Software Engineering Project

COMPUTER SCIENCE & ENGINEERING

(Artificial Intelligence & Machine Learning)

Submitted by

23WH1A6603 Ms. K. ANUSRI

23WH1A6604 Ms. N. MANASA MARY JYOTHI

23WH1A6618 Ms. V. CHANDU PRIYA

23WH1A6625 Ms. B. HARI PRIYA REDDY

23WH1A6626 Ms. N. CHARITHA

23WH1A6630 Ms. M. SHANMUKHI

23WH1A6648 Ms. A. SAHASRA SRI

23WH1A6650 Ms. S. MARY SRI KEERTHANA

23WH1A6661 Ms. S. SWETHA

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 a Bonafide work carried out by **Ms. K. ANUSRI (23WH1A6603), Ms. N. MANASA MARY JYOTHI (23WH1A6604), Ms. V. CHANDU PRIYA (23WH1A6618), Ms. B. HARI PRIYA REDDY (23WH1A6625), Ms. N.CHARITHA (23WH1A6626), Ms. M. SHANMUKHI (23WH1A6630), Ms. A. SAHASRA (23WH1A6648),Ms. S. MARY SRI KEERTHANA (23WH1A6650), Ms. S. SWETHA (23WH1A6661)** 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)

INDEX

Sl. No.	Title of the Project	Pages	Marks	Remarks
1.	Online Exam Registration	1 - 11		
2.	Hotel Management System	12 – 24		
3.	Hospital Management System	25 - 35		

ONLINE EXAM REGISTRATION

1.Problem Statement:

Online Exam Registration System Educational institutions often face challenges in managing the exam registration process, including manual data handling, verification, and scheduling. These processes are time-consuming, prone to errors, and inefficient when dealing with large volumes of students. The goal of the **Online Exam Registration System** is to streamline and automate the registration process, reducing manual work while ensuring accuracy and security.

The system will:

- Allow students to fill out an online registration form, verified against existing database records.
- Provide students with a list of available exam dates.
- Display fees and application status through an online interface.
- Forward verified information to the Exam Controller for manual review.
- Issue hall tickets once all criteria are met.

This system will enhance efficiency, reduce errors, and provide a user-friendly way for students to register for exams, ensuring seamless integration with institutional workflows.

2. Preparation of Software Requirement Specification Document:

Functional Requirements:

The increasing number of participating students has made the manual process of online exam registration both time-consuming and prone to errors. To address these challenges, an Automated Exam Registration System has been developed using advanced programming and database techniques. This system incorporates rigorous verification and validation procedures to ensure reliability and compliance with required standards. With an intuitive online interface, it allows for seamless input of personal details and convenient document submission through scanning. This user-friendly approach not only simplifies the registration experience for applicants but also significantly reduces administrative workload and speeds up application processing. By embracing modern technological trends, this implementation aims to enhance the efficiency and effectiveness of the exam registration process, meeting the evolving needs of students and administrative staff alike.

3. Preparation of Software Configuration Management

Software Requirements:

The following are the list of software requirements which we are using to implement this application.

Client-Side Technologies: HTML, CSS
Scripting Language: JavaScript
Business Logic Development Language: JS
Database Connectivity: JDBC
Database: MYSQL
Operating System: Ubuntu
Documentation: MS-Office

Hardware Requirements:

The following are the hardware requirements with minimum configuration to get better performance of our application.

Database Connectivity: JDBC
Database: MYSQL
Operating System: Ubuntu
Documentation: MS-Office

Deployment Requirements:

Front end: Java 1.8
Technologies: JSP and JDBC
Database: MYSQL server
Web Server: Apache Tomcat 8.5

4. Study and Usage of Design Phase CASE Tool:

StarUML is a software engineering tool that uses the Unified Modelling Language (UML) to model systems. It's available on Windows, Linux, and MacOS, and is published by MKLabs.

StarUML is designed for agile and concise modeling, and is used by small and agile development teams, educational institutions, and professionals. It supports multiple modeling languages, including flowchart and mind map.

Features of STAR UML:

- **Cross-platform:** Works on multiple platforms with the same UX
- **Auto update:** Checks for updates and installs them automatically for MacOS and Windows
- **Customizable:** Offers variables that can be customized to the user's software development methodology, project platform, and language
- **Extensible:** Has high extensibility in its functionality

Types of Diagrams in UML:

- Use case diagram
- Activity diagram
- Sequence diagram
- Collaboration diagram
- Class diagram
- State chart diagram
- Component diagram

5. Performing the design by using any Design phase.

1) Use Case Diagram: A use case diagram is a visual representation of the interactions between users (actors) and the system. It helps identify the functionalities of the system and the roles that users play in relation to those functionalities.

Actors: Actors represent entities that interact with the system. In the context of the Exam Registration system:

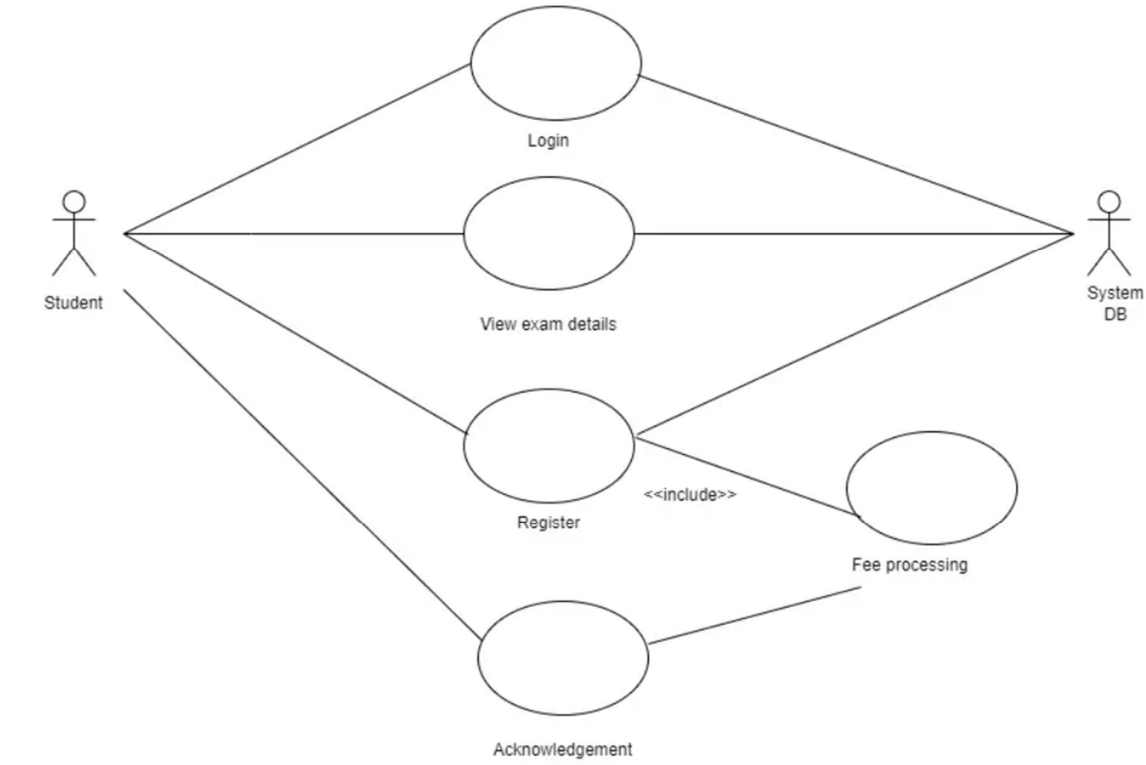
- **Student:** The primary actor who engages with the system to register for exams, view details, and manage fees.
- **Admin (optional):** An actor that may manage exam registrations and monitor student activities.

Use Cases: Use cases describe specific functionalities or actions that the system performs in response to an actor's input. The main use cases for the Exam Registration system are:

- **Login**: Allows students to access their accounts securely**.**

- **View Exam Details:** Enables students to see information about available exams, including schedules and subjects.
- **View Fees Details**: Provides students with information regarding the fee structure for the exams they intend to take.
- **Pay Fee:** Facilitates the payment process for exam fees, which may involve different payment methods.
- **Display Details:** Shows confirmation or additional information after actions like fee payment or exam registration**.**
- **Logout:** Allows students to securely exit the system after completing their tasks.

Conclusion: The use case diagram for the Exam Registration system provides a high-level overview of how students interact with the system to perform essential tasks like logging in, viewing exam and fee details, making payments, and logging out. serves as a foundational element in understanding the system's functionality and guiding its development.

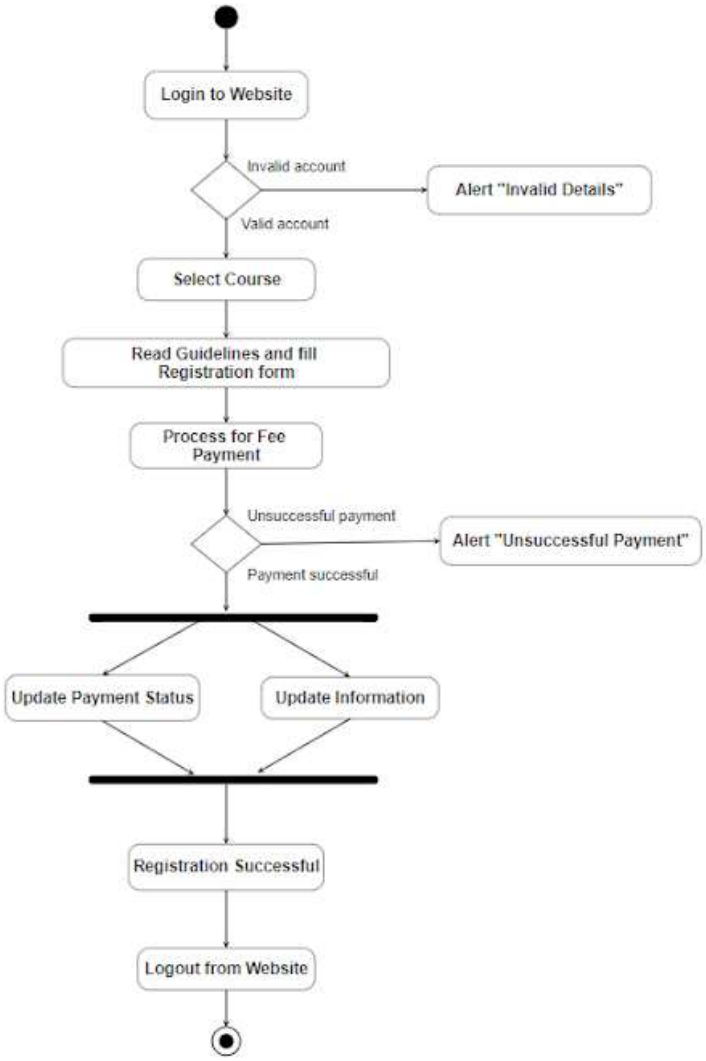


Use case diagram for Online Exam Registration.

2) Activity Diagram:

An activity diagram is a type of Unified Modeling Language (UML) flowchart that shows the flow from one activity to another in a system or process. It's used to describe the different dynamic aspects of a system and is referred to as a 'behavior diagram' because it describes what should happen in the modelled system.

Activity diagrams show the process from the start (the initial state) to the end (the final state). Each activity diagram includes an action, decision node, control flows, start node, and end node.



3) Class Diagram:

The class diagram, also referred to as object modeling is the main static analysis diagram. The main task of object modeling is to graphically show what each object will do in the problem domain. The problem domain describes the structure and the relationships among objects.

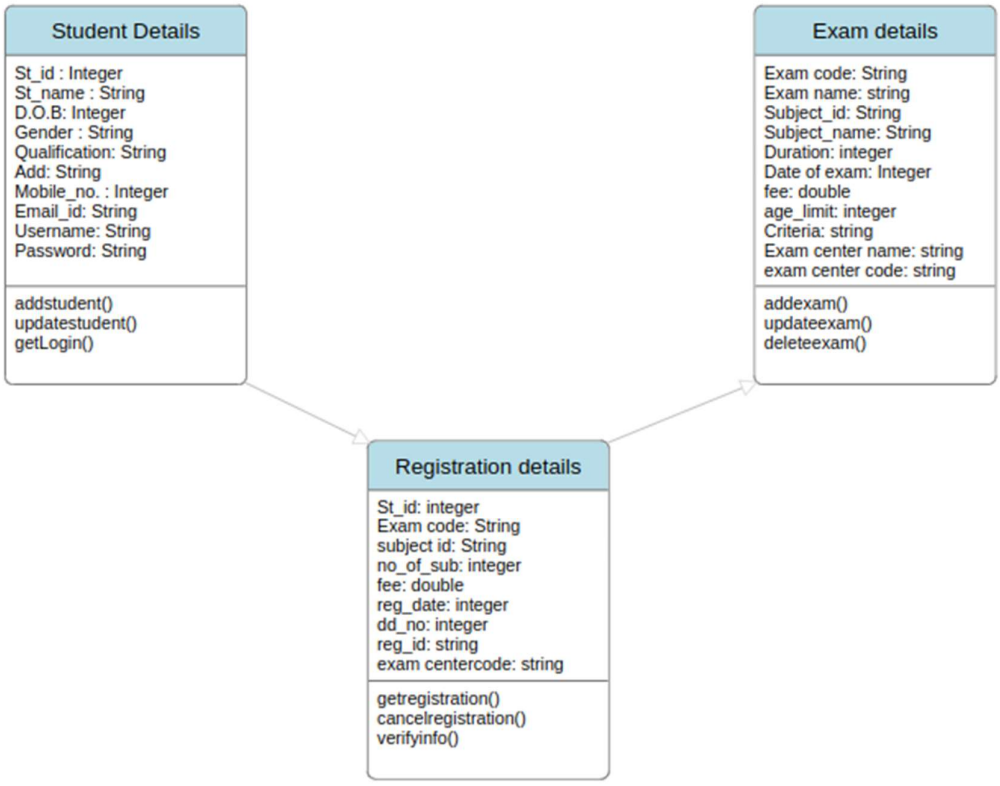
The Exam Registration System class diagram consists of four two classes of registration system.

1. Student_details
2. Exam_details
3. Register

1) STUDENT_DETAILS
It consists of six attributes and six operations. The attributes id, password, name, age, sex, course. The operations of this class are login (), logout (), conformation (), register (), newfeesdetails ().

2) EXAM_DETAILS
It consists of four attributes and six methods. The attributes are use rid, password,
exam fees, fees due. The methods are login (), logout (), fees details (), display fees (), conformation (), exam controller ().

3) REGISTER
This class is used to maintain the registered student information such as, subject registered, date of registration and etc.

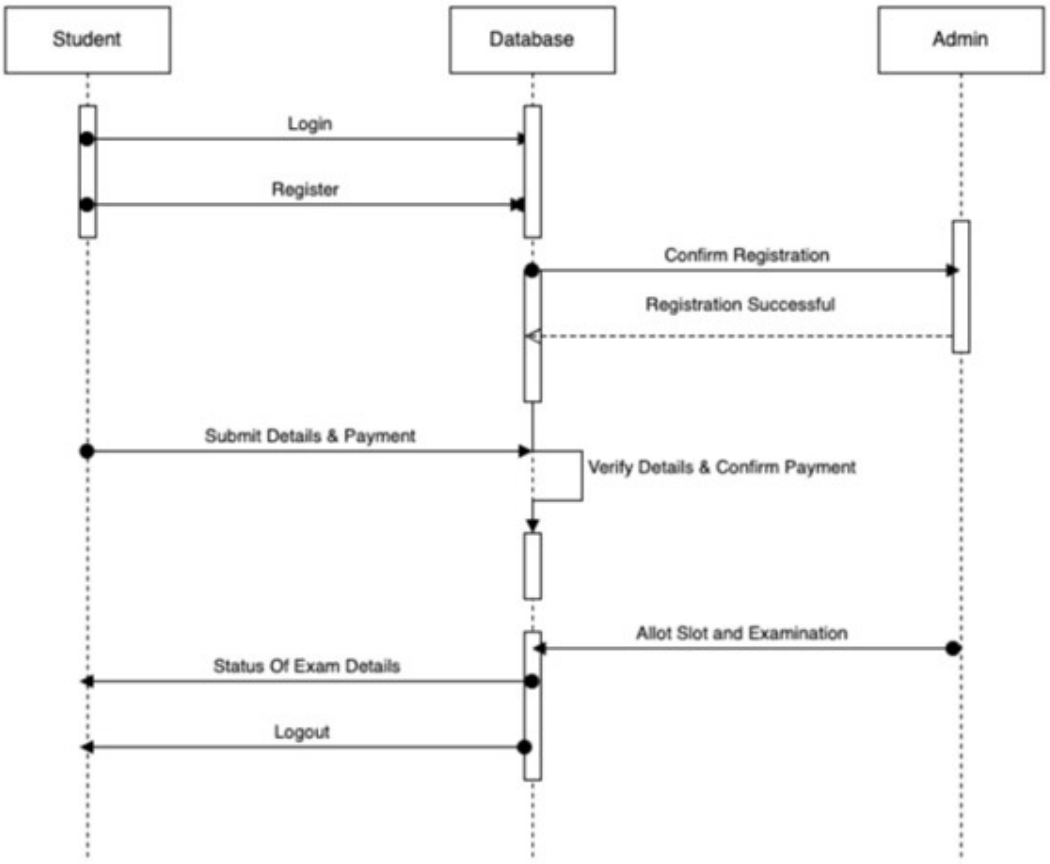


Class Diagram for Online Exam Registration.

4) Sequence Diagram:

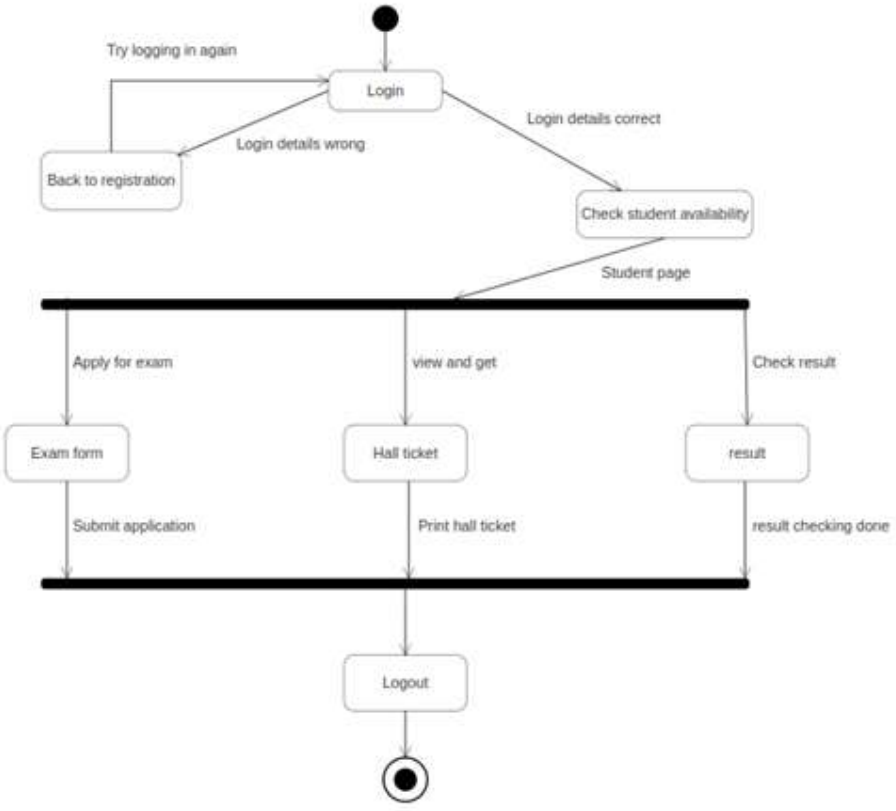
A sequence diagram is a diagram that shows the sequence of messages exchanged between objects in a system. It consists of a group of objects that are represented by lifelines, and the messages that they exchange over time during the interaction. Here, the lifelines or the main objects considered are the students, the database, and the admin or the controller.

