A group of women with luggage in a hotel lobby

Description automatically generated

**Hotel Management System**

**Group 4**

**Apekshya , Kuwar , Suyang**

Contents

[Executive Summary 2](#_Toc127454286)

[Design Analysis Process 2](#_Toc127454287)

[Entity Relationship Diagram 9](#_Toc127454288)

[Table Designs – Data Dictionary 10](#_Toc127454289)

[Contributions 11](#_Toc127454290)

# **Executive Summary**

The Hotel Management System is a comprehensive software solution designed to simplify and automate the core operations of a hotel, including customer registration, room reservations, check-ins, and check-outs. It provides an intuitive platform for customers to browse room types, check availability, and make reservations, while offering administrators powerful tools to manage room statuses, track customer activity, and generate detailed operational reports. With its robust database structure, the system ensures data integrity through well-defined relationships between entities such as customers, rooms, and reservations. By integrating essential hotel operations into a single platform, this system enhances efficiency, reduces manual effort, and provides a seamless experience for both customers and administrators, making it an ideal solution for modern hotel management.

# **Design Analysis Process**

**User Stories for the System**

Booking Module

1. As a customer, I want to view real-time room availability so that I can choose a room that fits my budget and preferences.
2. As a customer, I want to book a room online so that I can secure my stay without visiting the hotel.
3. As a customer, I want to receive booking confirmation and check-in reminders so that I can plan my trip without forgetting important details.
4. As a hotel manager, I want to monitor current reservations so that I can manage room availability and avoid overbooking.
5. As a hotel manager, I want to update room availability dynamically so that the system reflects accurate data for customers.

Check-in Module

1. As a customer, I want to check in using my online booking details so that I can avoid delays at the reception.
2. As an Administrator, I want to view the check-in details of all customers currently checked in, so that I can monitor room occupancy and manage hotel operations efficiently.
3. As a hotel staff member, I want to access a guest's booking details so that I can quickly complete the check-in process.

4.As an Administrator, I want to view the check-in details of all customers currently checked in, so that I can monitor room occupancy and manage hotel operations efficiently.

**Functional and non-functional requirements**

**Functional Requirements for the Hotel Management System**

The functional requirements outline the key features and operations the system must support to meet its goals effectively. These are categorized based on the system's components and user roles.

**Customer Functional Requirements**

1. **User Registration:**
   * Customers must be able to register an account by providing their name, email, phone number, and password.
   * The system should validate the uniqueness of the email during registration.
2. **User Login:**
   * Customers should log in using their registered email and password.
   * The system must authenticate the credentials before granting access.
3. **View Room Types:**
   * Customers should view detailed information about available room types, including pricing, guest capacity, and descriptions.
4. **Check Room Availability:**
   * Customers must check room availability by providing search criteria such as room type, check-in date, and check-out date.
   * The system should display a list of available rooms based on the criteria.
5. **Make Reservations:**
   * Customers should select room types, specify the number of rooms, and confirm their reservation.
   * The system must calculate the total price and save the reservation in the database.
6. **Modify Reservations:**
   * Customers must update their reservation details, such as check-in date, check-out date, or room type, before the reservation date.
7. **Cancel Reservations:**
   * Customers must be able to cancel their reservations, and the system should update room availability accordingly.
8. **View Reservation History:**
   * Customers should view a list of their past and current reservations with detailed information.

**Administrator Functional Requirements**

1. **Manage Room Types:**
   * Administrators must be able to add, update, or delete room types, including price and capacity details.
2. **Manage Rooms:**
   * Administrators should add, update, or remove room records, including room numbers, status, and type associations.
3. **Process Check-Ins:**
   * Administrators must process customer check-ins, assign specific rooms, and update the room's status to "occupied."
4. **Process Check-Outs:**
   * Administrators should process customer check-outs, release the rooms, and update the room's status to "available."
5. **Manage Customer Details:**
   * Administrators must view and manage customer details, including contact information and reservation history.
6. **Generate Reports:**
   * Administrators should generate detailed reports on room availability, reservation statistics, customer activity, and revenue.

**System Functional Requirements**

1. **Database Management:**
   * The system must store and manage data for customers, rooms, room types, reservations, and check-in/check-out details.
2. **Real-Time Updates:**
   * The system should provide real-time updates on room availability and reservation statuses.
3. **Security:**
   * The system must securely handle sensitive data, including passwords and payment details (if applicable).
   * Authentication and authorization should be enforced for both customers and administrators.
4. **Notification System:**
   * Customers should receive notifications for reservation confirmations, cancellations, and modifications.
5. **Payment Integration (Future Scope):**
   * The system should allow customers to make payments online during reservation or check-in (optional).
6. **Error Handling:**
   * The system must handle errors gracefully and provide appropriate feedback, such as invalid login attempts or unavailable room types.
7. **Scalability:**
   * The system should support the addition of new room types, rooms, and features as the hotel expands.

