**Book Finder & Personal Library Manager**

Project submitted to the

SRM University – AP, Andhra Pradesh

for the partial fulfillment of the requirements to award the degree of

**Bachelor of Technology**

**In**

**Computer Science and Engineering**

**School of Engineering and Sciences**

Submitted by

**R. Niharika - AP23110011226**

**Shaik. Sadiya parvin - AP23110011235**

**A picture containing text

Description automatically generated**

Under the Guidance of

**Yatharth Shahrawat**

**Department of Computer Science and Engineering**

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh – 522 240

[ December, 2025]

**Introduction:-**

Step into a new era of digital reading and personal library management with the **Book Finder & Online Library System**, crafted with the power and flexibility of **React.js**. Designed for modern readers and book enthusiasts, this application transforms how you discover, organize, and interact with your favorite books.

With seamless integration of the **Google Books API**, users can explore a vast universe of literature—searching by title, author, or keyword—while enjoying an intuitive and interactive interface. Whether you're managing your personal collection, saving favourites, or tracking reading progress, this app ensures a smooth and enjoyable experience.

On the technology side, the project demonstrates key full-stack concepts:

* Frontend Interaction using reusable React components
* State Management and authentication using React Context API
* Backend Data Handling through RESTful API endpoints
* Database Operations using the JSON Server (db.json)
* Form validations, price calculations, and real-time feedback using Toast notifications

Powered by **React.js** on the frontend and **JSON Server** as a lightweight mock backend, the system enables full CRUD functionalities, ensuring data persistence for your digital library. Whether accessing it on a desktop, tablet, or smartphone, the responsive design ensures a consistent and delightful experience across all devices.

Say goodbye to scattered book lists and embrace a digital transformation in reading and library management. Your next favorite book awaits—welcome to the Book Finder & Personal Library Manager.

**Scenario-Based Intro :-**

Imagine a student named *Ananya* who is preparing for an important exam. She needs books on data structures, algorithms, and problem-solving. She searches online, but most websites show limited book details, outdated descriptions, incomplete metadata, or irrelevant results. Some sites redirect her to external stores, while others don’t allow her to save books she wants to read later.

Frustrated, she ends up keeping dozens of browser tabs open, taking screenshots of book covers, and manually writing down titles to remember them.

This is not just her problem—readers everywhere face similar challenges.  
Without a centralized digital reading assistant, users struggle with:

* Scattered reading lists
* No way to save or organize favourite books
* Inability to track reading progress
* No simple interface for book discovery

To solve these real-world issues, the **Book Finder & Personal Library Manager** was built.

This system acts as a complete digital reading assistant.  
A user can:

* Search for books using **Google Books API**
* View detailed information (authors, description, page count, publisher, categories)
* Save favourite titles to their personal library
* Add notes or remarks
* Mark books as favourites
* Update reading status — *Not Started, Reading, Completed*
* Delete books anytime

Every book stored in the user’s library is managed using **JSON Server**, offering smooth and reliable CRUD operations.

For readers, this system provides a beautifully organized digital library.  
For developers, it demonstrates how React + API integration + JSON Server create a modern, interactive, real-world application.

By simulating a true book discovery and personal library workflow, this project shows how modern technology can turn an ordinary book search into a smart, automated, highly personalized reading experience.

**Target Audience :-**

**1. Students and Learners**

These users frequently search for academic books, reference materials, and learning resources.  
They look for:

* Accurate book metadata
* Easy search and filtering
* Ability to save important study books
* Notes and status management

The system helps them maintain organized study material.

**2. Casual Readers and Book Lovers**

People who read for pleasure or self-growth.  
They expect:

* Easy book search
* Clean book details
* Personal digital bookshelf
* Favourites and notes

The system acts like a personal “My Library” space.

**3. Researchers and Professionals**

They require quick access to titles for work-related research.  
They need:

* Reliable search results
* Detailed book descriptions
* Ability to save and revisit important titles

This system supports efficient research workflows.

**4. Reviewers, Bloggers, and Content Creators**

People who analyze or review books regularly.  
They use:

* Notes feature
* Reading status
* Saved books list

This helps them organize content ideas.

**Project Goals And Objectives :-**

The **Book Finder & Personal Library System** is designed with the following key goals and objectives:

**Provide a seamless book searching experience**

Allow users to discover books effortlessly by integrating the **Google Books API** for real-time, accurate search results.

**Enable users to build and manage a personal digital library**

Users can save books they like into their own library for quick access and long-term organization.

**Offer complete CRUD functionality**

Support all essential operations:

* **Create** (add a book)
* **Read** (view saved books)
* **Update** (edit notes, status, favourites)
* **Delete** (remove from library)

**Ensure reliable data persistence**

Use **JSON Server backend** to store library data securely and maintain user entries across sessions.

**Allow reading status tracking and note management**

Users can mark books as *Not Started*, *Reading*, *Completed*, and write personalized notes or remarks.

**Provide a modern, responsive UI**

Design a clean, intuitive, and mobile-friendly interface using **React + Tailwind CSS**.

**Demonstrate real-world API integration**

Use **Axios** for smooth communication between React components, Google Books API, and the backend server.

**Help users organize their reading experience**

Enable sorting, filtering, marking favourites, and browsing saved collections with ease.

**Modern Tech Stack**

**1. Frontend Technologies**

**React.js**

Component-based UI for building an interactive, dynamic web interface.

**React Router**

For smooth client-side navigation between pages.

**Tailwind CSS**

Modern utility-first CSS for responsive, elegant UI styling.

**React Icons**

For expressive icons across the interface.

**React Toastify**

For real-time notifications such as:

* Book added
* Book deleted
* Status updated

**Axios**

Used for API communication with Google Books and JSON Server.

**2. Backend Technologies**

**JSON Server**

Lightweight REST API backend for data storage and CRUD operations.

**Book Database (db.json)**

Stores:

* Saved books
* Notes
* Reading status
* Favourites

**3. State & Authentication**

**React Context API**

Manages global state (library, user preferences, notifications).

**LocalStorage**

Ensures persistent library data (optional for user sessions).

**4. Development Tools**

* **Node.js** – environment for running React + JSON Server
* **Git / GitHub** – version control
* **VS Code** – primary development IDE

**Key Features :-**

**1. Book Search System**

* Search books by **title, author, or keyword**
* Fetches live data from **Google Books API**
* Displays results with:
  + Cover image
  + Title
  + Author
  + Category
  + Page count
  + Description

**2. Book Details Preview**

Each book includes:

* High-quality cover image
* Full description
* Authors
* Publisher details
* Published date
* Categories
* Page count

**3. Add to Personal Library (Create Operation)**

Users can save books to their library with one click.

Data stored in JSON Server includes:

* Book details
* Notes
* Reading status
* Favourite toggle

**4. Reading Status Management**

Statuses include:

* Not Started
* Reading
* Completed

Users can update them anytime.

