Project Report: Booknest: Where Stories Nestle

1. Introduction

1.1 Project Overview

"Booknest" is an online platform designed to create a comprehensive and user-friendly environment for managing, discovering, and acquiring books. It aims to streamline the process of Browse, purchasing, and selling books, providing a centralized system for efficient book management.

1.2 Purpose

The purpose of "Booknest" is to provide a user-friendly software solution that streamlines the process of managing, discovering, and acquiring books. It aims to offer a centralized platform for efficient book management, allowing users to securely browse, purchase, and interact with a diverse collection of stories. It addresses the challenges of traditional book management by offering real-time updates on book availability, personalized recommendations, and a seamless transaction process. This system seeks to enhance user satisfaction by providing a transparent and efficient way to engage with books, whether for reading, purchasing, or selling.

2. Ideation Phase

2.1 Problem Statement

In the current book ecosystem, users often face challenges such as limited access to diverse titles, inefficient purchasing processes, and a lack of centralized platforms for both buyers and sellers. Managing inventory, tracking orders, and providing timely updates can be cumbersome for sellers, while buyers may struggle with discovery and reliable access. There is a need for a digital system to streamline and automate these processes for a befler book experience.

2.2 Empathy Map Canvas

- **Users:** Readers, book enthusiasts, students, authors, casual buyers.
  - **Needs:** Easy book discovery, quick purchase process, diverse genres, reliable delivery, secure transactions, ability to review books, track orders.
  - **Pains:** Limited access to certain books, difficult navigation on existing platforms, slow delivery, security concerns during online payments, lack of personalized recommendations.
  - **Gains:** Access to a vast library, seamless buying experience, personalized suggestions, a platform to sell books, reliable customer support.

2.3 Scenario (Brainstorming)

Scenario: Anya, an avid reader, is looking for a new fantasy novel and wants to explore independent authors. She decides to use "Booknest" to find her next read.

1. **User Registration and Login:**
   - Anya visits the "Booknest" website and clicks on "Sign Up" to create an account.
   - She fills in her name, email, and a secure password.
   - After receiving a verification email and confirming her account, she logs in.
2. **Book Browse and Selection:**
   - Upon logging in, Anya is directed to the "Booknest" homepage. She sees options to browse by genre and author.
   - She navigates to the "Fantasy" genre and uses the search filter to look for independent authors.
   - She finds "The Dragon's Whisper," an independent novel, and clicks on it to view details, including the description, reviews, and availability.
3. **Book Purchase:**

    - Impressed by the reviews, Anya adds "The Dragon's Whisper" to her cart. She also notices a recommendation for a related book and adds that as well.
    - She proceeds to checkout, specifies her shipping address, and securely completes the purchase. An order is generated, and the inventory for the purchased books is updated.
4. **Order Confirmation and History:**
    - Anya receives an order confirmation with details of her purchase and an order ID.
    - She visits the "My Orders" section of her profile to track the status of her new order and review her past purchases.
5. **Review and Feedback:**
    - Once she finishes "The Dragon's Whisper," Anya returns to "Booknest" and leaves a positive review and a 5-star rating for the book, sharing her thoughts with other readers.

3. Requirements Analysis

3.1 Key Features

- **User Registration and Authentication:** Allows users to register accounts securely, log in, and authenticate their identity to access the book store platform.
- **Book Listings:** Displays a comprehensive list of available books with details such as title, author, genre, description, price, and availability status.
- **Book Selection:** Provides users with options to select their preferred books based on factors like genre, author, ratings, and popularity.
- **Purchase Process:** Allows users to add books to their cart, specify quantities, and complete purchases securely. Upon successful completion, an order is generated, and the inventory is updated accordingly.
- **Order Confirmation:** Provides users with a confirmation page or notification containing details of their order, including book information, total price, and order ID.
- **Order History:** Allows users to view their past and current orders, providing options to track shipments, review purchased books, and rate their shopping experience.
- **Organizer Dashboard:** Offers administrators (or sellers/organizers as per roles) an interface to manage book listings, inventory levels, user accounts, orders, and other platform-related activities.
- **Create Item:** Organizer can create items, add new items, get existing items, and update items.
- **Admin Dashboard:** Offers administrators an interface to manage book listings, inventory

levels, user accounts, orders, and other platform-related activities. Manages users and organizers.

- **Reporting and Analytics:** Generates reports and analytics on book sales, popular genres, user demographics, and other relevant metrics to gain insights into platform usage and performance.
- **Integration with External APIs:** Integrates with third-party APIs for services like payment processing, shipping logistics, and book recommendations to enhance the functionality and user experience of the book store platform.

3.2 ER Diagram (Entity-Relationship Diagram)

The ER Diagram for Booknest defines the relationships between various entities:

- **User-Book Relationship (Many-to-Many):**
  - **Type:** Many-to-Many (M:M). A single user can read or interact with many books, and a single book can be accessed by many users.
  - **Implementation:** Introduce an intermediate entity, "Interaction", with foreign keys to both User and Book tables. This table could store additional information like reading progress, reviews, or ratings.
- **Book-Inventory Relationship (One-to-Many):**
  - **Type:** One-to-Many (1:M). Each book can have multiple copies in inventory, but each copy belongs to one book.
  - **Implementation:** Maintain a separate Inventory table with fields like BookID (foreign key), quantity, location, and condition.
- **User-Order Relationship (One-to-Many):**
  - **Type:** One-to-Many (1:M). A single user can place multiple orders, but each order belongs to one user.
  - **Implementation:** Keep the UserID foreign key in the Order table to track user purchase history.
- **Additional Relationships:**
  - **Book-Author Relationship (Many-to-Many):** A book can have multiple authors, and an author can write multiple books. (Similar to User-Book, use an intermediate "WriflenBy" table).
  - **Book-Genre Relationship (Many-to-Many):** A book can belong to multiple genres, and a genre can have many books. (Similar to User-Book, use an intermediate "CategorizedAs" table).
  - **Review-User Relationship (Many-to-One):** A review is wriflen by one user, but a user can write many reviews. (Keep UserID as a foreign key in the Review table).

