API Development and Integration Report

Date	22-07-2024
Team ID	SWTID1720162063
Project Name	Cab Booking Application
Maximum Marks	

Project Title: [Cab Booking Application]
Date: [22-07-24]
Prepared by: [Jaswanth(Team lead),Mothish(Team Member),Rishi(Team Member),Prasad(Team member).]

Objective

The objective of this report is to document the API development progress and key aspects of the backend services implementation for the Cab Booking Application project. This includes a detailed overview of the technologies used, the project structure, and the design of API endpoints. The report aims to provide a comprehensive understanding of how different components of the backend interact to deliver a seamless and secure cab booking experience. Additionally, it covers the integration points with the frontend application, error handling strategies, validation mechanisms, and security measures to ensure the reliability and integrity of the system.

Technologies Used

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

Project Structure

Provide a screenshot of the backend project structure with explanations for key directories and files.

Key Directories and Files

1. **/controllers**
    - Contains functions to handle requests and responses.
2. **/models**
    - Includes Mongoose schemas and models for MongoDB collections.
3. **/routes**
    - Defines the API endpoints and links them to controller functions.
4. **/middlewares**
    - Custom middleware functions for request processing.
5. **/config**
    - Configuration files for database connections, environment variables, etc.

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.

Cab Management

- **GET /api/cabs** - Retrieves all cabs.
- **POST /api/cabs** - Adds a new cab.
- **PUT /api/cabs/**

- Updates cab details by ID.

- **DELETE /api/cabs/**

  - Deletes a cab by ID.

Driver Management

- **GET /api/drivers** - Retrieves all drivers.
- **POST /api/drivers** - Adds a new driver.
- **PUT /api/drivers/**

  - Updates driver details by ID.

- **DELETE /api/drivers/**

  - Deletes a driver by ID.

Booking Management

- **GET /api/bookings** - Retrieves all bookings.
- **POST /api/bookings** - Creates a new booking.
- **GET /api/bookings/**

  - Retrieves booking details by ID.

- **PUT /api/bookings/**

  - Updates booking details by ID.

- **DELETE /api/bookings/**

  - Cancels a booking 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.
- **Validation:** Input validation using libraries like Joi or express-validator.

Security Considerations

Outline the security measures implemented:

- **Authentication:** Secure token-based authentication.
- **Data Encryption:** Encrypt sensitive data at rest and in transit.