**HOTEL MANAGEMENT SYSTEM**

**Shivani Sugurushetty - 002305622**

**Ashlesha Vivek Donde - 002537305  
Mohammed Sayed Hossain – 002562175**

**Meghana Thota – 002591062**

**Pranav Waghmare – 002513911**

**Dai Zhichao – 003183468**

**Github link:** [**https://github.com/dondeash2025/DMDD-Project-Group-4**](https://github.com/dondeash2025/DMDD-Project-Group-4)

**Mission:**

Hotel Management System is to design and implement a centralized, consistent and scalable solution that helps day – to – day operations of a hotel. The system will support the efficient management of reservations, staff duties, payments and bills, hotel availability, guest bookings, check-in and check-out procedures. It reduces manual labor, minimize errors, provides real – time information access, enhance customer satisfaction, provides prompt and precise service.

**Objectives:**

1. Booking Management

* Make new bookings and cancellations in real time.
* Keep track of room availability.
* Prevents double bookings by updating room availability.

1. Guest information management

* Maintain guest profiles like first name, last name, age, address, nationality etc.
* Ensure secure storage of personal data.

1. Room and facilities management

* Manage room assignments and status of the room (booked, available, under maintenance).
* Issue alerts for out – of – service rooms.

1. Check – in and Check – out process

* Log guest arrivals and departures efficiently.
* Create invoices at checkout that include information about room fees and services.

1. Billing and payment management

* Manage several payments options (online, card and cash)
* Maintain detailed billing history.

1. Analytics and reporting

* Generate reports

1. Staff scheduling and allocation

* Assign housekeeping tasks based on guest check-ins and check-outs.
* Track employee shifts and availability.
* Ensure fair and efficient workload distribution.

1. Data security and integrity

* Implement user authentication and role – based access control.
* Protect sensitive information from unauthorized access or manipulation.
* Maintain data accuracy and consistency.

1. Data backup and recovery

* Automated backup schedules to ensure data safety.
* Recovery procedures for system failures.

1. Scalability and maintainability

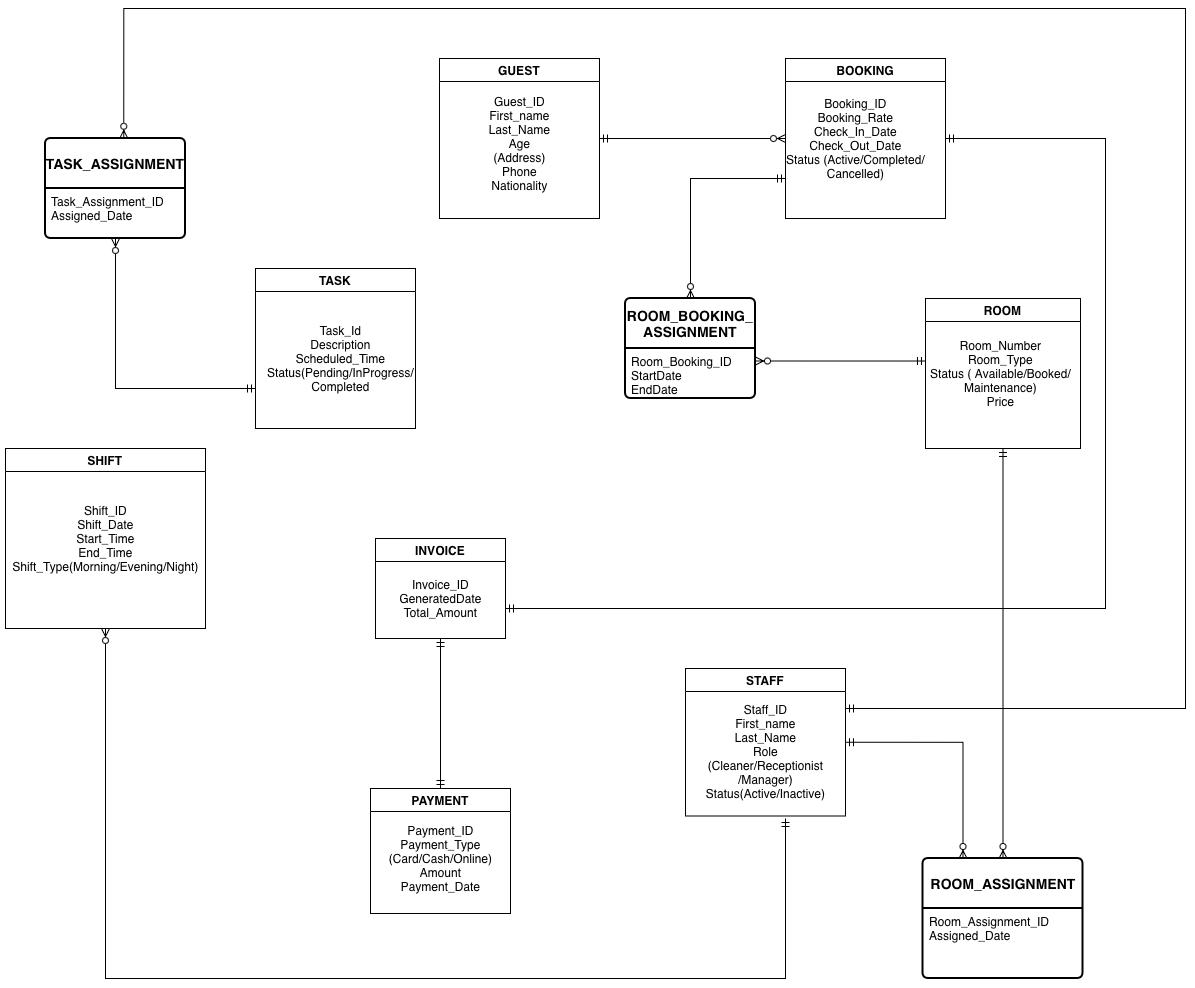
* Support future enhancements such as mobile integration, loyalty programs and third-party APIs.

