KM^3 Project

Hospital Management Desktop Application.

Name
Kareem Amr
Mazen Ahmed
Muhammad Salah
Mohamed Magdy

Computer Engineering – Helwan University

Contents

Introduction

This documentation is a description of KM^3 Project , it aims to give users and clients a complete picture of the entire system, by providing information about the functionalities of the software system, and how clients can benefit from it. KM^3 is a Hospital Software System, A desktop Application made with Java programming Language. We've Used JFrame Swing in GUI making and for database we've used SQLITE. our software system should give solution for the administrative and managing staff. Also, to help doctors and patients with easy procedures and steps. This project can help not only with hospitals but also with health clinics and centers whatever their sizes are. Privacy And safety for these data is also obtained. Data is well protected for personal users; every User has his own username and password. our application is friendly user and so simple. Without this application data in the hospital will be still papers and some data maybe lost, this application helps in automating the hospital by organizing data. This software will help the hospitals/Clinics to be more efficient in registration of their patients and manage appointments, records of patients and other administrative staff. The purpose of this project is computerizing all details regarding patient details and hospital details. A hospital system is a software designed to manage all the areas of a hospital such as medical, financial, administrative and the corresponding processing of services. Daily functions like patient registration, managing admission and overall management of various departments can be easily performed with higher accuracy after the installation of hospital software. The modules of hospital management software are user-friendly and easy to access. Hospital Management System is a system enabling hospitals to manage information and data related to all aspects of healthcare – processes, providers, patients, and more, which in turn ensures that processes are completed swiftly and effectively. When one thinks of the various aspects and departments of a hospital, it becomes apparent that an HMS is critical. The hospital database management system was introduced in 1960 and has greatly

evolved since then – with the ability to integrate with the existing facilities, technologies, software, and systems of a hospital. Today, patients can begin the process of healthcare in the palm of their hand – the mobile devices and apps – make this possible. This process then moves to the healthcare providers and hospitals. With the large amounts of data, people involved and innumerable processes, a hospital is an ideal candidate for data management software. If hospitals are to run efficiently, provide top line care, ensure patient and other data confidentiality, and work seamlessly – they cannot hope to do so without an effective Hospital Management System Software. Reduced human intervention for paperwork, less paperwork, reduced staff headcount for jobs that can be easily managed within the HMS, speedier processes, reduction of errors, and data privacy and safety – are just some of the benefits of a Hospital Management System. For the hospitals, HMS translates to being able to track patient history, provide better care, keep track of appointments, save patient insurance and payment data, enable doctors and clinicians to check patient history, maintain patient care continuity, and save time and effort on unnecessary tedious manual tasks. This Electronic Medical Record (EMR) or Electronic Health Record (EHR) is the journey of a patient with the hospital – keeping track of the date of every visit, doctor consulted, medicines and advice prescribed, and other information for the patient. This ensures that even if a patient visits after a long break, the patient and hospital will not require going through the registration process again. Hospital records are easily audited and kept compliant with policies and laws. In addition, the Hospital Management System is cost effective – it reduces the need for staff to manage manual entries, manage paperwork, and ensure accurate filing. This in turn significantly reduces the possibility of human error, which can prove costly on many counts. Another significant benefit/factor of the HMS is it is customizable to the needs and requirements of a particular hospital/healthcare facility. If we want to check a previous record of a patient or other detail. Management will be in a great problem. It’s a tough

and time taking process to search for a record in a file. Keeping files takes much time and waste much precious man hours. The tendency of making mistakes is high when functioning manually. It is hard to relay on the accuracy of calculations done manually too. It is more obvious for problems to arise. We plan to overcome the above-mentioned problems through a standalone application, to manage the major functions of the Hospital System. The hospital management system we are going to implement will be covering all basic processes done in the hospital. It would handle Employee and Salary management, Patient and "Zumba exercise management, Theatre and ward Management, Laboratory management, Transport Management, Pharmacy Management, OPD management and emergency management. In OPD unit, with the OPD and Consultation Management system, the manual doctors channeling details entering process has automated. So, the staff does not need to spend time on writing appointment records and updating them in files. And the number issuing process becomes easier and efficient. And keeping the track of patients and medical prescription detail sallow them to review the details whenever needed. Implementing the Employee & Salary Management system we record Attendance, shifting of employees, their holidays and consulting doctors' schedules. And the system performs calculations of EPF/ETP and OT hours, Shares of consulting doctors and do the payroll part. This is more efficient and more reliable and accurate as the system avoids incorrect data in puts whenever they are occurred. The proposed system for Mini-theatre & Ward Management records details of surgeons, in- patients who are assigned for Wards, different ward details and surgery details. The pharmaceuticals used within the theatre are managed as well. Food menus for the patients according to their diseases based on wards is systemized too. All are digitalized in a systematic way. So, the details of surgeons, patients and surgeries are well organized and can be easily accessed whenever needed. Surgery reports, Ward progress reports, In-ward patient progress details are generated, and history can be tracked too. The Vehicle & Transport

Management system handles all the data on ambulance transport. It manages the time slots of ambulances, driver's and employee details of transport section and provides bill generating facility. And reliable time slot management provides the

System Requirements

Functional Requirements

REQ- 1 Add a new Patient/Doctors	- Admins can add/Edit doctors and edit patients by connecting to data base Receptionist can add patients - Receptionists have the ability to add new patients into the hospital record
REQ- 2 Modify Patient's data	Admins can modify patient data when needed
REQ- 3 View Patient's data	Doctors, Receptionist, managers and admins can access patient data base and view it's list.
REQ- 4 Search for a Patient by Name	This function is used for searching for specific patient by name
REQ- 5 Search for a Patient by National ID	This function is used for searching for specific patient by national id
REQ- 6 Generate State report for any Patient	This function allows Doctors to add reports to the appointed patient
REQ- 7 View History of Patient	This function occurs by connecting to patient database and get patient's reports and medicines written and inserted by the doctor.
REQ- 8 Management Employees	This function is for adding, removing, and editing data of all hospital employees
REQ- 9 Data Base	This is the project base that all data stored in it

Non-Functional Requirements

Availability

The system should be available all the time.

Security

No one can access the system from outside the faculty.

Safety

Servers will be kept in the vice manager office.

Maintainability

Back Up The system shall provide the capability to back-up the Data Errors The system shall keep a log of all the errors.

Security Patient Identification

The system requires the patient to identify himself /herself using mail. Any use who uses the system shall have a Logon Email and Password. Any modification (insert, delete, update) for the Database shall be synchronized and done only by the administrator in the ward.

Functional Requirements Specification

Stakeholders: -

1- Admins
2- Managers
3- Doctors
4- Receptionist

Admins: Can add and edit Doctors table, can edit Patient Table, can add, and edit Pharmacist table, can add, and edit Hospital Rooms.

Managers: Can View Doctors and Patient Table, can see Hospital Analysis.

Doctors: Can add reports to an appointed patient, can view appointed patients, can assign medicines to appointed patient.

Receptionist: Can add patient, search for a patient, book appointment to a patient, view doctors table, assign an ambulance, view patient table.

The whole Functions Are described in next pages

Project's Features (Methods & Functions)

Main Section

rec_doc: this method is used for taking data of doctors from database and return result this method is used for login

rec: this method is used for taking data of patient from database and return result this method is used for login

farm: this method is used for taking data of pharmacist from database and return result this method is used for login

OurTeamActionPreformed: this method is a button that directs to Team Page

LoginMouseClicked: this method is a button that when clicked the program will search if the written data is found in the database or not if yes, the button directs to the page according to entered data

MainLogoMouseClicked: this method is a button that directs to hospital information

showpasswordActionPreformed: this method is used for unhide written password just to make user check if he wrote his password correct

NationalIDTextKeyTyped: this method is used for preventing the user from typing anything but digits

NationalIDTextKeyPressed: this method is used for when jfield is clicked by the mouse the word if it was National ID then the jfield will be null or empty else nothing happens

Set and get password: those methods for set and get admin password

Config class

connectDB: is a method only for connecting to database stored in system

Admins Section

update_table_Doc: this method is used for updating doctors' data from jframe table and store it in the database (doctors table)

update_table_farm: this method is used for updating pharmacy's data from jframe table and store it in the database (pharmacy table)

Delete_pat: this method is used for deleting a patient from database (patientData table) by selecting the user from jframe table

update_table_pat: this method is used for updating patient's data from jframe table and store it in the database (patientData table)

Update_Rooms: this method is used for updating Hospital Emergency Rooms' data from jframe table and store it in the database (Emgroom table)

delete_room: this method is used for deleting a room from database (Emgroom table) by selecting the user from jframe table

delete_farm: this method is used for deleting a pharmacist from database (farmacyDoc table) by selecting the user from jframe table

delete: this method is used for deleting a doctor from database (doclist table) by selecting the user from jframe table

edit_pat: this method is used to save patient data written by Admins and be stored in database (paitnetData table)

edit_farm: this method is used to save pharmacist data written by Admins and be stored in database (farmacyDoc table)

edit: this method is used to save doctor's data written by Admins and be stored in database (doclist table)

add_pd: this method is used for compinning two databases and put them in a new database (Adoc table) this for when patient having an appointment with a doctor

de: this method is used for deleting a data from database (Adoc table) by selecting the user from jframe table

All importd methods are used for connection to data base but every method connects to its table stored in database

edit_Rooms: this method is used to save room data written by Admins and be stored in database (EmgRoom table)

All check_if_null methods are used for checking if ID In delete jtext field is null or not if null then admin cannot update the data with this id if not data updated

DeleteIDKeyTyped: this method preventing the admin from entering non digits

Pharmacy Section

Pharmacy_Main class

find: this method is used to search for the id of pharmacist in database (FarmacyDoc) written by the user in id jtextfield

updatetable: this method is used for connecting to database where table name is Farmacy

search: this method is used to search for the medicine code in database (Farmacy) written by the user in id jtextfield

update: this method is used to update information that is stored in database previously if the medicine is already added before so this method updates data in database (Farmacy)

showtableActionPerformed: this is a button that when clicked program connects to database and imports all data stored in database in jframe table

addbotActionPreformed: this is a button that when clicked program connects to database and add data written by user in database (Farmacy table)

medcodeKeyReleased: this method is used make the program search automatically when the user finishes typing in the textfield (Medicine Code)

dridKeyReleased: this method is used make the program search automatically when the user finishes typing in the textfield (Dr ID)

UpdatebotActionPreformed: this is a button that when clicked the program connects to database and update information that is already stored in database (Farmacy Table)

View_Stock class

find_ID: this method is used to search for the id and password of pharmacist in database (FarmacyDoc) written by the user in id jtextfield

updatetable: this method is used for connecting in database and importing all stored date in jframe table

showtableActionPerformed: this is a button that when clicked the program first searches for id and password that was written by user in database (FarmacyDoc table) if found then the program connects to database of (Farmacy table) and gets all data and imports it in the jframe table if the user is not found show table button will be unavailable

LogOutMouseClicked: this is a button that directs to Main page

Doctors Section

This is the main class that has the **connectDb** class which is the connection class to the data base.

doctor1 class:

This class provides the main menu page for the doctor's page which connects to the other two pages the Reports and the View patient list page.

ViewPatientsActionPerformed: opens the GUI page for ViewPatients

AddReportsActionPerformed: opens the GUI page for AddReports page

BackActionPerformed: goes to the main page for the doctor's system

Reports class:

This page is for the doctors to add reports for the patients by adding to the data base.

BackButtonActionPerformed: goes back to the main page of the doctors GUI doctor1.

SaveButtonActionPerformed: saves the input data in the text field to the data base the GUI is connected to.

NameFieldActionPerformed: takes the name of the patient from the doctor

IDFieldActionPerformed: takes the ID from the doctor

DateFieldActionPerformed: takes the date where the patient did his check up from the doctor

MedicineFieldActionPerformed: takes the name of the medicine for the patient

ViewPatients class:

The functionality of the class is to view the patient's data from the data base as it connects to it via the connectDb method in the GuiDoctors page

BackActionPerformed: goes back to the main page of the doctors GUI doctor1.

Receptionist Section:

Reception Main Page:

This Page have the main functions that Recpetionist deals with

- Add New Patient

- Patients Research

- View Patient Report

- Book Appointment

- View Doctors

- Book Surgery

- Call Ambulance

Add New Patient: in this page there are functions that's used to sign up new patients and take their data.

Patients Search: in this page there're functions that preview the table of patients and allow you to search for a particular patient with his National Identification

View Patient Report: Here there're functions that preview the report data table for patients also function for searching to get a particular and the specified report for patient with his National ID

Book Appointment: this page should help the receptionist with booking for patients, there's a table with doctors data also provided with searching field, when the receptionist enters the doctor ID , the other fields will be filled automatically this also happens with the searching field for the patient

When receptionist enters his national id , other fields filled automatically.

IF one of the IDs for Patient Or Doctor wasn't find or both of them weren't, a message will pop up to tell the receptionist.With the option to add new patient if the patient wasn't registered into the system before.

View Doctors : it previews the Doctors recorded table.

Book Surgery : this page should help the receptionist to book surgery rooms for patient with table of the status of every room

Also supported with searching field which shows the status of the room.

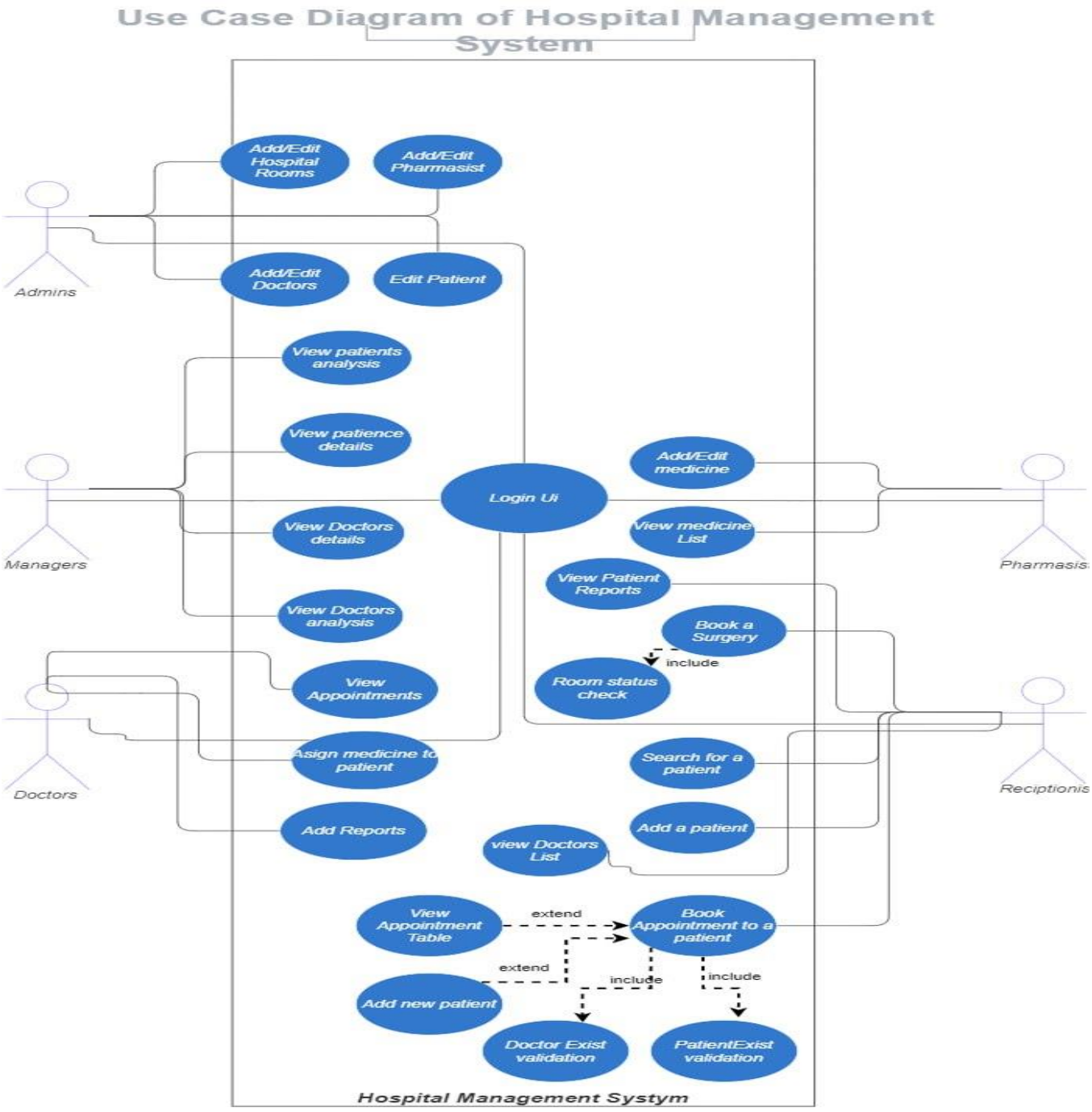
Call Ambulance : this page has the function to get the required data for calling ambulance.

Manager Section

Managers functions: importDoctor: this method imports data of doctors from data base and display it in jtable and make analysis and operations on it. importPatient: this method imports data of patients from data base and display it in jtable and make analysis and operations on it. BarChartActionPerformed: display the numbers of both doctors and patients in format of bar-chart. piechartActionPerformed: display the numbers of both doctors and patients in format of pie-chart.

Diagrams

Use Case Diagram



Use Cases Description

Receptionist use cases

user case name: Add New patient
participating actors : Receptionist
entry conditions: Patient Wants to register
exit condition: patient data registered
flow of events: Receptionist logins
takes data from patient
Enter data in the form
Register the patient
enters the data into the
Exceptions: none
Special Requirements : none

user case name: search for a patient
participating actors : Receptionist
entry conditions: Receptionist wants to search patient with his national id
exit condition: patient data found
flow of events: Receptionist logins
Enter National ID in the field
Search , get the data.
Exceptions: none
Special Requirements : none

user case name: View Patient Report
participating actors : Receptionist
entry conditions: Receptionist wants to view patient reports with his national id
exit condition: patient data found|
flow of events: Receptionist logins
Enter National ID in the field
Search , get the data.
Exceptions: none
Special Requirements : none

user case name: Book an appointment
participating actors : Receptionist
entry conditions: Receptionist wants to book appointment for a patient
exit condition: booking done, patient not found
flow of events: Receptionist logins
Enter Patient Data

Enter Doctors Data
book
Includes the patient and doctors exist validation
If patinets isn't registered , system asks to exit and open registration page or checking data
If doctor not found, sys asks to check data.
If data found , book done.
Exceptions: none
Special Requirements : none

user case name: Book a surgery
participating actors : Receptionist
entry conditions: Receptionist wants to book a surgery room for a patient
exit condition: booking done, room isn't available (full)
flow of events: Receptionist logins
Enter Patient Data
Enter Room Number
book
Includes the room status validation
if room is full , sys asks to enter new room
If data found , book done.
Exceptions: none
Special Requirements : none

Manager Use cases

user case name: View patient/doctor data and analysis using Bar-chart and pie chart.
participating actors : Manager
entry conditions: Manager login
exit condition: Manager logout
flow of events:
-Manager logins
-view data of all patients/doctors in the hospital.
-view analysis of the patients/doctors.
Exceptions: manager can view any data except sensitive information about patient and doctors.
Special Requirements : none

Doctor Use cases

use case name: Add Reports
participating actors :doctors
entry conditions:doctor wants to add a report about a patient to the database
exit condition:none
flow of events:add name of patient,date,select medicine,ID and add report
Exceptions: none
Special Requirements :none

use case name:view doctors details
participating actors :doctors

entry conditions:doctors email is valid
exit condition:back button or N/A email
flow of events:when the doctor enter his email he will be able to access the doctors main menu
Exceptions: none
Special Requirements :none

use case name:View Patients
participating actors :doctors
entry conditions:doctor wants to view patients appointments date
exit condition:none
flow of events:when the doctor select the View patients option from the main menu he will be able to view the patients appointments list
Exceptions: none
Special Requirements :none

Admin Use cases

user case name: Add/Edit Doctor
participating actors: Admins
entry conditions: Admin can add/edit doctors' data
exit condition: Doctor added/edited
flow of events: Admins logins
takes data from HR (Accepted Doctors)
Exceptions: none
Special Requirements : none

user case name: Edit Patient
participating actors: Admins
entry conditions: Admin can edit Patients' data
exit condition: Patient edited
flow of events:
Admins logins
takes data from Patient database
Exceptions: none
Special Requirements :none

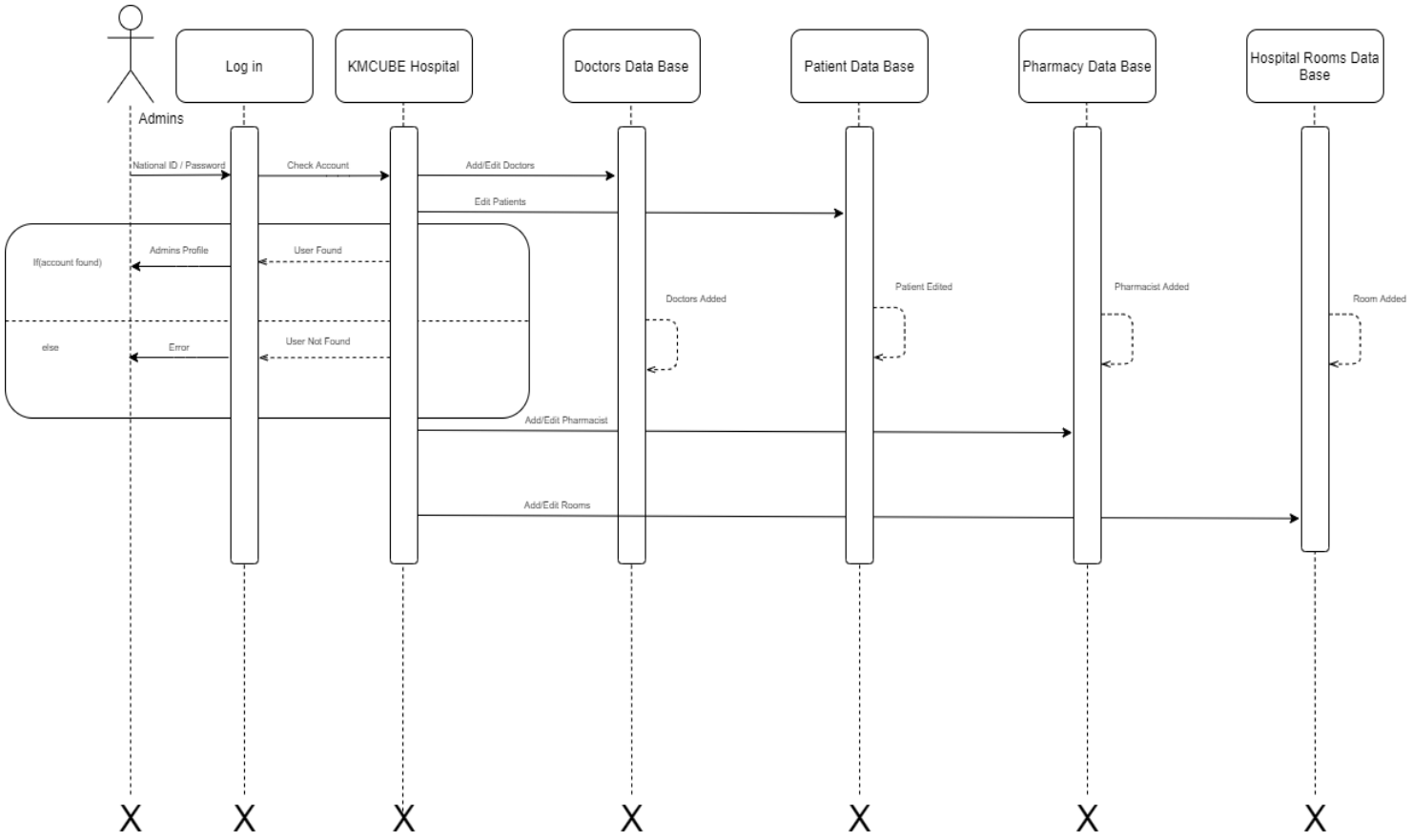
user case name: Add/Edit Pharmacist
participating actors: Admins
entry conditions: Admin can add/edit Pharmacists' data
exit condition: Pharmacist added/edited
flow of events: Admins logins
takes data from HR (Accepted Pharmacists)
Exceptions: none
Special Requirements : none

Pharmacist Use case

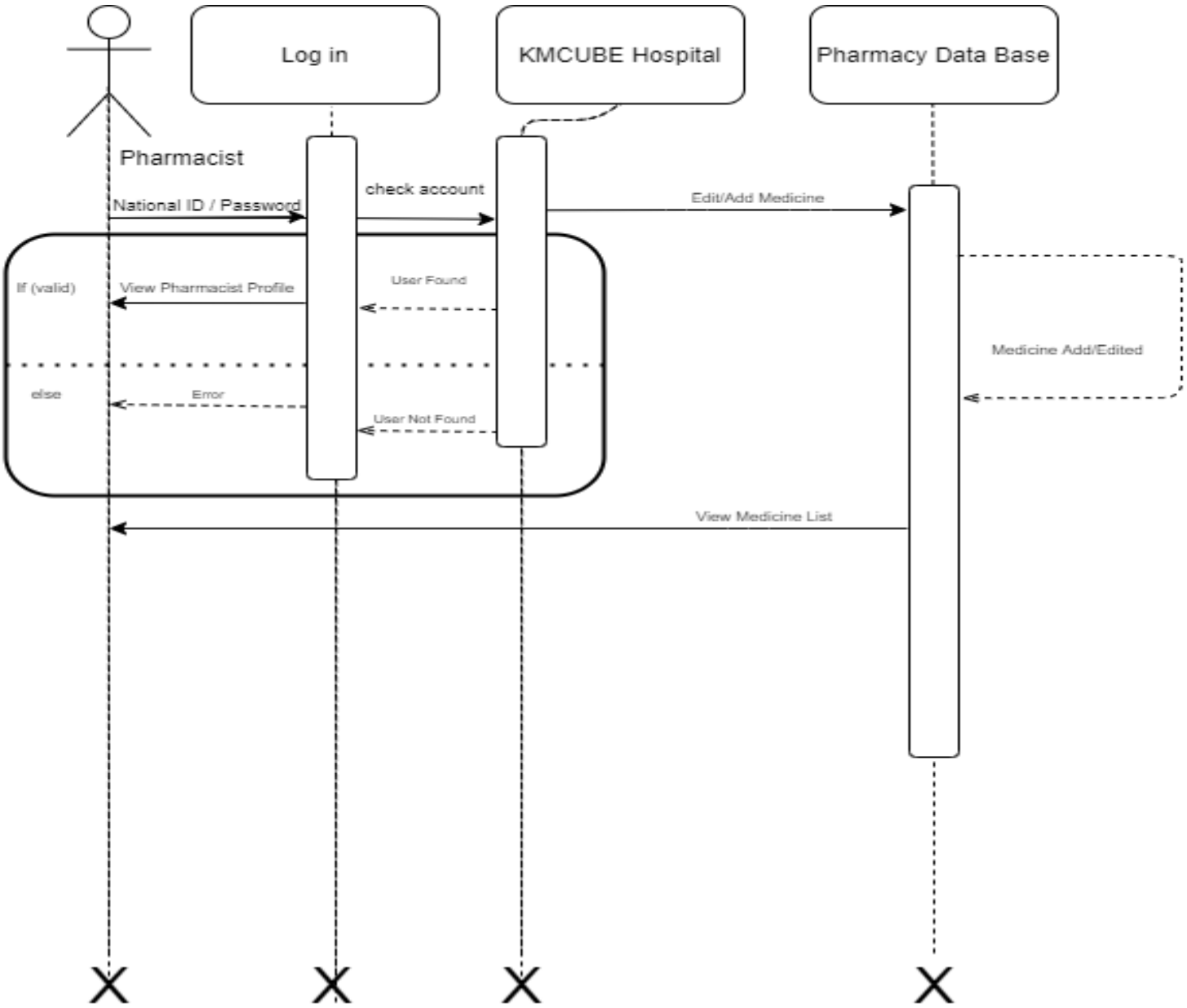
user case name: Add/Edit/view Medicine
participating actors: Pharmacist
entry conditions: Pharmacist can add/edit Medicines' data
exit condition: Medicine added/edited
flow of events:
Pharmacist logins
takes data from Medicine DataBase
Exceptions: none
Special Requirements :none

