

Користувач повинен мати можливості пошуку, як за назвою, так і за автором або жанром книг, що його цікавлять.

Застосунок не буде реєструвати користувача у системі, щоб не створювати зайві незручності з необхідності реєструватися або заходити у застосунок заради можливості пошуку книг, а тому рекомендації книг будуть засновані на поточному пошуковому запиті користувача або пошукових запитах поточної сесії користувача у застосунку.

Відповіддю на пошуковий запит користувача у системі буде уявляти собою множину книг, що прямо або опосередковано задовольняють пошуковий запит користувача та до цього пошукового запиту також будуть додані книги, що є схожими (за жанром або автором) на ті, які користувач отримує у пошуковому запиті.

З кожною книгою зі списку користувач зможе ознайомитися або змінити свій пошуковий запит та продовжити роботу із застосунком.

Також застосунок має забезпечувати можливість зручної роботи з ним, як у денний, так і у вечірній час дня. Під цим розуміються не агресивні кольори

дизайну інтерфейсу, без занадто яскравих чи квітчастих кольорів, що можуть заважати зручній роботі із застосунком.

Ще одним критерієм до застосунку має бути сумісність з сучасними браузерами та можливість користуватися ним у будь-якому з них (Mozilla Firefox, Google Chrome, Vivaldi, Safari, Opera та інші), що забезпечить доступ до застосунку можливим та комфортним для користувачів будь-яких браузерів.

2.3.2 Робота користувача із застосунком

Робота користувача із застосунком у вигляді діаграми прецедентів представлена на рисунках 2.7 – 2.8.

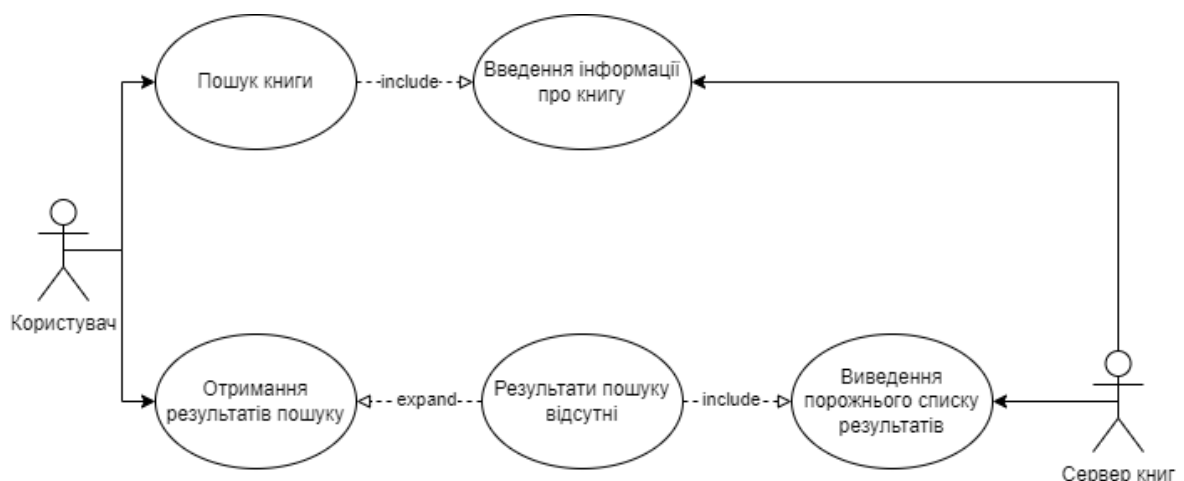


Рисунок 2.7 – Діаграма прецеденту «Пошук книги»

Потік подій для розробленої діаграми прецеденту «Пошук книги»:

Передумова: доступ до мережі інтернет, доступ до Google Books.

Етапи основного потоку подій:

а) пошук книги:

1) введення інформації про книгу;

б) отримання результатів пошуку, у разі відсутності – а1. результати пошуку відсутні.

А1. Результати пошуку відсутні:

- а) користувачу виводиться порожній список результатів;
- б) потік повертається до пункту 1 основного потоку.

Пост умова: знайдений список книг, що задовольняють пошуковому запиту користувача.

Цей потік дій користувача описує прецедент «Пошук книги» у системі та ілюструє як буде виглядати процес пошуку книги користувачем у застосунку.

У результаті завершення цього прецеденту користувач отримає список книг, що задовольняють його пошуковому запиту або ж отримає порожній список, що означатиме, що не знайдено жодної книги, що відповідає пошуковим критеріям користувача.

Ще одним прецедентом є прецедент «Ознайомлення з книгою» (рис. 2.8) на якому зображується, як користувач буде взаємодіяти з книгою, що його зацікавила.

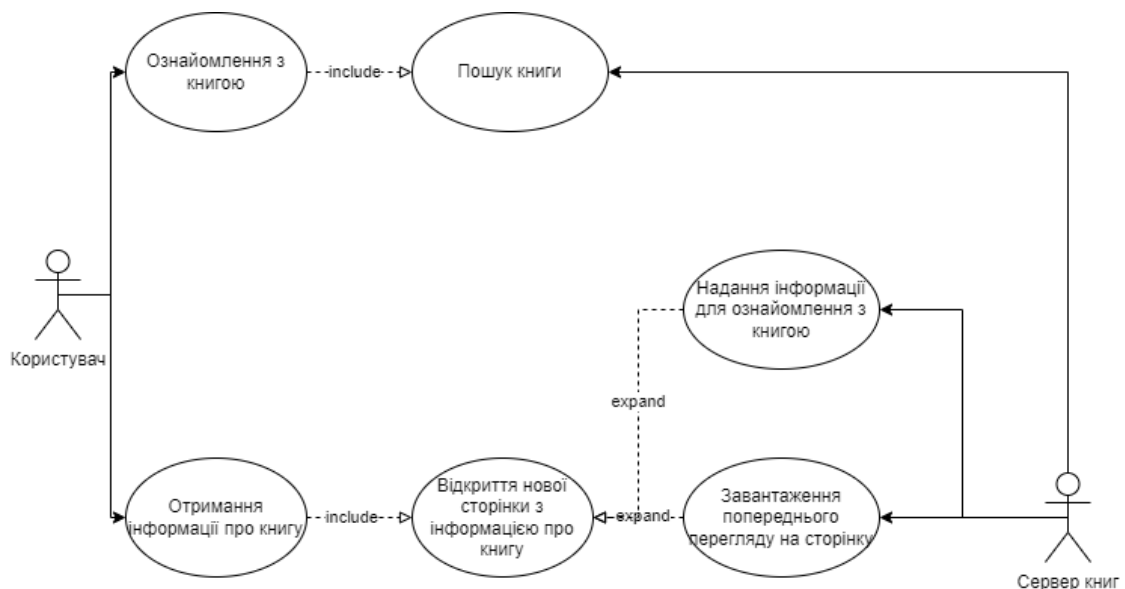


Рисунок 2.8 – Діаграма прецеденту «Ознайомлення з книгою»

Потік подій для розробленої діаграми прецеденту «Ознайомлення з книгою»:

Передумова: доступ до мережі інтернет, доступ до Google Books, не порожній результат пошукового запиту.

Етапи основного потоку подій:

а) ознайомлення з книгою:

1) прецедент пошук книги;

б) отримання інформації про книгу:

2) відкриття сторінки з інформацією про книгу:

– надання стислої інформації для ознайомлення з книгою;

– завантаження попереднього перегляду книги на сторінку.

Пост умова: відкрита сторінка зі стислою інформацією про книгу та попереднім переглядом.

Цей прецедент демонструє, як користувач, що отримав відповідь на свій пошуковий запит буде проводити більш детальне знайомство з книгою, що його цікавить.

Сторінка з інформацією та можливістю попереднього перегляду буде відкриватися, як нова сторінка, щоб забезпечити користувач можливість у декількох вкладках відкривати книги, що його цікавлять для подальшого ознайомлення.

Система не буде навантажена іншими логіками та прецедентами, що робить роботу користувача з системою максимально швидкою та простою.

2.3.3 Архітектура застосунку

Більшість традиційних програм розгортається у вигляді одного елемента, який відповідає файлу, або одної вебпрограми, що виконується в домені. Це найпростіша модель розгортання, яка оптимально підходить для безлічі внутрішніх та невеликих загальнодоступних програм. Проте, навіть у

такій простій моделі розгортання, більшість бізнес-додатків використовує переваги логічного поділу на шари.

Монолітна програма повністю замкнута в контексті поведінки. Під час роботи вона може взаємодіяти з іншими службами або сховищами даних, однак основа його поведінки реалізується у власному процесі, а вся програма зазвичай розгортається як один елемент. Для горизонтального масштабування така програма зазвичай повністю дублюється на декількох серверах або віртуальних машинах.

В залежності від розподілення навантаження між клієнтською та серверною частиною можливо виділити декілька типів архітектури: від переважно серверних до переважно клієнтських. Окремо стоять прогресивні вебзастосунки, яким доступні деякі можливості десктопних додатків.

Вебпрограми складаються з серверної частини (back-end, бекенд) та клієнтської частини (front-end, фронтенд). Користувачі взаємодіють із клієнтською частиною через інтерфейс, який відображається у браузері (Chrome, Firefox, Safari, Edge та ін.). За командою користувача запит надсилається на сервер через Інтернет. На сервері його обробляє серверний код та повертає клієнту відповідь (рис. 2.9) [10].



Рисунок 2.9 – Взаємодія клієнта та сервера

У відповіді може бути як готова HTML-сторінка, так і шаблон сторінки або дані, наприклад у форматі XML або JSON. Це залежить від вибраного типу рендерингу (формування) сторінки. Тобто, сторінка може відправлятися

взагалі без змін (статична сторінка) або бекенд вносить до неї зміни, після чого відправляє її браузеру (динамічна сторінка). Рендеринг може здійснюватися або повністю на сервері, або у різних співвідношеннях розподілятися між сервером та клієнтом, або виконуватись лише клієнтом [10].

У поточному випадку користувач буде надсилати пошукові запити до серверів Google, використовуючи API Google Books для отримання інформації про книги, а також використовувати API Embedded Viewer задля додавання попереднього перегляду книги на вебсторінку.

3 РОЗРОБКА ПРОГРАМИ

3.1 Вимоги до інтерфейсу користувача

Для взаємодії користувача та застосунку застосовується користувацький інтерфейс, що є комплексом засобів для взаємодії користувача з технічною системою.

Основними елементами користувацького інтерфейсу є елементи керування, а саме: кнопки, списки, діалогові вікна тощо. Вони дозволяють користувачу виконувати якісь дії з комп'ютерною системою.

Основними вимогами до майже усіх користувацьких інтерфейсів є:

- функціональність(відповідність до задач користувача);
- зрозумілість і логічність;
- забезпечення високої швидкості роботи користувача;
- забезпечення захисту від людських помилок;
- швидке навчання користувача;
- суб'єктивне задоволення користувача;
- для того, щоб ці вимоги виконувались необхідно виконувати наступні правила;
- для підвищення швидкості виконання роботи елементи управління реалізувати помітними і зрозумілими;
- для зменшення кількості помилок користувача: не робити кнопки занадто маленького розміру, через те, що на таку кнопку складно потрапити;
- робити так, щоб користувачеві було важко натиснути не на ту кнопку (допомогтися цього можна або зміною стану кнопки при наведенні на неї курсором, або за допомогою візуальних пауз між кнопками);
- не робити небезпечні для користувача кнопки кнопками за замовчуванням;
- завжди показувати межі діапазону значень, що вводяться

користувачем;

- уникати елементів управління, функції яких змінюються в залежності від контексту;
- для підвищення суб'єктивної швидкості виконання роботи заповнювати паузи між подіями.

Також важливим критерієм для комфорту користувача при користуванні застосунком є правильний підбір кольорів, з урахуванням поточних потреб користувача.

А тому, для можливості зручно користуватися застосунком у будь-яку пору доби було прийнято рішення використовувати приглушені кольори та не використовувати світлі кольори, як кольори фону, щоб користувач комфортно почувався при пошуку у обставинах з низьким освітленням.

На рисунку 3.1 зображено схему сторінки користувацького інтерфейсу з результатами пошуку за пошуковим запитом користувача, де користувач буде бачити усі знайдені варіанти книг, що задовольняють його пошуковому запиту.

Назва застосунку

Уведіть пошуковий запит

Текст пошукового запиту

Назва книги

Ознайомитися

Назва книги

Ознайомитися

Назва книги

Ознайомитися

Назва книги

Ознайомитися

Назва книги

Ознайомитися

Назва книги

Ознайомитися

Назва книги

Ознайомитися

Назва книги

Ознайомитися

Рисунок 3.1 – Приклад розмежованої сторінки користувацького інтерфейсу

При відсутності збігів за пошуковим запитом користувача, він буде потрапляти на сторінку пошуку, що буде відображати порожній список пошуку та з можливістю продовжити пошук (рис. 3.2).