**ERD:**

**A diagram of a room assignment

AI-generated content may be incorrect.**

**Updated ERD:**

****

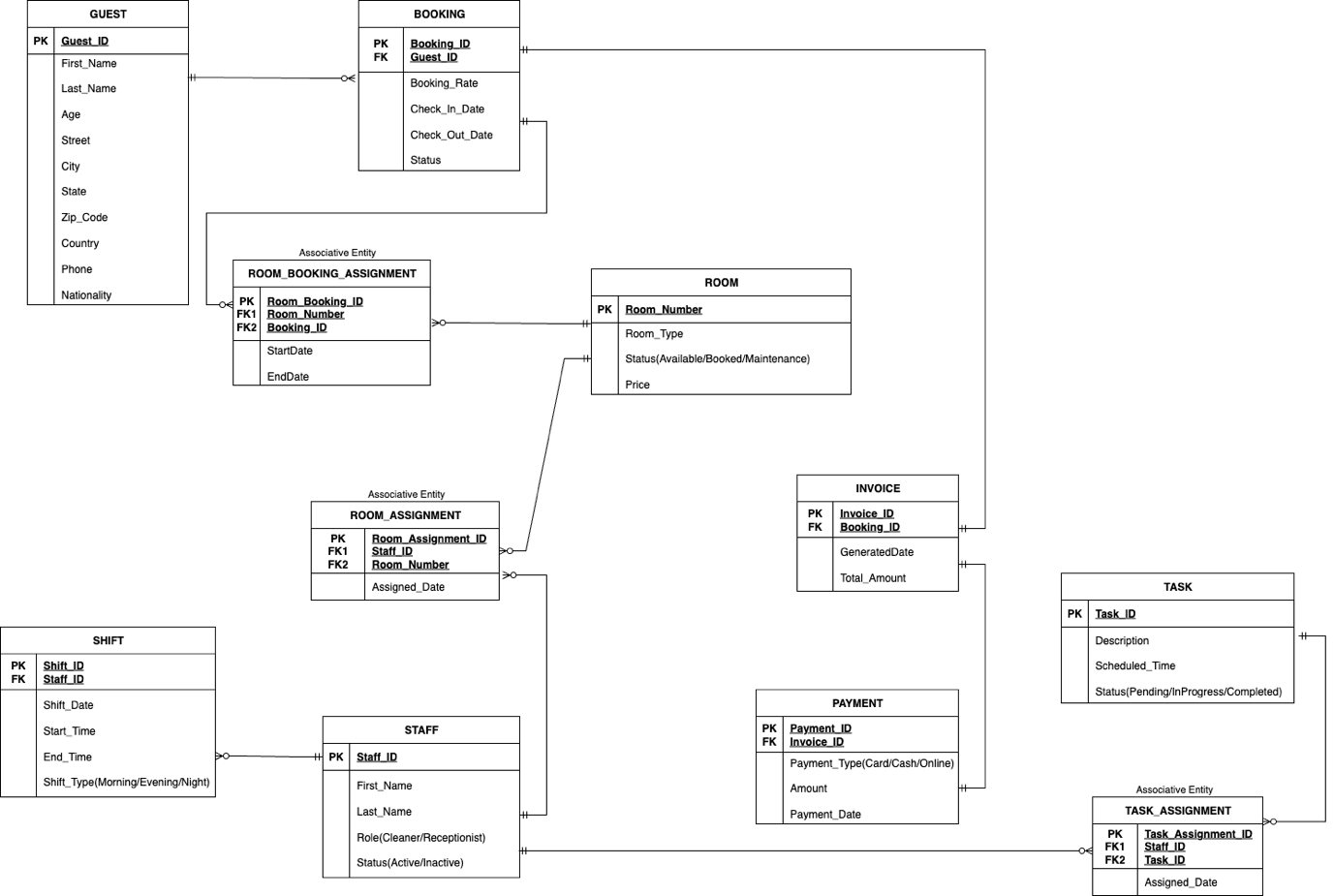
**Key Changes in ERD:**

* Introduced associative entity ROOM\_BOOKING\_ASSIGNMENT for ROOM – BOOKING relationship.
* Introduced associative entity TASK\_ASSIGNMENT for STAFF – TASK relationship.
* Expanded ROOM\_ASSIGNMENT as associative entity for STAFF – ROOM relationship.
* Added SHIFT entity linked to TASK.

**List of entities and their relationships:**

* Guest represents hotel customers. A guest can make multiple bookings but each booking is associated with only one guest.
* Booking represents a guest commitment to stay. A booking can include multiple rooms, and a room can appear in multiple bookings over time. This many-to-many relationship is implemented using the **Room\_Booking\_Assignment** entity.
* Rooms are physical units in the hotel. Rooms can be assigned to multiple staff for cleaning or maintenance.
* **Room\_Booking\_Assignment** links rooms and bookings. It captures the specific period Start\_Date and End\_Date for which each room is reserved under a booking.
* **Invoice** is generated once a booking is completed. Each booking generates one invoice that records the total amount due.
* **Payment** represents financial transactions associated with invoices. Each invoice has exactly one payment, and each payment is linked to a single invoice.
* **Staff** includes hotel employees such as housekeepers, maintenance workers, receptionists, and managers. Each staff member can have multiple shifts, handle multiple rooms, and be assigned multiple tasks.
* **Shift** records the working schedule for each staff member, including shift date, start and end times, and shift type (Morning, Evening, Night).
