ODD SEMESTER 2023
Software Engineering Lab
(BTCSE 307)

Submitted for
BACHELOR OF TECHNOLOGY
(COMPUTER SCIENCE & ENGINEERING)

Submitted to:

Pooja Gupta
Assistant Professor
Deptt. of Computer Science & Engg.
Jamia Hamdard, New Delhi, India

Submitted by:

Shafia Ameeruddin
Enroll No: 2022-310-207
B. Tech (CSE), 3rd Semester
Section D

Department of Computer Science and Engineering
School of Engineering Sciences and Technology
Jamia Hamdard, New Delhi -110062

PROBLEM STATEMENT

A hotel management system (HMS) is a software solution designed to streamline and automate various operational and administrative tasks within a hotel or hospitality establishment. These systems help hotel staff manage day-to-day operations efficiently, improve guest services, and enhance overall guest experiences.

In the rapidly evolving hospitality industry, there is a pressing need for an efficient and integrated Hotel Management System (HMS) that addresses the multifaceted challenges faced by hotels of all sizes. The existing manual and disjointed systems employed by many hotels result in operational inefficiencies, subpar guest experiences, and missed revenue opportunities. These issues include:

1. Reservation Inefficiency: Hotel staff often struggle with manual reservation management, leading to overbooking, double bookings, and missed revenue from unoptimized room allocation.

2. Check-In and Check-Out Delays: Cumbersome check-in and check-out processes not only frustrate guests but also impact the hotel's ability to turn rooms over quickly, reducing overall occupancy rates.

3. Inadequate Guest Experience: The lack of a centralized system for managing guest preferences and service requests hinders the hotel's ability to provide a personalized and seamless experience.

4. Revenue Loss: Poor rate and inventory management can lead to revenue leakage as hotels struggle to adapt to dynamic pricing strategies and allocate rooms effectively across different distribution channels.

5. Resource Inefficiency: Manual management of housekeeping, maintenance, and inventory results in inefficiencies and increased operational costs.

6. Data Silos: Siloed information across departments and outdated technology hinders data-driven decision-making, preventing hotels from optimizing their operations and marketing efforts.

7. Security Concerns: Guest data security is a growing concern, and the absence of robust data security measures can lead to breaches, harming the hotel's reputation and causing legal issues.

8. Competitive Disadvantage: In a highly competitive industry, hotels that cannot keep up with technological advancements risk losing market share and falling behind competitors.

The objective of this project is to develop and implement a comprehensive Hotel Management System (HMS) that addresses these issues by offering an integrated, user-friendly, and scalable solution. This system will streamline hotel operations, enhance the guest experience, optimize revenue generation, and improve data security while ensuring the hotel remains competitive and adaptive in a rapidly evolving industry.

REQUIREMENTS FROM PROBLEM STATEMENT

Based on the problem statement for the hotel management system, several key requirements have been identified that the system should meet. These requirements are essential for addressing the identified challenges and achieving the solution's objectives. Here are the requirements derived from the problem statement:

1. Reservation Efficiency:
   - The system must support efficient reservation handling, including room availability checks, booking, modification, and cancellation.
   - It should prevent overbooking and double bookings through real-time room allocation updates.

2. Check-In and Check-Out Process:
   - The system should streamline the check-in and check-out processes, reducing waiting times and improving guest satisfaction.
   - It should enable quick and secure guest registration, room assignment, and key card issuance.

3. Guest Experience Enhancement:
   - The system must maintain a centralized guest profile database to capture and access guest preferences and previous stay history.
   - It should support in-room service requests, wake-up calls, and concierge services to provide a personalized guest experience.

4. Revenue Management Requirements:
   - The system needs to offer rate management capabilities, allowing for dynamic pricing adjustments based on demand and seasonality.
   - It should provide real-time inventory updates and distribution channel integration to maximize revenue.

5. Resource Efficiency:
   - The system should facilitate efficient housekeeping scheduling, maintenance requests, and inventory and stock control.

6. Data Silo Elimination Requirements:
   - A system that centralizes information and facilitates data sharing across departments.

7. Security Measures:
   - Robust data security measures must be implemented to protect guest data and financial transactions, including encryption and access control.

8. Competitive Advantage:
   - The system should be scalable and adaptable to meet changing industry demands and technology advancements, ensuring the hotel remains competitive.

These requirements serve as the foundation for developing and implementing an effective hotel management system that addresses the identified challenges and objectives. The successful implementation of such a system can significantly improve operational efficiency, guest satisfaction, and revenue generation for hotels.

