**Project Documentation for Book-Store**

**1. Introduction**

**Project Title**: Book Store Web Application

**Team Members:**

* + Mukesh D
  + Pranesh R K
  + Sham Ganesh
  + Thulasirajan T

**2. Project Overview**

* **Purpose:**

A bookstore web application provides an online platform for users to browse, purchase, and read books conveniently. It enhances the book-buying experience with user-friendly navigation, reviews, and recommendations. Additionally, it helps store owners manage inventory and track sales efficiently.

**Features:**

1. **Extensive Book Catalog:** Provides a wide range of books with detailed descriptions, reviews, and ratings for easy browsing and informed purchasing.
2. **Secure and Convenient Shopping:** Includes a shopping cart, multiple payment options, and order tracking to ensure a smooth and secure purchasing process.
3. **Personalized Recommendations:** Utilizes user preferences and past purchases to suggest books, enhancing the user experience with tailored recommendations.

**3. Architecture:**

* Frontend:

The frontend is built using React.js, providing a responsive and dynamic user interface. React components manage the different pages and functions, allowing users to add Wishlist, order, and view dashboard.

* Backend:

Node.js and Express.js power the backend, handling user requests, authentication, Manages business logic, user authentication, and other backend services., and other business logic. The backend follows a RESTful API approach to communicate with the frontend.

* Database:

MongoDB is used as the database, Stores user information, book details, orders, Wishlist etc.

**4. Setup Instruction:**

* Prerequisites:
* [Node.js](https://nodejs.org/): v14+
* [MongoDB](https://www.mongodb.com/): Community Edition or MongoDB Atlas (for cloud setup)
* npm or yarn as a package manager
* Installation:

1. Clone the repository: `git clone [repository link]`

2. Navigate to the project folder.

3. Install dependencies:

* For the backend: `cd server && npm install`
* For the frontend: `cd client && npm install`

**4. Configure environment variables:**

* Create a `.env` file in the server directory with required keys (e.g., database URI, JWT secret).

**5. Folder Structure**

* Frontend:
* `src/components`: Contains reusable UI components.
* `src/pages`: Houses different pages (User, Seller, Admin).
* `src/services`: Manages API calls and handles communication with the backend.
* Server:
* `controllers/`: Holds logic for handling various API requests.
* `models/`: Contains Mongoose schemas and models for MongoDB collections.
* `routes/`: Defines all API routes.
* `middlewares/`: Houses middleware for authentication and error handling.

**6. Running the Application**

* Frontend:
* Navigate to the client directory and run `npm start` to start the React application.
* Backend:
* Navigate to the server directory and run `npm start` to start the Express.js server.

**7. API Documentation**

**Add a Book:**

* **Method:** POST /books
* **Description:** Adds a new book to the inventory.

**View All Books:**

* **Method:** GET /books
* **Description:** Retrieves a list of all books in the inventory.

**Update a Book:**

* **Method:** PUT /books/{id}
* **Description:** Updates the details of a specific book.

**Delete a Book:**

* **Method:** DELETE /books/{id}
* **Description:** Removes a book from the inventory.

**Search Books:**

* **Method:** GET /books/search
* **Description:** Searches for books by title, author, or category.

