Full Stack Development with MERN

1. Introduction

- **Project Title:** Book Store App(Tales and Tomes)
- **Team Members:**
    - **Nishanth SPK –** Full Stack Developer: Handles both client and server development, ensuring smooth integration between frontend and backend components while implementing core features
    - **Rahul Ragavendar -** Backend Developer: Specializes in backend development, focusing on creating and managing server routes, implementing logic with Express, and integrating APIs.
    - **Nithish Kumar -** Frontend Developer: Concentrates on frontend development with React, focusing on user interface design, creating interactive elements, and enhancing user experience.
    - **Sanjay -** Database Administrator: Manages MongoDB databases, focusing on schema design, efficient data storage, and query optimization to ensure smooth data handling.
    - **Vinodhini S -** Quality Assurance and Deployment Manager: Responsible for testing and debugging the application, ensuring its reliability, and overseeing the deployment process to production.

    This Book Store application was developed as part of a full-stack development project using the MERN (MongoDB, Express.js, React, Node.js) stack. The project simulates a real-world e-commerce application specifically tailored for books, allowing users to browse, manage, and purchase books seamlessly online.

2. Project Overview

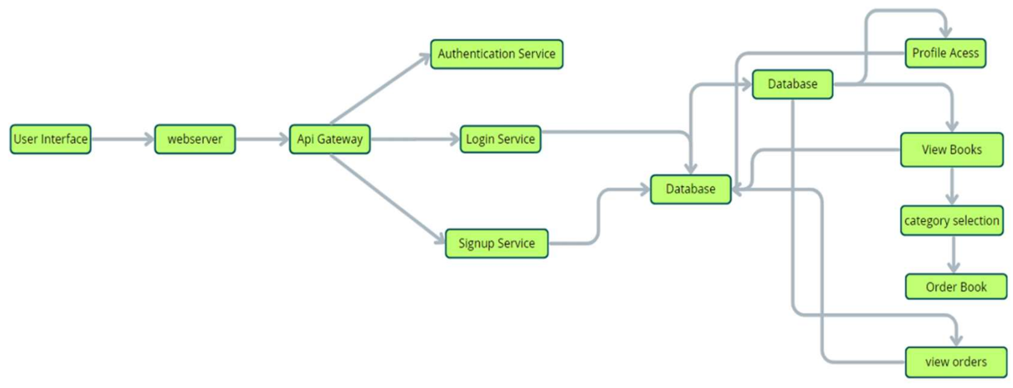
- **Purpose:**
  The primary goal of the Book Store application is to provide an efficient and user-friendly platform for users to browse and purchase books from an extensive collection. The application incorporates a

well-structured user interface, secure payment processing, and robust data management to deliver a comprehensive shopping experience for book enthusiasts.

- **Features:**
  - **User Registration & Authentication:** Users can register and log in securely. JWT tokens are used to manage sessions.
  - **Book Browsing & Search:** Users can browse through various categories of books or search by title, author, or genre.
  - **Shopping Cart & Checkout:** Users can add selected books to their cart, modify quantities, and proceed to checkout.
  - **Order Management:** Track and view past purchases with order details.
  - **Admin Functionality:** Admins can add, update, or remove books, manage inventory, and oversee order history.

Responsive Design: Optimized for both desktop and mobile viewing to enhance accessibility across devices.

3. Architecture



Product Flow

- **Frontend:**
  Developed using React, the frontend is designed with reusable components for ease of maintenance and scalability. Libraries like Redux are used to manage the application's state, ensuring smooth user interactions and efficient data handling across components.
- **Backend:**
  Node.js and Express.js form the backend foundation, handling API requests, processing business logic, and managing user sessions. Authentication and authorization are enforced through JWT, and middlewares are utilized to handle errors and route protection effectively.
- **Database:**
  MongoDB stores user data, book catalog information, orders, and transaction history. The document-oriented nature of MongoDB allows for flexible data modeling, particularly useful for handling complex relationships between users, books, and orders. Schemas include:
  - **User Schema:** Stores user credentials, role (user/admin), and order history.
  - **Book Schema:** Stores book metadata (title, author, genre, price, stock).
  - **Order Schema:** Links user purchases with book items, including quantity and order status.

4. Setup Instructions

- **Prerequisites:**

  Node.js (version X.X.X)

  MongoDB (version X.X.X)

  Git for cloning the repository

- **Installation:**

1. Clone the repository:

   ```
   git clone https://github.com/nishanthspk/BOOK-STORE.git
   ```

2. Navigate to the project directory and install dependencies for both the client and server:

   ```
   cd bookstore
   npm install
   ```

3. Configure environment variables for MongoDB and JWT secrets.

5. Folder Structure

- **Client (Frontend):**
  The client folder is structured by pages, components, and services, making it modular and maintainable. Key folders include:
  - **Components:** Contains UI components such as book cards, headers, and footers.
  - **Pages:** Houses the main views (Home, Book Details, Cart, Checkout).
  - **Services:** API services that connect the frontend to backend endpoints for CRUD operations.
- **Server (Backend):**
  - **Controllers:** Manages the core logic for each route (e.g., user login, book retrieval).
  - **Routes:** Defines all endpoints (e.g., `/api/books`, `/api/users`).
  - **Models:** Holds schemas for MongoDB collections, including User, Book, and Order.
  - **Middlewares:** Handles authentication checks, error handling, and request validation.

6. Running the Application

Running the Application Locally

To start both the frontend and backend servers, follow these steps:

Frontend

1. Navigate to the frontend directory: cd book-store\frontend

2. Start the frontend server: npm start This command will run the frontend in development mode, typically on http://localhost:3000.

Backend

1. Open a new terminal window and navigate to the backend directory: cd book-store\backend
2. Start the backend server: npm start.

This will start the backend server on the port defined in your .env file (e.g., http://localhost:4000). With both servers running, the frontend will communicate with the backend, allowing full functionality for the Book-Store app.