The image shows a web interface for a search application. At the top, there is a header bar with the text "Назва застосунку". Below this, there is a search input area containing a text box with the placeholder "Текст пошукового запиту" and a button labeled "Уведіть пошуковий запит". The main content area below the search bar is empty, indicating that no search results were found.

Рисунок 3.2 – Сторінка з порожнім списком пошукового запиту користувача

Книги при не порожньому пошуковому запиті будуть відображатися у вигляді плиток по 4 книги у рядку. Для кожної з книг буде відображена обкладинка та назва книги, а також кнопка «Ознайомитись» для подальшого відкриття сторінки з більш детальною інформацією про книгу та її попереднім переглядом (рис. 3.3).

На сторінці ознайомлення у користувача буде можливість ознайомитися з більш детальною інформацією про книгу (назва, автор, жанр, категорія, короткий опис) та прочитати перші декілька сторінок книги, що надаються у попередньому перегляді для ознайомлення користувача.

Також, нижче зображення обкладинки книги знаходиться кнопка повернення до сторінки пошуку (рис. 3.1), що повертає користувача у поточній вкладці на сторінку пошуку книги.

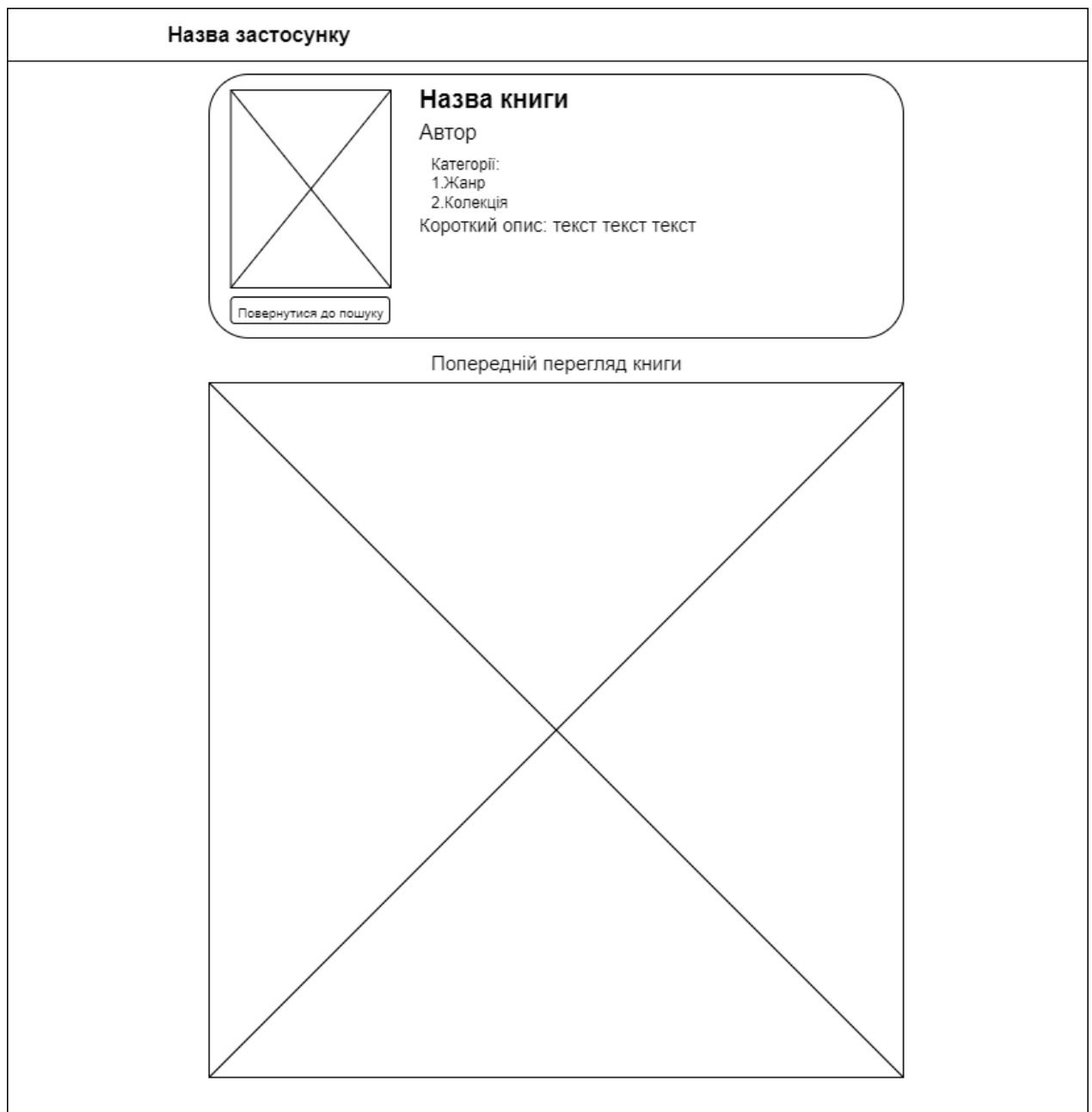


Рисунок 3.3 – Сторінка з детальною інформацією та попереднім переглядом
книги

Даний інтерфейс користувача є достатньо простим та не зможе викликати у користувача труднощів з його освоєнням, бо є логічним та послідовним і дозволяє виконати потреби користувача з пошуком та ознайомленням з книгами, що його цікавлять.

3.2 Розробка інтерфейсу користувача

3.2.1 Опис графічного інтерфейсу користувача

Користувальницький інтерфейс застосунку був розроблений за допомогою HTML, CSS та Bootstrap, а також використовуючи Adobe Photoshop для редагування необхідних візуальних елементів.

Після того, як з'ясовано основні завдання, і вимоги користувачів, необхідно скласти структуру і макет інтерфейсу веб застосунку.

Стартовим вікном програми буде форма для пошуку книг, що дозволить користувачу шукати книги за його пошуковим запитом (рис. 3.4).

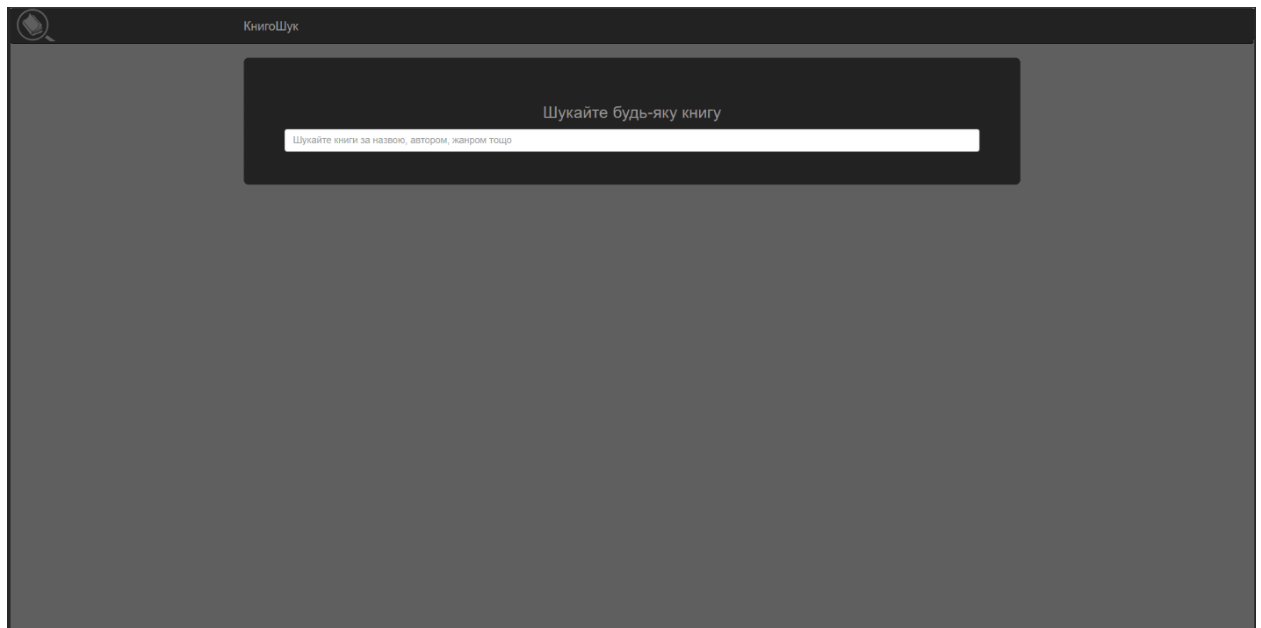


Рисунок 3.4 – Початковий екран застосунку

Елемент у верхньому лівому куті екранів на рисунках 3.4 та 3.6 буде логотипом застосунку та водночас кнопкою для переходу до стартового вікна з пустим списком пошуку книг (рис. 3.4). Далі за логотипом на рисунках слідує другий (рис. 3.4 та рис. 3.6) – назва застосунку, що також є кнопкою, що переводить користувача на стартовий екран з пустим списком книг. За другим елементом на стартовому екрані (рис. 3.4) розташований третій найголовніший – блок для введення пошукового запиту користувача. Сюди

користувач може ввести пошуковий запит про книгу, що його цікавить або жанр чи автора, і отримати відповідь на свій пошуковий запит у вигляді списку книг (рис. 3.5).

Після того, як користувач отримає відповідь на його пошуковий запит, він отримає книги, що його цікавлять у вигляді списку книг, що будуть знаходитися нижче елементу для введення пошукового запиту (рис. 3.5).

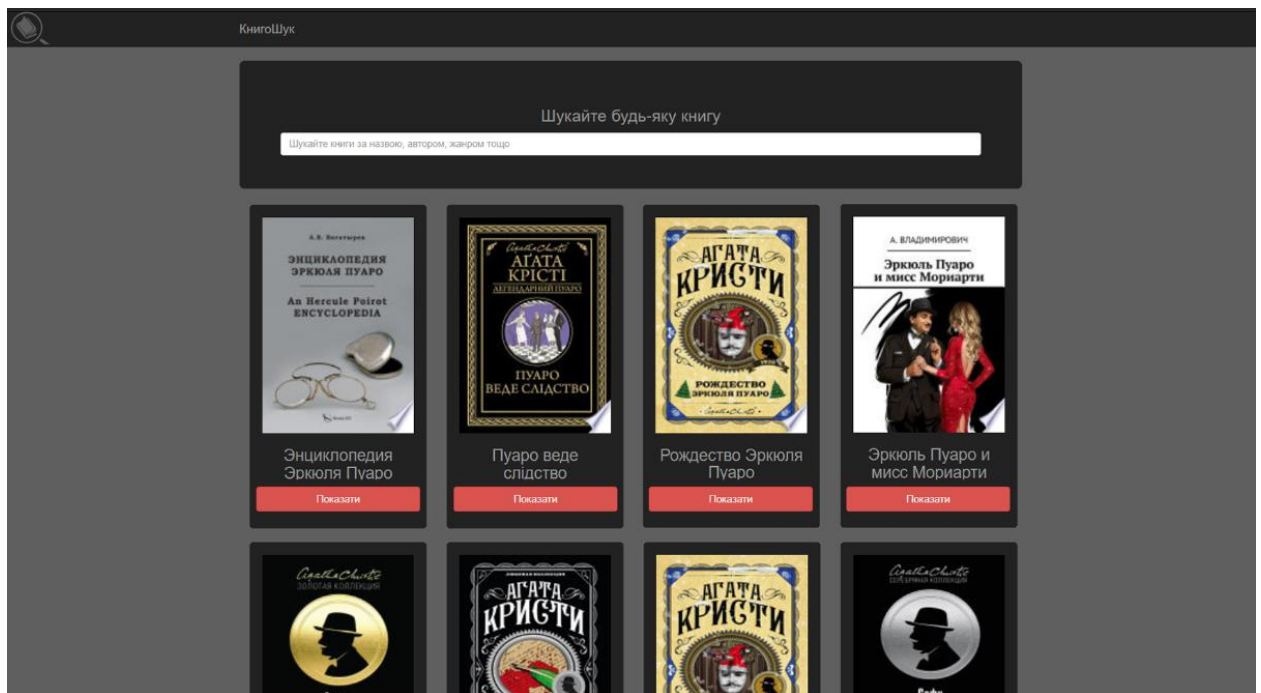


Рисунок 3.5 – Екран з прикладом результатом пошукового запиту

Коли користувач отримує відповідь на свій пошуковий запит, він отримує список книг, що розташовані у центрі екрана у вигляді плиток, у декілька рядків, по 4 книги у кожному задля більш зручної можливості перегляду без надмірного зменшення деталей (рис. 3.5).

Кожна плитка становить собою елемент, що зображує обкладинку книги, її назву та надає можливість за допомогою кнопки «Переглянути» перейти (рис. 3.5) на сторінку з більш детальним описом книги та її попереднім переглядом (рис. 3.6).

