BookNest: Where Stories Nestle

BY

BELLAPUKONDA GAYATHRI

TABLE OF CONTENTS

1. INTRODUCTION

BookNest is an online bookstore platform built using the MERN stack (MongoDB, Express.js, React.js, Node.js). The application allows users to register, browse books, view book details, add books to a cart, place orders, and view order history.

BookNest: Where Stories Nestle

Welcome to the literary haven of the digital age—introducing our revolutionary Book-Store Application, a masterpiece crafted with precision 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 MERN-based Book-Store Application seamlessly blends robust functionality with an intuitive user interface. From the joy of discovering new releases to the nostalgia of revisiting timeless classics, our platform promises an immersive reading experience customized to cater to your literary preferences.

Fueling the backbone of our application is MongoDB, ensuring a scalable and efficient database infrastructure that facilitates swift access to an extensive collection of literary works. Express.js, with its streamlined web application framework, establishes a responsive and efficient server, while Node.js ensures high-performance, non-blocking I/O operations—resulting in a seamless and enjoyable user experience.

At the heart of our Book-Store Application lies React, a dynamic and feature-rich JavaScript library. Dive into a visually enchanting and interactive interface where every click, search, and book selection feels like a literary journey. Whether you're exploring on a desktop, tablet, or smartphone, our responsive design ensures a consistent and delightful experience across all devices.

Say farewell to the constraints of traditional bookstores and embrace a new era of possibilities with our MERN Stack Book-Store Application. Join us as we transform how you connect with literature, making the discovery of your next favorite read an effortless and enriching experience. Get ready to turn the digital pages of a new chapter in reading, where every book is just a click away, and the literary world is at your fingertips. It's time to open the door to a future where the love for books meets the convenience of modern technology.

Scenario Based Case Study:

Sarah is an avid reader with a passion for exploring new genres and authors. However, her busy schedule often leaves her with limited time to visit physical bookstores. Sarah is looking for a solution that allows her to discover and purchase books conveniently, without compromising her reading preferences or the joy of browsing through a bookstore.

User Registration and Authentication: Allow users to register accounts securely, log in, and authenticate their identity to access the book store platform.

Book Listings: Display a comprehensive list of available books with details such as title, author, genre, description, price, and availability status.

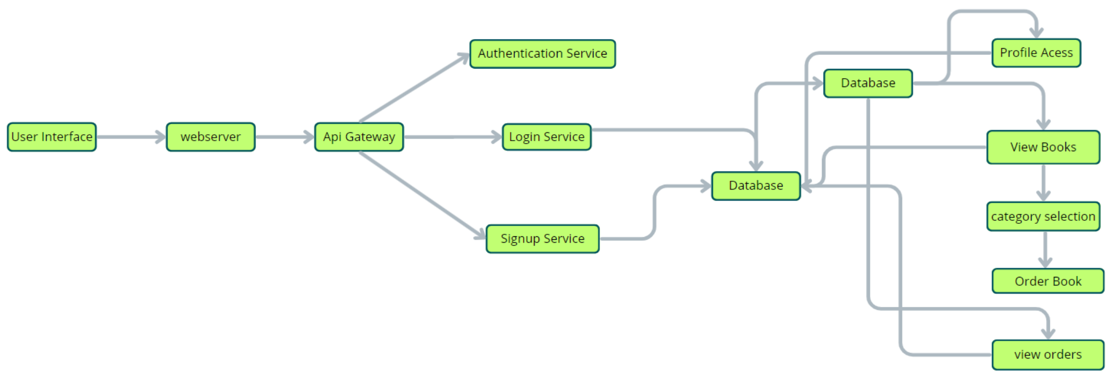
Book Selection: Provide users with options to select their preferred books based on factors like genre, author, ratings, and popularity.

Purchase Process: Allow users to add books to their cart, specify quantities, and complete purchases securely. Upon successful completion, an order is generated, and the inventory is updated accordingly.

Order Confirmation: Provide users with a confirmation page or notification containing details of their order, including book information, total price, and order ID.

