SOFTWARE ENGINEERING PROJECT

COMPUTER SCIENCE & ENGINEERING
(Artificial Intelligence & Machine Learning)

Submitted By

24WH1A6657 Ms. R. LAYA SREE
24WH1A6658 Ms. V. GAYATHRI
24WH1A6659 Ms. S. RISHIKA
24WH1A6660 Ms. E. KRITIKA
24WH1A6661 Ms. B. SAIPRIYA
24WH1A6662 Ms. N. YASHASWINI
24WH1A6663 Ms. K. SHRUTHI

under the esteemed guidance of
Ms. V. ASHA
Assistant Professor CSE (AI & ML)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
(Artificial Intelligence & Machine Learning)
BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN
(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)
Accredited by NAAC with A Grade
Bachupally, Hyderabad – 500090

BVRIT HYDERABAD
COLLEGE OF ENGINEERING FOR WOMEN

(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad) Accredited by NAAC with A Grade
Bachupally, Hyderabad – 500090

Department of Computer Science & Engineering
(Artificial Intelligence & Machine Learning)

CERTIFICATE

This is to certify that the **Software Engineering Project** is a bonafide work carried out by **Ms. R. LAYA SREE(24WH1A6657), Ms. V. GAYATHRI(24WH1A6658), Ms. S. RISHIKA (24WH1A6659), Ms. E. KRITIKA(24WH1A6660), Ms. B. SAIPRIYA(24WH1A6661), Ms. N. YASHASWINI (24WH1A6662),Ms. K. SHRUTHI (24WH1A6663)** in partial fulfilment for the award of B.Tech degree **in Computer Science & Engineering (AI & ML), BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad,** affiliated to Jawaharlal Nehru Technological University Hyderabad, under my guidance and supervision. The results embodied in the project work have not been submitted to any other**.**

Internal Guide
Ms. V .ASHA
Assistant Professor
Dept of CSE(AI&ML)

Head of the Department
Dr. B Lakshmi Praveena
HOD & Professor
Dept of CSE(AI&ML)

Structural Diagrams

1. **Class Diagram**: Shows the static structure of the system, including classes, their attributes, operations (or methods), and the relationships among objects.
2. **Object Diagram**: Represents a snapshot of instances of classes at a particular moment, showing how instances of the classes interact.
3. **Component Diagram**: Depicts how components are wired together to form larger components or software systems.
4. **Composite Structure Diagram**: Illustrates the internal structure of a class and the collaborations that this structure makes possible.
5. **Deployment Diagram**: Shows the physical deployment of artifacts on nodes, including the hardware and software components.
6. **Package Diagram**: Organizes elements of a system into related groups to minimize dependencies between packages.
7. **Profile Diagram**: Provides a way to extend UML for particular domains or platforms (e.g., for modeling business processes or system architectures).

Behavioral Diagrams

1. **Use Case Diagram**: Represents the functionality of a system from the user's perspective. It includes actors (users) and use cases (system functionalities).
2. **Activity Diagram**: Illustrates the dynamic aspects of a system, showing the flow from one activity to another.
3. **State Machine Diagram (or State Diagram)**: Describes the states of an object and transitions between these states.
4. **Sequence Diagram**: Details how objects interact in a particular sequence, focusing on the time sequence of messages.
5. **Communication Diagram (or Collaboration Diagram)**: Similar to sequence diagrams but focuses on the structural organization of objects that send and receive messages.
6. **Interaction Overview Diagram**: Combines elements of activity and sequence diagrams, showing an overview of the control flow.
7. **Timing Diagram**: Focuses on timing constraints of messages passed between objects.

These diagrams collectively provide a comprehensive view of a system from different perspectives, making them useful for both high-level planning and detailed design.

Hotel Management System

1. Problem Statement

A Hotel Management System (HMS) is designed to automate and manage all operations of a hotel efficiently. The system handles room bookings, check-ins, check-outs, guest details, staff management, and billing. Guests can book rooms in advance, modify or cancel reservations, and the system keeps track of room availability. Popular rooms or suites can have multiple bookings simultaneously. Old or unavailable rooms are updated automatically in the system. The system allows administrators to create, update, delete, and view information about rooms, guests, staff, bookings, and services easily.