7. API Documentation

- **Endpoints:**
  - **GET /api/books -** Retrieves a list of books with optional search parameters.
  - **POST /api/users/register** - Registers a new user.
  - **POST /api/auth/login** - Authenticates and generates a JWT token for the user.
  - **POST /api/cart** - Adds a book to the user's shopping cart.
  - **GET /api/orders** - Returns the order history for the logged-in user.

- **Example Responses:**

Success Response:

```
{
   "status": "success",
   "data": {
      "books": [
         {
            "title": "Sample Book",
            "author": "Author Name",
            "price": 15.99
         }
      ]
   }
}
```

Error Response:

```
{
   "status": "error",
   "message": "Book not found."
}
```

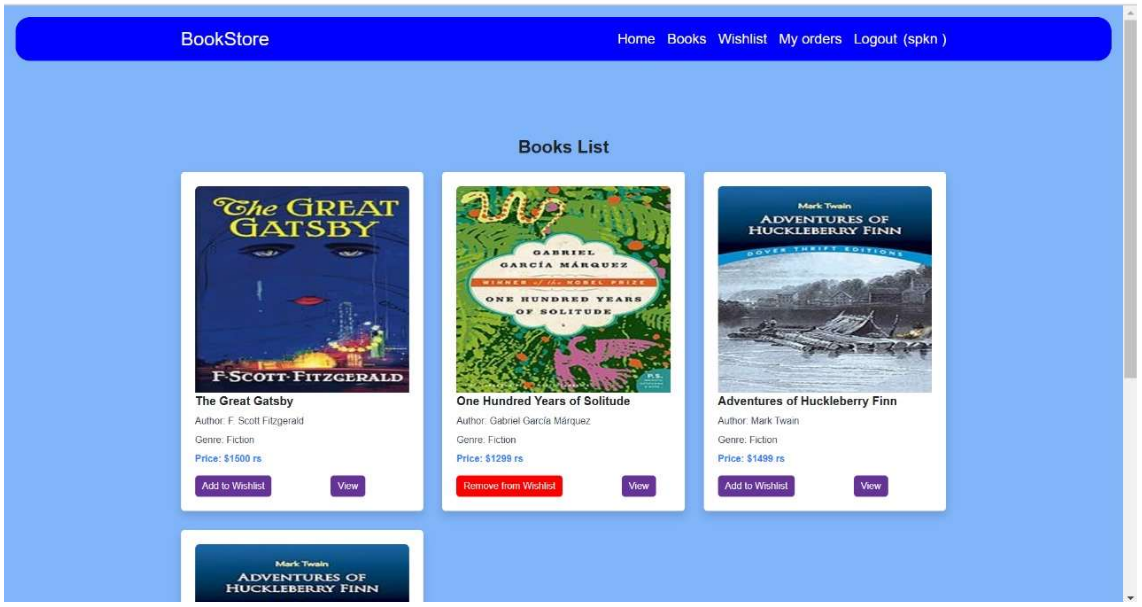
8. Authentication

The project employs JWT tokens to handle user authentication. Users receive a token upon login, which is stored in local storage for frontend access. Every API call requiring authentication includes this token in the headers for authorization.

9. User Interface

- **Homepage** showing featured and popular books

- **Book Detail Page** with book descriptions, reviews, and add-to-cart functionality



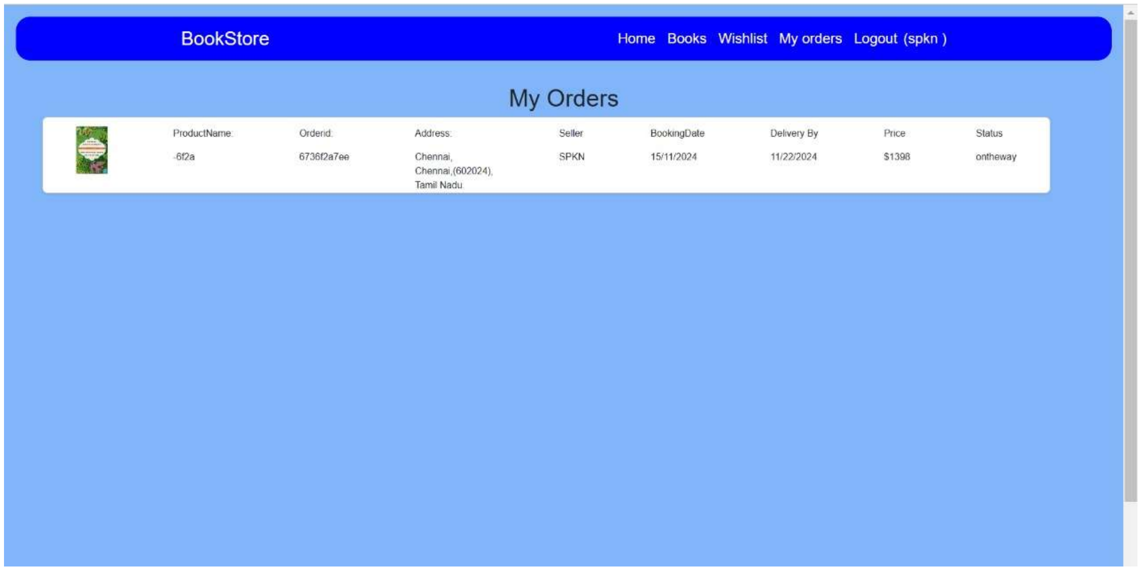
Books List

The Great Gatsby
Author: F. Scott Fitzgerald
Genre: Fiction
Price: $1500 rs
Add to Wishlist View

One Hundred Years of Solitude
Author: Gabriel García Márquez
Genre: Fiction
Price: $1299 rs
Remove from Wishlist View

Adventures of Huckleberry Finn
Author: Mark Twain
Genre: Fiction
Price: $1499 rs
Add to Wishlist View

