JAGRAN LAKECITY UNIVERSITY

igniting minds; changing lives

Faculty of Science and Technology (fast)
Jagran School of Artificial intelligence (JSAI)

Program : BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING WITH SPECIALIZATION IN ARTIFICIAL INTELLIGENCE

Session 2025-26

semester-3

Course Name : Web programming
Course Code : STDN1104
Project report

SUBMITTED BY :

Name : Shefali khan
JLU ID : jlu09263

SUBMITTED TO :

Dr.Dileep Singh
(HOD)

Book Recommender Web Application

1-Introduction

The InfiniteShelf Book Recommender is a JavaScript-based interactive web application that allows users to search for books, receive random recommendations, save their preferences, and manage favorites. The application provides dynamic content updates through DOM manipulation and fetches book data from the OpenLibrary Public API using asynchronous JavaScript (Fetch API & async/await).

The system also stores user preferences, history, and favorite books using LocalStorage, ensuring a personalized and persistent browsing experience

2-Objectives

- To build an interactive web application using modern JavaScript.
- To demonstrate DOM manipulation, event handling, and form validation.
- To integrate a public API with asynchronous programming.
- To implement browser data storage using localStorage.
- To enable users to search books and manage favorites.
- To encourage teamwork and collaborative project development

3- Features Implemented

Requirement	Implementation
Variables, loops, conditionals	Used extensively in API processing & validation
Functions & arrow functions	Both used in various operations
DOM manipulation	querySelector, appendChild, innerHTML, dynamic cards
Event Handling	Search, filter change, form submit, random recommendation, show favorites
Form validation	Name & genre required before saving
Objects & Arrays	Book objects stored, favorites array managed

4-Advanced (Module 4)

Requirement	Implementation
ES6+ (let, const, template strings, destructuring)	Applied throughout the JavaScript
Array methods	map, filter, reduce used for books & search stats
Async Programming	Fetch API used to call OpenLibrary API
Async/Await + try/catch	Implemented in book search function
Local/Session Storage	Stores preferences, favorites, search history

5-System Architecture

Frontend Technologies Used

- HTML5 (UI structure)
- CSS3 (Styling/Responsive design)
- JavaScript ES6+ (Logic, event handling, API integration)

External Service

- OpenLibrary API for real-time book data
  Storage
- LocalStorage: preferences, favorites, search history

6-Working Flow Diagram

User Opens App → Form Input (Name & Genre) → Save Preferences

↓

Search Books → Fetch from API → Display Book Cards

↓

Add to Favorites → Save to localStorage → Show Favorites

↓

Random Recommendations → Based on saved genre

↓

Search History → Calculate Top Searches using reduce()

7-Screens & Components

Home Page:

- App introduction and genres display

Main Application Page:

- Preferences form
- Search bar & filters
- Recommendations section
- Favorite books section
- Statistics panel (top search history)

8. Public API Used

API	Purpose
https://openlibrary.org/search.json?q=	Fetch search results for books based on keywords

9. LocalStorage Data Structure

Key	Stored Value
prefs	name & genre
favs	favorite book list
history	search history

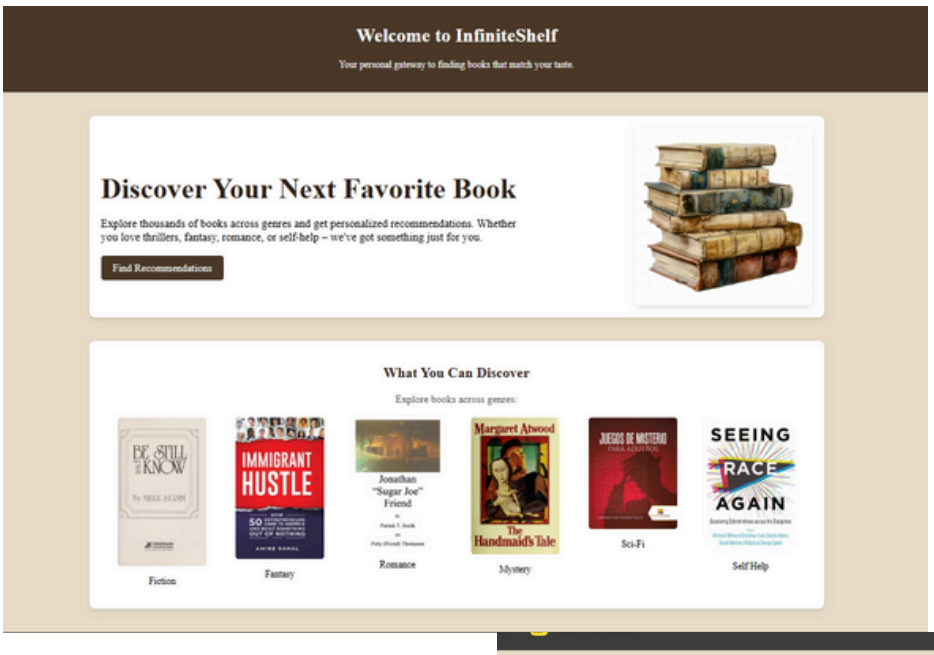
10. Testing & Validation

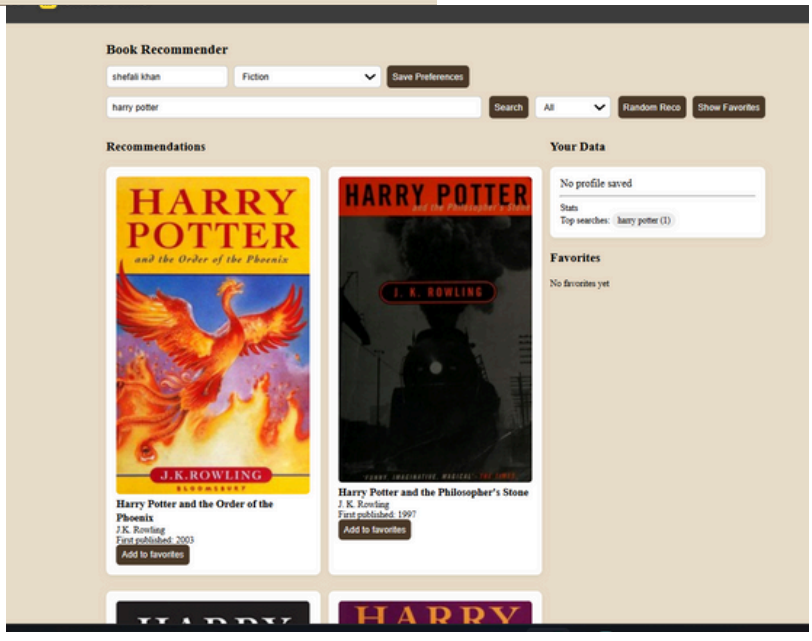
Scenario	Result
Empty form submit	Shows alert validation
Duplicate favorite	Prevents duplicates
Search without keyword	Alerts user
API failure	Handled safely using try/catch

11. Future Enhancements

- User login using Firebase authentication
- UI redesign with animations & dark mode
- Display book details in modal popup
- Book recommendation based on machine learning preferences

12. Screenshots





Conclusion

The InfiniteShelf Book Recommender demonstrates complete understanding of JavaScript DOM, events, API interaction, async programming, and browser storage. The application successfully provides a personalized experience and meets all project requirements.
This project helped us improve our skills in JavaScript development, teamwork, UI/UX considerations, and real-time API integration.