The system improves operational efficiency, reduces manual errors, and provides real-time information to both management and guests.

2. Software Requirement Specification (SRS) Document

User Characteristics

- **Guest**: Individuals who want to book rooms, check availability, or request hotel services.
- **Receptionist/Administrator**: Has privileges to manage bookings, check-ins, check-outs, room allocation, and guest records.
- **Manager**: Can generate reports, manage staff, room inventory, and oversee overall hotel operations.

System Modules

1. **Login Module**
   Secure login for guests, receptionists, and managers via username and password.
2. **Room Management Module**
   - Add new rooms with details like room number, type, capacity, price, and amenities.
   - Update room information. ○ Delete rooms if out of service or removed.
   - Check room availability for booking.
3. **Booking Module** ○ Guests can book, modify, or cancel rooms.
   - Assign rooms to guests based on availability and preferences.
   - Handle advance bookings and walk-in guests.
4. **Check-in & Check-out Module** ○ Track guest arrival and departure.
   - Calculate billing based on stay duration and services availed.
5. **Billing & Payments Module** ○ Generate invoices.
   - Accept multiple payment modes (cash, card, online).
6. **Operations Module**
   Receptionists and administrators perform operations such as add booking, update room info, check-in/out, and view guest details.

Non-Functional Requirements

- **Privacy**: User credentials and guest data remain secure.
- **Portability**: The system can be installed on multiple platforms and devices.
- **Performance**: Quick response for booking, check-in, check-out operations.

3. Software Configuration Management (SCM) Requirements

- **Operating System**: Windows 7/10
- **Front End**: J2EE (Java EE)
- **Back End**: MySQL Server
- **IDE Used**: NetBeans

Hardware Requirements

Component	Requirement
Processor	i3 or higher
RAM	4 GB or more
Hard Disk	500 GB

4. Study and Usage of CASE Tool

CASE Tool: **StarUML**

Description:
StarUML is a UML modeling tool supporting UML 2.x standards. It allows creation of **Use Case, Class, Sequence, Collaboration, State Chart, Component, Deployment diagrams**, etc. It is suitable for designing systems like HMS and is widely used in educational and professional settings.

Features:

- Multi-platform support (Windows, macOS, Linux)
- ERD, DFD, Flowchart diagrams
- Modern UI with dark/light themes
- Code generation support
- Model-driven development support

Installation Steps:

1. Search "StarUML" in the browser and download the software.
2. Choose the correct OS version.
3. Run the downloaded file, accept permissions, and complete installation.
4. Close the license popup to start using StarUML.