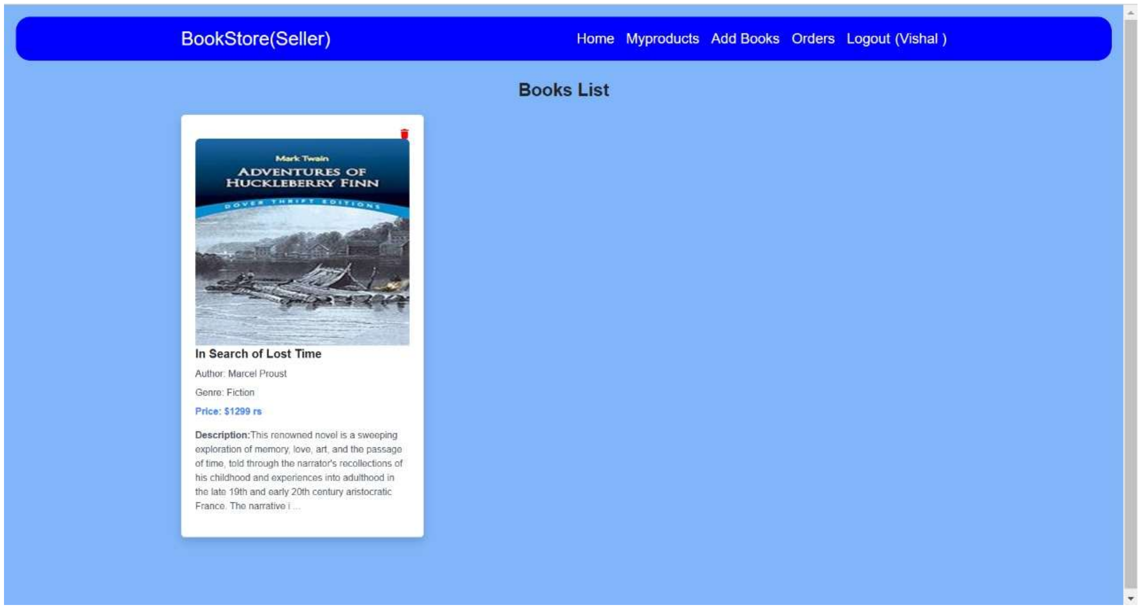
- **Cart Page** displaying selected items with quantity adjustments.



My Orders

	ProductName:	Orderid:	Address:	Seller	BookingDate	Delivery By	Price	Status
	-6f2a	6736f2a7ee	Chennai, Chennai,(602024), Tamil Nadu	SPKN	15/11/2024	11/22/2024	$1398	ontheway

- **Checkout Page** for reviewing items and completing purchases.


BookStore(Seller)
Home Myproducts Add Books Orders Logout (Vishal)
Mark Twain
Adventures of Huckleberry Finn
Dover Thrift Editions

Books List

In Search of Lost Time

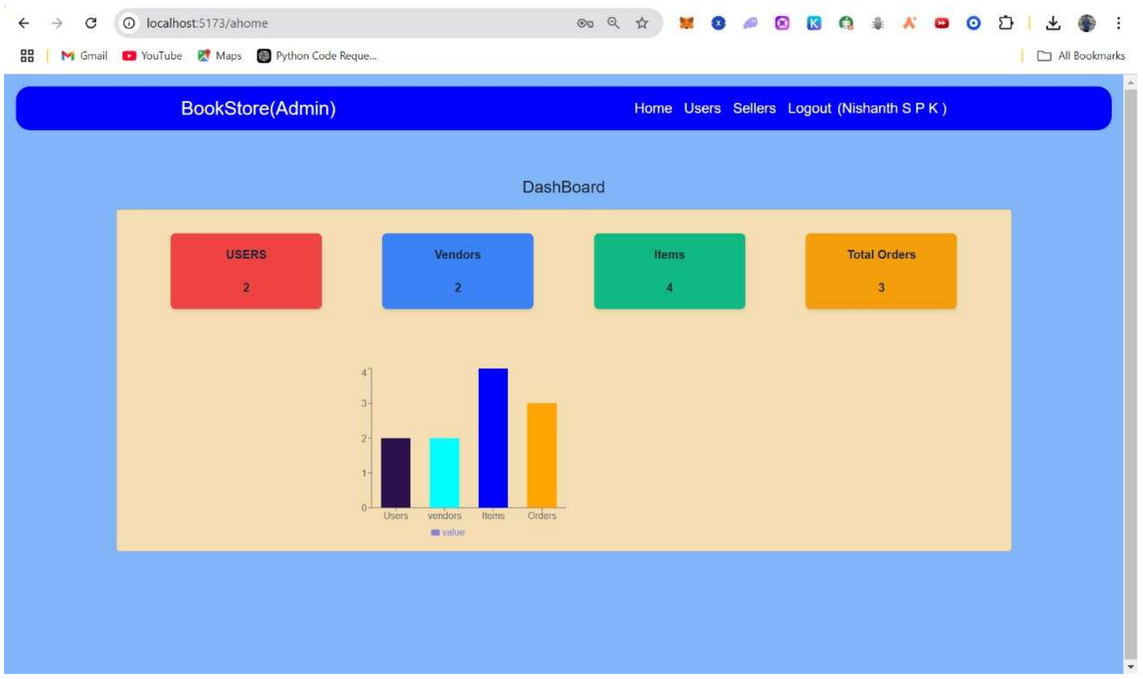
Author: Marcel Proust

Genre: Fiction

Price: $1299 rs

Description: This renowned novel is a sweeping exploration of memory, love, art, and the passage of time, told through the narrator's recollections of his childhood and experiences into adulthood in the late 19th and early 20th century aristocratic France. The narrative i ...

- **Admin Dashboard** with CRUD functionalities for book management.