**5. Notes & Annotations**

Users can write:

* Personal thoughts
* Key points
* Quotes
* Study notes

Notes are saved in the backend.

**6. Favourite Books**

Users can bookmark any book as a **favourite** for quick access.

**7. Update Book Information (Update Operation)**

Users can update:

* Notes
* Reading status
* Favourite toggle

**8. Delete Book (Delete Operation)**

Remove books from My Library instantly—stored data is deleted from JSON Server.

**9. User Dashboard**

Displays:

* Total saved books
* Favourite count
* Reading progress indicators

Easy navigation to:

* My Library
* Favourites
* Search Books

**10. Modern UI/UX**

* Built using Tailwind
* Mobile responsive
* Clean grid layout for book cards
* Consistent color palette
* Floating search bar
* Smooth button animations

**11. Notification System**

Real-time toasts for:

* Book added
* Book updated
* Book removed
* Errors or API failures

**12. REST API Integration**

Axios used for:

* Fetching Google Books API data
* Performing CRUD with JSON Server
  + POST: Add book
  + GET: View library
  + PATCH: Update book
  + DELETE: Remove book

All data is stored reliably in **db.json**.

**Re-Requisites :-**

**1. Software Requirements**

* **Node.js (Latest LTS)**
* **npm or yarn**
* **VS Code**
* **Google Chrome** (recommended)

**2. Technical Knowledge**

* JavaScript ES6+
* Basics of React (components, hooks, routing)
* REST API concept
* JSON format
* Basic HTML & CSS
* Tailwind CSS knowledge (optional but helpful)

**3. Backend Tools**

* JSON Server
* Axios

**4. Required Dependencies**

Must be installed using npm:

react

react-dom

react-router-dom

axios

react-icons

react-toastify

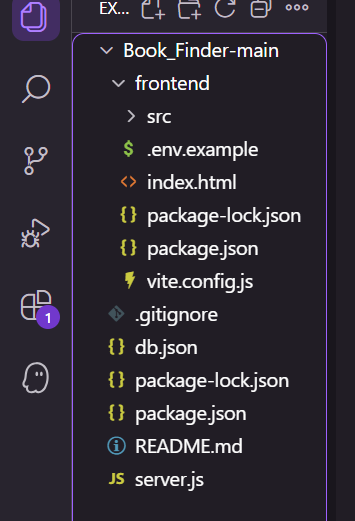
tailwindcss

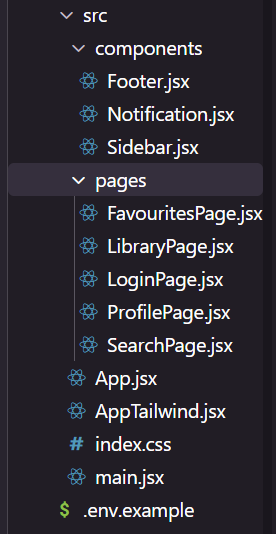
**5. Hardware Requirements**

Minimum:

* 4 GB RAM (8 GB recommended)
* Internet connection
* At least 1 GB free storage

**Project Structure :-**





**Project Flow:-**

**Project demo:**

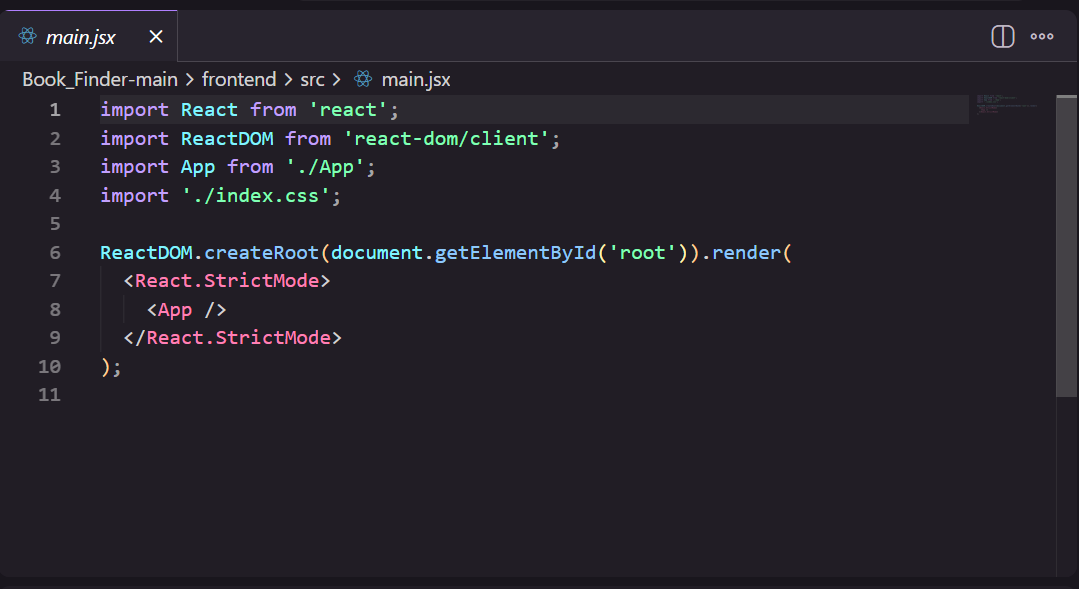
Before starting to work on this project, let’s see the demo.

