Full Stack Development with MERN

1. Introduction

Welcome to the literary haven of the digital age—introducing our revolutionary BookNest Application, a masterpiece crafted 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.

Tailored for the modern book enthusiast, our platform blends robust functionality with an intuitive interface. From discovering new releases to revisiting timeless classics, it promises an immersive experience. MongoDB fuels scalable data infrastructure, Express.js supports responsive routing, Node.js delivers performance, and React offers a visually rich interactive interface across devices.

Project Title: BOOKNEST-Where Stories Nestle

Team ID: LTVIP2025TMID59395

Team Leader: Pothuganti Sriasha

Team Members:

- Palla Geetha Varshini

- Nunavath Kamal

- Nemalikanti Deena

Everyone is involved in all the phases of development.

2. Project Overview

Purpose:

BookNest is an innovative online bookstore developed using the MERN stack. It offers readers a seamless experience in exploring, discovering, and purchasing books while providing sellers and administrators tools for inventory and order management.

Features:

- - User Registration and Authentication
- - Book Listings and Category Browsing
- - Secure Checkout and Order History

- - Admin Dashboard for Order and Book Management
- - Seller Panel for Listing and Inventory Control
- - API Integration for Payment and Shipping
- - Analytics for Book Sales and Popularity Trends

3. Architecture

Frontend:

- - React.js for interactive UI components
- - React Router for navigation
- - Axios for HTTP communication - Bootstrap for styling and responsiveness

Backend:

- - Node.js for runtime environment
- - Express.js for server-side routing and APIs
- - MongoDB with Mongoose for NoSQL data handling
- - JWT for authentication and authorization
- - Bcrypt for password encryption

4. Setup Instructions

Prerequisites:

- - Node.js and npm
- - MongoDB (local or cloud)
- - Git
- - React.js environment via Vite

Installation Steps:

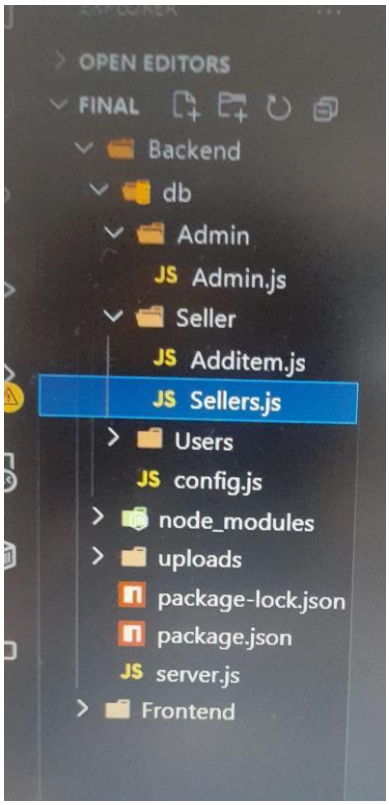
- - Clone the repository
- - Install backend dependencies: `npm install`
- - Install frontend dependencies: `npm install` in /client
- - Run backend: `nodemon app.js`
- - Run frontend: `npm run dev`

5. Folder Structure

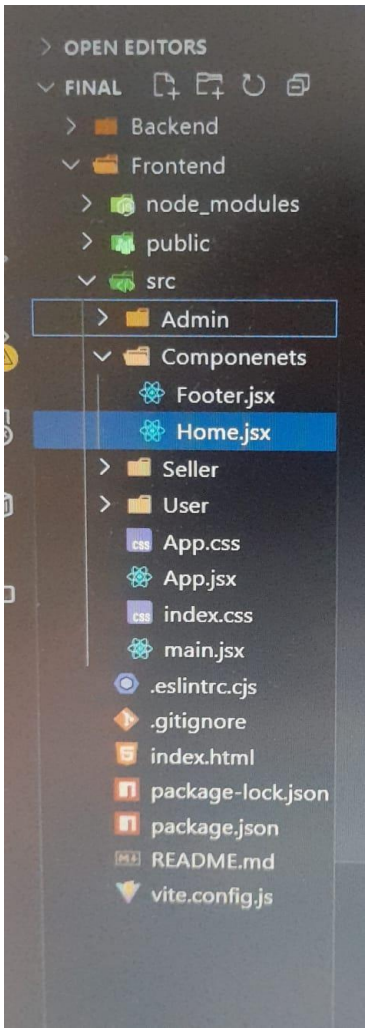
Backend:

- /Backend/db
- /Backend/db/Admin
- /Backend/db/Seller

- /Backend/db/User



Frontend:



- - / Frontend
- - /Frontend/src/components
- - /Frontend/src/Seller
- - /Frontend/src/Admin
- - /Frontend/src/User □ - /Frontend/src/context

6. Running the Application

- - Frontend: npm run dev (inside /client)
- - Backend: node server.js (inside /server)

7. API Documentation

User Authentication:

- - POST /api/auth/register – Register a new user □ - POST /api/auth/login – Login and receive JWT

- - POST /api/auth/logout – Logout the user

User Management:

- - GET /api/users – List all users
- - PUT /api/users/:id – Update user profile
- - DELETE /api/users/:id – Delete user - GET /api/users/orders – View user orders

Book Management:

- - GET /api/books – Get all books
- - GET /api/books/:id – Get book by ID
- - POST /api/books – Add new book
- - PUT /api/books/:id – Update book - DELETE /api/books/:id – Delete book

Order Management:

- - POST /api/orders – Place an order
- - GET /api/orders – View orders
- - GET /api/orders/:id – View specific order
- - PUT /api/orders/:id – Update order status

8. Authentication

BookNest uses JWT for user authentication. Passwords are encrypted using Bcrypt, and each API route is secured using middleware that verifies token validity and user roles.

9. User Interface

The UI is built with React and Bootstrap, ensuring responsiveness across devices. Screens include login, registration, book listing, order history, and admin dashboards.

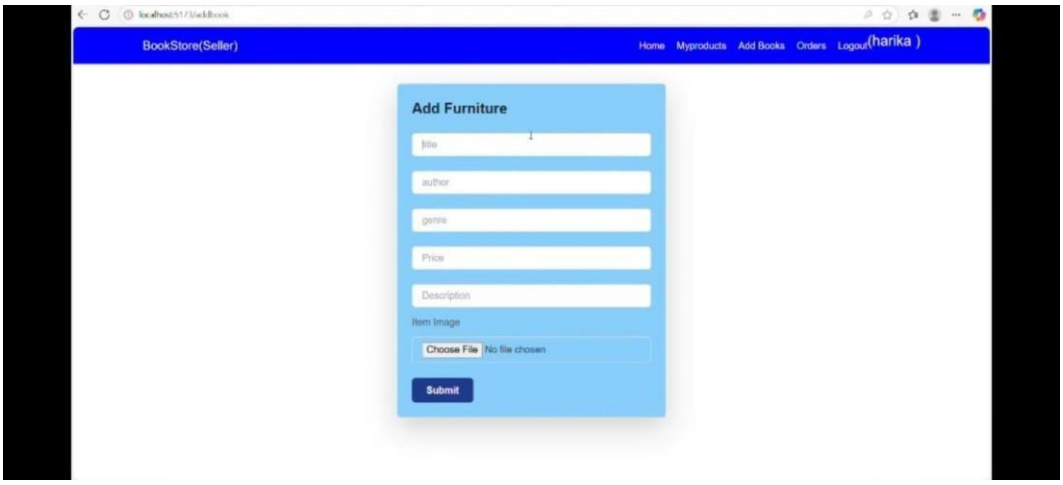
10. Testing

Testing included manual verification of functionality (login, orders, etc.) and automated testing of APIs using Postman.

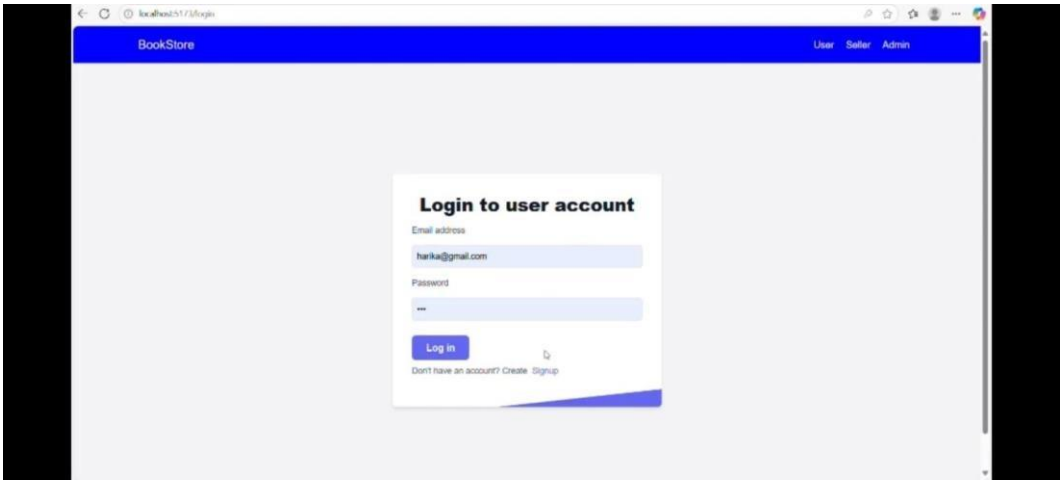
11. Screenshots or Demo

Below are UI screenshots demonstrating key features of the BookNest application.

