**Hotel Management**

**Introduction:** Welcome to our advanced and efficient hotel management system! Our all-inclusive software has been carefully designed to cater to every aspect of running a successful hotel. With features such as "room" management, you can effortlessly keep track of all available rooms, their types, and occupancy statuses, ensuring seamless guest accommodation. The "reservation" module allows you to handle bookings flawlessly, keeping guests informed of their check-in and check-out dates. You can rest easy with our secure "payment" processing, allowing smooth transactions and providing guests with a convenient payment experience. Our "special offer" section enables you to attract more guests by creating and managing attractive deals and promotions. With "users management," you have complete control over access and permissions, ensuring secure and personalized interactions with the system. Lastly, the "check in out" and "credit card" functionalities make guest arrivals and departures a breeze, while also securely managing credit card details. Say goodbye to tedious manual tasks and embrace our hotel management system to elevate your hotel's efficiency and provide an exceptional experience for both your guests and staff.

**Technology**:  
Our cutting-edge hotel management project boasts a powerful and robust backend, leveraging the latest technologies. The backend is built on Spring Boot, a framework renowned for its scalability and rapid development capabilities. With the inclusion of JPA (Java Persistence API), data management becomes a breeze, simplifying database interactions and ensuring seamless communication with our Postgres database. The combination of Spring Boot and JPA provides a strong foundation for handling reservations, payments, user management, and room availability, all with optimal performance.

For the front end, we have harnessed the power of React JS, a highly popular and efficient JavaScript library. With its component-based architecture and virtual DOM, React JS enables smooth, interactive, and responsive user interfaces. Our modern and sleek user interface design allows guests to effortlessly browse available rooms, make reservations, and explore special offers with ease and finesse.

To ensure seamless deployment and hosting of our hotel management system, we have chosen the reliable and efficient Tomcat server. Tomcat is a trusted and widely used Java application server, known for its stability and high performance. It ensures smooth handling of web requests and seamless communication between the frontend and backend, providing an unparalleled user experience for our valued guests.

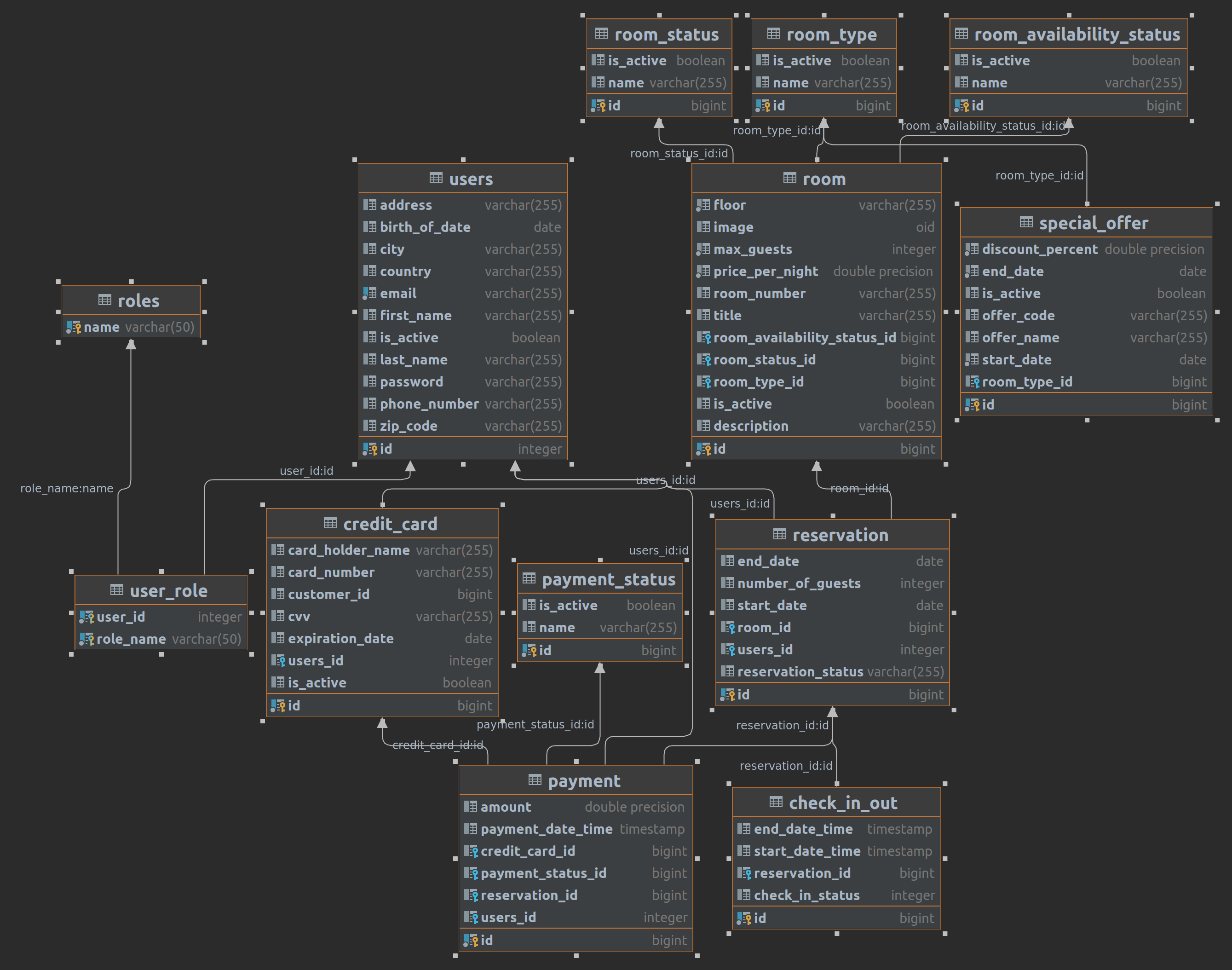
In summary, our hotel management project utilizes the best of breed technologies to deliver a high-performance, user-friendly, and secure application. The Spring Boot, JPA, and Postgres backend, coupled with the React JS front end and Tomcat server, come together to provide an exceptional hotel management solution that meets the needs of both guests and hotel staff alike. With our technology stack, you can elevate your hotel's operations to new heights, ensuring a seamless and delightful experience for everyone involved.

