**1. Introduction**

**Project Title:**

**BOOK STORE**

**Team Members:**

1. **SURENDHAR, YOKESH** - Frontend Developer
2. **SYED IBRAHIM** - Backend Developer
3. **ASHOK KUMAR**  - Database Architect

**2. Project Overview**

**Purpose:**

The **Book Store** project aims to provide a user-friendly platform where users can browse, search, and purchase books. The application caters to both customers and sellers, ensuring seamless management of book listings, orders, and payments.

**Features:**

1. **User Registration/Login:** Secure user authentication with role-based access (Admin, Seller, Customer).
2. **Book Catalog:** Browse books by categories, authors, or keywords.
3. **Cart Management:** Add books to a cart for purchase.
4. **Order Management:** View order history and track status.
5. **Admin Panel:** Manage users, orders, and inventory.
6. **Responsive Design:** Optimized for both desktop and mobile devices.

**3. Architecture**

**Frontend:**

* Built with **React.js**, ensuring a responsive and dynamic UI.
* Components are modular, reusable, and structured using React Router for routing.

**Backend:**

* Developed using **Node.js** and **Express.js**, ensuring scalable and efficient API endpoints.
* RESTful APIs handle CRUD operations, authentication, and business logic.

**Database:**

* **MongoDB** is used for storing user information, books, and orders.
* Mongoose ORM provides schema validation and easy database interactions.

**4. Setup Instructions**

**Prerequisites:**

1. **Node.js** (v14 or above)
2. **MongoDB** (Local or Atlas instance)
3. **npm** (v6 or above)

**Installation:**

1. Clone the repository:

git clone https://github.com/username/bookstore.git

cd bookstore

1. Install dependencies:

cd client && npm install

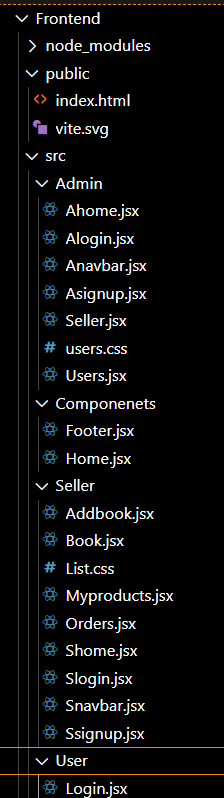
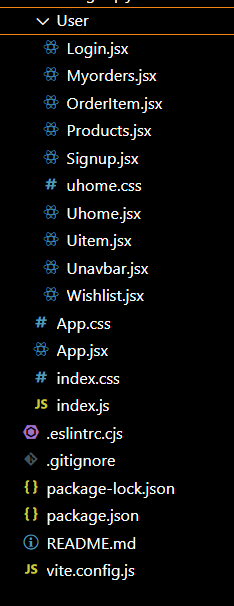
cd ../server && npm install

1. Set up environment variables:  
   Create .env in the server directory with:

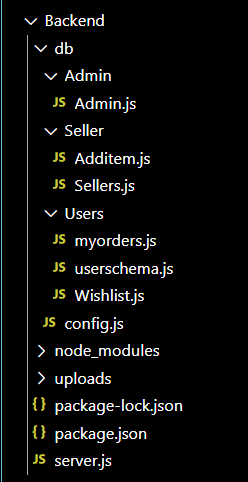
MONGO\_URI=mongodb+srv://username:password@cluster.mongodb.net/bookstore

**5. Folder Structure**

**Client (React Frontend):**

**Server (Node.js Backend):**



**6. Running the Application**

**Frontend:**

Run the React development server:

cd client

npm start

**Backend:**

Run the Node.js server:

cd server

npm start

**7. API Documentation**

#### **User Authentication**

* **POST /login**  
  **Description:** Authenticate the user and return a JWT token.  
  **Request Body:**

{

"email": "user@example.com",

"password": "securepassword"

}

**Response:**

{

"status": "Success",

"token": "jwt\_token",

"user": {

"id": "12345",

"role": "customer"

}

}

* **POST /signup**  
  **Description:** Register a new user.  
  **Request Body:**

{

"firstname": "John",

“lastname”: “doe”

"email": "user@example.com",

"password": "securepassword"

}

**Response:**

{

"status": "Success",

"message": "User registered successfully"

}

#### **Books Management**

* **GET /books**  
  **Description:** Retrieve all books available in the store.  
  **Response:**

