Project Title

BookNest: Where Stories Nestle (MERN Stack)

Team Members:

Here List team members and their roles:

1. B. Ravi Teja (Full Stack Developer): Combines both frontend and backend responsibilities, ensuring smooth communication between the two. This role also handles bug fixing, feature integration, and overall system performance.

2. B. Tejaswi (Frontend Developer): Responsible for designing the user interface using React.js. This role focuses on ensuring a responsive, user-friendly design, as well as integrating the frontend with backend APIs.

3. B. Asrithnath (Backend Developer): Develops the backend server using Node.js and Express.js, ensuring the creation of secure, scalable RESTful APIs, as well as handling authentication, data processing, and business logic.

Introduction:

Tired of scattered book collections? Introducing BookNest: Where Stories Nestle—a revolutionary platform that redefines reading. Discover, organize, and share your favorite books effortlessly. With seamless book tracking, personalized recommendations, and a vibrant reading community, your next great story is just a click away!

Project Overview:

Get ready to transform your reading experience with BookNest: Where Stories Nestle, an innovative platform designed for book lovers. Discover, organize, and share your favorite books effortlessly while engaging with a vibrant community of readers.

Effortlessly track your reading progress, create personalized book collections, and receive tailored recommendations that match your literary taste. Whether you're diving into a new novel, sharing insights with fellow readers, or saving must-read titles for later, BookNest makes it all seamless.

We prioritize your reading experience and data security. Our platform ensures a user-friendly interface with secure access, allowing you to explore, connect, and organize your books without worry.

Goals of Project:

• Enhance reader engagement by encouraging discussions, book reviews, and shared reading experiences.• Facilitate knowledge sharing by allowing users to save, recommend, and review books.• Ensure data privacy and security with robust protection for user information and reading preferences.• Promote book discovery through personalized recommendations and curated reading lists.

Features of BookNest:

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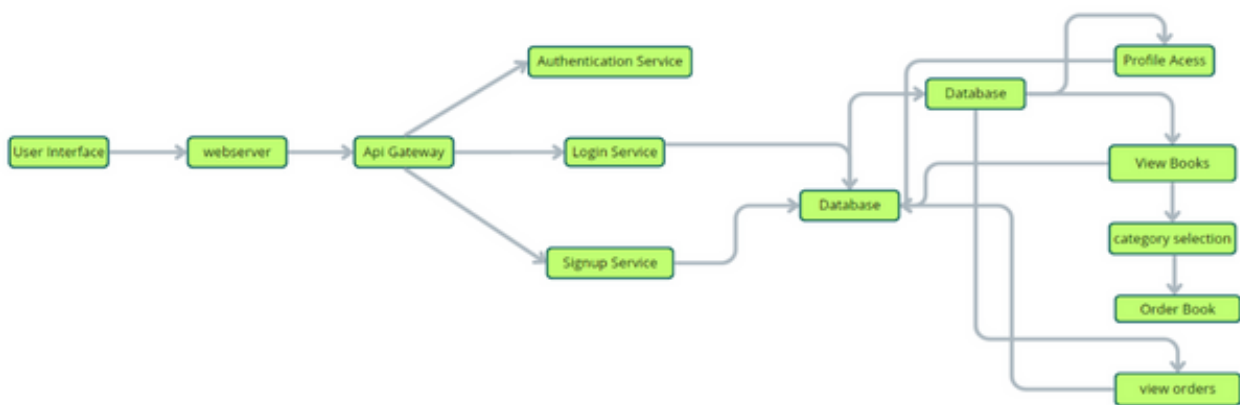
User Interface: The user interface will serve as the platform where customers can browse books, search for specific titles or authors, read book descriptions, and make purchases. It should be intuitive and user-friendly, enabling easy navigation and exploration of available books.

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.

ER Diagram:
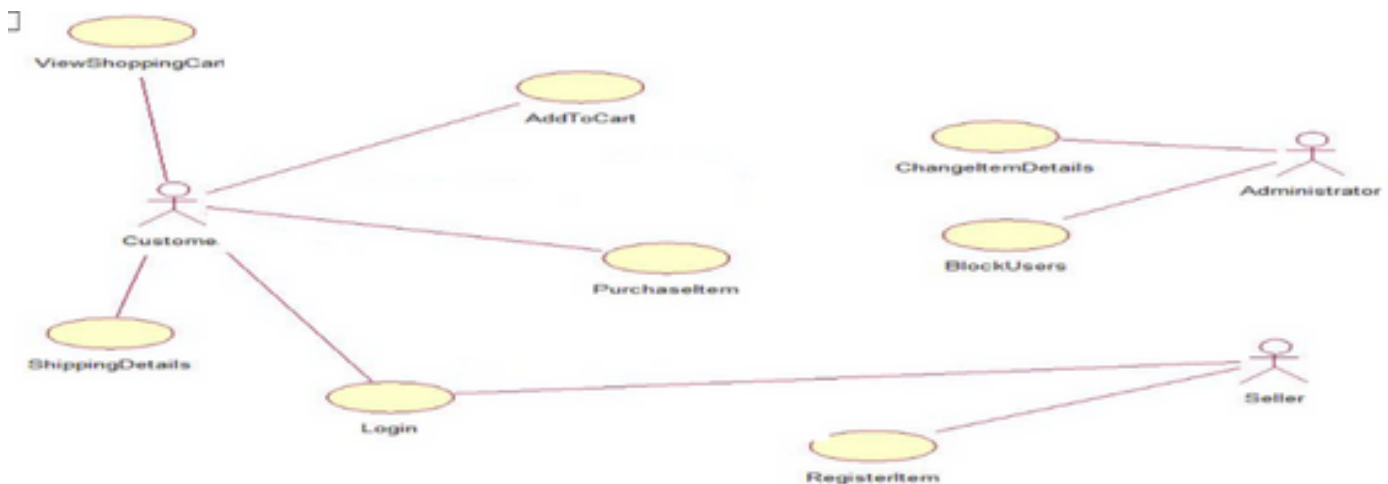
User-Book Relationship:

Type: Many-to-Many (M:M). A single user can read or interact with many books, and a single book can be accessed by many users.

Implementation: Introduce an intermediate entity, "Interaction", with foreign keys to both User and Book tables. This table could store additional information like reading progress, reviews, or ratings.

Book-Inventory Relationship:

Type: One-to-Many (1:M). Each book can have multiple copies in inventory, but each copy belongs to one book.

Implementation: Maintain a separate Inventory table with fields like BookID (foreign key), quantity, location, and condition.

User-Order Relationship:

Type: One-to-Many (1:M). A single user can place multiple orders, but each order belongs to one user.Implementation: Keep the UserID foreign key in the Order table to track user purchase history.

Additional Relationships:

Book-Author Relationship: Many-to-Many (M:M). A book can have multiple authors, and an author can write multiple books. (Similar to User-Book, use an intermediate "WrittenBy" table)

Book-Genre Relationship: Many-to-Many (M:M). A book can belong to multiple genres, and a genre can have many books. (Similar to User-Book, use an intermediate "CategorizedAs" table)

Review-User Relationship: Many-to-One (M:1). A review is written by one user, but a user can write many reviews. (Keep UserID as a foreign key in the Review table)

PRE-REQUISITES:

Here are the key prerequisites for developing a full-stack application using Node.js, Express.js, MongoDB, React.js, Socket.io:

- **Node.js and npm**:

Node.js is a powerful JavaScript runtime environment that allows you to run JavaScript code on the server-side. It provides a scalable and efficient platform for building network applications.

Install Node.js and npm on your development machine, as they are required to run JavaScript on the server-side.

- Download: https://nodejs.org/en/download/
- Installation instructions: https://nodejs.org/en/download/package-manager/

- **Express.js**:

Express.js is a fast and minimalist web application framework for Node.js. It simplifies the process of creating robust APIs and web applications, offering features like routing, middleware support, and modular architecture. Install Express.js, a web application framework for Node.js, which handles server-side routing, middleware, and API development.

Installation: Open your command prompt or terminal and run the following command:

```
npm install express
```

- **MongoDB**:

MongoDB is a flexible and scalable NoSQL database that stores data in a JSON-like format. It provides high performance, horizontal scalability, and seamless integration with Node.js, making it ideal for handling large amounts of structured and unstructured data.

Set up a MongoDB database to store your application's data.

- Download: https://www.mongodb.com/try/download/community
- Installation instructions: https://docs.mongodb.com/manual/installation/

- **React.js**:

React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications.

Install React.js, a JavaScript library for building user interfaces.

Follow the installation guide: https://reactjs.org/docs/create-a-new-react-app.html

Getting Started

Create React App is an officially supported way to create single-page React applications. It offers a modern build setup with no configuration.

Quik Start

npm create vite@latest ./

cd BookNest

npm install

npm run dev

If you've previously installed create-react-app globally via npm install -g create-react-app, we recommend you uninstall the package using npm uninstall -g create-react-app or yarn global remove create-react-app to ensure that npx always uses the latest version.

Create a new React project:

• Choose or create a directory where you want to set up your React project.

• Open your terminal or command prompt.

• Navigate to the selected directory using the cd command.

• Create a new React project by running the following command: npx create-react-app your-app-name.Wait for the project to be created:

• This command will generate the basic project structure and install the necessary dependencies

Navigate into the project directory:

• After the project creation is complete, navigate into the project directory by running the following command: **cd your-app-name**

Start the development server:

• To launch the development server and see your React app in the browser, run the following command: **npm run dev**

• The npm start will compile your app and start the development server.

• Open your web browser and navigate to https://localhost:5173 to see your React app.

You have successfully set up React on your machine and created a new React project. You can now start building your app by modifying the generated project files in the src directory.

Please note that these instructions provide a basic setup for React. You can explore more ad- vanced configurations and features by referring to the official React documentation: https://react.dev/

- **HTML, CSS, and JavaScript**: Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

- **Database Connectivity**: Use a MongoDB driver or an Object-Document Mapping (ODM) library like Mongoose to connect your Node.js server with the MongoDB database and perform CRUD (Create, Read, Update, Delete) operations. To Connect the Database with Node JS go through the below provided link:

• https://www.section.io/engineering-education/nodejs- mongoosejs-mongodb/

- **Version Control**: Use Git for version control, enabling collaboration and tracking changes throughout the development process. Platforms like GitHub or Bitbucket can host your repository.

• Git: Download and installation instructions can be found at: https://git-scm.com/downloads

- **Development Environment**: Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code.

• Visual Studio Code: Download from https://code.visualstudio.com/download

To run the existing BookNest App project downloaded from google drive:

Firstly, download the code from the google drive link provided below

Install Dependencies:

• Navigate into the cloned repository directory:

```
cd BookNest
```

• Install the required dependencies by running the following commands:

```
cd frontend
```

```
npm install
```

```
cd ../backend
```

```
npm install
```

● **Start the Development Server**:

• To start the development server, execute the following command:

```
npm run dev
```

• The video conference app will be accessible at http://localhost:3000

● **Access the App:**

• Open your web browser and navigate to http://localhost:3000.

• You should see the login page, indicating that the installation and setup were successful.

You have successfully installed and set up the social media app on your local machine. You can now proceed with further customization, development, and testing as needed.

Project structure:

- Inside the BookNest directory, we have the following folders
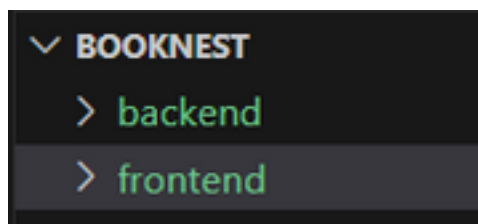
- **frontend directory:**

