**Full Stack Development with MERN**

**1. Introduction**

**Project Title:** Book Store Platform using MERN

**Team Members:**

* Santhosh M
* Mathan A
* Vinayagamoorthy P
* Melvin Joseph J
* Vinodh D

**2. Project Overview**

**Purpose:** This project aims to create a responsive and user-friendly Book-Store Application, a masterpiece crafted with precision using the powerful MERN (MongoDB, Express.js, React, Node.js) Stack. Immerse yourself in a world where the love for reading converges seamlessly with cutting-edge technology, redefining the way bibliophiles explore, discover, and indulge in their literary pursuits. **Features:**

* User registration and login with JWT authentication.
* Admin dashboard for tracking the application and content management
* Seller dashboard for tracking the orders
* Simple and efficient page for making orders

**3. Architecture**

* **Frontend:** Built using React, the frontend provides a dynamic user interface. It includes components for user authentication, book browsing, and content consumption, leveraging React Router for seamless navigation.
* **Backend:** The backend is built with Node.js and Express.js. It handles requests, manages authentication, and connects with MongoDB for data storage. Controllers manage book data and user interactions.
* **Database:** MongoDB stores information about users, books, category, and progress. It is structured with collections for each major feature (e.g., genre, books), enabling quick retrieval and updates.4. Setup Instructions

**4. Setup Instructions**

**Prerequisites:**

* Node.js v14+
* MongoDB v4+
* (Optional) npm or yarn for package management

**Installation:**

1. Clone the repository: git clone <repository-url>
2. Navigate to both the backend and frontend directories and install dependencies:

For Backend: cd backend

npm install

For Frontend: cd ../frontend

npm install

1. Set up environment variables by creating a .env file in the backend directory with database connection strings, JWT secret, etc.

**5. Folder Structure**

**Client:**  
The frontend directory contains:

* src/components - Contains reusable React components.
* src/pages - Different pages (e.g., seller dashboard, user Login, Register).
* src/utils - Utility functions and API calls.
* src/styles - CSS files for styling.

**Server:**  
The backend directory is organized as follows:

* config - Database connection configuration.
* controllers - Functions to handle business logic.
* routers - Defines routes for various API endpoints.
* middlewares - Middleware for authentication and error handling.
* schemas - MongoDB schemas and models.

**6. Running the Application**

**Frontend:**  
Start the frontend server: npm start

**Backend:**  
Start the backend server: npm start

**7. API Documentation**

**7.1 Authentication Endpoints**

* POST /api/auth/register
  + Register a new user.
* POST /api/auth/login
  + Log in a user and return a JWT token.
* GET /api/auth/profile
  + Get the logged-in user's profile (requires authentication).

**2. Book Endpoints**

* **GET** /api/books
  + Retrieve all books.
* **GET** /api/books/:id
  + Retrieve a single book by its ID.
* **POST** /api/books
  + Add a new book (Admin only).
* **PUT** /api/books/:id
  + Update a book's details by its ID (Admin only).
* **DELETE** /api/books/:id
  + Delete a book by its ID (Admin only).

**8. Authentication**

Authentication is handled using JWT tokens. After login, users receive a token stored in local storage. Protected routes require a valid token for access, which is verified using middleware.

**9. User Interface**

The interface includes:

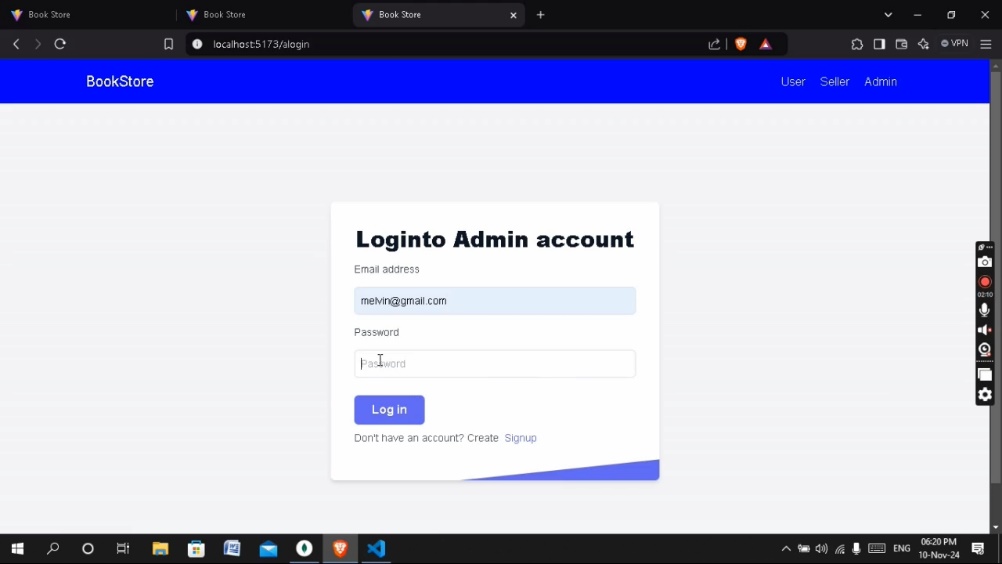
* A login and registration form.
* Dashboard page where users and admin can track the progress.
* Simple and efficient page to order the book

**10. Testing**

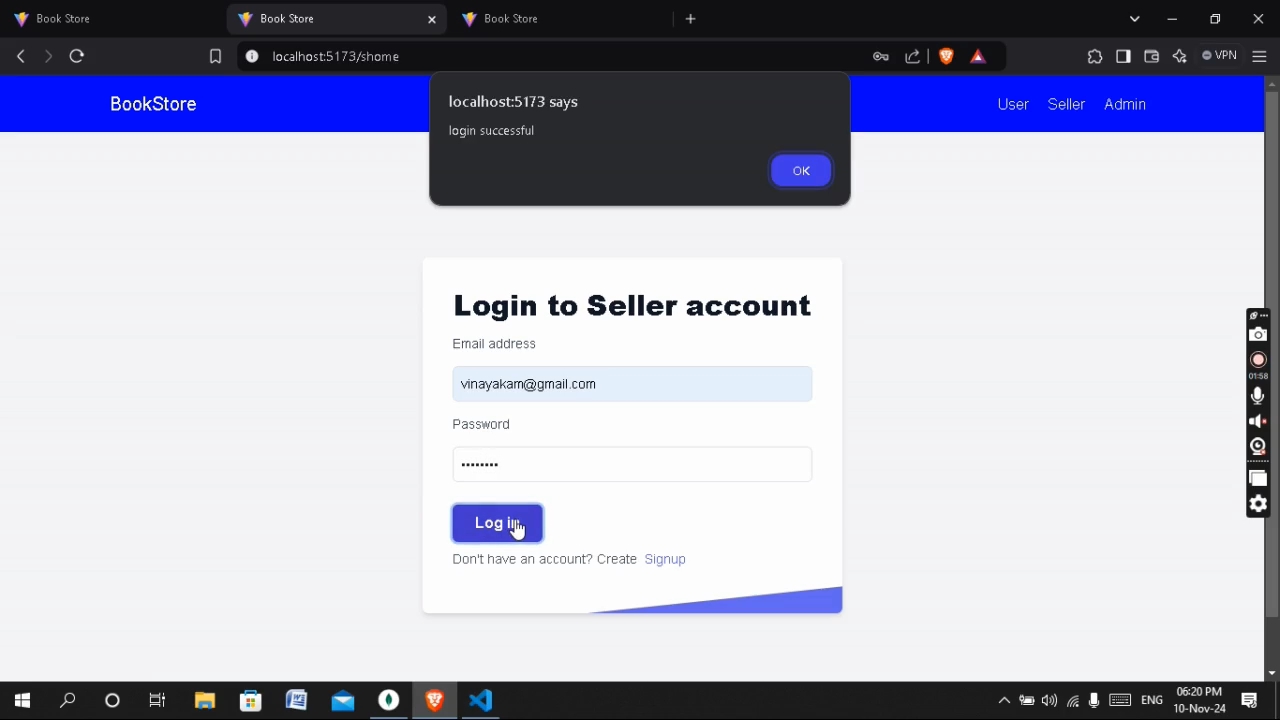
Testing is performed using tools like Jest and Postman for API testing. Unit tests cover component functionality and API response validation.