[

{

"id": "12345",

"title": "The Great Gatsby",

“genre”: “novel”

"author": "F. Scott Fitzgerald",

"price": 10.99,

},

{

"id": "67890",

"title": "1984",

“genre”: “fiction”

"author": "George Orwell",

"price": 12.99,

}

]

* **POST /books**  
  **Description:** Add a new book to the inventory (Admin/Seller only).  
  **Request Body:**

{

"title": "New Book",

"author": "Author Name",

"price": 14.99,

"stock": 20

}

**Response:**

{

"status": "Success",

"message": "Book added successfully"

}

#### **Order Management**

* **POST /order**  
  **Description:** Place an order for the books in the cart.  
  **Response:**

{

"status": "Success",

"message": "Order placed successfully",

"orderId": "98765"

}

* **GET /orders**  
  **Description:** Retrieve the user's past orders.  
  **Response:**

[

{

"orderId": "98765",

"items": [

{

"title": "The Great Gatsby",

"quantity": 1

}

],

"totalPrice": 21.98,

"status": "Delivered"

}

]

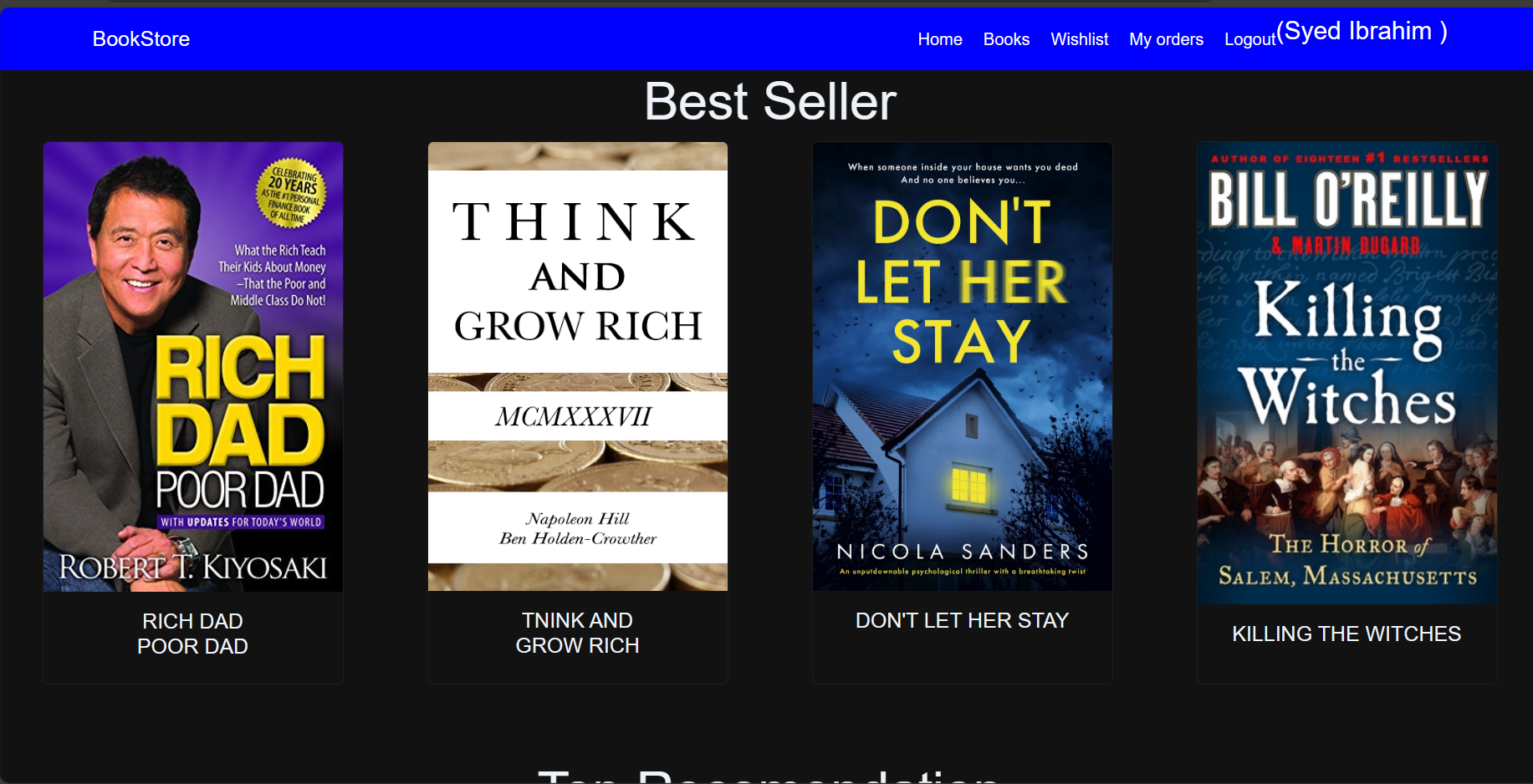
**8. Authentication**

* **JWT Authentication:**
  + Tokens are generated on login and used for protected routes.
  + Middleware validates tokens and grants access based on user roles.

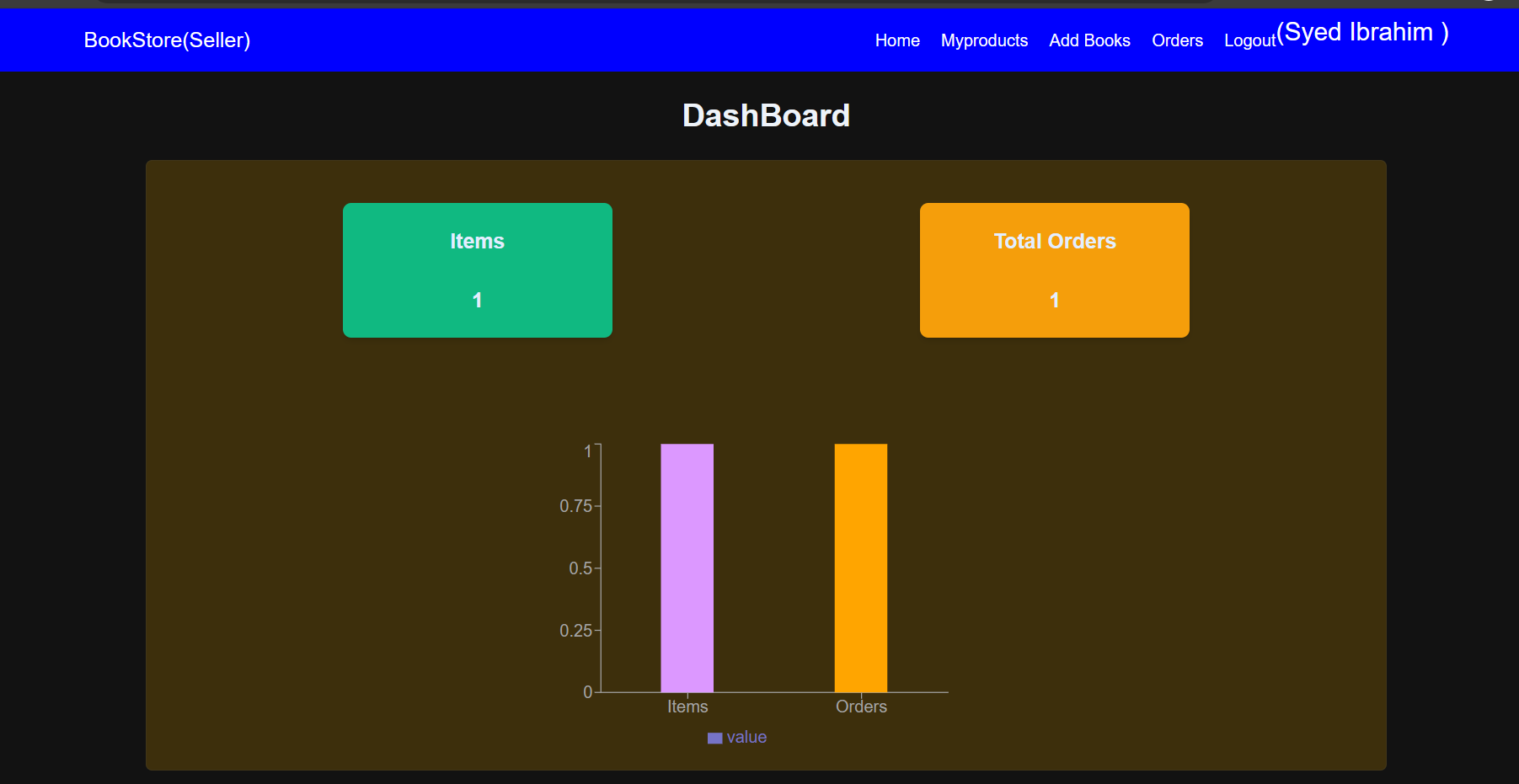
**9. User Interface**

* **Login Page:** User authentication with visual cues for errors.