**Non-Functional Requirements**

1.Performance and Scalability: Quick Search and Response Time: Even with a high load, the system should react to room searches and booking activities in two to three seconds.   
Scalable Infrastructure: Especially during busy times (holidays, promotions), the database and server architecture should be able to accommodate growing user and transaction volumes.   
High Availability: 99.9% uptime is required for the system to guarantee that visitors can make or change reservations at any time.

2. Privacy and Security of Data:   
Secure Data Storage: Encryption technologies like AES-256 and secure password hashing (like bcrypt) should be used to safely store sensitive data, such as payment information and personal information.   
Legal Regulation Compliance: When managing personal and payment data, the system must abide by the CCPA, GDPR, and PCI-DSS.   
Use two-factor authentication (2FA) for admin logins and visitor accounts.

3. Reliability and Fault Tolerance: Recovery and Backup: In the case of a system failure, daily database backups and disaster recovery protocols should guarantee little data loss.   
Fault Detection: With auto-recovery and administrator notifications, the system should be able to identify and manage failures with grace.   
Using automatic session timeouts for security, session management makes ensuring that user sessions are appropriately preserved across interactions.   
4. Accessibility and Usability:   
Intuitive UI/UX: To cut down on time spent on tasks (such as making a reservation or browsing reservations), the system should offer a user-friendly interface for both visitors and hotel employees.   
Mobile Accessibility: To ensure a smooth experience on smartphones and tablets, the system should be responsive and mobile device optimized.   
Compliance with Accessibility: WCAG (Web Content Accessibility) should be followed by the system.

5.Extensibility and Maintainability: Modular Design: Simple updates, feature additions (like a loyalty program), and connections with external tools are made possible by a modular system architecture.   
Monitoring and Logging: Detailed logs should document transactions, problems, and system behavior. Real-time system performance monitoring should be possible for administrators.   
Automated Testing: For essential elements (such as the booking flow and payment processing), the system should have automated testing processes.   
6. Localization and Internationalization:   
Support for Multiple Currencies: Depending on the user's location, the system ought to enable payments in various currencies.   
Support for Multiple Languages: For visitors from abroad, the system should be able to communicate in English, Spanish, and French, among other languages.   
Timezone Handling: The system should appropriately handle reservations made in different time zones, converting check-in and check-out hours according to the user's location.

7.Data Integrity and Consistency: ACID Compliance: To guarantee transaction integrity, especially for booking, payment, and cancellation procedures, the database should adhere to ACID (Atomicity, Consistency, Isolation, Durability) standards.   
Uniformity Among Systems: Make sure that the database, backend processes, and front-end user interface are all consistent to avoid inconsistencies like double booking or inaccurate billing.   
8. Resource Optimization and Cost Efficiency:   
Effective Resource Usage: To cut down on resource usage (such as CPU and memory) and hosting expenses, optimize database queries and backend operations.   
Cloud Scalability: If the system is hosted in the cloud, elastic scaling should be used to modify resources in response to demand, guaranteeing cost effectiveness during off-peak hours.

**User cases**

**User Case 1: User Registration**

Description: Users can register an account by providing their name, email, phone number, and password.  
Actor: Customer  
Precondition: The user does not already have an account.  
Postcondition: A user account is created and stored in the CUSTOMER table.  
Main Flow:

1. The user provides registration information.
2. The system validates the data.
3. The system saves the user details into the database.

**User Case 2: User Login**

Description: Users can log into the system using their email and password.  
Actor: Customer  
Precondition: The user is already registered.  
Postcondition: The user is authenticated and gains access to the system.  
Main Flow:

1. The user enters their email and password.
2. The system verifies the credentials.
3. The user is granted access to the dashboard.

**User Case 3: View Room Types**

Description: Users can view available room types with details such as price and guest capacity.  
Actor: Customer  
Precondition: None  
Postcondition: Room type details are displayed to the user.  
Main Flow:

1. The user requests room type information.
2. The system fetches data from the ROOM TYPE table.
3. The system displays the room types and their details.