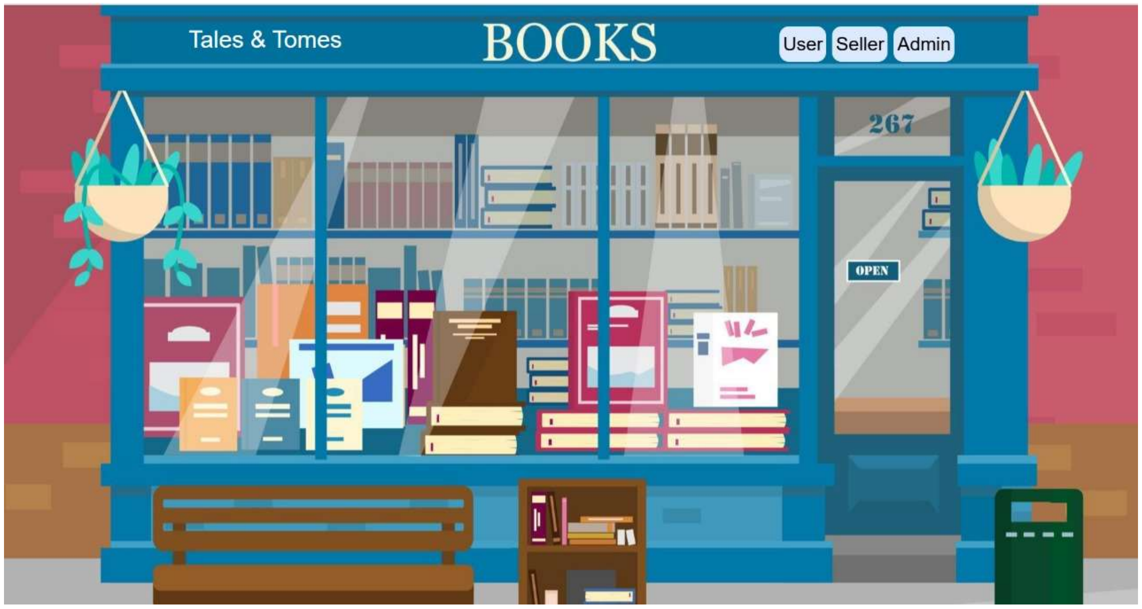
.

10. Testing

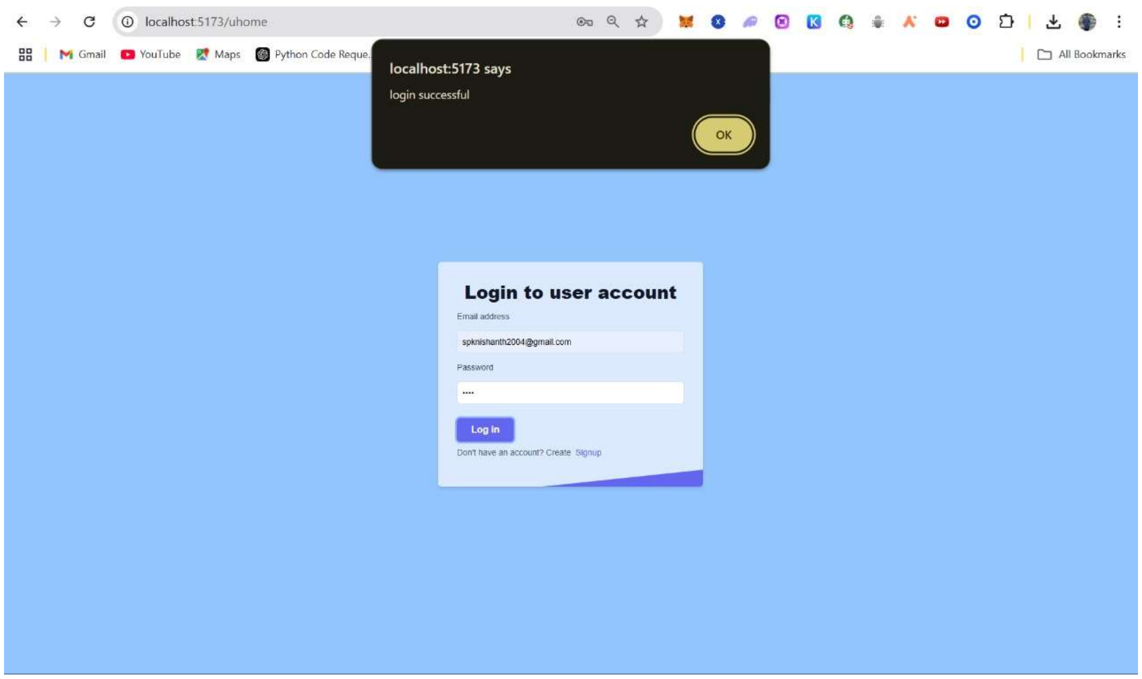
- **Tools Used:** Jest and React Testing Library for frontend component testing, and Mocha or Chai for backend API testing.
- **Testing Strategy:**
    - **Unit Testing:** Tests individual functions for book CRUD operations.
    - **Integration Testing:** Verifies interactions between frontend and backend components.
    - **End-to-End Testing:** Simulates user interactions from browsing to purchase, ensuring the entire flow functions as intended.