Order History: Allow users to view their past and current orders, providing options to track shipments, review purchased books, and rate their shopping experience.

Technical Architecture:



User Interface: The user interface will serve as the platform where customers can browse books, search for specific titles or authors, read book descriptions, and make purchases. It should be intuitive and user-friendly, enabling easy navigation and exploration of available books.

Web Server: The web server hosts the user interface of the book store app, serving dynamic web pages to users and ensuring a seamless browsing and shopping experience.

API Gateway: Similar to the original architecture, the API gateway will serve as the central entry point for client requests, directing them to the relevant services within the system. It will handle requests such as fetching book information, processing orders, and managing user accounts.

Authentication Service: The authentication service manages user authentication and authorization, ensuring secure access to the book store app and protecting sensitive user information during the browsing and purchasing process.

Database: The database stores persistent data related to books, including information such as titles, authors, genres, descriptions, prices, and availability. It also stores user profiles, purchase history, and other essential entities crucial to the book store app.

View Books: This feature allows users to browse through the available books. They can explore different categories and genres to discover books of interest.

Category Selection: Users can select specific categories or genres to filter and refine their book browsing experience, making it easier to find books tailored to their preferences.

Inventory Management Service: This service manages information about available books, including their availability, stock levels, and ratings. It ensures efficient management of the book inventory and seamless integration with the browsing and purchasing process.

Order Management Service: This service facilitates the ordering process, allowing users to add books to their cart, specify quantities, and complete purchases securely. It also handles order tracking and status updates in real-time.

2. OBJECTIVES

- To build a full-stack web application for managing online book sales.
- To allow users to register, login, browse books, and place orders.
- To provide role-based access for Sellers and Admins.
- To perform CRUD operations on books and orders.
- To integrate secure authentication and authorization.
- To maintain a centralized MongoDB database.

3. TOOLS & TECHNOLOGIES USED

Frontend:

- React.js
- React Router
- Axios
- CSS / Tailwind (optional)

Backend:

- Node.js
- Express.js
- MongoDB
- Mongoose
- JSON Web Tokens (JWT)
- bcrypt.js
- dotenv

Dev Tools:

- Visual Studio Code
- Postman
- Git & GitHub
- MongoDB Compass

4. SYSTEM ARCHITECTURE

- The system follows the MVC architecture.
- Frontend (React) interacts with the backend server via REST APIs.
- The backend (Node.js + Express) handles routing, authentication, and database logic.
- MongoDB stores the data in collections: Users, Books, Orders, etc.
- JWT is used for secure authentication.

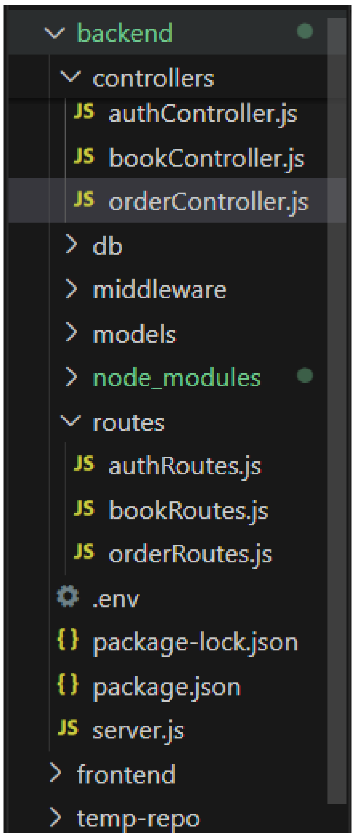
[Insert Architecture Diagram Here – or describe]

- Users → Login/Register → Browse/Order Books
- Sellers → Manage Inventory
- Admin → Manage Users/Sellers/Orders4. SYSTEM ARCHITECTURE