This diagram shows the sequential order of registration, followed by payment and then the exam details that are to be provided to the student.



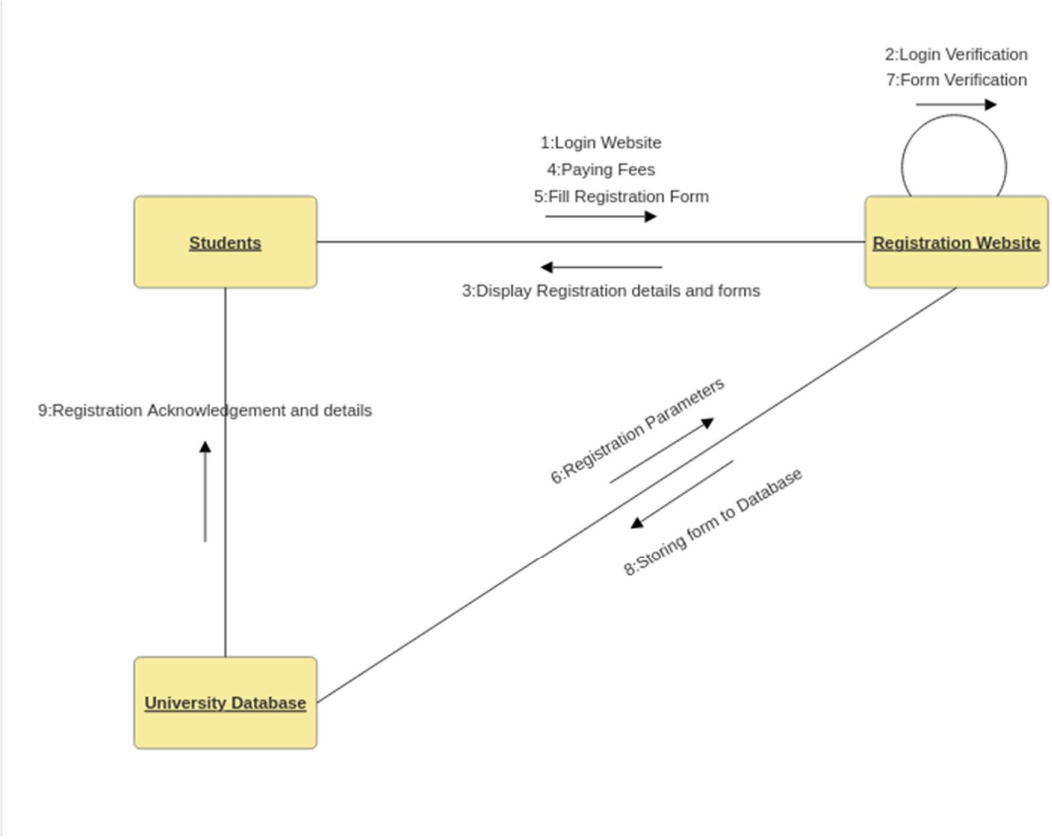
5) State Chart Diagram:

A state chart diagram is a modeling tool that is most often applied in software development to describe the behaviour of a system or an object through time detailing all the possible states occupied and the changes occurring in those states as a result of events. The most important and basic constituents are states, transitions, events, initial and final states, and actions. These diagrams are useful for modeling life cycles, for communication among the parties involved in a project, and for documentation of the system and its components. Thus, these diagrams are very helpful in situations which include but are not limited to user interfaces, embedded systems, and game designs**.**



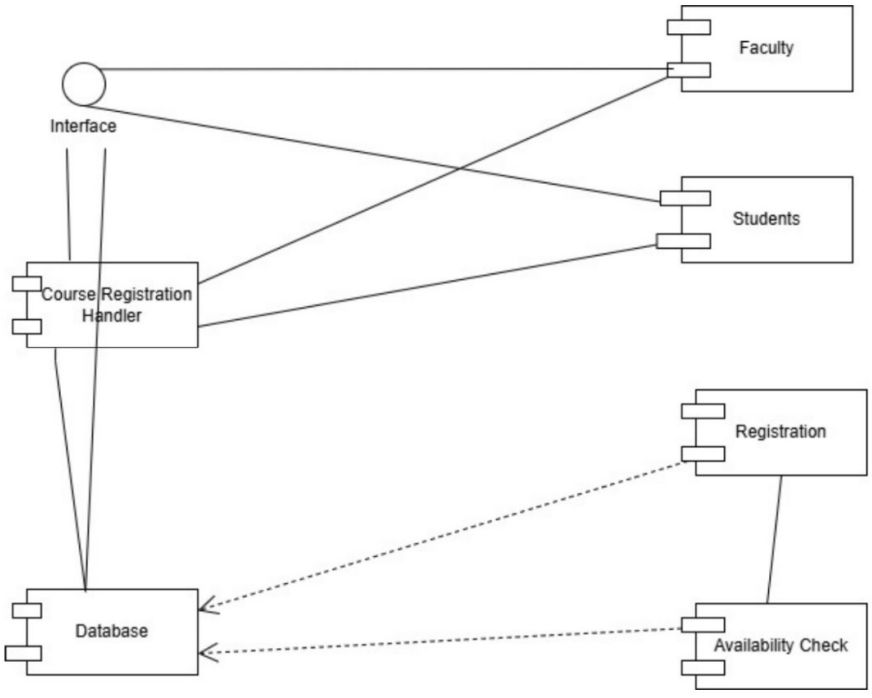
6) Collaboration Diagram:

This collaboration diagram for online exam registration illustrates the interaction between the Student, Registration Website, and University Database. The student begins by logging into the website, where their login is verified. After reviewing registration details, they pay the required fees and fill out the registration form. The form data, along with registration parameters, is sent to the university database for storage. Once verified, the website confirms the registration, and the database sends an acknowledgment and registration details back to the student. This process ensures secure and efficient registration management.



7) Component Diagram:

A component diagram for an online exam registration system visually represents its key components and their interactions. It includes the User Interface (both web and mobile applications), which communicates with the Application Server that manages business logic and processes user requests. The Application Server interacts with the User and Exam Databases to store profiles and exam information. Additionally, it connects to the Authentication Service for user management, the Notification Service for sending updates, and the Payment Gateway for processing registration fees. This diagram clarifies the system's structure, facilitates independent development of components, and supports scalability for future enhancements.



HOTEL MANAGEMENT SYSTEM

Problem statement: The Hotel Management System is a web-based application designed to streamline hotel operations for managers through an interactive GUI, enabling efficient handling of room bookings, staff management, and various hotel activities. Tailored for busy managers, this system centralizes management tasks, eliminating the need for manual, paper-based processes. Managers can post available rooms, and customers can easily view and book them online. An admin feature allows for booking approvals, ensuring controlled and organized reservations. Additionally, customers can access and book other hotel services, making the system convenient for both managers and customers by providing an all-in-one platform to efficiently manage hotel activities.

Features:

- **Admin login & admin dashboard:** It has admin login who has the authority of the system & he is responsible for approving & disapproving the users request for room booking. Admin can add & delete notifications & updates in the system.
- **User registration:** There is user registration form available where new users can create their account by providing required information to the system.
- **Booking system:** User can request for the table booking for a particular date & time.
- **Approving/Disapproving request:** The booking requests are directly sent to admin account by the system. Admin can view all the requests along with respective user details & therefore make decisions for cancelling the requests.

Software Requirements Specification:

- **Functional requirements:**
  These describe what the system should do, i.e., the core functionalities needed to manage the operations of a hotel.

1. User Management:
   - The system should allow administrators to create, update, and delete user profiles for different roles (e.g., admin, staff, guest).
   - The system should allow guests to create their own accounts and manage personal information.
2. Room Management:
   - The system should allow the hotel to manage room availability, types, and prices.
   - The system should display available rooms based on guest search criteria (e.g., room type, number of guests, check-in/check-out dates).
   - The system should allow staff to update room status (e.g., clean, dirty, maintenance).
3. Reservation Management:
   - Guests should be able to make room reservations online or at the hotel front desk.
   - The system should validate room availability during booking and prevent overbooking.
   - Guests should receive confirmation of their reservation via email or SMS.
   - The system should allow cancellations and modifications to bookings within specified policies.
4. Check-in and Check-out:
   - The system should support check-in and check-out processes, including issuing room keys (physical or digital).
   - The system should track check-in and check-out times.
   - The system should automatically calculate the total bill based on stay duration, room type, and any additional services.
5. Billing and Payments:
   - The system should support generating invoices based on room charges, services used (e.g., minibar, room service), and taxes.

- The system should allow guests to pay using various methods (e.g., credit/debit cards, cash, online payments).
- The system should allow guests to view a detailed breakdown of charges at check-out.