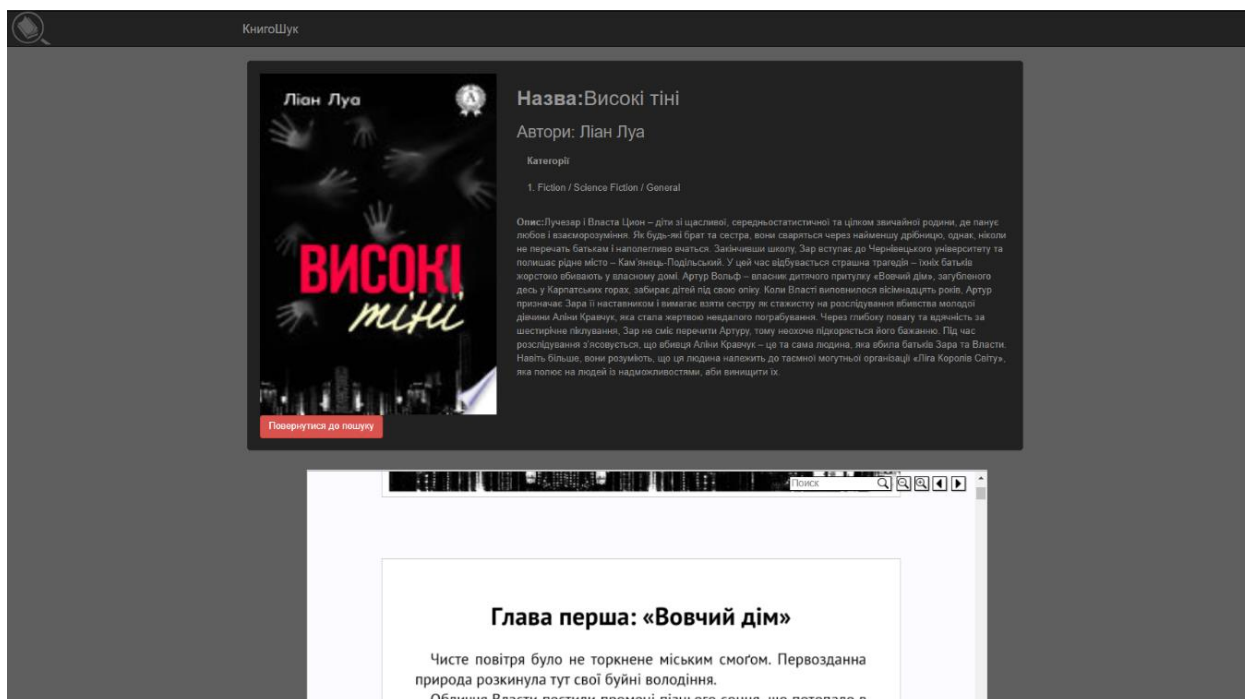


Рисунок 3.6 – Екран конкретної книги

При переході на екран бажаної книги, користувач отримує про неї більш повну інформацію, а саме:

- назву;
- автора (ів);
- категорії;
- короткий опис;
- попередній перегляд книги.

На цій сторінці (рис. 3.6) користувач може більш докладний опис книги, побачити великим планом обкладинку та інформацію про книгу, що допоможе користувачу визначитися з вибором книги, а також за допомогою додаткової інформації про книгу продовжити свій пошук за її автором або за однією з категорій.

У цьому ж блоці знаходиться кнопка «Повернутися до пошуку», що повертає користувача на початковий екран з порожнім списком пошуку для продовження подальшого пошуку книг (рис. 3.6).

Ця інформація знаходиться у верхній частині екрана по центру у вигляді блоку.

Також користувач отримує доступ до попереднього перегляду, що знаходиться нижче блоку з додатковою інформацією про книгу, і становить собою великий блок з pdf версією, а саме: з певною кількістю початкових сторінок, що надаються користувачу для ознайомлення.

Блок з попереднім переглядом реалізований за допомогою API Embedded Viewer, що дозволяє вставляти попередні перегляди Google Books на вебсторінку і програмно керувати цими попередніми переглядами.

Також Embedded Viewer надає змогу користувачу шукати фрази або слова у тій частині попереднього перегляду книги, що він переглядає та переключатися між результатами пошуку (рис. 3.7) [11].

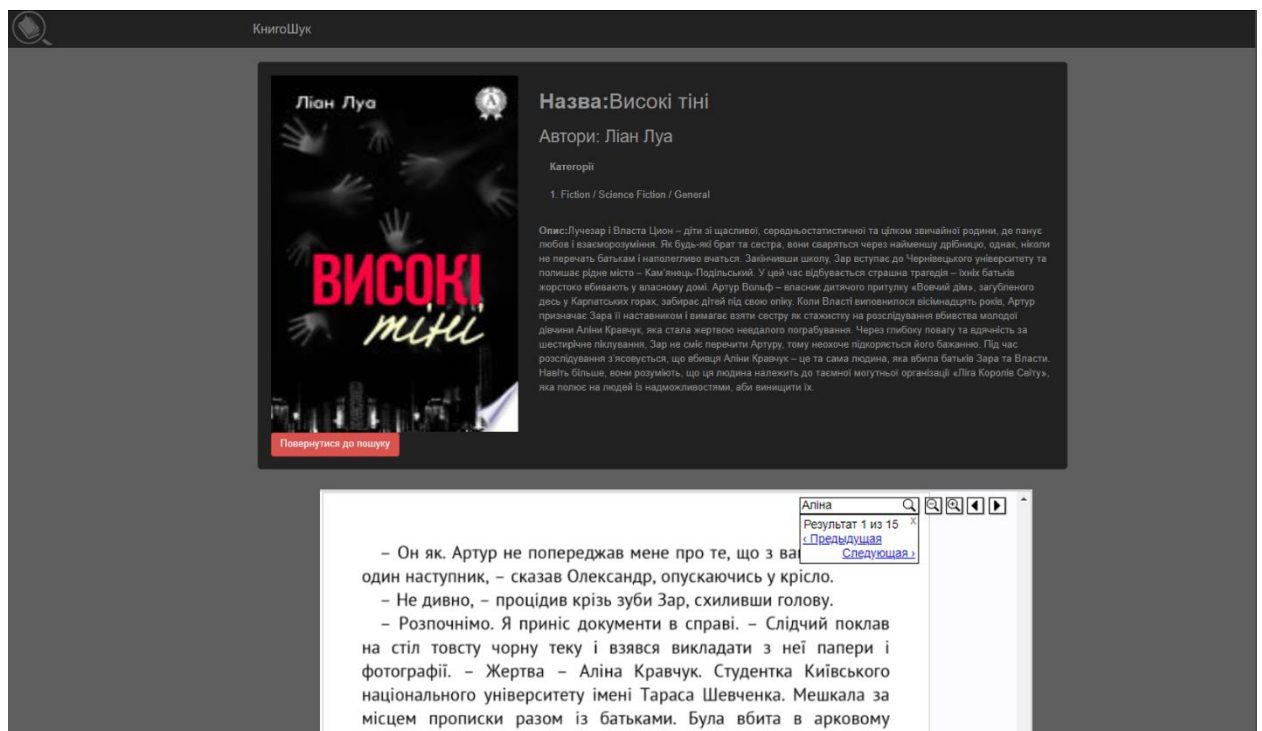


Рисунок 3.7 – Пошук слова «Аліна» у попередньому перегляді книги

У випадках, коли користувач не має доступу до мережі інтернет або у сервера Google Books чи Embedded Viewer є проблеми з доступом, користувач буде бачити на будь-якій сторінці застосунку анімацію завантаження сторінки

(рис. 3.8), яка зникне, як тільки проблеми з доступом до мережі інтернет або серверів Google Books або Embedded Viewer зникнуть, після чого користувач отримає доступ до бажаної сторінки вебзастосунку.

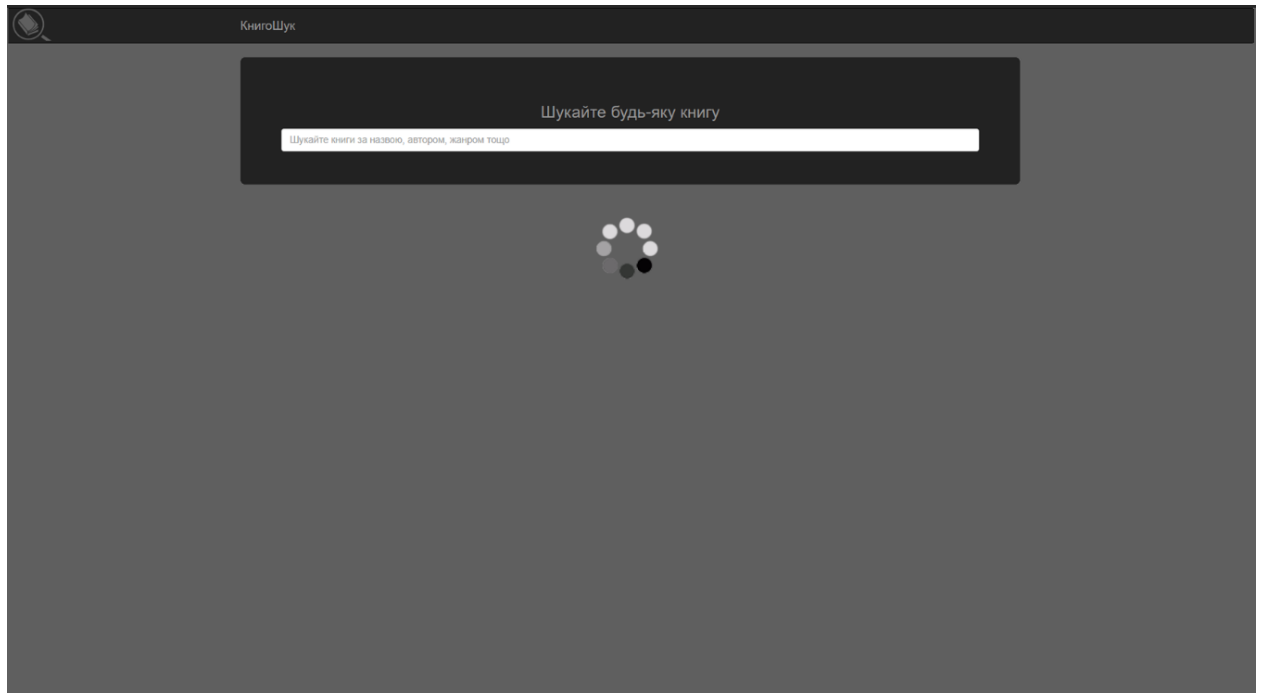


Рисунок 3.8 – Випадок з проблемою завантаження змісту сторінки вебзастосунку

3.2.2 Структура та реалізація інтерфейсу

Інтерфейс користувача представлений у вигляді двох вебсторінок, що зверстані за допомогою HTML та CSS та зроблені інтерактивними за допомогою JavaScript.

Весь інтерфейс утримується у файлах:

- “MainPage.html”;
- “Book.html”;
- “style.css”;
- “Scripts.js”.

Також інтерфейс використовує ресурси (зображення та GIF), що знаходяться у теках:

- “assets”;
- “img”.

Файл “MainPage.html” є реалізацією головної сторінки застосунку (рис. 3.4). Він являє собою HTML сторінку, з версткою відповідною до рис. 3.4.

Для розробки головної сторінки застосунку використовувалась базова структура HTML документів (рис. 3.9).

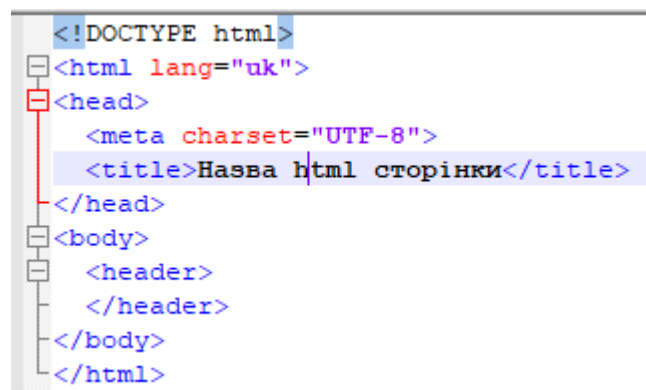


Рисунок 3.9 – Базова структура HTML-документу

Перший елемент у будь-якому HTML-документі є DOCTYPE. Він не належить до тегів і не може відображатися на сторінці. Він існує задля того, щоб давати браузеру вказівки, який стандарт HTML використовується у цьому документі (рис.3.9).

Тег `<html></html>` є основою основ. Саме всередині нього міститься вся інформація. Завдяки цьому тегу браузер розуміє де починається контент, який необхідно обробити як HTML.