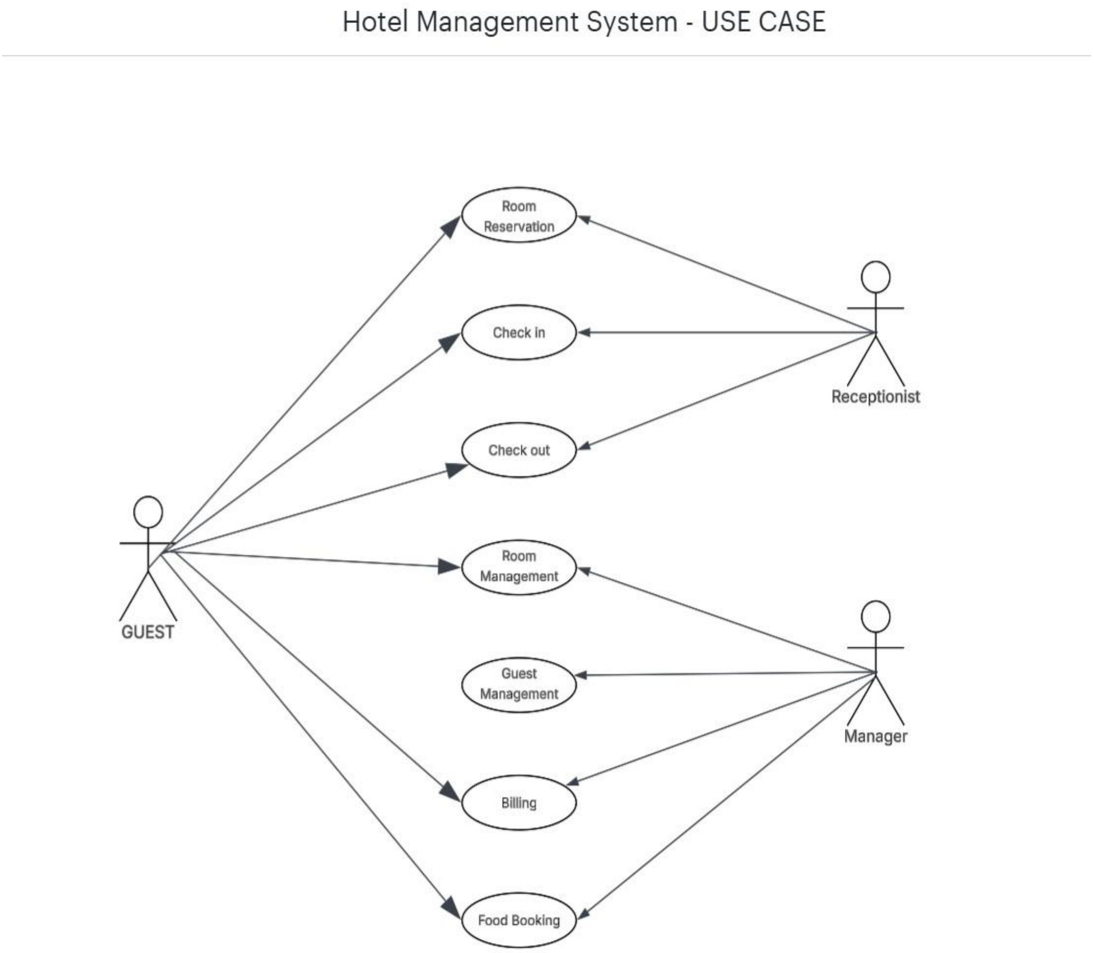
5. Design

Use Case Diagram

Use Cases for Hotel Management System:

1. Room Booking
2. Room Check-in
3. Room Check-out
4. Room Management
5. Guest Management
6. Billing **Actors**:

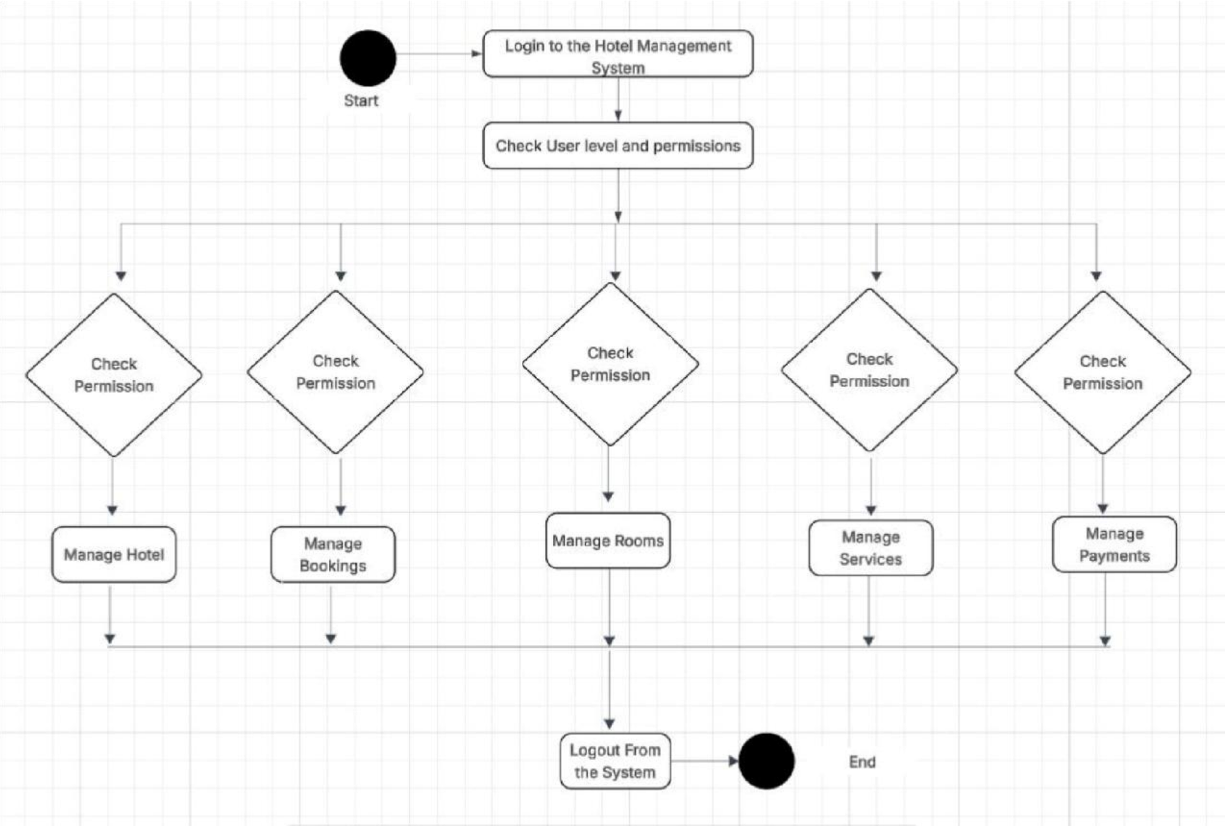
- Guest
- Receptionist
- Manager



Activity Diagram

1. **Booking a Room**:
   Guest selects dates → System checks availability → Booking confirmed → Payment processed → Confirmation sent

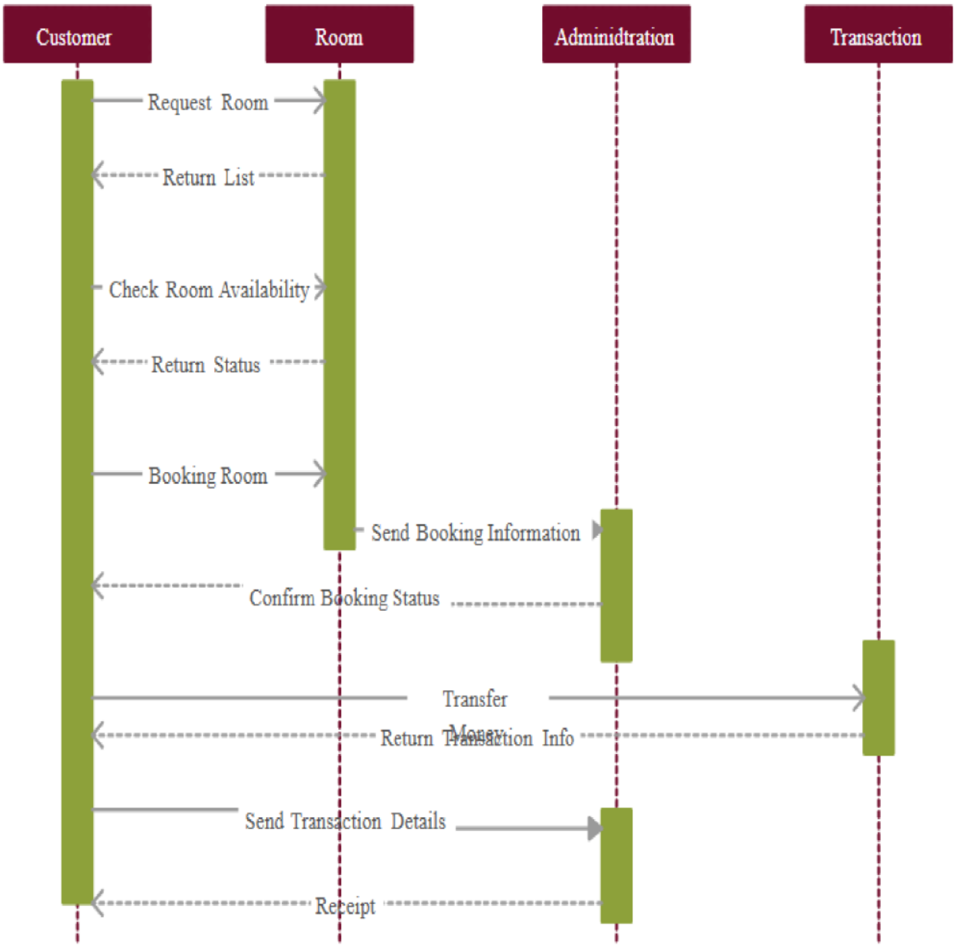
2. **Check-in Process**:
   Guest arrives → Validates booking → Allocates room → Updates occupancy status
3. **Check-out Process**:
   Guest requests check-out → Calculates billing → Processes payment → Updates room availability