Software Requirements Specification

for

HOTEL MANAGEMENT SYSTEM

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to provide a detailed description of the requirements for the development of a Hotel Management System (HMS). The HMS is intended to streamline and automate various operational and administrative processes within a hotel, enhancing overall efficiency and customer service.

1.2 Scope

The Hotel Management System will cover key aspects of hotel operations, including but not limited to:

- *Reservation Management: Handling room bookings, cancellations, and modifications.*
- *Guest Management: Maintaining guest information, preferences, and history.*
- *Room Management: Tracking room availability, cleaning schedules, and maintenance.*
- *Billing and Invoicing: Generating invoices, handling payments, and tracking expenses.*
- *Employee Management: Managing staff roles, schedules, and performance.*
- *Reporting and Analytics: Providing insights into occupancy rates, revenue, and other relevant metrics.*

1.3 Definitions, Acronyms, and Abbreviations

- *HMS: Hotel Management System*
- *GUI: Graphical User Interface*
- *API: Application Programming Interface*

2. Overall Description

2.1 Product Perspective

The HMS will be a standalone system, interacting with external systems for payment processing, online booking, and other integrations. It will provide a user-friendly GUI for both staff and administrators.

2.2 Product Features

2.2.1 Reservation Management

- *R1: Allow users to check room availability for a specific date range.*
- *R2: Enable users to make, modify, or cancel reservations.*
- *R3: Provide real-time updates on room availability.*

2.2.2 Guest Management

- *G1: Store and manage guest information securely.*

- *G2: Track guest preferences and history for personalized service.*
- *G3: Enable guest check-in and check-out processes.*

2.2.3 Room Management

- *RM1: Maintain a database of room types, rates, and amenities.*
- *RM2: Schedule and track room cleaning and maintenance.*
- *RM3: Manage room status (occupied, vacant, under maintenance).*

2.2.4 Billing and Invoicing

- *B1: Generate invoices for room charges, additional services, and taxes.*
- *B2: Accept and process various payment methods.*
- *B3: Provide receipts and billing history.*

2.2.5 Employee Management

- *EM1: Administer staff roles, responsibilities, and permissions.*
- *EM2: Create and manage employee schedules.*
- *EM3: Monitor employee performance and attendance.*

2.2.6 Reporting and Analytics

- *RA1: Generate reports on occupancy rates, revenue, and other key metrics.*
- *RA2: Provide analytics for informed decision-making.*
- *RA3: Support export of data for external analysis.*

2.3 User Classes and Characteristics

- *Guests: Individuals seeking accommodation.*
- *Staff: Hotel employees responsible for reservations, check-ins, maintenance, etc.*
- *Administrators: Managers and decision-makers overseeing the entire system.*

2.4 Operating Environment

The HMS will be developed as a web-based application, accessible through popular web browsers such as Chrome, Firefox, and Safari. The system will be hosted on a secure server with regular backups.

2.5 Design and Implementation Constraints

- *The system should be compatible with modern browsers.*
- *Compliance with data protection and privacy regulations.*
- *Integration capabilities with third-party payment gateways.*

2.6 Assumptions and Dependencies

- *Availability of a reliable internet connection for real-time updates.*
- *External APIs for payment processing and online booking.*

3. External Interface Requirements

3.1 User Interfaces

The system will feature an intuitive web-based GUI for both staff and administrators.

3.1.1 Staff Interface

- *Easy navigation for room management, guest check-in/out, and reservations.*
- *At-a-glance room status and availability.*
- *Secure login with role-based access.*

3.1.2 Administrator Interface

- *Access to comprehensive reports and analytics.*
- *User management for creating, modifying, and deleting accounts.*
- *Configuration options for system settings.*

3.2 Hardware Interfaces

The system will be accessed through standard computer hardware with internet connectivity.

3.3 Software Interfaces

The HMS will integrate with external systems for payment processing and online booking through well-defined APIs.

3.4 Communications Interfaces

The system will communicate with external payment gateways and online booking platforms using secure protocols.

