
Software Requirements and Design Document

For

Hospital Management System

Prepared by

**Muhammad Sarmad (22i-0941),
Muhammad Huzaifa (22i-1220),
Muhammad Umar Hassan (22i-0942)**

Iceberg Solutions

November 24, 2024

Table of Contents

1. Introduction.....	1
1.1 Purpose	1
1.2 Product Scope	1
1.3 Title.....	1
1.4 Objectives	1
1.5 Problem Statement.....	2
2. Overall Description	2
2.1 Product Perspective	2
2.2 Product Functions.....	3
2.3 List of Use Cases	3
2.4 Extended Use Cases.....	3
Use Case No. 1: Handle Patient Visit	3
Use Case No. 2: Admit Patient	6
Use Case No. 3: Checkup Patient	9
Use Case No. 4: Perform Tests	11
Use Case No. 5: Discharge Patient.....	12
Use Case No. 6: Track and update schedule of staff	14
Use Case No. 7: Update Patient Health.....	15
Use Case No. 8: Update Medicine Stock	17
Use Case No. 9: Manage Staff	19
Use Case No. 10: Handle Sale of Medicine	20
2.5 Use Case Diagram	22
3. Other Nonfunctional Requirements.....	23
3.1 Performance Requirements.....	23
3.2 Safety Requirements.....	23
3.3 Security Requirements.....	23
3.4 Software Quality Attributes.....	23
3.5 Business Rules.....	24
3.6 Operating Environment	24
3.7 User Interfaces.....	24
4. Domain Model	27
5. System Sequence Diagram	28
6. Sequence Diagram	33
7. Class Diagram	38
8. Component Diagram	39
9. Package Diagram	40
10. Deployment Diagram.....	41
11. Extended Class Diagram	42

1. Introduction

1.1 Purpose

The product specified in this document is the **Hospital Management System (HMS)**, developed as part of the CS3004 Software Design and Analysis course. The HMS project covers critical operational areas within a hospital, including patient management, staff scheduling, inventory control, billing, and medicine sales. This SRS document encompasses the entire HMS, outlining use cases and system interactions across multiple departments such as reception, medical, lab, and pharmacy. The system is intended to streamline hospital operations, enhance patient care, and optimize resource management.

1.2 Product Scope

The **Hospital Management System (HMS)** streamlines hospital operations by managing in-patients, out-patients, pharmacy services for medicines, and laboratory services for lab tests. It aims to enhance efficiency, reduce wait times, and improve patient care while supporting the hospital's goal of delivering high-quality, timely healthcare through automated processes.

1.3 Title

Hospital Management System (HMS): An Integrated Solution to Streamline Hospital Operations, Enhance Patient Care, and Optimize Resource Management.

1.4 Objectives

- **Efficient Patient Management**
 - Streamline patient registration, admission, and discharge processes.
 - Ensure accurate and timely patient information accessibility for healthcare providers.
- **Optimized Staff Scheduling**
 - Automate scheduling for doctors, nurses, and other medical staff.
 - Prevent scheduling conflicts and ensure optimal staff availability.
- **Effective Inventory Control**
 - Track hospital supplies, medications, and equipment in real-time.
 - Provide alerts for low stock levels to ensure continuous availability.
- **Improved Billing and Payment Handling**
 - Facilitate quick and accurate billing for patients and manage payments.
 - Ensure transparency and accuracy in financial transactions.
- **Real-Time Data and Reporting**

- Provide real-time updates on patient health, inventory, and staff schedules.
- Generate detailed reports for hospital administration to monitor and optimize operations.
- **Enhanced Patient Care**
 - Improve communication between patients, doctors, and medical staff.
 - Ensure timely treatments and minimize patient wait times.

1.5 Problem Statement

Hospitals often face challenges in managing their operations efficiently due to the complexity of patient registration, staff scheduling, inventory management, and billing. These manual processes can lead to errors, delays, and increased administrative workload, affecting the quality of patient care and overall hospital efficiency. Moreover, hospitals struggle with maintaining accurate, real-time records, leading to miscommunications and potential delays in treatments, tests, or medication. The lack of automation in these key areas contributes to operational inefficiencies and decreases the overall patient experience.

The **Hospital Management System (HMS)** aims to address these issues by automating and integrating hospital functions. This solution ensures accurate data handling, efficient resource allocation, and timely decision-making, ultimately improving patient care and hospital management. With its user-friendly interface, the system is feasible for implementation in hospitals of various sizes and can be easily adapted to meet the evolving needs of healthcare providers, reducing manual errors and optimizing resource utilization.

2. Overall Description

2.1 Product Perspective

The **Hospital Management System (HMS)** is a new, self-contained product designed to automate hospital operations. It follows a 3-layer architecture:

1. **Presentation Layer (FXML and Controllers):** This layer provides the user interface, allowing hospital staff to interact with the system through FXML files and controllers.
2. **Business Layer:** This core layer handles system functionalities such as patient management, staff scheduling, and inventory control. It is built using design patterns for high cohesion and low coupling, ensuring modular and maintainable code.
3. **Data Layer (SQL Database):** The business layer communicates with the SQL database to store and retrieve data, managing hospital records and schedules.

This architecture ensures efficient data flow and clear separation of concerns, making the system scalable and maintainable.

2.2 Product Functions

The **Hospital Management System (HMS)** must support the following major functions:

1. Patient and Appointment Management

- **Appointment Scheduling:** Allows patients to book, modify, or cancel appointments.
- **Patient Medical History:** Maintains a detailed record of patient medical history for healthcare providers.
- **Prescription Management:** Stores prescriptions digitally for future access.

2. Hospital Resource and Operations Management

- **Pharmacy Management:** Manages drug inventory, prescription fulfillment, and dispensing.
- **Bed and Ward Management:** Tracks bed availability and occupancy for patient admissions.
- **Laboratory System Management:** Handles lab tests, sample tracking, and report generation.

3. Billing and Staff Management

- **Billing and Invoicing:** Automates billing for services like consultations, treatments, and tests.
- **Insurance Claims Management:** Integrates with insurance for efficient claims processing.
- **Staff Scheduling and Management:** Organizes and optimizes staff shifts, duties, and workload.

2.3 List of Use Cases

Use Case No. 1: Handle Patient Visit
Use Case No. 2: Admit Patient
Use Case No. 3: Checkup Patient
Use Case No. 4: Perform Tests
Use Case No. 5: Discharge Patient
Use Case No. 6: Track and update schedule of staff
Use Case No. 7: Update Patient Health
Use Case No. 8: Update Medicine Stock
Use Case No. 9: Manage Staff
Use Case No. 10: Handle Sale of Medicine

2.4 Extended Use Cases

NOTE: For the receptionist tasks we have used the Actor Word ‘Patient’ as a collective name for Patient and Family for the ease of writing.

Use Case No. 1: Handle Patient Visit

Scope: Hospital Management System

Level: User Goal

Primary Actor: Receptionist

Stake Holders and Interests:

- Receptionist: Wants to efficiently handle patient check-ins and visits, ensuring accurate and quick registration and updates to the system.
- Patient: Expects minimal wait time during check-in, accurate and private handling of personal and medical information.
- Doctor: Requires timely and accurate patient information to be available before the patient's visit.
- Lab technician: Requires accurate information to be available before the patient's visit to the lab.
- Administrative Staff: Seeks to maintain a smooth and efficient operation of the hospital.
- Nurse: Need to be informed about patient arrivals to prepare for treatments or examinations.

Pre-Conditions: Receptionist is identified and authenticated.

Post-Conditions: Patient's data is updated. Fee is obtained. Patient is directed to the Doctor's room or the Laboratory or is handed the Lab Report.

Main Success Scenario:

Actor Action	System Responsibility
1. Patient arrives at the Receptionist desk.	
2. Receptionist asks patient for his/her CNIC and patient gives his/her CNIC.	
3. Receptionist enters the CNIC in the system.	4. System verifies that the patient is registered.
5. Patient asks to consult doctor.	
6. Receptionist asks for the consult fee and Patient pays the fee.	
7. Receptionist creates new visit for the patient and adds the patient in the doctor queue.	8. System assigns the patient a visit number.

9. Receptionist hands the visit number to the patient.	
10. The receptionist receives confirmation from the doctor to bring a new patient in.	
11. Receptionist signals the patient with the lowest visit number to go in the doctor room.	

Alternate Scenarios:

Actor	System
A*: Any time a system crash occurs: 1. Receptionist restarts the system. B*: Any time a required operation initiated by the receptionist fails: 1. Receptionist retries the operation.	2. System enters the recovery mode and loads the previous state.
1.Receptionist asks patient for the required information and enters this information in system.	4a. System indicates that the patient is not registered. 2. System registers the patient.
5a. Patient asks for a Medical Test: 1.Receptionist asks for the type of medical test. 2. Patient responds with the type of test. 3a. This test isn't done in the laboratory: 1. Receptionist tells the customer that this test is not possible here. 3b. Test is done in the laboratory: 1. Receptionist asks for the test payment and receives the payment. 1a. Patient doesn't have enough amount: 1. Receptionist cancels the lab test procedure. 2. Receptionist adds the patient in the lab queue. 4. Receptionist hands the token number to the patient. 5. Receptionist receives confirmation from the lab technician and signals	3. System assigns a token number to the patient.

<p>the patient with lowest token number to move to the lab.</p> <p>4a. Patient is not found in the waiting room:</p> <ol style="list-style-type: none"> 1.Receptionist removes the patient from the queue. 	
<p>5b. Patient asks for a medical report:</p> <ol style="list-style-type: none"> 1. Receptionist searches for the particular report. 3. Receptionist prints the report. 4. Receptionist hands the report to the patient and the patient leaves. <p>1b. Receptionist doesn't find the report or the report is not yet completed.</p> <ol style="list-style-type: none"> 1. Receptionist informs the patient. 	<ol style="list-style-type: none"> 2. System presents the reports. 3. System generates the report.
<p>5c. Patient/Family asks to pay pending dues of an admitted patient:</p> <ol style="list-style-type: none"> 1. Receptionist enters the patient's CNIC in the system 3. Receptionist asks system for the pending dues of the patient. 4.Patient/Family pays the pending dues. 5. Receptionist receives the payment and updates the patient's data. 	<ol style="list-style-type: none"> 2. System verifies that the patient is admitted in the hospital 2b. Patient is not admitted: <ol style="list-style-type: none"> 1. Receptionist tells that the patient is not admitted in the hospital. 4. System provides pending dues. 4b: There are no pending dues of patient: <ol style="list-style-type: none"> 1. Receptionist tells that there are no pending dues for the patient. 6.System handles the updates.
<p>6a. Patient doesn't have enough money:</p> <ol style="list-style-type: none"> 1.Receptionist cancels the patient visit. 	
<p>11a.The patient is not found in the waiting room:</p> <ol style="list-style-type: none"> 1.Receptionist removes the patient from the queue. 	

Use Case No. 2: Admit Patient

Scope: Hospital Management System

Level: User Goal

Primary Actor: Receptionist

Stake Holders and Interests:

- Receptionist: Wants to efficiently handle patient admission process.
- Patient: Wants to receive timely and efficient care during the admission process and informed about the admission process, costs, and estimated duration of stay.
- Doctor: Requires constant update for the admitted patient's health.
- Administrative Staff: Seeks to maintain a smooth and efficient admission operation.
- Nurse: Need to be informed about patient arrivals to prepare for treatments or examinations.

Pre-Conditions: Receptionist is identified and authenticated.

Post-Conditions: Patient's data is updated. Bed is allocated for the patient. A nurse and doctor is assigned to the patient. The patient can go to his/her bed.

Main Success Scenario:

Actor	System
1. Patient arrives at the Receptionist desk.	
2. Receptionist asks patient for his/her CNIC and patient gives his/her CNIC.	
3. Receptionist enters the CNIC in the system.	4. System verifies that the patient is registered.
5. Patient asks to be admitted in the hospital.	
6. Receptionist searches for the admission request for the patient.	7. System verifies that there is a request to admit this patient.
8. Receptionist searches for available beds.	9. System provides a list of available beds.
10. Receptionist finds a free bed and asks the patient for a buffer payment amount.	
11. Patient pays the amount.	

12. Receptionist assigns the bed and doctor to the patient and enters any required information .	
13. Receptionist hands a bed allocation receipt to the patient.	
14. Patient receives the receipt and goes to the allocated bed.	

Alternate Scenarios:

Actor	System
A*: Any time a system crash occurs: 1. Receptionist restarts the system. B*: Any time a required operation initiated by the receptionist fails: 1. Receptionist retries the operation.	2. System enters in the recovery mode and loads the previous state.
	4a. System indicates that the patient is not registered:
1. Receptionist tells the customer that the bed cannot be reserved without doctor's prescription. 1b. The patient was received by an ambulance: 1. Receptionist asks for the information required and registers the patient. 2. Receptionist asks for the buffer amount. 3. Patient pays the amount. 3b.Patient doesn't have the amount: 1.Receptionist will add the payment in pending payments list of the patient. 4. Receptionist will search for the bed reserved by the admin for the ambulance patient. 5. Receptionist assigns the bed and doctor to the patient and hands the patient a bed allocation receipt.	5.System will give the list of beds reserved by admin for ambulance patients.
7a. System indicates that there is no admission request for this patient: 1. Receptionist tells customer that bed	

cannot be reserved without doctor's consent.	
10a. Receptionist can't find a free bed: 1. Receptionist tells the patient that no bed is available yet and cancels the admit request.	
11a. Patient doesn't have enough amount: 1. Receptionist tells the patient that the bed cannot be reserved.	

Use Case No. 3: Checkup Patient

Scope: Hospital Management System

Level: User Goal

Primary Actor: Doctor

Stake Holders and Interests:

- Patient: Wants timely and accurate checkups, clear communication about his/her health status,
- Doctor: Wants efficient access to the patient's medical history, ability to update checkup results, and an organized system for recording patient visits.
- Receptionist: Wants the checkup process to be fast and efficient so that he/she can manage the waiting queue correctly and manage any admission requests.
- Admin Staff: Wants optimized checkup workflow to reduce patient wait times.

Pre-Conditions: Doctor is identified and authenticated.

Post-Conditions: Patient's data is updated. Patient can be admitted to the hospital as per prescription and can buy medicines prescribed by the doctor. Patient can also ask for medical tests (if prescribed).

Main Success Scenario:

Actor	System
1. Doctor signals receptionist to bring new patient in.	
2. Patient arrives in the room with a token number.	

3. Doctor takes the token number from patient and searches for the token number in queue.	4. System verifies that the token number exists and is the lowest token number.
5. Doctor performs the necessary checkup procedure.	
6. Doctor enters the necessary prescription details and any medicine and lab tests needed.	7. System records the prescription data.
8. Doctor prints the prescription.	9. System generates the prescription.
10. Doctor hands the prescription to the patient.	
11. Patient receives the prescription and leaves the doctor room.	

Alternate Scenarios:

Actor	System
A* Any time a system crash occurs: 1. Doctor restarts the system. B* : Any time the required operation initiated by doctor fails: 1. Doctor retires the operation.	2. System enters the recovery mode and loads the previous state.
1a. There is no patient in the waiting queue yet: 1. Doctor waits for a new patient to arrive in hospital.	
(Unlikely to happen as receptionist checks the incoming patient's token number.) 1. Doctor sends the patient back to the waiting room and signals the receptionist to bring the correct patient in.	4a. System indicates invalid or out of order token number.
5a. Doctor searches for patient's medical history and tests.	1. System provides the necessary details.

6a. The patient needs to be admitted: 1. Doctor generates an admission request for the patient.	2. System stores this request for further operation by the receptionist.
---	--

Use Case No. 4: Perform Tests

Scope: Hospital Management System

Level: User Goal

Primary Actor: Lab Technician

Stake Holders and Interests:

- Lab Technician: Wants access to accurate patient details, clear instructions on tests to be performed, ability to input test results easily.
- Patient: Wants accurate and timely results, confidentiality of their medical data, clear communication on test procedures, and minimal wait time for test results.
- Doctor: Wants timely access to accurate test results to make informed decisions on patient diagnosis and treatment.
- Admin Staff: Wants efficient use of lab resources, ensuring that lab processes adhere to hospital protocols and standards, reducing turnaround time for tests, and maintaining high patient satisfaction.
- Receptionist: Wants to timely receive the completed reports to give them to the respective patients.

Pre-Conditions: Lab Technician is identified and authenticated.

Post-Conditions: Test results are accurately recorded in the system. Receptionist receives the completed reports. Tests results are properly linked with the patient's medical history.

Main Success Scenario:

Actor	System
1. Lab Technician signals receptionist to bring new patient in.	
2. Patient arrives in the room with a token number.	
3. Lab Technician takes the token number from patient and searches for the token number in queue.	4. System verifies that the token number exists and is the lowest token number.

5. Lab technician takes the patient's sample.	6. System assigns a unique number to the sample.
7. Lab technician puts a label with the number on the physical sample and stores it.	
8. Lab technician gives a date/time to the patient to collect his/her report from the receptionist desk.	
9. Patient leaves the lab.	

Alternate scenarios:

Actor	Patient
A* . Any time a system crash occurs: 1. Lab technician restarts the system B* . Any time the initiated operation fails to execute: 1. Lab technician retries the operation.	2. System enters the recovery mode and restores it previous state.
1a . There is no waiting patient: 1. Lab technician performs tests on the pending samples (if available). 2. Lab technician generate reports for the tests.	3. System stores the reports.
(Unlikely to happen as receptionist checks the incoming patient's token number.) 1. Lab technician sends the patient back to the waiting room and signals the receptionist to bring the correct patient in.	4a . System indicates invalid or out of order token number.

Use Case No. 5: Discharge Patient**Scope:** Hospital Management System**Level:** User Goal**Primary Actor:** Receptionist**Stake Holders and Interests:**

- Receptionist: Wants to efficiently handle patient discharge process and wants no errors.
- Patient: Wants to be discharged promptly and with clear instructions for post-discharge care, medication.

- Administrative Staff: To streamline discharge processes to improve bed availability and reduce patient wait times while ensuring regulatory and billing compliance.
- Family: to be informed of the patient's condition and receive guidance on how to assist with post-discharge care at home

Pre-Conditions: Receptionist is identified and authenticated and the patient is Admitted in the hospital.

Post-Conditions: Patient's data is updated. Patient receives the discharge slip. Allocated bed is marked available. The patient is ready to leave the hospital.

Main Success Scenario:

Actor	System
1. Patient arrives at the Receptionist desk.	
2. Patient asks to be discharged.	
3. Receptionist asks patient for his/her CNIC and patient gives his/her CNIC.	
4. Receptionist enters the CNIC in the system.	5. System verifies that there is a discharge request for the patient.
6. Receptionist searches for the pending dues of the patient.	7. System presents the pending dues.
8. Receptionist asks for the payment of pending dues.	
9. Patient pays the dues.	
10. Receptionist receives the payment and marks the bed available.	11. System updates the bed state.
12. Receptionist enters the CNIC of the patient to print the discharge slip.	13. System generates the discharge slip.

Alternate Scenarios:

Actor	System
A* Any time a system crash occurs: 1. Receptionist restarts the system. B* : Any time the required operation initiated by doctor fails: 1. Receptionist retires the operation.	2. System enters the recovery mode and loads the previous state.
1. Receptionist tells patient that the patient cannot be discharged without doctor consent.	5a. System indicates that there is no discharge request for the patient:
1. Receptionist marks the bed as available and enters the CNIC of the patient to print the discharge slip.	7a. System indicates that there are no pending dues: 2. System prints the discharge slip.
9a. Patient doesn't have enough amount: 1. Receptionist asks to arrange the amount.	

Use Case No. 6: Track and update schedule of staff

- a) **Use case name:** Track and update schedule of staff
- b) **Scope Name:** Hospital Management System (HMS)
- c) **Level:** User Goal
- d) **Primary Actor:** Admin
- e) **Stakeholders and interests:**
1. **Staff (Doctors, Nurses, etc.):** Ensure fair and accurate scheduling for work shifts.
 2. **Admin:** Efficiently manage staff and ensure no shift is left unstaffed.
- f) **Preconditions:**
1. Staff members must be registered in the system.
 2. Shifts must be predefined and stored in the scheduling system.
 3. Staff availability details (e.g., vacations, leaves) are up-to-date.
- g) **Post conditions:**
- The schedule is updated for relevant staff members, and notifications are sent to them.

h) Main success scenario:

Actor Action (or Intention)	System Responsibility
1. Admin logs into the system.	Verifies login credentials and grants access to the scheduling module.
2. Admin navigates to the staff scheduling section.	Displays the current schedule for staff.
4. Admin selects a specific department and staff member to modify.	Shows available shifts and schedule for the selected staff.
5. Admin updates, adds, or deletes shifts.	Validates changes and ensures there are no conflicts (e.g., overlapping shifts).
6. Admin confirms the changes.	Updates the schedule in the system and Sends notifications to affected staff members.
7. Admin logs out.	

i) Extensions:

Actor Action (or Intention)	System Responsibility
• Admin selects a staff member but the record is not found.	System displays an error: "Staff member not found."
• Admin tries to assign a shift, but the selected staff is unavailable (e.g., on leave).	System suggests other available staff or notifies about the conflict.
• Admin confirms the changes, but a shift conflict is detected.	System prompts a conflict warning and suggests possible resolutions (e.g., reassign shift or adjust timings).
• Admin accidentally assigns overlapping shifts.	System prevents the changes from being saved and prompts a correction.
• Admin confirms changes, but the system experiences a network failure.	System retries the update, logs the error, and notifies the manager to try again later.
• Notifications fail to send to the staff.	System retries notification and sends a follow-up email if the issue persists.

Use Case No. 7: Update Patient Health

a) Use case name: Update Patient Health

b) Scope Name: Hospital Management System (HMS)

c) Level: User Goal

d) Primary Actor: Doctor/Nurse

e) Stakeholders and interests:

1. **Patient:** Accurate and up-to-date health records for proper treatment.
2. **Doctor/Nurse:** Reliable and real-time access to patient health information for decision-making.

f) Preconditions:

1. Patient must have an existing record in the system.
2. Doctor or Nurse must be logged into the system with proper access rights.
3. Patient's latest test results, vitals, or medical condition updates must be available.

g) Post conditions:

Patient's health records are updated, and alerts are triggered for abnormal data.

h) Main success scenario:

Actor Action (or Intention)	System Responsibility
1. Doctor or Nurse logs into the system.	Verifies login credentials and grants access to the patient health module.
2. Doctor or Nurse searches for and selects the patient.	Displays the selected patient's current health record.
4. Doctor or Nurse updates the patient's vitals, diagnosis, or medical test results.	Validates the data entry (checks for errors or inconsistencies).
5. Doctor or Nurse confirms the updates.	Saves the updates and triggers alerts if abnormal values are detected.
6. Doctor or Nurse logs out.	Ensures the updated health records are accessible to all relevant medical staff.

i) Extensions

Actor Action (or Intention)	System Responsibility
• Doctor or Nurse Searches for a patient, but the record is not found.	System displays an error message: "Patient not found." It suggests checking the patient ID or searching by other criteria.
• Doctor or Nurse enters new vitals, but the values are out of the acceptable range.	System prompts a warning and requests confirmation before saving the data.
• Doctor or Nurse tries to enter duplicate information (e.g., same test results	System alerts about the duplication and asks if the new data should overwrite the old

multiple times).	entry.
<ul style="list-style-type: none"> Doctor or Nurse inputs incomplete information (e.g., missing fields). 	System highlights the missing fields and prevents submission until the data is complete.
<ul style="list-style-type: none"> Doctor or Nurse tries to save the updates, but the system crashes or loses connection. 	System temporarily saves the data locally and attempts to sync once the connection is restored.
<ul style="list-style-type: none"> Doctor or Nurse attempts to log out, but there are unsaved changes. 	System prompts a warning about unsaved data and asks for confirmation before logging out.
<ul style="list-style-type: none"> Updated records fail to sync with external systems (e.g. pharmacy systems). 	System logs the issue and retries syncing at regular intervals. If it persists, it notifies the technical support team.
<ul style="list-style-type: none"> Doctor or Nurse generates a discharge request for the patient and generates Discharge Slip 	System send request to Receptionist

Use Case No. 8: Update Medicine Stock

- a) **Use case name:** Update Medicine Stock
- b) **Scope Name:** Hospital Management System (HMS)
- c) **Level:** User Goal
- d) **Primary Actor:** Pharmacist
- e) **Stakeholders and interests:**
1. **Administration:** Ensures availability of necessary items and minimizes wastage.
 2. **Medical Staff (Doctors, Nurses):** Ensure medicines, equipment, and consumables are available when needed for patient care.
 3. **Pharmacists:** Needs up-to-date drug stock to avoid shortages and ensure timely delivery of medication.
 4. **Surgical Team:** Requires surgical instruments and equipment to be available and maintained for operations.
 5. **Lab Technicians:** Need reagents, test kits, and equipment for conducting medical tests.

f) Preconditions:

1. The Pharmacist is authenticated in the system.
2. The system has access to current inventory data.
3. All inventory categories (Ambulances, Beds, Nurse's Supplies, Doctor's Equipment, Lab Equipment, and Stationery) are predefined in the system.

g) Post conditions:

Stock levels are updated accurately, and reports on inventory status are generated and available.

h) Main success scenario:

Actor Action (or Intention)	System Responsibility
1. Pharmacist logs into the Hospital Inventory Management System.	System authenticates the Pharmacist and grants access to the inventory management features.
2. Pharmacist selects the "Manage Stock" option and the desired department.	System displays current inventory levels for items in the selected department.
4. Pharmacist reviews low stock alerts and decides to add or update stock for specific items.	System triggers notifications for items with low stock levels that require replenishment.
5. Pharmacist performs actions to add, update, or remove stock items as necessary.	System validates the inputs, processes the actions, and confirms successful updates.
6. Pharmacist reviews recent changes and generates inventory reports to assess stock status for the selected department.	System compiles and displays updated inventory reports, reflecting recent changes and any alerts for further action.
7. Pharmacist logs out of the system after verifying all updates and ensuring data integrity.	System securely logs out the Pharmacist, saving all updates and ensuring data integrity.

j) Extensions:

Actor Action (or Intention)	System Responsibility
<ul style="list-style-type: none"> • Pharmacist tries to perform an action on an item (e.g., Ambulances, Beds, and Nurse's Supplies) in the selected department, but the item is not found in the system. 	System displays an error message: "Item not found." It suggests checking the item ID or searching by category.
<ul style="list-style-type: none"> • Pharmacist enters new stock levels for a 	System prompts an error message indicating

category (e.g., Doctor's Equipment), but the values are negative or invalid.	invalid input and requests the Pharmacist to correct the values.
<ul style="list-style-type: none"> Pharmacist attempts to update stock levels, but the system encounters a database error. 	System displays a failure message, advising the Pharmacist to check the connection and allowing a retry option.
<ul style="list-style-type: none"> Pharmacist removes an item from stock (e.g., Lab Equipment), but an item is not listed in the selected department. 	System alerts the Pharmacist that the item cannot be removed and suggests reviewing the stock list for accuracy.
<ul style="list-style-type: none"> Pharmacist updates stock levels for an item in the selected department but tries to enter duplicate data for the same item at same time. 	System alerts the Pharmacist about the duplication and asks if the new data should overwrite the old entry or add it as it is.
<ul style="list-style-type: none"> Pharmacist attempts to log out without saving changes to stock updates. 	System prompts a warning about unsaved changes and asks for confirmation before logging out.
<ul style="list-style-type: none"> System detects a significant discrepancy in stock levels for items in a department (e.g., an Ambulance missing from inventory). 	System alerts the Higher Admin and other relevant stakeholders about the discrepancy for further investigation.
<ul style="list-style-type: none"> Pharmacist tries to access stock management but faces permission issues. 	System displays an error message indicating insufficient permissions and advises contacting a higher-level administrator.

Use Case No. 9: Manage Staff

Scope: Hospital Management System

Level: User Goal

Primary Actor: Admin

Stakeholders and Interests:

- **Admin:** Wants to efficiently manage the staffing of different departments and ensure optimal resource allocation.
- **Doctors/Nurses/Technicians:** Seek clarity in roles, responsibilities, and schedules to perform their tasks effectively.
- **Patients:** Expect timely and quality care from adequately staffed departments.
- **Administrative Staff:** Needs a structured approach to staffing to ensure smooth operations across departments.

Pre-Conditions: Admin is identified and authenticated.

Post-Conditions: Staff records are updated. Scheduling changes are reflected in the system.

Main Success Scenario:

Actions	System
1. Admin logs into the system.	System verifies admin credentials.
2. Admin accesses the staff management members and their roles.	System displays the list of current staff module.
3. Admin selects a department to manage.	System displays staff details for the selected department.
4. Admin adds a new staff member or existing staff details.	System validates and updates staff updates Information.
5. Admin sets staff schedules and availability.	System saves and confirms schedule changes.
6. Admin reviews staff performance metrics.	System generates reports on staff performance and attendance.

Alternate Scenarios:

Actions	System
1. Admin encounters an error while the updating staff details.	System prompts admin to retry operation.
2. Staff member is already assigned to a different department.	System alerts admin of the conflict.
3. Admin wishes to remove a staff member.	System prompts for confirmation before deletion.

Use Case No. 10: Handle Sale of Medicine

Scope: Hospital Management System

Level: User Goal

Primary Actor: Pharmacist

Stakeholders and Interests:

- **Pharmacist:** Wants to efficiently manage sales transactions and inventory.
- **Patients:** Expect accurate billing and availability of prescribed medications.
- **Doctors:** Require assurance that patients receive necessary medications.
- **Admin:** Needs accurate reporting for inventory and sales tracking.

Pre-Conditions: Pharmacist is identified and authenticated.

Post-Conditions: Sales transactions are recorded, and inventory is updated.

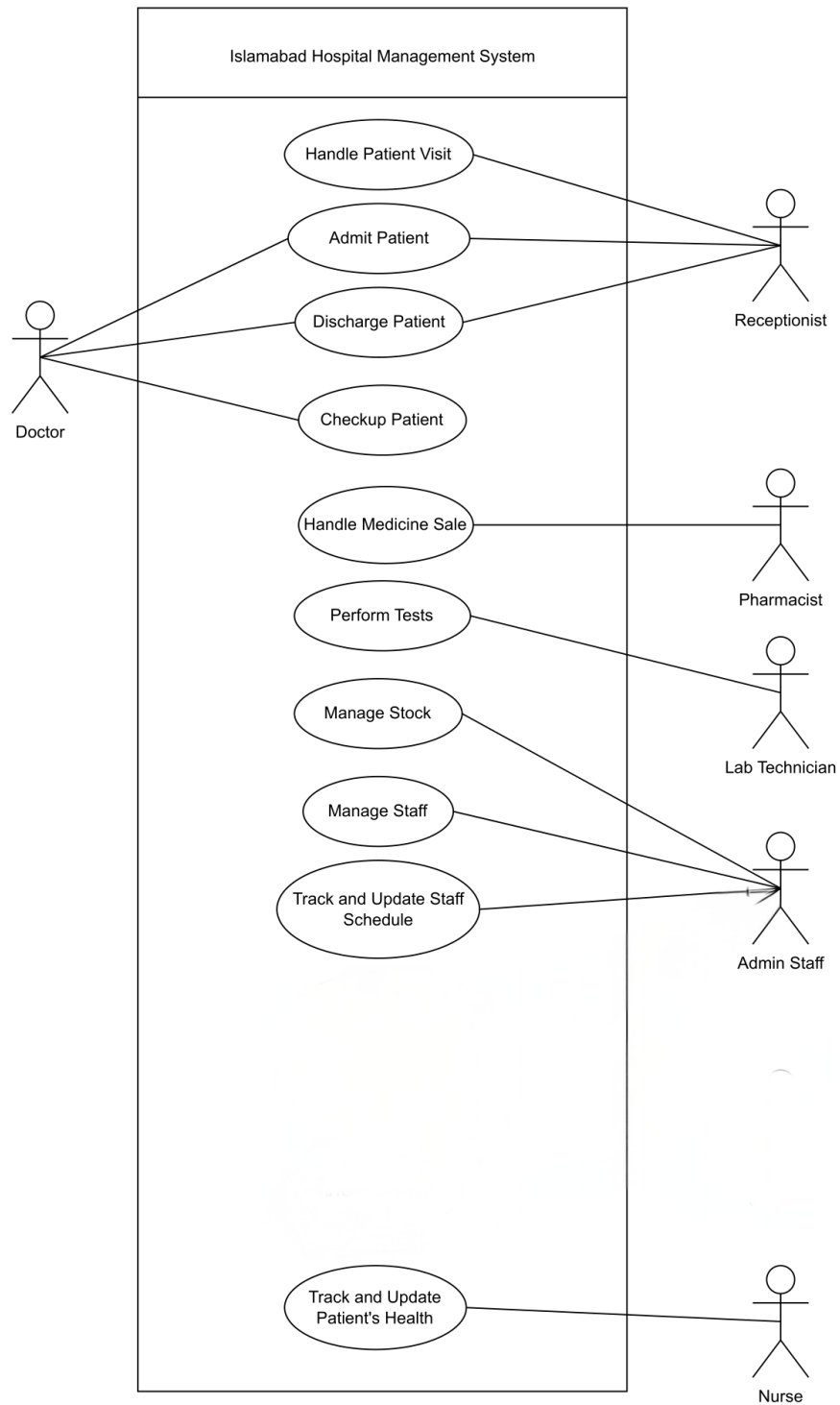
Main Success Scenario:

Actions	System
1. Patient presents a prescription to the pharmacist	
2. Pharmacist searches for the medicine	System presents a list of medicines
3. Pharmacist selects a medicine specifies quantity and adds. < Steps 2,3 are performed until pharmacist confirms the sale >	System presents a running total
4. Pharmacist confirms the sale and provides medication.	System processes the transaction and updates inventory.
5. Pharmacist issues a receipt to the patient.	System generates the sales receipt.

Alternate Scenarios:

Actions	System
Medication is out of stock.	System alerts the Admin and suggests alternatives.
Patient has questions about medication dosage.	System provides dosage information based on the prescription.

2.5 Use Case Diagram

**Team Members**

22i-0941 Muhammad Sarmad

22i-0942 M.Umar Hassan

22i-1220 Muhammad Huzaifa

3. Other Nonfunctional Requirements

3.1 Performance Requirements

Performance requirements for the Hospital Management System (HMS) must ensure that the system operates efficiently under various circumstances. The system should handle up to 100 concurrent users without degradation in response time, maintaining an average response time of less than 2 seconds for user queries. For real-time operations, such as patient check-ins and updates, the system must process requests within 1 second to ensure smooth workflow and minimize patient wait times. These requirements are critical for maintaining operational efficiency and enhancing user satisfaction.

3.2 Safety Requirements

Safety requirements focus on preventing loss, damage, or harm during the use of the HMS. The system must implement safeguards such as data encryption and secure access controls to protect sensitive patient information. Additionally, it should include automated alerts for critical situations, such as low stock levels of essential medications. Compliance with healthcare regulations like HIPAA is mandatory to ensure patient data safety and confidentiality. The system must also undergo safety certifications to validate its reliability in a healthcare environment.

3.3 Security Requirements

Security requirements are essential to protect user data and ensure privacy within the HMS. The system must enforce user identity authentication through multi-factor authentication (MFA) for all users accessing sensitive information. Regular security audits should be conducted to identify vulnerabilities, and compliance with external regulations such as GDPR must be maintained. Certifications such as ISO/IEC 27001 for information security management should be pursued to demonstrate commitment to data protection.

3.4 Software Quality Attributes

Quality attributes for the HMS include:

- **Reliability:** The system should maintain 99.9% uptime.
- **Usability:** User satisfaction ratings should exceed 85% based on usability testing.
- **Maintainability:** Code should be modular, allowing updates without significant downtime.
- **Interoperability:** The system must integrate seamlessly with existing hospital software and hardware.
- **Testability:** Automated tests should cover at least 80% of the code to ensure functionality before deployment.

These attributes guide developers in prioritizing features that enhance user experience and operational efficiency.

3.5 Business Rules

Business rules dictate operational principles within the HMS, such as:

- Only authorized personnel (e.g., doctors and administrative staff) can access patient records.
- Patients must provide valid identification (e.g., CNIC) for registration and check-in.
- Staff scheduling must adhere to labor regulations.

These rules imply functional requirements that enforce compliance and operational integrity.

3.6 Operating Environment

The HMS will operate on a standard hardware platform equipped with at least:

- Processor: Intel i5 or equivalent.
- RAM: Minimum of 8 GB.
- Operating System: Windows Server 2019 or Linux-based systems.

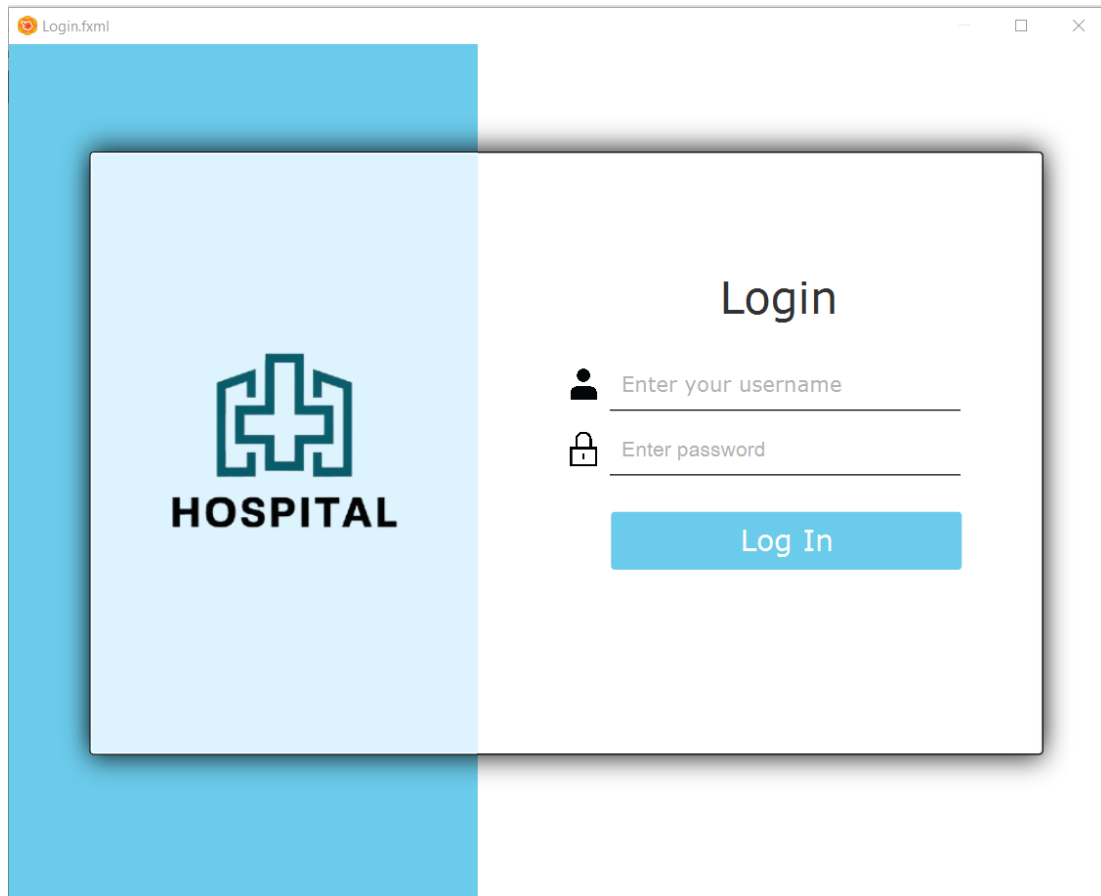
The software must coexist with existing applications such as electronic health records (EHR) systems and laboratory management software, ensuring compatibility and efficient data exchange.

3.7 User Interfaces

The user interface (UI) for the Hospital Management System (HMS) is designed to ensure ease of navigation and functionality for various roles within the hospital, including doctors, lab technicians, and administrative staff. The layout consists of a vertical button panel on the left side of the screen, providing quick access to essential functions specific to each user role.

UI Structure:

- **Login Page:**
All users must first pass through the login page, ensuring secure access to the system.
- **Main Interface:**
 - **Button Panel (Left Side):**
Each user role has a dedicated set of buttons that lead to their respective functionalities. For example: Doctor View has checkup, generate admit/discharge request.
 - **Logout Button:**
Located at the bottom of the button panel for easy access, allowing users to securely exit the system.



Login Page

Receptionist_menu.fxml

— □ ×

Receptionist Profile

Profile

Username:

Name:

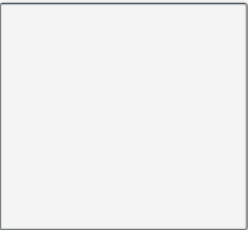
Age:

Experience:

Phone No. :

Gender:

Address:



Home

Register Patient

Doctor Visit

Lab Visit

Admit Patient

Discharge Patient

Pay Dues

Log Out

Receptionist Menu

The diagram illustrates the following entities and their attributes:

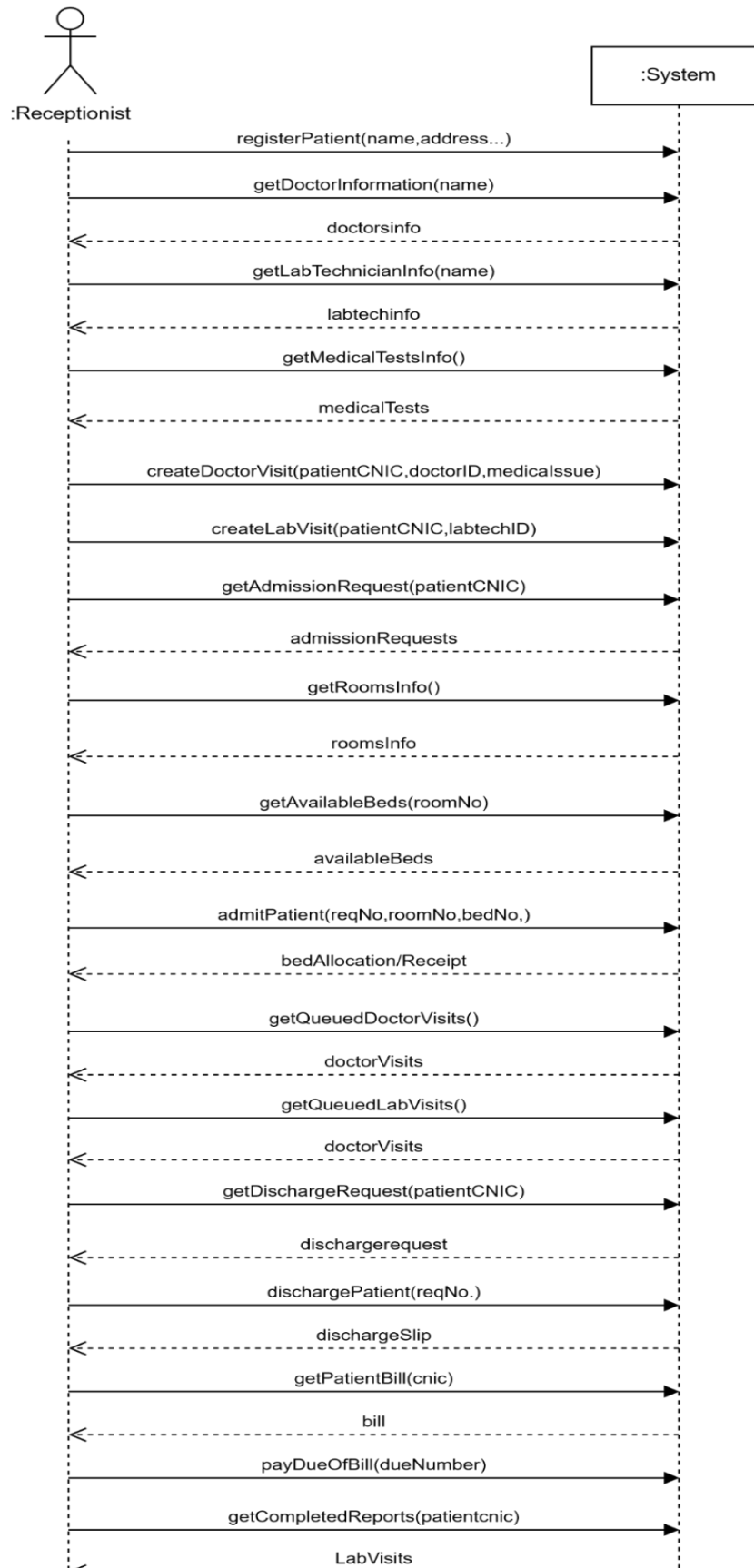
- Hospital**: Name, Address
- Catalog**: (No attributes listed)
- Bed**: Room Number, Rent, BedNumber
- Receptionist**: ID
- Doctor**: ID
- Patient**: ID, Name, Address
- Nurse**: ID
- Pharmacist**: ID
- Surgeon**: ID
- Medicine**: MedicineNo, Description, Price, Medicine Name, Quantity
- MedicineSale**: (No attributes listed)
- SaleLineMedicine**: Quantity, Date/Time, /SubTotal
- Payment**: Amount
- Discharge Slip/Transaction**: Date/Time
- InPatientBill**: field: type, field: type, Amount, Description
- Dues**: (No attributes listed)
- Bed Allocation Transaction**: Date/Time
- Doctor Queue**: (No attributes listed)
- Lab Queue**: (No attributes listed)
- Doctor Visit**: Visit Number, Date/Time
- Lab Visit**: Visit Number, Date/Time, Sample Number
- Medical Report**: Detail
- Medical Test Description**: Test ID, Test Name, Price
- LT Register**: (No attributes listed)
- Lab Technician**: ID
- Operation Request**: Date/Time, RequestNumber
- Admin**: ID
- Ambulance**: PlateNumber
- AmbulanceVisit**: Location, Date/Time
- Operation**: Date/Time, Operation Number
- Post Operation Care**: Detail
- Operation Room**: Room Number
- Schedule/Duty**: Days, Time
- Admission Request**: Date/Time
- Discharge Request**: Date/Time

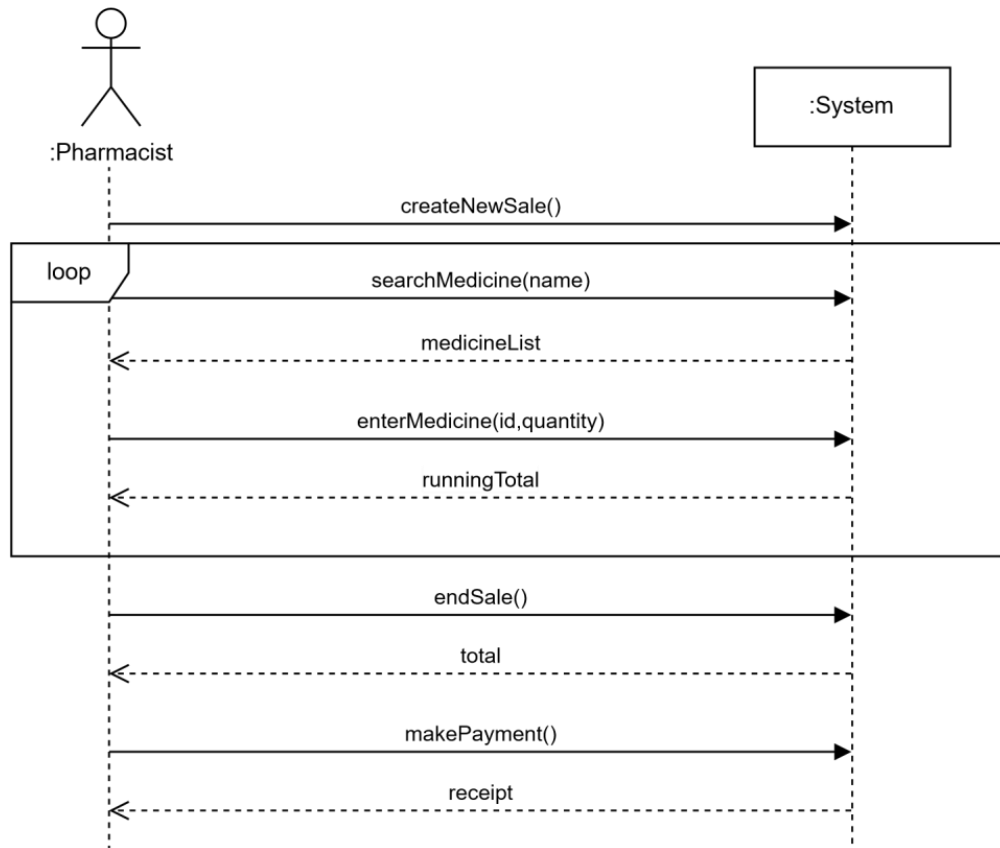
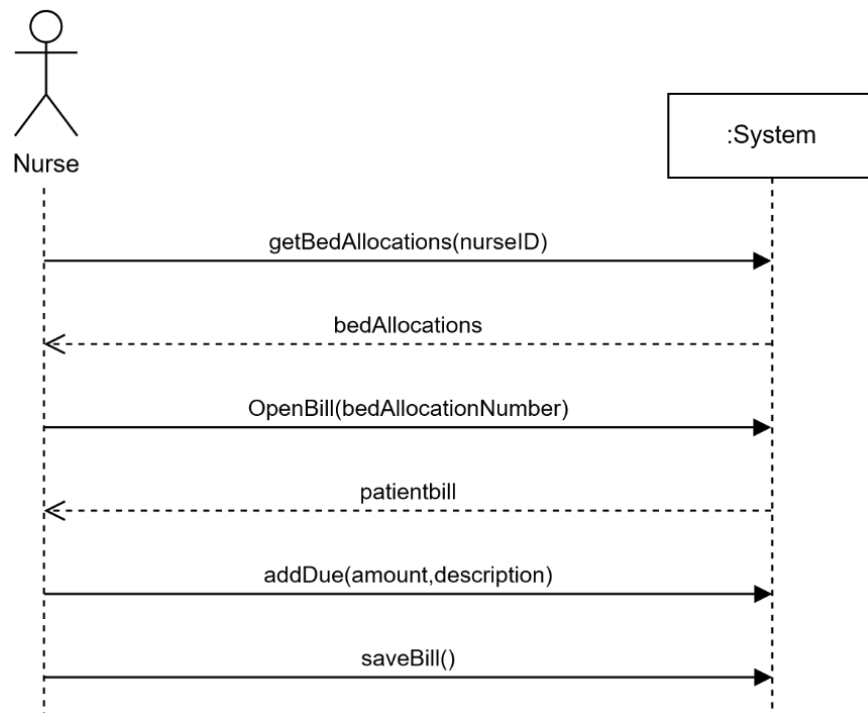
Key relationships and cardinalities:

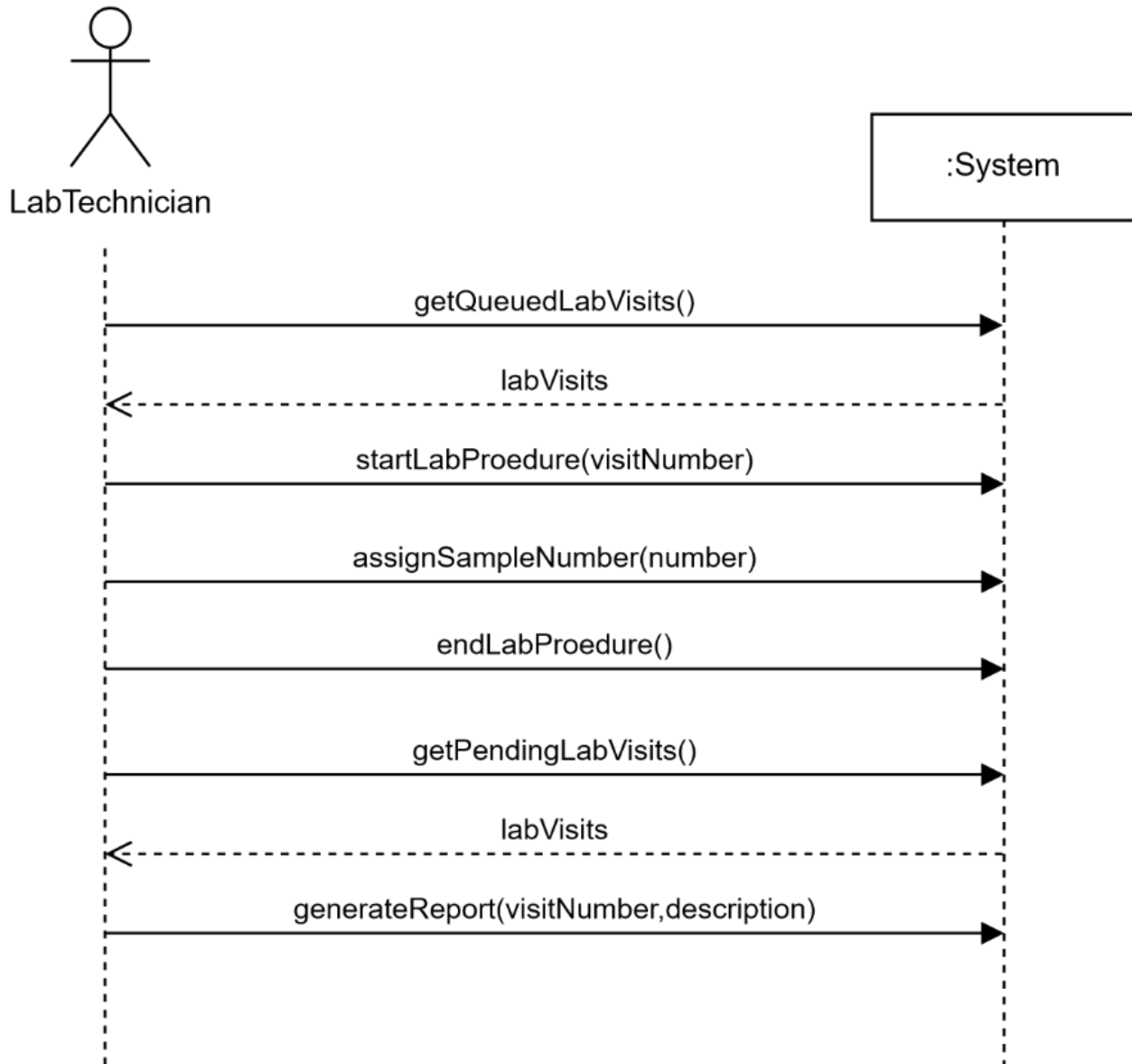
- Hospital** (1) *Stores* **Catalog** (1)
- Hospital** (1) *Used By* **Receptionist** (1)
- Hospital** (1) *includes* **Patient Room** (1)
- Patient Room** (1) *has* **Bed** (1)
- Receptionist** (1) *Generates* **Discharge Slip/Transaction** (1)
- Receptionist** (1) *Generates* **InPatientBill** (1)
- Receptionist** (1) *Handles* **Bed Allocation Transaction** (1)
- Receptionist** (1) *Handles* **Doctor Queue** (1)
- Receptionist** (1) *Handles* **Lab Queue** (1)
- Receptionist** (1) *Creates* **Patient Information** (1)
- Patient Information** (1) *Described by* **Patient** (1)
- Patient** (1) *Pays* **Doctor** (1)
- Doctor** (1) *Works On* **Doctor Register** (1)
- Doctor** (1) *initiates* **Admission Request** (1)
- Doctor** (1) *initiates* **Discharge Request** (1)
- Doctor** (1) *initiates* **Operation Request** (1)
- Doctor** (1) *initiates* **Surgeon** (1)
- Doctor** (1) *initiates* **Operation** (1)
- Doctor** (1) *initiates* **Pharmacist** (1)
- Doctor** (1) *initiates* **Pharmacist Register** (1)
- Doctor** (1) *initiates* **Medicine** (1)
- Doctor** (1) *initiates* **MedicineSale** (1)
- Doctor** (1) *initiates* **SaleLineMedicine** (1)
- Doctor** (1) *initiates* **Payment** (1)
- Doctor** (1) *initiates* **Ambulance** (1)
- Doctor** (1) *initiates* **AmbulanceVisit** (1)
- Doctor** (1) *initiates* **Operation Room** (1)
- Doctor** (1) *initiates* **Post Operation Care** (1)
- Doctor** (1) *initiates* **Schedule/Duty** (1)
- Doctor** (1) *initiates* **Admission Request** (1)
- Doctor** (1) *initiates* **Discharge Request** (1)
- Doctor** (1) *initiates* **Operation Request** (1)
- Doctor** (1) *initiates* **Surgeon** (1)
- Doctor** (1) *initiates* **Operation** (1)
- Doctor** (1) *initiates* **Pharmacist** (1)
- Doctor** (1) *initiates* **Pharmacist Register** (1)
- Doctor** (1) *initiates* **Medicine** (1)
- Doctor** (1) *initiates* **MedicineSale** (1)
- Doctor** (1) *initiates* **SaleLineMedicine** (1)
- Doctor** (1) *initiates* **Payment** (1)
- Doctor** (1) *initiates* **Ambulance** (1)
- Doctor** (1) *initiates* **AmbulanceVisit** (1)
- Doctor** (1) *initiates* **Operation Room** (1)
- Doctor** (1) *initiates* **Post Operation Care** (1)
- Doctor** (1) *initiates* **Schedule/Duty** (1)
- Doctor** (1) *initiates* **Admission Request** (1)
- Doctor** (1) *initiates* **Discharge Request** (1)
- Doctor** (1) *initiates* **Operation Request** (1)
- Doctor** (1) *initiates* **Surgeon** (1)
- Doctor** (1) *initiates* **Operation** (1)
- Doctor** (1) *initiates* **Pharmacist** (1)
- Doctor** (1) *initiates* **Pharmacist Register** (1)
- Doctor** (1) *initiates* **Medicine** (1)
- Doctor** (1) *initiates* **MedicineSale** (1)
- Doctor** (1) *initiates* **SaleLineMedicine** (1)
- Doctor** (1) *initiates* **Payment** (1)
- Doctor** (1) *initiates* **Ambulance** (1)
- Doctor** (1) *initiates* **AmbulanceVisit** (1)
- Doctor** (1) *initiates* **Operation Room** (1)
- Doctor** (1) *initiates* **Post Operation Care** (1)
- Doctor** (1) *initiates* **Schedule/Duty** (1)
- Doctor** (1) *initiates* **Admission Request** (1)
- Doctor** (1) *initiates* **Discharge Request** (1)
- Doctor** (1) *initiates* **Operation Request** (1)
- Doctor** (1) *initiates* **Surgeon** (1)
- Doctor** (1) *initiates* **Operation** (1)
- Doctor** (1) *initiates* **Pharmacist** (1)
- Doctor** (1) *initiates* **Pharmacist Register** (1)
- Doctor** (1) *initiates* **Medicine** (1)
- Doctor** (1) *initiates* **MedicineSale** (1)
- Doctor** (1) *initiates* **SaleLineMedicine** (1)
- Doctor** (1) *initiates* **Payment** (1)
- Doctor** (1) *initiates* **Ambulance** (1)
- Doctor** (1) *initiates* **AmbulanceVisit** (1)
- Doctor** (1) *initiates* **Operation Room** (1)
- Doctor** (1) *initiates* **Post Operation Care** (1)
- Doctor** (1) *initiates* **Schedule/Duty** (1)
- Doctor** (1) *initiates* **Admission Request** (1)
- Doctor** (1) *initiates* **Discharge Request** (1)
- Doctor** (1) *initiates* **Operation Request** (1)
- Doctor** (1) *initiates* **Surgeon** (1)
- Doctor** (1) *initiates* **Operation** (1)
- Doctor** (1) *initiates* **Pharmacist** (1)
- Doctor** (1) *initiates* **Pharmacist Register** (1)
- Doctor** (1) *initiates* **Medicine** (1)
- Doctor** (1) *initiates* **MedicineSale** (1)
- Doctor** (1) *initiates* **SaleLineMedicine** (1)
- Doctor** (1) *initiates* **Payment** (1)
- Doctor** (1) *initiates* **Ambulance** (1)
- Doctor** (1) *initiates* **AmbulanceVisit** (1)
- Doctor** (1) *initiates* **Operation Room** (1)
- Doctor** (1) *initiates* **Post Operation Care** (1)
- Doctor** (1) *initiates* **Schedule/Duty** (1)
- Doctor** (1) *initiates* **Admission Request** (1)
- Doctor** (1) *initiates* **Discharge Request** (1)
- Doctor** (1) *initiates* **Operation Request** (1)
- Doctor** (1) *initiates* **Surgeon** (1)
- Doctor** (1) *initiates* **Operation** (1)
- Doctor** (1) *initiates* **Pharmacist** (1)
- Doctor** (1) *initiates* **Pharmacist Register** (1)
- Doctor** (1) *initiates* **Medicine** (1)

5. System Sequence Diagram

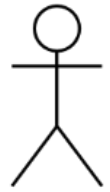
Receptionist



Pharmacist:**Nurse:**

Lab Technician:

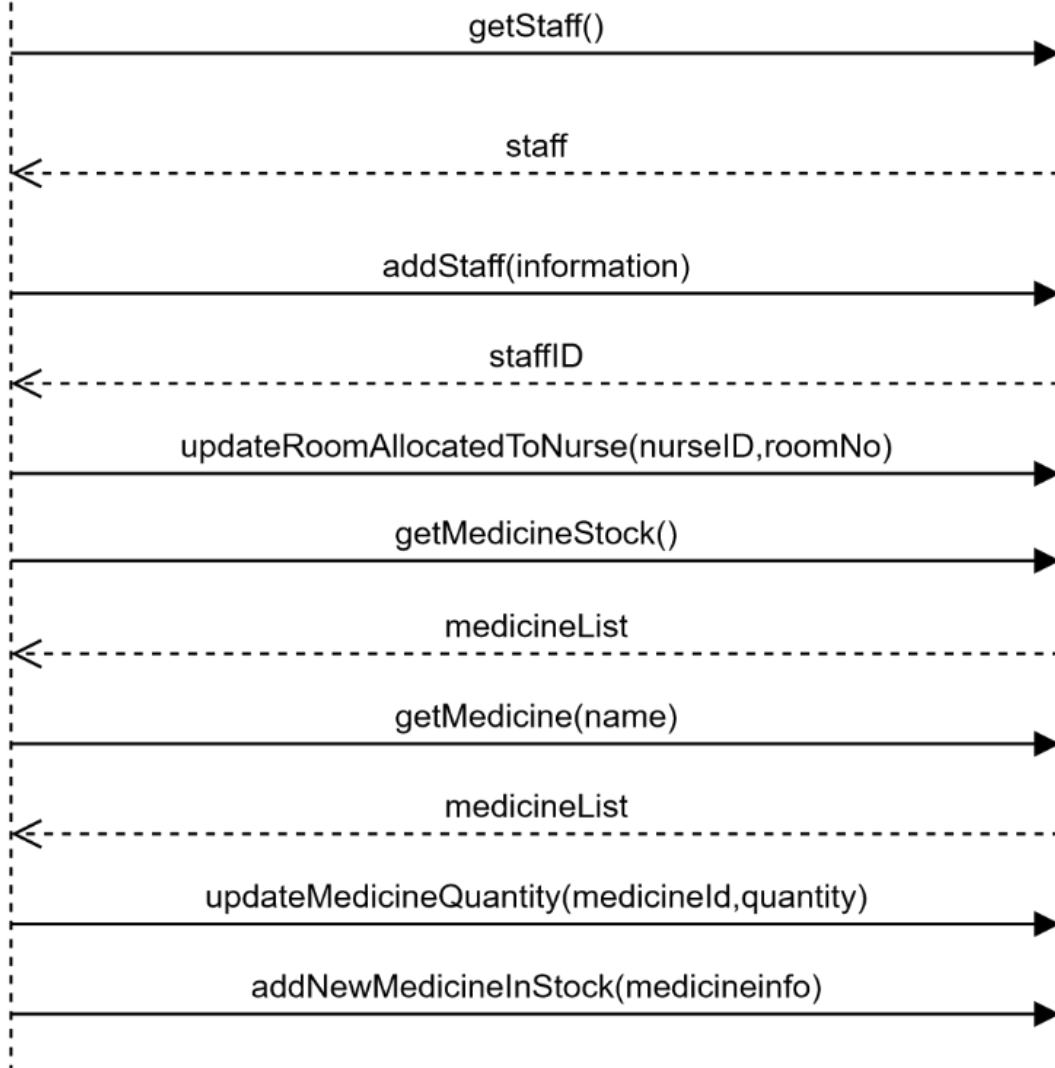
Admin:

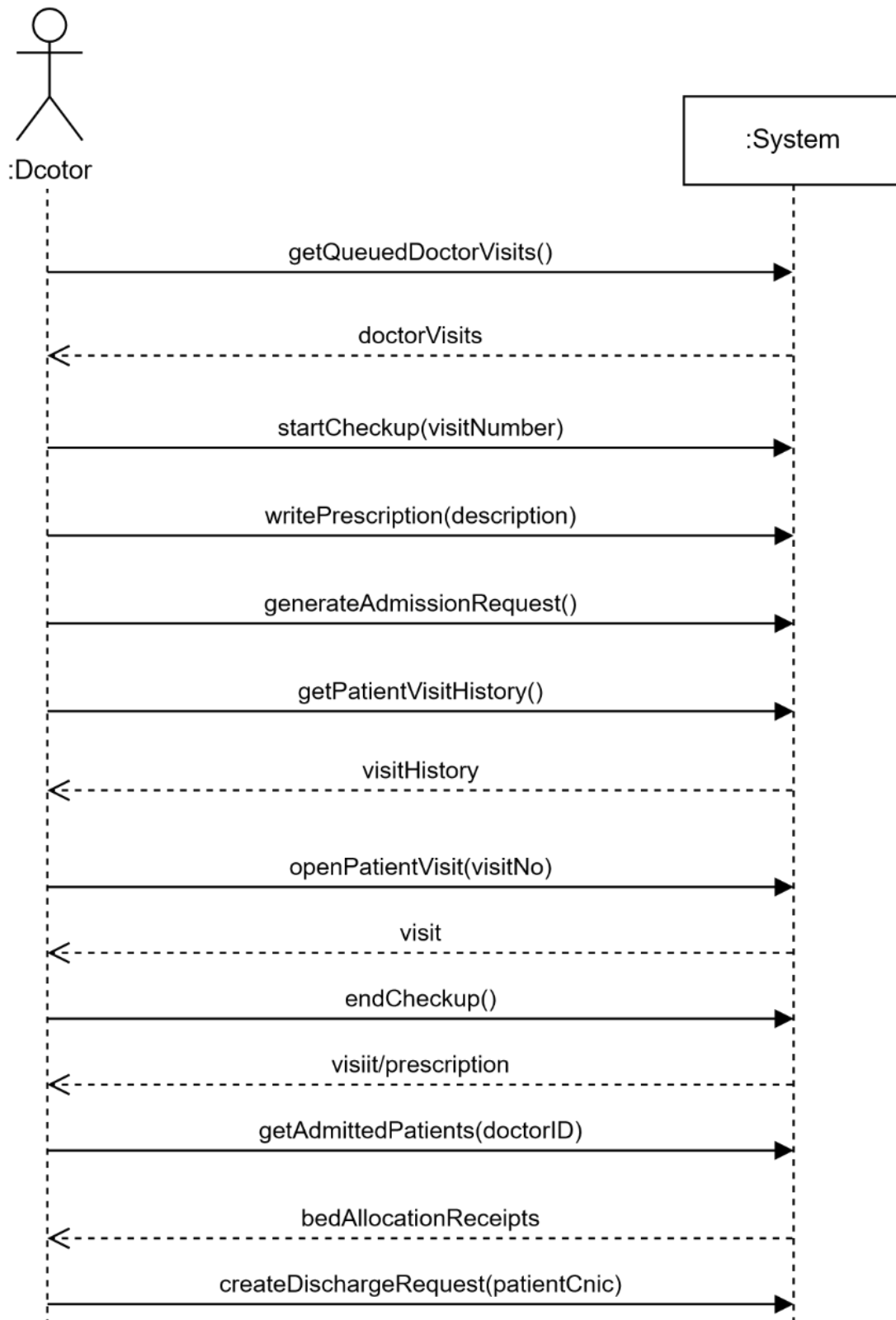


Admin

Note: We used a general term 'staff' for different types of staff for ease

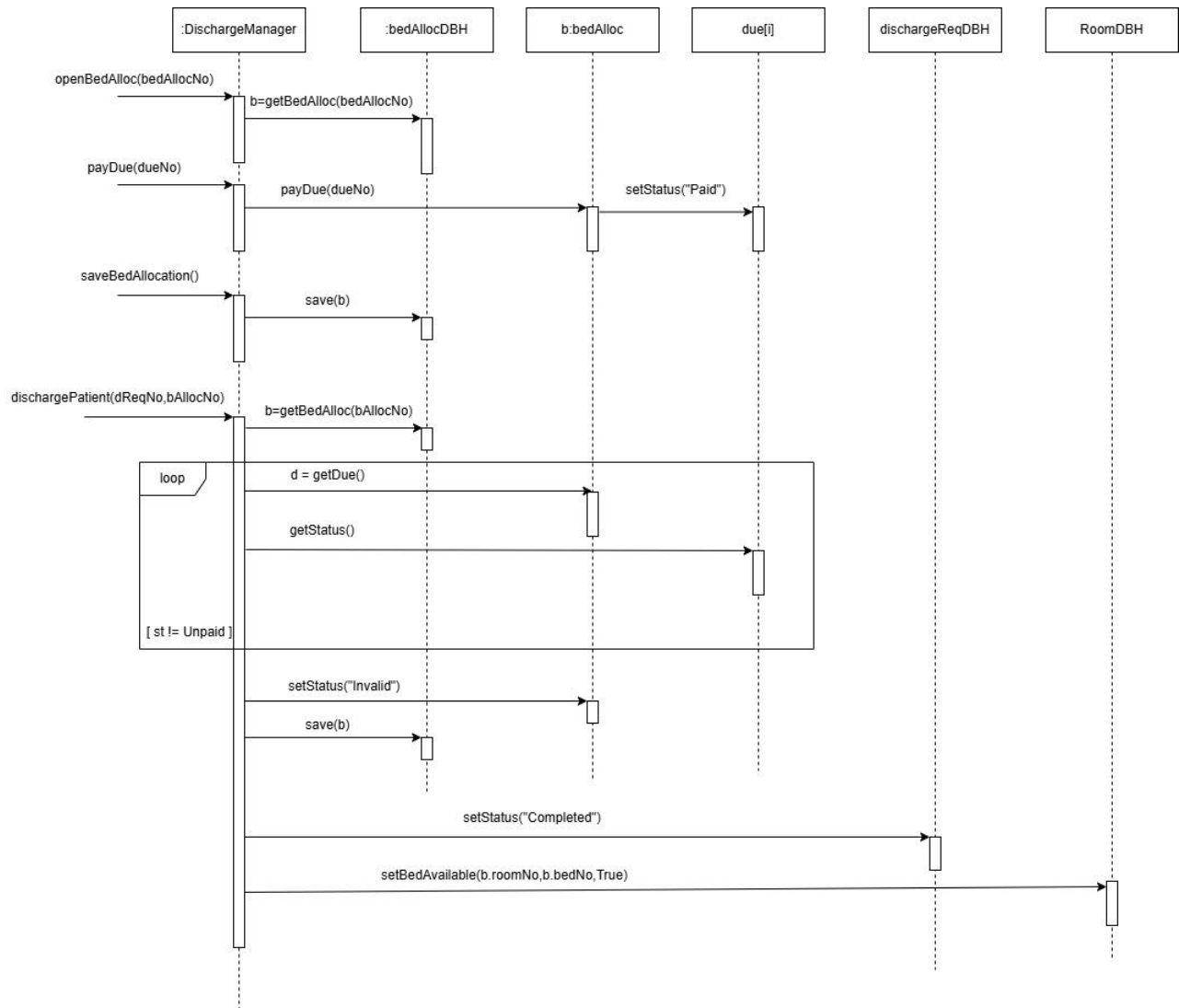
:System



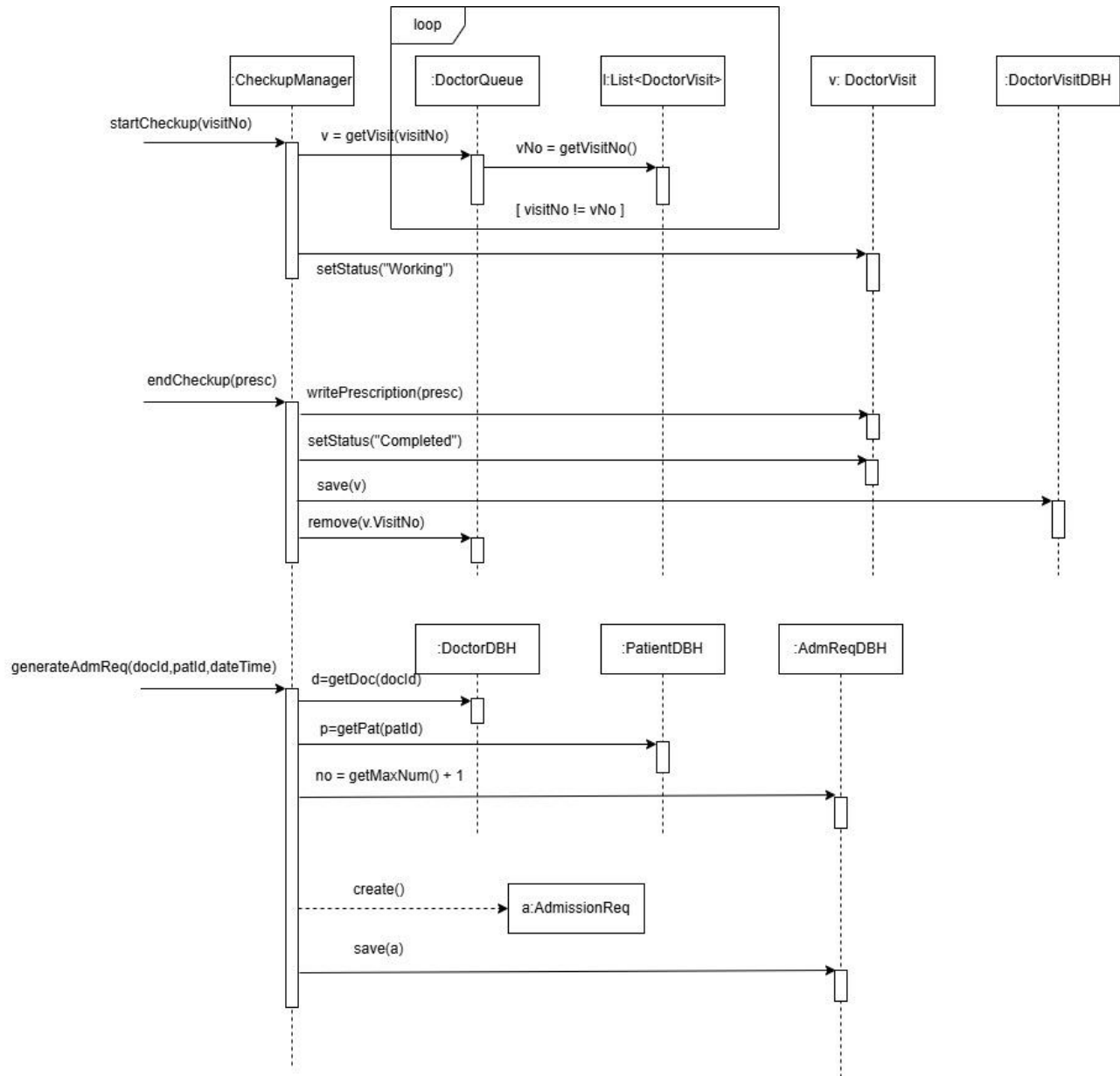
Doctor:

6. Sequence Diagram

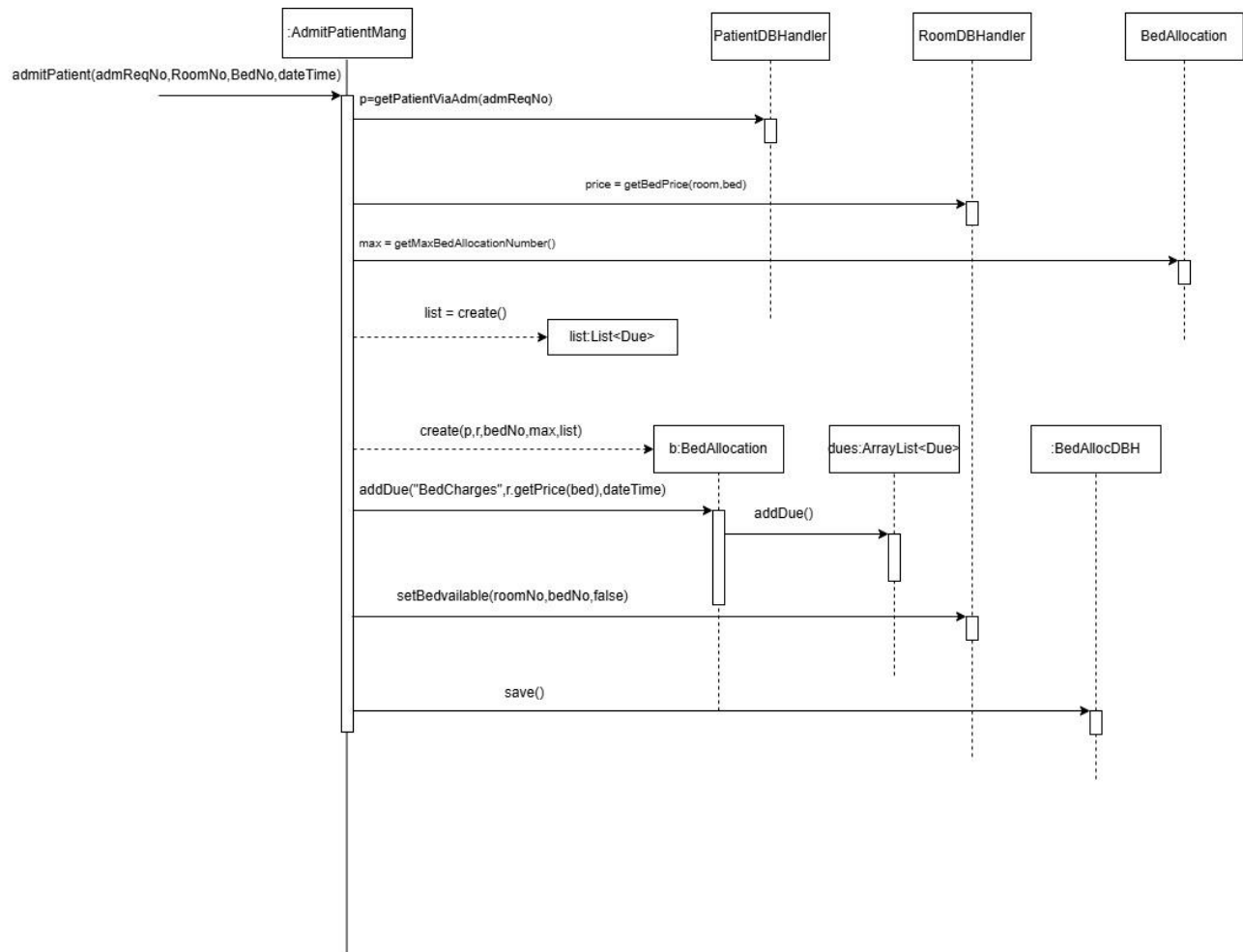
1. Discharge Patient:



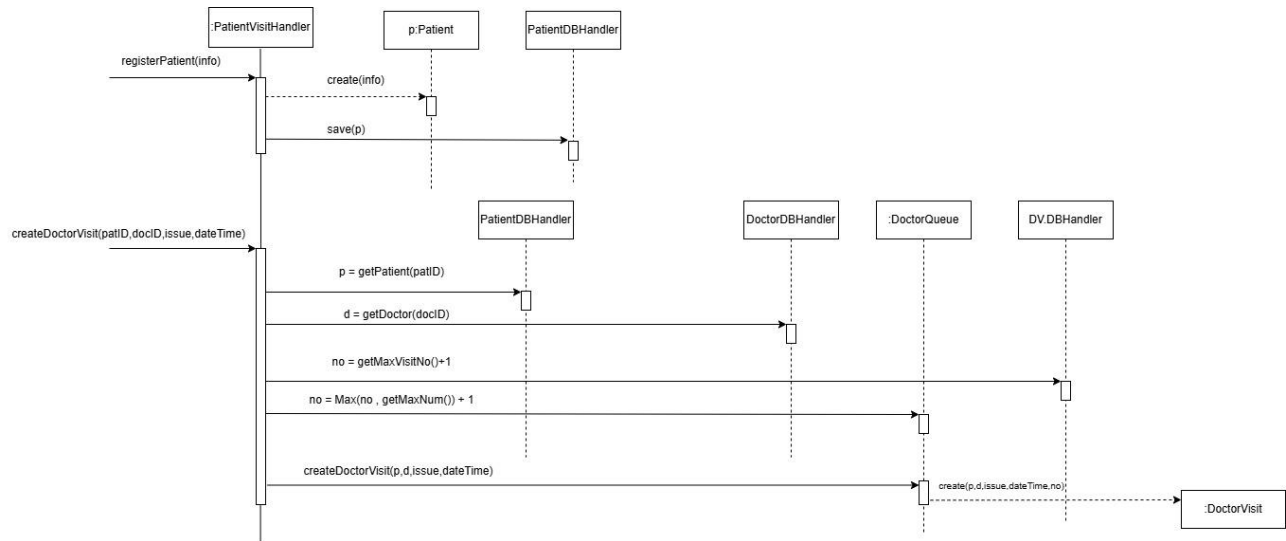
2. Doctor Check up



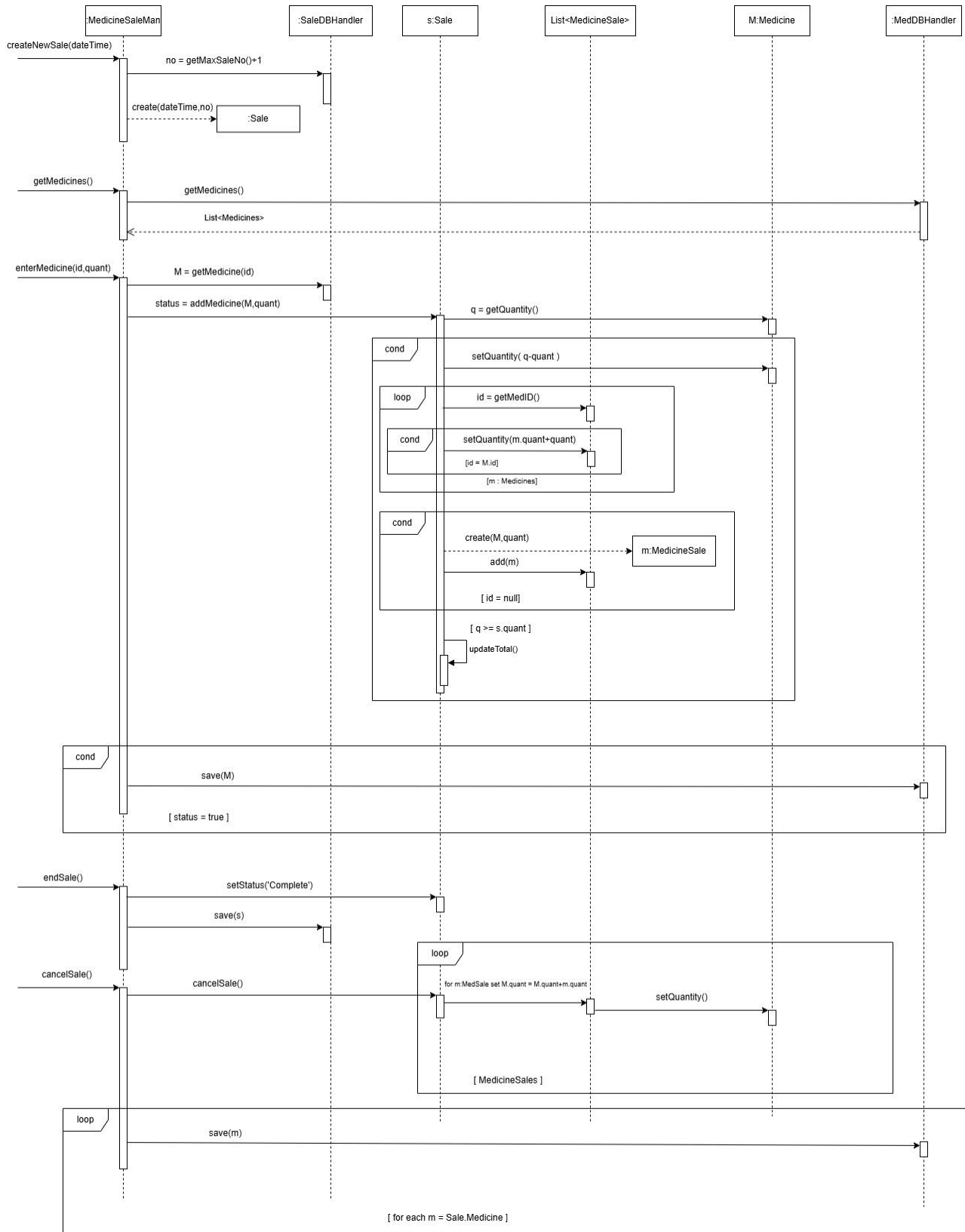
3. Admit Patient



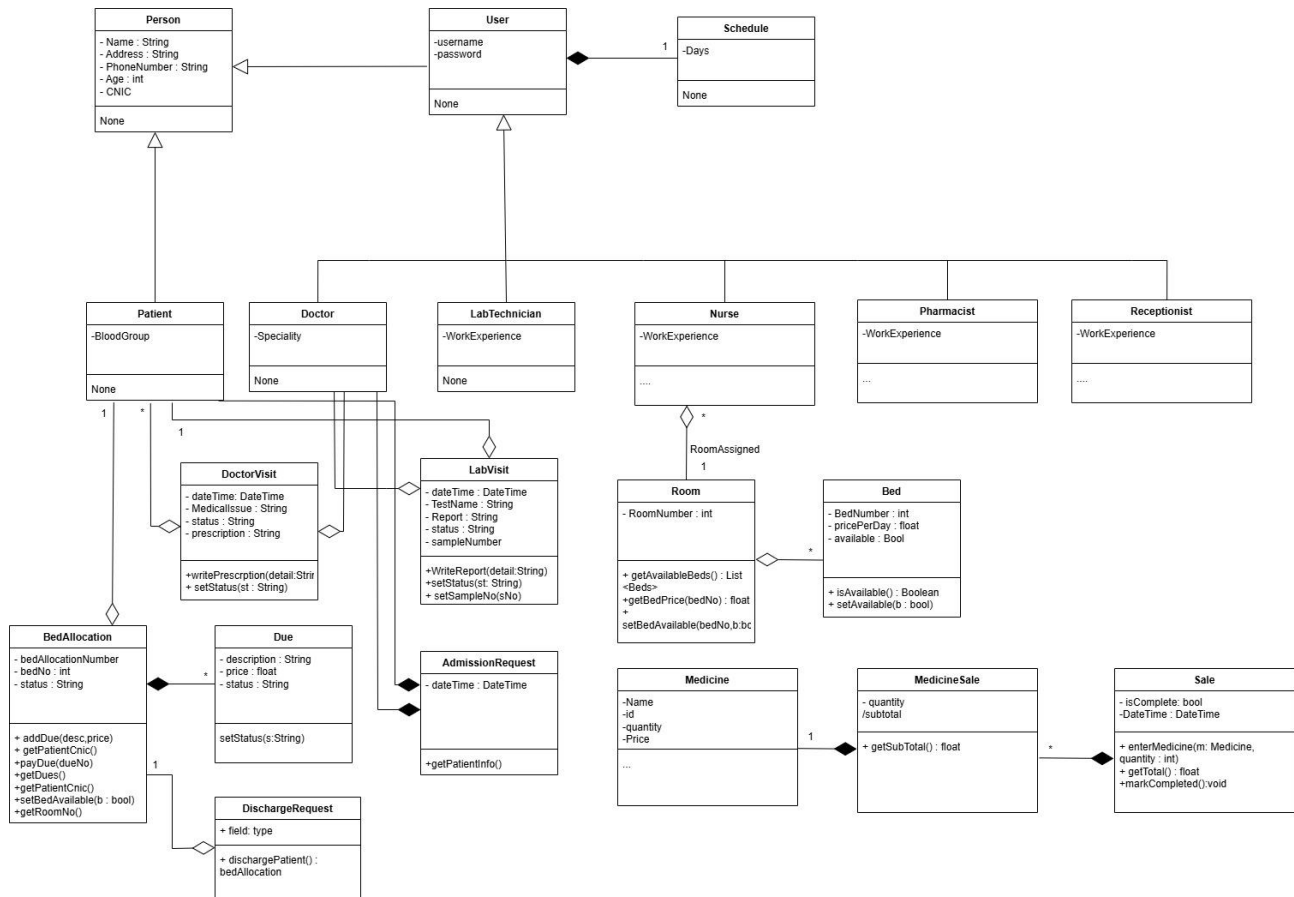
4. Patient Visit



5. Pharmacist

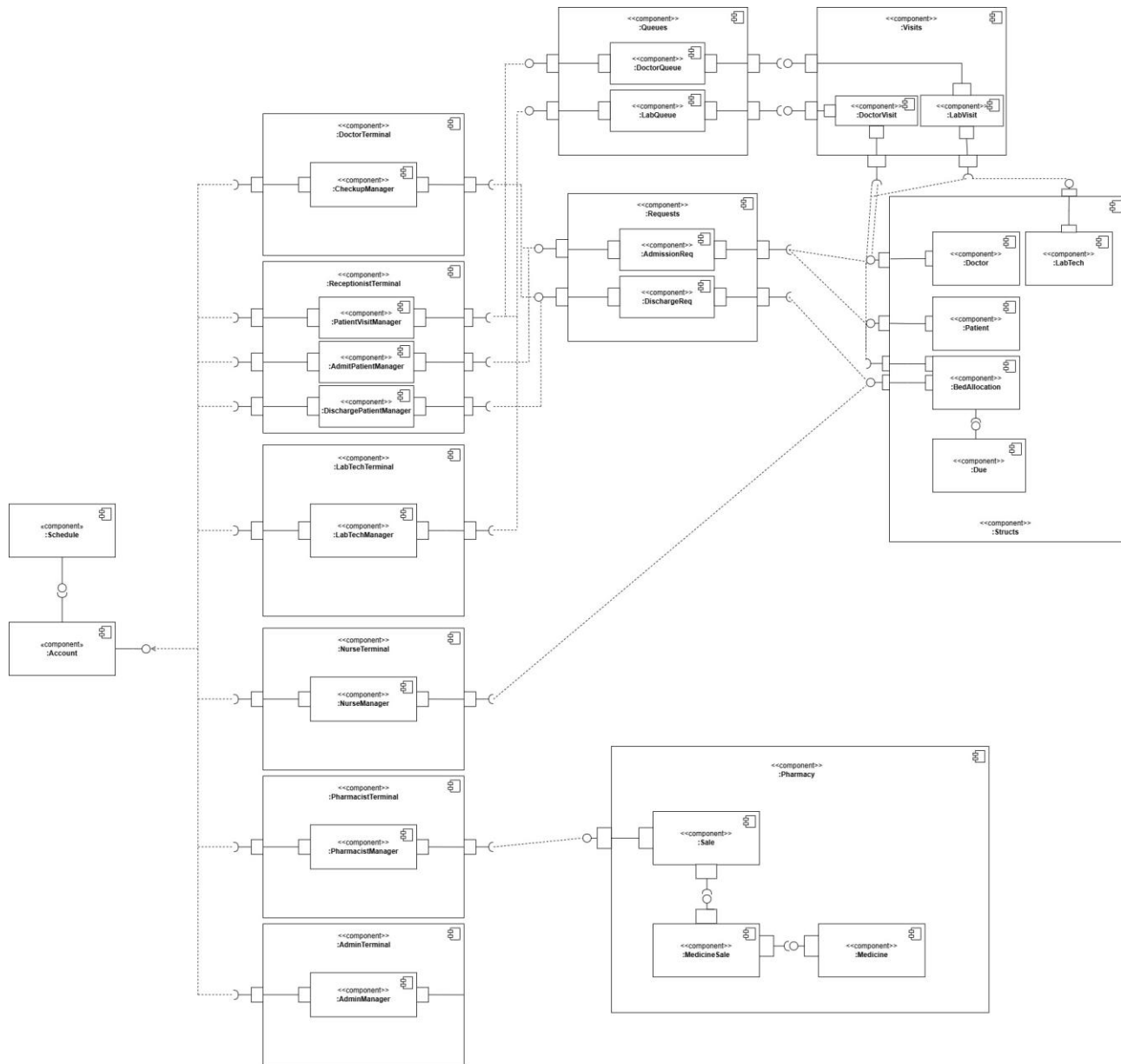


7. Class Diagram

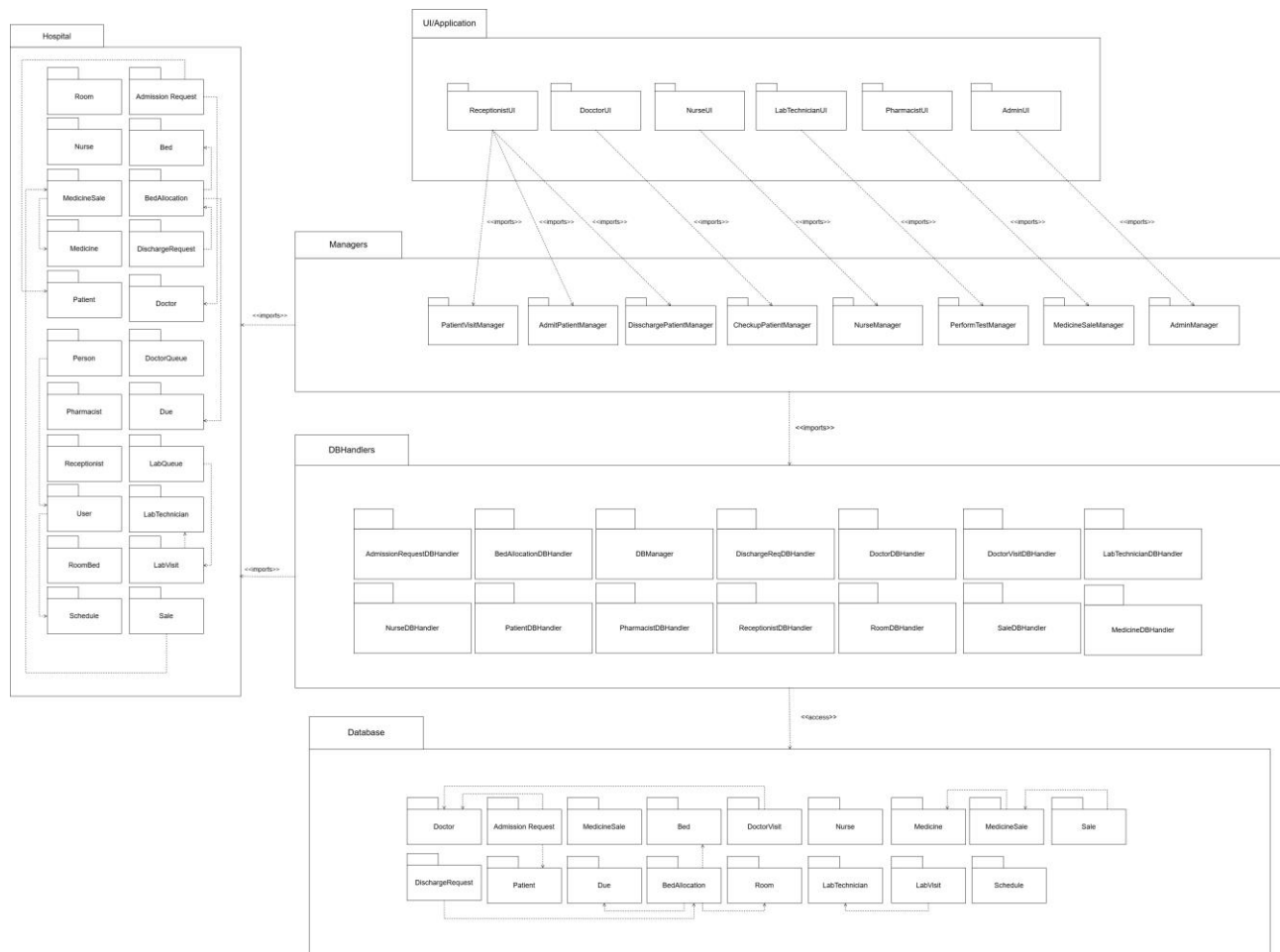


Note: Extended Design Class Diagram is at the last page

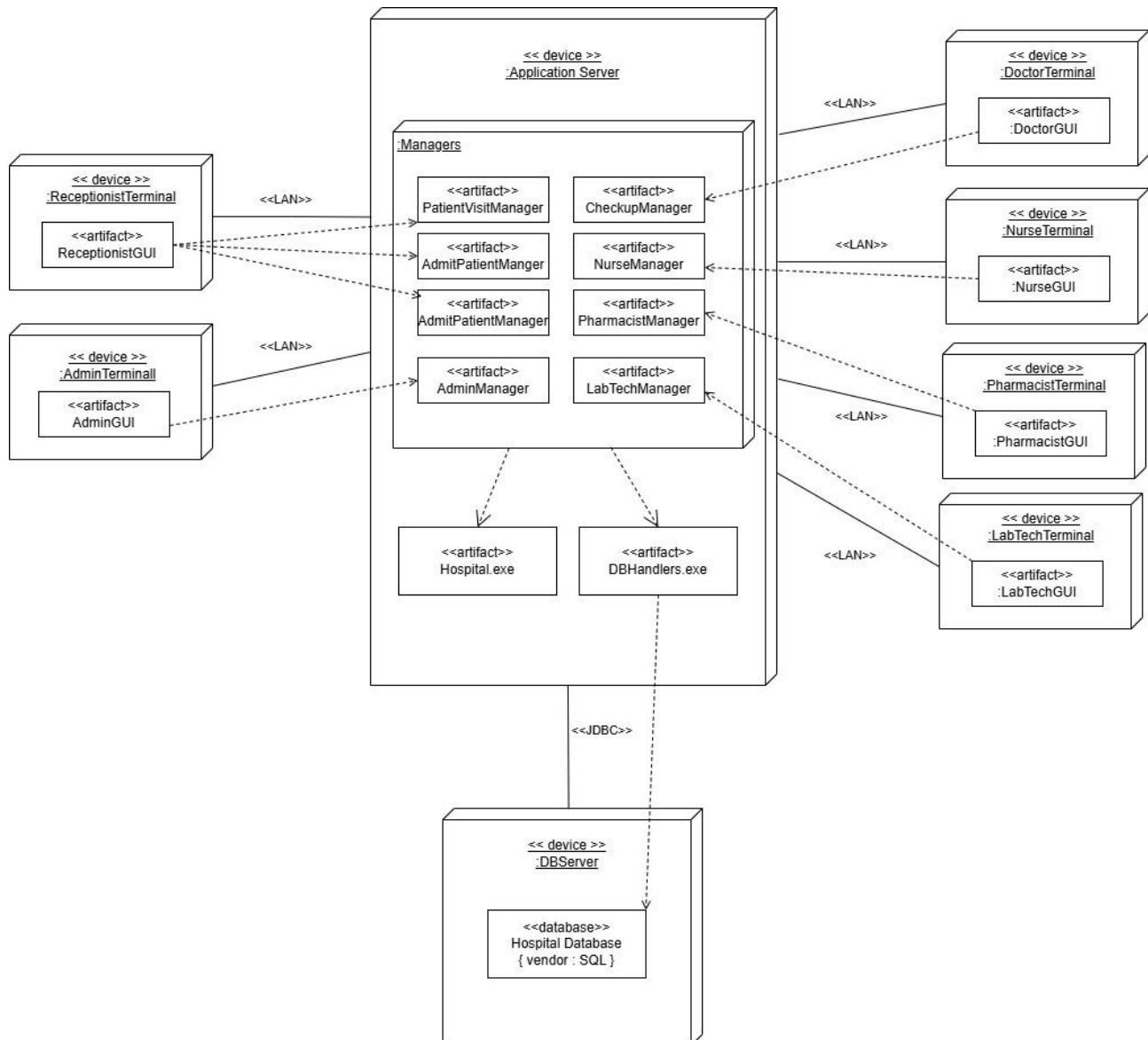
8. Component Diagram



9. Package Diagram



10. Deployment Diagram



11. Extended Class Diagram