Sequence Diagram

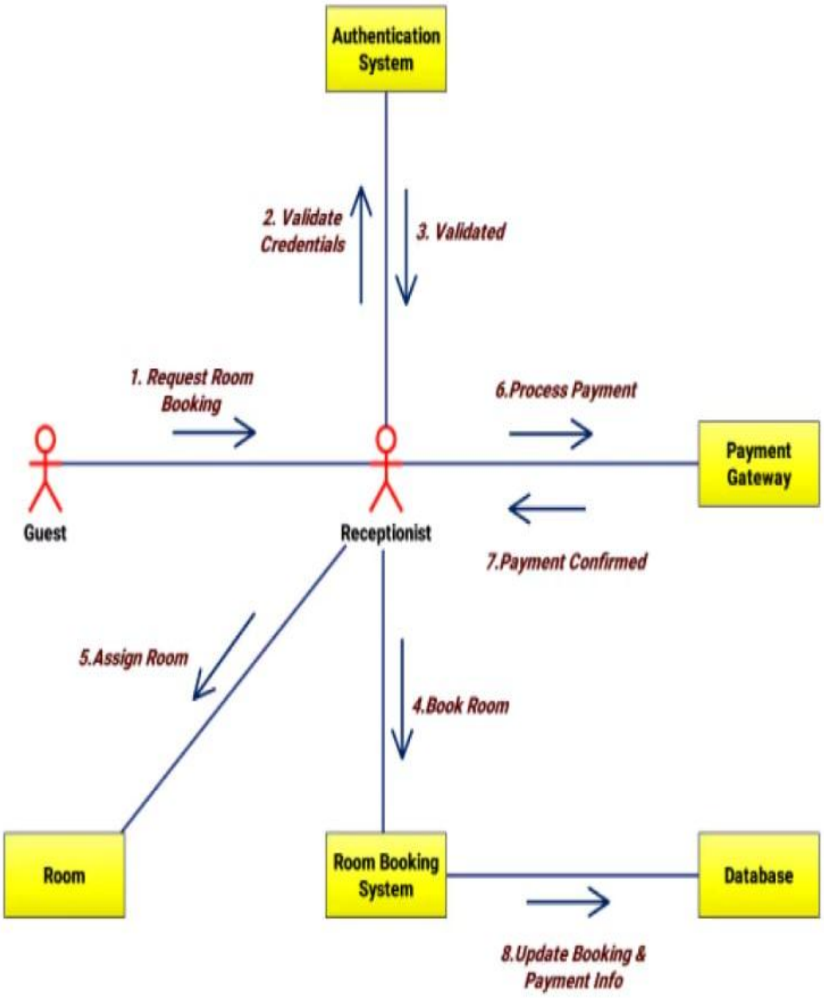
Room Booking Sequence:

Guest → System: Request Room Availability → System → Database: Check Room → System → Guest: Display Available Rooms → Guest → System: Confirm Booking → System → Database: Update Booking → System → Guest: Send Confirmation



Collaboration Diagram

Represents interaction among **Guest, Receptionist, System, Database** for booking, check-in, and billing. Messages are exchanged in order to complete operations.