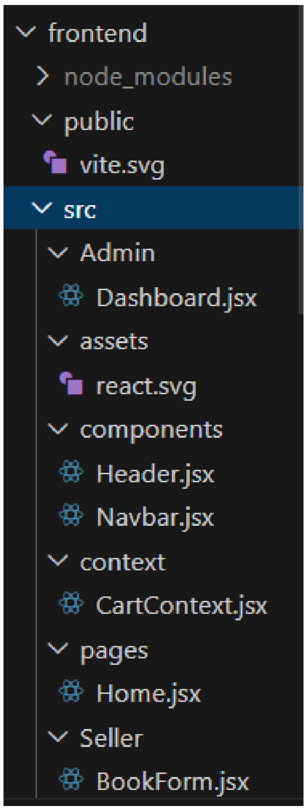
- The system follows the MVC architecture.
- Frontend (React) interacts with the backend server via REST APIs.
- The backend (Node.js + Express) handles routing, authentication, and database logic.
- MongoDB stores the data in collections: Users, Books, Orders, etc.
- JWT is used for secure authentication.

[Insert Architecture Diagram Here – or describe]

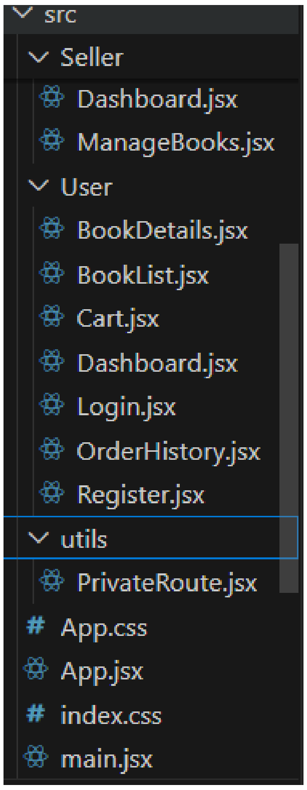
- Users → Login/Register → Browse/Order Books
- Sellers → Manage Inventory
- Admin → Manage Users/Sellers/Orders



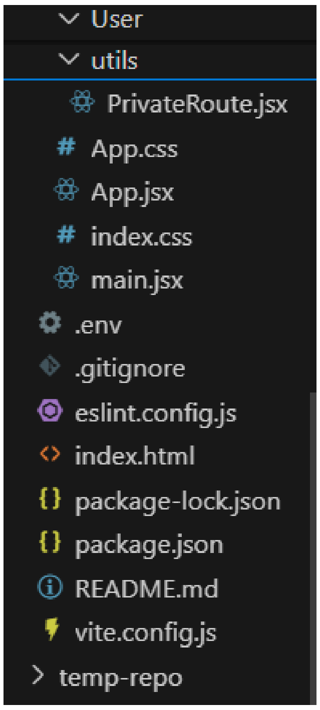
```
backend
  controllers
    authController.js
    bookController.js
    orderController.js
  db
  middleware
  models
  node_modules
  routes
    authRoutes.js
    bookRoutes.js
    orderRoutes.js
  .env
  package-lock.json
  package.json
  server.js
frontend
temp-repo
```



```
frontend
  node_modules
  public
    vite.svg
  src
    Admin
      Dashboard.jsx
    assets
      react.svg
    components
      Header.jsx
      Navbar.jsx
    context
      CartContext.jsx
    pages
      Home.jsx
    Seller
      BookForm.jsx
```



```
src
  Seller
    Dashboard.jsx
    ManageBooks.jsx
  User
    BookDetails.jsx
    BookList.jsx
    Cart.jsx
    Dashboard.jsx
    Login.jsx
    OrderHistory.jsx
    Register.jsx
  utils
    PrivateRoute.jsx
  App.css
  App.jsx
  index.css
  main.jsx
```



```
  User
  utils
    PrivateRoute.jsx
  App.css
  App.jsx
  index.css
  main.jsx
.env
.gitignore
eslint.config.js
index.html
package-lock.json
package.json
README.md
vite.config.js
temp-repo
```