* **Task** includes operational activities like cleaning or repairs. Each task is scheduled for a specific time and has a status (Pending, In Progress, Completed).
* **Task\_Assignment** links staff and tasks, allowing multiple staff to perform multiple tasks. It records the date when each task is assigned.
* **Room\_Assignment** links staff and rooms, tracking which staff are responsible for which rooms and the date of assignment. It supports daily cleaning and maintenance scheduling.

**Logical ERD:**

****

**Key database design decisions:**

* Booking is flexible in tracking guest’s intent versus actual room allocation through Room\_Booking\_Assignment associative entity.
* Payment and Invoice are distinct to ensure financial integrity and traceability of transactions.
* **Room\_Assignment** and **Task\_Assignment** are associative entities designed to handle many-to-many relationships between rooms–staff and tasks–staff respectively.
* The **Shift** entity enables efficient scheduling of staff, tracking daily shift dates, start and end times, and shift types.
* **Composite attributes** like Address for Guest is decomposed into atomic attributes such as Street, City, State, Zip\_Code, and Country in the logical model to satisfy normalization.
* The model ensures full normalization up to Third Normal Form - no repeating groups, no partial dependencies, and no transitive dependencies.
* **Data integrity** is maintained using primary and foreign key relationships. All many-to-many relationships are resolved using associative entities.
* The database design is scalable for future extensions such as service-based billing, customer feedback, and staff performance tracking.