Class Diagram

Main Classes:

1. **Guest** ○ Attributes: guestID, name, contact, email, address, bookingHistory ○ Operations: addGuestInfo(), modifyGuestInfo(), deleteGuestInfo()

2. **Room**
   - ○ Attributes: roomNo, type, capacity, price, amenities, status ○ Operations: addRoom(), updateRoom(), deleteRoom()

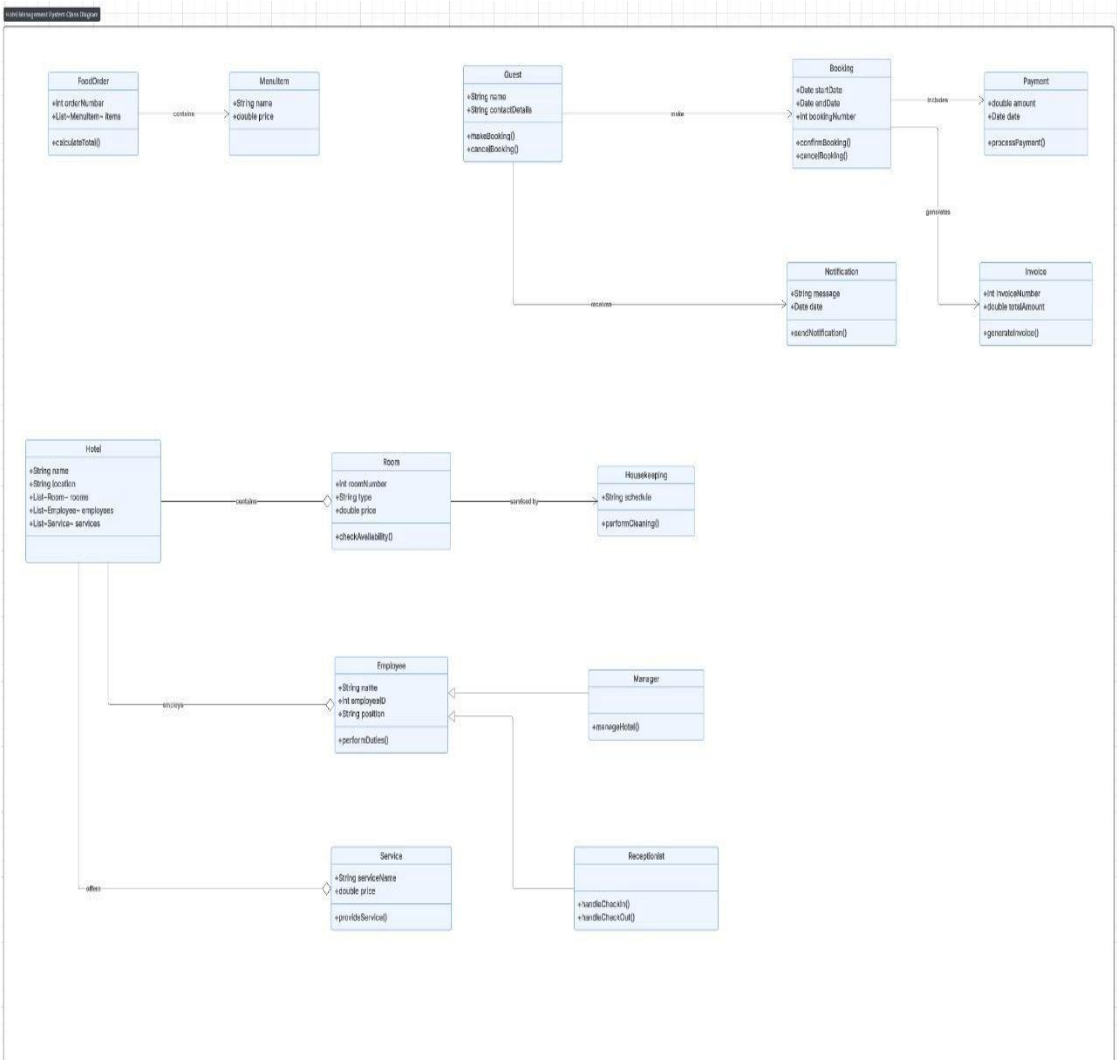
3. **Booking** ○ Attributes: boID, gokinguestID, roomNo, checkInDate, checkOutDate, status ○ Operations: createBooking(), cancelBooking(), updateBooking()