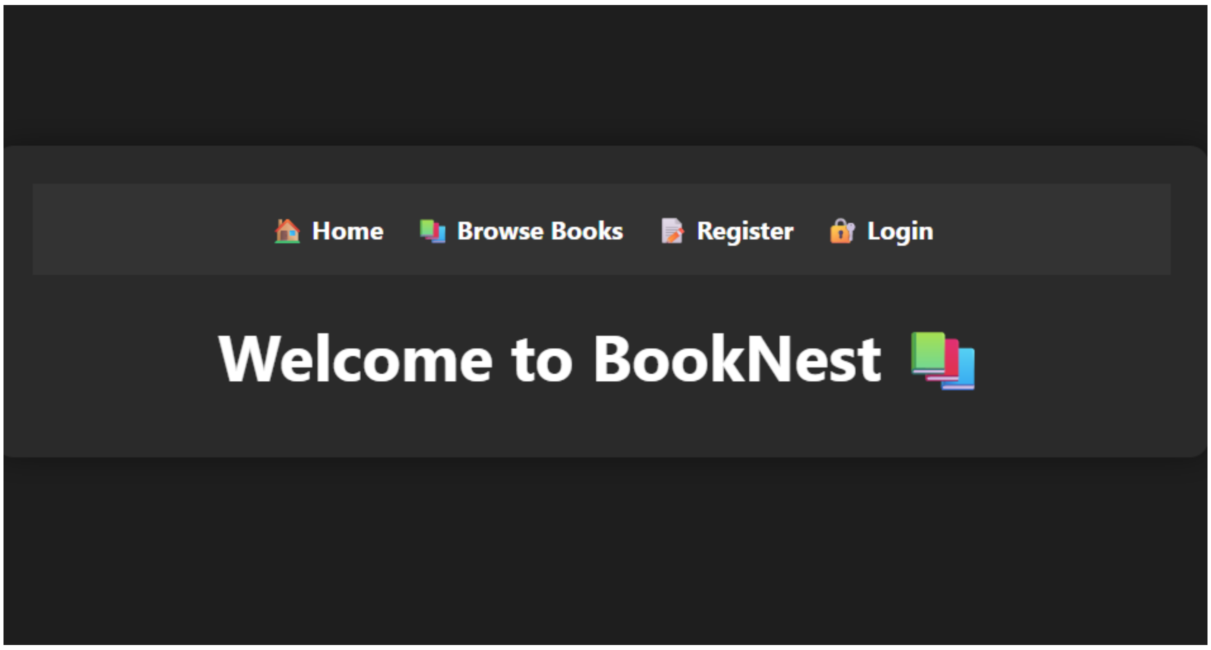
5. ENTITY RELATIONSHIP DIAGRAM

- User ↔ Book → via “Interaction” (many-to-many)
- Book ↔ Inventory (one-to-many)
- User ↔ Order (one-to-many)
- Book ↔ Genre (many-to-many)
- Book ↔ Author (many-to-many)