**Demo link:** <https://drive.google.com/file/d/1mizwQsyAozRrEOIcnFIcRBoY_VHIy0c/view?usp=sharing>

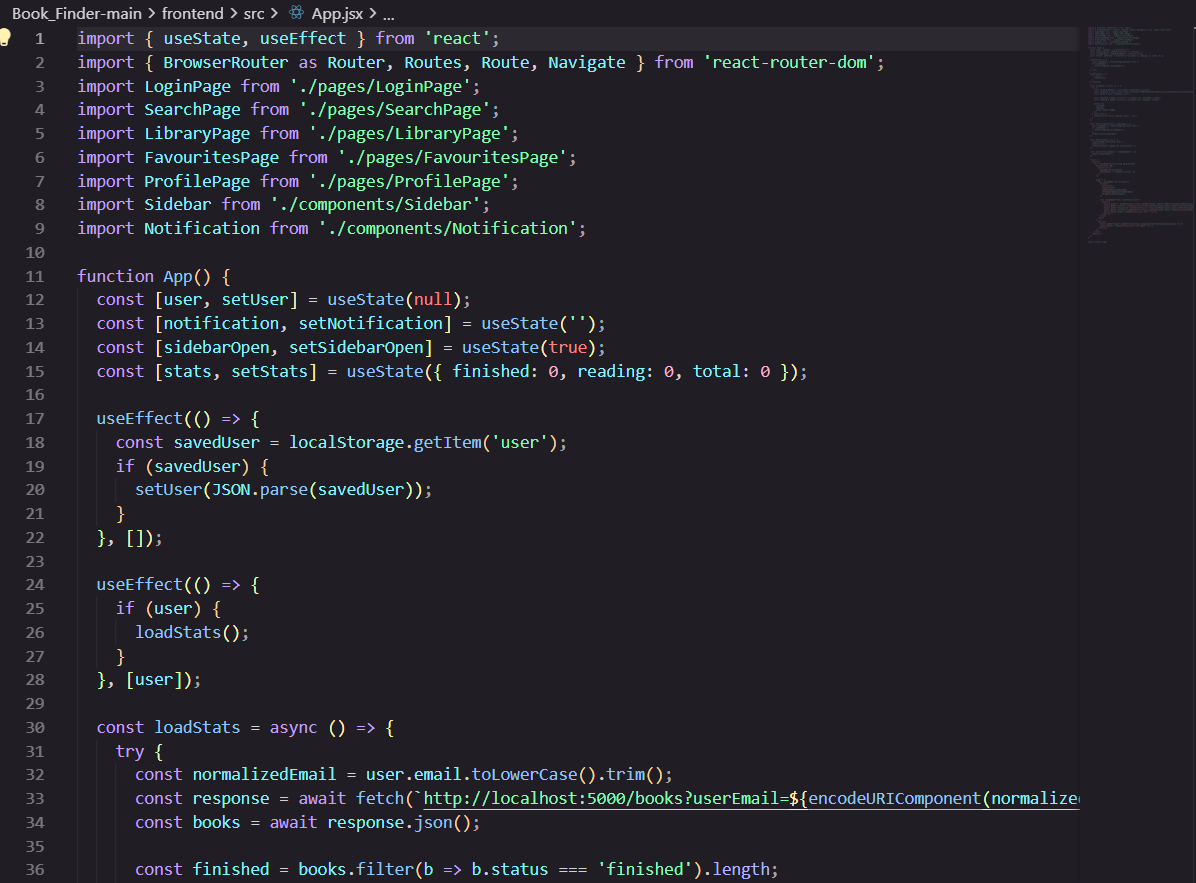
**Use the code Explanation link in:**

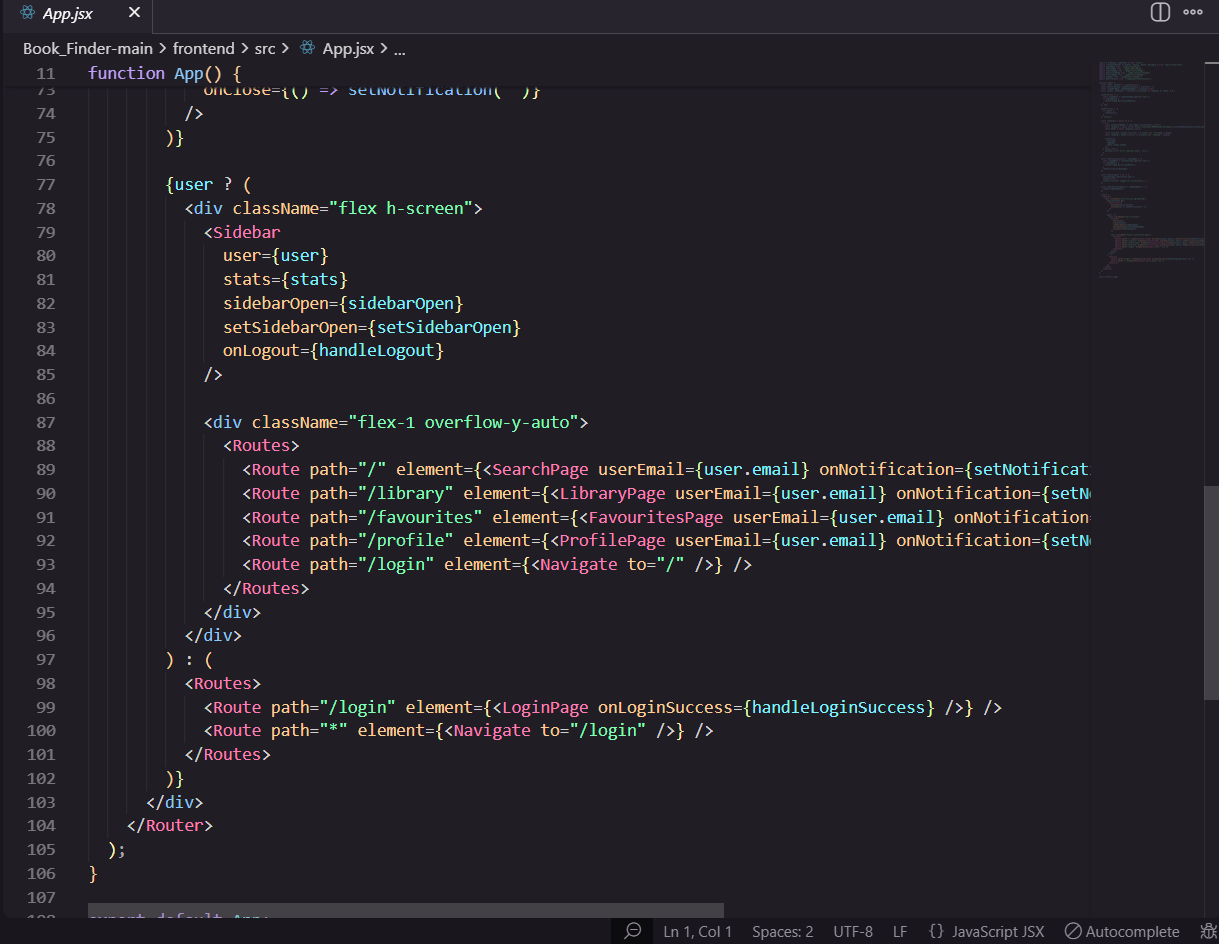
<https://drive.google.com/file/d/1HHIf7ijrgK-jffUuiMOJj162FotPc8D8/view?usp=sharing>

**Main Component**



**App.jsx Component**





**Code Description**

This React code is the main application file.

It controls how the app behaves based on whether the user is logged in or not.

**✔ 1. Manages User Login**

Checks if a user is already saved in localStorage.

If the user exists, it loads their details into the app.

If the user logs out, it removes the saved user.

**✔ 2. Loads Book Statistics**

After the user logs in, it fetches:

how many books they finished,

how many they are currently reading,

the total number of books.

These statistics are shown inside the sidebar.

**✔ 3. Shows Notifications**

There is a notification system that displays messages like:

login successful

logout successful

updates successful

**✔ 4. Handles Sidebar Toggle**

It keeps track of whether the sidebar is open or close.