**User Case 4: Room Availability Check**

Description: Users can check the availability of rooms based on type and date.  
Actor: Customer  
Precondition: The user is logged in.  
Postcondition: Available rooms matching the criteria are displayed.  
Main Flow:

1. The user inputs search criteria (e.g., room type, dates).
2. The system checks the ROOM and RESERVATIONS tables for availability.
3. The system returns the available rooms.

**User Case 5: Make a Reservation**

Description: Users can reserve a room for a specified period.  
Actor: Customer  
Precondition: The user is logged in and has selected an available room.  
Postcondition: Reservation details are recorded, and the room is marked as reserved.  
Main Flow:

1. The user selects a room and submits a reservation request.
2. The system creates a record in the RESERVATIONS table.
3. The system updates the status of the reserved room in the ROOM table.

**User Case 6: Modify Reservation**

Description: Users can modify their reservation details, such as dates or room type.  
Actor: Customer  
Precondition: The user has an existing reservation.  
Postcondition: The updated reservation is saved in the system.  
Main Flow:

1. The user selects an existing reservation.
2. The user updates the reservation details.
3. The system updates the RESERVATIONS table.

**User Case 7: Cancel Reservation**

Description: Users can cancel an existing reservation.  
Actor: Customer  
Precondition: The user has an existing reservation.  
Postcondition: The reservation is canceled, and the room is marked as available.  
Main Flow:

1. The user selects a reservation to cancel.
2. The system deletes the record from the RESERVATIONS table.
3. The system updates the room status in the ROOM table.

**User Case 8: Check-In**

Description: The system administrator processes a customer check-in and assigns a room.  
Actor: System Administrator  
Precondition: The customer has a valid reservation.  
Postcondition: The check-in is recorded, and the room status is updated.  
Main Flow:

1. The administrator selects a customer’s reservation.
2. The system creates a record in the Check-In Details table.
3. The system updates the room’s status in the ROOM table.

**User Case 9: Check-Out**

Description: The system administrator processes a customer check-out, releasing the room.  
Actor: System Administrator  
Precondition: The customer has checked in.  
Postcondition: The check-out is recorded, and the room status is updated.  
Main Flow:

1. The administrator selects the check-in record.
2. The system records the check-out date in the Check-In Details table.
3. The system updates the room’s status to available in the ROOM table.

**User Case 10: View Customer Details**

Description: The system administrator can view customer information.  
Actor: System Administrator  
Precondition: The customer’s account exists in the system.  
Postcondition: Customer details are displayed to the administrator.  
Main Flow:

1. The administrator searches for a customer.
2. The system retrieves data from the CUSTOMER table.
3. The system displays the customer’s information.

**User Case 11: View Reservation History**

Description: Customers can view their past and current reservations.  
Actor: Customer  
Precondition: The customer has made at least one reservation.  
Postcondition: Reservation details are displayed to the user.  
Main Flow:

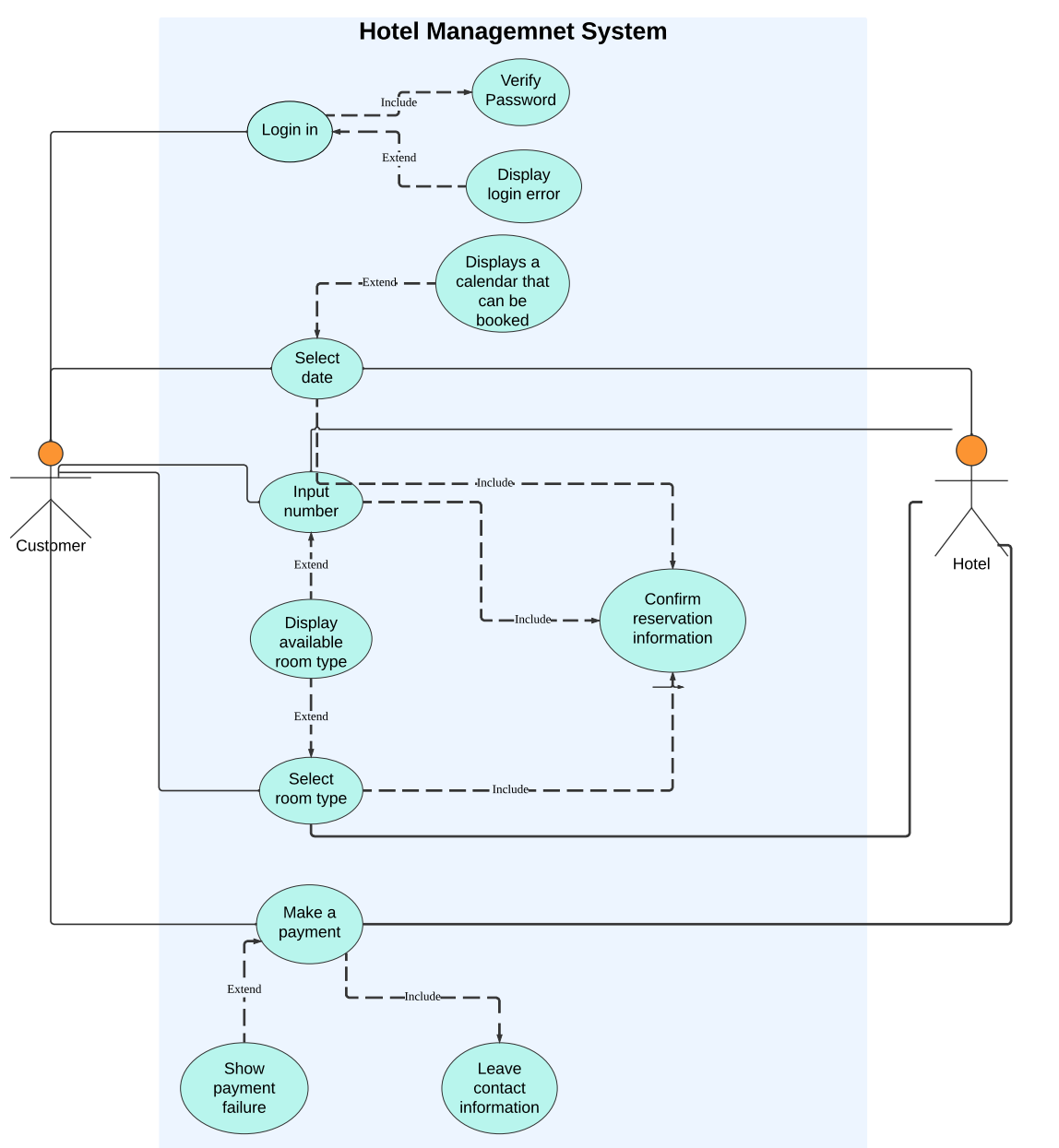
1. The customer requests their reservation history.
2. The system fetches data from the RESERVATIONS table.
3. The system displays the customer’s reservation history.

**User Case 12: Generate Reports**

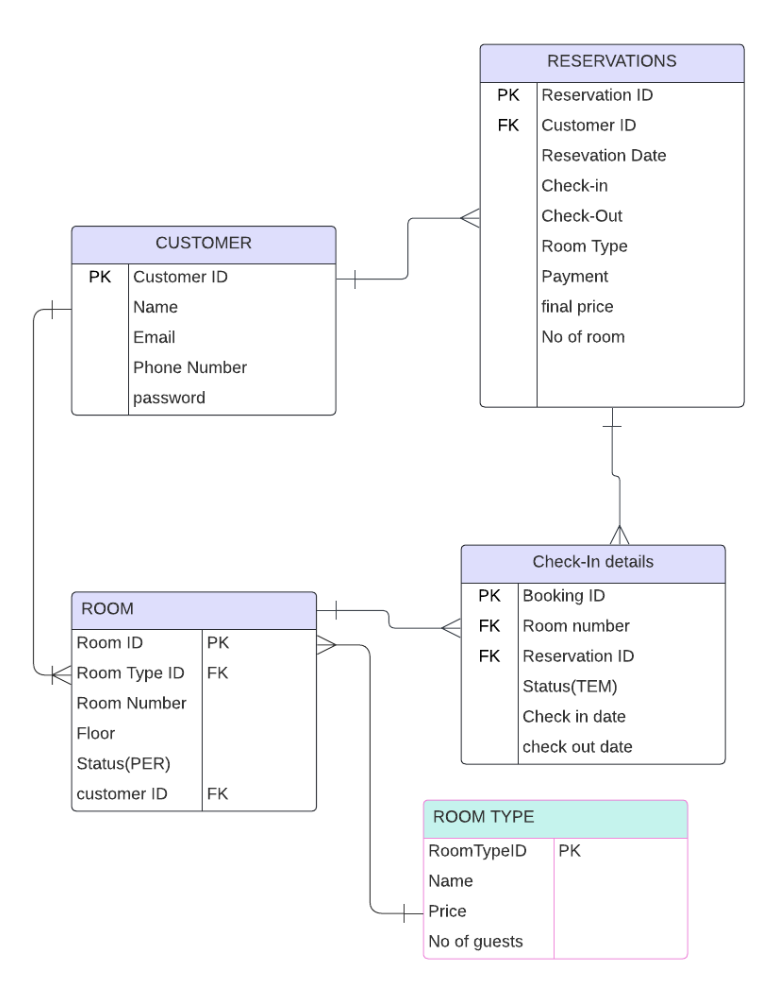
Description: The system administrator can generate reports on reservations, customer activity, and room availability.  
Actor: System Administrator  
Precondition: The system contains reservation, room, and customer data.  
Postcondition: The administrator receives a report.  
Main Flow:

1. The administrator selects the type of report to generate.
2. The system fetches relevant data.
3. The system generates and displays the report.

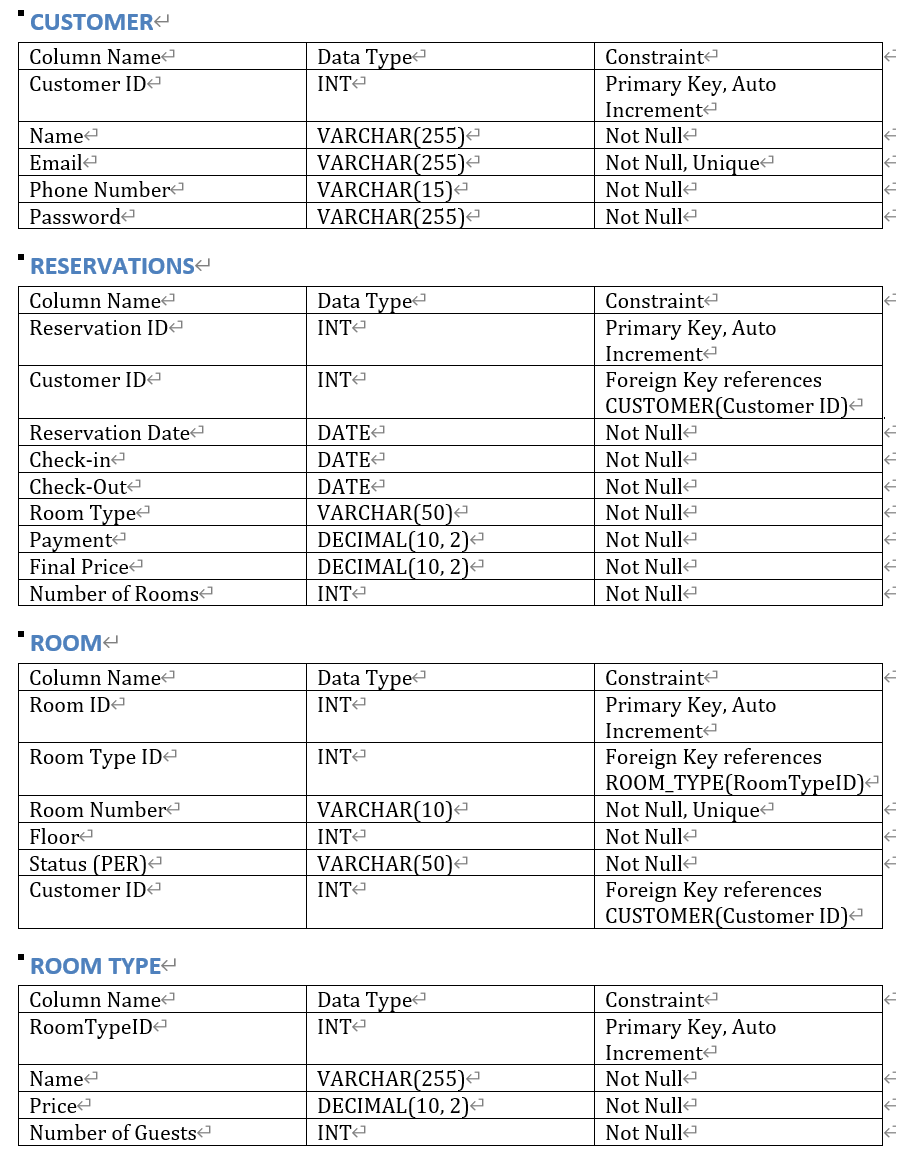
**use case diagrams**



# **Entity Relationship Diagram**



# **Table Designs – Data Dictionary**



A table of information

Description automatically generated with medium confidence

# **Contributions**

**Group number：**

Apekshya , Kuwar , Suyang

**GitHUb：**

https://github.com/Shiori0825-gao/Group4.git