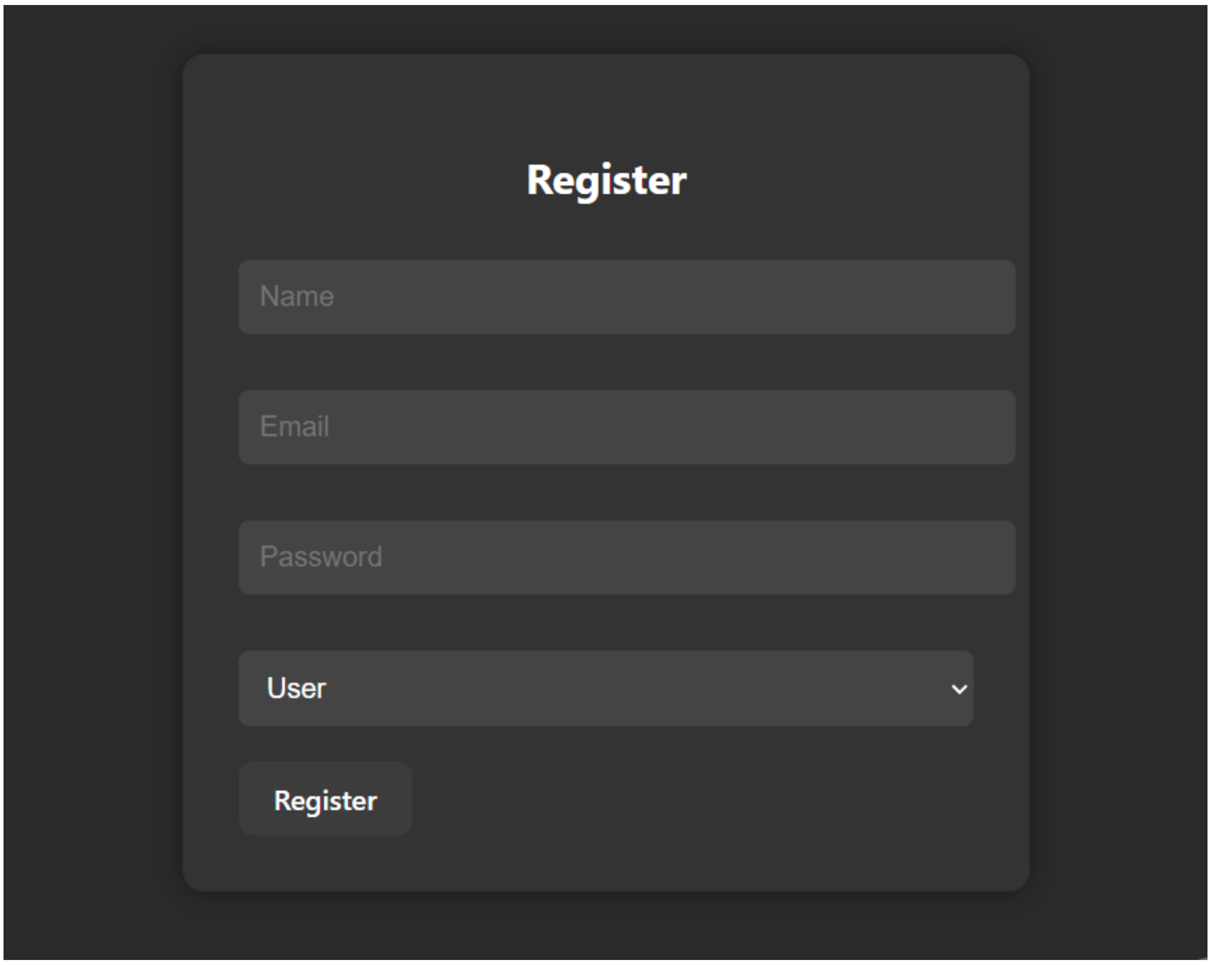
[Paste ER diagram image or sketch]