Важливою частиною тега HTML є наявність атрибута `lang`. У ньому вказується мова, на якій відображається вебсторінка. За допомогою цього атрибуту браузери можуть правильно відображати безліч специфічних символів, які присутні у різних мовах. Крім цього, атрибут `lang` починає використовуватися і CSS. У нових стандартах CSS з'являються властивості, що

спираються на цей атрибут. Наприклад, дозволяють правильно перекладати слова у тексті [13].

Як значення атрибут `lang` приймає коди мов, наприклад: для англійської – `en`, для української – `uk`.

Тег `<head></head>` служить для зберігання службової інформації. Тут можливі різні поєднання тегів, які підказують браузеру назву сторінки, опис, ключові слова та інше, тобто утримує метадані сторінки. На даний момент метадані відповідає не тільки за службову інформацію для браузера, але й активно використовується для просування сайту за допомогою SEO. Пошукові системи видають користувачу результати його пошукових запитів засновуючись і не метаданих також та на основі безлічі алгоритмів визначають місце сайту при різних пошукових запитах.

Будь-які дані, вказані всередині тегу `head`, не видно користувачу при відображенні сторінки в браузері.

Метатег з атрибутом `charset`, що дозволяє встановити кодування документа.

Для створення заголовка сайту використовується тег `<title></title>`, у якому вказується потрібна інформація.

Після тегу `head` у документі вказується парний тег `<body></body>`, який є основною частиною усієї сторінки. Саме тут розміщується вся інформація, яка виводиться на сторінці [13].

Відповідно до основної структури HTML документу була наповнена головна сторінка, на якій знаходиться заголовок з логотипом та назвою застосунку, а також блок з полем для введення пошукового запиту користувача.

Заголовок був реалізований за допомогою тегу `<nav></nav>`, що визначає окрему секцію документа, призначення якої – позначення посилань навігації (як усередині поточного документа, так і ведучих на іншу сторінку).

Також використовувався тег `<div></div>`, що є універсальним контейнером потокового контенту. Він не впливає на контент або макет, доки

не буде стилізований за допомогою CSS. Він часто використовується для групування контенту на сторінці, що дозволяє легко його стилізувати.

Ще одним тегом, що був використаний є тег `<a></a>`, що є тегом який визначає гіперпосилання для переходу на певне місце на сторінці або на іншу сторінку в Інтернеті.

За допомогою цих тегів будується структура HTML сторінки, стилізація її відбувається за допомогою CSS, використовуючи глобальні атрибути, такі як, `class` або `id`.

`Class` – це перелік розділених пробілами класів елемента. Класи дозволяють CSS та JavaScript обирати та отримувати доступ до конкретних елементів через селектори або через функції [14].

`ID` – визначає ідентифікатор, який має бути унікальним для всього документу. Він призначений для ідентифікації елемента під час створення посилань на нього, виконання скриптів або застосування стилів (за допомогою CSS) [14].

За допомогою класів і створюється стилізація документу. Можна створювати власні класи або використовувати вже створені користувацькі класи, що надає, наприклад Bootstrap.

Bootstrap – це набір інструментів для створення сайтів і вебдодатків. Включає HTML і CSS-шаблони оформлення для типографії, вебформ, кнопок, міток, блоків навігації та інших компонентів вебінтерфейсу, включаючи JavaScript-розширення. Він є дуже зручним для швидкої верстки адаптивних дизайнів [15].

Для верстки сторінок застосунку були використані, як свої класи, так і шаблони, що надає Bootstrap.

Були використані такі класи та теги, як:

- `Container` – що є найпростішим елементом макету в Bootstrap і є обов'язковими при використанні системи сіток за замовчуванням. Контейнери використовуються для розміщення, заповнення та (іноді) центрування вмісту в них. Хоча контейнери можуть бути вкладеними, більшість макетів не

потребують вкладеного контейнера [16];

- Navbar – це частина інтерфейсу користувача, що допомагає користувачеві перескакувати на різні сторінки (page) у вебсайті. Був використаний, як клас navbar, так і його різновиди (navbar-brand, navbar-nav, navbar-inverse та інші) [17];

- Jumbotron – що є легким, гнучким компонентом, який за бажанням може розширити всю область перегляду, для демонстрації ключових повідомлень на сторінці [18];

- використовувались також звичайні текстові утиліти для контролю вирівнювання, обтікання, ваги, такі як: text, text-center, text-sm-left та інші;

- для створення блоку пошуку книги був використаний тег `<form></form>`, що є тегом, що представляє розділ документа, що містить інтерактивні елементи керування, які дозволяють користувачеві надсилати інформацію на вебсервер та клас `form-control`, що є класом для створення загального вигляду, стану фокусування, розміру та інших параметрів форми [19];

- при створенні gif був використаний власний id `preloader` для задання `margin` елемента, що буде з'являтися при неможливості завантажити сторінку та клас `hidden`, що є класом для контролю відображення або не відображення елементів;

- для створення рядків книг при відображенні результату пошукового запиту користувача був використаний id `books` для задання розміру плиток та класи `row` і `col`, що є класами, що реалізують рядки і стовпці у системі сіток Bootstrap, яка використовує серію контейнерів, рядків і стовпців для компоновання та вирівнювання вмісту;

- для створення кнопок використовувалися класи `btn`, `btn-block`, `btn-danger` та інші дочірні класи класу `btn` з Bootstrap. Класи `.btn` призначені для використання з елементом `<button>`. Однак вони також можуть бути використані з елементами `<a>` або `<input>` (хоча деякі браузерери можуть

застосовувати дещо інше відтворення). При використанні класів кнопок з тегами `<a>`, які використовуються для активації внутрішньої функціональності (наприклад, згортання вмісту), а не посилань на нові сторінки або розділи на поточній сторінці, цим посиланням, слід надати `role="button"` та передати своє призначення допоміжним технологіям, таким як програми зчитування з екрана [20].

3.3 Реалізація функціоналу застосунку

3.3.1 Робота з даними та реалізація алгоритмів

Для реалізації інтерактивності сторінок застосунку використовувалась мова програмування JavaScript. Вона використовувалась для доступу до Google Books та Embedded Viewer, а також для передачі пошукових запитів користувача на сервер, отримання відповіді та формування візуальної частини при виведенні результатів пошуку користувачу.

Одним із інструментів, що використовувалися є JQuery, що є швидкою, і багатофункціональна бібліотека JavaScript, що спрощує такі речі, як обхід і маніпуляції з документами HTML, обробку подій, анімацію та Ajax, завдяки простому у використанні API, який працює в багатьох браузерах .

Для імплементації JavaScript до HTML сторінок застосовувався тег `<script></script>`, що використовується для вбудовування або підключення коду JavaScript, що буде виконуватися [21].

Також для коректного підключення JQuery був використаний механізм цілісності субресурсів, використовуючи атрибути `integrity` та `crossorigin`, за допомогою якого користувацькі агенти можуть перевірити, що отриманий ресурс був доставлений без несподіваних або несанкціонованих змін [22].

Атрибут `integrity` дозволяє браузеру звіряти вміст вихідного файлу з наданим значенням хеша, щоб переконатися, що код ніколи не завантажуватиметься, якщо джерело було змінено.

Атрибут `crossorigin` вказує, чи буде використовуватися механізм передачі облікових даних CORS (Cross-Origin Resource Sharing) [24].

Ще одним інструментом, що використовувався є бібліотека `Axios`, що є HTTP-клієнтом для `node.js` і браузера. Вона є ізоморфний (може працювати у браузері та в `NodeJS` з однаковою кодовою базою). На стороні сервера вона використовує власний `http`-модуль `node.js`, тоді як на клієнті (браузері) вона використовує `XMLHttpRequests` [25].

`Axios` використовується для:

- створення `XMLHttpRequests` з браузера;
- створення `http`-запитів з `node.js`;
- підтримки `Promise API`;
- перехоплення запитів та відповідей;
- трансформування даних запитів та відповіді;
- скасування запитів;
- автоматичного перетворення для даних `JSON`;
- підтримки на стороні клієнта для захисту від `XSRF`.

За допомогою `JavaScript` та його бібліотек був реалізований основний механізм – тобто система пошуку та рекомендацій книг користувачу, а також функції щодо створення інтерактивності застосунку, тобто:

- можливості пошуку книг;
- отримання доступу та використання `Google Books API` та `Embedded Viewer` для отримання інформації про книги, користувачу;
- перевірки готовності пошукового запиту користувача до відображення у коректному вигляді;
- механізм відображення пошукового запиту користувачу;
- відстеження та обробка помилок;
- перехід між сторінками застосунку.

Система пошуку книг заснована на пошуковому запиті користувача, що передається до `Google Books`. У випадку, якщо пошуковий запит користувача

був назвою або частковою назвою книги, він отримує усі книги, які мають у базі даних Google Books, що мають схожі назви починаючи з найбільш схожих на пошуковий запит користувача (рис 3.5).

У випадку, коли користувач шукає автора (вибір чи зміна параметрів пошуку за назвою, автором чи жанром відбувається автоматично), як результат свого пошукового запиту, він отримає як результат пошукового запиту у першу чергу список книг автора, що його цікавить. Після книг автора, що цікавить користувача будуть виведені книги, що є схожими на книги, що вже є у пошуковому запиті. Критерієм схожості для пошуку книг є категорії. У поточному пошуку отримується найбільш популярна категорія, тобто категорія, до якої належить найбільша кількість книг, та за цією категорією (категоріями) додаються до поточного пошукового запиту книги, що задовольняють критеріям.

Для розрахунку критерію рекомендації книг система аналізує результат поточного пошукового запиту користувача, отримуючи інформацію про книги у результаті запиту та розраховує кількість книг, що мають відношення до усіх категорій у результаті пошукового запиту. Також до уваги беруться такі критерії, як мова, вікове обмеження, доступність книги та її pdf-версії і можливість відкрити її у Embedded Viewer. Після знаходження усіх категорій запиту та підрахунку екземплярів книг, що входять до кожної з категорій, система сортує категорії за популярністю та розраховує для кожної з категорій долю екземплярів, що входять до кожної з категорій. Базуючись на цих долях система розширює пошуковий запит ще п'ятдесятьма книгами, і відповідно до часток додає до пошукового запиту популярні книги відповідних категорій. Аналогічна ситуація існує і при пошуку за назвою книги.

