**Full Stack Development with MERN**

**API Development and Integration Report**

|  |  |
| --- | --- |
| Date | 11 July 2024 |
| Team ID | SWTID1720010842 |
| Project Name | Book a Doctor |
| Maximum Marks | 10 Marks |

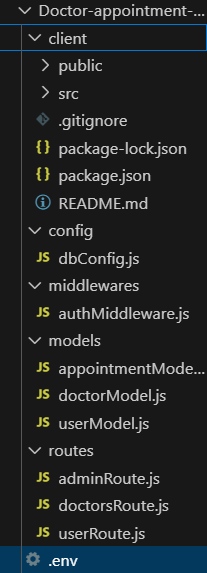
**Project Title:** Book a Doctor  
**Date:** 11-07-24  
**Prepared by:** Swathie P, Jaiganesh S

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

**Technologies Used**

* **Backend Framework:** Node.js with Express.js
* **Database:** MongoDB
* **Authentication:** JWT (JSON Web Token)

**Project Structure**



**Key Directories and Files**

1. **/controllers**

Contains functions to handle requests and responses for various endpoints, such as appointment booking, user management, and doctor management.

1. **/models**

Includes Mongoose schemas and models for MongoDB collections, such as User, Doctor, and Appointment schemas.

1. **/routes**

Defines the API endpoints and links them to controller functions. This includes routes for user authentication, doctor listings, appointment bookings, etc.

1. **/middlewares**

Custom middleware functions for request processing, such as authentication checks, error handling, and validation.

1. **/config**

Configuration files for database connections, environment variables, and other project-wide settings.

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

**User Authentication**

* **POST /api/user/register** - Registers a new user.
* **POST /api/user/login** - Authenticates a user and returns a token.

**User Management**

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

**Doctor Management**

* **GET /api/** **doctors** - Retrieves all doctors.
* **POST /api/** **doctors** - Adds a new doctor.
* **GET /api/doctors/ -** Retrieves a doctor by ID.
* **PUT/api/** **doctors/** - Updates doctor information by ID.

**Appointment Management**

* **GET /api/** **appointments** - Retrieves all appointments.
* **POST /api/** **appointments** - Books a new appointment.
* **GET /api/** **appointments/** - Retrieves an appointment by ID.
* **PUT /api/appointments/** - Updates appointment details by ID.

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

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

* **Error Handling:** Centralized error handling using middleware to catch and respond to errors consistently.
* **Validation:** Input validation using libraries like Joi or express-validator to ensure data integrity and correctness.

**Security Considerations**  
Outline the security measures implemented:

* **Authentication:** Secure token-based authentication using JWT to ensure only authorized users can access certain endpoints.
* **Data Encryption:** Encrypt sensitive data at rest and in transit to protect user information and appointment details.