**8. Authentication**

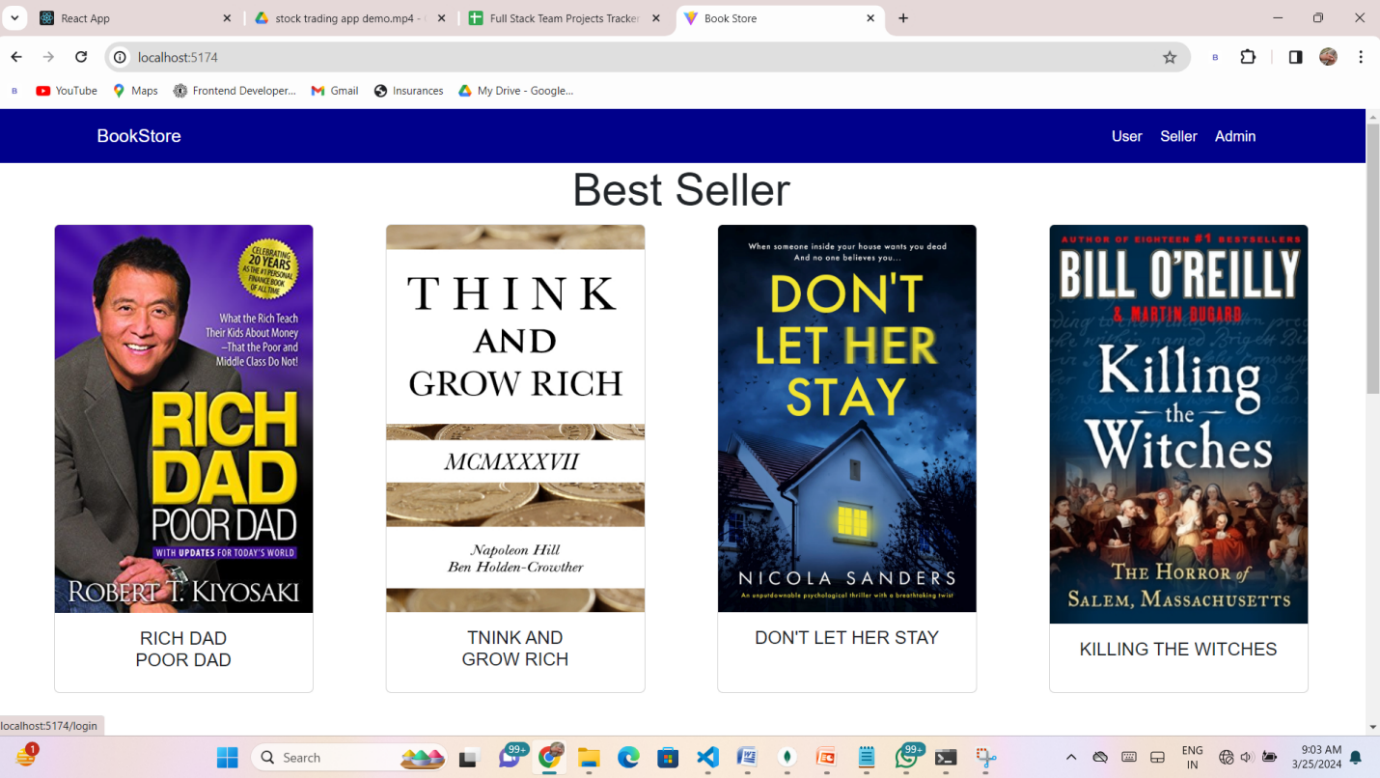
**Authentication Mechanism:**

Authentication is managed using JSON Web Tokens (JWT). When a user logs in, a JWT is generated and sent to the frontend to be stored locally (typically in cookies or localStorage).