Дані про екземпляри книг результату пошукового запиту утримуються у JSON файлах, що надаються сервером Google Books у відповідь на пошуковий запит (рис. 3.10).

```

{
  "kind": "books#volume",
  "id": "KUX1DQAAQBAJ",
  "etag": "McJMuRjYotA",
  "selfLink": "https://www.googleapis.com/books/v1/volumes/KUX1DQAAQBAJ",
  "volumeInfo": {
    "title": "Бисоки тіні",
    "authors": [
      "Ліан Ляа"
    ],
    "publisher": "Litres",
    "publishedDate": "2022-05-15",
    "description": "Лучезар і Власта Ціон – діти зі щасливої, середньостатистичної та цілком звичайної родини, де панує любов і взаєморозуміння. Як будь-які брат та сестра, вони сваряться через найменшу дрібницю, однак, ніколи не перечать батькам і наполегливо вчаться. Закінчивши школу, Зар вступає до Чернівецького університету та полишає рідне місто – Кам'янець-Подільський. У цей час відбувається страшна трагедія – їхніх батьків жорстоко вбивають у власному домі. Артур Вольф – власник дитячого притулку «Вовчий дім», загубленого десь у Карпатських горах, забирає дітей під свою опіку. Коли Власті виповнилося вісімнадцять років, Артур призначає Зара її наставником і вимагає взяти сестру як стажистку на розслідування вбивства молодшої дівчини Ліани Кравчук, яка стала жертвою нещасного пограбування. Через глибоку повагу та відчуженість за шестирічне піклування, Зар не сміє перечити Артуру, тому неохоче підкоряється його бажанню. Під час розслідування з'ясовується, що вбивця Ліани Кравчук – це та сама людина, яка вбила батьків Зара та Власти. Навіть більше, вони розуміють, що ця людина належить до таємної могутньої організації «Ліра Королів Світ», яка полюбляє на людей із надможливостями, аби винищити їх.",
    "industryIdentifiers": [
      {
        "type": "ISBN_10",
        "identifier": "5040313918"
      },
      {
        "type": "ISBN_13",
        "identifier": "9785040313914"
      }
    ],
    "readingModes": {
      "text": true,
      "image": true
    },
    "pageCount": 194,
    "printedPageCount": 194,
    "printType": "BOOK",
    "categories": [
      "Fiction / Science Fiction / General"
    ],
    "maturityRating": "NOT_MATURE",
    "allowAnonLogging": false,
    "contentVersion": "1.1.2.0.preview.3",
    "panelizationSummary": {
      "containsImageBubbles": false,
      "containsImageBubbles": false
    },
    "imageLinks": {
      "smallThumbnail": "http://books.google.com/books/publisher/content?id=KUX1DQAAQBAJ&printsec=frontcover&img=1&zoom=5&edge=cur&imgtk=AFLRE712TuaWuGkgnjTqHeg8Yf9kQFZckTSH5Yg1fRm_Ym3tbEci2hItMCSJd-Z1o0ztrq4PR_gEdAQ5Dno9Wox4-ThSCjwR_1ebJkC_xv0IMGZ2pinS918PQZ4ytpL2D0G1ZTRD3zB&source=gb_s_api",
      "thumbnail": "http://books.google.com/books/publisher/content?id=KUX1DQAAQBAJ&printsec=frontcover&img=1&zoom=1&edge=cur&imgtk=AFLRE7014wL-xxTPPpRm1CRoPxvteAzPwK8pNjGpAuL2Vy6w0Udq8buUE_1f4CRZnbYb67ZZLv71nCNxjescFHQsEmLk77GWFwedEO852F7606EKEP3JfscIvsiVgYpN1XD5EKO-u&source=gb_s_api",
      "small": "http://books.google.com/books/publisher/content?id=KUX1DQAAQBAJ&printsec=frontcover&img=1&zoom=2&edge=cur&imgtk=AFLRE70VdsBICAHuqTkecKvW8zp8zQP8zXwTsZkEw_Auxsa0pG2UTQNSDaZooOzCuHTKDav68ZQEkdMDbDP_65Z5T2s9taggbUHgg-SNjzusa0UQ9am_D_ZXtXmmtmPhXAb1vHq-s&source=gb_s_api",
      "medium": "http://books.google.com/books/publisher/content?id=KUX1DQAAQBAJ&printsec=frontcover&img=1&zoom=3&edge=cur&imgtk=AFLRE71cuFJpei9isyKev2-1oT04vtmSLGZEG58Bp3J1WCJaRo2hzmPACqNzYGVFCS92-d8-puqpgqFLWQe-IFKub410wou02vKZL2HjPyieMF-HH2HMJ3Tut8Hhun-gN2mAZXJ-FBdj&source=gb_s_api",
      "large": "http://books.google.com/books/publisher/content?id=KUX1DQAAQBAJ&printsec=frontcover&img=1&zoom=4&edge=cur&imgtk=AFLRE72eMwHCqFOEEr7KFAjGgEXKvX8wXol3YQdZk1nxt2KqNRDwJugimFR-D5q9ngudzkmgwq6QkpbN35tX6Uhl09ITP20ztR2f1pfhKuj4RX8KXUhsMEApMx6OxUKRQ-agpb&source=gb_s_api",
      "extraLarge": "http://books.google.com/books/publisher/content?id=KUX1DQAAQBAJ&printsec=frontcover&img=1&zoom=6&edge=cur&imgtk=AFLRE7164kbzyeB08H8_P6vVf2M18Cjx2G941j38FGMT2xqfLo7WygZ3jyjoYu0hcezzDv9Jb1Pq12cxt14eQ2jR39bGMMKKTfa uyvUvR8NOSCy3UujyUhdCZVH5VjXhJmOfqR8R16&source=gb_s_api"
    },
    "language": "uk",
    "previewLink": "http://books.google.com.ua/books?id=KUX1DQAAQBAJ&hl=&source=gb_s_api",
    "infoLink": "https://play.google.com/store/books/details?id=KUX1DQAAQBAJ&source=gb_s_api",
    "canonicalVolumeLink": "https://play.google.com/store/books/details?id=KUX1DQAAQBAJ"
  },
  "layerInfo": {
    "layers": [
    ]
  }
}

```

Рисунок 3.10 – Приклад формату даних про екземпляр книги

У JSON файлах для кожного з екземплярів книги утримується уся інформація, що є на сервері Google Books про поточну книгу, а саме:

- ID книги;
- електронний tag;
- посилання на книгу;
- назва;
- автор;
- видавець;
- опис;
- ISBN номер, що необхідний для відображення книги у Embedded

Viewer;

- наявність зображення обкладинки;
- посилання на зображення обкладинки у п'яти різних розмірах та розширеннях;
- метадані, такі як:
 - вікове обмеження;
 - категорії;
 - кількість сторінок;
 - мова написання;
 - система монетизації;
 - ціна;
 - валюта продажу;
 - кількість цифрових копій книги;
 - посилання на pdf-документ та інші технічна інформація щодо книги.

Відповідно, з JSON файлів до іншого JSON файлу “QueryInfo” (рис. 3.11) збирається інформація необхідна для аналізу поточного пошукового запиту. Тобто, з файлів витягається необхідна інформація про кожен з екземплярів книги, після чого заноситься до JSON файлу поточного результату пошукового запиту, після чого ця інформація агрегується і виводяться відсотки категорій книг та книги, що є доступними для ознайомлення відповідною мовою відповідно до відсотків кожної з категорій додаються до результату пошукового запиту користувача.

```
[
  "books":
  [
    {
      "id": "AHW1DQAAQBAJ",
      "language": "uk",
      "maturityRating": "NOT_MATURE",
      "categories": [
        "Fiction / Science Fiction / General"
      ],
      "isEbook": true,
      "embeddable": true
    },
    {
      "id": "KUX1DQAAQBAJ",
      "language": "uk",
      "maturityRating": "NOT_MATURE",
      "categories": [
        "Thriller / General"
      ],
      "isEbook": true,
      "embeddable": true
    },
    {
      "id": "aqAS1AAQBAJ",
      "language": "uk",
      "maturityRating": "NOT_MATURE",
      "categories": [
        "Fiction / Detective / Drama"
      ],
      "isEbook": true,
      "embeddable": false
    },
    {
      "id": "KUX1DQf00BAJ",
      "language": "uk",
      "maturityRating": "MATURE",
      "categories": [
        "Fiction / Science Fiction / General"
      ],
      "isEbook": false,
      "embeddable": true
    },
    {
      "id": "EIX1ASUAQBAJ",
      "language": "uk",
      "maturityRating": "NOT_MATURE",
      "categories": [
        "Fiction / Science Fiction / General"
      ],
      "isEbook": true,
      "embeddable": true
    },
  ],
]
]
```

Рисунок 3.11 – Структура JSON файлу результату пошукового запиту

Також пошук проводиться за категоріями, де категорії – це список предметних заголовків BISAC, що найкраще описують зміст книг у декількох словах, наприклад:

- архітектура;
- мистецтво;
- біографії та автобіографії;

- бізнес та економіка;
- комп'ютери;
- дизайн;
- драма та інші.

Користувач при введенні пошукового запиту, що є категорією книги, отримає найбільш популярні книги категорії, яку він шукав.

Можливості пошуку за видавцем, номером ISBN (International Standard Book Number), пошуку у LCCN (Library of Congress Control Number) або OCLC(Online Computer Library Center number) не були реалізовані через те, що є занадто специфічними і не мають сенсу для пересічного користувача.

3.3.2 Реалізація попереднього показу книги

Відображення книги на вебсторінці було реалізовано за допомогою Embedded Viewer API. Він дає змогу вставляти вміст книг із Google Books безпосередньо на вебсторінки за допомогою JavaScript.

Після того, як вбудований засіб перегляду було додано до вебсторінки, ним можна керувати за допомогою функцій JavaScript. Це дозволяє виконувати дії, подібні до того, що може зробити користувач, натиснувши елементи керування: є можливість перейти на наступну сторінку, збільшити або зменшити масштаб, виділити пошукові терміни тощо. Також можна отримати номер поточної сторінки, дозволяючи коду реагувати, коли користувач переміщається по книзі.

Відповідно, за допомогою HTML та CSS було створено блок, у якому буде розміщуватися вбудований у сторінку переглядач книги, та за допомогою JavaScript, а саме: тим самим тегам `<script></script>` був підключений API, створена функція, що відправляє запит до серверу Google Books (рис. 3.12), після чого, *.pdf версія книги завантажується у попередньо визначене місце на сторінці (рис. 3.6).

```
function initialize(isbn) {
    var viewer = new google.books.DefaultViewer(document.getElementById('bookCanvas'));
    viewer.load('ISBN:' + isbn);
}
```

Рисунок 3.12 – Функція, що відправляє запит до сервера Google Books

Після чого, користувач має доступ, як до інформації про книгу, так і має можливість ознайомитися з її вмістом прямо на сторінці.

3.3.3 Вхідні та вихідні дані

Вхідними даними є ті дані, що користувач буде вносити у систему, а саме пошукові запити користувача до системи.

Вони будуть представлені у вигляді текстового формату даних, що користувач буде власноруч вводити до пошукової строки застосунку (рис. 3.4).

До вихідних даних застосунку відносяться результати пошукового запиту користувача, що представлені у вигляді плиток з книгами (рис. 3.5). Кожна з плиток є окремим об'єктом, що утримує у собі інформацію про книгу, таку як:

- назва;
- картинка обкладинки;
- кнопка, що відкриває сторінку з більш детальною інформацією

про книгу.

На етапі відповіді на пошуковий запит користувача у вигляді списку книг ніяка інформація користувачу не надається.

При відсутності результатів пошукового запиту вихідними даними є початкова сторінка застосунку (рис. 3.4).

Вебзастосунок також утримує сторінку з більш детальною інформацією про книгу (рис. 3.7), де користувачу надається така інформація про книгу, як:

- велика картинка обкладинки;
- назва;
- автор;
- категорії до яких відноситься книга;
- короткий опис книги;
- кнопка для повернення до головної сторінки застосунку для продовження пошуку;
- попередній перегляд книги у вигляді вбудованої у вебсторінку pdf версії книги з обмеженою кількістю сторінок для ознайомлення.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення та умови застосування програми

Призначенням програми «КнигоШук» є надання можливостей користувачам щодо пошуку та підбору книг за їх пошуковими запитами. Це забезпечує швидкий та зручний пошук книг, що цікавлять користувача.

Програма забезпечує наступні функції:

- пошук книг за пошуковими запитами;
- рекомендація книг за пошуковим запитом користувача;
- перегляд списків книг та скороченої інформації про них;
- відкриття сторінок з детальною інформацією про книги;
- надання можливості попереднього перегляду обраною книги на вебсторінці застосунку.

Для серверу, що буде забезпечувати роботу застосунку «КнигоШук» на 1000 - 3000 осіб, що одночасно користуються сервісом:

- кількість серверів не менше за 3, з урахування відмовостійкості – 4;
- середовище функціонування – віртуальна інфраструктура;
- процесор: віртуальних ядер – 4, частота > 2,6 GHz (покоління Haswell або новіше);
- оперативна пам'ять 10 ГБ;
- дисковий простір (об'єм, IOPS, latency) – операційна система - 60 GB, 100 IOPS, <10 мс, додаток – 20 GB, 100 IOPS, <10 мс;
- співвідношення віртуальних і фізичних ядер має бути не більше 2 віртуальних ядер на 1 фізичне ядро;
- мережевий інтерфейс кількість – 1, швидкість > = 1 GbE.

Для серверу, що буде забезпечувати роботу БД на 1000 – 3000 осіб, що одночасно користуються сервісом:

- кількість серверів – 1, з урахування відмовостійкості – 2;

- середовище функціонування – віртуальна інфраструктура (переважно фізичні сервера);
- процесор: віртуальних ядер – 12, частота > 2.6 GHz (покоління Haswell або новіше);
- оперативна пам'ять 86GB;
- дисковий простір (об'єм, IOPS, latency) – операційна система – 80 GB, 100 IOPS, < 7 мс, додаток – 1000 GB, 10000 IOPS, < 5 мс;
- співвідношення віртуальних і фізичних ядер не більше 1 віртуального ядра на 1 цієї статті фізична ядро;
- мережевий інтерфейс, кількість – 2, швидкість > = 10 GbE.

Для нормального функціонування програми на стороні користувача, повинні виконуватися наступні умови до апаратної частини клієнту:

- IBM-сумісний комп'ютер з підтримкою технології Internet;
- характеристиками не нижче: процесор Pentium 2.0 GHz, RAM 1 GB;
- клавіатура, маніпулятор типу миша або тачскрін.

Умови до програмної частини, для нормального функціонування програми:

- остання версія ПЗ КнигоШук;
- встановлена Microsoft Visual Studio 2019(або новіша);
- встановленні пакети NuGet: Bootstrap – бібліотека для коректного відображення стилів у поданнях.

4.2 Характеристики програми

Проект виконаний за допомогою мову програмування високого рівня JavaScript, HTML та CSS у середовищі розробки Microsoft Visual Studio 2017. Тека «assets» містить *.gif файли, що використовуються у застосунку. Тека «css» містить файли зі стилями, що використовуються у застосунку. Проект

містить також теку «img», де знаходяться файли *.png та *.jpg. Проєкт містить теку «js», де знаходяться файли JavaScript, що використовуються у застосунку, а також теку «HTML», у якій знаходяться реалізації усіх сторінок вебзастосунку.

4.3 Звертання до програми

Для звертання до вебзастосунку користувачами здійснюється за допомоги персонального комп'ютера, браузеру та мережі Інтернет. Застосунок надсилає та отримує з серверу запити типу «GET», «POST».