**✔ 5. Controls All Page Routing**

If the user is logged in, they can access:

Search Page

Library Page

Favourites Page

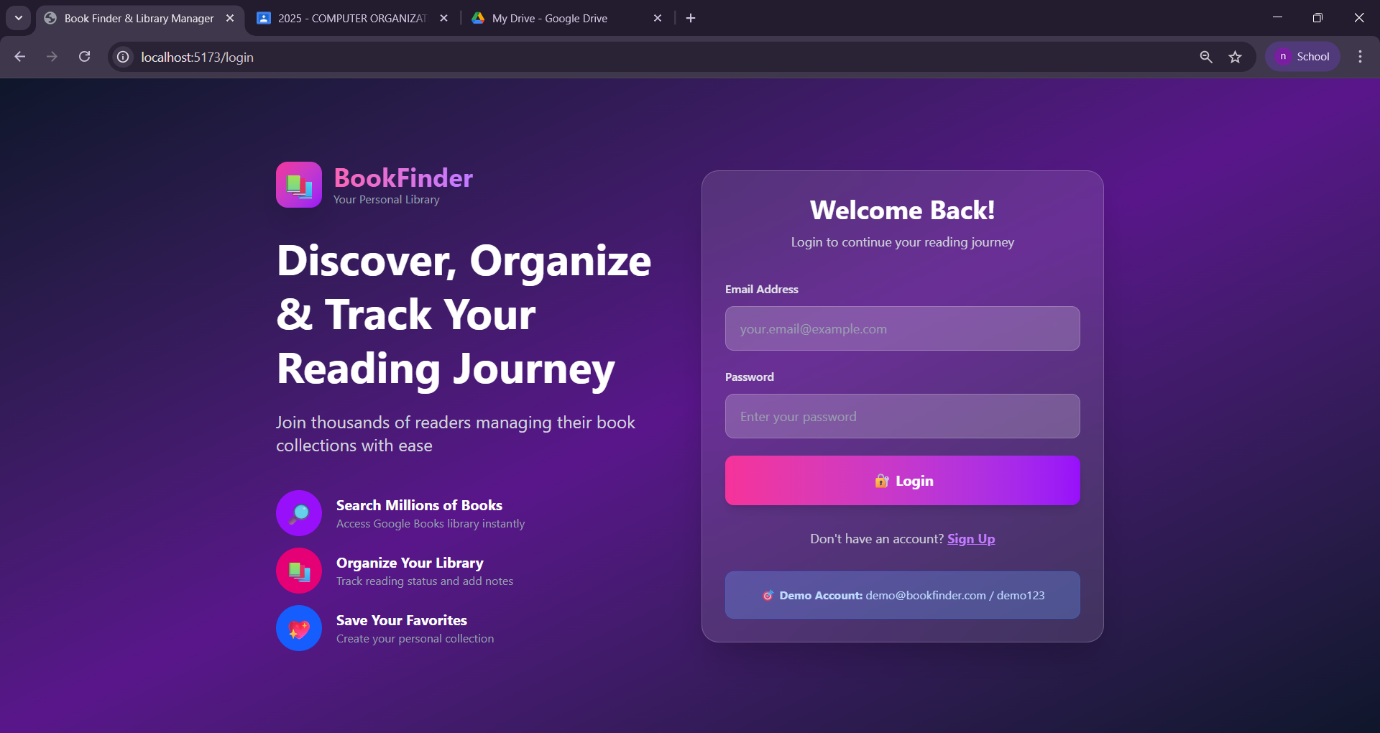
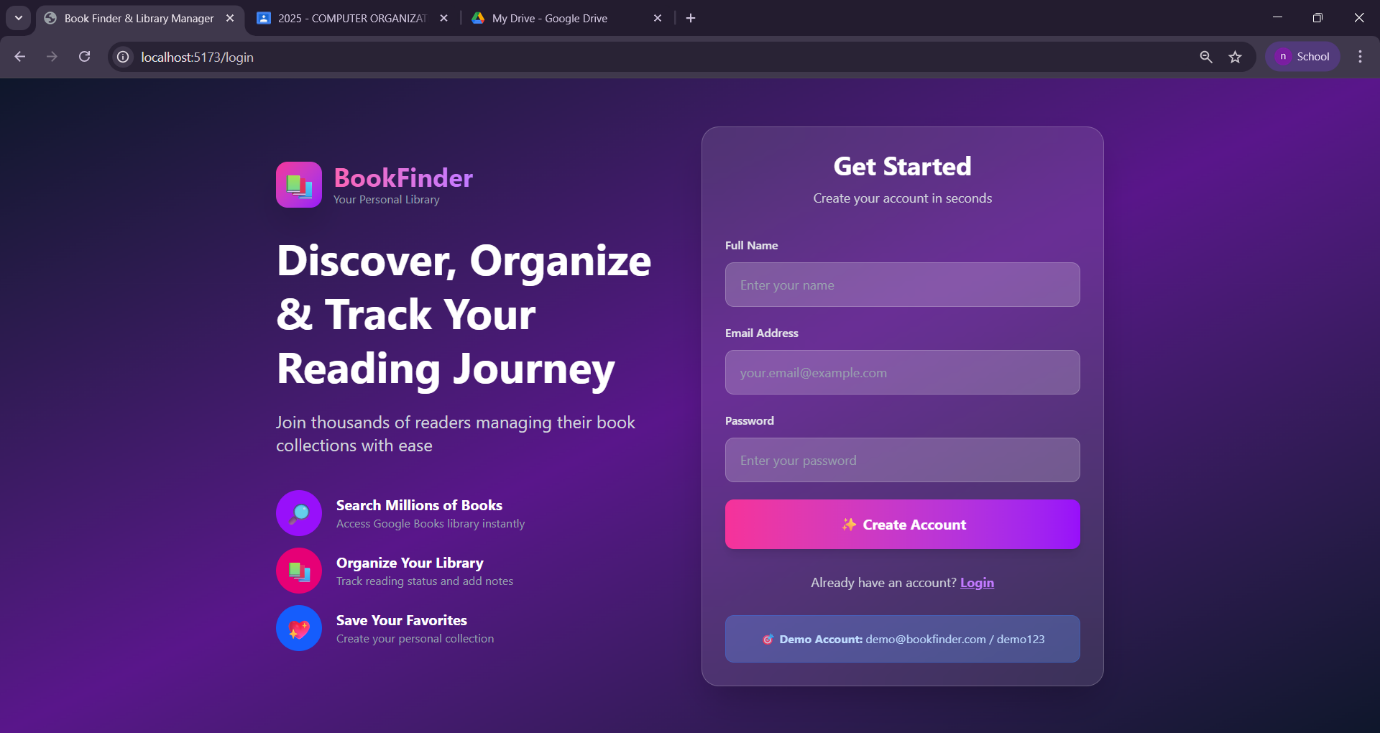
Profile Page