3.3 Application Flow

The application flow of Booknest guides users through their interaction with the platform:

- **Start:** Users open the Booknest app to explore a vast collection of books.
- **Home Page:** Users land on the home page, which provides an overview of the book store's offerings. From here, they can navigate to various sections of the app.
- **Access Profile:** Users have the option to access their profiles, allowing them to view or update personal information, preferences, and order history.
- **Book Selection:** After accessing their profiles, users proceed to browse and select books to purchase. The app presents a list of available books, along with details such as title, author, genre, and price.
- **Book Purchase:** Users navigate through the available book options and specify the quantity of each book they wish to purchase. They can also choose additional options such as e-book format or special editions.
- **View Orders:** Users have the option to view their current and past orders. This section provides details about ordered books, order status, and payment history.
- **Order Confirmation:** For new purchases, users can initiate the ordering process. This involves selecting books, specifying quantities, confirming the order, and receiving an order confirmation.
- **End:** The flow concludes as users have completed their desired actions within the Booknest app.

(See flow chart image for visual representation in the "Images" section at the end of this report: unnamed (14).png)

3.4 User Roles and Responsibilities

User:

- **Registration:** Users are responsible for registering an account on the Booknest book store app by providing essential details such as name, email, and password.
- **Profile Management:** Users have the capability to manage their profiles, allowing them to update information like email, name, and password.
- **Book Browse:** Users can browse through the available books, explore different genres, and search for specific titles or authors.
- **Purchase:** Users can add books to their cart, specify quantities, and complete purchases securely.
- **Feedback:** Provide feedback and ratings for purchased books and sellers on the Booknest platform.

- **Logout:** Lastly, they can logout from the Booknest book store app.

Seller:

- **Registration:** Sellers register an account on the Booknest book store app by providing necessary details such as business name, email, and password.
- **Profile Management:** Sellers have the capability to manage their profiles, allowing them to update information like email, business name, and password.
- **Book Listing:** Sellers can add new books to the platform, including details such as title, author, genre, description, price, and quantity available.
- **Inventory Management:** Sellers can manage their book inventory, updating stock levels, removing inactive listings, and handling book ratings.
- **Order Fulfillment:** Sellers are responsible for fulfilling orders placed by users, including packaging and shipping books in a timely manner.
- **Logout:** Finally, they can logout from the Booknest book store app.

Admin:

- **System Management:** Admins have full control over all aspects of the book store system, overseeing functionalities, configurations, and security.
- **User Management:** Admins can manage user information, including creating, updating, and deleting accounts. They also have authority over user ratings.
- **Book Management:** Admins can manage book listings, including adding new books, updating details, and removing inactive listings from the platform.
- **Seller Management:** Admins have the authority to manage seller information, including approving new seller accounts, updating profiles, and handling seller ratings.
- **Logout:** Finally, they can logout from the Booknest book store app.

3.5 Project Structure

The project follows a typical MERN stack architecture with distinct frontend and backend directories:

```
BOOK-STORE/
├── Backend/
│   ├── db/
│   │   ├── Seller/
│   │   └── Users/
│   ├── config.js
│     ├──  node_modules/
│   ├── uploads/
│   ├─── package-lock.json
│     ├── package.json
│   └── server.js
├── Frontend/
│     ├──  node_modules/
│   ├── public/
│   ├── src/
│   │   ├── Admin/
│     │    ├── Componenets/
│   │   ├── Seller/
│   │   ├── User/
│   │   ├── App.css
│   │   ├── App.jsx
│   │    ├── index.css
│   │   └── main.jsx
│    ├── .eslintrc.cjs
│   ├── .gitignore
│   ├── index.html
│   ├─── package-lock.json
│     ├── package.json
│     └── vite.config.js
└── README.md
```

(See unnamed (15).png for visual representation of the project structure in the "Images" section)

4. Technology Stack/Pre-requisites

To develop a full-stack Book Store App using React.js, Node.js, Express.js, and MongoDB, the following prerequisites are essential:

- **Node.js and npm:** Node.js is required to run JavaScript code on the server side. It provides a scalable and efficient platform for building network applications. npm (Node Package Manager) is included with Node.js.
  - Download: Node.js Downloads
  - Installation instructions: Node.js Package Manager
- **MongoDB:** A flexible and scalable NoSQL database that stores data in a JSON-like

format. It provides high performance, horizontal scalability, and seamless integration with Node.js, making it ideal for handling large amounts of structured and unstructured data.

  - Download: MongoDB Community Server
  - Installation instructions: MongoDB Manual Installation

- **Express.js:** A fast and minimalist web application framework for Node.js. It simplifies API and web application creation, offering routing and middleware support.
  - Installation: npm install express
- **React.js:** A popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components.
  - Quick Start with Vite: npm create vite@latest, cd my-app, npm install, npm run dev
  - To create a new React project: npx create-react-app your-app-name
  - Navigate into project directory: cd your-app-name
  - Start development server: npm run dev or npm start
  - Official React documentation: React.dev
- **HTML, CSS, and JavaScript:** Basic knowledge of HTML for structure, CSS for styling, and JavaScript for client-side interactivity is essential.
- **Database Connectivity:** Use a MongoDB driver or an Object-Document Mapping (ODM) library like Mongoose to connect Node.js server with MongoDB and perform CRUD operations.
- **Front-end Library:** Utilize React to build the user-facing part of the application, including product listings, booking forms, and user interfaces for the admin dashboard. Libraries like Material UI and Bootstrap were also used for befler UI.
- **Version Control:** Use Git for version control, enabling collaboration and tracking changes. Platforms like GitHub or Bitbucket can host your repository.
  - Git: Git Downloads
- **Development Environment:** Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.
  - Visual Studio Code: VS Code Download
  - Sublime Text: Sublime Text Download
  - WebStorm: WebStorm Download

5. Project Setup and Configuration

1. **Install required tools and software:**
   - Node.js
   - MongoDB
   - React.js
2. **Create project folders and files:**
   - Client folders.