5 КЕРІВНИЦТВО ОПЕРАТОРА

5.1 Призначення та умови застосування програми

Призначенням програми «КнигоШук» є надання можливостей користувачам щодо пошуку та підбору книг за їх пошуковими запитами. Це забезпечує швидкий та зручний пошук книг, що цікавлять користувача.

Програма забезпечує наступні функції:

- пошук книг за пошуковими запитами;
- рекомендація книг за пошуковим запитом користувача;
- перегляд списків книг та скороченої інформації про них;
- відкриття сторінок з детальною інформацією про книги;
- надання можливості попереднього перегляду обраної книги на вебсторінці застосунку.

5.2 Умови виконання програми

Для нормального функціонування програми, повинні виконуватися наступні умови до апаратної частини клієнту:

- IBM-сумісний комп'ютер з підтримкою технології Internet;
- характеристиками не нижче: процесор Pentium 2.0 GHz, RAM 1 GB;
- клавіатура, маніпулятор типу миша або тачскрін.

Програмні вимоги:

- на комп'ютері повинна бути встановлена операційна система сімейства Microsoft Windows, Unix або macOS;
- наявність одного з браузерів: Mozilla FireFox, Microsoft Edge, Google Chrome або іншого.

5.3 Виконання програми

Для доступу до вебзастосунку необхідно перейти на сайт, на якому буде розміщено вебзастосунок. Після цього користувач побачить головну сторінку 5.1).

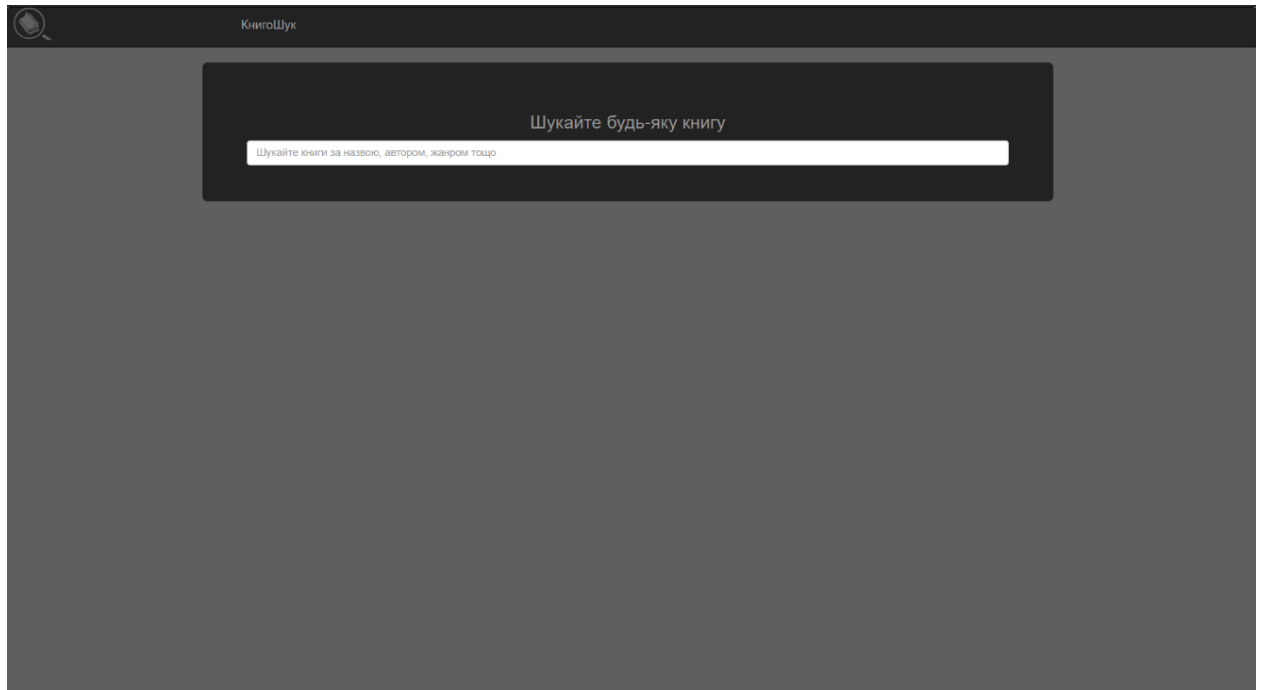


Рисунок 5.1 – Головна сторінка застосунку

На головній сторінці застосунку користувач має можливість почати пошук книг, у пошуковому рядку, що знаходиться у середині верхньої частини головного екрану (рис. 5.2).

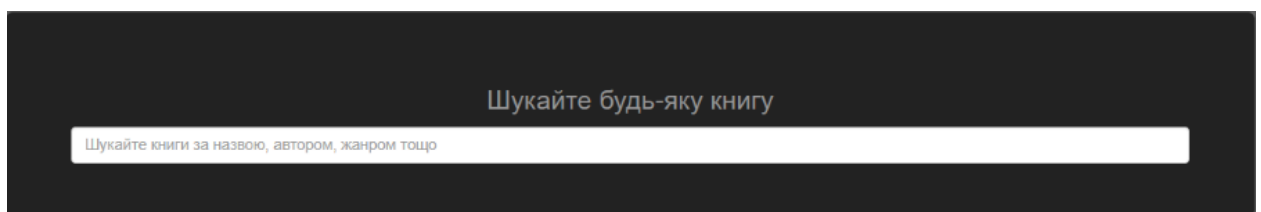


Рисунок 5.2 – Пошуковий рядок головної сторінки застосунку

5.4 Повідомлення оператора

Повідомлення для користувача можуть з'явитися з наступних причин:

- за пошуковим запитом користувача нічого не знайдено (рис 5.3);
- немає доступу до мережі інтернет або серверів Google Books (рис.

5.4).

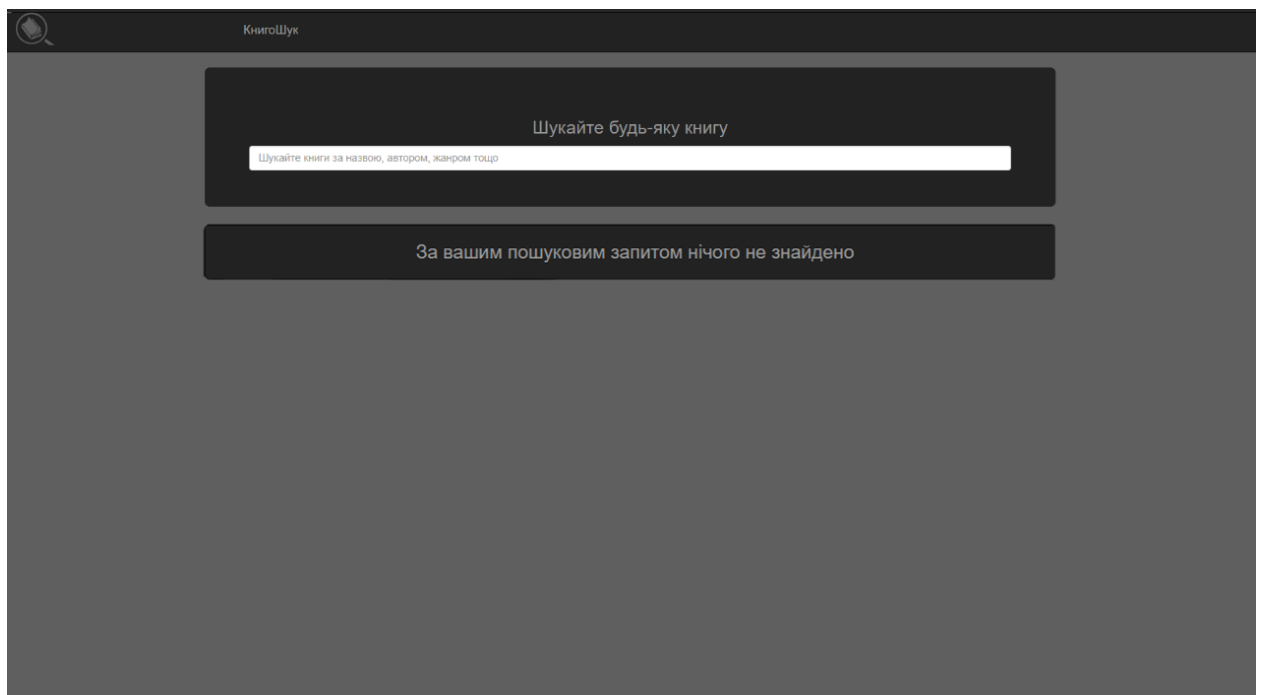


Рисунок 5.3 – Повідомленні при відсутності результатів за пошуковим запитом користувача

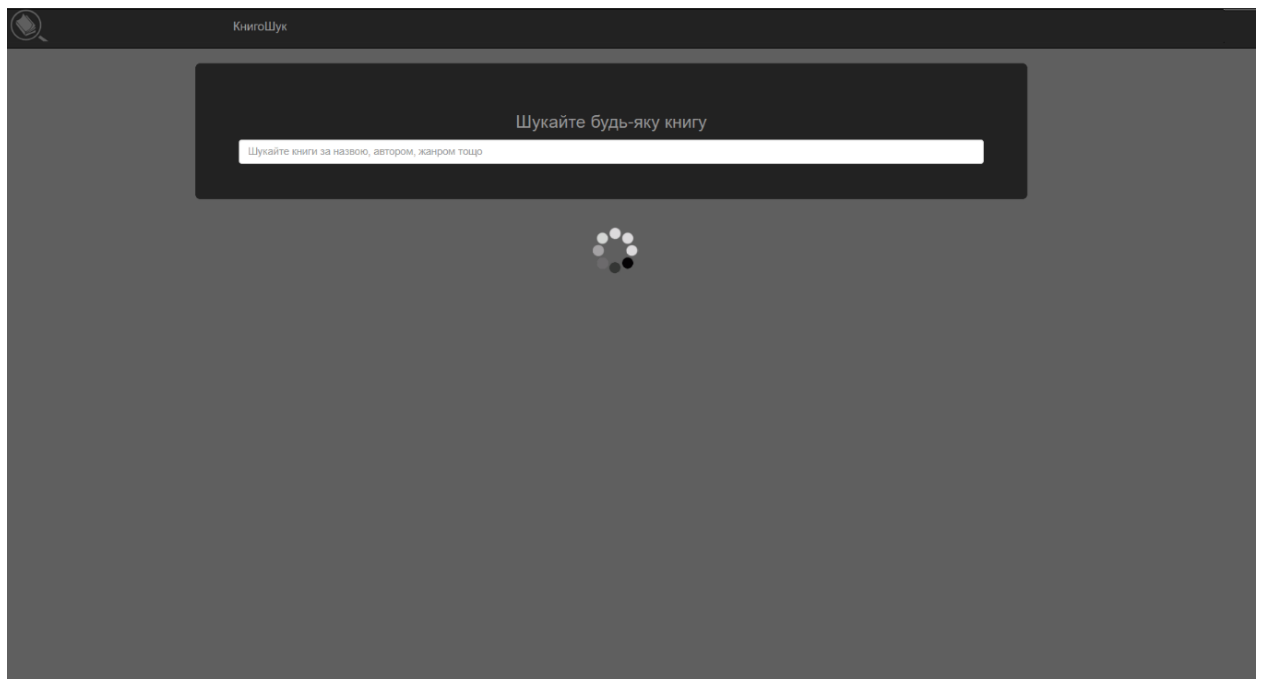


Рисунок 5.4 – Повідомлення при неможливості отримати доступ до серверів
Google Books

6 ТЕСТУВАННЯ І РЕЗУЛЬТАТИ

6.1 Опис методики проведення тестування

Тестування буде проведене за таким сценарієм:

Для користувача:

- відкриття головної сторінки;
- пошук книги за назвою;
- пошук книги за автором;
- пошук книг за категоріями;
- пошук книги за неінформативним пошуковим запитом;
- відкриття сторінки з обраною книгою;
- перегляд попереднього перегляду книги;
- відкриття головної сторінки.

Тестування буде проводитися на ПЕОМ, з характеристиками:

- процесор: intel core i5;
- RAM: 32ГБ;
- відеокарта: NVidia GeForce GTX 1080Ti;
- жорсткий диск : 1ТБ;
- система: Windows 10.

6.2 Результати використання розробленого програмного забезпечення

У таблиці 6.1 приведені результати тестування застосунку у браузерях.

Таблиця 6.1 – Чек-лист тестування функціональності для користувача.