The below directory structure represents the directories and files in the client folder (front end) where, react Js is used.

```
∨ frontend
  > public
  ∨ src
    > assets
    ∨ components
        AdminLogin.jsx
        Footer.jsx
        Loading.jsx
        Login.jsx
        Navbar.jsx
        Register.jsx
    ∨ context
        AuthContext.jsx
    ∨ firebase
      JS firebase.config.js
    ∨ pages
      ∨ books
          BookCard.jsx
          CartPage.jsx
          CheckoutPage.jsx
          OrderPage.jsx
          SingleBook.jsx
      ∨ dashboard
          > addBook
          > EditBook
          > manageBooks
          > users
          Dashboard.jsx
```
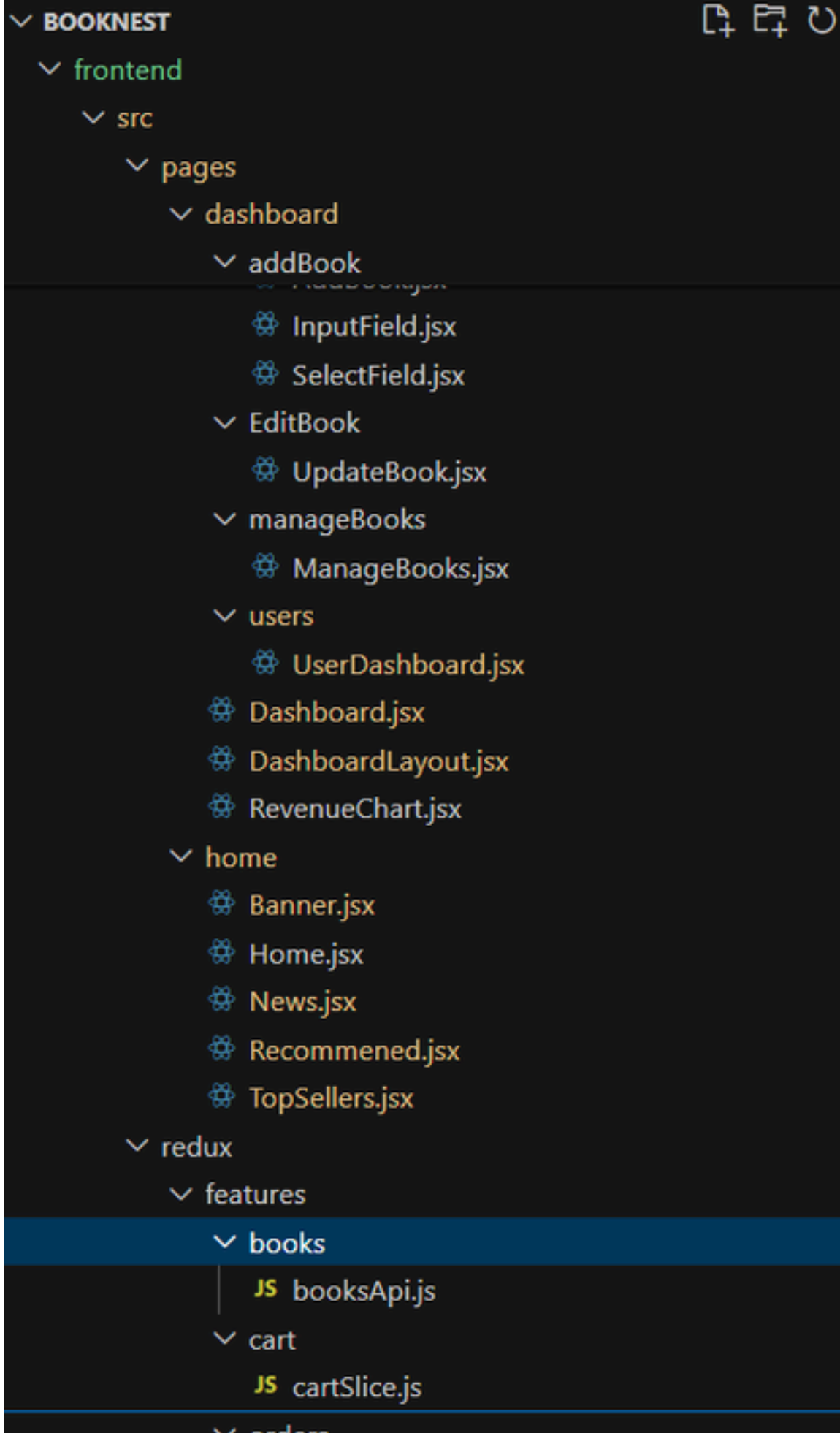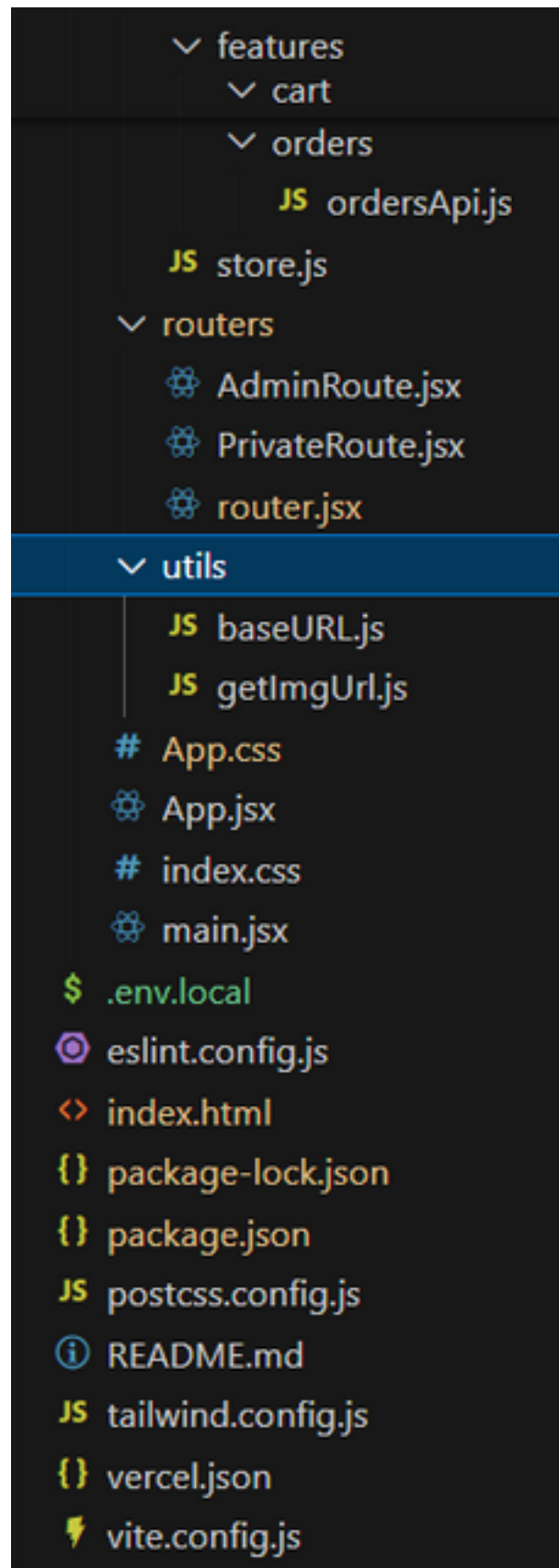
- ∨ BOOKNEST
  - ∨ frontend
    - ∨ src
      - ∨ pages
        - ∨ dashboard
          - ∨ addBook
            - ⚛ InputField.jsx
            - ⚛ SelectField.jsx
          - ∨ EditBook
            - ⚛ UpdateBook.jsx
          - ∨ manageBooks
            - ⚛ ManageBooks.jsx
          - ∨ users
            - ⚛ UserDashboard.jsx
          - ⚛ Dashboard.jsx
          - ⚛ DashboardLayout.jsx
          - ⚛ RevenueChart.jsx
        - ∨ home
          - ⚛ Banner.jsx
          - ⚛ Home.jsx
          - ⚛ News.jsx
          - ⚛ Recommened.jsx
          - ⚛ TopSellers.jsx
      - ∨ redux
        - ∨ features
          - ∨ books
            - JS booksApi.js
          - ∨ cart
            - JS cartSlice.js
          - ∨ orders

```
∨ features
  ∨ cart
    ∨ orders
      JS ordersApi.js
  JS store.js
∨ routers
  ⚛ AdminRoute.jsx
  ⚛ PrivateRoute.jsx
  ⚛ router.jsx
∨ utils
  JS baseURL.js
  JS getImgUrl.js
# App.css
⚛ App.jsx
# index.css
⚛ main.jsx
$ .env.local
◎ eslint.config.js
<> index.html
{} package-lock.json
{} package.json
JS postcss.config.js
ⓘ README.md
JS tailwind.config.js
{} vercel.json
⚡ vite.config.js
```