**11. Screenshots**

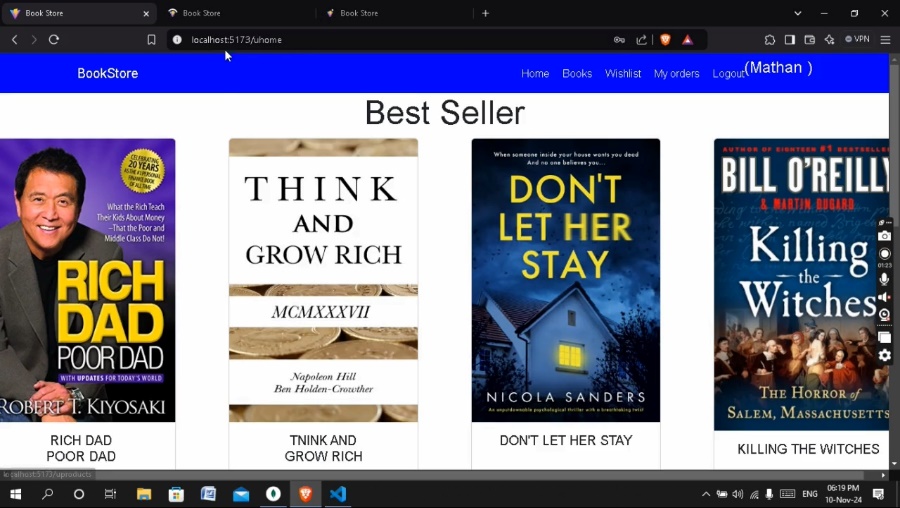
* Admin login and registration form.