Перевірка	Mozilla Firefox	Google Chrome	Microsoft Edge
Відкриття головної сторінки;	+	+	+
Пошук книги за назвою	+	+	+
Пошук книги за автором	+	+	+
Пошук книг за категоріями	+	+	+
Пошук книги за неінформативним пошуковим запитом	+	+	+
Відкриття сторінки з обраною книгою;	+	+	+
Перегляд попереднього перегляду книги	+	+	+
Тестування можливостей попереднього перегляду	+	+	+

Функціонування вебзастосунку було протестовано у трьох різних браузерів, помилок не виявлено. Тому, виходячи з результатів тестування можна стверджувати, що застосунок підтримує кросбраузерність й коректно виконує весь наявний функціонал.

У таблиці 6.2 буде протестовано якість верстки.

Таблиця 6.2 – Чек-лист тестування верстки:

Перевірка	Mozilla Firefox	Google Chrome	Microsoft Edge
1	2	3	4
Масштабованість	+-	+-	+-
Коректне відображення зображень	+	+	+
Кросбраузерність	+		
Зручність	+	+	+

Продовження таблиці 6.2

1	2	3	4
Безпека даних		- (немає захисту від взлому, протоколи шифрування відсутні)	
Адаптивність		-	
Технічна складова верстки			
Наявність DOCTYPE		+	
Немає помилок синтаксису CSS		+	
Кодування UTF-8		+	
Коректне відображення без JavaScript		-	

По-третє, зараз застосунок розміщено на локальному хості, та при перенесенні на сервер необхідним буде його розміщення на HTTPS-сервері, це не лише підвищить безпеку передачі даних між клієнтом та сервером, а й посприє SEO-просуванню.

Також слабкою стороною є масштабованість, при зменшенні або збільшенні сторінки більш ніж 30% на сторінках авторизації, реєстрації та діалогів, елементи можуть незначно зміщуватись, що змінює первинний дизайн.

ВИСНОВКИ

Під час виконання бакалаврської роботи було розроблено програму з графічним інтерфейсом "КнигоШук".

Під час дослідження було виявлено проблему: відсутність простих та швидких механізмів, що дозволяють шукати та рекомендувати книги без реєстрації та зайвого навантаження для користувача на різних ресурсах.

Для розв'язання проблем було створено програму, що є вебзастосунком і надає можливість з будь-якого пристрою, що має браузер та доступ до мережі Інтернет, ознайомитися з книгами, що його цікавлять та частково їх прочитати, щоб зробити свій вибір щодо книги.

Програму "КнигоШук" можна використовувати в щоденному житті та будь-якими групами людей. Результат цієї програми це розширення книг, що можуть сподобатися та бути прочитаними користувачем та віднаходження для себе літератури, що була порекомендована йому застосунком.

Проаналізовано сучасні технології у створенні програмного забезпечення. Найпопулярнішою тенденцією останніх років є мінімалістичний дизайн, який виключає нагромадження об'ємних об'єктів.

Проаналізовано наявні засоби розробки програм: порівняння об'єктно орієнтовані мови програмування C#, Java, Python, JavaScript та різноманітні середовища розробки та редакторів.

Розроблені основні модулі для швидкого аналізу, створення раціону та додавання до бази, створення графіків інтервального аналізу та створення стислого списку можливих захворювань.

Розроблено дизайн програми згідно з основними принципами мінімалістичного дизайну. Створено інтерфейси сторінок пошуку та перегляду книг.

Протестована програма на перевірку введеної інформації та придатність ПЗ в екстремальних умовах, а саме зменшення рекомендованих

вимог. В ході тестування виявлено, що програма працює коректно, крім того, було зроблено опис програми.

Отже, в ході виконання бакалаврської роботи було продемонстровано практичні навички створення та розробки вебзастосунків.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Google книги розширений пошук книг [Електронний ресурс]. – Режим доступу: https://books.google.com.ua/advanced_book_search?hl=uk.
2. BookFiner Поиск книг по интернет-магазинам Украины [Электрон. ресурс]. – Режим доступа: <https://bookfinder.com.ua/>.
3. GoodReads [Electronic resource]. – Access mode: <https://www.goodreads.com/>.
4. Jevix Top 15 JavaScript IDEs and JS Editors for Frontend Development 2022 [Electronic resource]. – Access mode: <https://jelvix.com/blog/best-javascript-ides>.
5. MDN webdocs [Electronic resource]. – Access mode: https://developer.mozilla.org/ru/docs/Learn/HTML/Introduction_to_HTML.
6. Википедия – CSS [Электрон. ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/CSS>.
7. MDN webdocs [Electronic resource]. – Access mode: https://developer.mozilla.org/ru/docs/Learn/JavaScript/First_steps/What_is_JavaScript/.
8. Вікіпедія – Прикладний програмний інтерфейс [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%B8%D0%B9_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81.
9. Google Books APIs [Electronic resource]. – Access mode: <https://developers.google.com/books/docs/v1/libraries>.
10. MDN wbdocs [Electronic resource]. – Access mode: https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Client-Server_overview?retiredLocale=uk.
11. Google Books APIs using [Electronic resource]. – Access mode:

<https://developers.google.com/books/docs/v1/using>.

12. Google Books APIs Embedded viewer [Electronic resource]. – Access mode: https://developers.google.com/books/docs/viewer/developers_guide.

13. MDN webdocs Структура документа и веб-сайта [Электрон. Ресурс]. – Режим доступа: https://developer.mozilla.org/ru/docs/Learn/HTML/Introduction_to_HTML/Document_and_website_structure.

14. CSS – подклассы и селекторы [Электрон. Ресурс]. – Режим доступа: <https://pechenek.net/web-design/css-web-design/css-selektory/>.

15. Bootstrap docs [Electronic resource]. – Access mode: <https://getbootstrap.com/docs/5.2/getting-started/introduction/>.

16. CSS Container Queries [Electronic resource]. – Access mode: https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Container_Queries.

17. Bootstrap Navbar [Electronic resource]. – Access mode: <https://getbootstrap.com/docs/4.0/components/navbar/>.

18. Bootstrap Jumbotron [Electronic resource]. – Access mode: <https://getbootstrap.com/docs/4.0/components/jumbotron/>.

19. MDN webdocs form [Electronic resource]. – Access mode: <https://developer.mozilla.org/ru/docs/Web/HTML/Element/form>.

20. Bootstrap btn [Electronic resource]. – Access mode: <https://getbootstrap.com/docs/4.0/components/buttons/>.

21. MDN webdocs <script> [Electronic resource]. – Access mode: <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/script?retiredLocale>.

22. MDN webdocs [Electronic resource]. – Access mode: <https://developer.mozilla.org/en-US/docs/Web/Guide/AJAX?retiredLocale=uk>.

23. jQuery [Electronic resource] – Access mode: <https://jquery.com/>.

24. HTML <script> crossorigin Attribute [Electronic resource]. – Access mode: https://www.w3schools.com/tags/att_script_crossorigin.asp.

25. Axios [Electronic resource]. – Access mode: API https://axios-http.com/docs/api_intro.

ДОДАТОК А
Технічне завдання

Вступ

У зв'язку з всесвітньою тенденцією до диджиталізації в усіх сферах життя, а також тим, що девайси майже завжди є під рукою – це найзручніший спосіб обирати, що прочитати.

Програмний засіб (ПЗ) призначено для пошуку, рекомендації та читання книг у онлайн режимі за допомогою браузера. Це дозволить користувачу полегшити процедуру пошуку книжок для особистого читання та розширення своєї віртуальної бібліотеки.

A.1 Назва програми

Вебзастосунок має назву «КнигоШук».

A.2 Підстави для розробки

Підставою для розробки є дипломна кваліфікаційна робота бакалавра з теми «Програмна реалізація вебзастосунку для пошуку та рекомендації книг» згідно наказу №102 від 27 квітня 2022 р.

A.3 Призначення розробки

Застосунок призначено для пошуку та рекомендації і ознайомлення користувача з книгами, що його цікавлять.

A.4 Вимоги до програми

A.4.1 Вимоги до функціональних характеристик

До застосунку висунуто наступні функціональні вимоги:

- пошук, рекомендація книг у браузері;

- отримання інформації про книги, що цікавлять користувача;
- ознайомлення та надання можливості ознайомитися з безкоштовною частиною книги користувачу, для його подальшого рішення про придбання книг.

A.4.2 Вимоги до надійності

Застосунок «КнигоШук» має вміти оброблювати ситуації некоректного пошукового запиту та повідомляти користувача про те, що його пошуковий запит не може бути оброблений застосунком. А також, тому що деякий функціонал застосунку залежить від наявності інтернет-зв'язку, то за його відсутності користувач має бути попереджений про це.

A.4.3 Умови експлуатації

Для нормального функціонування застосунку, до девайсу користувача висунуто наступні програмно-апаратні вимоги:

- не менше 4 гб оперативної пам'яті;
- не менше 100 мб вільної пам'яті девайсу;
- підтримка Google сервісів;
- наявність інтернет-зв'язку.

A.5 Стадії і етапи розробки

A.5.1 Стадії розробки

Розробка програмного продукту пройшла 4 стадії:

- аналіз вимог та потреб;
- проєктування застосунку;
- розробка застосунку;

- розробка програмної документації.

A.5.2 Етапи розробки

На першій стадії – «Аналіз вимог та потреб» виконують наступні етапи:

- збір вимог та потреб;
- аналіз та систематизування вимог та потреб.

На стадії «Проектування застосунку»:

- аналіз наявних технологій та вибір засобів для реалізації;
- вибір архітектури застосунку;
- вивчення можливостей програмної реалізації алгоритмів та

інтерфейсу користувача;

- аналіз вимог та потреб до ПЗ.

На стадії «розробка застосунку»:

- розробка алгоритмів;
- розробка інтерфейсу користувача;
- написання програмного коду;
- тестування ПЗ.

На стадії «Розробка програмної документації»:

- розробка опису програми;
- розробка супровідної документації.

A.6 Порядок контролю та приймання

Контроль та приймання застосунку здійснюються згідно з порядком приймання дипломних кваліфікаційних робіт бакалаврів. Програмний продукт має відповідати усім заявленим у розділі A.4 вимогам.

ДОДАТОК Б
Текст програми

Б.1 Файл MainPage.html

```
<!DOCTYPE html>

<html lang="uk">

<head>

    <title>КнигоШук</title>

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

    <link rel="stylesheet" type="text/css" href="css/style.css">

</head>

<body>

    <nav class="navbar navbar-inverse">

        <div class="logo-container">

            <a class="logo-link" href="#">

            </a>

        </div>

        <div class="container">

            <div class="navbar-header">

                <a class="navbar-brand"
href="index.html">КнигоШук</a>

            </div>

        </div>

    </nav>

    <div class="container">
```

```

<div class="jumbotron">

    <h3 class="text-center">Шукайте будь-яку книгу</h3>

    <form id="searchForm">

        <input type="text" name="search" id="searchText"
class="form-control" placeholder="Шукайте книги за назвою, автором, жанром
тощо">

    </form>

</div>

</div>

<div id="preloader" class="hidden">

</div>

<div class="container">

    <div class="row" id="books">

        </div>

    </div>

</div>

<script
src="https://code.jquery.com/jquery-3.2.0.min.js"
integrity="sha256-
JAW99MJVpJBGcbzEuXk4Az05s/XyDdBomFqNlM3ic+I="
crossorigin="anonymous"></script>

<script type="text/javascript"
src="https://cdnjs.cloudflare.com/ajax/libs/axios/0.15.3/axios.js"></script>

<script type="text/javascript" src="js/main.js"></script>

</body>

```

</html>

Б.2 Код файлу Book.html

```

<!DOCTYPE html>

<html>

<head>

    <title>КнигоШук</title>

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

    <link rel="stylesheet" type="text/css" href="css/style.css">

</head>

<body>

    <nav class="navbar navbar-inverse">

        <div class="logo-container">

            <a class="logo-link" href="#">

            </a>

        </div>

        <div class="container">

            <div class="navbar-header">

                <a class="navbar-brand"
href="index.html">КнигоШук</a>

            </div>

        </div>

    </nav>

    <div id="preloader" class="">

```

```

        
    </div>

    <div class="container">

        <div class="well hidden" id="book">

            </div>

            <div style="display:flex;justify-content:center;" id="book-viewer">

                </div>

            </div>

</div>

<script
    src="https://code.jquery.com/jquery-3.2.0.min.js"
    integrity="sha256-
JAW99MJVpJBGcbzEuXk4Az05s/XyDdBomFqNlM3ic+I="
    crossorigin="anonymous"></script>

<script type="text/javascript"
src="https://cdnjs.cloudflare.com/ajax/libs/axios/0.15.3/axios.js"></script>

<script type="text/javascript" src="js/main.js"></script>

<script type="text/javascript">

    getbook();

</script>

</body>

</html>