- **Server directory:**

The below directory structure represents the directories and files in the server folder (back end) where, node js, express js and mongodb.
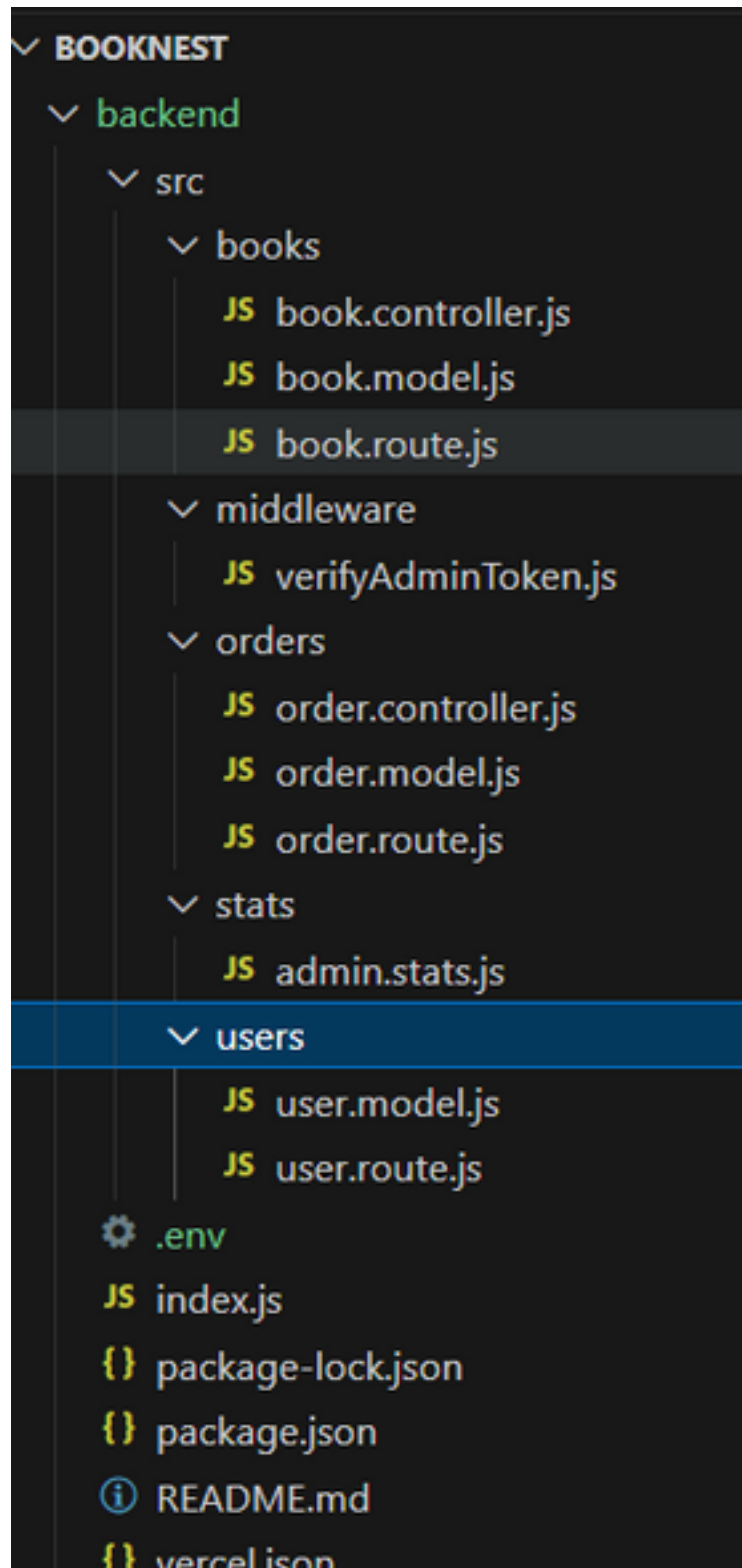
BOOKNEST
- backend
  - src
  - .env
  - index.js
  - package-lock.json
  - package.json
  - README.md
  - vercel.json

Project Flow:

Project demo:

Before starting to work on this project, let's see the demo.

Demo link:

https://drive.google.com/file/d/1rMNIIJWn3BMdSqNJ4W-XtAgL3gRCSVZ/view?usp=drive_link

Use the code in:

https://github.com/Suryanarayana1269/BookNest-Mern

Milestone 1: Project setup and configuration.

