

**Full Stack Development with MERN**  
**Project Documentation**

Date	22 June 2024
Team ID	SWTID1720433526
Project Name	Project - Book-Store
Maximum Marks	2 Marks

1. Introduction

- Project Title: Book-Store
- Team Members: Matthew Jeannot Aranjani ( Leader, Full Stack Developer)

2. Project Overview:

Purpose: To create a comprehensive book store platform that allows users to browse, search and purchase books online. Users can view order history and their wishlists.

Features:

- a. Book Listing with detailed information.
- b. Book Selection based on various criteria.
- c. Loading Spinner.
- d. Books shown in card format.

3. Architecture:

- a. Frontend: The frontend is built with React.js with Tailwind CSS for styling. The structure is organized into components and pages, providing a modular and maintainable codebase. It uses React Router for navigation and Axios for API requests.
- b. Backend: the backend is developed using Node.js

and Express.js. It follows the REST API principle to handle CRUD operations for books and user data.

- c. Database: MongoDB is used as the database to store user and book information. The Mongoose library is used for schema definitions and interactions with the MongoDB database.

#### 4. Setup Instructions:

- a. Prerequisites: Node.js, MongoDB, Git

```
git clone https://github.com/your-repo/book-store-platform.git
```

```
cd book-store-platform
```

```
cd frontend
```

```
npm install
```

```
cd ../backend
```

```
npm install
```

```
MONGO_URI='mongodb+srv://root:root@books-storemern.kc6yg4d.mongodb.net/bookscollection?retryWrites=true&w=majority&appName=Books-Store-MERN'
```

#### 5. Folder Structure:

- a. Client:

- i. src:

1. assets: Static assets like images and icons.
2. components: Reusable components such as Backbutton, Spinner.
3. pages: Main application pages such as Home, EditBook, ShowBook.
4. App.jsx: Main application component.
5. index.css: Global CSS file.
6. main.jsx: Entry point of the application.

b. Server:

- i. models: Mongoose schemas such as bookModel.js.
- ii. routes: Express routes such as booksRoute.js.
- iii. config.js: Configuration file for database connection.
- iv. index.js: Entry point of the server.

6. Running the Application:

Frontend:

```
cd frontend  
npm start
```

Backend:

```
cd backend  
npm start
```

7. API Documentation:

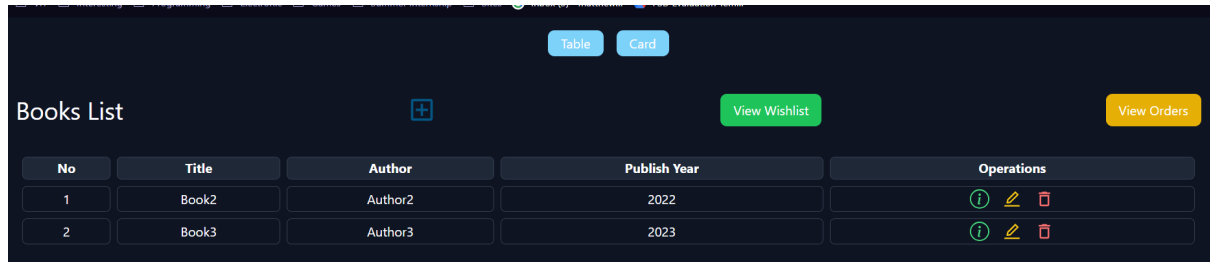
Endpoints:

- a. Get all books:  
GET /books - List of all books
- b. Get a single book by ID:  
GET /books/:id - Details of the book
- c. Create a new book:  
POST /books - { title, author, publishYear } -  
Created Book
- d. Update a book by ID:  
PUT /books/:id - { title, author, publishYear } -  
Updated Book
- e. Delete a book by ID:  
DELETE /books/:id - Deletion confirmation

8. Authentication:

Authentication not done.







## 9. User Interface:



Books List

Table Card

View Wishlist View Orders

No	Title	Author	Publish Year	Operations
1	Book2	Author2	2022	  
2	Book3	Author3	2023	  

## 10. Testing Strategy:

Testing manually all possible major functionalities.

## 11. Demo:

<https://drive.google.com/file/d/14ntCoZrm9B8W37ucCsINDotIxe0iULRs/view>

## 12. Known Issues:

Bugs related to Wishlist and Orders

## 13. Future Enhancements:

Admin, Seller integration.

Buying and Selling Books.

Authentication.