11.Screenshots or Demo

- **Homepage**



- **User Login**



localhost:5173 says

login successful

OK

Login to user account

Email address

spknishanth2004@gmail.com

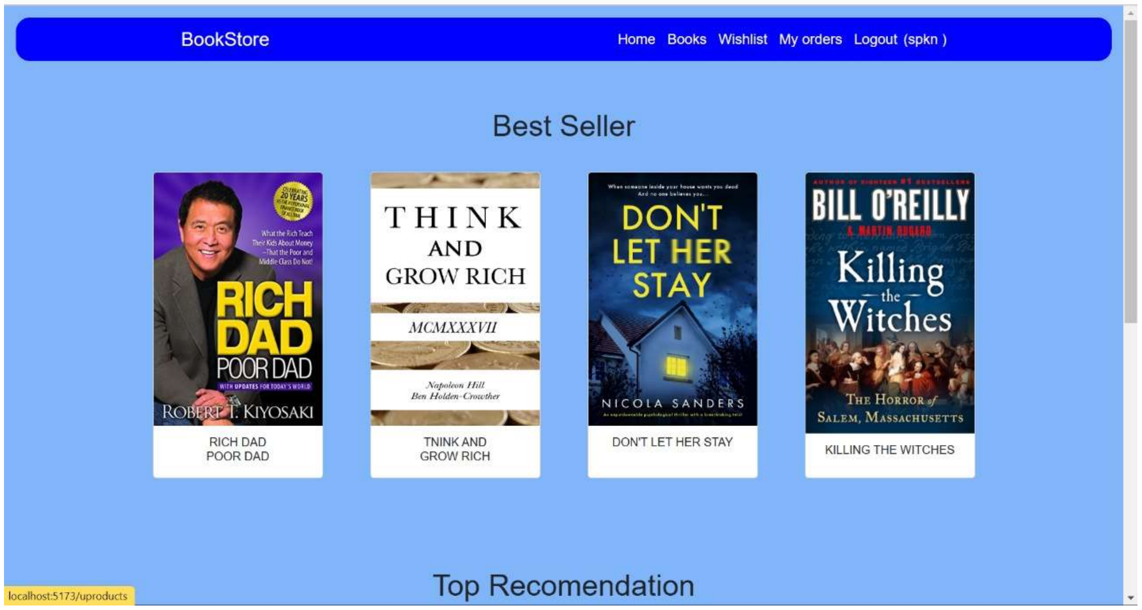
Password

....