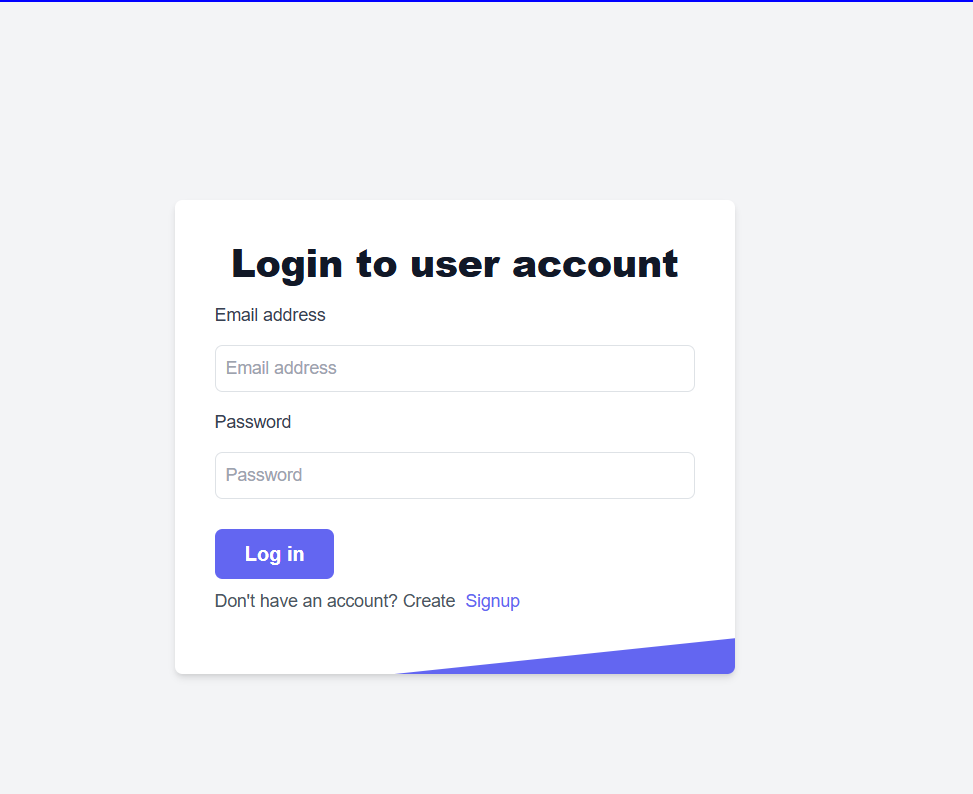
**Authorization:**

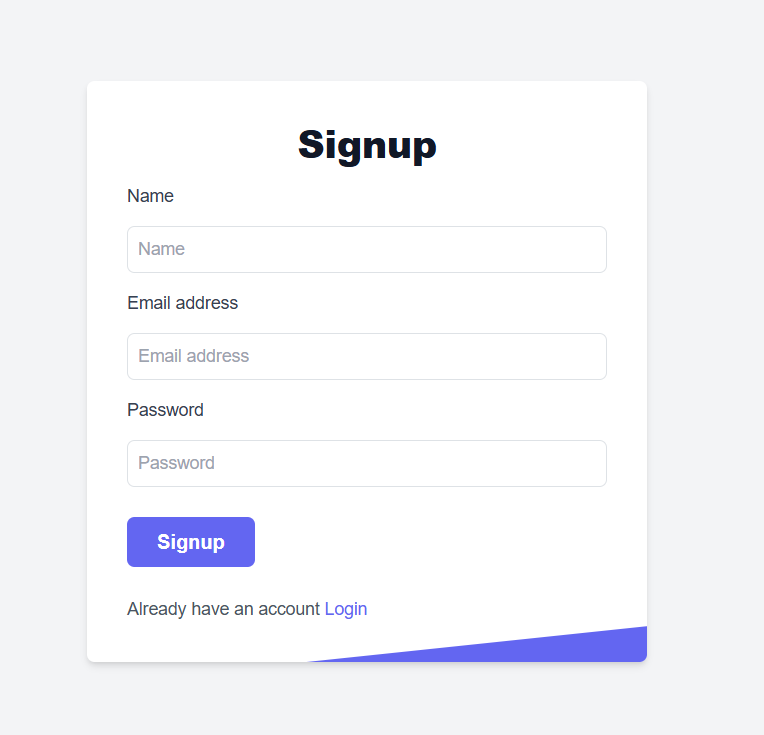
Roles are assigned as `user`, `seller`, or `admin`, with each role having specific permissions enforced via middleware.

