**Full Stack Development with MERN**

**API Development and Integration Report**

|  |  |
| --- | --- |
| Date | 18th July,2024 |
| Team ID | SWTID1720110658 |
| Project Name | Book Nest |
| Maximum Marks | 100 |

**Project Title:** **Book Nest**

**Date:** 18th July,2024

**Prepared by:**

Shreya Agarwal (TEAM LEADER) 22BCE2179

Monesha Ra 22BPS1190  
 Venu Gopal 22BCE2829  
 Anisha Plawat 22BCE0275

**Objective**  
The objective of this report is to document the API development progress and key aspects of the backend services implementation for the Book Nest project.

**Technologies Used**

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

**Project Structure**

* Home
  + src
    - assets
      * banner-books
      * awardbook.jpg
      * favoritebook.jpg
      * profile.jpg
      * react.svg
    - cart
      * Cart.jsx
      * CartContext.jsx
    - Components
      * About.css
      * About.jsx
      * Banner.jsx
      * BookCard.jsx
      * BookCard.css
      * Login.jsx
      * loginStyle.css
      * MyFooter.jsx
      * Navbar.jsx
      * SignUp.jsx
    - Contects
      * AuthProvider.jsx
    - Dashboard
      * DashBoard.jsx
      * DasBoardLayout.jsx
      * EditBooks.jsx
      * ManageBooks.jsx
      * SideBarDash.jsx
      * UploadBook.jsx
    - Firebase
      * firebase.config.js
    - home
      * BannerCard.css
      * BannerCard.jsx
      * BestSellerBooks.jsx
      * FavBook.jsx
      * Home.jsx
      * OtherBooks.jsx
      * PromoBanner.jsx
      * Review.jsx
    - images
    - privateroutes
      * PrivateRoute.jsx
    - Routers
      * Router.jsx
    - Shop
      * Shop.jsx
      * SingleBook.jsx
    - Wishlist
      * Wishlist.jsx
      * WishlistContext.jsx
* Server
  + index.js

**Key Directories and Files**

**Directories:**

1. **db**: This directory likely contains database-related files or configurations.
2. **node\_modules**: This directory contains dependencies installed using npm (Node Package Manager).

**Files:**

1. **index.js**: This is a JavaScript file that likely serves as the entry point for the server application.
2. **package-lock.json**: This file contains metadata about the project's dependencies, including version numbers and installation instructions.
3. **package.json**: This file contains metadata about the project, including its name, version, dependencies, and scripts.

**API Endpoints**  
A summary of the main API endpoints and their purposes:

**1. GET /** Returns a welcome message: "Welcome to the backend demo"

**2. POST /upload-book** Creates a new book document in the **bookCollection** collection. Expects a JSON body with book data.

**3. GET /get-books** Retrieves all book documents from the **bookCollection** collection.

**4. PATCH /book/:id** Updates a book document with the specified **id** in the **bookCollection** collection. Expects a JSON body with updated book data.

**5. DELETE /books/:id** Deletes a book document with the specified **id** from the **bookCollection** collection.

**6. GET /all-books** Retrieves all book documents from the **bookCollection** collection, filtered by **category** if a query parameter **category** is provided.

**7. GET /book/:id** Retrieves a single book document with the specified **id** from the **bookCollection** collection.

**User Management**

* **GET /api/user/** - Retrieves user information by ID.
* **PUT /api/user/** - Updates user information by ID.

**Book Management**

* **GET /api/books** - Retrieves all books.
* **POST /api/books** - Creates a new book entry.
* **GET /api/books/** - Retrieves a book by ID.
* **PUT /api/books/** - Updates a book by ID.
* **DELETE /api/books/** - Deletes a book by ID.

**Cart Management**

* **GET /api/cart** - Retrieves all items in the user's cart.
* **POST /api/cart** - Adds an item to the cart.
* **PUT /api/cart/** - Updates the quantity of an item in the cart.
* **DELETE /api/cart/**- Removes an item from the cart.

**Wishlist Management**

* **GET /api/wishlist** - Retrieves all items in the user's wishlist.
* **POST /api/wishlist** - Adds an item to the wishlist.
* **DELETE /api/wishlist/** - Removes an item from the wishlist.

### **Integration with Frontend** The backend communicates with the frontend via RESTful APIs. Key points of integration include:

### **User Authentication**

* Tokens are passed between frontend and backend to handle authentication.
* Frontend components include login and registration forms that interact with /api/user/login and /api/user/register.

### **Data Fetching**

* Frontend components make API calls to fetch necessary data for display and interaction.
  + Book details: API calls to /api/books and /api/books/:id.
  + Cart management: API calls to /api/cart.
  + Wishlist management: API calls to /api/wishlist.

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

**Error Handling**

* Centralized error handling using middleware. Middleware functions catch errors and format them consistently before sending responses to the client.
* Example middleware:

javascript

Copy code

function errorHandler(err, req, res, next) {

res.status(err.status || 500).json({

message: err.message || 'Internal Server Error',

});

}

.