6. Inventory and Housekeeping Management:
   - The system should track inventory levels for consumables (e.g., linens, toiletries).
   - The system should allow housekeeping staff to track room cleaning schedules and requirements.
   - The system should notify staff if supplies are running low.
7. Customer Feedback and Complaints:
   - The system should allow guests to provide feedback or lodge complaints about their stay.
   - The system should track customer feedback and allow staff to respond or resolve complaints.
8. Reporting:
   - The system should provide reports on occupancy rates, revenue, guest demographics, and room usage.
   - It should allow administrators to generate financial reports, including income, expenses, and outstanding payments.
9. Multi-language Support:
   - The system should support multiple languages for international guests.

- **Non-functional requirements:**

These describe the qualities the system must exhibit, ensuring it performs well under various conditions and is maintainable.

1. Performance:
   - The system should handle up to [X] simultaneous users without significant degradation in performance.
   - The system should process room reservations and check-ins within [X] seconds to ensure quick service.
   - The system should be able to generate reports within [X] seconds/minutes.

2. Scalability:
    - The system should be able to scale horizontally to accommodate increasing user load as the hotel chain grows (e.g., adding more locations or users).
3. Availability:
    - The system should be available 24/7 with an uptime of [99.9%] or better.
    - Maintenance windows should be scheduled and communicated in advance.
4. Security:
    - The system should comply with data protection regulations (e.g., GDPR, CCPA) to safeguard guest information.
    - The system should implement encryption for sensitive data (e.g., payment information, personal guest details).
    - Access control should be implemented to ensure that only authorized personnel can access specific data or functions (e.g., admin access, financial reports).
    - The system should support two-factor authentication (2FA) for sensitive user accounts.
5. Usability:
    - The system should have an intuitive and user-friendly interface, making it easy for staff and guests to use.
    - The system should support mobile devices and provide a responsive design for both hotel staff and guests.
6. Compatibility:
    - The system should be compatible with different browsers (e.g., Chrome, Firefox, Safari) and mobile operating systems (iOS, Android).
    - It should integrate with third-party tools (e.g., payment gateways, CRM systems, channel managers).

7. Maintainability:
    - The system should be easy to update and maintain, with clear documentation for both users and developers.
    - It should support modular development to add new features with minimal disruption.
8. Backup and Recovery:
    - The system should have an automated backup mechanism to ensure data is regularly backed up.
    - In the event of a failure, the system should have a disaster recovery plan to restore data within [X] hours.
9. Compliance:
    - The system should meet all local and international regulations for handling guest data, payments, and business operations.

Software requirements:

- Windows XP, Windows 7(ultimate, enterprise)
- Visual Studio 2010
- SQL 08

Hardware components:

- Process - p4
- Hard disk - 5GB
- Memory - 1GB RAM

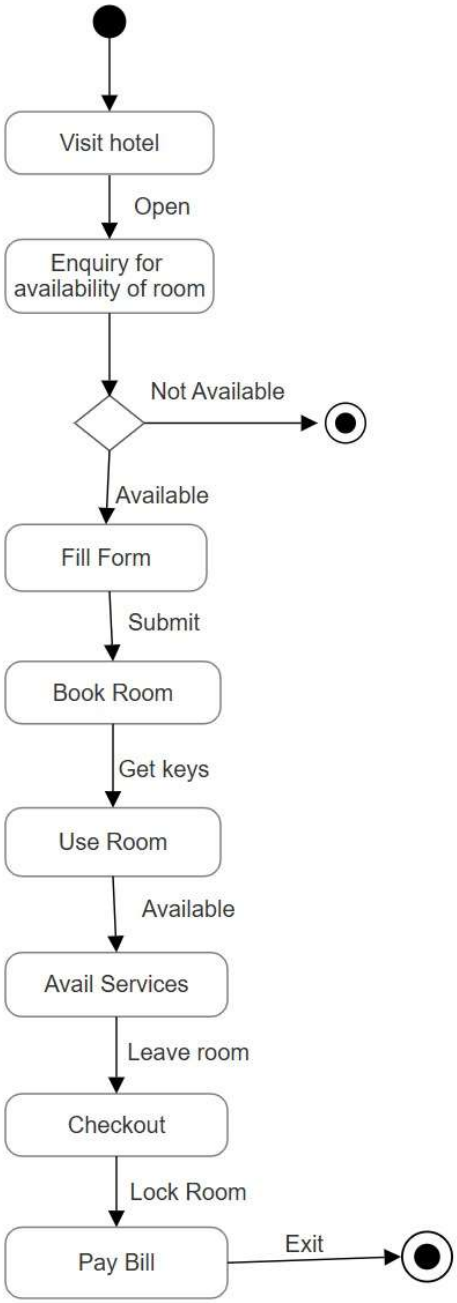
UML Diagrams:

Use-case diagram:

A use case diagram is a tool to visualize system requirements by showing how actors interact with the system. In the Hotel Management System, there are two main actors: the Receptionist and the Administrator. The Receptionist handles tasks like guest registration, booking rooms and halls, managing food orders, processing billing, and generating reports. The Administrator has control over all functions, including monitoring and adjusting operations. The diagram uses stick figures to represent actors, connecting them to system functions to clarify user interactions and system requirements.

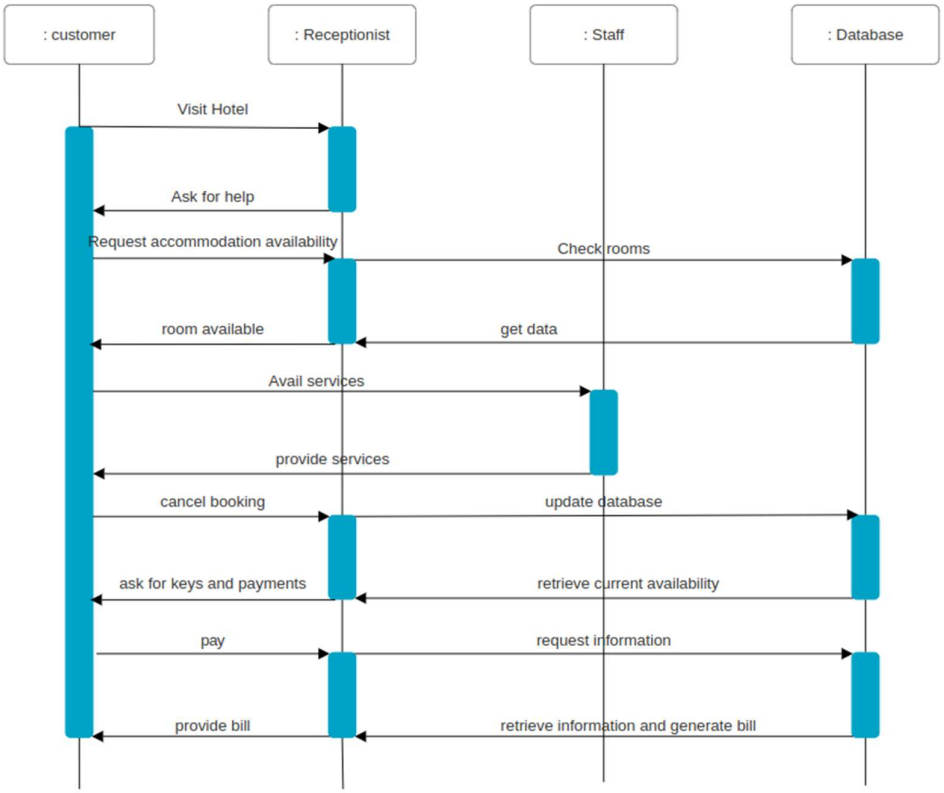
Activity diagram:

An activity diagram shows the flow of actions in a system. In the Hotel Management System, the Receptionist registers guests, checks availability, books room, manages orders, and processes billing. Decisions like availability or payment affect the next steps. The Administrator oversees and manages these tasks, ensuring smooth operations. The diagram uses actions and decision points to represent the workflow.



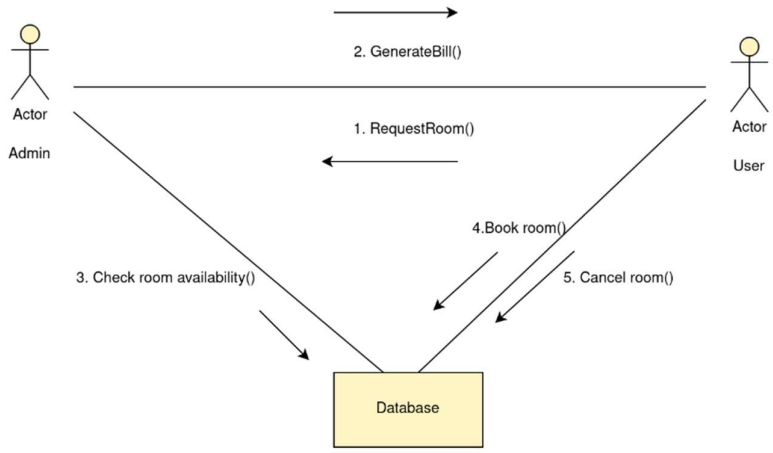
Sequence diagram:

A sequence diagram shows how actors interact with the system over time. It displays the order of messages exchanged between actors and the system to perform tasks such as booking rooms or processing payments. The diagram focuses on the sequence of actions, helping visualize the flow of operations in the system.