- **Folder setup**:

Now, firstly create the folders for frontend and backend to write the respective code and install the essential libraries.

- ○ frontend folders.
- ○ backend folders

- **Installation of required tools**:

Open the frontend folder to install necessary tools. For frontend (client), we use:

Tools/libraries	Installation command
React Js	`npx create-react-app .`
Vite	`npm create vite@latest ./`
Tailwind css	`npm install bootstrap`
Axios	`npm install axios`
Firebase	`npm install firebase`
Frame-Motion	`npm install frame-motion`

Open the backend folder to install necessary tools. For backend (server), we use:

Tools/libraries	Installation command
Express Js	`npm install express`
Mongoose	`npm install mongoose`
Bcrypt	`npm install bcrypt`
Body-parser	`npm install body-parser`
Cors	`npm install cors`
Dotenv	`npm install dotenv`

Milestone 2: Backend development

- **Set Up Project Structure:**
- Create a new directory for your project and set up a package.json file using npm initcommand.
- Install necessary dependencies such as Express.js, Mongoose, and other requiredpackages.
- **Set Up Project Structure:**
  - Create a new directory for your project and set up a package.json file using npm initcommand.
  - Install necessary dependencies such as Express.js, Mongoose, and other requiredpackages.
- **Create Express.js Server:**
  - Set up an Express.js server to handle HTTP requests and serve API endpoints.
  - Configure middleware such as body-parser for parsing request bodies and cors forhandling cross-origin requests.

- **Define API Routes:**
  - Create separate route files for different API functionalities such as authentication, create Books, upload Books, etc.,
  - Implement route handlers using Express.js to handle requests and interact with thedatabase.

- **Implement Data Models:**
  - Define Mongoose schemas for the different data entities.
  - Create corresponding Mongoose models to interact with the MongoDB database.

- ○ Implement CRUD operations (Create, Read, Update, Delete) for each model to perform database operations.

- **User Authentication:**
  - ○ Implement user authentication using strategies like JSON Web Tokens (JWT) or session-based authentication.
  - ○ Create routes and middleware for user registration, login, and logout.
  - ○ Set up authentication middleware to protect routes that require user authentication.

- **Handle new chats and posts:**
  - ○ Allow users to chat with other users using the userId.
  - ○ Also the users will make the posts on social timeline.
- **Error Handling:**
  - ○ Implement error handling middleware to catch and handle any errors that occur during the API requests.
  - ○ Return appropriate error responses with relevant error messages and HTTP status codes.

Reference video for backend code:

https://drive.google.com/drive/folders/1eiTGiVpYe3nzfsOiWX-hRC0pKniRf24M?usp=drive_link

Milestone 3: Database development:

- Set up a MongoDB database either locally or using a cloud-based MongoDB service like MongoDB Atlas.
- Create a database and define the necessary collections for users, posts, stories, chats, etc.,

The code for the database connection to the server is

```js
const PORT = process.env.PORT || 6001;
mongoose.connect(process.env.MONGO_URL, {
    useNewUrlParser: true,
    useUnifiedTopology: true
}).then(()=>{


    server.listen(PORT, ()=>{
        console.log(`Running @ ${PORT}`);
    });


}).catch((err)=>{
    console.log("Error: ", err);
})
```

The code for the schemas in the database is provided below

```js
JS User.js   ✕

server > models > JS User.js > ...
    1    import mongoose from 'mongoose';
    2
    3    const UserSchema = new mongoose.Schema({
    4        username:{
    5            type: String,
    6            require: true
    7        },
    8        email:{
    9            type: String,
   10            require: true,
   11            unique: true
   12        },
   13        password:{
   14            type: String,
   15            require: true
   16        },
   17    });
   18
   19    const User = mongoose.model("users", UserSchema);
   20    export default User;
```

Milestone 4: Frontend development & Integration

1. Setup React Application:

• **Create React application.**

• **Configure Routing.**

• **Install required libraries.**

2. Design UI components:

• **Create Components.**

• **Implement layout and styling.**

• **Add navigation.**

3. Implement frontend logic:

• Integration with API endpoints.

**• Implement data binding.**

Reference for frontend code:

https://drive.google.com/drive/folders/1XjDCrpv7oe1lHBDm-nE-AeCjJQX7ns9U?usp=drive_link

Milestone 5: Project Implementation:

On completing the development part, we then run the application one last time to verify all the functionalities and look for any bugs in it. The user interface of the application looks a bit like the one's provided below.

Finally, after finishing coding the projects we run the whole project to test it's working process and look for bugs. Now, let's have a final look at the working of our Cab Booking application.

