Full Stack Development with MERN

Project Documentation

1. Introduction

• **Project Title:** BookNest: Where Stories Nestle

• **Team Members:**

Team ID: LTVIP2026TMIDS81807

Team Size: 4

Team Leader: Guttula Venkata Sivaram

Team member: Shubham Kumar

Team member: Pooja Sree Talliboyina

Team member: Basudora Jagadeeswara Rao

2. Project Overview

• **Purpose:**

BookNest is an online bookstore developed using the MERN stack. The purpose of this project is to provide a centralized digital platform where customers can browse and purchase books, sellers can manage inventory, and administrators can oversee platform activities.

• **Features:**

- User Registration and Login
- Role-based dashboards (Customer, Seller, Admin)
- Book browsing and search
- Cart and order placement
- Seller inventory management
- Admin user management
- Order status updates

3. Architecture

- **Frontend:** The frontend is developed using React.js. It provides user interfaces for Customers, Sellers, and Admins. Axios is used to communicate with backend APIs, and React Router handles navigation.

- **Backend:** The backend is built using Node.js and Express.js. It exposes REST APIs for authentication, book management, cart operations, and order processing. JWT is used for secure authentication.

- **Database:** MongoDB is used to store users, sellers, books, orders, and platform data. Mongoose is used for schema modeling and database interactions.

4. Setup Instructions

Prerequisites: Node.js,MongoDB (Local or Atlas),npm, Code Editor •
Installation: 1.Extract project ZIP folder

2.Navigate to backend directory and run.

- npm install, npm start

3.Open new terminal, go to frontend directory and run:

- npm install, npm run dev

4.Open browser and visit:

- http://localhost:5173

5. Folder Structure

Client:

- src/components – UI components
- src/pages – Page-level components
- src/services – API calls
- App.jsx – Main application entry.

Server:

- models – MongoDB schemas
- routes – API endpoints
- controllers – Business logic
- server.js – Server entry poin

6. Running the Application

Frontend:

- cd frontend, npm run dev

Backend:

- cd backend ,npm start .

7. API Documentation

- POST /api/auth/register – User registration
- POST /api/auth/login – User login
- GET /api/books – Fetch books
- POST /api/cart – Add to cart
- POST /api/orders – Place order
- GET /api/orders – View orders
- POST /api/seller/books – Add book
- GET /api/admin/users – Manage users

8. Authentication

- Authentication is implemented using JWT (JSON Web Tokens). Passwords are encrypted using bcrypt.
- Role-based authorization ensures separate access for Customer, Seller, and Admin.

9. User Interface

The UI includes:

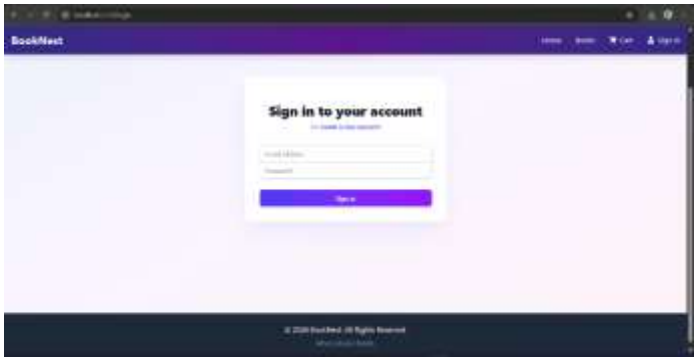
- Login & Registration pages
- Customer dashboard
- Seller inventory dashboard
- Admin control panel
- Book listing and cart pages

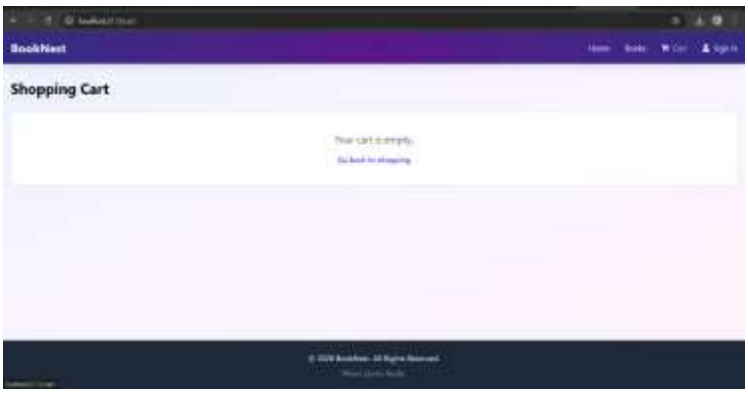
10. Testing

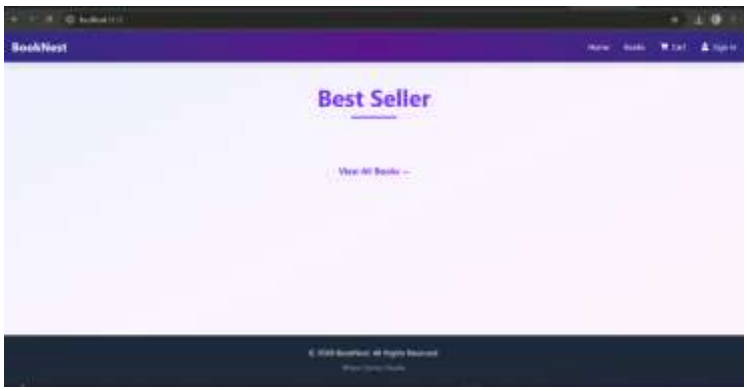
Manual testing is performed for:

- Registration & Login
- Book browsing
- Cart and order placement
- Seller inventory updates
- Admin management.

11. Screenshots







12. Known Issues

- Minor UI alignment issues
- Slow image loading on weak networks

13. Future Enhancements

- Payment gateway integration
- Recommendation system
- Review and rating system