4. System Features

4.1 Feature 1: Reservation Management

4.1.1 Description

- *This feature allows users to check room availability, make reservations, modify existing reservations, and cancel bookings.*

4.1.2 User Stories

- *As a guest, I want to check room availability for my preferred dates.*
- *As a guest, I want to make a new reservation for a specific room type.*
- *As a guest, I want to modify or cancel my existing reservation.*

4.1.3 Acceptance Criteria

- *The system should display real-time room availability.*
- *Users should receive confirmation emails upon successful reservation.*

4.2 Feature 2: Guest Management

4.2.1 Description

- *This feature involves the storage and management of guest information, preferences, and check-in/check-out processes.*

4.2.2 User Stories

- *As a staff member, I want to check in a guest efficiently.*

- *As a staff member, I want access to a guest's history and preferences.*
- *As a staff member, I want to process guest check-outs smoothly.*

4.2.3 Acceptance Criteria

- *The system should capture and store guest details securely.*
- *Staff should be able to retrieve guest information quickly during check-in.*

4.3 Feature 3: Room Management

4.3.1 Description

- *This feature involves the management of room types, rates, cleaning schedules, and maintenance.*

4.3.2 User Stories

- *As a staff member, I want to view and update room status.*
- *As a staff member, I want to schedule room cleaning and maintenance.*
- *As a staff member, I want to update room rates and amenities.*

4.3.3 Acceptance Criteria

- *The system should display the current status of each room.*
- *Cleaning and maintenance schedules should be visible and editable.*

4.4 Feature 4: Billing and Invoicing

4.4.1 Description

- *This feature involves generating invoices for room charges, handling various payment methods, and providing receipts.*

4.4.2 User Stories

- *As a guest, I want a clear and detailed invoice at the time of check-out.*
- *As a staff member, I want to process payments securely.*
- *As a staff member, I want to provide guests with electronic receipts.*

4.4.3 Acceptance Criteria

- *Invoices should include a breakdown of charges, taxes, and additional services.*
- *The system should support multiple payment methods.*

4.5 Feature 5: Employee Management

4.5.1 Description

- *This feature involves the administration of staff roles, schedules, and performance monitoring.*

4.5.2 User Stories

- *As an administrator, I want to assign roles and permissions to staff members.*
- *As a manager, I want to create and manage employee schedules.*
- *As a manager, I want to track employee performance and attendance.*

4.5.3 Acceptance Criteria

- *The system should allow administrators to define and modify staff roles.*
- *Managers should be able to create, modify, and view employee schedules.*

4.6 Feature 6: Reporting and Analytics

4.6.1 Description

- *This feature involves generating reports on occupancy rates, revenue, and providing analytics for decision-making.*

4.6.2 User Stories

- *As an administrator, I want to view daily, weekly, and monthly occupancy reports.*
- *As a manager, I want access to revenue reports and analytics for strategic planning.*
- *As a user, I want to export data for external analysis.*

4.6.3 Acceptance Criteria

- *The system should generate accurate and visually appealing reports.*
- *Analytics should provide actionable insights for decision-makers.*

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- *The system should handle a minimum of 500 simultaneous users.*
- *Response time for user interactions should be less than 2 seconds.*

5.2 Security Requirements

- *User authentication and authorization mechanisms.*
- *Encryption of sensitive data during transmission.*
- *Regular security audits and updates.*

5.3 Reliability and Availability

- *The system should have an uptime of at least 99.5%.*
- *Regular backups of the database to prevent data loss.*

5.4 Scalability

- *The system should be scalable to accommodate future growth.*
- *Ability to add more rooms, users, and features without significant performance degradation.*

5.5 Maintainability

- *Code should be well-documented.*
- *System updates and patches should be easy to deploy.*

6. Other Requirements

6.1 Legal and Regulatory Requirements

- *Compliance with data protection laws.*

- *Security measures in line with industry standards.*

6.2 Documentation Requirements

- *User manuals for staff and administrators.*
- *Technical documentation for system maintenance.*

6.3 Training Requirements

- *Training sessions for staff on system usage.*
- *Periodic refresher courses for updates and new features.*

7. Other Nonfunctional Requirements

7.1 Glossary

- *HMS: Hotel Management System*
- *GUI: Graphical User Interface*
- *API: Application Programming Interface*

7.2 Revision History

Version	***Date***	***Description***
1.0	*2023-11-10*	*Initial release of the SRS*

Conclusion

This Software Requirement Specification document provides a comprehensive overview of the requirements for the development of a Hotel Management System. It outlines the features, user stories, acceptance criteria, and non-functional requirements necessary for the successful implementation of the system. This document will serve as a guide for the development team, ensuring that the final product meets the needs and expectations of the stakeholders.