Landing page:-
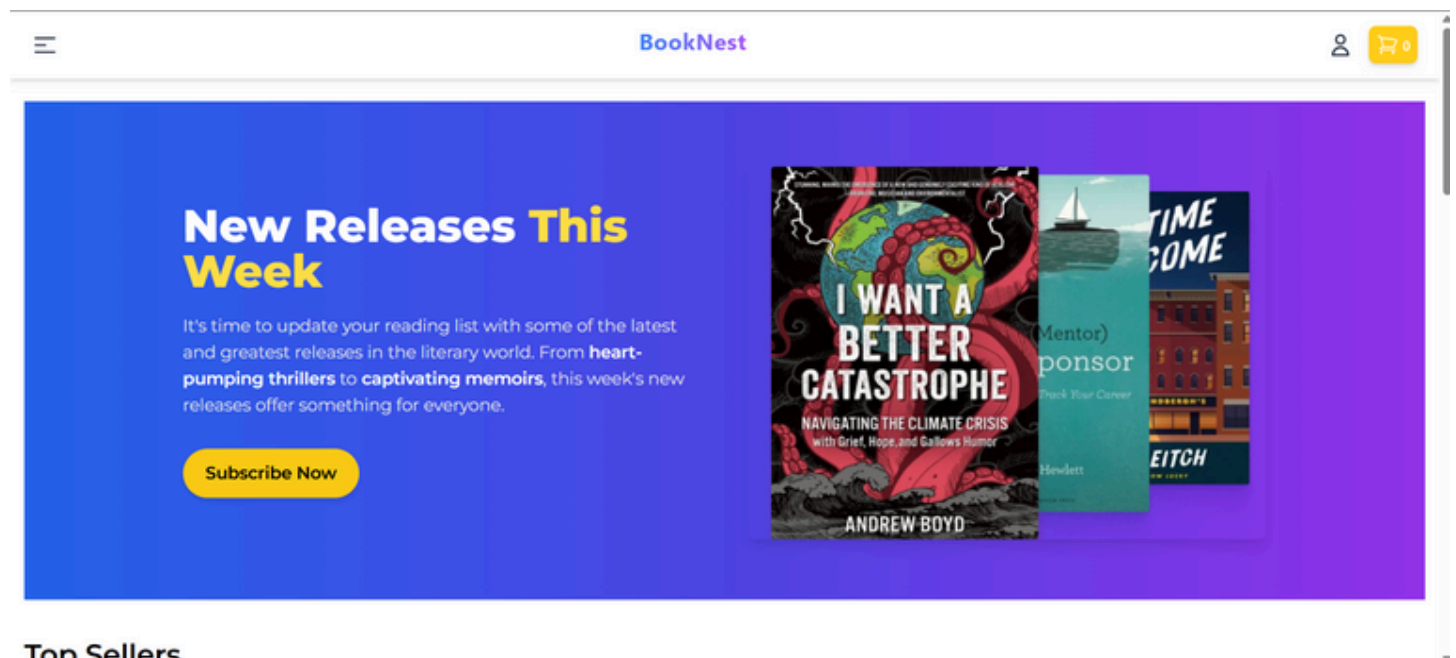
Login Page:-

Welcome Back

Email

Enter your email

Password

Enter your password

Login

Don't have an account? Sign up

G Sign in with Google

Admin Login? Click here

©2025 BookNest. All rights reserved.

Home Page:-

New Releases This Week

It's time to update your reading list with some of the latest and greatest releases in the literary world. From **heart-pumping thrillers** to **captivating memoirs**, this week's new releases offer something for everyone.

Subscribe Now

Top Sellers

Books Page:-

Harry Potter and the Order of the Phoenix

Author: admin

Published: 2/26/2025

Category: Adventure

Description: Harry Potter is about to start his fifth year at Hogwarts School of Witchcraft and Wizardry. Unlike most schoolboys, Harry never enjoys his summer holidays

Add to Cart

Top Seller Page:

Top Sellers

Choose a genre
- Choose a genre
- Business
- Fiction
- Horror
- Adventure

Four Thousand Weeks

Nobody needs to be told there isn't enough time. We're obsessed with our lengthe...

$23 ~~$15~~

🛒 Add to Cart

The Alchemist

Paulo Coelho's masterpiece tells the mystical story of Santiago, an Andalusian s...

$27.99 ~~$35.99~~

🛒 Add to Cart

Divergent

On an appointed day of every year, all sixteen-year-olds must select the fac...

$12.99 ~~$18.99~~

🛒 Add to Cart

Recommended For You:

Recommended for you

Harry Potter and the Order of the Phoenix

Harry Potter is about to start his fifth year at Hogwarts School of Witchcraft a...

$18.99 ~~$27.99~~

🛒 Add to Cart

The Hunger Games

Could you survive on your own in the wild, with every one out to make sure you d...

$16.99 ~~$21.99~~

🛒 Add to Cart

The First Days

Katie is driving to work one beautiful day when a dead man jumps into her car...

$49.99 ~~$59.99~~

🛒 Add to Cart

Cart Page:

Shopping cart

Clear Cart

No product found!

Subtotal $0.00

Shipping and taxes calculated at checkout.

Checkout

or Continue Shopping →

Checkout Page:

Cash On Delivery

Total Price: $0.00

Items: 0

Personal Details
Please fill out all the fields.

Full Name

Email Address

suryanarayanabodapati1269@gmail.com

Phone Number

+123 456 7890

Address / Street

City

Country / region

Country

State / province

State

Zipcode

☐ I am agree to the Terms & Conditions and Shoping Policy.