**9. User Interface and Screenshot**

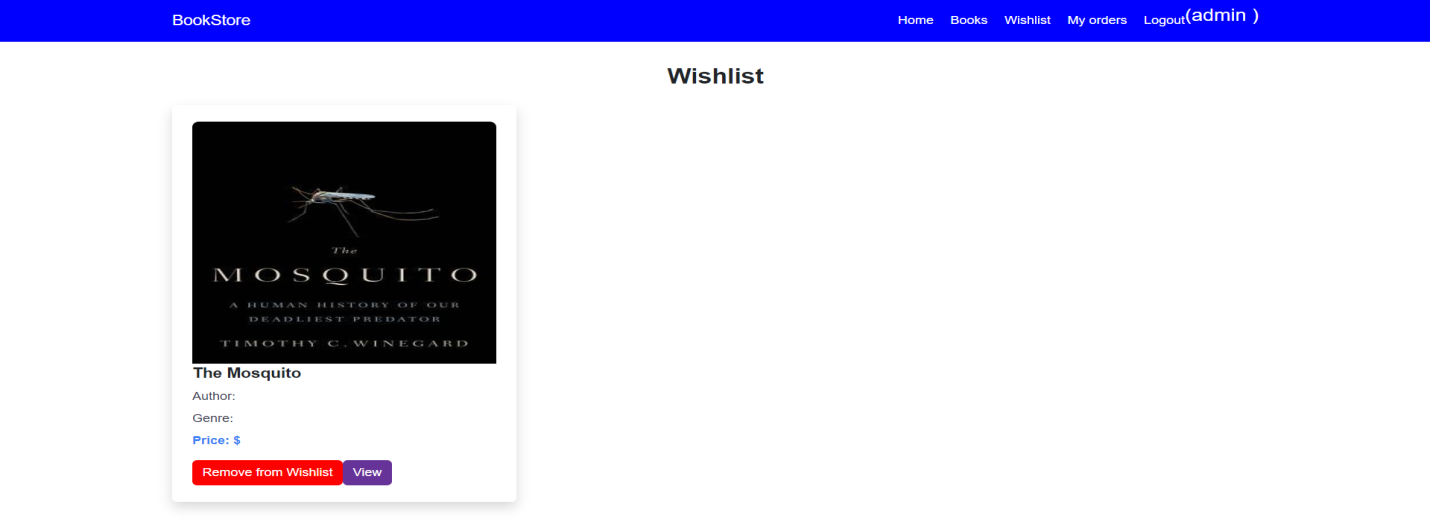
**Home Page**

**User Login**

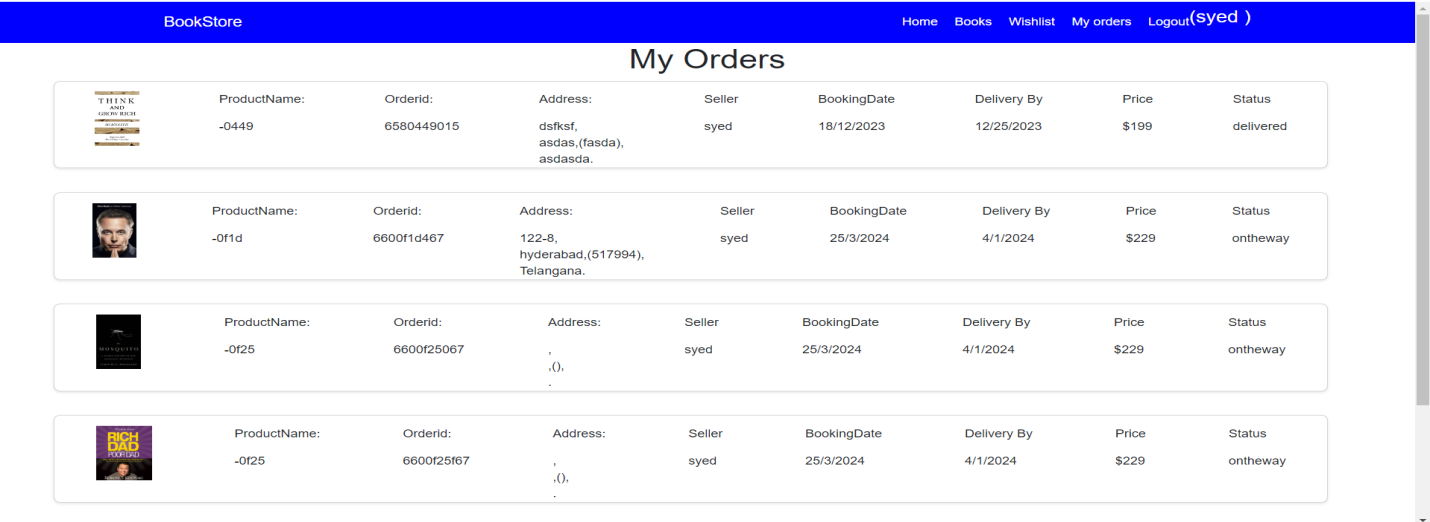


**User Signup**

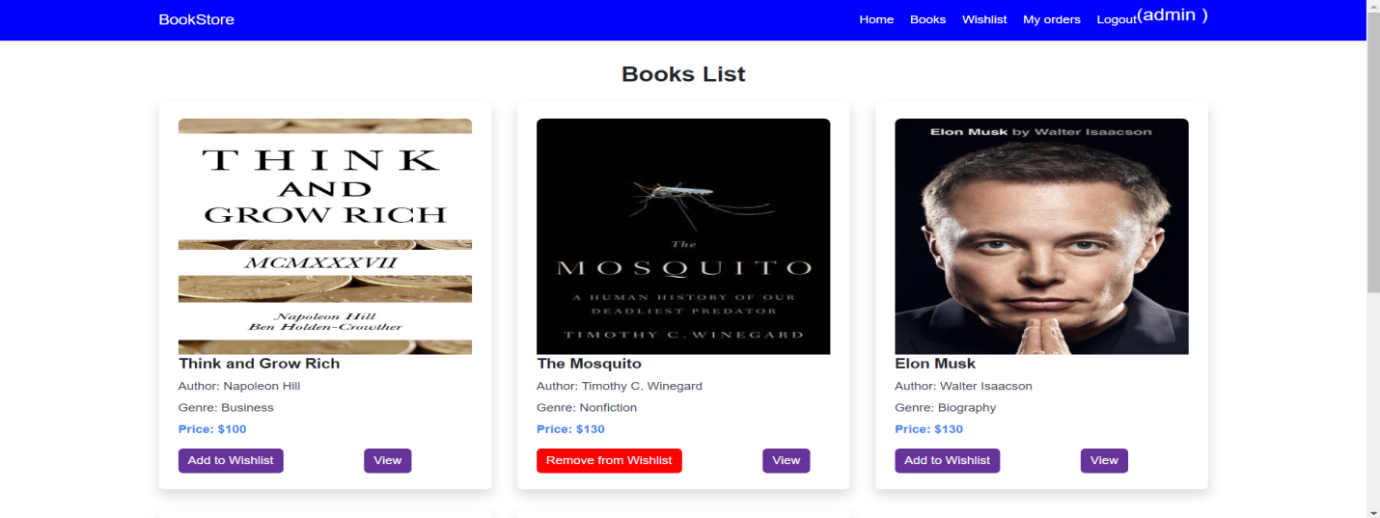