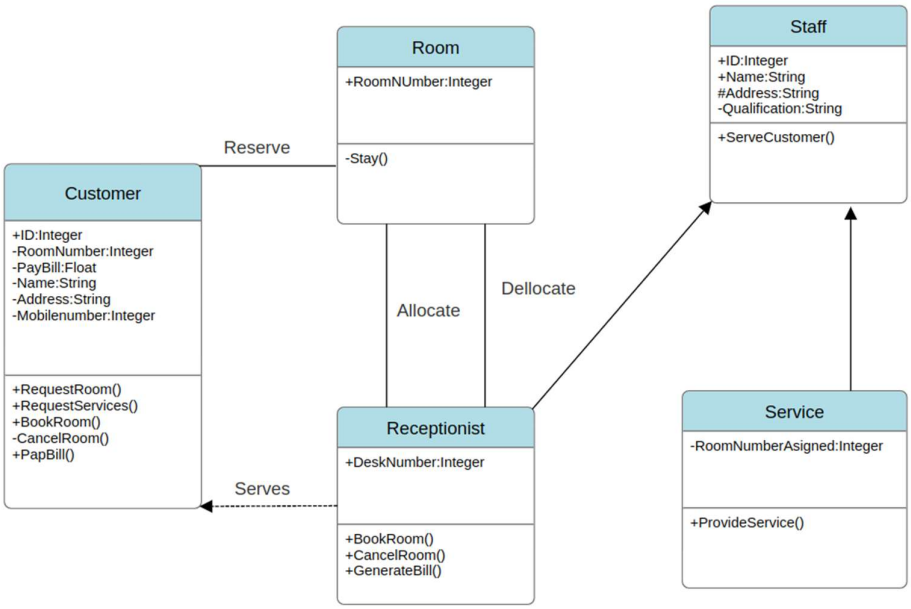
Collaboration diagram:

A collaboration diagram shows how actors and objects interact within a system. It focuses on the relationships between components, highlighting the messages exchanged to complete tasks. In the Hotel Management System, the diagram illustrates how the Receptionist and Administrator interact with different system objects to perform tasks, emphasizing the structure and flow of communication between them.



Class diagram:

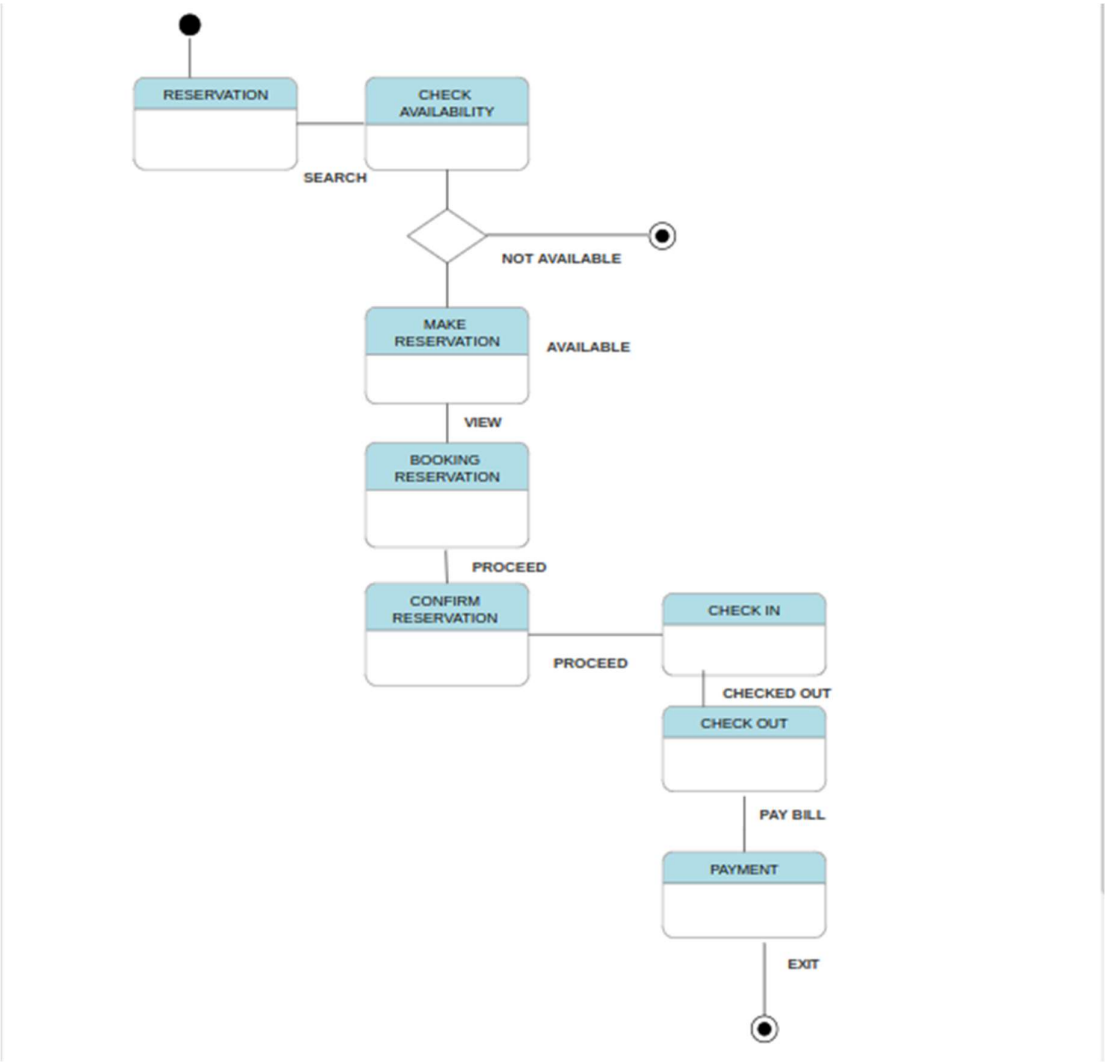
A class diagram shows the structure of a system by defining its classes, attributes, methods, and relationships. It represents real-world entities and how they interact within the system. The diagram helps visualize the system's architecture and the connections between its components. For a Hotel Management System, you'd have classes like Guest, Room, Reservation, and Billing, each with their own details, like the guest's name or the room number, and functions, such as checking in or processing payments.



Class Diagram for Hotel Mangement System

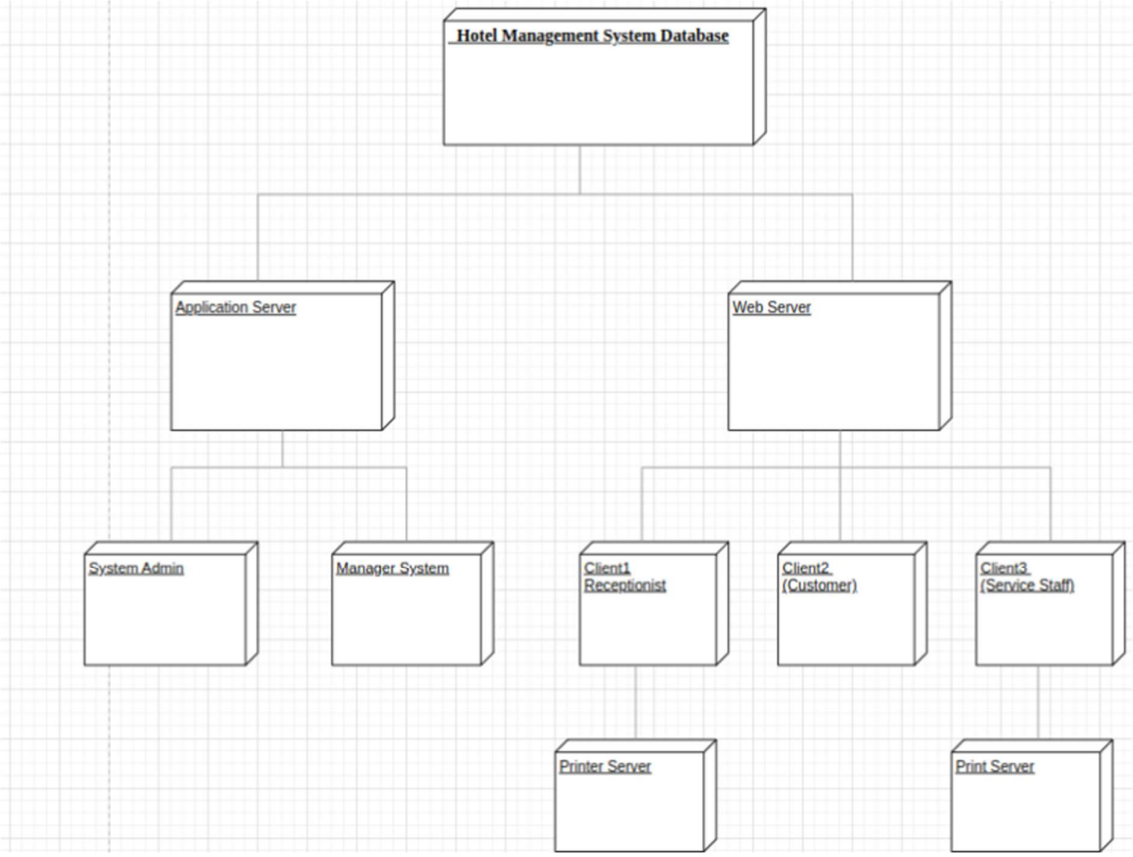
State chart diagram:

A state chart diagram shows the different states an object can be in and how it moves between those states based on events. In a Hotel Management System, a guest might move from "Reserved" to "Checked-in" or "Checked-out" depending on actions taken. It helps track the changes an object goes through over time.



