**SuiteSpot**

**Project Description**

This is a fullstack hotel website created using the MERN stack. This website contains multiple pages that showcase the hotel's amenties and offerings and allows users to browse multiple rooms and book them. All bookings are stored and managed in the backend and user's are only able to book rooms that are currently available based on their selected dates as well as number of guests.

**Requirement Document:**

The SuiteSpot consists of three main components:

* Frontend
* Backend
* Database

**Technologies**

The app was created with the following technologies:

* MongoDB
* Express.JS
* ReactJS
* NodeJS

**Features**

* User Authentication: Authentication functionality enables users to register, login, and manage their accounts securely.
* Room Browsing: Users can browse available rooms, view details, and amenities.
* Booking Management: Users can create, view, and delete bookings. The backend handles date collision to prevent double bookings.
* Responsive Design: The website is designed to be fully responsive, ensuring optimal user experience across various devices and screen sizes.

**Front-end**

To build the front end, we use frameworks and libraries such as:

* ReactJS, which is a popular JavaScript library for building user interfaces.
* We also use CSS which are styling frameworks that help make the user interface look good and responsive.
* Additionally, we use some npm packages to add extra functionality to the front end.
* Finally, we use a development environment called VSCode, which is a popular code editor, to develop the front end.

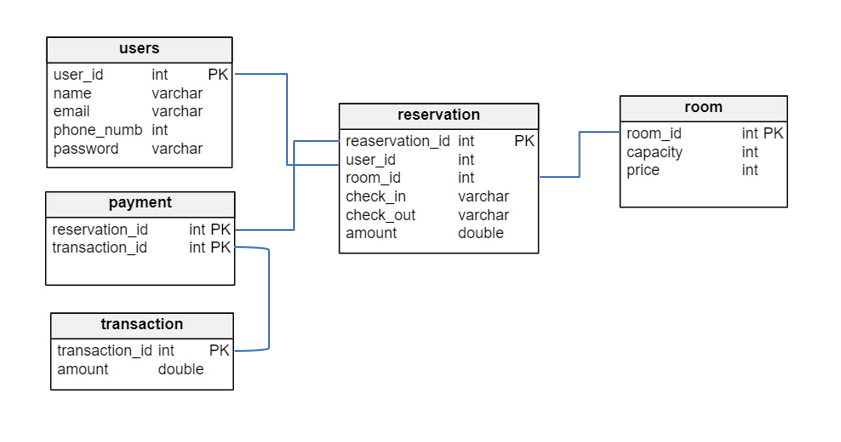
**Back-end**

* Express Routes: Routes are defined to handle HTTP requests from the client, including endpoints for room and booking operations.
* MongoDB Integration: Mongoose is used as an ODM (Object Data Modeling) library to interact with the MongoDB database, defining schemas for rooms and bookings.
* Middleware: Middleware functions are employed for tasks such as authentication, error handling, and date collision checking.

**Data Models and Database Schema:**

The back end of SuiteSpot uses a range of data models and database schemas to manage data, including:

* User schema: Includes fields such as name, email, password, phone number.
* Admin Schema: Admin contains all the details of hotel reservations.



**Frameworks, Libraries, and Tools used:**

The back end of SuiteSpot uses a range of frameworks, libraries, and tools to ensure its functionality and performance, including:

* Node.js: Node.js is used as the primary framework for the back end.
* MongoDB: MongoDB is used as the primary database, providing a flexible and scalable data storage solution.
* Express.js: Express.js is used as a web application framework, providing a range of features and tools for building web applications.
* Mongoose: Mongoose is used as an Object Data Modelling (ODM) library, providing a way to interact with MongoDB using JavaScript.

**API Design:**

The SuiteSpot platform's API is implemented using Node.js and Express.js. It follows standard HTTP request methods such as GET, PUT, and DELETE. Sample API requests and responses:

* GET /api/rooms: Get all room Response: A list of all rooms in the database
* GET /api/room/:id: Get a single room by ID Response: The room with the specified ID.
* PUT /api/room/:id: Add a new room by ID Request: The new room will be added
* DELETE /api/room/:id: Delete a room by ID Response: A success message indicating that the room has been deleted.

**Deployment:**

The deployment process for the SuiteSpot will involve hosting the application on various cloud-based services. The front end will be deployed using Vercel, a popular hosting service for static sites built with React. The back-end will be hosted on Render or Railway, two cloud-based hosting services for applications built with Node.js and MongoDB. The database will be hosted on MongoDB Atlas, a fully managed cloud database service. Overall, the deployment process for SuiteSpot will ensure a stable and scalable hosting environment for the application, allowing users to access the platform seamlessly from anywhere in the world.

**Future Enhancements:**

Enhancing a hotel booking website involves improving various aspects to provide a better user experience, increase customer satisfaction, and drive more bookings. Here are some potential future enhancements for a hotel booking website:

* Personalization: Implement algorithms to personalize the user experience based on user behavior, preferences, and past bookings. This could include suggesting relevant hotels, room types, amenities, and promotions tailored to each user.
* Virtual Reality (VR) Tours: Offer virtual tours of hotel rooms and facilities using VR technology. This allows potential guests to explore the property virtually, giving them a better sense of the accommodations before booking.
* Voice Search and Commands: Integrate voice search functionality, allowing users to search for hotels, rooms, and amenities using voice commands. Voice assistants like Amazon Alexa or Google Assistant could also be integrated to help users book rooms or answer questions.
* Reviews and Ratings System: Enhance the reviews and ratings system to provide more detailed and trustworthy feedback from previous guests. Implement features such as verified reviews, photo uploads, and star ratings for different aspects of the hotel (e.g., cleanliness, location, service).

**Conclusion**

The MERN stack hotel website serves as a comprehensive demonstration of fullstack development, incorporating frontend and backend technologies to deliver a functional and user-friendly application. The project provided valuable learning experiences in areas such as state management, backend development, and database interaction. Further enhancements and scalability can be achieved through additional features and optimizations in future iterations.