Log in

Don't have an account? Create Signup

- **User HomePage**



Best Seller

RICH DAD POOR DAD

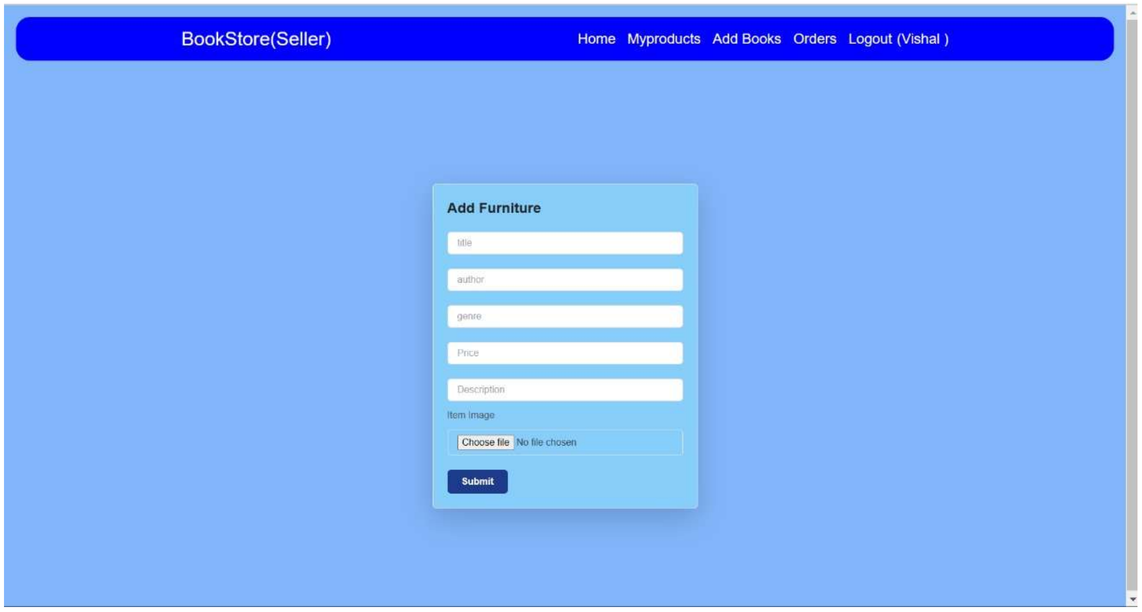
TNINK AND GROW RICH

DON'T LET HER STAY

KILLING THE WITCHES

Top Recomendation

- **Add Books**



Add Furniture

title

author

genre

Price

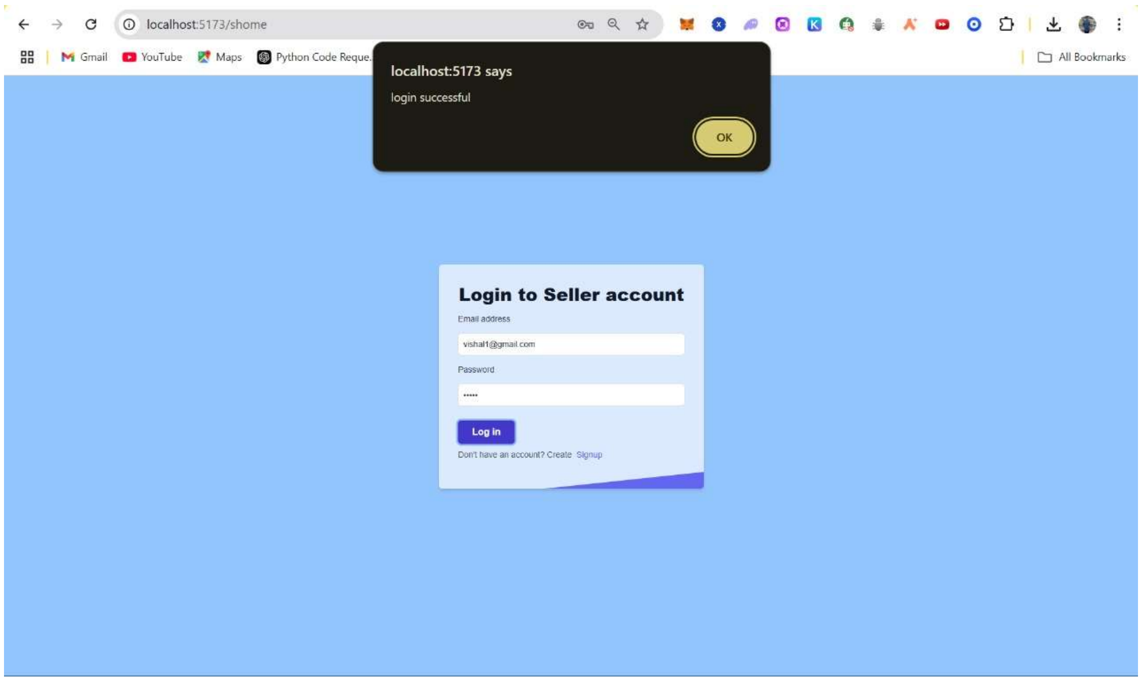
Description

Item image

Choose file No file chosen

Submit

- **Seller Login**


localhost:5173/shome

localhost:5173 says

login successful

OK

Login to Seller account

Email address

vishal1@gmail.com

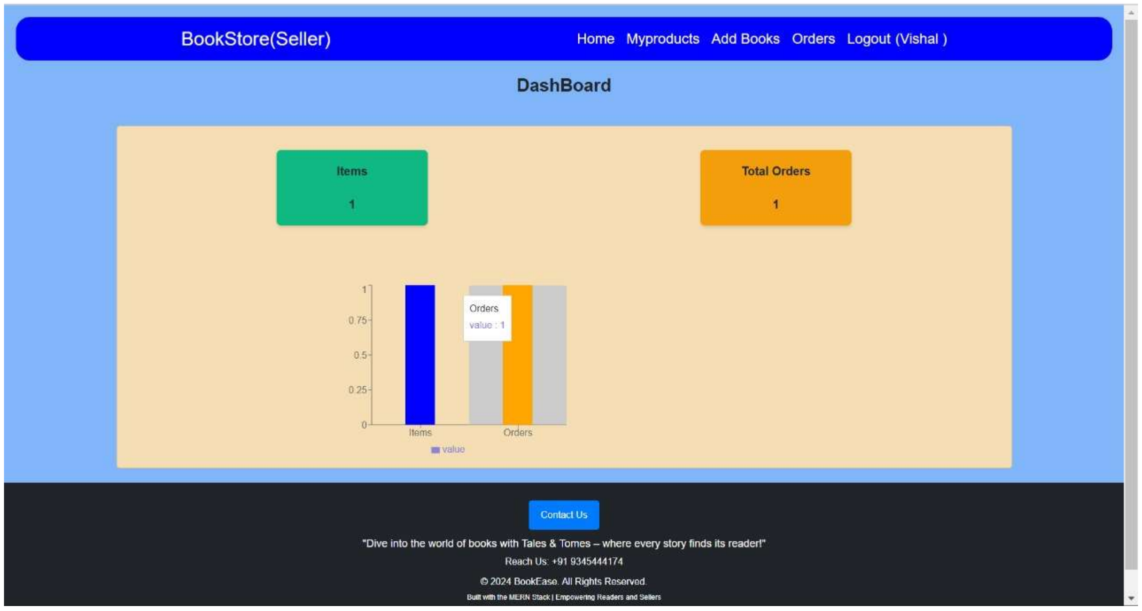
Password

•••••

Log in

Don't have an account? Create Signup

- **Seller Dashboard**



DashBoard

Items	Total Orders
1	1

Contact Us

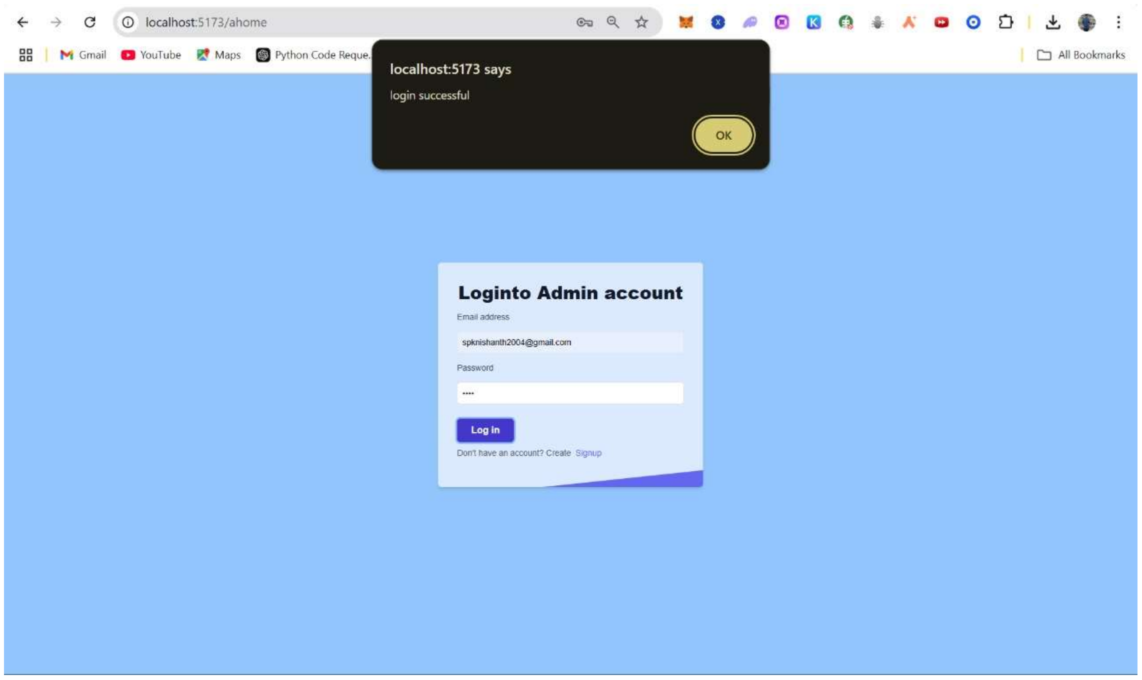
"Dive into the world of books with Tales & Tomes – where every story finds its reader!"

Reach Us: +91 9345444174

© 2024 BookEase. All Rights Reserved.