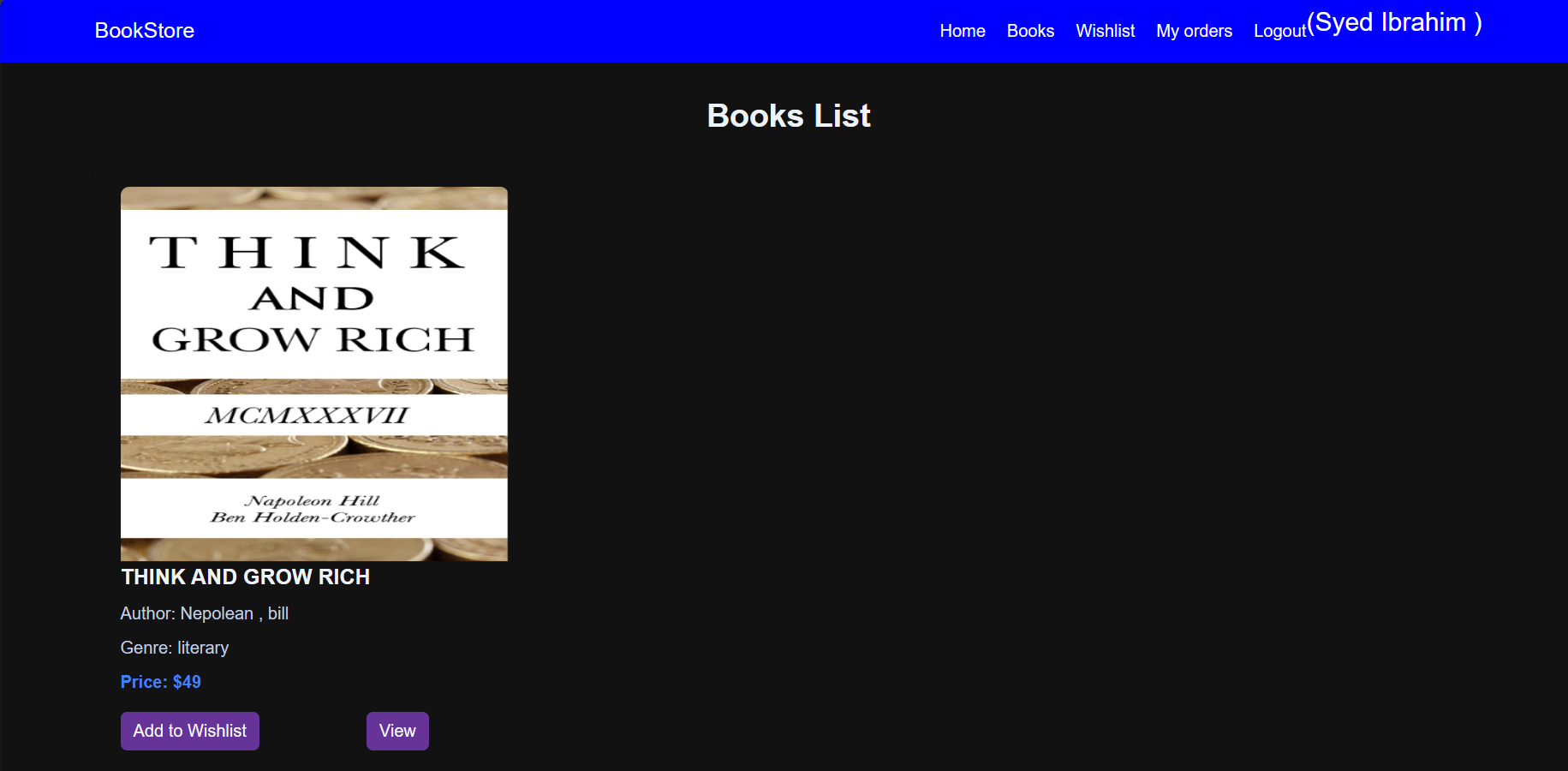
USER HOME PAGE



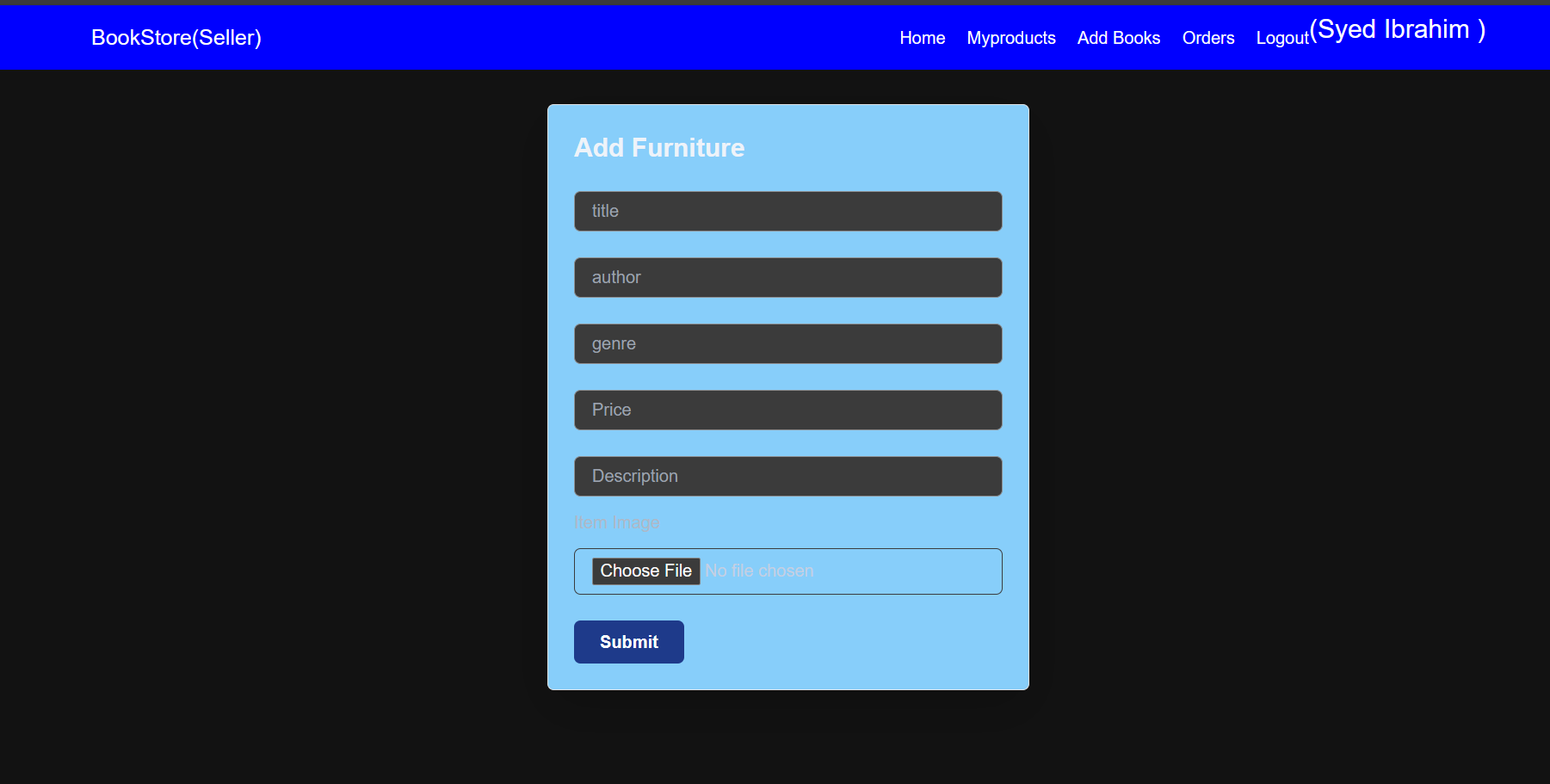
SELLER HOMEPAGE



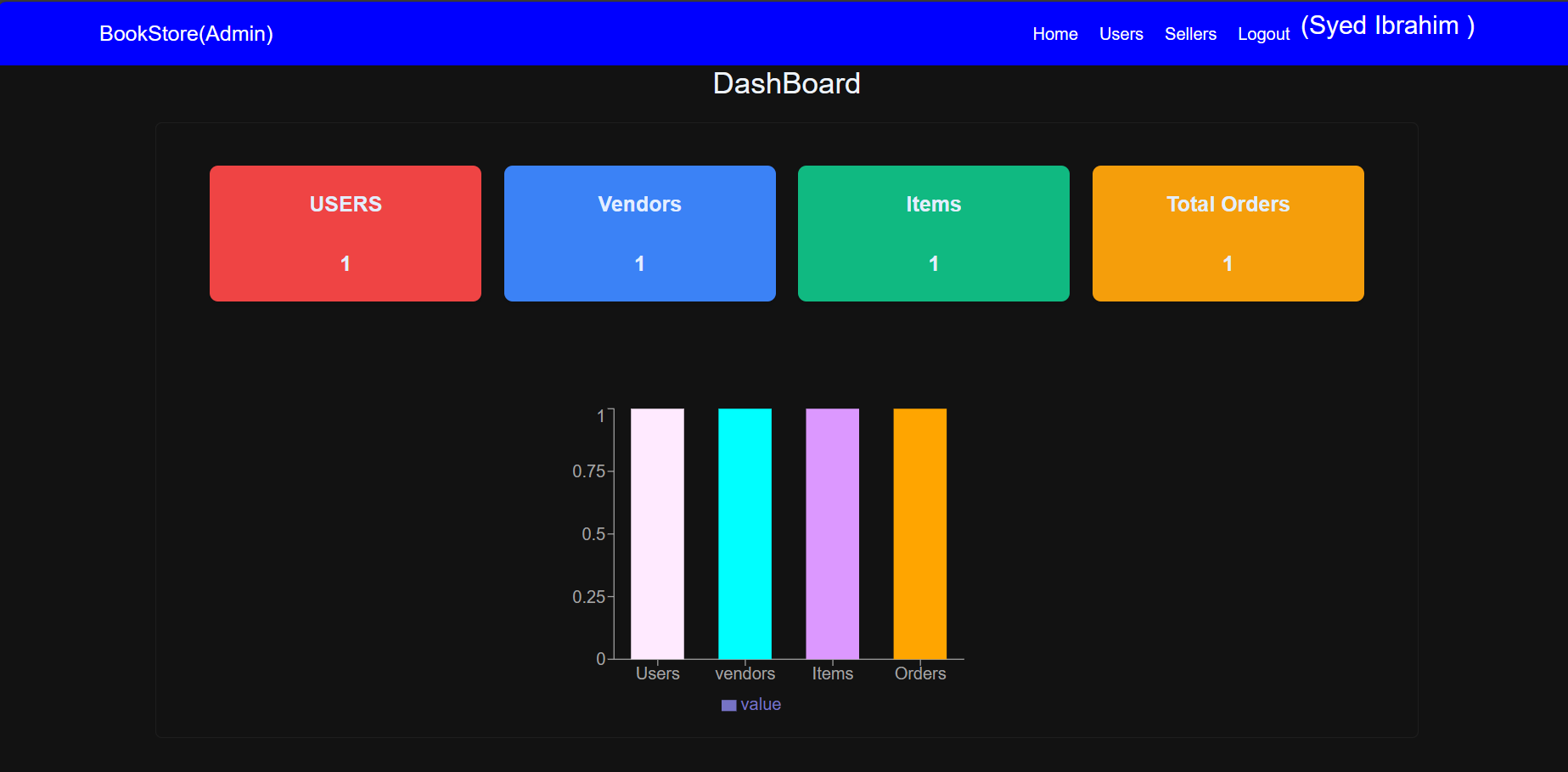
BOOK LIST VIEW



ADD BOOKS



* **Admin Dashboard:** User and inventory management panels.



**10. Testing**

**Strategy:**

* **Unit Tests:** Using **Jest** for testing React components and API endpoints.
* **Manual Testing:** Verifying user workflows such as login, adding books, and placing orders.

**11. Screenshots or Demo**

**Demo video available :** [**book store demo drive link**](https://drive.google.com/drive/folders/1Kk30GyCuYbAwvtC41kR_fkeU1W0sz71f?usp=drive_link)

**12. Known Issues**

1. **Error handling for large uploads:** File uploads larger than 10MB fail on certain devices.
2. **Mobile View:** Minor alignment issues on older devices.

**13. Future Enhancements**

1. **Payment Integration:** Add support for online payments via third-party services.
2. **Advanced Search:** Include filters for price range and publication year.