ENTITY RELATIONSHIP DIAGRAM

- Entity Relationship Model: Entity-Relationship model is used to represent a logical design of a database to be created. In ER model, real world objects (or concepts) are abstracted as entities, and different possible associations among them are modeled as relationships.
- Entity Set and Relationship Set: An entity set is a collection of all similar entities.
- Attributes of Entity: Attributes are the characteristics describing any entity belonging to an entity set. Any entity in a set can be described by zero or more attributes.
- Keys: One or more attribute(s) of an entity set can be used to define the following keys:
  - ➢ Super key: One or more attributes, which when taken together, helps to uniquely identify an entity in an entity set.
  - ➢ Candidate key: It is a minimal subset of a super key. In other words, a super key might contain extraneous attributes, which do not help in identifying an object uniquely. When such attributes are removed, the key formed so is called a candidate key.
  - ➢ Primary key: A database might have more than one candidate key. Any candidate key chosen for a particular implementation of the database is called a primary key.
  - ➢ Prime attribute: Any attribute taking part in a super key.
- Weak Entity: An entity set is said to be weak if it is dependent upon another entity set. A weak entity can't be uniquely identified only by its attributes. In other words, it doesn't have a super key.
- Entity Generalization and Specialization: Once we have identified the entity sets, we might find some similarities among them. ER model uses the "ISA" hierarchy to depict specialization (and thus, generalization).
- Mapping Cardinalities: One of the main tasks of ER modeling is to associate different entity sets. Let's consider two entity sets E1 and E2 associated by a relationship set R. Based on the number of entities in E1 and E2 are associated with, we can have the following four type of mappings:
  - ➢ One to one: An entity in E1 is related to at most a single entity in E2, and vice versa
  - ➢ One to many: An entity in E1 could be related to zero or more entities in E2. Any entity in E2 could be related to at most a single entity in E1.
  - ➢ Many to one: Zero or more number of entities in E1 could be associated to a single entity in E2. However, an entity in E2 could be related to at most one entity in E1.
  - ➢ Many to many: Any number of entities could be related to any number of entities in E2, including zero, and vice versa.
- ER Diagram: From a given problem statement we identify the possible entity sets, their attributes, and relationships among different entity sets. Once we have this information, we represent them pictorially, called an entity-relationship (ER) diagram.
  - ➢ Rectangles are used to represent entities.
  - ➢ Ellipse shapes are used to describe characteristics.
  - ➢ Diamond shapes are used to show relationships between entities.

Question 1: Draw the Entity-Relationship diagram for a project.
Answer 1: Hotel Management System

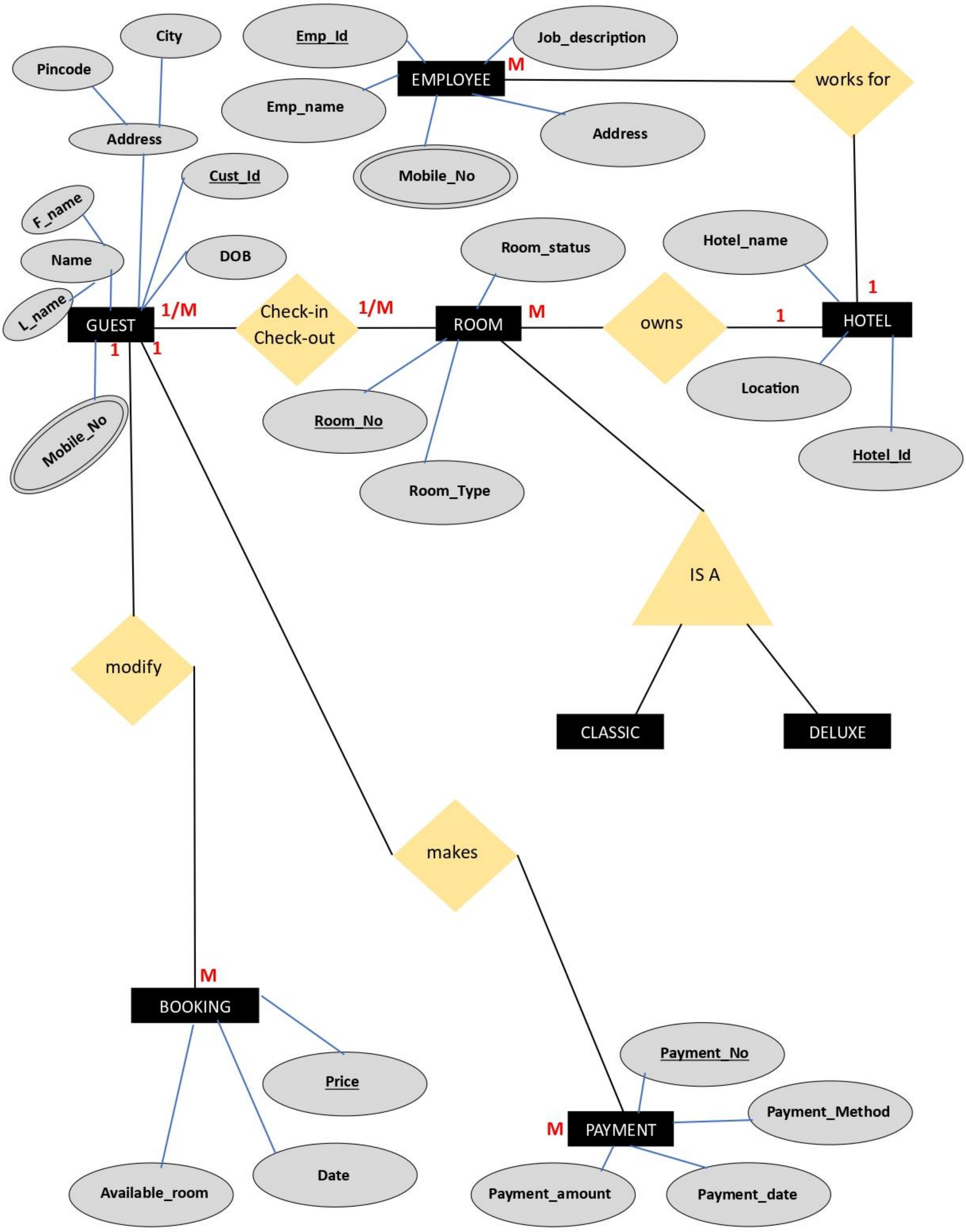
Entities and their attributes:

ENTITY	ATTRIBUTES	KEY
Employee	Emp_Id Emp_Name Job_description Address Mobile_No	Primary Key
Hotel	Location Hotel_id	Primary Key
Room	Room_No Room_type Room_Status	Primary Key
Guest	Name (F_name, L_name) Cust_id DOB Address (City, Pincode) Mobile_No	Primary Key
Booking	Date Available_Room Price	Primary Key
Payment	Payment_no Payment_method Payment_date Payment_amount	Primary Key

Relationship between entities:

ENTITY	RELATIONSHIP	ENTITY	CARDINALITY
Employee	works for	Hotel	Many to One
Hotel	owns	Room	One to Many
Guest	check-in/check-out	Room	Many to One
Guest	modify	Booking	One to Many
Guest	makes	Payment	One to Many

Entity-Relationship diagram:



Question 2: From the following problem statement identify the possible entity sets, their attributes, and relationships. SE VLabs Inc. is a young company with a few departments spread across the country. As of now, the company has a strength of 200+ employees. Each employee works in a department. While joining, a person has to provide a lot of personal and professional details including name, address, phone #, mail address, date of birth, and so on. Once all these information are furnished, a unique ID is generated for each employee. He is then assigned a department in which he will work. There are around ten departments in the company. Unfortunately, two departments were given same names. However, departments too have ID's, which are unique.

Answer 2:

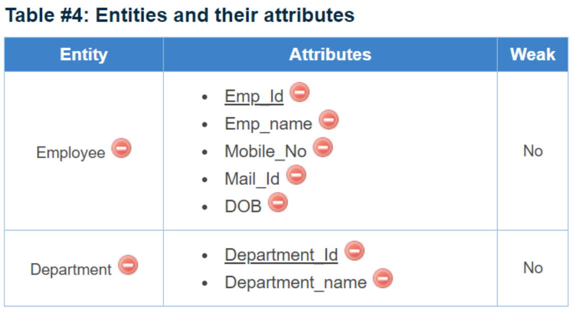


Table #4: Entities and their attributes

Entity	Attributes	Weak
Employee ⊖	• Emp_Id ⊖ • Emp_name ⊖ • Mobile_No ⊖ • Mail_Id ⊖ • DOB ⊖	No
Department ⊖	• Department_Id ⊖ • Department_name ⊖	No

