Parul University

FACULTY OF ENGINEERING AND TECHNOLOGY

BACHELOR OF TECHNOLOGY.

Project Documentation.

PROJECT: - BOOKSTORE E-COMMERCE APP.

(MERN - STACK).

DIV: - 6A6.

GROUP NUMBER: - 3.

Overview:

- **Abstract**
- **Introduction**
- **Features**
- **Technology Stack**
- **System Architecture**
- **API Reference**
- **References**

Abstract:

To give customers a flawless online book purchase experie nce, the Bookstore E-
commerce App is a comprehensive web-
based platform developed with the MERN stack (MongoDB, Express.js, React.js, and Node.js).
With features like order management, individual profiles, a nd user verification, the app functions as a digital marketpl ace where users can peruse, search, and buy books in a va riety of genres.
Administrators can track orders, maintain the platform's se amless operation, and manage the book catalog.
The application makes use of React.js for an easy-to-
use and responsive frontend, Express.js and Node.js for a s trong backend, and MongoDB for data storage's scalability and flexibility. High performance, security, and cross-
device compatibility are guaranteed by the app's use of co ntemporary web development techniques and technologies.

Introduction:

The Bookstore App is a feature-rich web application made to give users an engaging way to browse and organize books. The application, which was developed with the MERN stack (MongoDB, Express.js, React.js, and Node.js), combines a dynamic frontend with a strong backend to provide a remarkable user experience. This manual provides clear and understandable explanations of the app's functionality, technological stack, setup procedure, and much more.

Features of the Bookstore App:

Management of Users

Create an account and log in: With their login credentials, users can safely create accounts and log in. JSON Web Tokens are used to manage authentication (JWT).

Profile Management: Users have the ability to modify their personal information.

Role-based Access: Ordinary users can browse and buy, but administrators have more authority, including the ability to manage books.

Collection of Books

Browse Books: Users can see the books that are available along with information about the title, author, genre, and cost.

Search and Filter: Users have the option to apply filters and search for books by genre, author, or title.

Admins have the ability to add, modify, or remove books from the collection.

Purchases and Orders

Customers have the option to add books to their shopping cart for later purchases.

Ordering: Customers are able to place orders and get confirmation information.

Order History: Customers can see the specifics of their previous orders, including the total cost and book details.

Evaluations and Scores

User Reviews: Readers who have bought books can post reviews.

Rating System: Users can rank books to assist others in making purchasing decisions. Design That Responds

The application is made to function flawlessly on a variety of gadgets, including smartphones, tablets, and PCs.

Technology Stack Frontend: *React.js*: Manages the dynamic elements and user interface.

Redux: Controls the global state of the application, including cart information and user authentication.

React Router: Facilitates seamless page navigation.

CSS/SCSS: Used to style and add visual appeal to the user interface.

Backend: *Node.js*: Manages requests and powers the backend server.

Express.js: Offers middleware and routing to manage API requests.

Database: *MongoDB*: Holds orders, reviews, book details, and user data.

Extra Resources

- *Mongoose*: Provides models and schemas to streamline database interactions.

Bcrypt.js: Uses hashing to secure user passwords.

JWT: Securely manages user authentication.

Book cover photos are stored and served via *Cloudinary*.

How the App is Organized:

The app’s codebase is divided into the backend and frontend, ensuring modularity and ease of development.

```
BookstoreApp/
├── backend/
│   ├── config/        # Configuration files (e.g., database settings)
│   ├── controllers/   # Functions handling the core logic of APIs
│   ├── models/        # MongoDB schemas and models (e.g., User, Book, Order)
│   ├── routes/        # Defines API endpoints (e.g., userRoutes, bookRoutes)
│   ├── middleware/    # Custom middleware for authentication and error handling
│   └── server.js      # Main server entry point
├── frontend/
│   ├── src/
│   │   ├── components/    # Reusable UI components (e.g., Navbar, BookCard)
│   │   ├── pages/         # Page components (e.g., HomePage, BookDetailsPage)
│   │   ├── redux/         # State slices and actions (e.g., userSlice, cartSlice)
│   │   ├── App.js         # Main application component
│   │   └── index.js       # React entry point
├── .env               # Environment variables (e.g., database URI, JWT secret)
├── package.json       # Dependencies and project scripts
└── README.md          # High-level project overview
```

Overview of the API:

User Endpoints

Users can sign up using *POST /api/users/register*.

- *POST /api/users/login*: Provides a token after user authentication.

- *GET /api/users/profile*: Gets the protected login information for the user.

Book Endpoints

- *GET /api/books/*: Gets a list of books in paginated form.

***POST /api/books/*: This enables administrators to add new books.**

To update book details (admin only), use *PUT /api/books/:id*.

- *DELETE /api/books/:id*: Removes a book (admin only) from the catalog.

Endpoints of Order

To place a new order, use *POST /api/orders/*.

- *GET /api/orders/:id*: Gets information about a particular order.

***GET /api/orders/user/:userId*: This command retrieves a user's order history.**

Guidelines for Deployment

Backend Deployment: Set up the backend server on an AWS or Heroku cloud platform.

Verify that the server's environment variables are set safely.

Frontend Deployment: Use shell npm run build to create the React application.

Install the build and folder on hosting platforms such as Vercel or Netlify.

Database Setup: For production, use a MongoDB service hosted in the cloud, such as MongoDB Atlas.

Future Plans: Use sophisticated pricing, genre, and rating search criteria.

Use services like PayPal or Stripe to integrate payments.

Turn on social login using Facebook and Google.

Make a mobile application with React Native.

Incorporate push alerts for updates to orders.

-

The features and setup procedure of the Bookstore App are explained in full but in an easy-to-understand manner in this documentation. Please get in touch if you need any additional assistance!