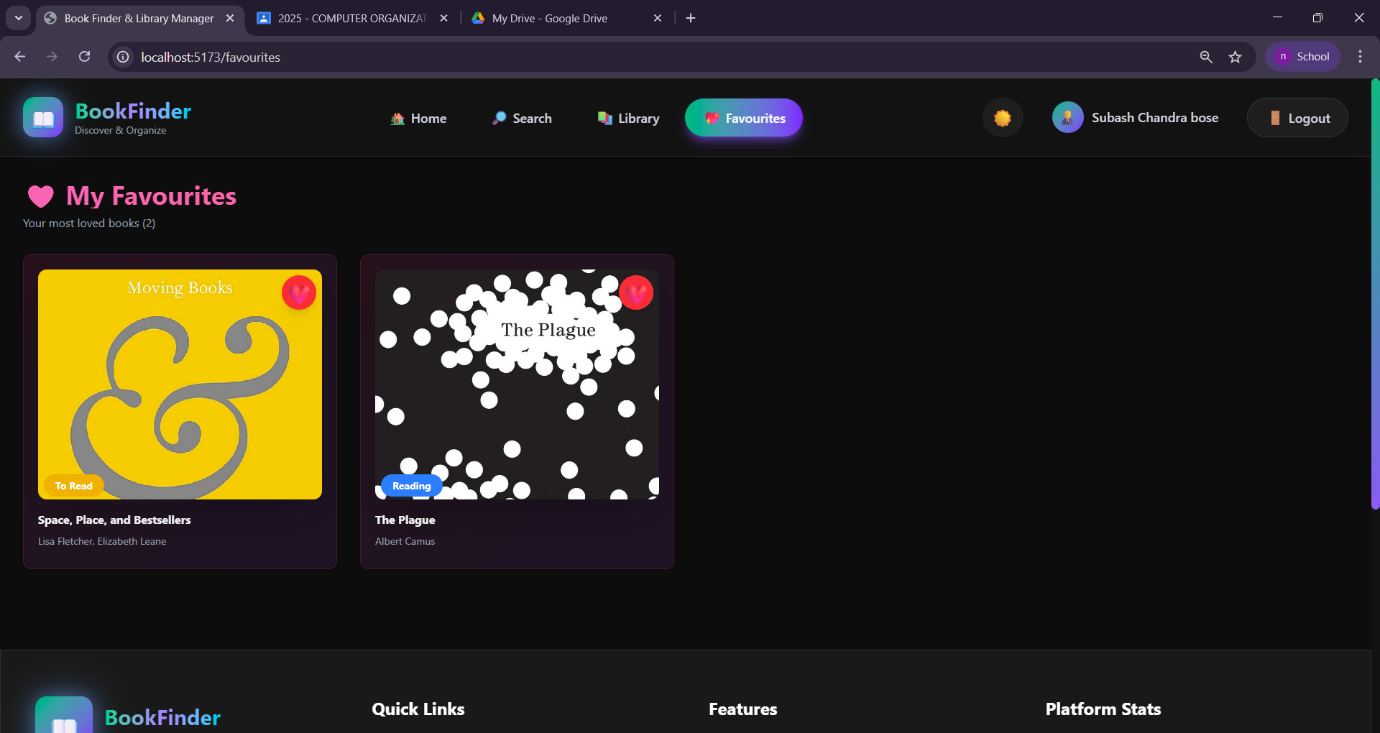
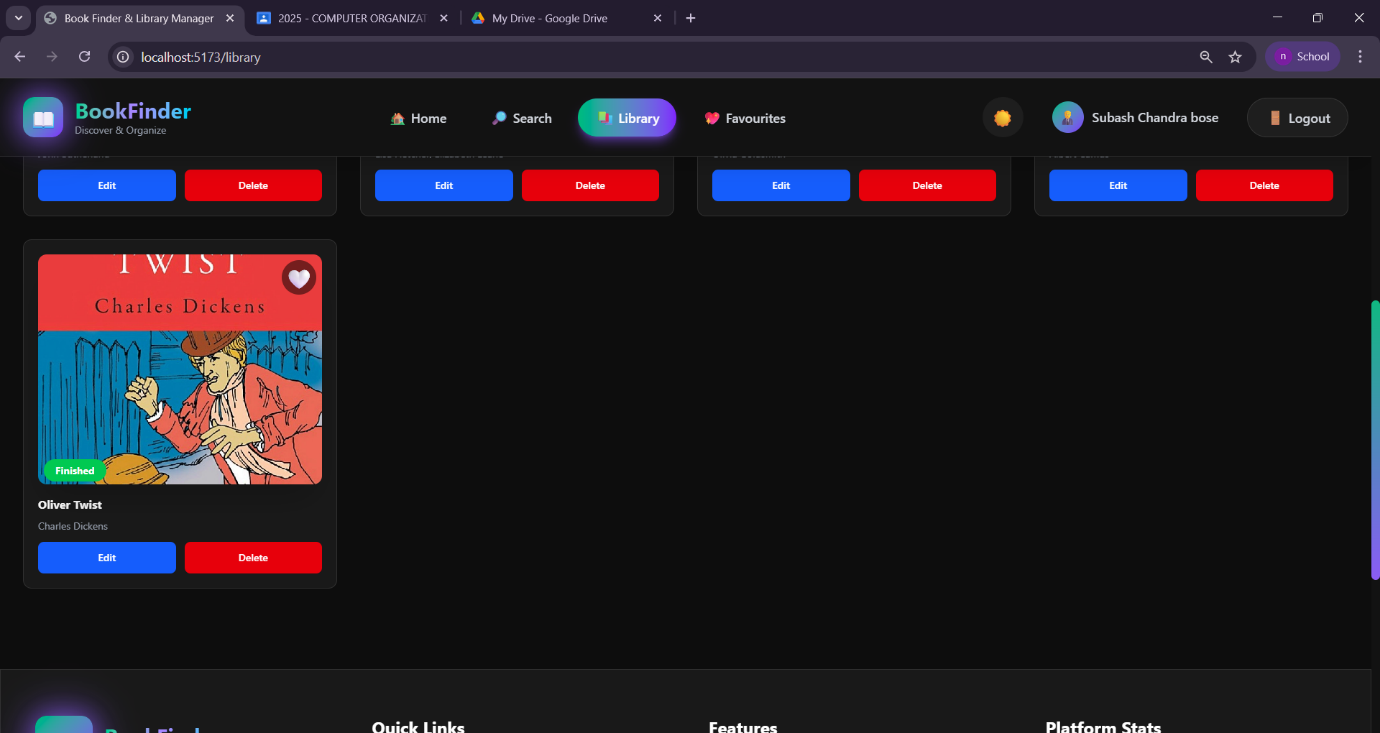
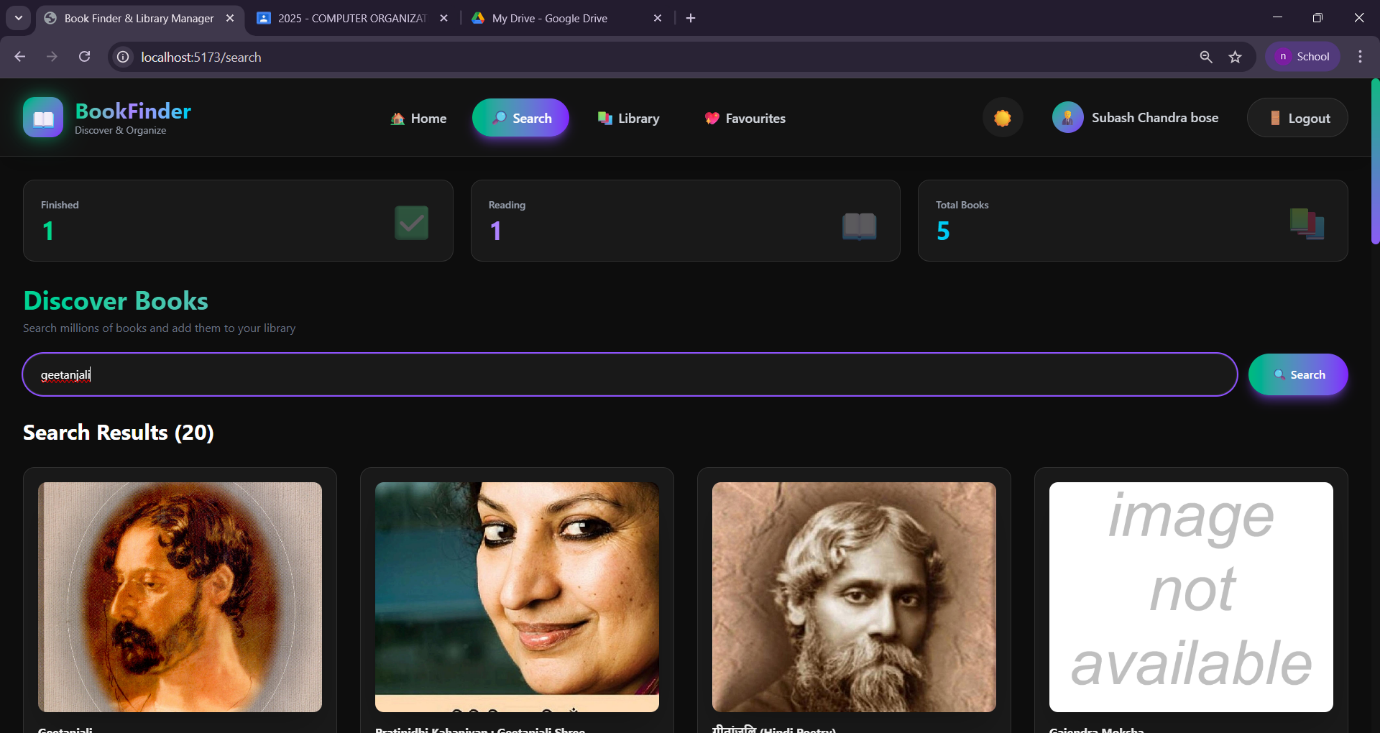
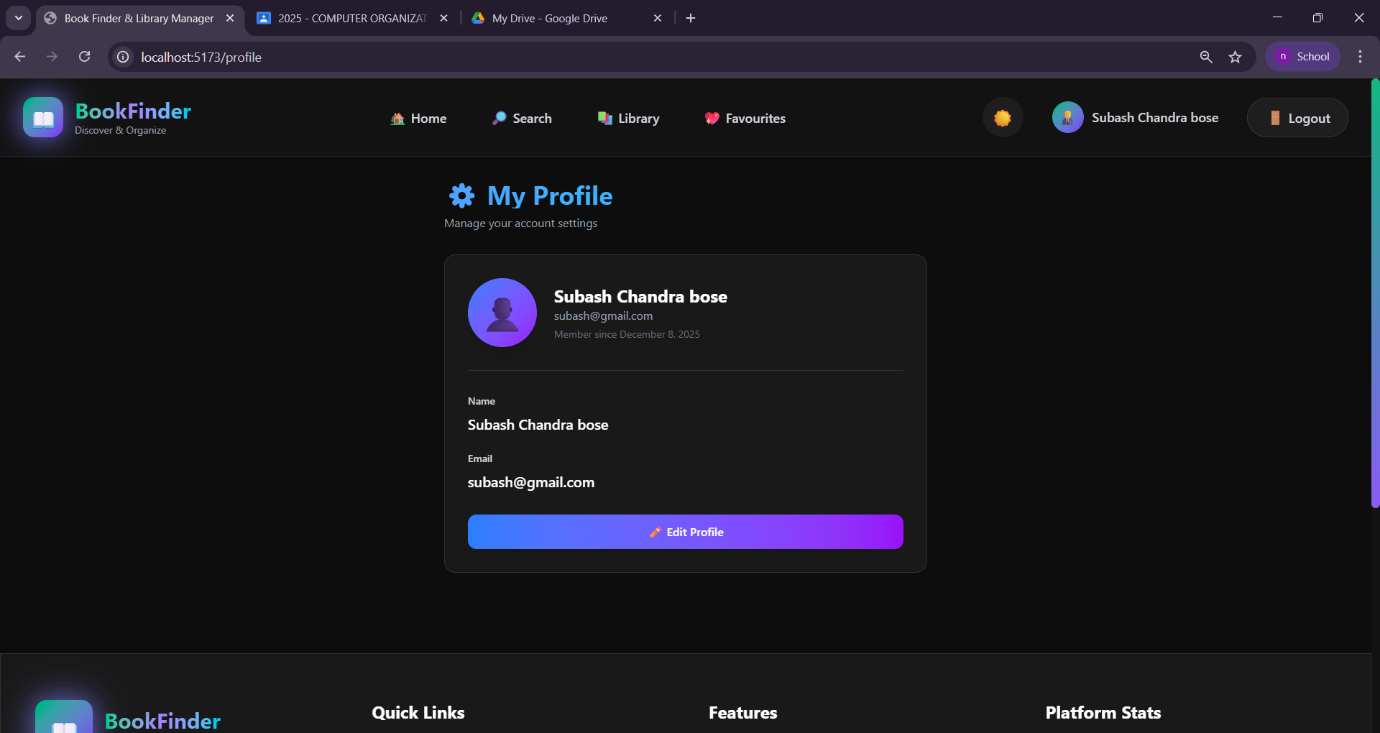
