**Full Stack Development with MERN**

**API Development and Integration Report**

| Date | 12-07-2024 |
| --- | --- |
| Team ID | SWTID1719925692 |
| Project Name | Bookstore |
| Maximum Marks |  |

**Project Title:** BookCorner  
**Date:** 12-07-2024  
**Prepared by:**

1. Shaneel reddy
2. Vinuthna swarna
3. Mahitha reddy
4. Allwin

**Objective :**  
The purpose of this document is to state the progress of API development and describe the working of the backend services in BookCorner project

**Technologies Used**

* **Backend Framework:** Node.js with Express.js
* **Database:** MongoDB
* **Authentication:** Jwt

**Project Structure**  
This contains the backend file structure for the project .



**Key Directories and Files**

1. **/controllers**

Basically contains the code for the logic to handle requests and responses with methods .

* + bookController : contains the logic for fetching books from mongodb for the bestseller part.
  + cartController : logic for adding and removing books from the cart.
  + gbooksController : methods for fetching books from Google Api.
  + adminController : logic for posting books into the database and tracking orders.
  + orderController : methods for giving a pop up message when order is placed .

1. **/models**

Includes Mongoose schemas and models for MongoDB collections.This contains

schemas for collections in the database.

* + books : schema for the bestsellers books
  + cart : schema for storing the books added in cart into the database .
  + gbooks : for structuring the books fetched from the google api
  + user : schema for storing the user info in the database .

1. **/routes**

Contains the API endpoints and links them to controller functions with HTTP Methods.

* + bookRoute : for getting the data from the database to a particular url
  + cartRoute : with post and delete methods to add and delete books in the cart .
  + gbooksRoute : get method for fetching the book data from the Api with controller methods
  + userRouter : for posting the user info with userController methods into user type schema .

1. **/middlewares**

Is used to process requests and responses, enabling reusable code for tasks like logging,

authentication, and error handling.

* + auth : It uses JSON web tokens to verify the requests are authenticated . A secret key is used to sign the JWT’s stored in the environment variables .Used to check if the incoming contains a valid JWT token or not .
  + authvalidation : validates the signup and login functionality using joi library.signup validation validates the email,name and password . login validates the email and password .When the validation fails it gives 400 error .

**API Endpoints**  
 API endpoints and their purposes:

**User Authentication**

* **post /** **login** - validates a user and directs them to .
* **post /signup** - registers a new user in the database .

**Selling used books**

* **get /** **usedbooks** - retrieves the books info from the database
* **post / submit** - After user enters the info of the book it stores in the database.

**Cart Management**

* **get /** **get**- Retrieves the cart information
* **post /add** - Adds new books to the cart
* **delete /remove/:bookId/** - removes book with a specific id from the cart

**Managing bestsellers**

* **get /getbooks**- gets info of bestselling books
* **post /postbooks**- adding new books into the database
* **put/books/:id** - making any changes(updating) in the particular book with id
* **delete/books/:id** - removing book data in the database

**Integration with Frontend**  
The backend interacts with the database with RESTful APIs and then we fetch the data using Api endpoints .

**User Authentication: JWT** Tokens are passed between frontend and backend to handle authentication.

**Data Fetching:** Frontend components make API calls using axios to fetch data for display and interaction.

**Error Handling and Validation**  
error handling strategy and validation mechanisms:

* **Error Handling:** error handling using middleware with routes.
* **Authentication:** Secured token-based authentication with JWT tokens.
* **Data Encryption:** Bcrypt for password encryption .