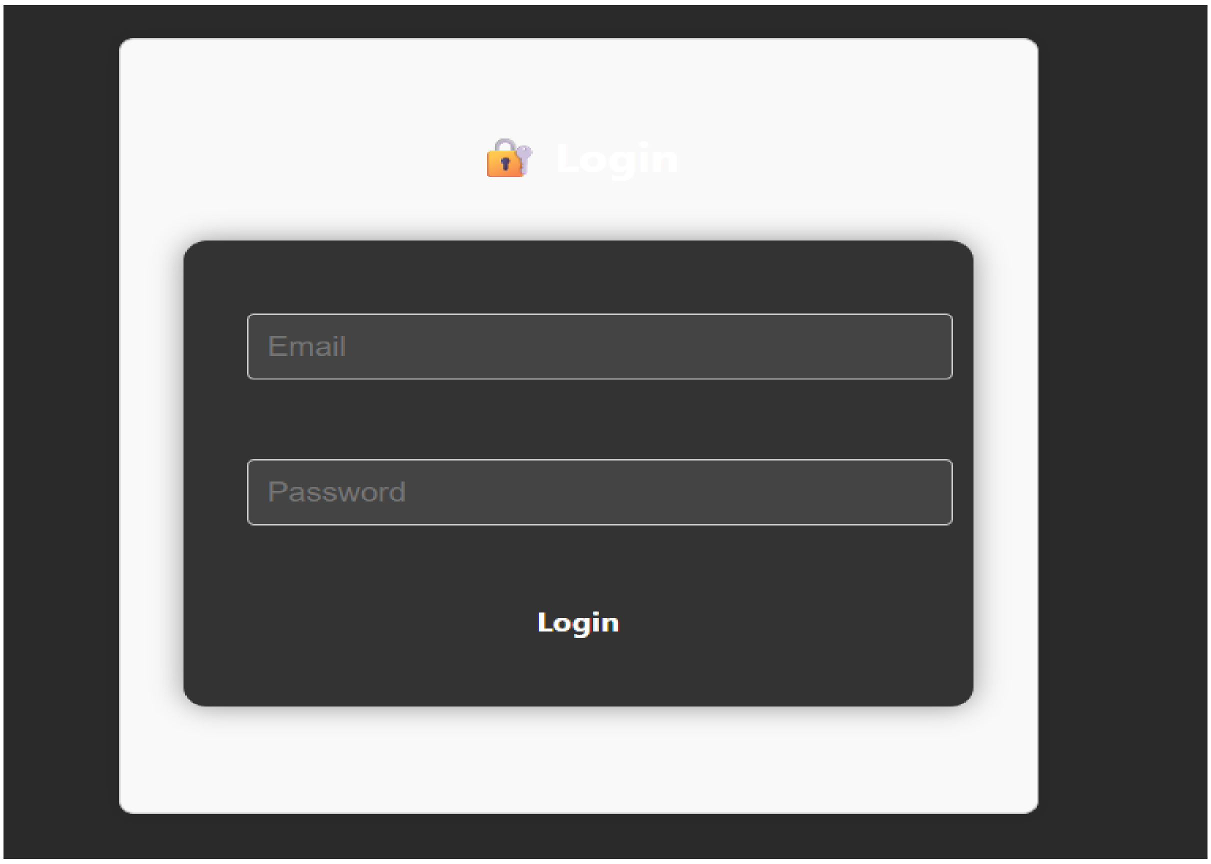
Tables:

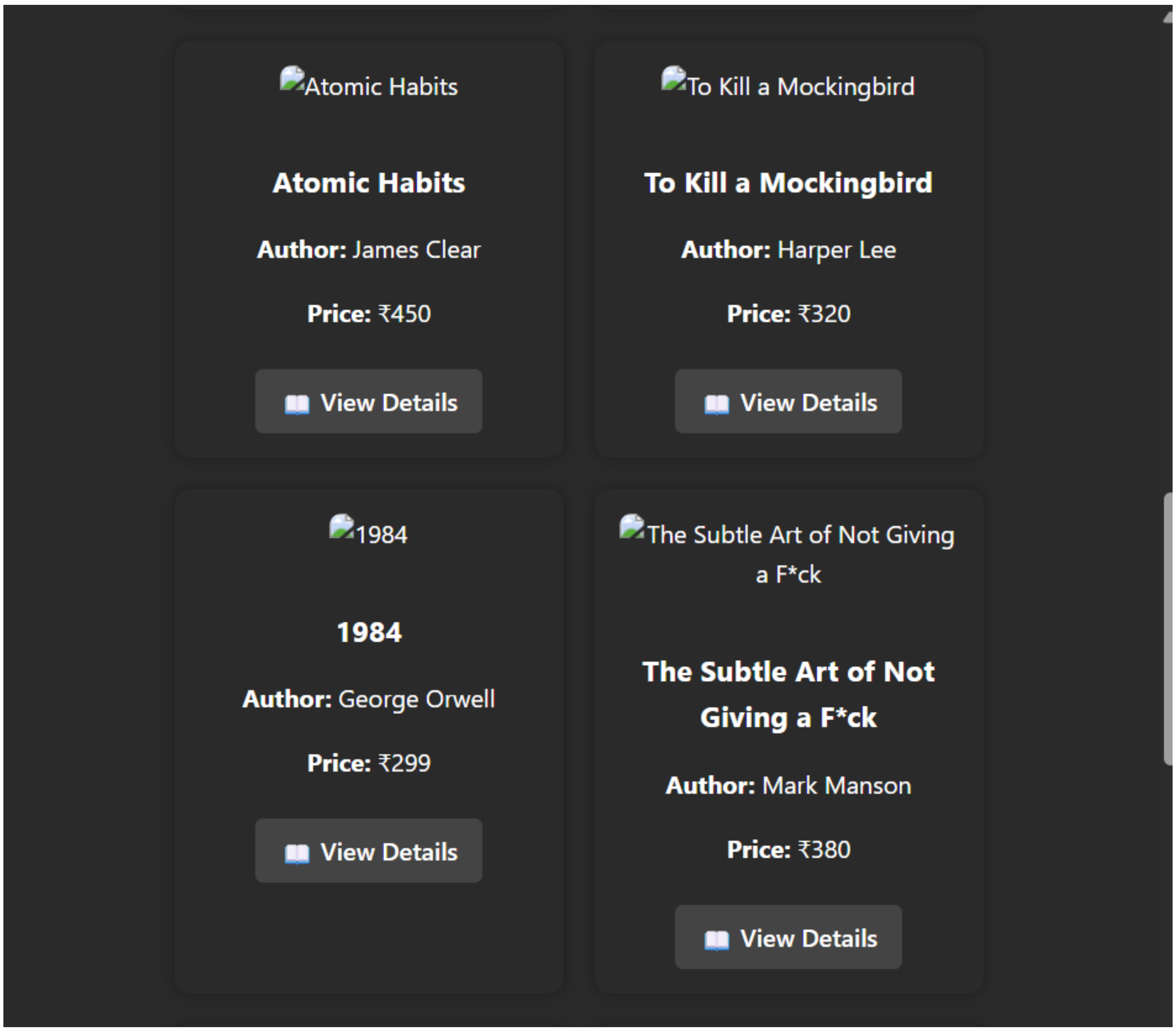
- User: id, name, email, password, role
- Book: id, title, author, price, quantity, description
- Order: id, userID, books[], status, total
- Review: id, userID, bookID, rating, comment