**Wishlist**



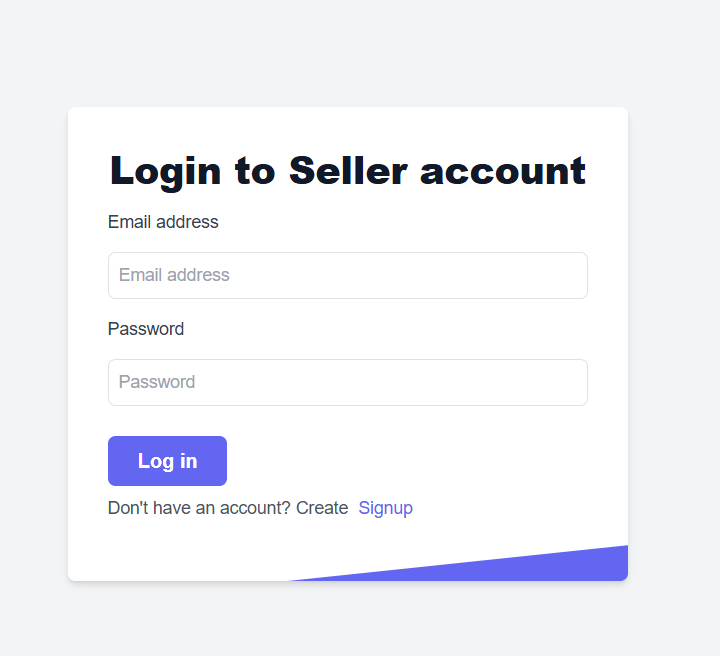
**Order**



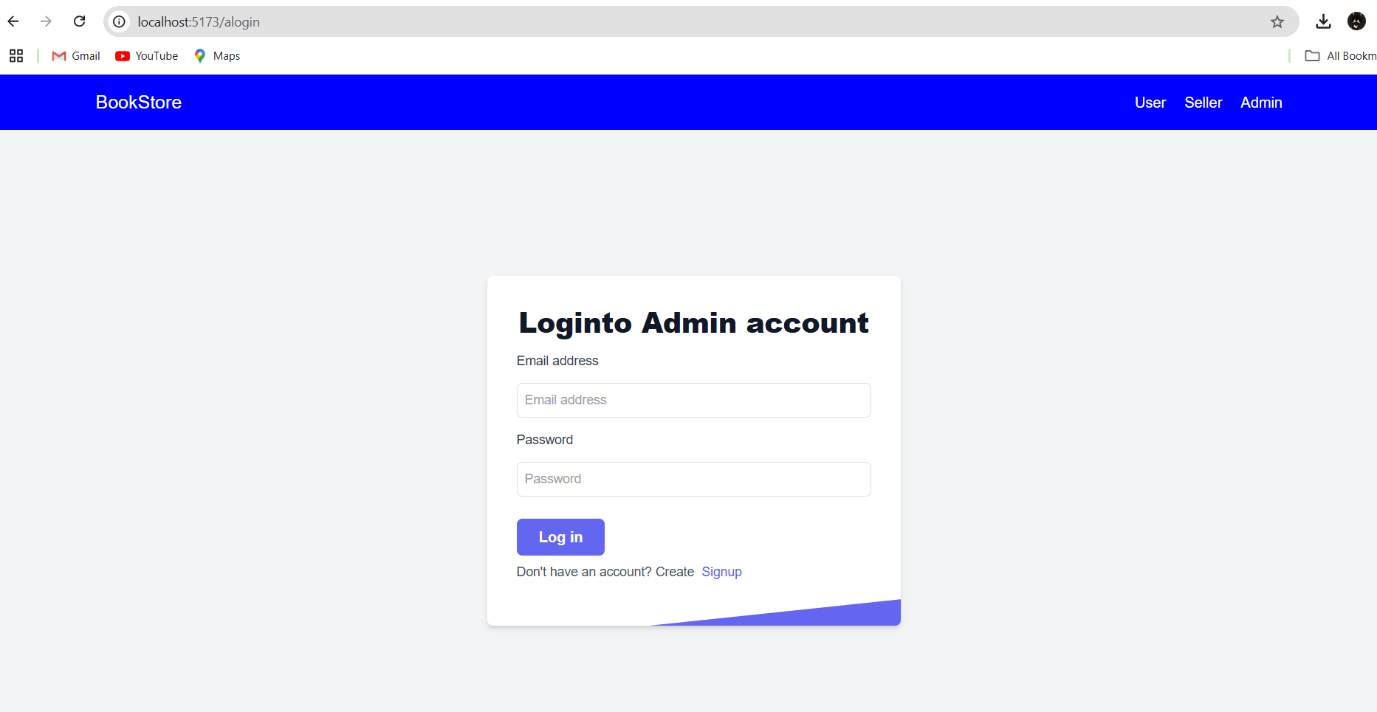
**Book List**



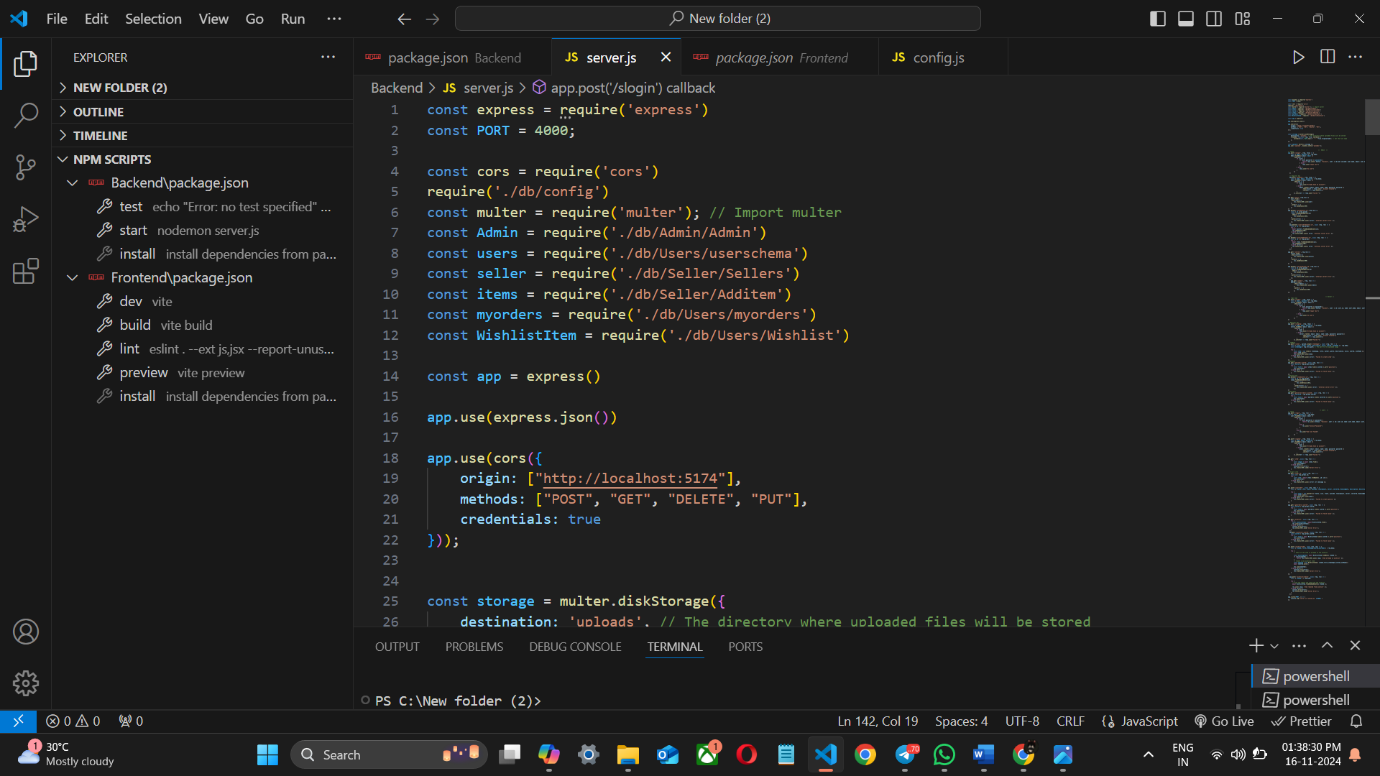
**Seller Login Page**



**Admin Login Page**



**Project Structure**



**11. Testing**

* Unit tests for individual components and functions using Jest.
* Integration tests for API endpoints using Mocha and Chai.
* UI testing with Cypress to ensure a seamless user experience.

**12. Known Issues**

* Adding more books in wishlist.
* Mobile layout needs further optimization for smaller screens.

**13. Future Enhancements**

* Immersive shopping experience
* AI support for guiding user and seller.