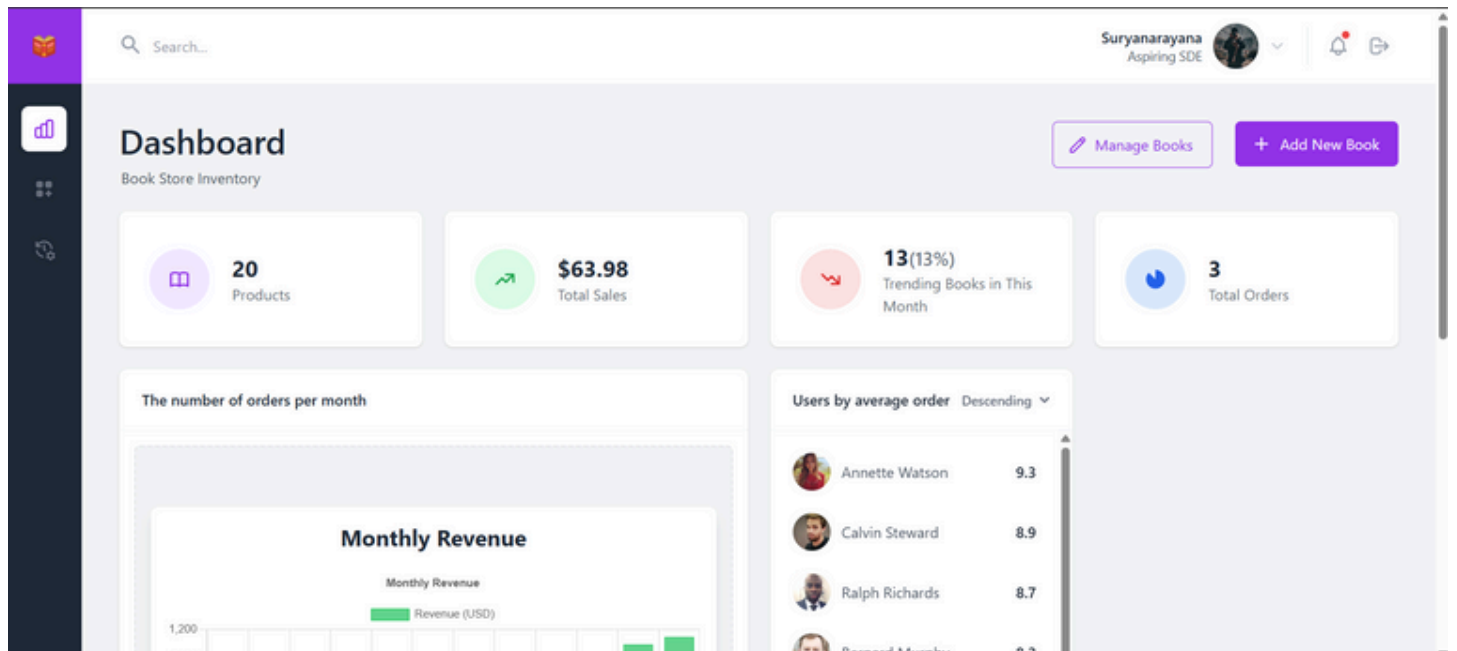**Place an Order**

Admin Login Page:

Admin Dashboard Login

Username

Enter username

Password

Enter password

Login

Admin Dashboard Page:

Manage Books Page:

All Books

#	BOOK TITLE	CATEGORY	PRICE	ACTIONS	
1	Four Thousand Weeks	business	$23	Edit	Delete
2	The Alchemist	adventure	$27.99	Edit	Delete
3	Divergent	business	$12.99	Edit	Delete
4	Gone with the Wind	fiction	$12.99	Edit	Delete
5	The Giving Tree	fiction	$24.99	Edit	Delete
6	The Picture of Dorian Gray	horror	$21.99	Edit	Delete
7	The Fault in Our Stars	business	$9.99	Edit	Delete
8	To Kill a Mockingbird	fiction	$25.99	Edit	Delete
9	Pride and Prejudice	fiction	$10.99	Edit	Delete

Add a New Book Page:

Add New Book

Title

Enter book title

Description

Enter book description

Category

Choose A Category

☐ Trending

Old Price

Old Price

New Price

New Price

Cover Image

Choose File   No file chosen

Add Book

User Page in Firebase:

User Orders:

User Dashboard

Welcome, **SURYANARAYANA BODAPATI**! Here are your recent orders:

Your Orders

Order ID: 67cb2fdbdd65b104e5238520

Date: 3/7/2025

Total: $19.99

Products:

- 67bee11b35a81fa1878a519c

Order ID: 67cb2fbddd65b104e523851e

Date: 3/7/2025

Total: $19.99

Products:

- 67bee11b35a81fa1878a519c

Finally, for any further assistance, use the links below:

Demo link: https://drive.google.com/file/d/16mLxrRt3TbonobTG-Ovt-wt7EWPx0hTx/view?usp=share_link