**Database:**

1. credit\_card: The credit\_card table stores information about the credit cards used by guests for payments. It may include details like credit card number, expiration date, cardholder's name, and a link to the guest or their reservation.
2. payment: The payment table records various payment transactions made by guests. It may include details like payment ID, guest ID, reservation ID, payment amount, payment date, and any other relevant payment-related information.
3. payment\_status: This table contains predefined payment statuses, such as "canceled" "paid" "failed" etc. It is likely used to keep track of the status of each payment transaction made by guests.
4. reservation: The reservation table stores information about guest reservations. It may include details like reservation ID, guest ID, room number, check-in date, check-out date, and any special requests or comments made during the reservation process.
5. roles: The roles table defines the different roles that users can have within the hotel management system. Roles may include "admin," "staff," "guest," etc., and are used to manage access and permissions for different users.
6. room: This table contains information about the rooms available in the hotel. It may include details like room number, room type ID (linked to room\_type table), maximum occupancy, amenities, and any other relevant room-related information.
7. room\_availability\_status: The room\_availability\_status table likely stores predefined room availability statuses, such as "reserved," "open," etc. It is used to track the current status of each room in the hotel.
8. room\_status: This table records the status of each room at any given time. It may include “ready”, “not ready”.
9. room\_type: The room\_type table defines various types of rooms available in the hotel, such as "gorgeous," "mid," "low," etc.
10. special\_offer: The special\_offer table stores information about any special offers or discounts provided by the hotel. It may include details like offer ID, offer code, offer name, offer description, applicable dates, and any conditions or requirements to avail of the offer.
11. user\_role: The user\_role table associates users with their corresponding roles. It establishes the relationship between users and the roles they have within the hotel management system.
12. users: The users table contains information about the system's users. It may include details like user ID, username, password (stored securely with encryption), contact information, and any other relevant user-related data.
13. check\_in\_out: This table tracks the check-in and check-out details of guests. It likely contains information such as guest ID, reservation ID, room number, check-in date, check-out date, and any additional relevant details related to the guest's stay.

These database tables form the foundation of the hotel management system, allowing hotel staff to efficiently manage reservations, check-ins, check-outs, payments, room availability, and other essential hotel operations. The system assists in providing excellent guest service while ensuring smooth and organized management of the hotel's resources.

**Database Diagram:**



Dependency:

1. Spring Boot Starter Data JPA:Spring Boot Starter Data JPA enables seamless integration with JPA (Java Persistence API), simplifying data access and management, and providing database support for the application.
2. Spring Boot Starter Web:Spring Boot Starter Web facilitates the development of web applications by providing necessary dependencies for handling HTTP requests and responses.
3. Spring Boot Starter Validation:Spring Boot Starter Validation includes validation capabilities to ensure data integrity and enforce business rules within the application.
4. Jackson Datatype JSR310:Jackson Datatype JSR310 enhances JSON serialization and deserialization with support for Java 8's JSR310 date and time classes.
5. Apache Commons Lang3:Apache Commons Lang3 is a utility library offering additional functionality and helper methods for common programming tasks.
6. Spring Boot Starter Security:Spring Boot Starter Security enables easy integration of security features, allowing the application to manage authentication and authorization.
7. Springfox Boot Starter and Springfox Bean Validators:Springfox Boot Starter and Springfox Bean Validators are used for automatic API documentation generation and validation, making it easier for developers to interact with the API.
8. PostgreSQL (Runtime Scope):PostgreSQL is a popular and feature-rich open-source relational database management system used as the runtime database for data storage.
9. Project Lombok (Optional):Project Lombok simplifies Java code by reducing boilerplate code, enhancing readability, and improving code maintainability.
10. Spring Boot Starter:Spring Boot Starter provides essential components to kickstart a Spring Boot application, offering a convenient and efficient way to build a robust backend.

Controller Diagram:

