Software Design & Architecture

Project Report

Topic: Hospital Management System

Submitted To: Ms. Nimra Waqar

Submitted By:

Name	Sap Id	Semester	Email
Falak Irfan	50029	BSSE-6	50029@students.riphah.edu.pk
Kubra Zareen	51111	BSSE-6	51111@students.riphah.edu.pk
Arusa Nadeem	44880	BSSE-6	44880@students.riphah.edu.pk

Faculty of Computing

Riphah International University, GGC Islamabad

Fall 2025

Table of Content

Artifact-1: Features and Software Requirements

Project Title: Hospital Management System

1. Introduction

Healthcare systems today operate in highly dynamic and complex environments where accuracy, speed, and reliable information flow are essential for delivering quality medical services. As hospitals grow in size, the volume of data they handle increases exponentially—ranging from patient registration records and diagnostic reports to staff information, billing transactions, room allocations, and administrative decisions. When these processes rely heavily on manual paperwork, physical files, and fragmented systems, inefficiencies rapidly emerge. These inefficiencies lead to delayed treatments, difficulty in retrieving information, miscommunication among departments, and overall reduced quality of patient care.

A Hospital Management System (HMS) plays a transformative role in overcoming these limitations by integrating hospital operations into a unified digital platform. Through automation and centralization, an HMS ensures that all departments—such as outpatient services, inpatient services, pharmacy, laboratory, administration, and finance—work together smoothly. This not only streamlines day-to-day processes but also eliminates redundancies, minimizes human error, and provides real-time access to critical patient and hospital data.

The primary aim of this project is to design and develop a comprehensive Hospital Management System that improves the efficiency and reliability of hospital operations. The system provides functionalities such as patient registration and management, doctor information handling, appointment and admission management, laboratory services, room and ward allocation, staff record maintenance, billing processes, and report handling. By offering these core features, the system enhances decision-making for administrators, improves workflow coordination among healthcare staff, and ensures timely and effective services for patients.

In addition to implementation, this project focuses on creating high-quality software architectural models that highlight the structure, behavior, and interrelationships of the system components. System-level diagrams such as the Use Case Diagram, Data Flow Diagrams (Level 0), Component Diagram, Deployment Diagram, Package Diagram, and Sequence Diagram are developed to provide a clear visualization of system requirements, interactions, and architecture. These diagrams help ensure that the system design is systematic, modular, scalable, and aligned with best practices in software engineering.

Overall, the Hospital Management System developed in this project aims to bridge the gap between traditional manual hospital processes and modern digital healthcare management. By integrating automation, efficient data handling, and structured software design principles, the system ultimately supports hospitals in delivering better patient care, improving operational efficiency, strengthening data security, and enabling informed decision-making across all departments.

2. Existing Features

ID	Feature Name	Description
F-1	Manage Doctor Information	This feature enables the admin to register, retrieve, update, and delete doctor details, including name, specialty, and contact information. It ensures that all changes whether registration, updates, or deletions are saved.
F-2	Manage Patient Information	This feature allows the admin to register, retrieve, update, and delete patient details, including personal information, medical history, and contact data. It ensures that all changes whether registration, updates, or deletions are saved.
F-3	Laboratory Services	This feature tracks patient laboratory services, including blood tests, imaging, and diagnostic reports. It also records the associated charges for each service provided.
F-4	Admit Patient	This feature allows the admin to admit registered patients to a ward or room by specifying their disease name and admission date, while checking room availability. It also enables assigning the patient to a specific doctor from the registered doctor list.
F-5	Ward Details	This feature stores details about hospital wards, such as ward names, bed count, and charges per bed. It helps track ward availability and associated costs.
F-6	Room Details	This feature provides hospital room details, including room number, type (e.g., general, deluxe), and daily charges. You can add, update, delete, and retrieve room records. It helps organize room availability and manage patient accommodations.
F-7	Discharge Patient	This feature enables the admin to discharge an admitted patient by retrieving their information, providing a discharge date, and adding discharge remarks.
F-8	Billing Information	This feature calculates and stores billing details for admitted patients, including room charges, length of stay, services availed,

		payment mode, and billing date. It ensures accurate and up-to-date billing records for each patient.
F-9	Staff Record	This feature stores and organizes records of hospital staff, including nurses and ward boys, their qualifications, and joining dates. It helps track staff details and roles.

3. Software Requirements

Feature-1: Manage Doctor Information

SRS-1.1: The system shall allow the admin to register a new doctor by entering their full name, specialty, contact information, and qualifications.

SRS-1.2: The system shall store doctor details such as name, specialty, and contact information in a secure, encrypted database.

SRS-1.3: The system shall allow the admin to get doctor information such as name, specialty, or doctor ID, and clicking the "Get" button to retrieve the relevant data.

SRS-1.4: The system shall allow the admin to update any of the doctor’s details, including name, specialty, and contact information.

SRS-1.5: The system shall allow the admin to delete a doctor’s record, ensuring that the data is permanently removed from the database.

Feature-2: Manage Patient Information

SRS-2.1: The system shall allow the admin to register a new patient by entering personal details such as name, date of birth, gender, and contact information.

SRS-2.2: The system shall store the patient’s medical history, including previous diagnoses, allergies, and medications in a secure database.

SRS-2.3: The system shall allow the admin to update patient records, including medical history and personal information.

SRS-2.4: The system shall allow the admin to get patient name, patient ID, or blood group.

SRS-2.5: The system shall allow the admin to delete a patient’s record, ensuring that it is permanently removed from the system.

Feature-3: Laboratory Services

SRS-3.1: The system shall allow the admin to add laboratory services such as blood tests, imaging, and diagnostic reports, along with the associated patient and charges.

SRS-3.2: The system shall store the details of laboratory services provided to a patient, including the service type, associated charges, and patient information.

SRS-3.3: The system shall allow the admin to modify the service details (e.g., service type, charges) for a patient when required.

SRS-3.4: The system shall allow the admin to delete laboratory services provided to a patient, ensuring the removal of associated data from the system.

Feature-4: Admit Patient

SRS-4.1: The system shall allow the admin to admit a patient by getting the patient’s disease name, admission date, and assigning a room or ward.

SRS-4.2: The system shall verify room or ward availability before proceeding with patient admission.

SRS-4.3: The system shall allow the admin to assign a registered doctor to the patient based on the doctor’s specialty and availability.

SRS-4.4: The system shall generate an admission record with details such as patient name, disease name, assigned room, and doctor.

SRS-4.5: The system shall notify the admin if a room or ward is unavailable during the admission process.

Feature-5: Ward Details

SRS-5.1: The system shall allow the admin to register and update ward details, including ward name, bed count, and bed charges.

SRS-5.2: The system shall track the occupancy status of each ward and show available beds in real time.

SRS-5.3: The system shall allow the admin to view ward availability and generate reports for ward occupancy.

SRS-5.4: The system shall calculate and display the charges per bed for each ward, including daily rates.

SRS-5.5: The system shall notify the admin if a ward exceeds its bed capacity.

Feature-6: Room Details

SRS-6.1: The system shall allow the admin to register room details, including room number, type (e.g., general, deluxe), and daily charges.

SRS-6.2: The system shall track the availability of each room, updating real-time occupancy status.

SRS-6.3: The system shall allow the admin to update or delete room details such as room type and daily charges.

SRS-6.4: The system shall allow the admin to view and manage the room availability.

Feature-7: Discharge Patient

SRS-7.1: The system shall allow the admin to enter the discharge date and any discharge remarks for a patient.

SRS-7.2: The system shall update the patient’s record to reflect the discharge, including the discharge date and summary of treatment.

SRS-7.3: The system shall update billing information to include final charges for room, services, and treatment during the patient’s stay.

SRS-7.4: The system shall notify the admin and other relevant personnel (e.g., billing department, doctor, nurse) when a patient is discharged.

Feature-8: Billing Information

SRS-8.1: The system shall automatically calculate the total charges for an admitted patient based on room charges, length of stay, and services availed during the admission period.

SRS-8.2: The system shall store and display detailed billing information for each patient, including the room charges, service charges, laboratory test charges, and any other additional costs incurred.

SRS-8.3: The system shall allow the admin to select and record the payment mode (e.g., cash, credit card, insurance) for each billing transaction.

SRS-8.4: The system shall ensure that the billing information is updated in real-time as services are availed, so that the patient’s total bill is always up to date.

SRS-8.5: The system shall generate and store an itemized invoice for each patient that includes all charges (room, services, tests, etc.), the payment mode, and the billing date.

Feature-9: Staff Record

SRS-9.1: The system shall allow the admin to register new hospital staff, including nurses, ward boys, and other personnel, by entering their name, qualifications, job role, and joining date.

SRS-9.2: The system shall allow the admin to retrieve staff records using filters such as name, job role, or department.

SRS-9.3: The system shall allow the admin to update existing staff records, ensuring that changes are correctly saved and reflected in the system.

SRS-9.4: The system shall allow the admin to delete staff records, ensuring that all associated data is permanently removed from the system.

SRS-9.5: The system shall ensure that all modifications to staff records (updates, deletions, and registration) are saved in the database.

5.Chosen Architecture: **Two-Tier Client–Server Architecture**

The Hospital Management System is developed using a **Two-Tier Client–Server Architecture**, where the application is divided into two main layers:

1. **Client Tier (Java Swing Application)**
2. **Database Tier (MySQL Server)**

This structure is common for desktop-based management systems that require direct database connectivity without a separate web server or middleware layer.

5.1. Client Tier (Presentation + Application Logic Layer)

The **Client Tier** is implemented using **Java Swing**. It includes all graphical user interfaces such as:

- Login Screen
- Admin Dashboard
- Doctor Management
- Patient Management
- Appointment Handling
- Hospital Details
- Update/Delete Screens

In this architecture, the client application handles both **presentation** and **business logic**. Each Swing form (.java files like `DoctorScreen.java`, `PatientScreen.java`) contains:

- UI components
- Event handling (buttons, actions)
- Application logic
- Direct SQL execution

The client communicates directly with the database to perform operations like inserting, updating, deleting, or retrieving records.

5.2. Database Tier (Data Storage Layer)

The **Database Tier** consists of a MySQL database that stores all system data, including:

- Doctor information
- Patient profiles
- Staff details
- Hospital records
- Appointments
- Billing information

The application uses **JDBC** to send SQL queries to the MySQL database. The class `MYSQLConnection.java` acts as the helper to establish database connectivity.

5.3. How the Architecture Works

1. **User interacts with the GUI** (Swing windows and forms).
2. **Client application processes logic** such as validation or data formatting.
3. **JDBC sends SQL queries directly to the MySQL server**.
4. **MySQL executes the query** and returns a result set.
5. **Client tier updates the interface** with new or modified data.

This direct communication makes the system fast and suitable for local machine deployments.

5.4. Why This Architecture is Used

This architecture fits your system because:

Simple & Efficient for Desktop Applications

The system runs locally, so direct client-to-database communication provides high speed.

Easy to Develop and Maintain

Only two layers are involved; no need for an additional server or middleware.

Real-Time Database Operations

All CRUD operations reflect immediately in the database.

Suitable for Small to Medium-Scale Systems

Hospitals with local networks or single-machine setups can easily deploy this solution.

6. Tools and Technologies Used

The development of the Hospital Management System utilizes a combination of reliable, widely adopted, and industry-standard technologies. These tools were selected to ensure system stability, ease of development, secure data handling, and long-term maintainability.

1. Java (Programming Language)

Java is used as the core programming language for building the application's logic and user interface. Its platform independence, object-oriented structure, and strong standard libraries make it ideal for developing robust desktop applications. Java provides high reliability, security features, and excellent support for structured application design through architectures such as MVC.

2. Java Swing (Graphical User Interface Framework)

Java Swing is used for designing the graphical user interface of the system. Swing allows the development of interactive forms, tables, and components required for modules like patient registration, billing, room management, and staff handling. It offers flexibility, customizable layouts, and a responsive UI suitable for desktop-based management systems.

3. MySQL (Database Management System)

MySQL serves as the backend database for storing all hospital data. It ensures efficient storage, retrieval, and updating of records related to patients, doctors, laboratory services, staff, rooms, and billing. MySQL is chosen because of its reliability, high performance, security features, and ease of integration with Java applications through JDBC.

4. JDBC (Java Database Connectivity)

JDBC acts as the connection bridge between the Java application and the MySQL database. It is used to perform operations such as inserting, updating, deleting, and retrieving records. JDBC enables seamless communication and ensures real-time data consistency throughout the system.

5. NetBeans

An Integrated Development Environment is used to write, debug, compile, and organize the project efficiently.

It simplifies project structure management and improves development speed through tools like autocomplete, GUI builders, and debugging features.

4. Implementation Overview

The implementation of the Hospital Management System is based on the MVC architecture, ensuring a clean separation between user interface, business logic, and data handling. The system is divided into multiple interactive modules that collectively support hospital operations.

Key Features Implemented

The major functional areas covered in the implementation include:

- **Patient Management**
  Registration, viewing, updating, and deleting patient records.
- **Doctor Management**
  Storing and managing doctor profiles, specialties, and contact details.
- **Staff Management**
  Adding and managing nurse, ward boy, and other staff information.
- **Room & Ward Management**
  Managing room types, wards, availability, and hospital accommodation details.
- **Patient Admission**
  Assigning rooms/wards and allocating doctors during admission.
- **Laboratory Services**
  Adding test results, lab charges, and diagnostic information.
- **Billing System**
  Calculating room charges, lab fees, and total bill based on the stay duration and services.

How the System Works

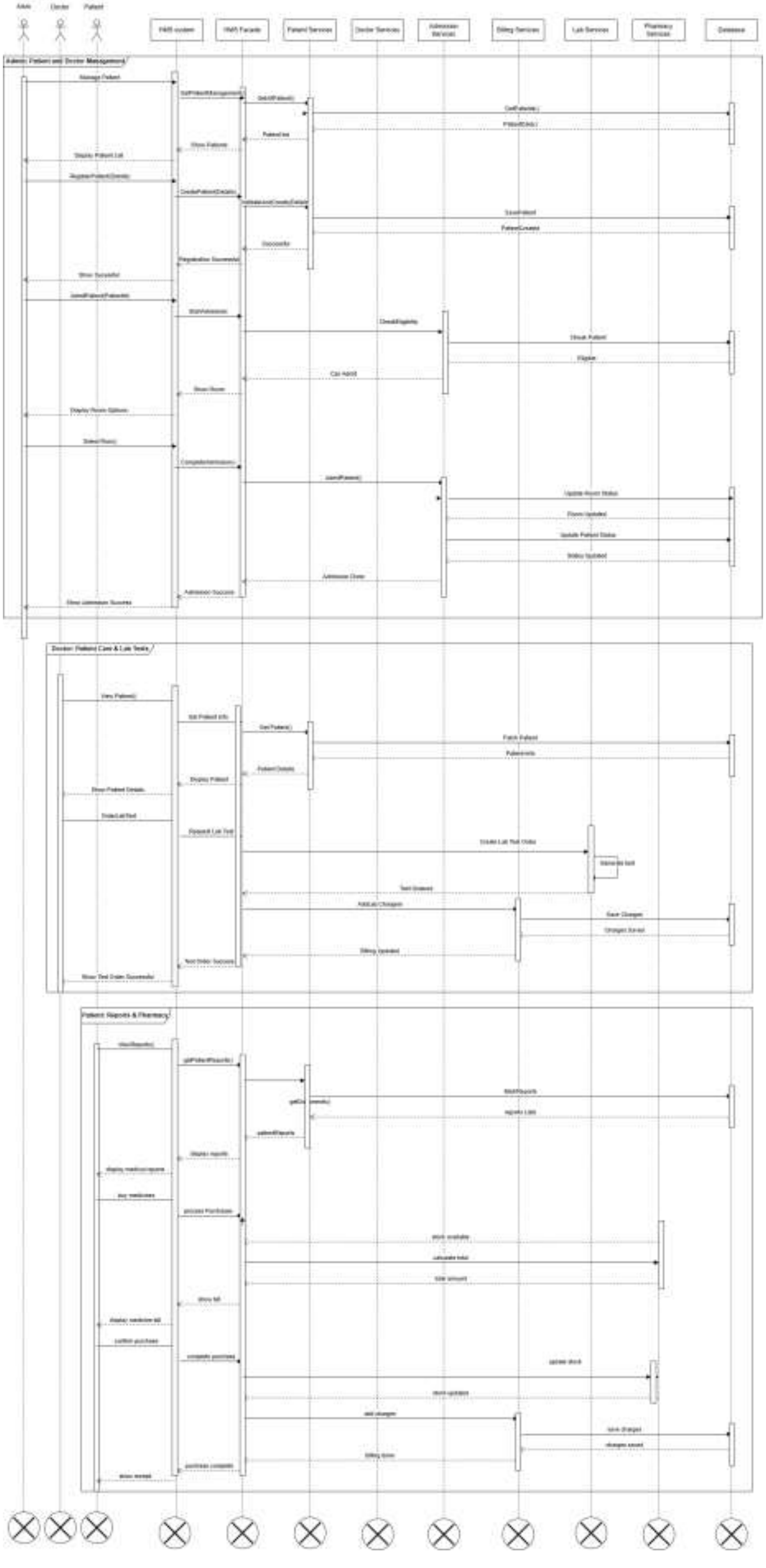
1. **User interacts with the GUI (View)**
   The admin or staff member enters data using forms and buttons provided on the interface.
2. **Controller validates and processes input**
   The controller reads the user action, validates fields, and sends requests to the Model.
3. **Model performs operations using JDBC + MySQL**
   The Model interacts with the MySQL database to insert, update, delete, or retrieve data.
4. **Updated results are reflected back to the View**
   The user interface refreshes to display updated records, tables, or messages.

This flow ensures structured communication and real-time updates across all modules.

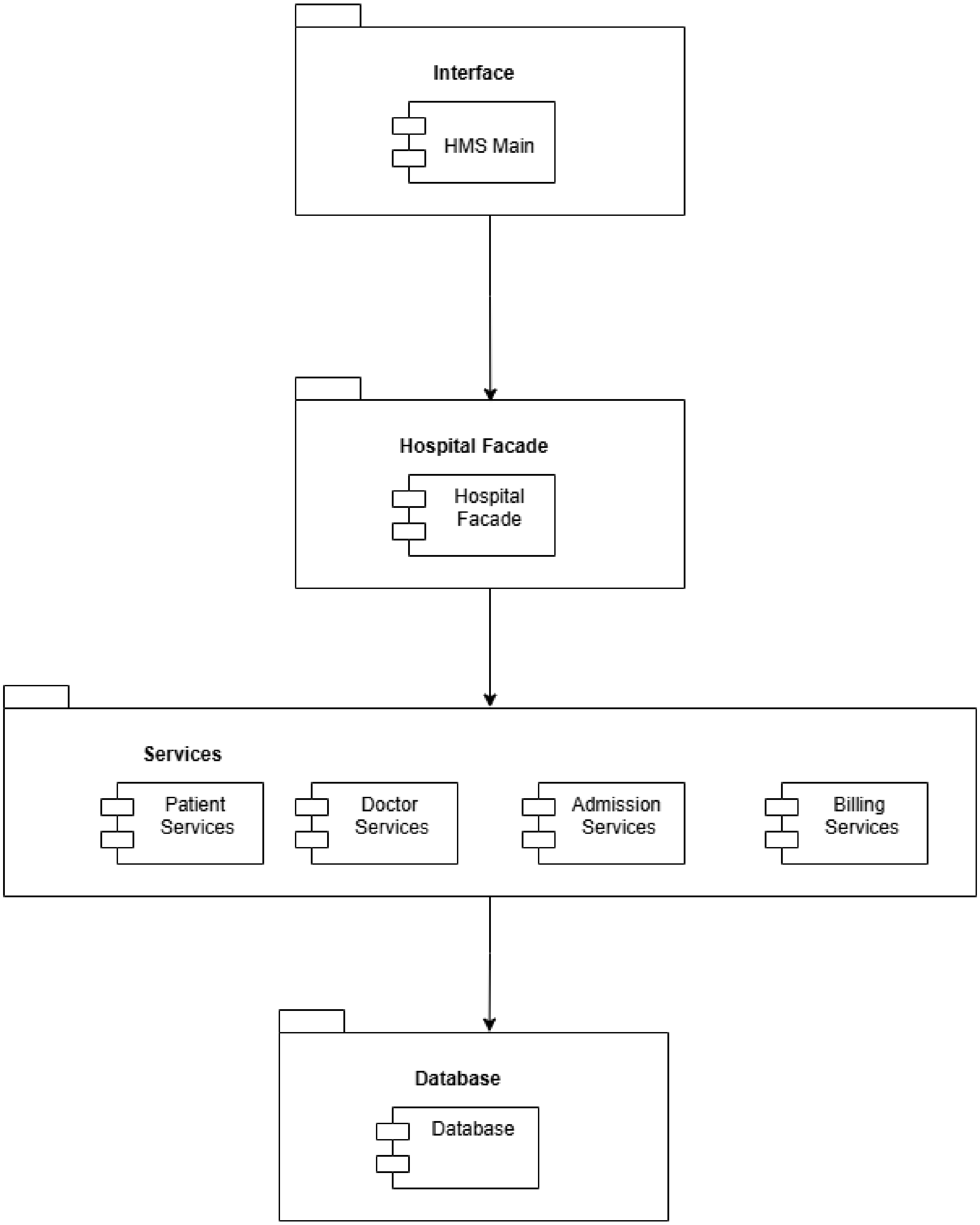
Artifact-2: Class & Sequence Diagrams

4. Design Diagrams

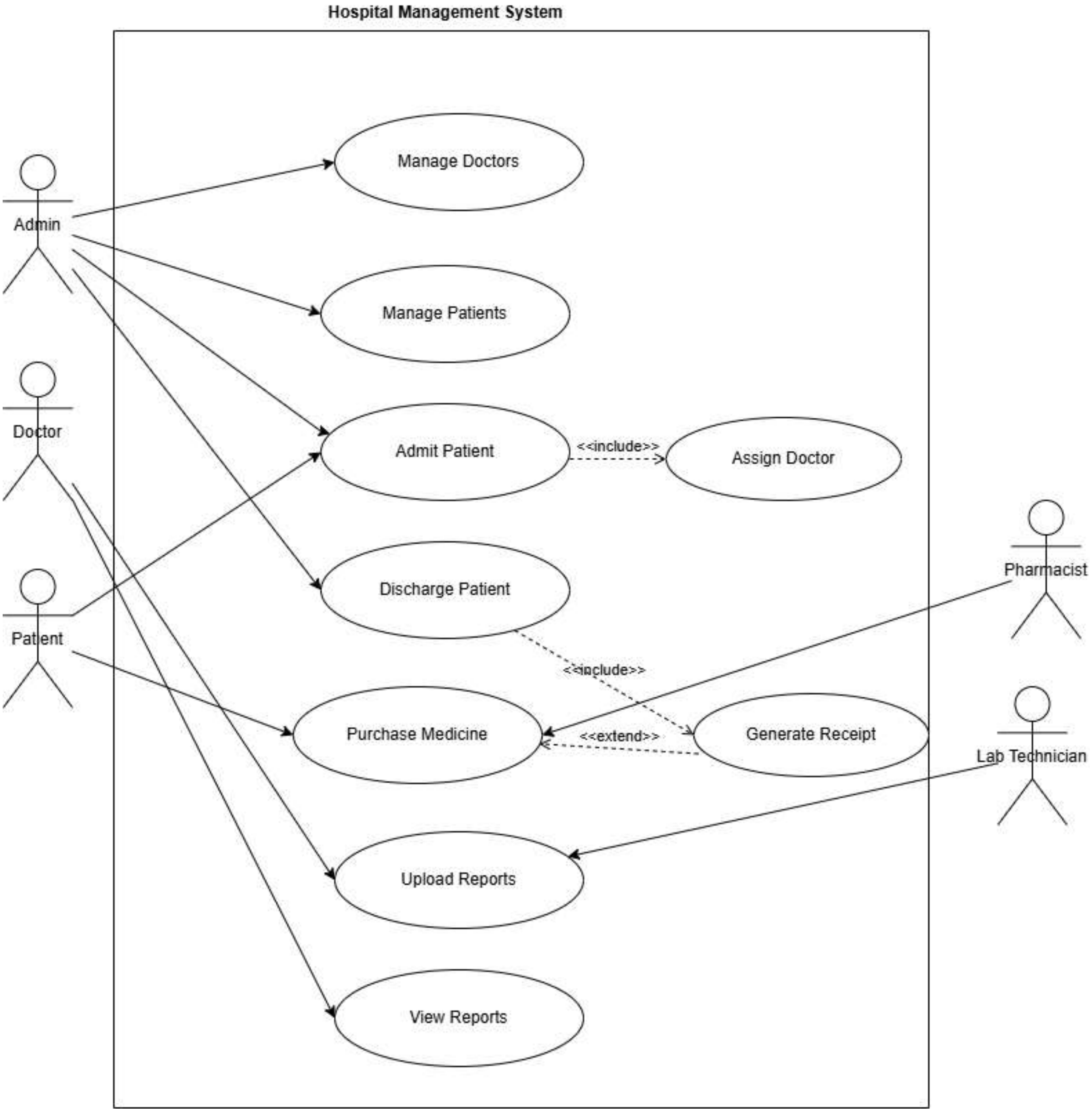
4.1. Sequence Diagrams:


Admin
Doctor
Patient
HMS system
HMS Facade
Patient Services
Doctor Services
Admission Services
Billing Services
Lab Services
Pharmacy Services
Database
Admin: Patient and Doctor Management
Manage Patient
GetPatientManagement()
GetAllPatients()
GetPatients()
PatientData()
PatientList
Show Patients
Display Patient List
RegisterPatient(Details)
CreatePatient(Details)
InitializeAndCreate(Details)
SavePatient
PatientCreated
Successful
Registration Successful
Show Successful
AdmitPatient(PatientId)
StartAdmission
CheckEligibility
Check Patient
Eligible
Can Admit
Show Room
Display Room Options
Select Room()
CompleteAdmission()
AdmitPatient()
Update Room Status
Room Updated
Update Patient Status
Status Updated
Admission Done
Admission Success
Show Admission Success
Doctor: Patient Care & Lab Tests
View Patient()
Get Patient Info
Get Patient()
Fetch Patient
PatientInfo
Patient Details
Display Patient
Show Patient Details
OrderLabTest
Request Lab Test
Create Lab Test Order
Generate test
Test Ordered
AddLabCharges
Save Charges
Charges Saved
Billing Updated
Test Order Success
Show Test Order Successful
Patient: Reports & Pharmacy
ViewReports()
getPatientReports()
getDocuments()
fetchReports
reports List
patientReports
display reports
display medical reports
buy medicines
process Purchase
stock available
calculate total
total amount
show bill
display medicine bill
confirm purchase
complete purchase
update stock
stock updated
add charges
save charges
charges saved
billing done
purchase complete
show receipt

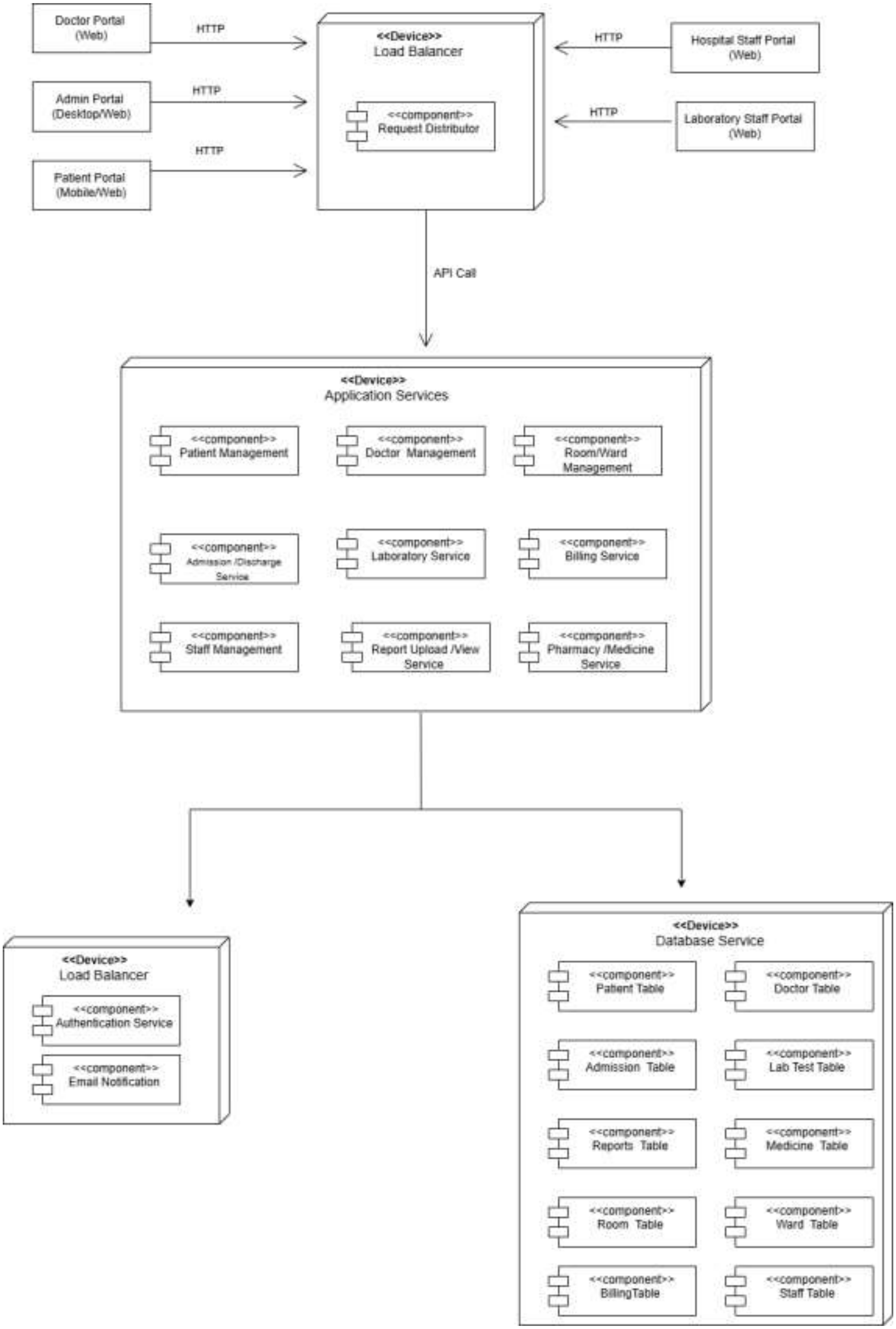
4.2. Package Diagram:



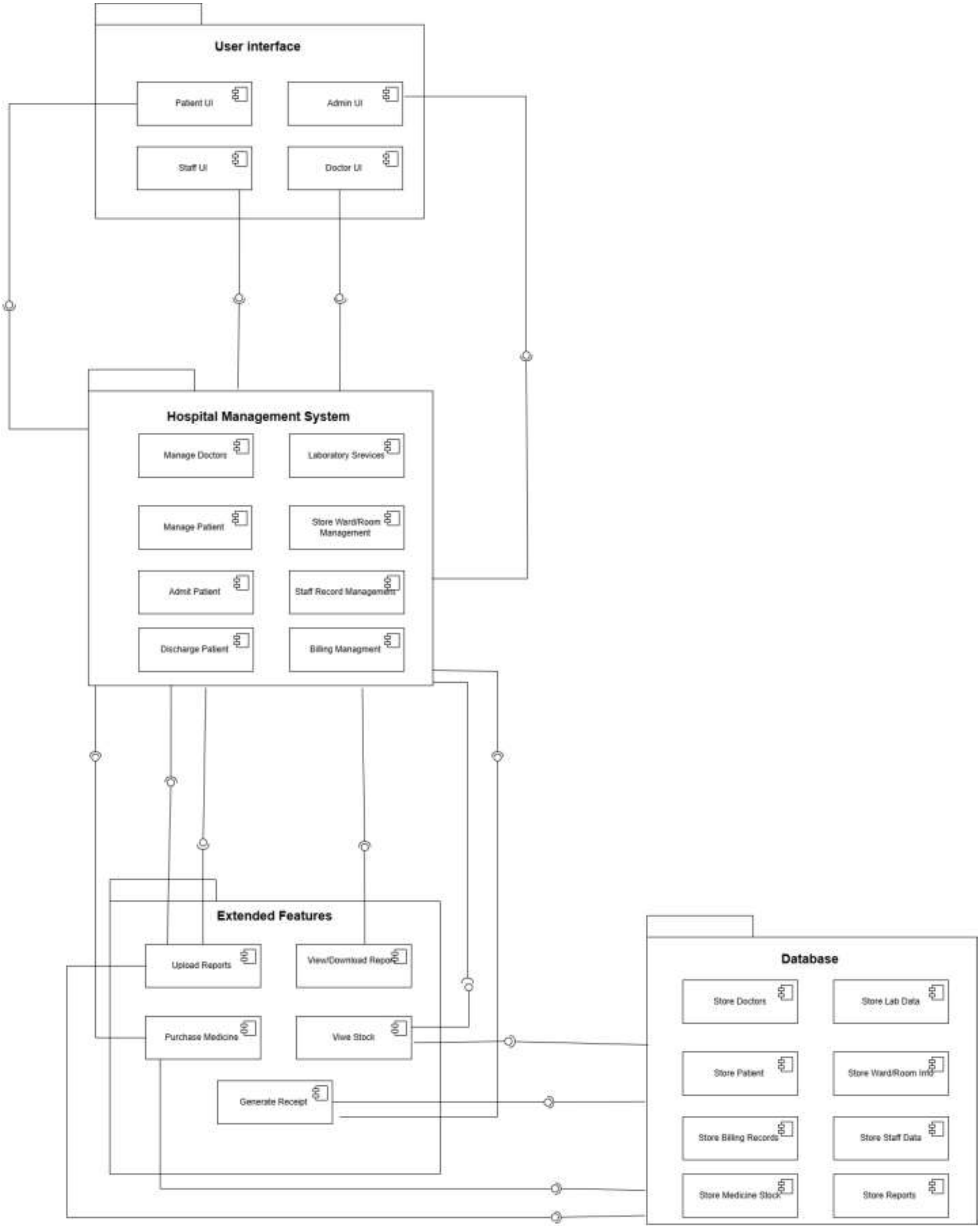
4.3. Use Case Diagram:


Hospital Management System
Admin
Doctor
Patient
Pharmacist
Lab Technician
Manage Doctors
Manage Patients
Admit Patient
<<include>>
Assign Doctor
Discharge Patient
<<include>>
Generate Receipt
<<extend>>
Purchase Medicine
Upload Reports
View Reports

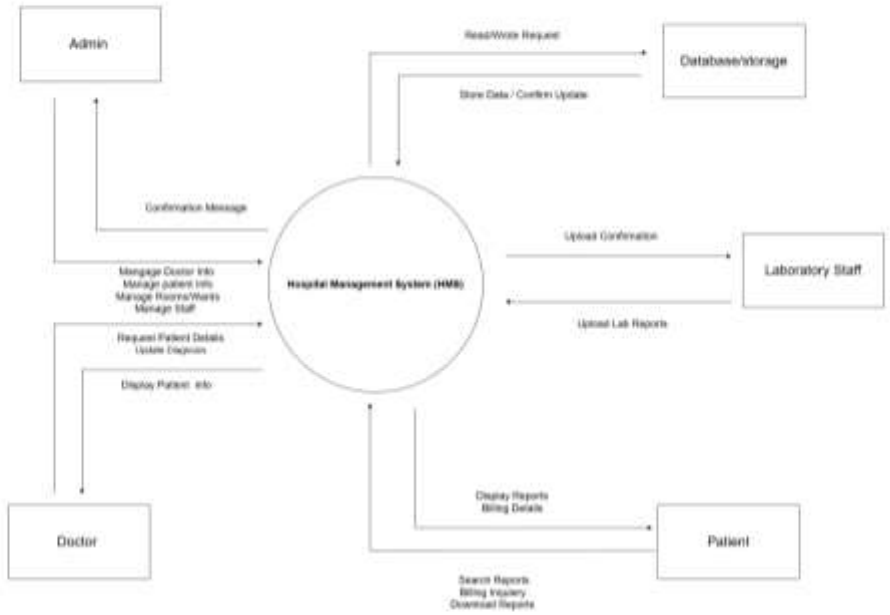
4.4. Deployment Diagram:



4.5. Component Diagram:


User interface
Patient UI
Admin UI
Staff UI
Doctor UI
Hospital Management System
Manage Doctors
Laboratory Srevices
Manage Patient
Store Ward/Room Management
Admit Patient
Staff Record Management
Discharge Patient
Billing Managment
Extended Features
Upload Reports
View/Download Report
Purchase Medicine
Viwe Stock
Generate Receipt
Database
Store Doctors
Store Lab Data
Store Patient
Store Ward/Room Info
Store Billing Records
Store Staff Data
Store Medicine Stock
Store Reports

4.6. DFD Diagram:

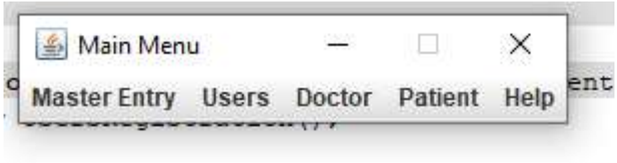


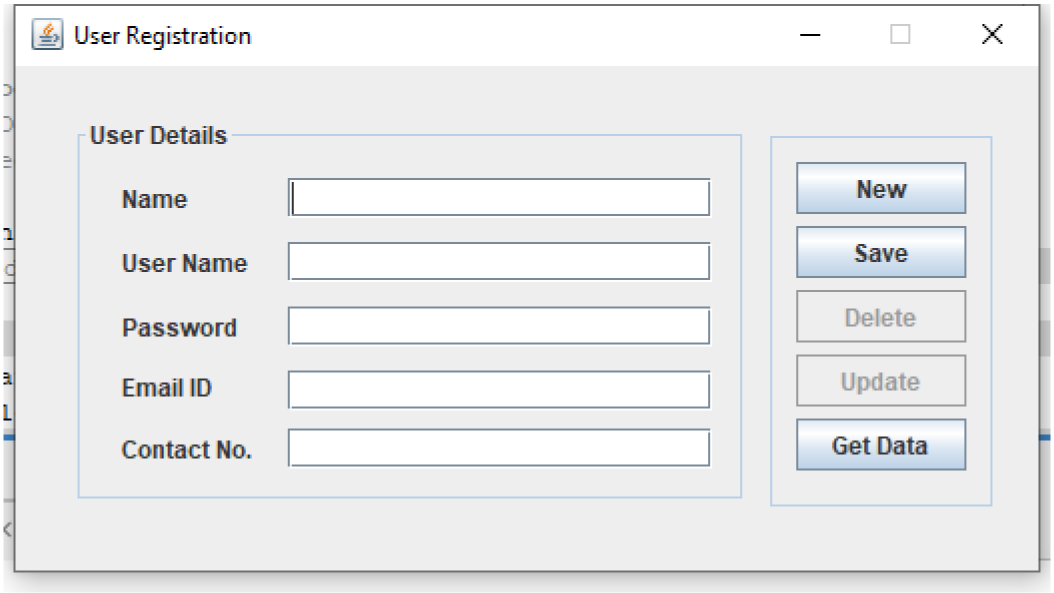
Artifact 03

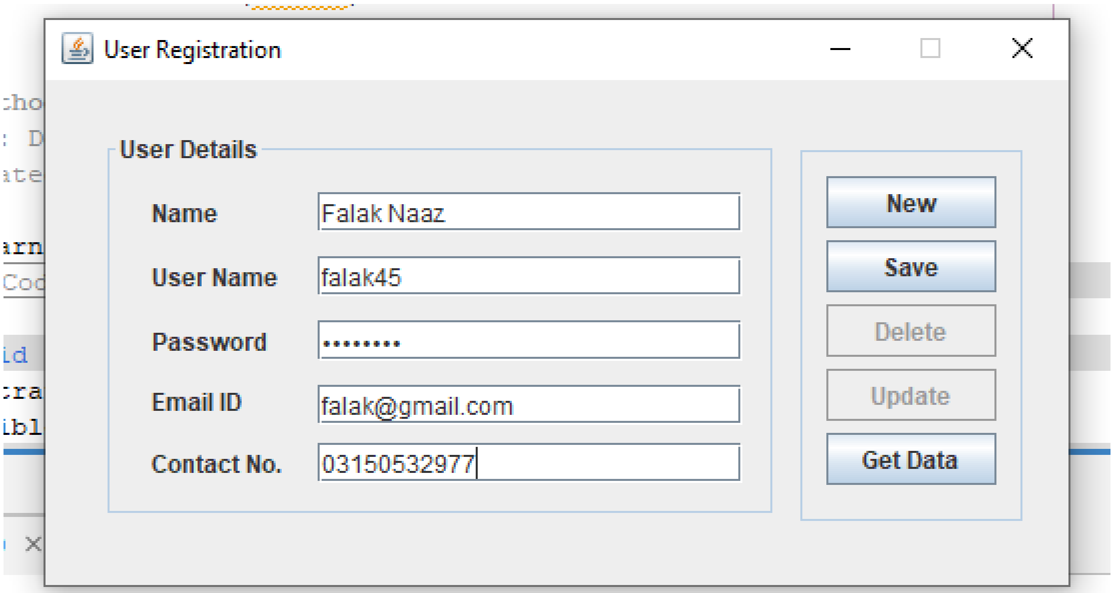
Coding

5. Coding and Implementation:

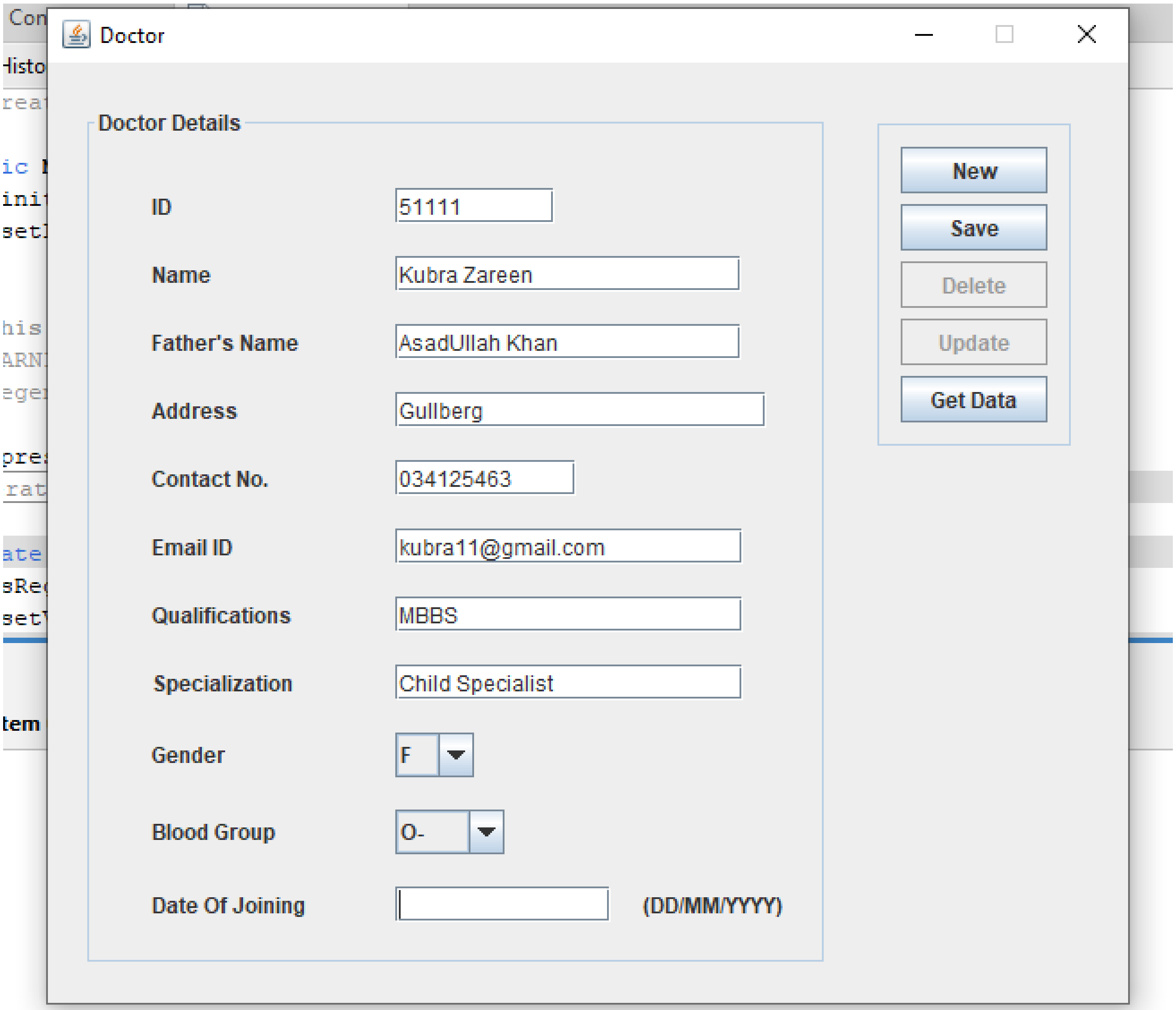
5.1. Output:





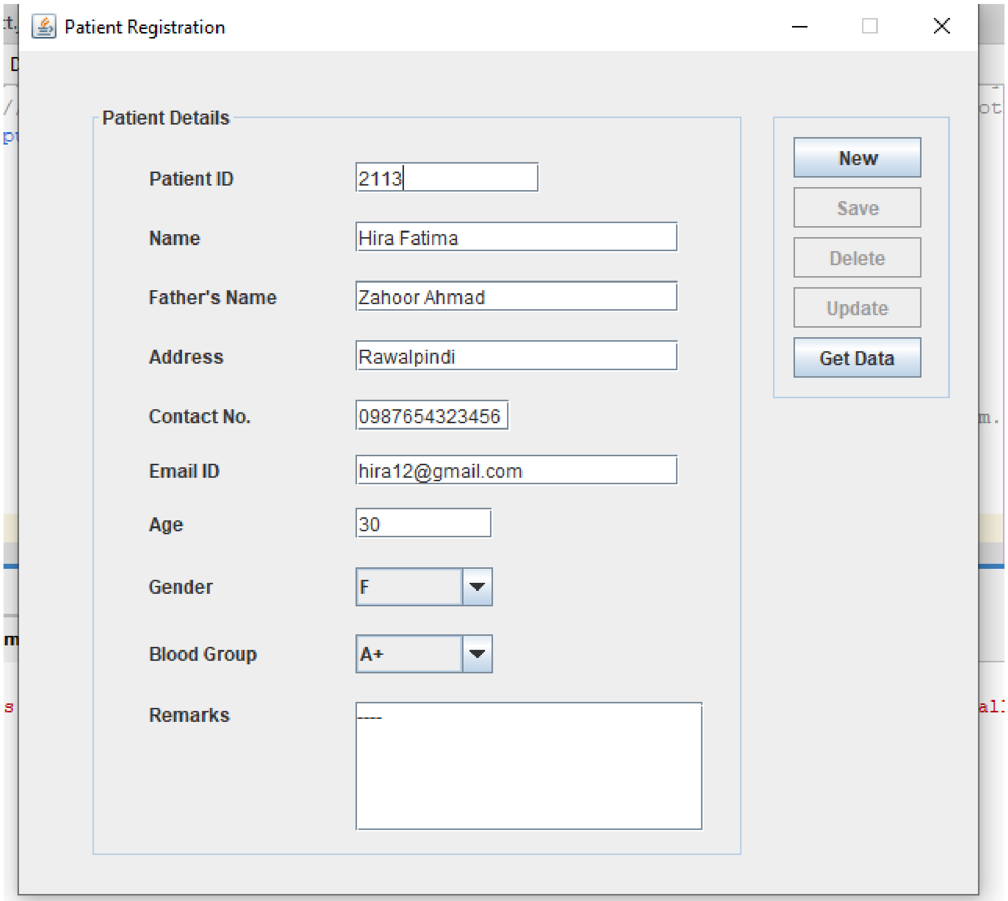


Name: Falak Naaz
User Name: falak45
Password: ••••••••
Email ID: falak@gmail.com
Contact No.: 03150532977



Doctor Details

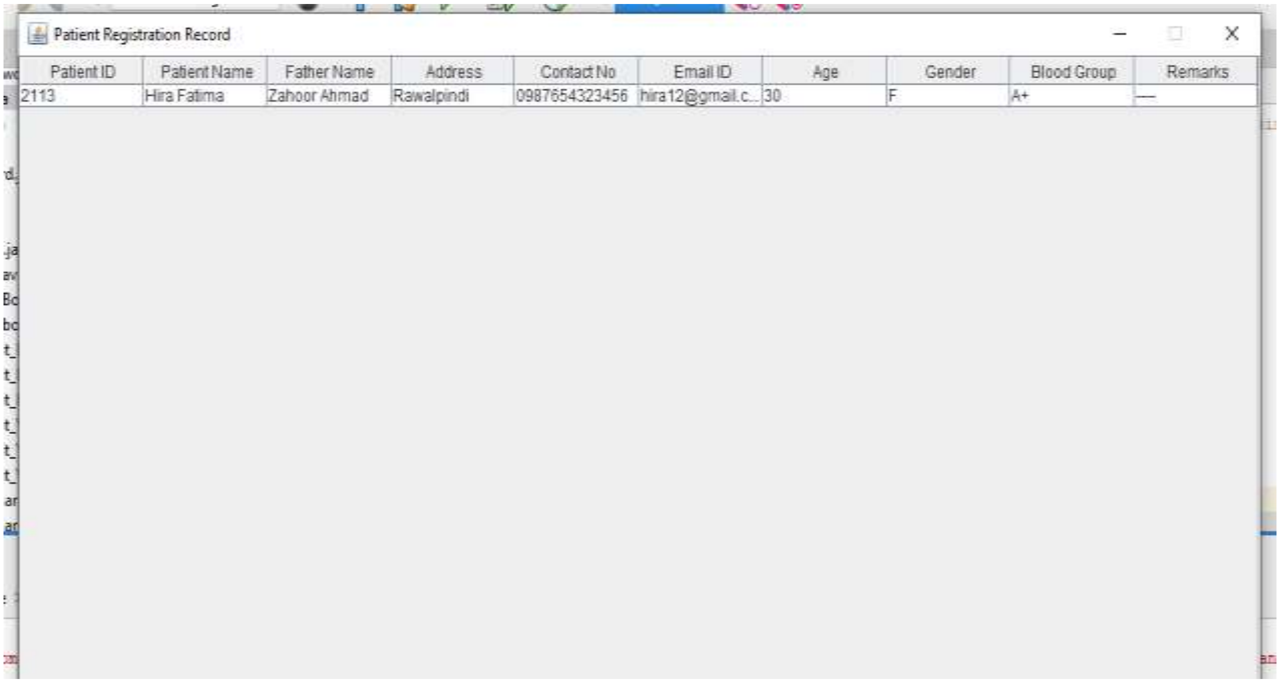
ID	51111
Name	Kubra Zareen
Father's Name	AsadUllah Khan
Address	Gullberg
Contact No.	034125463
Email ID	kubra11@gmail.com
Qualifications	MBBS
Specialization	Child Specialist
Gender	F
Blood Group	O-
Date Of Joining	(DD/MM/YYYY)



Patient Registration

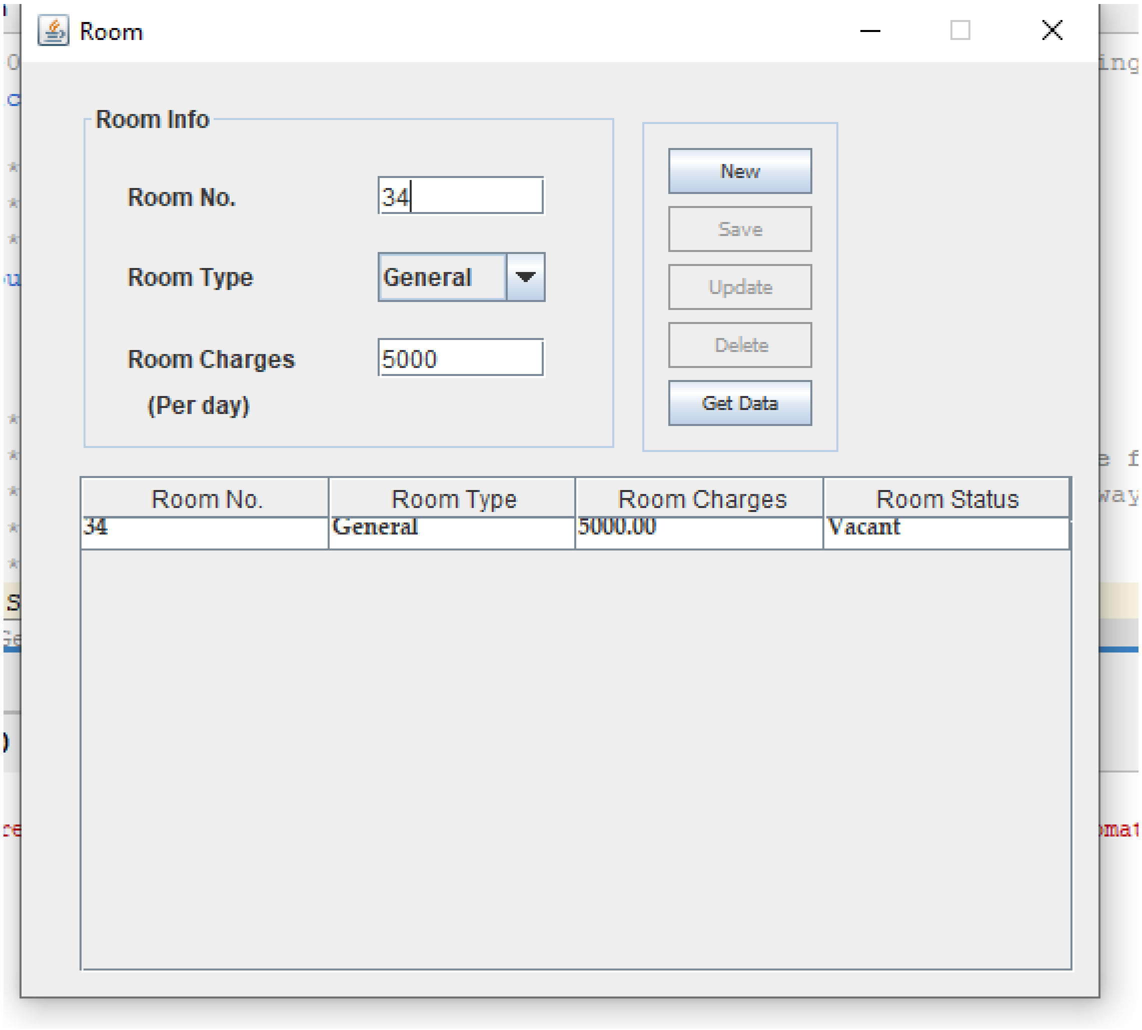
Patient Details

Patient ID	2113
Name	Hira Fatima
Father's Name	Zahoor Ahmad
Address	Rawalpindi
Contact No.	0987654323456
Email ID	hira12@gmail.com
Age	30
Gender	F
Blood Group	A+
Remarks	----



Patient Registration Record

Patient ID	Patient Name	Father Name	Address	Contact No	Email ID	Age	Gender	Blood Group	Remarks
2113	Hira Fatima	Zahoor Ahmad	Rawalpindi	0987654323456	hira12@gmail.c...	30	F	A+	----



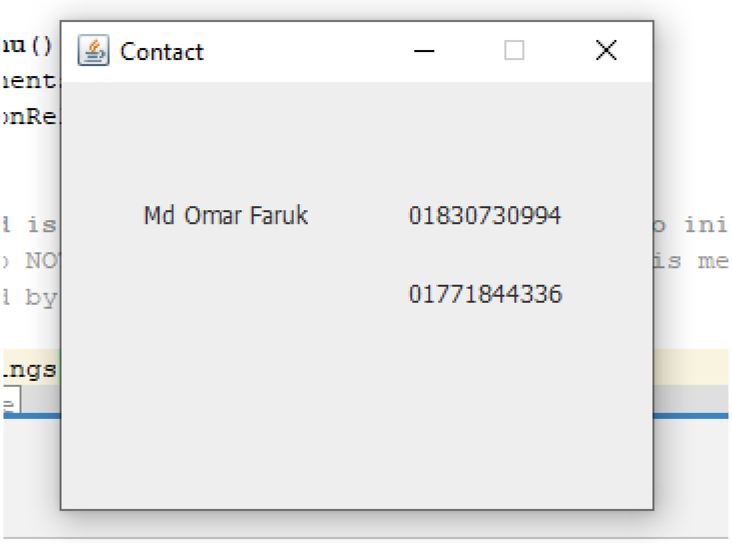
Room Info

Room No. 34

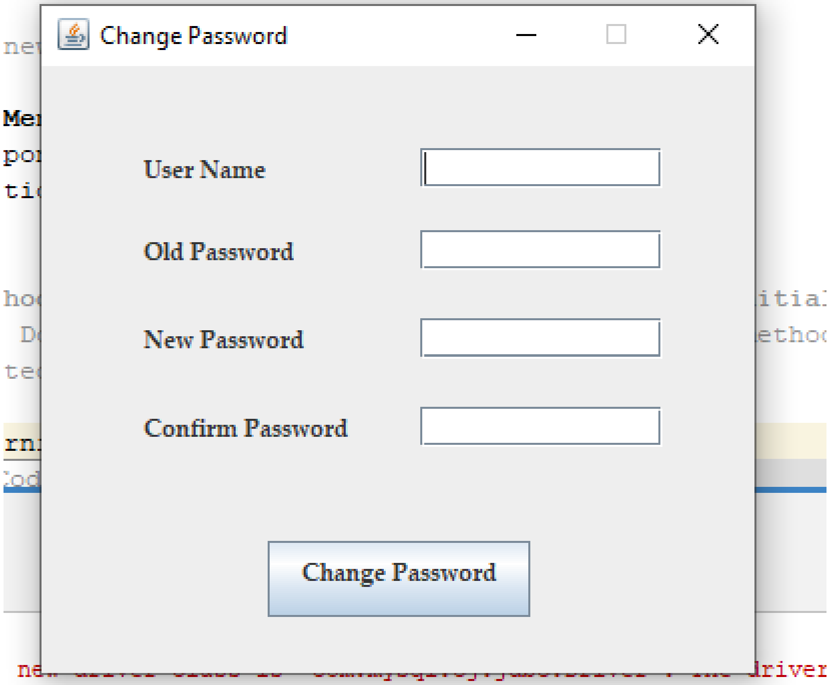
Room Type General

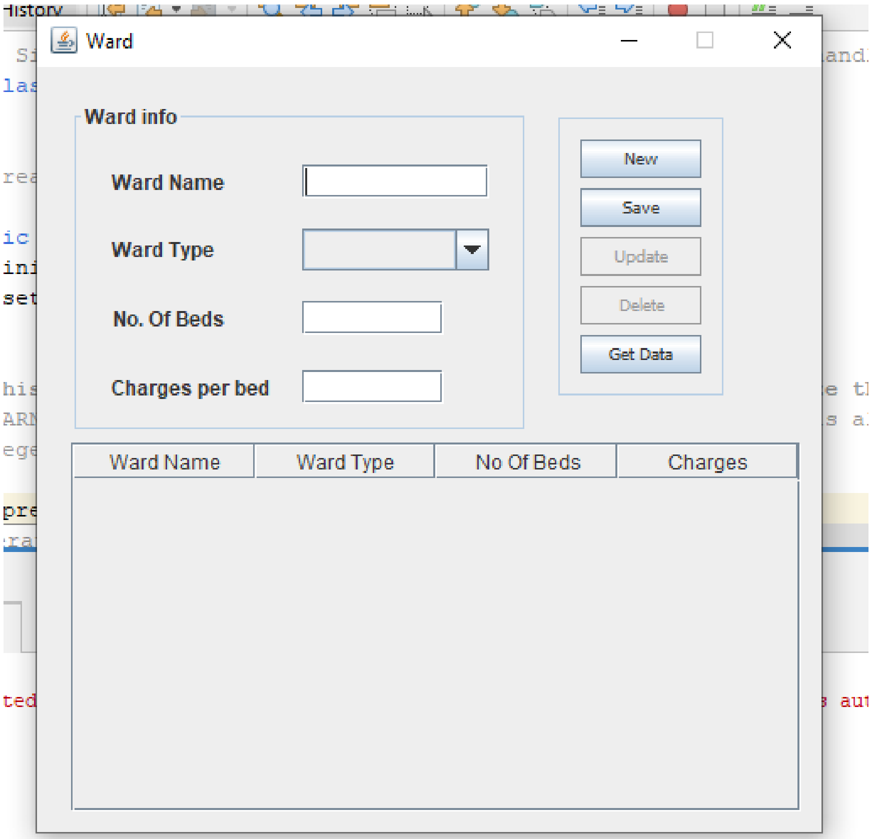
Room Charges (Per day) 5000

Room No.	Room Type	Room Charges	Room Status
34	General	5000.00	Vacant

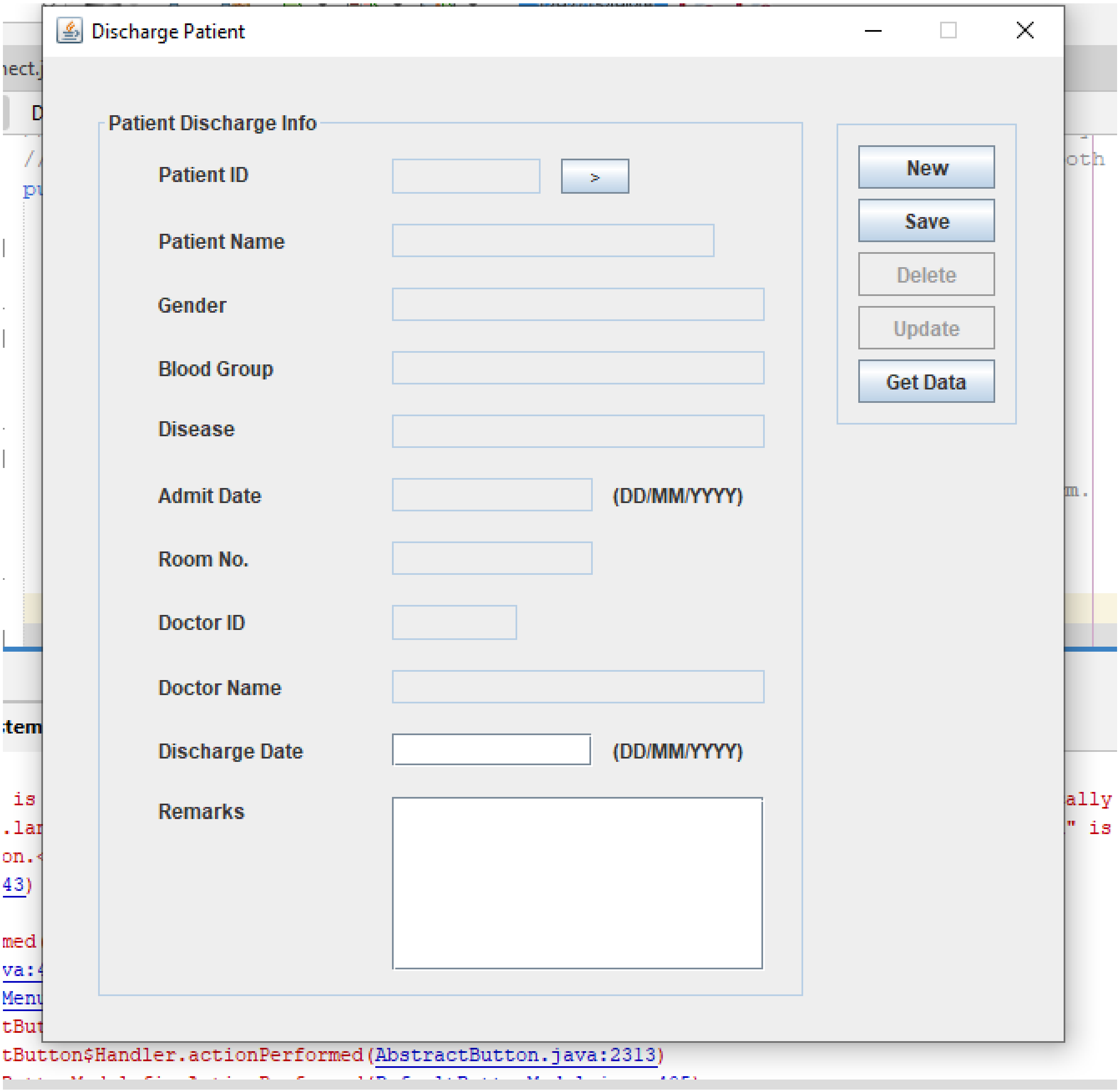


Md Omar Faruk	01830730994
	01771844336





Ward Name	Ward Type	No Of Beds	Charges



```
tButton$Handler.actionPerformed(AbstractButton.java:2313)
```

6. Conclusion:

The Hospital Management System developed in this project successfully addresses the growing need for digitalization, accuracy, and efficiency within healthcare operations. By replacing traditional manual processes with an integrated and automated platform, the system enhances coordination among hospital departments and ensures timely access to accurate patient and administrative information.

Through its comprehensive modules—including patient and doctor management, laboratory services, staff records, room and ward allocation, billing, and admission/discharge workflows—the system streamlines daily hospital activities and minimizes human error. Each feature is designed in alignment with clearly defined software requirements, ensuring reliability, usability, and consistency in real-time data handling.

The project adopts a **Two-Tier Client–Server Architecture**, where the Java Swing–based client communicates directly with the MySQL database. This architecture supports fast execution of CRUD operations, simplifies deployment, and makes the system well-suited for small to medium-scale hospital environments. Furthermore, the inclusion of architectural diagrams such as Use Case, DFD, Deployment, Component, Package, and Sequence diagrams establishes a strong structural and behavioral understanding of the system. The refactoring process also strengthened the design by improving code clarity, maintainability, and extensibility.

Overall, this Hospital Management System provides a reliable, organized, and scalable solution that enhances operational efficiency and supports better decision-making within hospitals. By combining robust software design principles with practical functionalities, the system significantly contributes to improving patient care quality and optimizing hospital workflows.