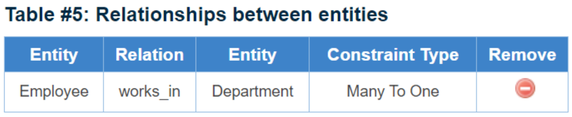
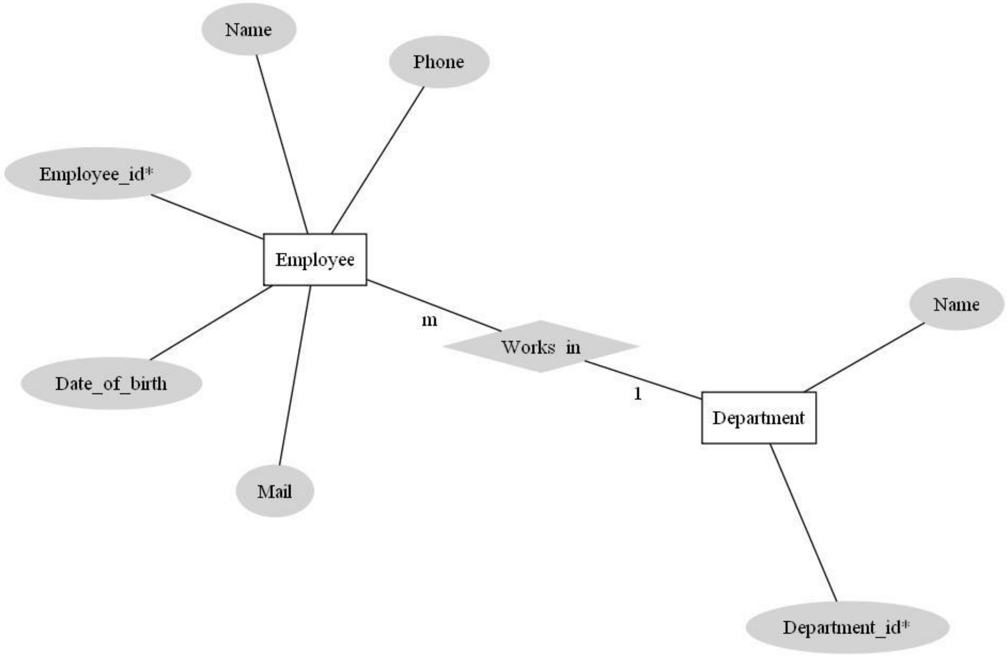


Table #5: Relationships between entities

Entity	Relation	Entity	Constraint Type	Remove
Employee	works_in	Department	Many To One	⊖

Entity-Relationship Diagram:



MODELING UML USE CASE DIAGRAMS

- Use case diagrams: Use case diagrams belong to the category of behavioral diagram of UML diagrams. Use case diagrams aim to present a graphical overview of the functionality provided by the system. It consists of a set of actions (referred to as use cases) that the concerned system can perform, one or more actors, and dependencies among them.
- Actor: An actor can be defined as an object or set of objects, external to the system, which interacts with the system to get some meaningful work done. Actors could be human, devices, or even other systems. Actors can be classified as below:
  - Primary actor: They are principal users of the system, who fulfill their goal by availing some service from the system.
  - Supporting actor: They render some kind of service to the system.

  In a use case diagram primary actors are usually drawn on the top left side of the diagram.

- Use Case: A use case is simply a functionality provided by a system. Use cases include both successful and unsuccessful scenarios of user interactions with the system.
- Subject: Subject is simply the system under consideration. Use cases apply to a subject.
- Graphical Representation: An actor is represented by a stick figure and name of the actor is written below it. A use case is depicted by an ellipse and name of the use case is written inside it. The subject is shown by drawing a rectangle. Label for the system could be put inside it. Use cases are drawn inside the rectangle, and actors are drawn outside the rectangle.
- Association between Actors and Use Cases: A use case is triggered by an actor. Actors and use cases are connected through binary associations indicating that the two communicates through message passing. An actor must be associated with at least one-use case. Similarly, a given use case must be associated with at least one actor. Association among the actors are usually not shown. However, one can depict the class hierarchy among actors.
- Three types of relationships exist among use cases:
  - Include Relationship: Include relationships are used to depict common behavior that are shared by multiple use cases. This could be considered analogous to writing functions in a program in order to avoid repetition of writing the same code. Such a function would be called from different points within the program. Include relationship is depicted by a dashed arrow with a «include» stereotype from the including use case to the included use case.
  - Extend Relationship: Use case extensions are used to depict any variation to an existing use case. They are used to the specify the changes required when any assumption made by the existing use case becomes false. Extend relationship is depicted by a dashed arrow with a «extend» stereotype from the extending use case to the extended use case.
  - Generalization Relationship: Generalization relationship is used to represent the inheritance between use cases. A derived use case specializes some functionality it has already inherited from the base use case. Generalization relationship is depicted by a solid arrow from the specialized (derived) use case to the more generalized (base) use case.