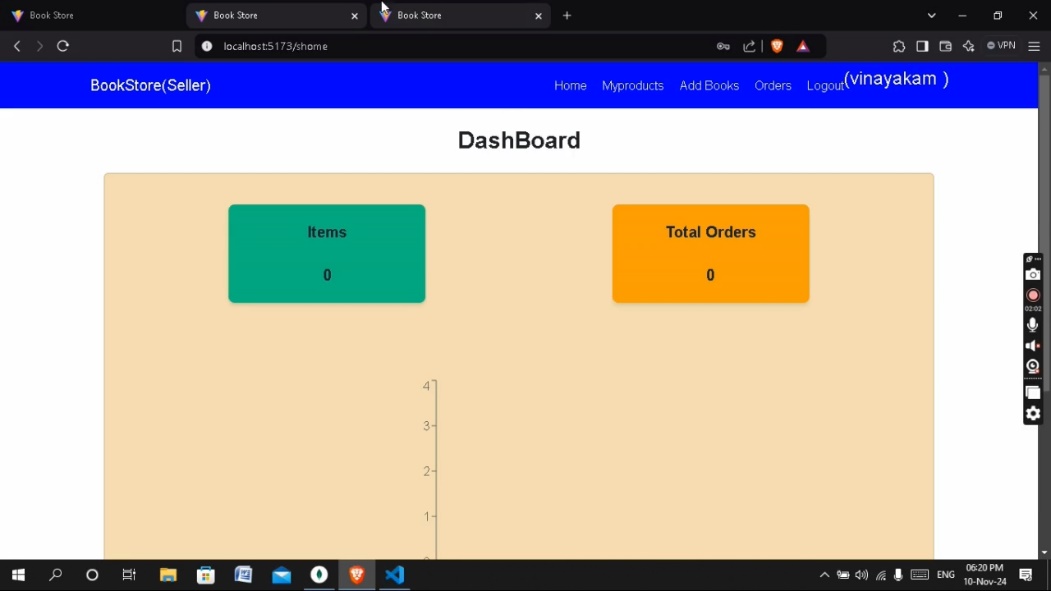


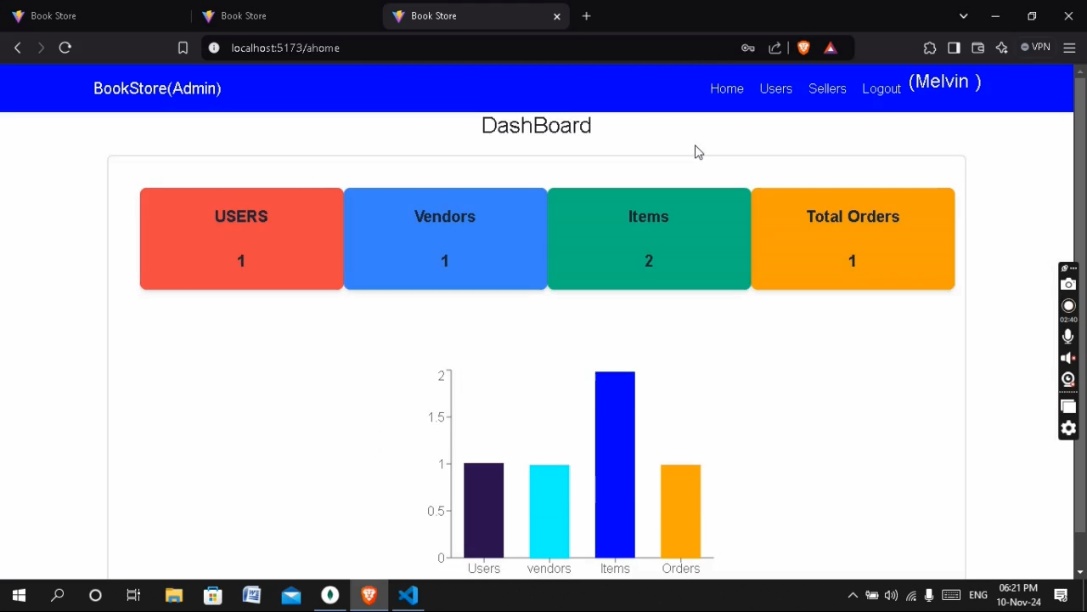
* Seller login and registration page



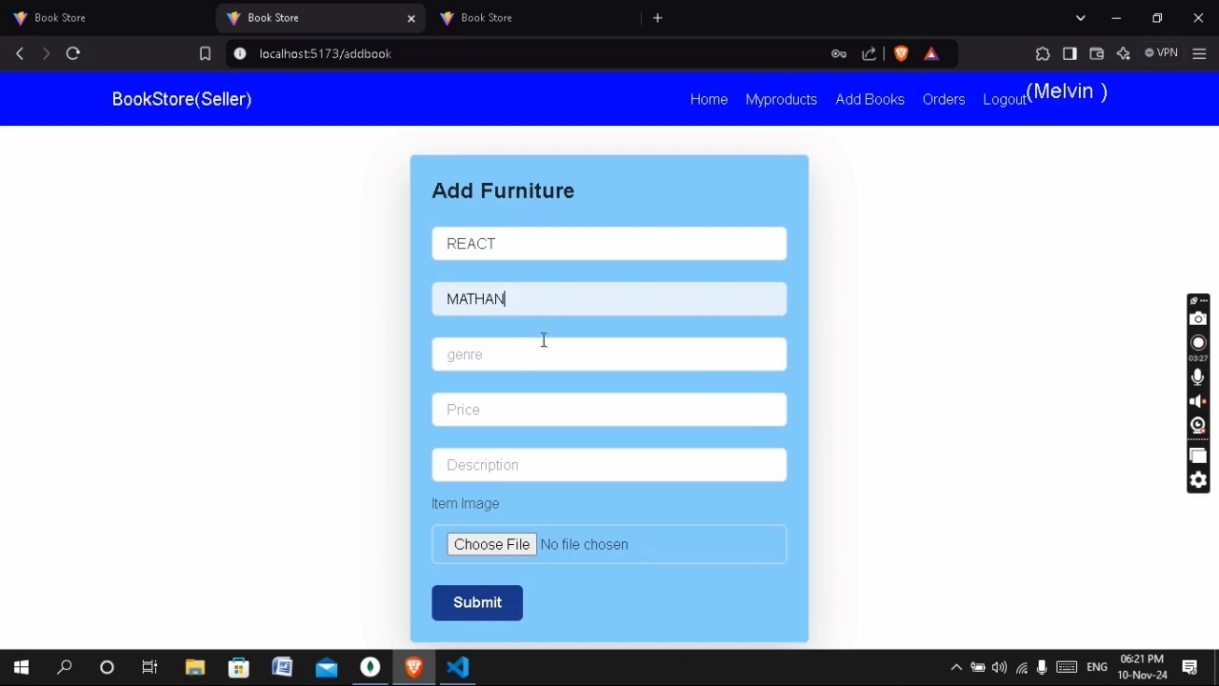
* SELLER PAGE



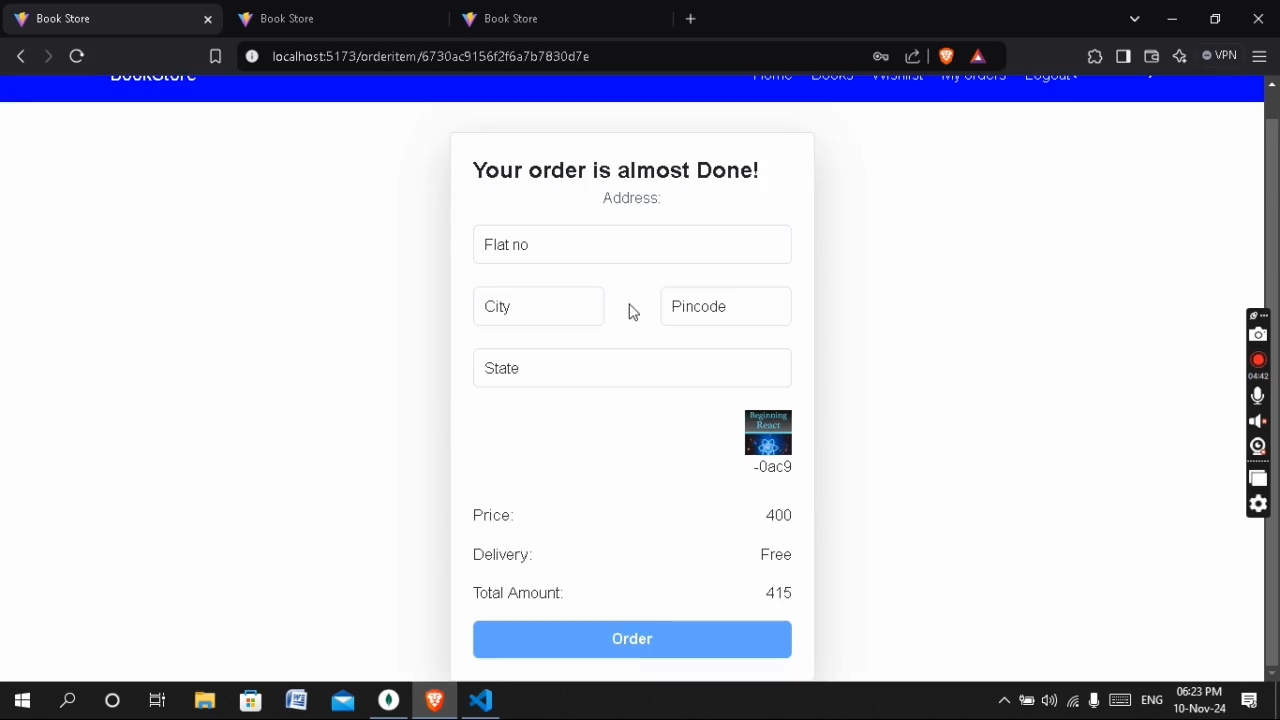
* SELLER DASHBOARD
* ADMIN DASHBOARD



* SELLER ADD PAGE



* ORDER ITEMS



.

**12. Known Issues**

* Token expiration may log users out unexpectedly.
* Slow response time when fetching large datasets

**13. Future Enhancements**

* Improve the search functionality by implementing full-text search using MongoDB's text indexes or integrating with a more advanced search engine like **Elasticsearch** for better results based on keywords, author names, or categories.
* Implement a recommendation engine using collaborative filtering or content-based filtering to suggest books based on user history, reviews, and browsing behavior.
* **Wishlist and Favorites**: Allow users to create a wishlist or mark books as favorites for future purchases.
* Enable users to share their favorite books or reviews on social media platforms directly from the app.