```

Б.3 Код файла “Style.css”

```
#books img , #book img{
    width: 100%;
}

.custom-spacing {
    margin-left: 20px;
}

.well , .list-group-item, .jumbotron{
    background: #222 !important;
    border: none;
    color: #999;
}

#go_butttton {
    margin-top: 20px !important;
}

#preloader{
    margin: 100px 46%;
}

@media(min-width: 960px) {
    #books .col-md-3 .well {
        height:480px;
    }
    #books img{
        height: 320px;
    }
    #book img {
        height: 500px;
    }
}

h3Text {
```

```
text-align: center;  
font-family: Helvetica, Verdana, sans-serif;  
font :italic normal 700 /1.2 cursive;  
}
```

Б.4 Код файла “Sripts.js”

```
$(document).ready(() => {
  $('#searchForm').on('submit', (e) => {
    var searchText = $('#searchText').val();
    getBooks(searchText);
    e.preventDefault();
  });
});

function getBooks(searchText) {
  $('#preloader').removeClass('hidden');
  axios
    .get('https://www.googleapis.com/books/v1/volumes?q=' + searchText)
    .then((response) => {
      console.log(response);
      var books = response.data.items;
      var output = "";
      $.each(books, (index, book) => {
        //console.log(book.accessInfo)
        output += "<div class='col-md-3'>";
        output += `<div class='well text-center'>`;
        if (book.volumeInfo.imageLinks != undefined) {
          output +=
            '<img src= ' +
            book.volumeInfo.imageLinks.thumbnail +
```

```

    ' style="margin-bottom:10px"/>';
  } else {
    output += `<img src= 'img/book.jpg' style="margin-bottom:10px"/>`;
  }

  var book_title =

    book.volumeInfo.title.length > 100

    ? book.volumeInfo.title.substring(0, 10) + '...'

    : book.volumeInfo.title;

  output +=

    `<div style="overflow-y:scroll;margin-bottom:10px" class="text-
container"><h3 class='text-center' style="height:40px">` +

    book_title +

    '</h3></div>';

  output +=

    "<a class='btn btn-danger btn-block' href='#' onclick= bookselected('" +

    book.id +

    "')>Показати</a>";

  output += '</div>';

  output += '</div>';

});

$('#preloader').addClass('hidden');

$('#books').html(output);

})

.catch((err) => {

  console.log(err);

```

```

    });
}
function bookselected(id) {
    sessionStorage.setItem('bookId', id);
    window.open('book.html', '_blank');
    return false;
}
function getbook() {
    var id = sessionStorage.getItem('bookId');
    axios
    .get('https://www.googleapis.com/books/v1/volumes/' + id)
    .then((response) => {
        $('#book').removeClass('hidden');
        console.log(response);
        var info = response.data.volumeInfo;
        var cat = info.categories;
        console.log(cat);
        var authors = info.authors;
        var output = "";
        var viewer = "";
        output += "<div class='row'>";
        output += "<div class='col-md-4'>";
        if (info.imageLinks !== undefined) {
            output += '<img src= ' + info.imageLinks.large + ' /><br/>';
        }
    })
}

```

```

output +=
    "<a class='btn btn-danger' id='go_button' href='index.html'>Повернутися до
пошуку</a>";
output += '</div>';
output += "<div class='col-md-8'>";
output += '<h2><strong>Назва: </strong>' + info.title + '</h2>';
output += '<h3>Автори: ';
for (var i = 0; i < authors.length; i++) {
    output += authors[i] + ' ';
}
output += '</h3>';
output += "<ul class='list-group'>";
output += "<li class='list-group-item'><strong>Категорії</strong></li>";
for (var j = 0; j < cat.length; j++) {
    output +=
        "<li class='list-group-item'>" + (j + 1) + '. ' + cat[j] + '</li>';
}
output += '</ul>';
output += '<p><strong>Опис:</strong>';
var dis =
    info.description.length > 1500
        ? info.description.substring(0, 1500) + '...'
        : info.description;
output += dis + '</p>';
output += '</div>';

```

```
output += '</div>';

viewer += `<iframe src="new.html" frameborder="0" border="0"
  style="border-style: none; width: 800px; height: 1100px;"></iframe>`;

$('#preload').addClass('hidden');

$('#book').html(output);

$('#preload').addClass('hidden');

$('#book-viewer').html(viewer);

})

.catch((err) => {
  console.log(err);
});
}
```

Б.5 Код файла “Recommendation.js”

```

function RecommendBooks(searchResult) {
var euclidean_score = function(dataset,point1,point2){
    var existpoint1point2 = {};
    for(var key in dataset[point1]){
        if(key in dataset[point2]){
            existpoint1point2[key] = 1
        }
    }
    if(len(existpoint1point2) == 0) return 0;
    var sum_of_euclidean_dist = [];
    for(item in dataset[point1]){
        if(item in dataset[point2]){
            sum_of_euclidean_dist.push(Math.pow(dataset[point1][item]-
dataset[point2][item],2));
        }
    }
    var sum=0;
    for(var i=0;i<sum_of_euclidean_dist.length;i++){
        sum +=sum_of_euclidean_dist[i]; //calculate the sum of the
euclidean
    }
    var sum_sqrt = 1/(1 +Math.sqrt(sum));
    return sum_sqrt;
}
var len = function(obj){
    var len=0;
    for(var i in obj){
        len++
    }
}

```

```

    return len;
}
var pearson_correlation = function(dataset,point1,point2){
var existpoint1point2 = {};
for(item in dataset[point1]){
    if(item in dataset[point2]){
        existpoint1point2[item] = 1
    }
}
var num_existence = len(existpoint1point2);
if(num_existence ==0) return 0;
var point1_sum=0,
    point2_sum=0,
    point1_sq_sum=0,
    point2_sq_sum=0,
    prod_point1point2 = 0;
for(var item in existpoint1point2){
    point1_sum += dataset[point1][item];
    point2_sum += dataset[point2][item];
point1_sq_sum += Math.pow(dataset[point1][item],2);
    point2_sq_sum += Math.pow(dataset[point2][item],2);
prod_point1point2 += dataset[point1][item]*dataset[point2][item];
}
var      numerator      =prod_point1point2      -
(point1_sum*point2_sum/num_existence);
var st1 = point1_sq_sum - Math.pow(point1_sum,2)/num_existence;
    var st2 = point2_sq_sum -Math.pow(point2_sum,2)/num_existence;
var denominator = Math.sqrt(st1*st2);
if(denominator ==0) return 0;
    else {

```

```

        var val = numerator / denominator;
        return val;
    }

}

var similar_user = function(dataset,book,num_user,distance){
var scores=[];
for(var others in dataset){
    if(others != book && typeof(dataset[others])!="function"){
        var val = distance(dataset,book,others)
        var p = others
        scores.push({val:val,p:p});
    }
}
scores.sort(function(a,b){
    return b.val < a.val ? -1 : b.val > a.val ? 1 : b.val >= a.val ? 0 : NaN;
});
var score=[];
for(var i = 0;i<num_user;i++){
    score.push(scores[i]);
}
return score;
}

var recommendation_eng = function(dataset,book,distance){
    var totals = {
        setDefault:function(props,value){
            if(!this[props]){
                this[props] = 0;
            }
        }
    }
    this[props] += value;

```

```

    }
  },
  simsum = {
    setDefault:function(props,value){
      if(!this[props]){
        this[props] = 0;
      }
      this[props] += value;
    }
  },
  rank_lst = [];
for(var other in dataset){
  if(other === book) continue;
  var similar = distance(dataset,book,other);

  if(similar <=0) continue;
  for(var item in dataset[other]){
    if(!(item in dataset[book]) || (dataset[book][item]==0)){
      totals.setDefault(item,dataset[other][item]*similar);
      simsum.setDefault(item,similar);
    }
  }
}

for(var item in totals){
  if(typeof totals[item] !== "function"){

    var val = totals[item] / simsum[item];
    rank_lst.push({val:val,items:item});
  }
}

```

```
    }  
  }  
  rank_lst.sort(function(a,b){  
    return b.val < a.val ? -1 : b.val > a.val ?  
      1 : b.val >= a.val ? 0 : NaN;  
  });  
  var recommend = [];  
  for(var i in rank_lst){  
    recommend.push(rank_lst[i].items);  
  }  
  return [rank_lst,recommend];  
}}
```

ДОДАТОК В
Слайди презентації

Розробка вебзастосунку для пошуку та рекомендації книг

Дипломна робота

Виконав:
ст. гр. КНТ-128

Р. В. Сокол

Керівник:
к. т. н. доцент

М. М. Касьян

Рисунок В.1 – Слайд 1

Мета роботи

- Розробка вебзастосунку для пошуку книг за конкретними назвами, авторами, жанрами та наявністю рекомендованої літератури

Об'єкт дослідження

- Запит користувача на отримання інформації про книгу, що його цікавить

Предмет дослідження

- Вебзастосунок пошуку та рекомендації книг «КнигоШук»

2

Рисунок В.2 – Слайд 2

Постановка завдання

Завдання дипломної роботи, це програмна реалізація вебзастосунку пошуку і рекомендації книг.

Вимоги до застосунку :

- пошук книг за назвою, автором, жанром або категорією;
- відображення результатів пошукового запиту у вигляді списку коротких відомостей про книги;
- надання можливостей детального ознайомлення з книгою;
- реалізація вбудованої можливості перегляду книги у на сторінці застосунку.

3

Рисунок В.3 – Слайд 3

Порівняння існуючих аналогів



Рисунок 1 – Google пошук



Рисунок 2 – BookFinder



Рисунок 3 – Застосунок GoodReads

4

Рисунок В.4 – Слайд 4

Порівняння існуючих аналогів

Критерій	Google пошук	BookFinder	GoodReads	КнигоШук
Можливість формувати гнучкий пошуковий запит	+	-	-	+
Мультимовність	+	-	-	-
Рекомендація книг користувачу	+	±	+	+
Попередній перегляд книг	-	-	-	+
Регістрація користувача	-	-	+	-
Зрозумілий, не перенасичений інтерфейс	-	+	±	+
Структурованість інформації про книги	+	-	±	+

5

Рисунок В.5 – Слайд 5

Аналіз існуючих технологій Порівняння фреймворків

Критерій	Webstorm	Atom	Microsoft Visual Studio
Багатоплатформність	+	+	+
Підтримка репозиторіїв	+	+	+
Автоматичне доповнення коду	+	+	+
Інтегрований відладчик	+	+	+
Плагіни, що надаються середовищем	+	+	++
Безкоштовна версія	+	+	+
Багатоплатформність	+	+	+

6

Рисунок В.6 – Слайд 6

Архітектура застосунку

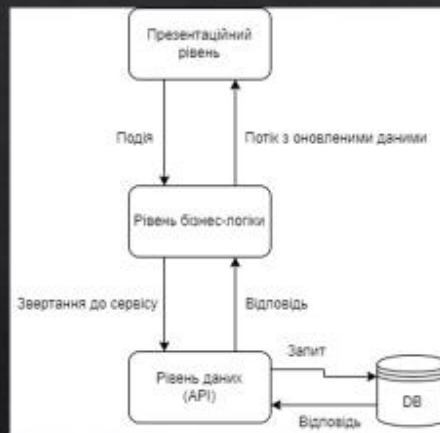


Рисунок 4 - Діаграма обраної архітектури

Рисунок В.7 – Слайд 7

Функціональні можливості розробленої програми

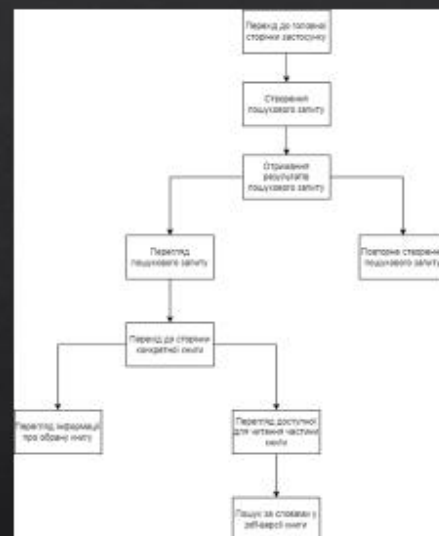


Рисунок 5 – Функціональні можливості розробленої програми

Рисунок В.8 – Слайд 8

Приклад інтерфейсу застосунку



Рисунок 9 – Головна сторінка застосунку



Рисунок 10 – Результат пошукового запиту

9

Рисунок В.9 – Слайд 9

Приклад інтерфейсу застосунку



Рисунок 11 – Сторінка пошуку при відсутньому інтернет-зв'язку



Рисунок 12 – Сторінка обраної книги

10

Рисунок В.10 – Слайд 10

Висновки

У результаті дипломної роботи:

- ◆ Було розглянуто проблемну область застосунку, проаналізовано існуючі аналоги програм;
- ◆ Визначено вимоги до застосунку, проаналізовано існуючі технології, обрано архітектуру;
- ◆ Розроблено та протестовано програмний продукт “КнигоШук”, що відповідає заявленим до нього вимогам, та стане зручним засобом вести облік нападів.

11

Рисунок В.11 – Слайд 11

Дякую за увагу!

12

Рисунок В.12 – Слайд 12