4. **Billing**

- Attributes: billID, bookingID, amount, paymentMode, paymentStatus ○ Operations: generateBill(), processPayment()

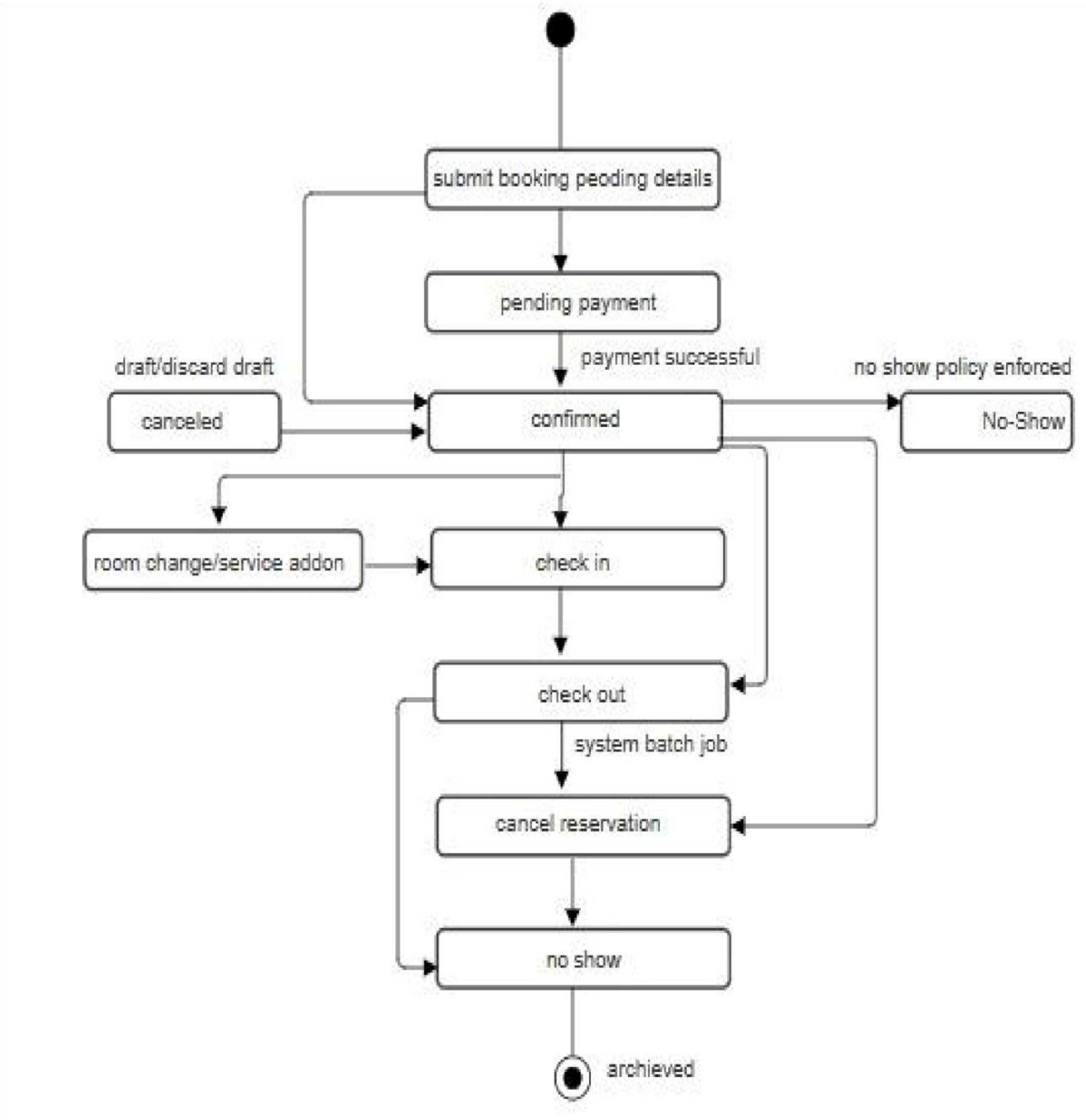
5. **Receptionist** ○ Attributes: employeeID, name, contact
   - Operations: manageBooking(), manageCheckInOut()