Built with the MERN Stack | Empowering Readers and Sellers

- **Admin Login**


localhost:5173/ahome

localhost:5173 says

login successful

OK

Loginto Admin account

Email address

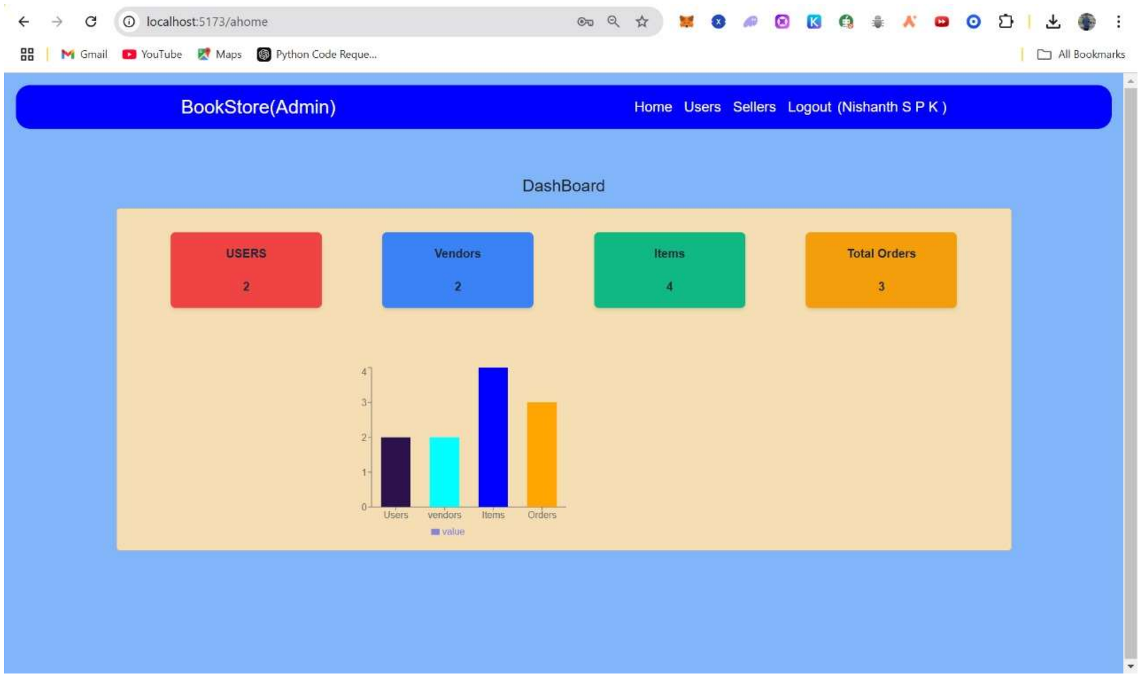
spknishanth2004@gmail.com

Password

Log in

Don't have an account? Create Signup

- **Admin Dashboard**

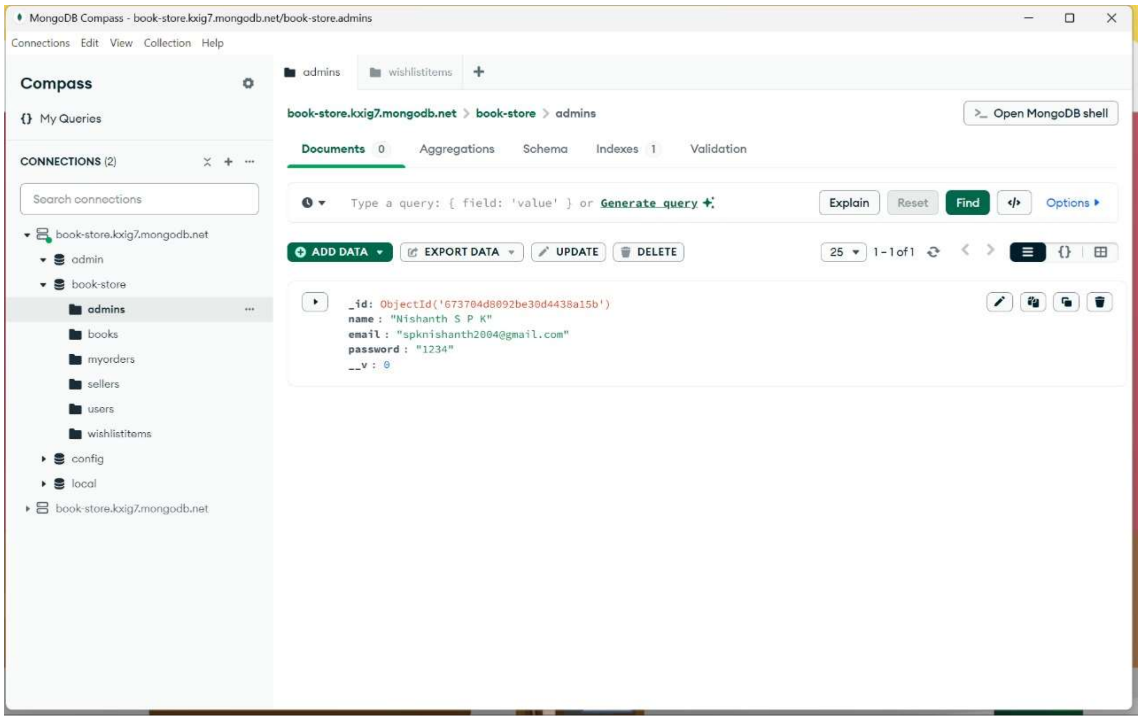

localhost:5173/ahome

BookStore(Admin) Home Users Sellers Logout (Nishanth S P K)