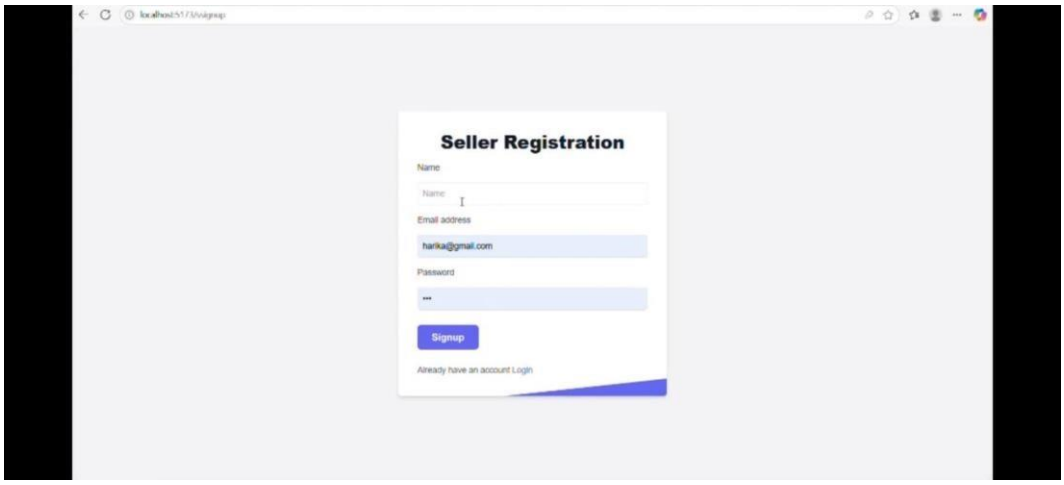
Add Book (Seller View)