- Server folders.

3. **Install Packages:**
   - **Frontend npm Packages:**
     - Axios
     - React-Router-dom
     - Bootstrap
     - React-Bootstrap
     - Material UI (specifically @emotion/react, @emotion/styled, mdb-react-ui-kit)
   - **Backend npm Packages:**
     - Express
     - Mongoose
     - Cors
     - Bcrypt
     - Nodemon
     - Express-session

6. Backend Development

- **Setup Express Server:**
  - Create an index.js file in the server (backend folder).
  - Create a .env file and define the port number to access it globally.
  - Configure the server by adding cors and body-parser.
- **Define API Routes:**
  - Create separate route files for different API functionalities such as users, orders, and authentication.
  - Define the necessary routes for listing products, handling user registration and login, managing orders, etc.
  - Implement route handlers using Express.js to handle requests and interact with the database.
- **Implement Data Models:**
  - Define Mongoose schemas for the different data entities like products, users, and orders.
  - Create corresponding Mongoose models to interact with the MongoDB database.
  - Implement CRUD operations (Create, Read, Update, Delete) for each model to perform database operations.[1]

- **User Authentication:**[2]

  - Create routes and middleware for user registration, login, and logout.[3]

- Set up authentication middleware to protect routes[4] that require user authentication.

- **Error Handling:**
  - Implement error handling middleware to catch and handle any errors that occur during the API requests.
  - Return appropriate error responses with relevant error messages and HTTP status codes.

7. Database

1. **Configure MongoDB:**
   - Install Mongoose.
   - Create database connection.
   - Create Schemas & Models.
2. **Connect database to backend:**
   - Ensure the database is connected before performing any actions through the backend. The connection typically involves mongoose.connect() and server.listen(). (See unnamed.png for an example of MongoDB connection code in the "Images" section)
3. **Configure Schema:**
   - Configure the Schemas for MongoDB database to store the data in a structured paflern, using information from the ER diagrams.
   - User Schema Example: Defines the structure of user data including username, email, and password.

     (See unnamed (1).png for visual representation of User.js schema in the "Images" section)

8. Frontend Development

1. **Setup React Application:**
   - Create the React application.
   - Configure routing using libraries like React Router DOM.
   - Install required libraries for UI and functionality (e.g., Axios, Bootstrap, React-Bootstrap, Material UI).
   - This involves creating the initial application structure, installing necessary libraries, and organizing project files for efficient development.
2. **Design UI components:**

- Create reusable components for all interactive elements like book listings, buflons, and user profiles.
- Implement a layout and styling scheme to define the overall look and feel of the application, ensuring a visually appealing and intuitive interface.
- Integrate a navigation system to allow effortless exploration of different sections of the application.

3. **Implement frontend logic:**
   - Integrate with API endpoints to fetch and send data to the backend.
   - Implement data binding to display dynamic data from the backend on the UI.

9. Project Implementation

Finally, after finishing coding the projects, the whole project is run to test its working process and look for bugs. Now, let's have a final look at the working of our Booknest application.

- **Landing page:** (See unnamed (4).jpg in the "Images" section)
- **Login Page:** (See unnamed (3).png in the "Images" section)
- **Home Page:** (See unnamed (2).jpg for "Best Seller" in the "Images" section)
- **Books Page:** (See unnamed (5).jpg for "Books List" in the "Images" section)
- **Wishlist Page:** (See unnamed (6).png in the "Images" section)
- **Seller Dashboard:** (See unnamed (8).png in the "Images" section)
- **Admin Dashboard:** (See unnamed (10).png in the "Images" section)
- **Admin Vendor Products Page:** (See unnamed (9).png in the "Images" section)
- **Admin Users Page:** (See unnamed (11).png in the "Images" section)
- **Admin User Orders Page:** (See unnamed (12).png in the "Images" section)
- **Admin Vendors Page:** (See unnamed (13).png in the "Images" section)

10. Conclusion

"Booknest" is a full-stack web application designed to streamline the process of managing, discovering, and acquiring books efficiently. The platform empowers users to browse and purchase books, while allowing sellers to list and manage their inventory. Admins can oversee the entire system, creating a smooth and organized book management pipeline.

Key Takeaways:

- Built using the MERN stack (MongoDB, Express.js, React.js, Node.js).
- Supports user registration, book listings, secure purchases, and order tracking.
- Prioritizes data security through authentication and role-based access.
- Offers a clean frontend UI and REST API-driven backend.
- Can be extended to include real-time notifications, charts, analytics, and video support.

Project Benefits:

- Reduces manual effort in book management.
- Improves user satisfaction through transparency and ease of access.
- Scales easily for both small and large book collections and user bases.
- Enhances accountability and traceability within the platform for sellers and administrators.