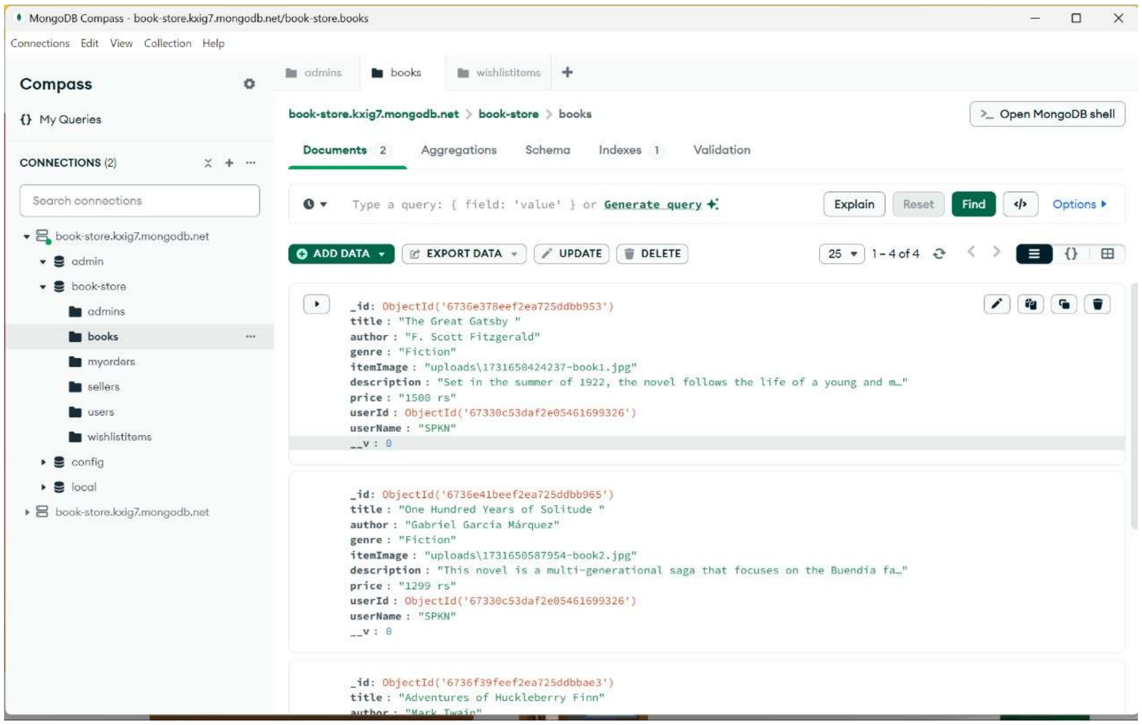
DashBoard

USERS	Vendors	Items	Total Orders
2	2	4	3

- **Database Implementation**



```
_id: ObjectId('673704d8092be30d4438a15b')
name : "Nishanth S P K"
email : "spknishanth2004@gmail.com"
password : "1234"
__v : 0
```



```
_id: ObjectId('6736e378eef2ea725ddbb953')
title : "The Great Gatsby "
author : "F. Scott Fitzgerald"
genre : "Fiction"
itemImage : "uploads\1731650424237-book1.jpg"
description : "Set in the summer of 1922, the novel follows the life of a young and m…"
price : "1500 rs"
userId : ObjectId('67330c53daf2e05461699326')
userName : "SPKN"
__v : 0

_id: ObjectId('6736e41beef2ea725ddbb965')
title : "One Hundred Years of Solitude "
author : "Gabriel García Márquez"
genre : "Fiction"
itemImage : "uploads\1731650587954-book2.jpg"
description : "This novel is a multi-generational saga that focuses on the Buendía fa…"
price : "1299 rs"
userId : ObjectId('67330c53daf2e05461699326')
userName : "SPKN"
__v : 0

_id: ObjectId('6736f39feef2ea725ddbbae3')
title : "Adventures of Huckleberry Finn"
author : "Mark Twain"
```

Demo Video Link :

https://drive.google.com/file/d/1owsKEhpBUeeRzlCPokni1fEflEybzCVe/view?usp=drivesdk

12. Known Issues

- **Slow Response Time:** Some pages may take longer to load due to heavy data processing in specific routes.
- **Limited Search Functionality:** Current search options are limited to basic keyword matches.
- **Payment Gateway:** A real payment gateway integration is pending, with current checkout limited to simulated transactions.
- **Session Expiry Handling:** If a user's session expires mid-action, they may encounter errors without immediate redirection to the login page, potentially causing confusion.
- **Limited Mobile Optimization:** While the app is accessible on mobile devices, certain UI elements are not fully responsive, which may impact usability on smaller screens.
- **Inventory Management:** Inventory tracking is basic, and there's no automated notification system to alert when stock is low or out of stock.
- **Rate Limiting:** The application lacks rate limiting, which could make it vulnerable to spam or DoS attacks from multiple API requests in a short period.

13. Future Enhancements

Potential features for future versions:

- **Improved Search Filters:** Enable more advanced filters by genre, author, price, and popularity.
- **User Reviews and Ratings:** Allow users to rate books and leave reviews, enhancing user engagement.
- **Integration with a Payment Gateway:** Implement a live payment solution like Stripe or PayPal to allow secure payments.
- **Wish List Feature:** Enable users to save books for future purchase consideration.
- **Push Notifications:** Notify users of new arrivals, promotions, or restocked books.
- **Enhanced Book Recommendations:** Use machine learning algorithms to recommend books based on users' reading history, preferences, and trending titles, creating a more personalized experience.
- **Gift Cards and Vouchers:** Add a feature allowing users to purchase and redeem gift cards or vouchers, which can also be shared with friends and family.
- **Loyalty Rewards Program:** Implement a loyalty points system where users earn points on purchases and can redeem them for discounts or exclusive offers.
- **Offline Access for E-books:** Allow users to download e-books purchased through the app for offline reading, improving usability for mobile readers.
- **Social Sharing:** Enable users to share their favorite books or recommendations on social media platforms directly from the app, boosting brand visibility.
- **In-App Chat Support:** Integrate a chat support feature to provide real-time assistance for users, helping with order inquiries, book suggestions, or technical support.
- **Enhanced Order Tracking:** Offer real-time order tracking, where users can see their order status, shipping details, and estimated delivery dates.