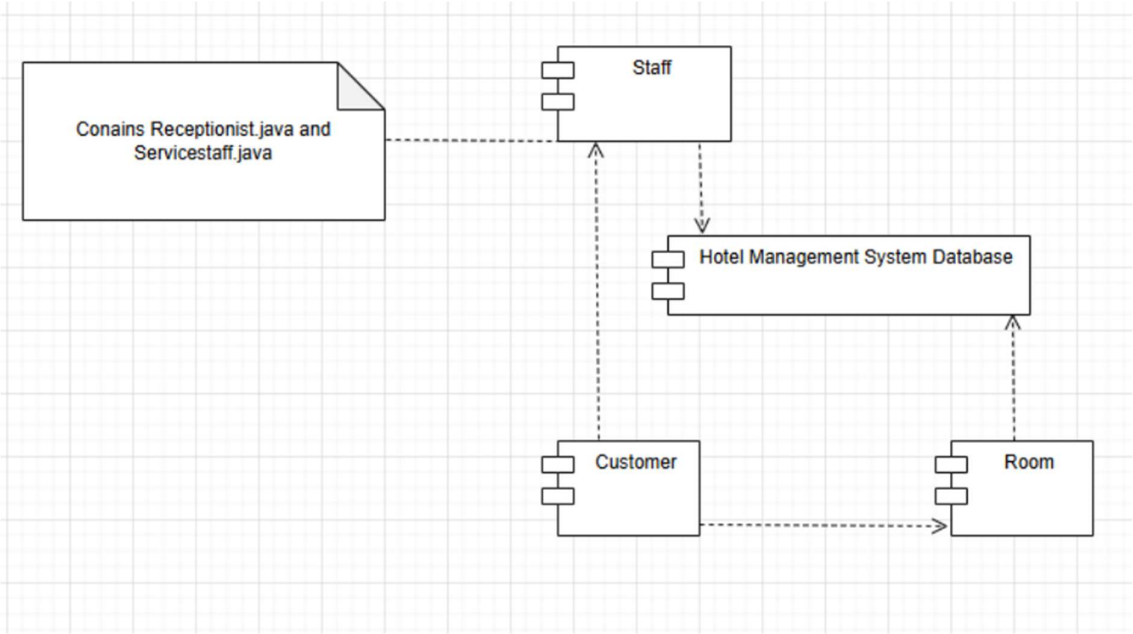
Deployment diagram:

A deployment diagram shows the physical setup of a system, including hardware components and their connections. In a Hotel Management System, it illustrates how devices like servers and reception computers are linked, showing the system's architecture.



Component diagram:

A component diagram shows the major components of a system and how they interact. It focuses on the system's structure by breaking it down into smaller, functional pieces. In a Hotel Management System, components might include the booking system, billing system, and database, each connected to show how they work together to handle tasks like reservations and payments.



HOSPITAL MANAGEMENT SYSTEM

Problem statement: The Hospital Management System (HMS) addresses the challenges faced by healthcare facilities in managing patient records, appointments, billing, and inventory. Manual processes often lead to inefficiencies, data errors, and delays in service delivery, impacting patient care. The system aims to automate and centralize these functions, improving operational efficiency and communication. It provides seamless management of patient data, scheduling, and billing, ensuring accurate records and compliance with regulations. Ultimately, the HMS enhances the overall hospital experience for both patients and staff, optimizing healthcare delivery

Features:

- **Admin login & admin dashboard:** The admin can securely log in to the system, accessing a centralized dashboard to manage hospital operations, view statistics, and oversee users and appointments.
- **User registration:** Patients and hospital staff can register on the system, providing necessary personal details and medical information for efficient service management.
- **Booking system:** Patients can book appointments with doctors through an easy-to-use interface, choosing their preferred doctor and available time slots.
- **Approving/Disapproving request:** The admin or relevant authority can approve or disapprove patient requests, such as appointment bookings or service access, based on availability and requirements.

Software Requirements Specification:

- **Functional requirements:**

These describe what the system should do, i.e., the core functionalities needed to manage the operations of a Hospital Management System.

1 Patient Management

- **Registration:** Create and update patient profiles.
- **EHR Management:** Maintain and update patient's medical history, diagnoses, and treatment records.
- **Patient Search:** Easily retrieve patient records using patient ID, name, or contact details.

2 Appointment Management

- **Appointment Scheduling:** Patients can book, reschedule, or cancel appointments.
- **Doctor's Schedule:** Doctors can update their availability, view, and manage appointments.

- **Appointment Notifications:** Send reminders to patients and doctors via email or SMS.

3 Billing and Payments

- **Invoice Generation:** Generate invoices for consultations, procedures, and treatments.
- **Payment Processing:** Integrate with payment gateways for online payments.
- **Insurance Integration:** Handle insurance claims and generate insurance reports.

4 Inventory and Pharmacy Management

- **Stock Management:** Track and manage stock levels for medicines, medical supplies, and equipment.
- **Inventory Alerts:** Notify administrators of low stock or expired items.
- **Pharmacy POS System:** Simplify medicine dispensing and billing at the pharmacy.

5 Reporting and Analytics

- **Operational Reports:** Generate reports for patient inflow, financials, inventory levels, and staff performance.
- **Regulatory Compliance Reports:** Produce reports for healthcare regulations (e.g., HIPAA compliance).
- **Dashboard:** Visual display of hospital performance metrics, such as patient wait times and revenue.

6 Security and Access Control

- **Role-based Access Control:** Different user roles (e.g., Administrator, Doctor, Patient) with specific access permissions.
- **Audit Logs:** Track system usage and changes made to sensitive data.
- **Data Encryption:** Ensure patient data is encrypted during storage and transmission to maintain privacy.

Non-functional requirements:

These describe the qualities the system must exhibit, ensuring it performs well under various conditions and is maintainable.

1. Performance:
    - The system must provide high responsiveness and efficiency under normal and peak usage conditions. All user actions, such as booking appointments, retrieving medical records, and generating invoices, should have a response time of no more than 2 seconds. Additionally, the system should be capable of handling a significant number of concurrent users (e.g., 500+ simultaneous users) without experiencing performance degradation.

2. Scalability:
    - The HMS should be designed to scale as the hospital grows. It must accommodate an increasing number of patients, doctors, and other users. The system architecture should allow for the addition of new modules (e.g., additional branches, new departments, or specialized services) and should support horizontal scaling, allowing it to handle more users or transactions as needed, without requiring a complete redesign of the system.
3. Availability:
    - The system should be available 99.9% of the time, ensuring that users can access the hospital services without interruptions. This includes considerations for system uptime, backup mechanisms, and disaster recovery procedures. Planned maintenance should be scheduled during off-peak hours, with advance notice given to users.
4. Security:
    - The system must meet high-security standards to protect sensitive patient data and comply with regulations such as HIPAA or GDPR. This includes secure user authentication (e.g., multi-factor authentication), encryption of patient records during transmission and storage, access control based on user roles, and regular security audits to detect vulnerabilities. The system must also prevent unauthorized access to medical and financial records.
5. Usability:
    - The system must be user-friendly, ensuring that all types of users, from patients to hospital staff, can navigate and operate the system easily. The interface should be intuitive, with clear labeling and minimal learning curves. It should also be designed to accommodate users with disabilities, meeting accessibility guidelines.
6. Compatibility:
    - The HMS should be compatible with a wide range of operating systems (e.g., Windows, macOS, Linux) and devices (desktop computers, laptops, tablets, and smartphones). It should also support different web browsers (e.g., Chrome, Firefox, Safari) and allow integration with external systems, such as third-party payment gateways, laboratory systems, and medical equipment.
7. Maintainability:
    - The system should be designed for easy updates and maintenance with minimal downtime and clear documentation.

8. Backup and Recovery:
    - The system must implement automatic daily backups of all critical data (e.g., patient records, appointments, billing) to ensure data integrity. In case of system failure or data loss, recovery processes should allow for quick restoration of data with minimal downtime, ensuring that operations can resume smoothly without significant impact on hospital services.
9. Compliance:
    - The system must adhere to healthcare regulations such as HIPAA and GDPR to ensure the privacy and security of patient data. It should also meet industry standards for data protection and accessibility.

Software requirements:

- Windows XP, Windows 7(ultimate, enterprise)
- Visual Studio 2010
- SQL 08

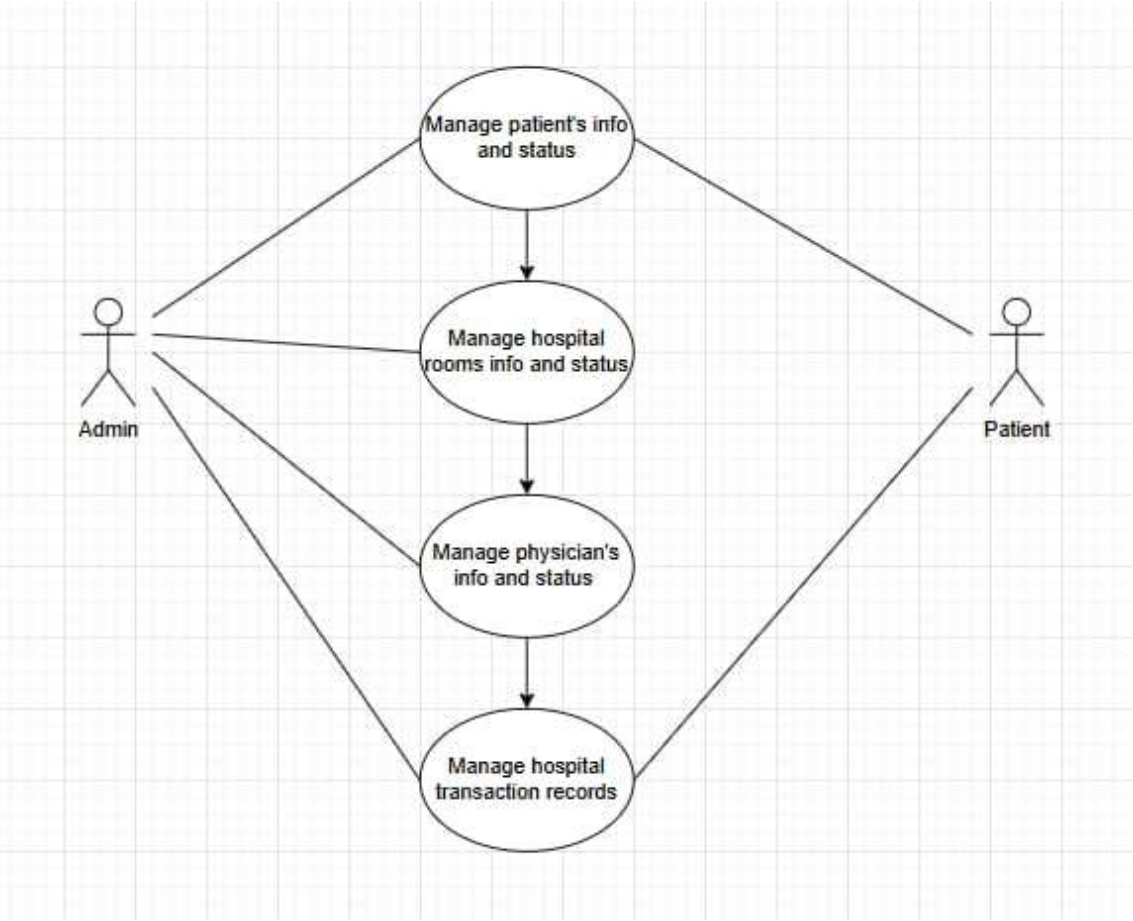
Hardware components:

- Process - p4
- Hard disk - 5GB
- Memory - 1GB RAM

UML Diagrams:

Use-case diagram:

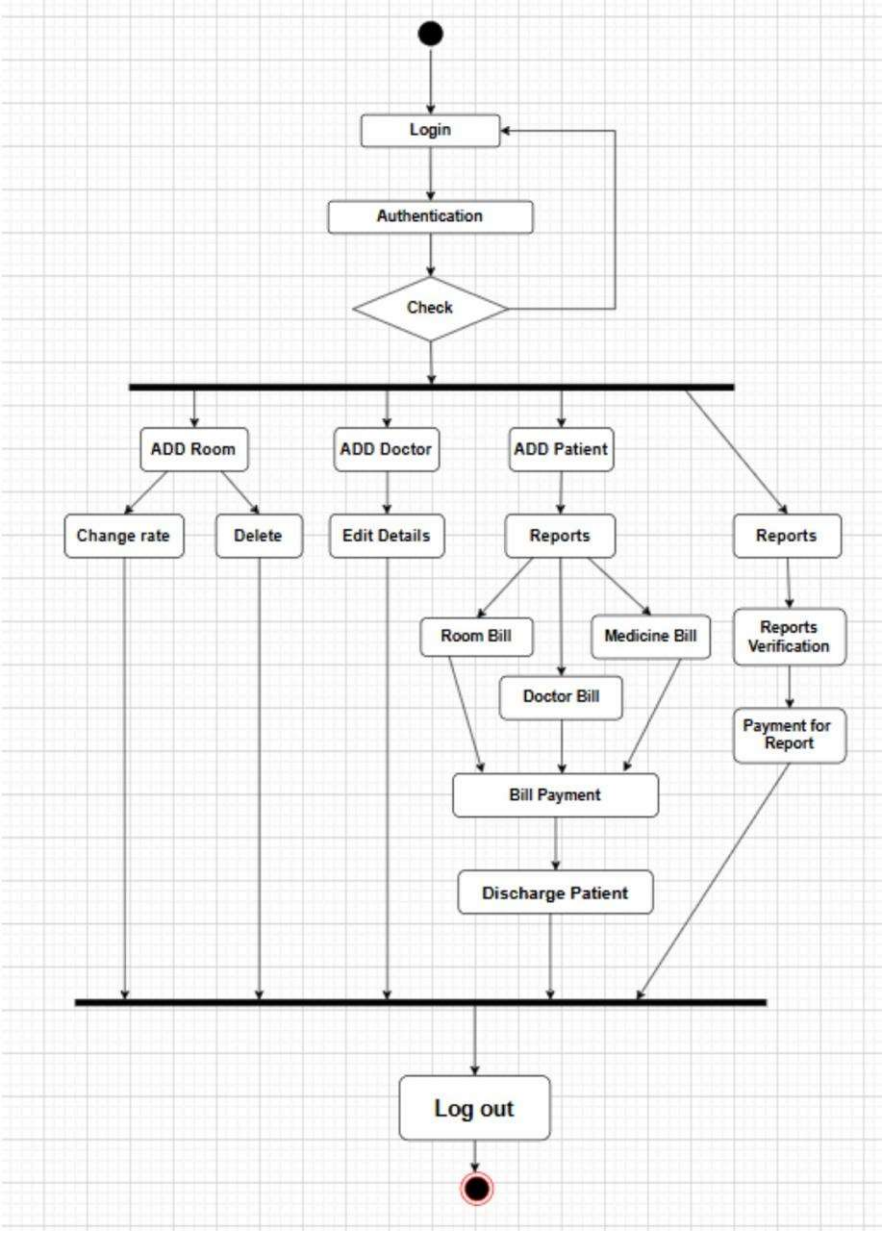
The use case in a Hospital Management System enables patients to book consultations with doctors seamlessly. The patient logs into the system, selects the desired specialization, and views available doctors and time slots. After choosing a doctor and an appropriate time, the patient confirms the appointment, which is then updated in both the patient's and the doctor's schedules. Notifications are sent to both parties via SMS or email to ensure reminders. If no slots are available, the system suggests alternate dates or doctors with similar expertise. The receptionist can assist patients who are unable to book appointments themselves by performing the same process on their behalf. In case of emergencies or changes in the doctor's availability, the system notifies the patient to reschedule. This feature streamlines appointment management, reduces wait times, and enhances patient experience.



Use-case diagram for hospital management system

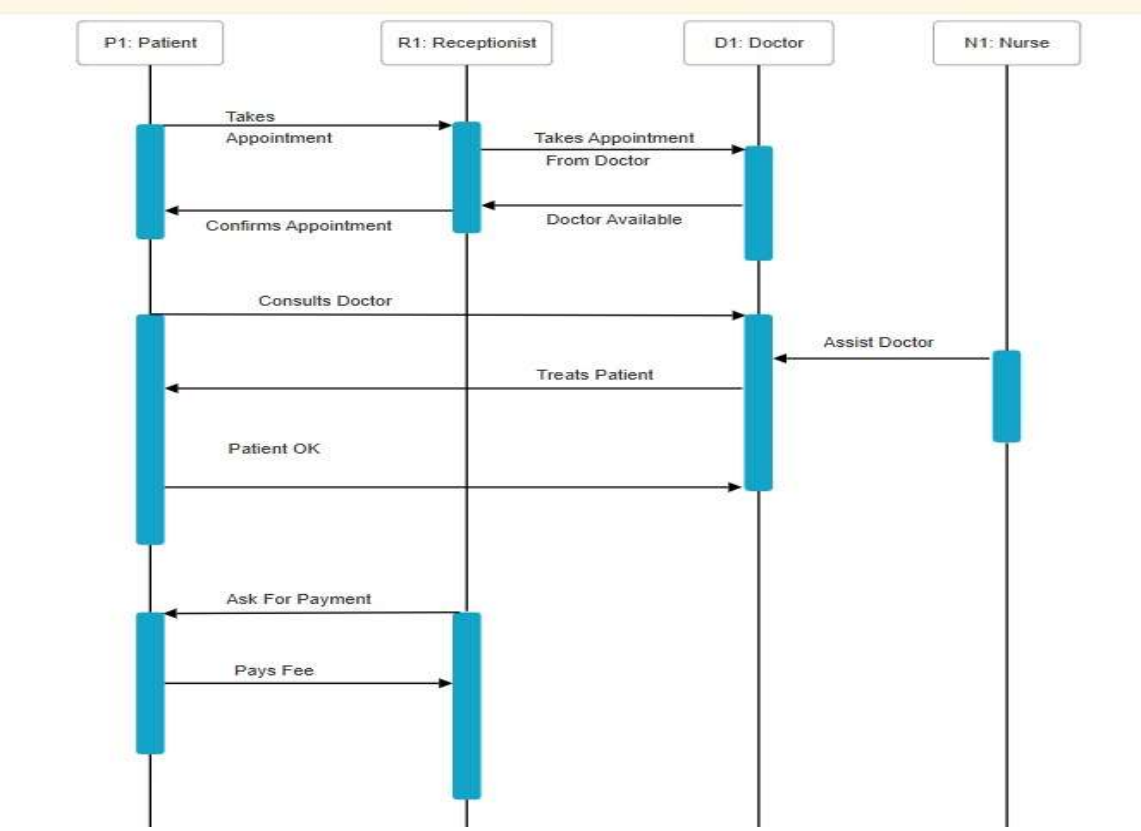
Activity diagram:

An activity diagram for a Hospital Management System outlines the workflow of processes such as patient registration, appointment scheduling, billing, and inventory management. It visualizes the flow of activities, from a patient booking an appointment to the completion of medical treatment and the final billing process. The diagram also includes decision points, like handling payment methods or managing inventory stock levels, ensuring clear understanding of system processes and roles.