11. Future Scope

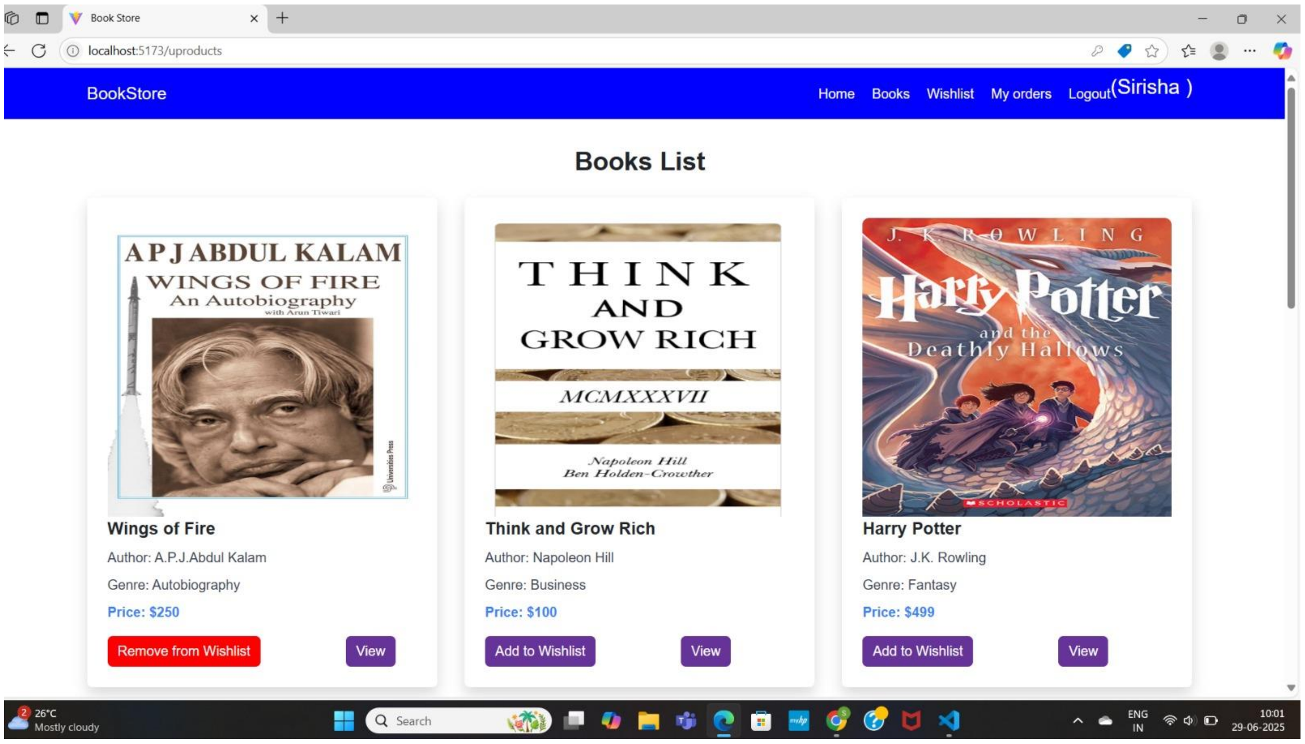
- Mobile App development for wider accessibility.
- AI-based book recommendations and genre categorization.
- Advanced Analytics Dashboard for insights into sales and user behavior.
- Multi-language support to cater to a diverse user base.
- Integration with SMS/Email services for order updates and promotions.
- Integration with external review platforms.
- Author profiles and direct messaging features.

12. Team Details

- **Team Leader:** LALAM LAXMIKANTH (ROLL NO: 23P31A05M2)
- **Team Member:** MAHAMMAD ABUBAKAR SIDDIQ (ROLL NO: 23P31A05M3)
- **Team Member:** MAKA SRUJANA (ROLL NO: 23P31A05M4)
- **Team Member:** MEDISETTI SIRISHA (ROLL NO: 23P31A05M8)

Images

Books Page (Books List):