6. **Manager** ○ Attributes: employeeID, name, role ○ Operations: generateReports(), manageStaff()



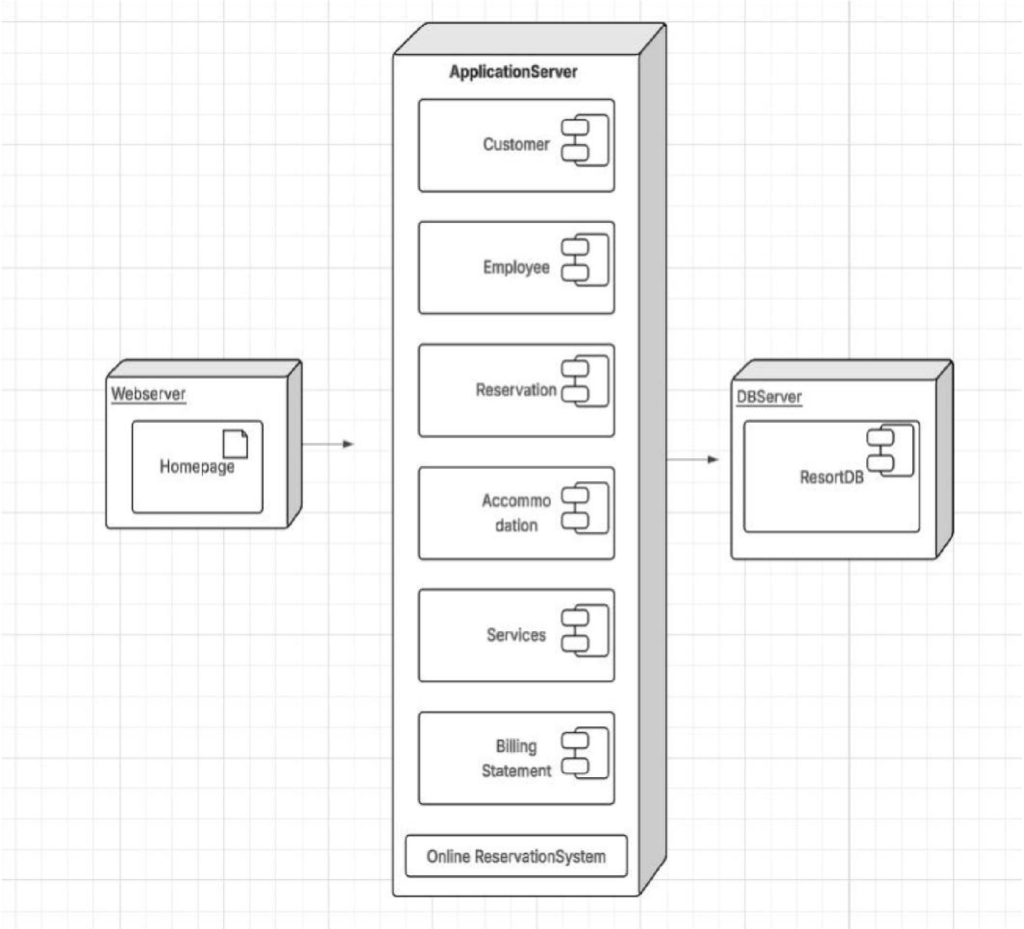
StateChart Diagram

- States of **Room**: Available → Booked → Occupied → Vacated → Maintenance → Available



Deployment & Component Diagram

- **Deployment**: Web-based application deployed on server with database backend.
- **Components**: Frontend (JSP/Servlets), Backend (Java), Database (MySQL), Payment Gateway API.


ApplicationServer
Customer
Employee
Reservation
Accommo dation
Services
Billing Statement
Online ReservationSystem
Webserver
Homepage
DBServer
ResortDB