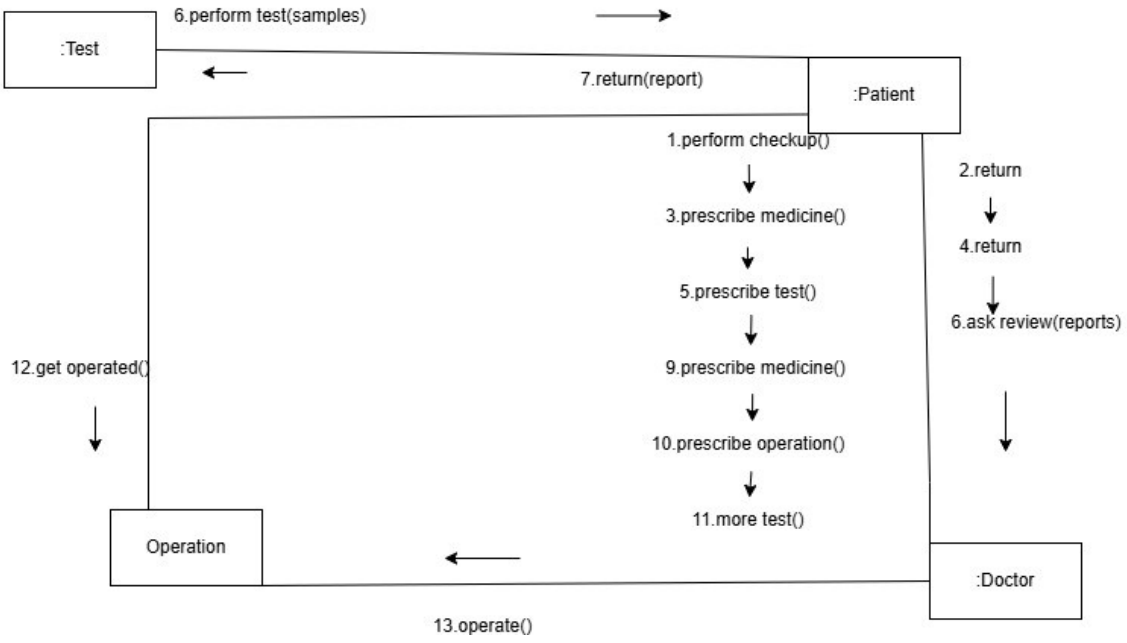
Sequence diagram:

A sequence diagram for a Hospital Management System illustrates the interaction between objects for processes like appointment scheduling. It shows the sequence of messages exchanged between the patient, the system, and the doctor, starting from the patient requesting an appointment to the system confirming the booking. This diagram helps visualize the step-by-step communication flow, ensuring that each actor and system component performs its role in a timely and synchronized manner.



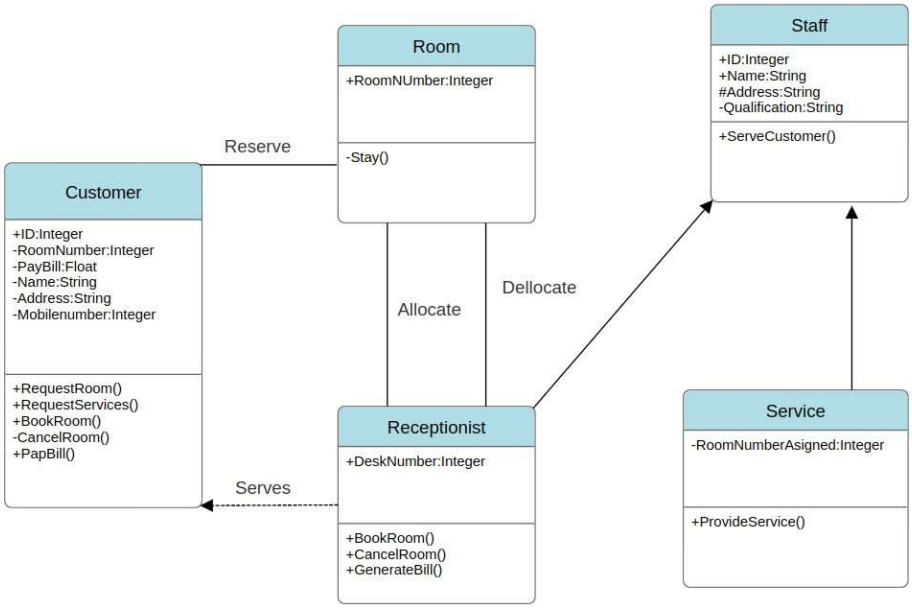
Collaboration diagram:

A collaboration diagram for a Hospital Management System represents the interaction between system components and actors, focusing on how objects collaborate to complete a process. For example, when a patient schedules an appointment, the diagram shows the interactions between the patient, the appointment system, the doctor's schedule, and notification services. This diagram highlights the relationships and communication paths between objects, making it easier to understand the flow of data and responsibilities within the system.



Class diagram:

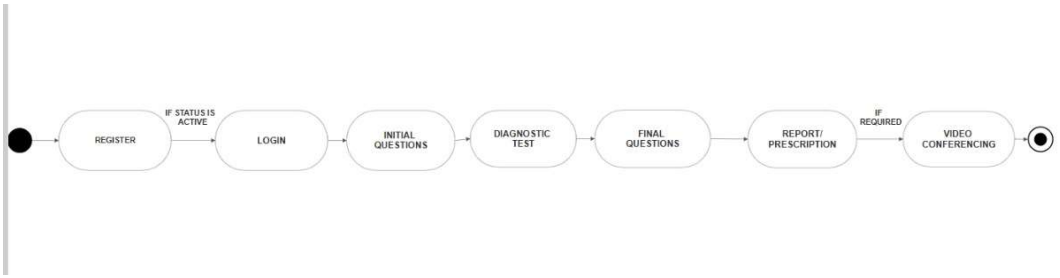
A class diagram for a Hospital Management System defines the structure of the system by representing key classes, their attributes, and methods. It includes classes such as Patient, Doctor, Appointment, Billing, and Inventory, with relationships like associations, inheritances, and dependencies between them. This diagram provides a blueprint for the system's object-oriented design, helping to visualize how different entities within the hospital interact and collaborate to manage operations efficiently.



Class Diagram for Hotel Mangement System

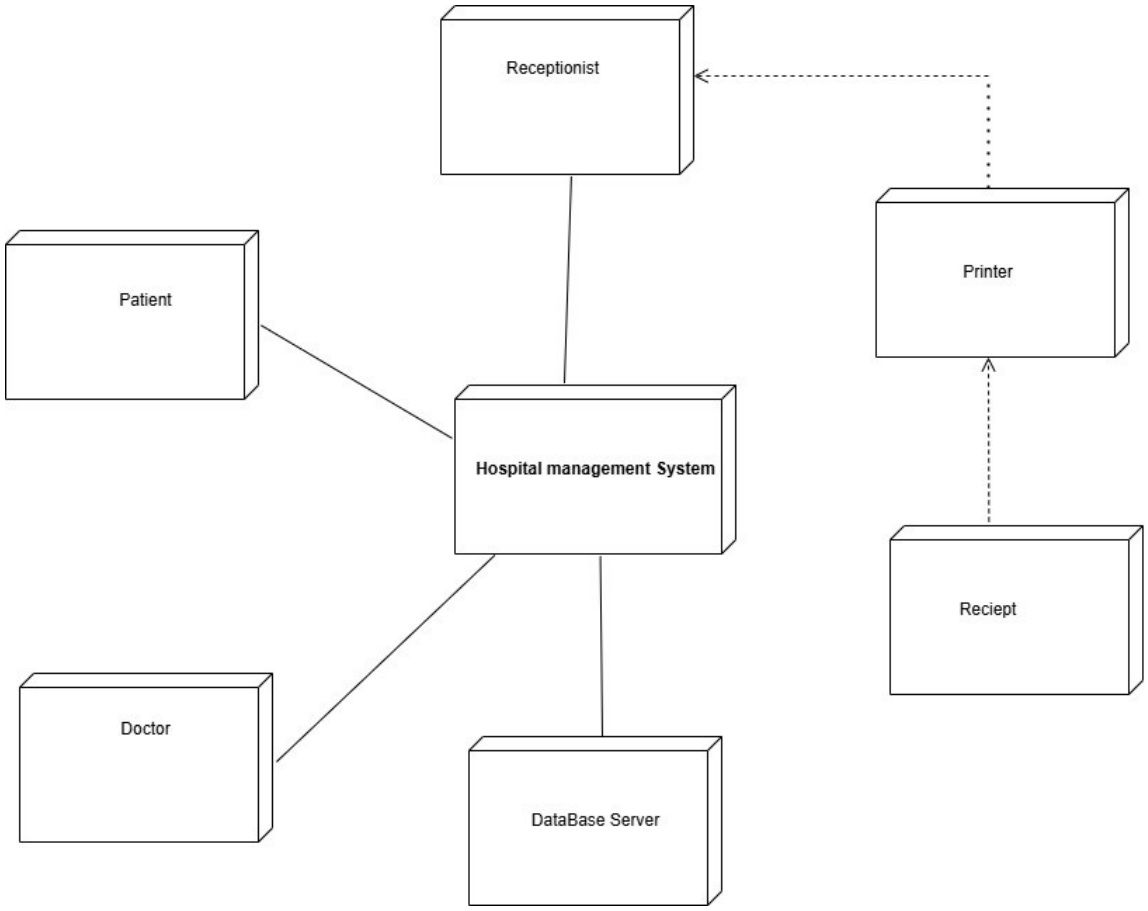
State chart diagram:

A state chart diagram for a Hospital Management System illustrates the different states an object, such as an Appointment or Patient, can transition through during its lifecycle. For example, an appointment can move through states like Scheduled, In Progress, Completed, and Cancelled based on interactions with the system. This diagram helps visualize the dynamic behavior of objects, ensuring that the system responds appropriately to different events or changes in state throughout hospital operations.



Deployment diagram:

A deployment diagram for a Hospital Management System outlines the physical deployment of the system components across hardware nodes. It shows how the system's software, including the web application, database server, and client devices, is distributed across different servers and devices. This diagram helps visualize the system's architecture, ensuring efficient resource allocation, scalability, and communication between various components such as the user interface, application server, and database server.



Component diagram:

A component diagram for a Hospital Management System depicts the high-level structure of the system by showing the main components and their relationships. It includes components such as the User Interface, Backend Server, Database, Payment Gateway, and Notification Service. This diagram helps to understand how each part of the system interacts with others, ensuring smooth data flow and integration between modules like patient management, billing, and appointment scheduling.