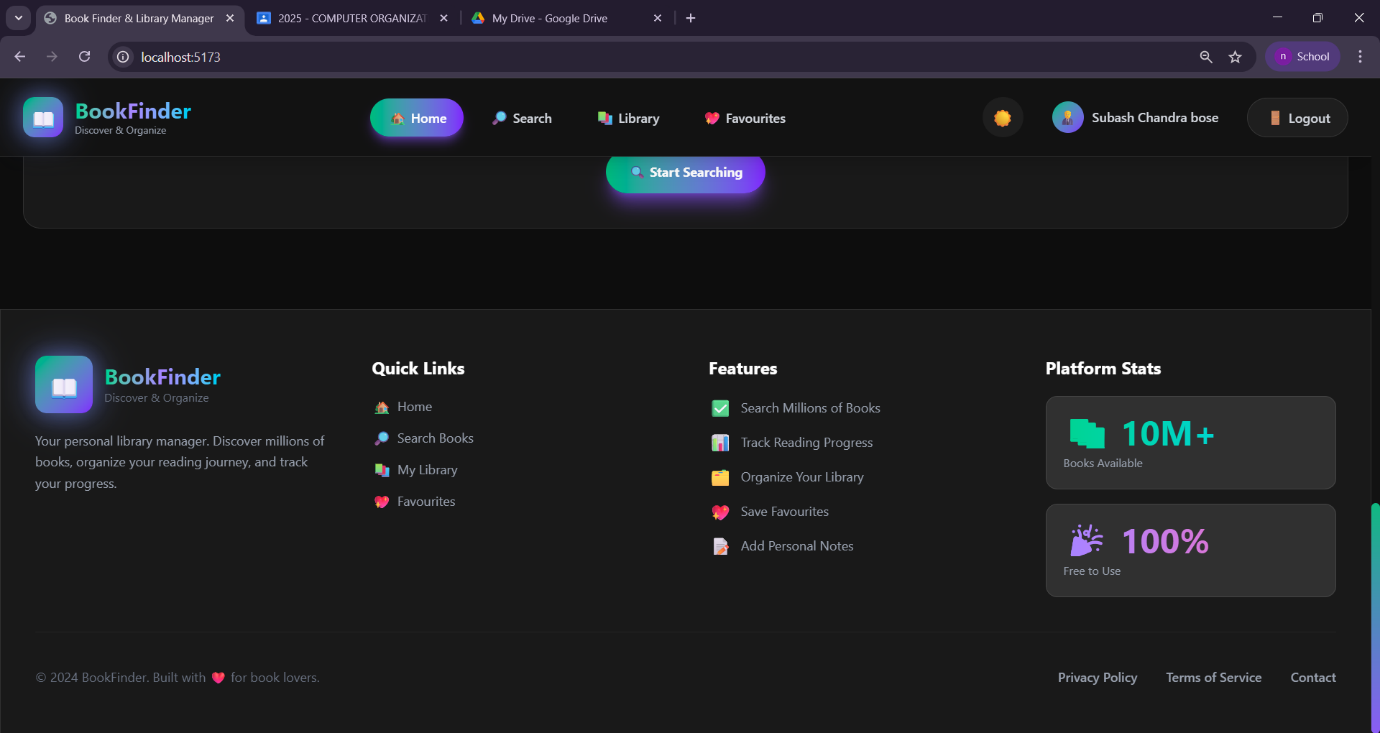
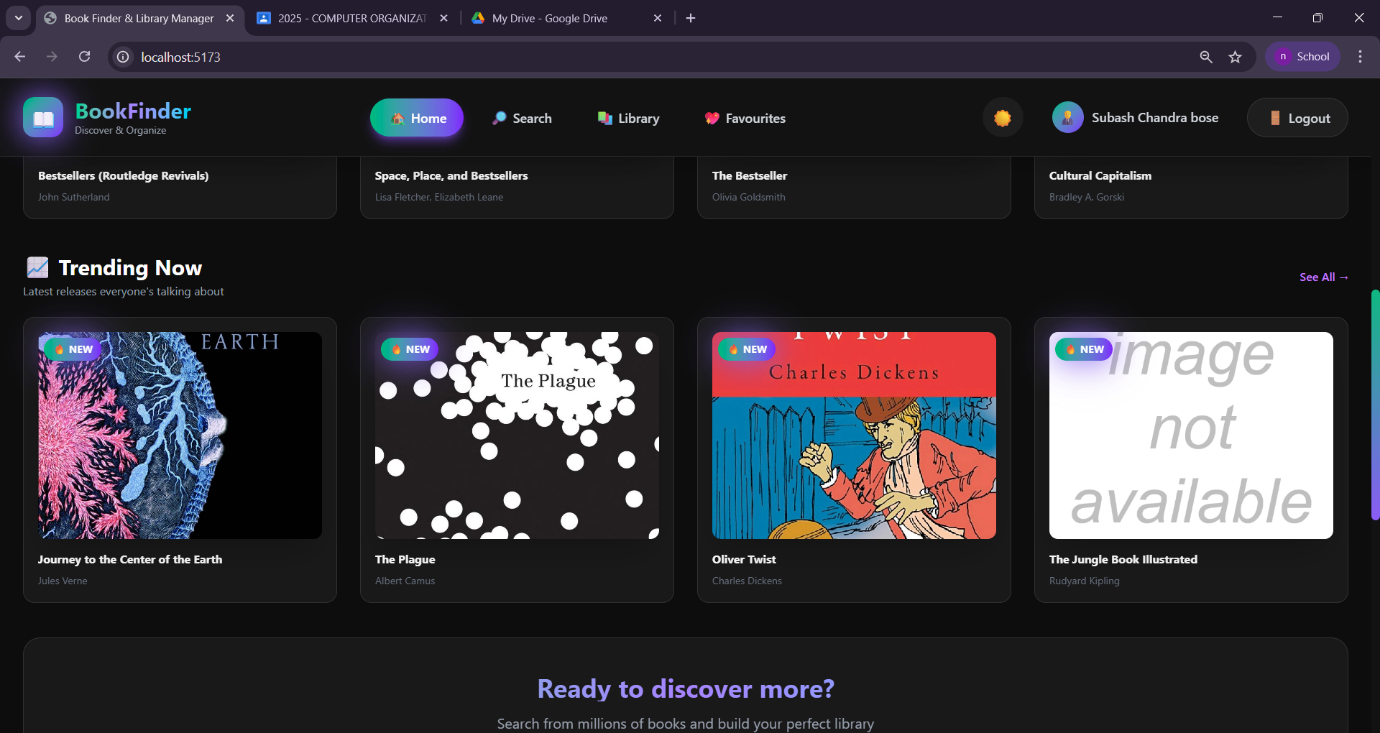
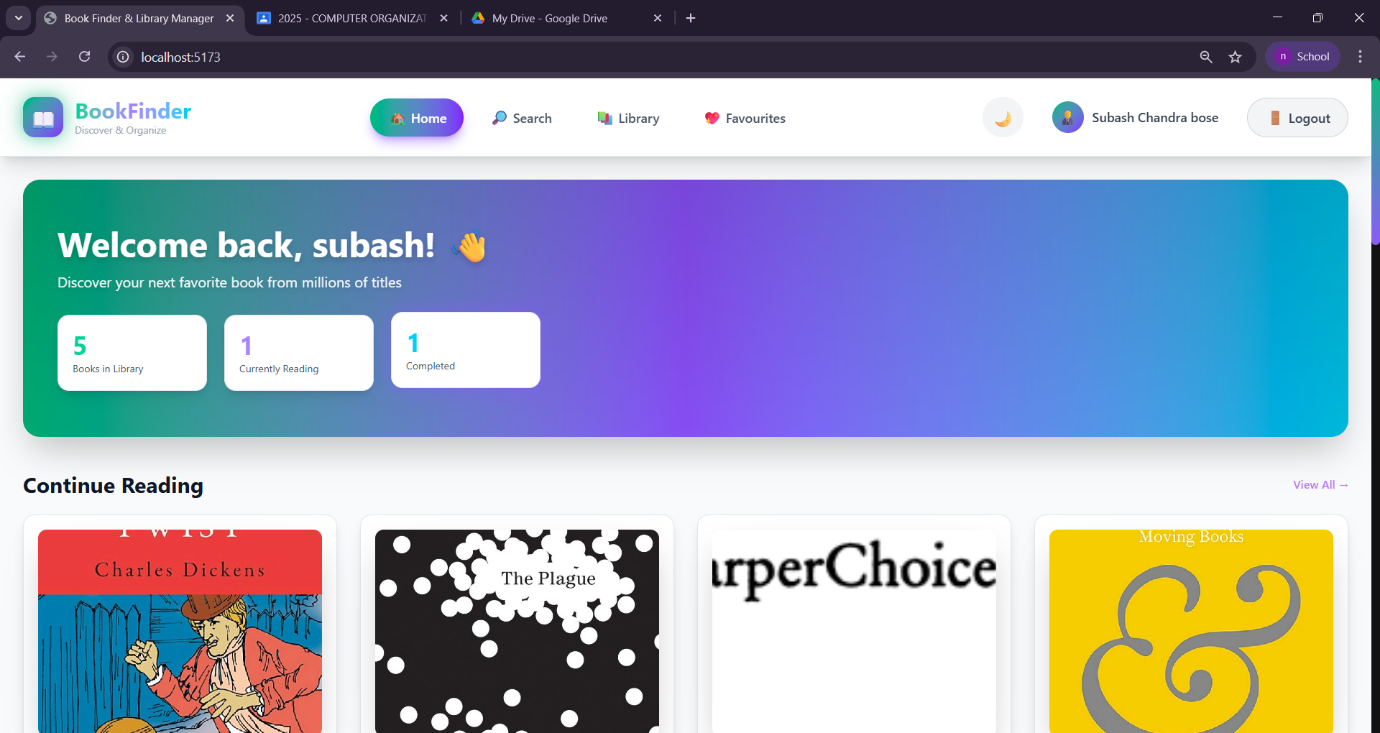
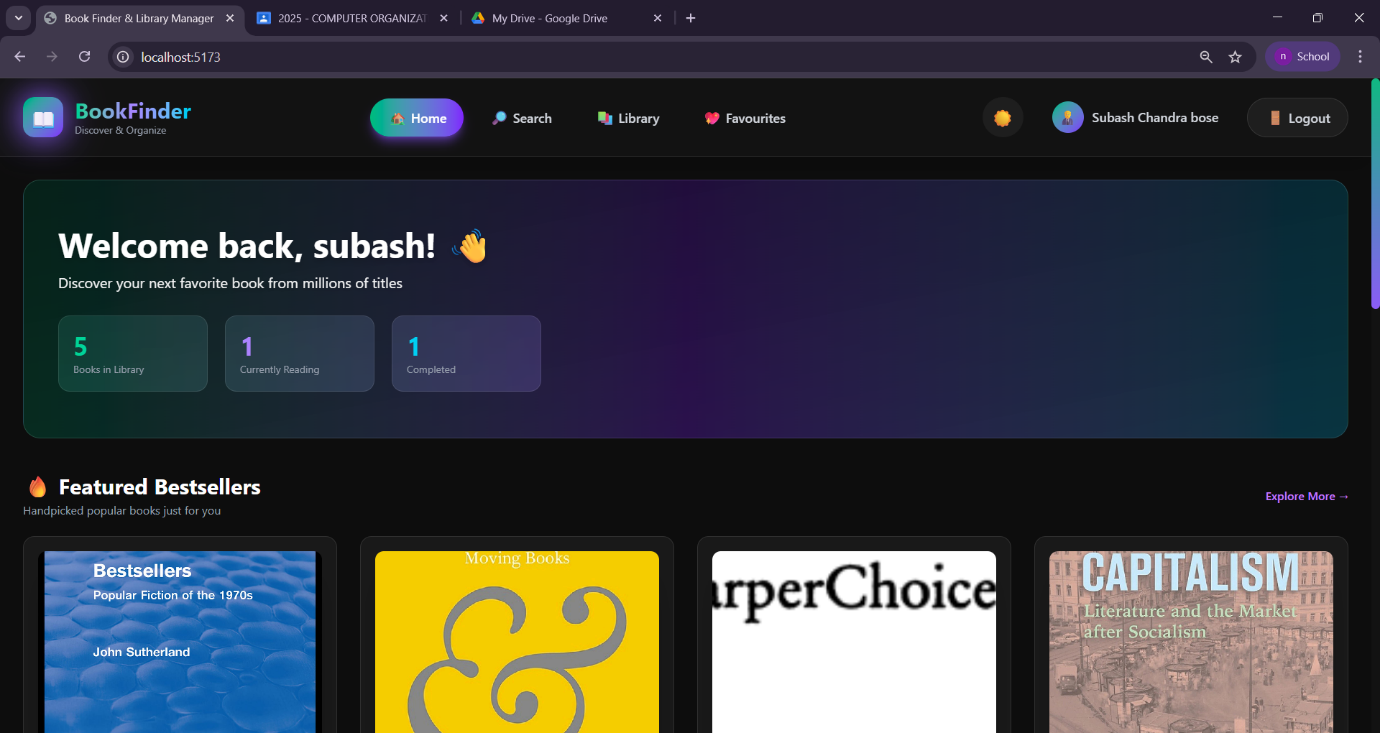
If the user is not logged in, they are forced to stay on the Login Page.

✔ 6. Displays the Layout

Shows the sidebar and main content when the user is logged in.

Otherwise, only the login page is shown.

**Output** 



**Project Demo link:**

<https://drive.google.com/file/d/1mizwQsyAozRrEOIcnFIcRBoY_VHIy0c/view?usp=sharing>