Question 1: Draw a use case diagram for your project.

Answer 1: Hotel Management System

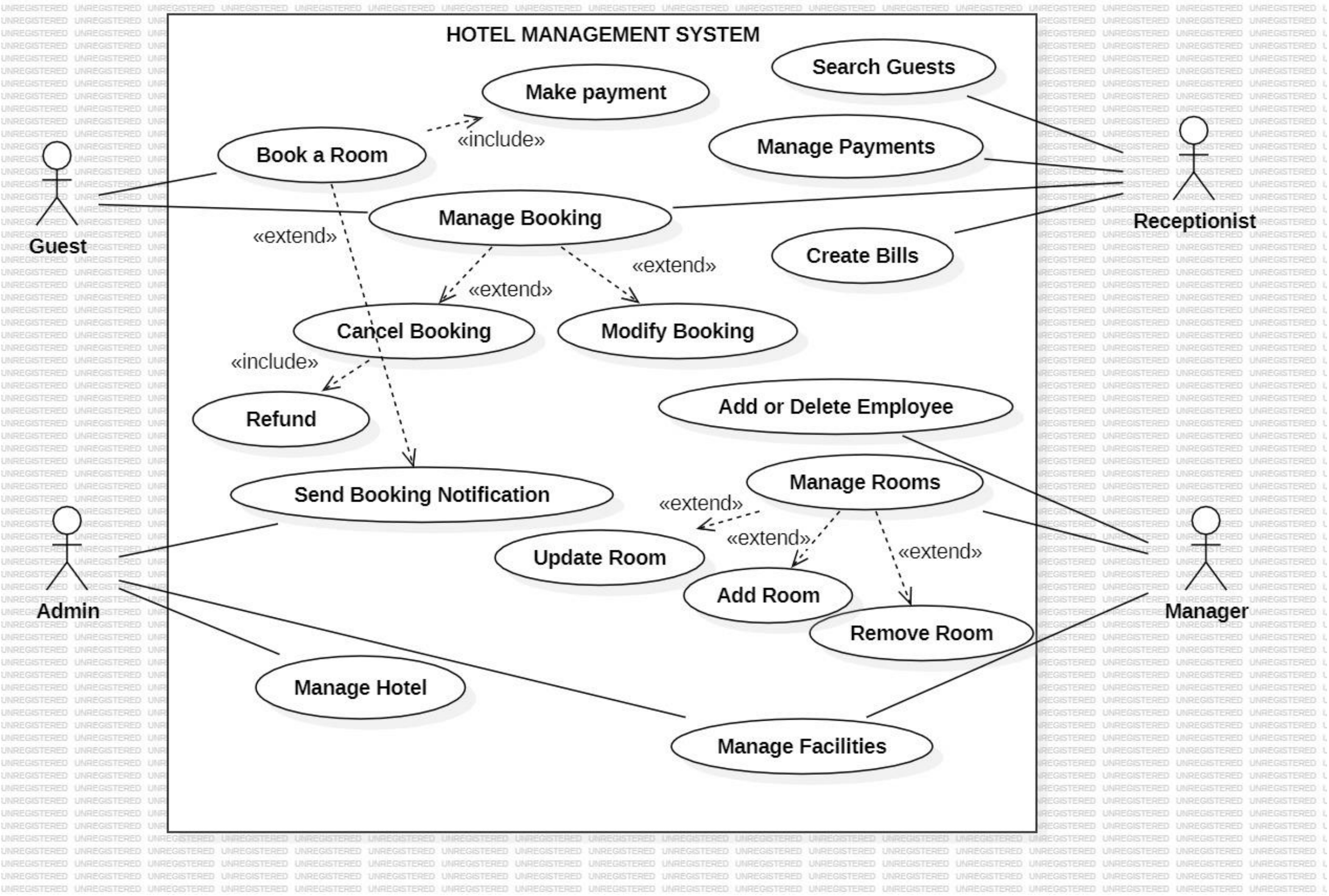
Actors and their use cases:

ACTORS	USE CASES
Guest Manager Receptionist Admin	Book a room Make a payment Manage Booking Cancel Booking Modify Booking Refund Search Guests Manage Payments Create Bills Add or delete employee Manage rooms Add room Remove room Update room Send booking notification Manage hotel Manage facilities

List of Relationships:

ACTOR / USE CASE	RELATIONSHIP	ACTOR / USE CASE
Guest	Association	Book a room Manage booking
Manage booking	Extend	Cancel booking Modify booking
Book a room	Include Extend	Make payment Send booking notification
Cancel Booking	Include	Refund
Receptionist	Association	Search guests Manage payments Create bills Manage booking
Admin	Association	Send booking notification Manage facilities Manage hotel
Manager	Association	Add or delete employee Manage rooms Manage facilities
Manage rooms	Extend	Update room Add room Remove room

Use Case Diagram:



Question 2: Consider a library, where a member can perform two operations: issue book and return it. A book is issued to a member only after verifying his credentials. Draw a use case diagram for the problem.

Answer 2:

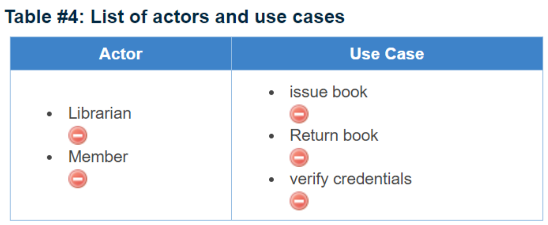


Table #4: List of actors and use cases

Actor	Use Case
- Librarian ⊖ - Member ⊖	- issue book ⊖ - Return book ⊖ - verify credentials ⊖

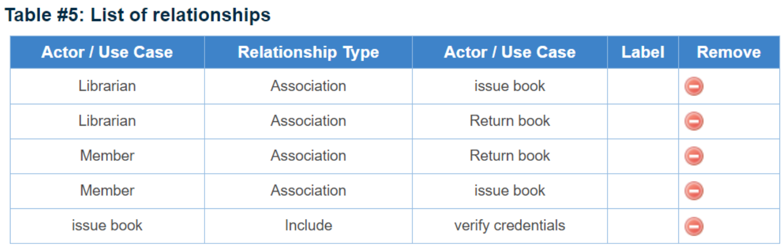
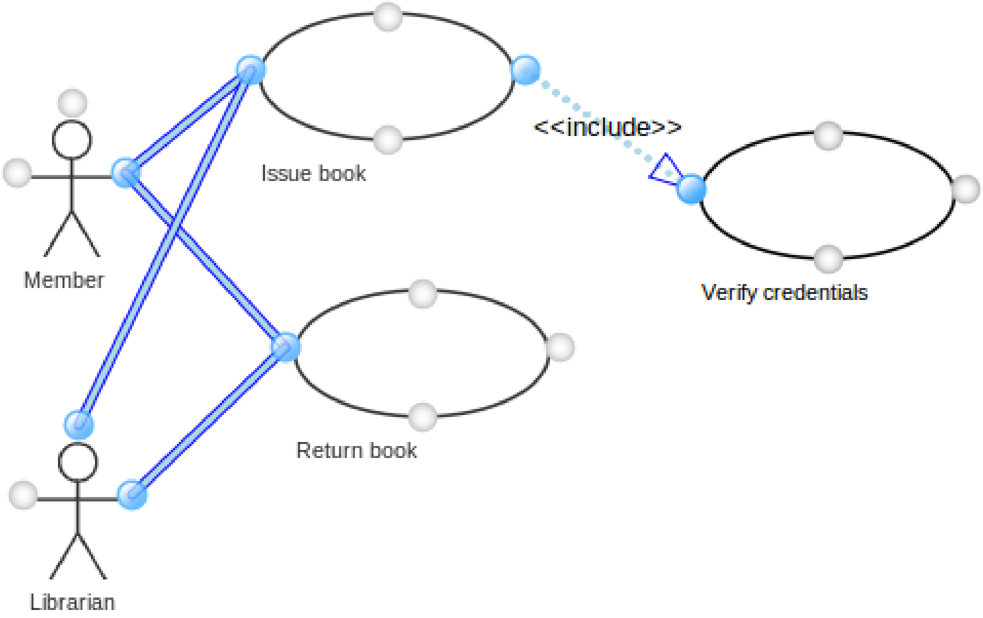


Table #5: List of relationships

Actor / Use Case	Relationship Type	Actor / Use Case	Label	Remove
Librarian	Association	issue book		⊖
Librarian	Association	Return book		⊖
Member	Association	Return book		⊖
Member	Association	issue book		⊖
issue book	Include	verify credentials		⊖

Use case diagram:



Issue book
<<include>>
Verify credentials
Member
Return book
Librarian