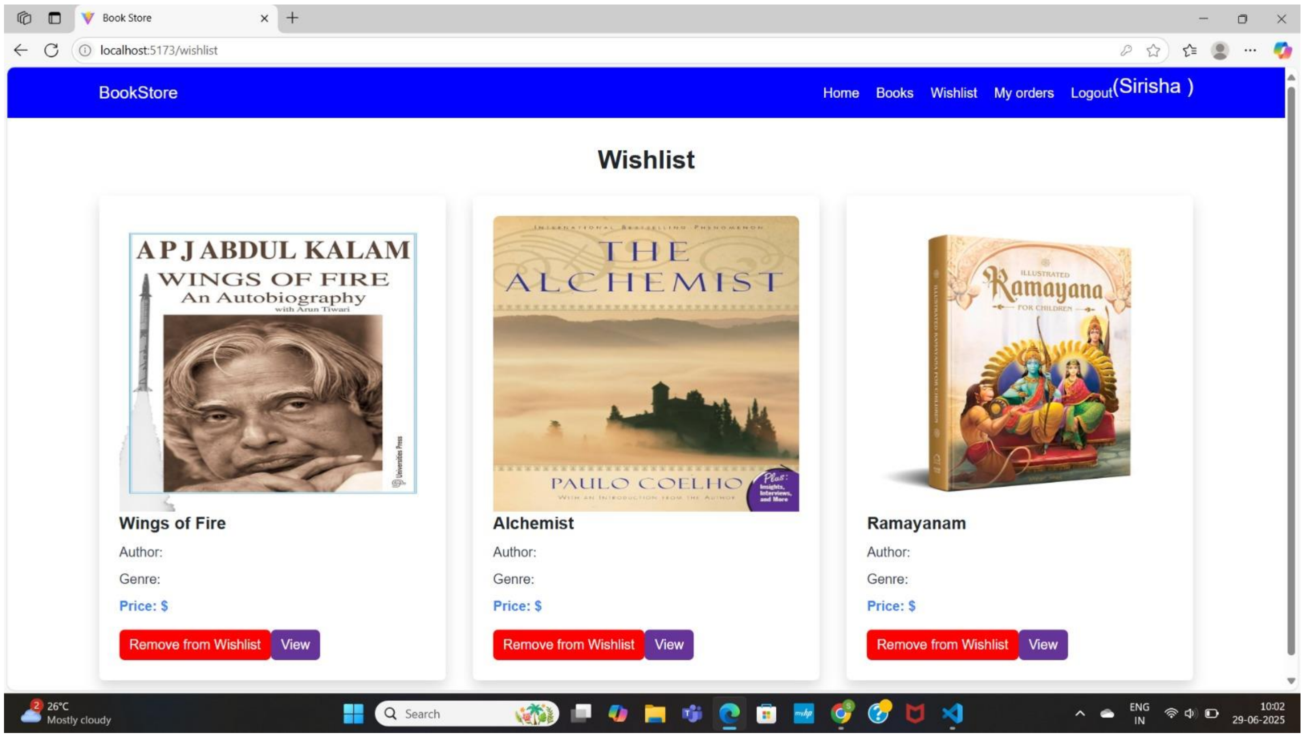
Books List

Wings of Fire
Author: A.P.J.Abdul Kalam
Genre: Autobiography
Price: $250

Think and Grow Rich
Author: Napoleon Hill
Genre: Business
Price: $100

Harry Potter
Author: J.K. Rowling
Genre: Fantasy
Price: $499

Wishlist Page:



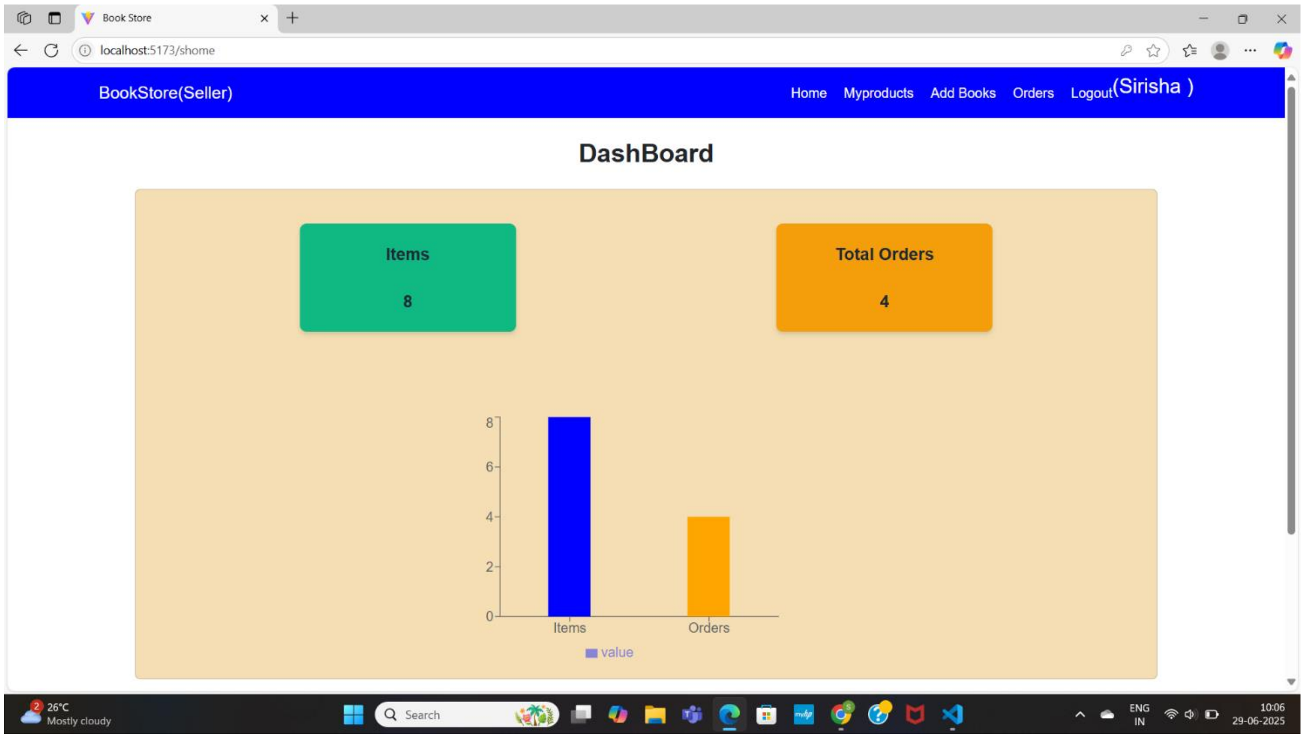
Wishlist

Wings of Fire
Author:
Genre:
Price: $

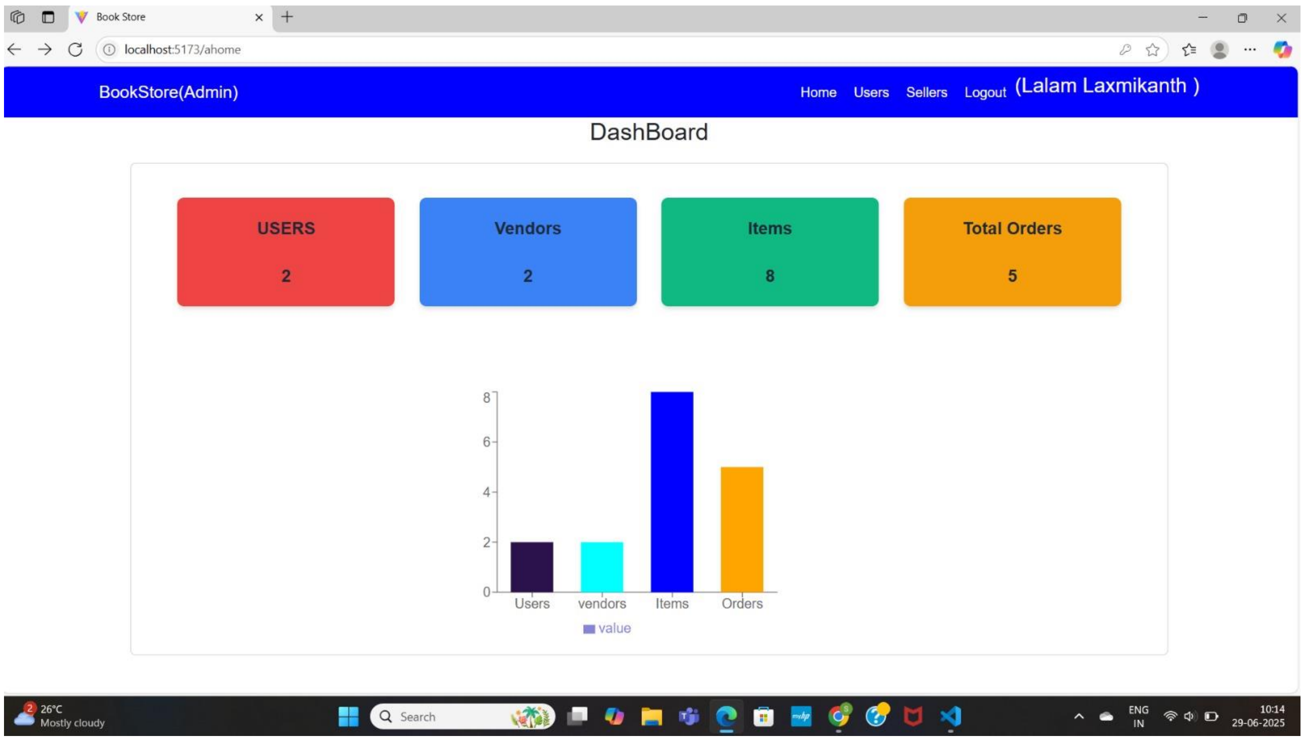
Alchemist
Author:
Genre:
Price: $

Ramayanam
Author:
Genre:
Price: $

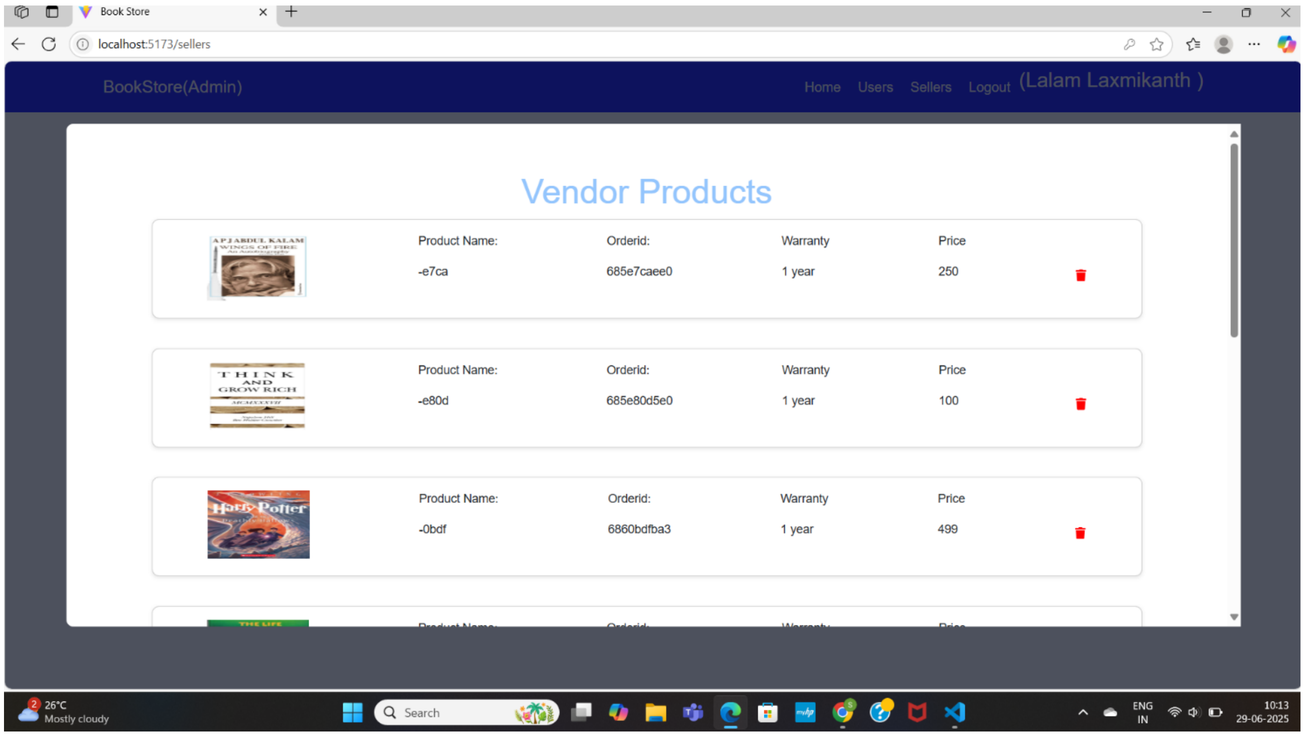
Seller Dashboard:



Admin Dashboard:



Admin Vendor Products Page:



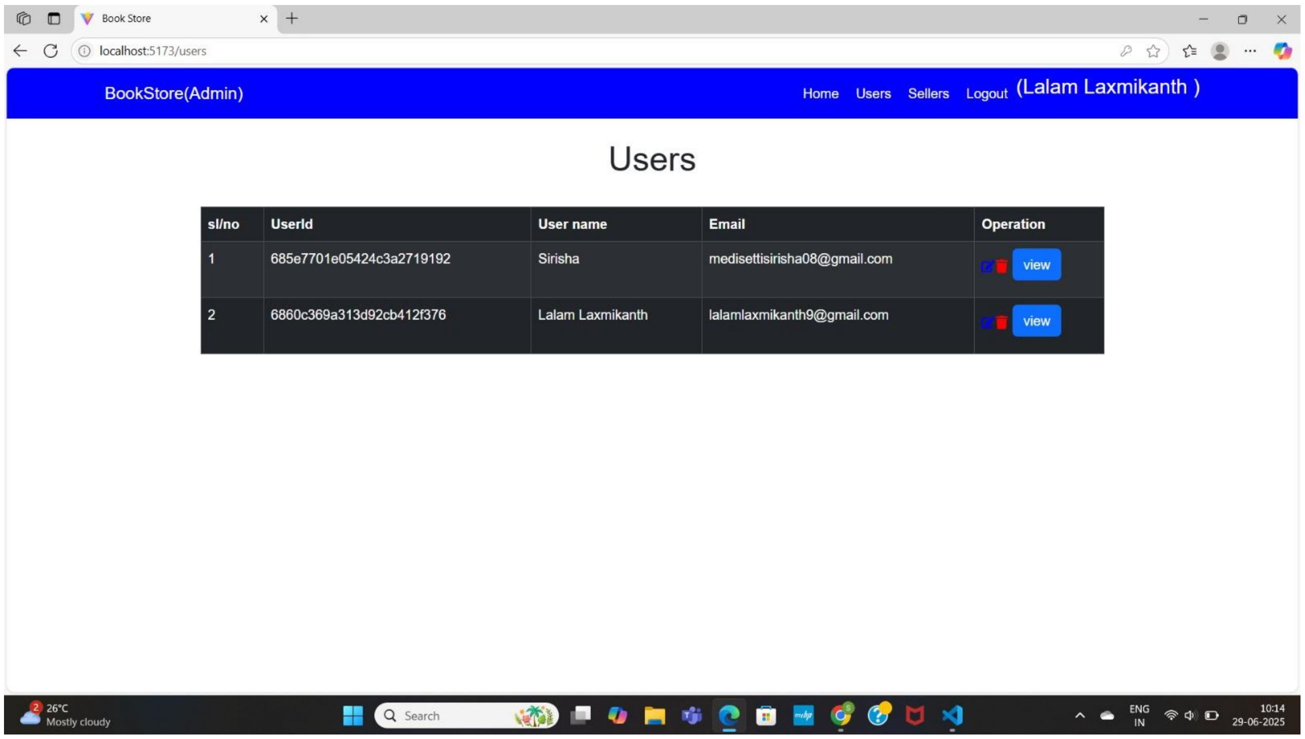
BookStore(Admin)

Home Users Sellers Logout (Lalam Laxmikanth)

Vendor Products

	Product Name:	OrderId:	Warranty	Price
	-e7ca	685e7caee0	1 year	250
	-e80d	685e80d5e0	1 year	100
	-0bdf	6860bdfba3	1 year	499

Admin Users Page:



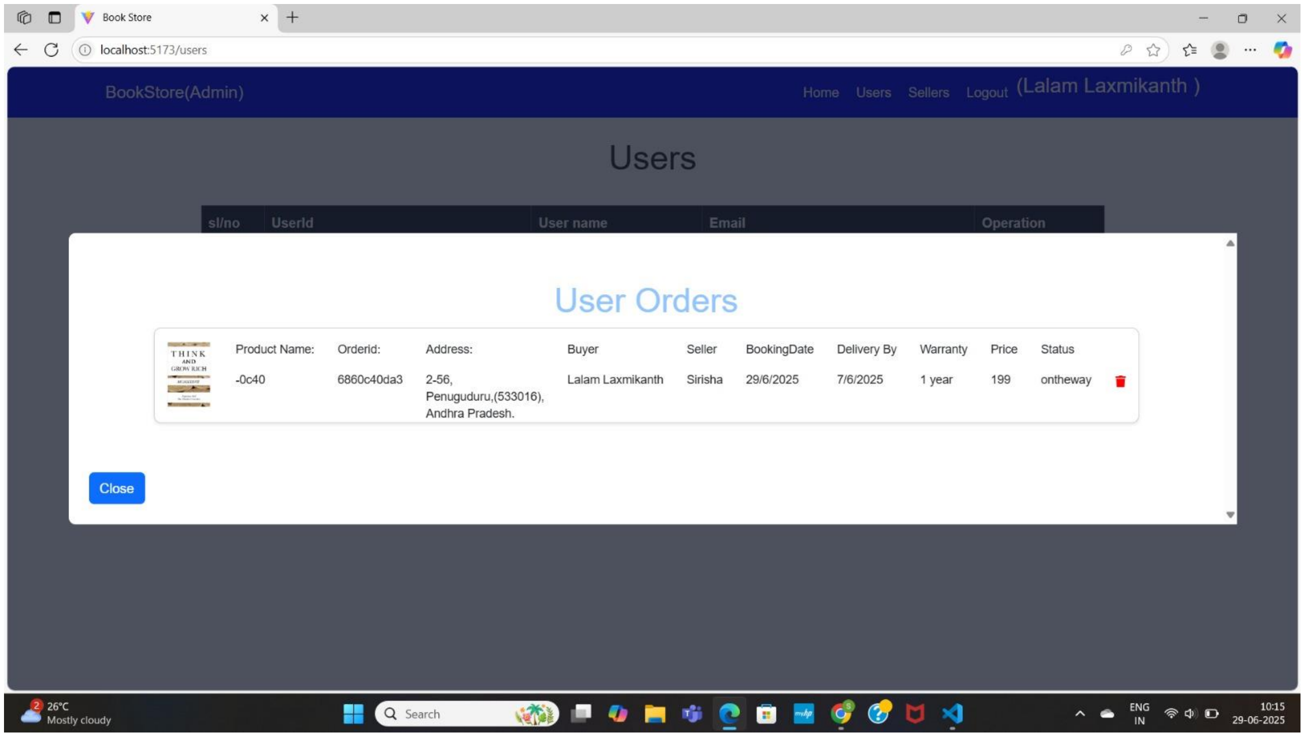
BookStore(Admin)