User Login Screen



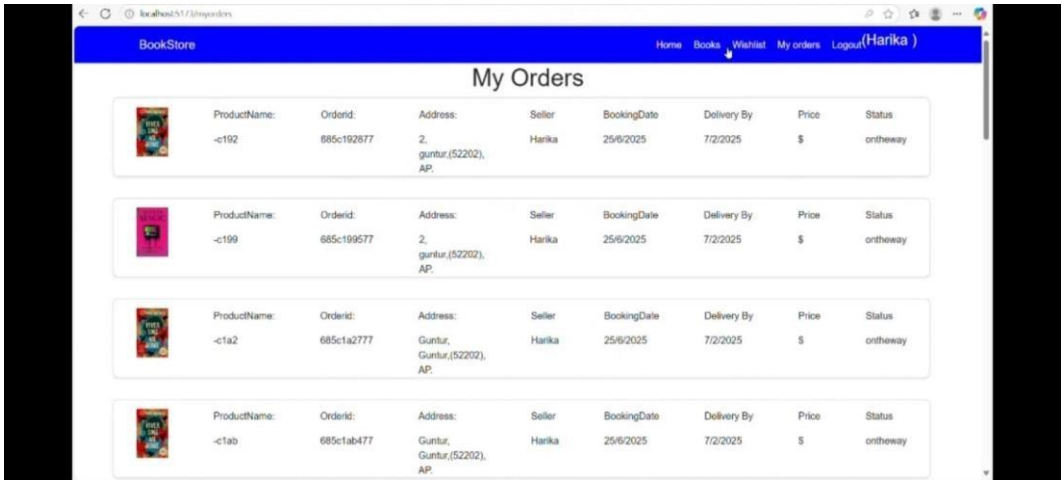
Seller Registration Page



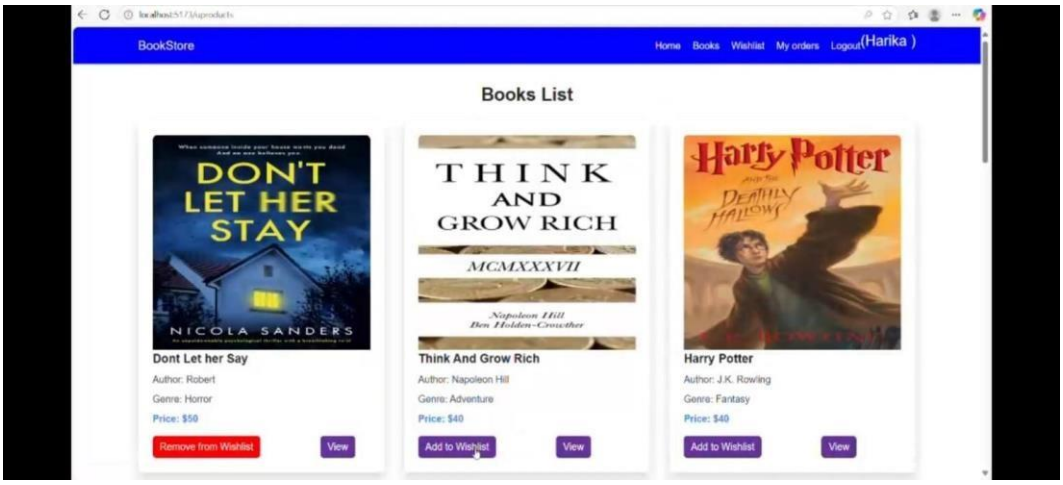
Seller Dashboard Overview

Admin Dashboard Overview

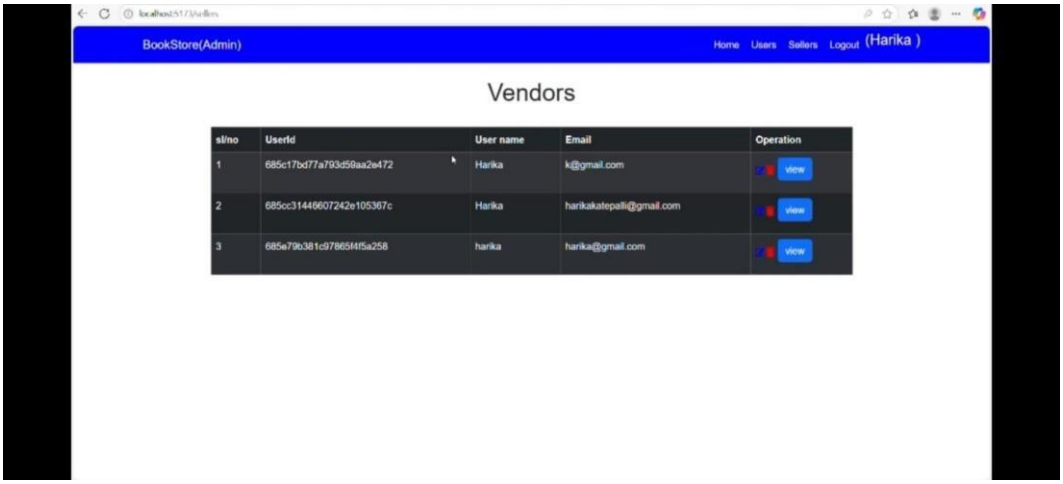
My Orders Page



Books List Page



Vendors Table (Admin View)



12. Known Issues

- - Some form validations need enhancement
- - Token expiration handling could be more user-friendly
- - UI alignment issues on smaller screens

13. Future Enhancements

- - AI-based book recommendations
- - Mobile app version
- - Push notifications for offers and orders

- - Multi-language and currency support

Additional Technical Architecture:

User Interface: Enables users to browse books, search by title or author, read descriptions, and purchase.
Web Server: Hosts the UI and ensures smooth user interaction.
API Gateway: Manages routing of client requests to internal services like authentication, books, and orders.
Authentication Service: Handles login, registration, and secure user access.
Inventory Service: Manages stock levels and availability.
Order Service: Facilitates order placement, confirmation, and tracking. Database:
Stores books, users, orders, genres, reviews, and transaction record

Additional Prerequisites:

- - Node.js and npm: Required to run server-side JavaScript and manage packages.
- - MongoDB: Database system to store book, user, and order data.
- - Express.js: Framework to create server-side APIs.
- - React.js: For creating the client-facing single-page application.
- - Git: Version control system for code management.
- - Development Tools: Visual Studio Code, WebStorm, Sublime Text for coding. - Basic HTML, CSS, and JavaScript knowledge for client-side development.