Welcome to BookNest 📚


Register
Name
Email
Password
User
Register


Login
Email
Password
Login



Atomic Habits

Atomic Habits

Author: James Clear

Price: ₹450

📖 View Details

To Kill a Mockingbird

To Kill a Mockingbird

Author: Harper Lee

Price: ₹320

📖 View Details

1984

1984

Author: George Orwell

Price: ₹299

📖 View Details

The Subtle Art of Not Giving a F*ck

The Subtle Art of Not Giving a F*ck

Author: Mark Manson

Price: ₹380

📖 View Details

6. SYSTEM MODULES

- Authentication Module
  - Registration
  - Login
  - JWT token creation

- User Module
  - Browse books
  - Book details
  - Add to cart
  - Place orders
  - View order history

- Seller Module
  - Add/Edit/Delete books
  - View orders for books

- Admin Module
  - Manage Users
  - Manage Sellers
  - View all orders

7. FEATURES IMPLEMENTED

- ✅ User Registration & Login
- ✅ Secure JWT-based Authentication
- ✅ Book Listing and Details Page
- ✅ Cart Functionality
- ✅ Place Orders & View Order History
- ✅ Seller Dashboard to Manage Books
- ✅ Admin Panel to Manage System
- ✅ Role-based Protected Routes
- ✅ MongoDB Integration with Mongoose
- ✅ Full Backend-Frontend Synchronization

9. CONCLUSION

This project provided hands-on experience in building a full-stack MERN application. From setting up Express APIs to integrating React components and managing database records via Mongoose, BookNest demonstrated a complete eCommerce workflow.

Key learning outcomes included:

• Role-based route protection

• Modular backend and frontend structure

• MongoDB schema design

• React component routing and state handling

10. REFERENCES

• https://react.dev

• https://nodejs.org

• https://expressjs.com

• https://mongoosejs.com

• https://jwt.io

• https://www.mongodb.com

• https://vitejs.dev

• ChatGPT – AI assistance & debugging support

• Official documentation for React Router, Axios, bcryptjs