Home Users Sellers Logout (Lalam Laxmikanth)

Users

sl/no	UserId	User name	Email	Operation
1	685e7701e05424c3a2719192	Sirisha	medisettisirisha08@gmail.com	view
2	6860c369a313d92cb412f376	Lalam Laxmikanth	lalamlaxmikanth9@gmail.com	view

Admin User Orders Page:



localhost:5173/users

BookStore(Admin) Home Users Sellers Logout (Lalam Laxmikanth)

Users

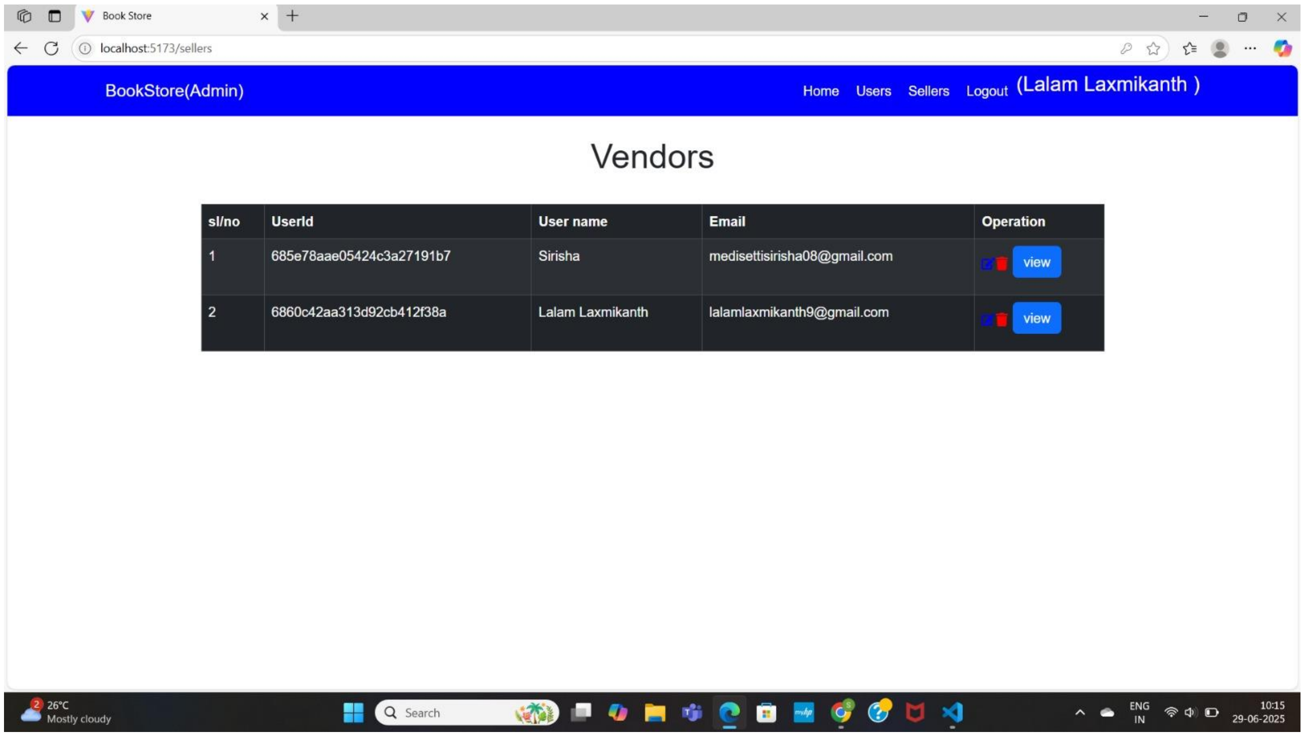
sl/no	UserId	User name	Email	Operation

User Orders

	Product Name:	OrderId:	Address:	Buyer	Seller	BookingDate	Delivery By	Warranty	Price	Status
	-0c40	6860c40da3	2-56, Penuguduru,(533016), Andhra Pradesh.	Lalam Laxmikanth	Sirisha	29/6/2025	7/6/2025	1 year	199	ontheway

Close

Admin Vendors Page:



localhost:5173/sellers

BookStore(Admin) Home Users Sellers Logout (Lalam Laxmikanth)

Vendors

sl/no	UserId	User name	Email	Operation
1	685e78aae05424c3a27191b7	Sirisha	medisettisirisha08@gmail.com	view
2	6860c42aa313d92cb412f38a	Lalam Laxmikanth	lalamlaxmikanth9@gmail.com	view