System Sequence Diagrams

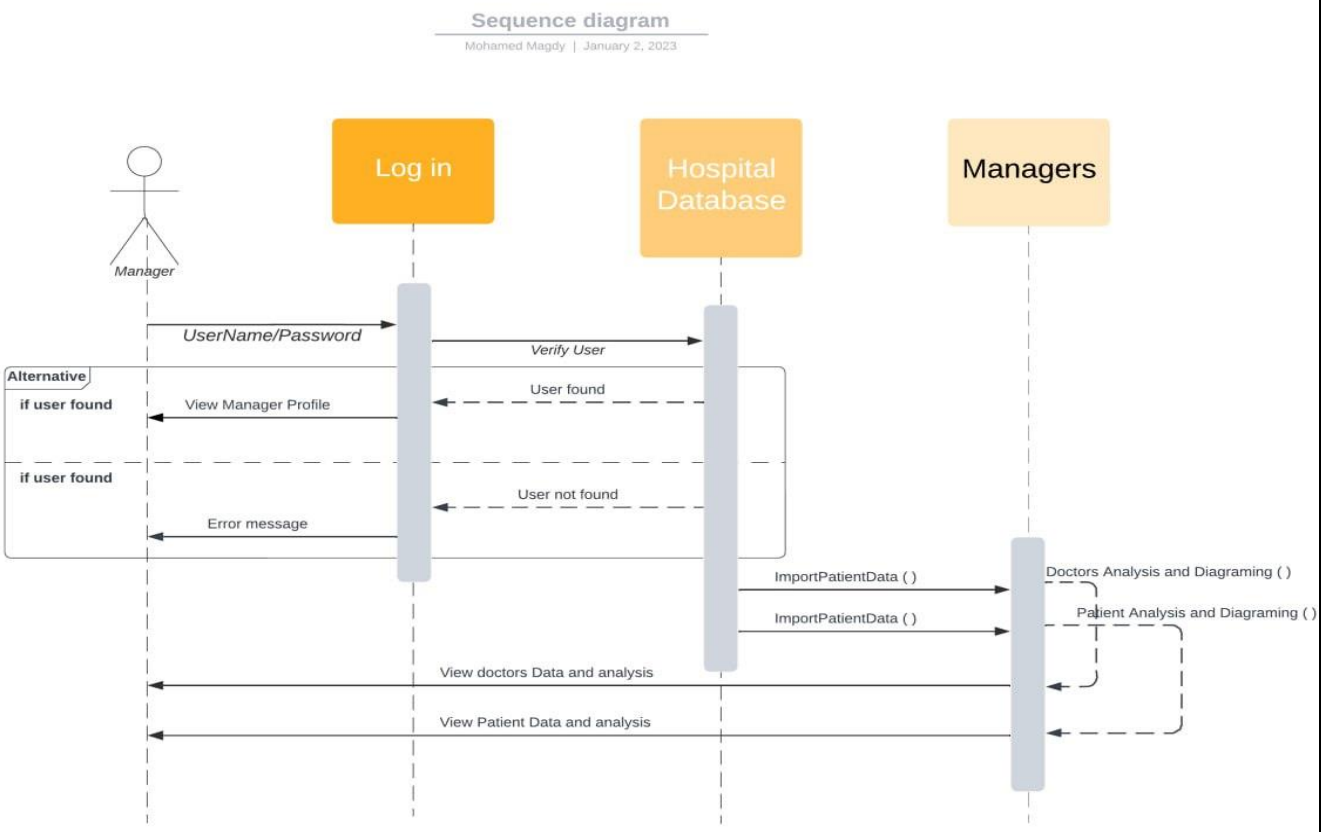
Admins



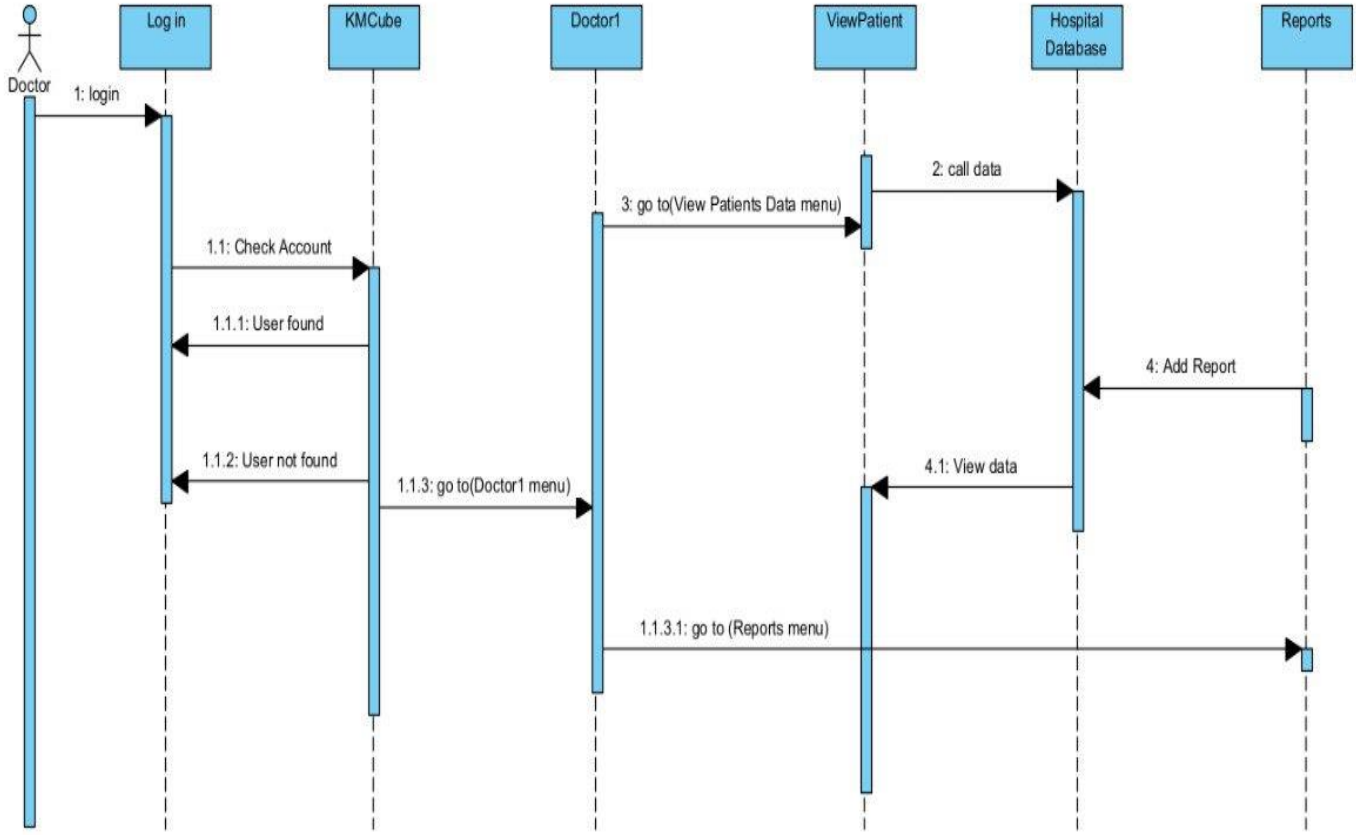
Pharmacy


Log in
KMCUBE Hospital
Pharmacy Data Base
Pharmacist
National ID / Password
check account
Edit/Add Medicine
If (valid)
View Pharmacist Profile
User Found
Medicine Add/Edited
else
Error
User Not Found
View Medicine List

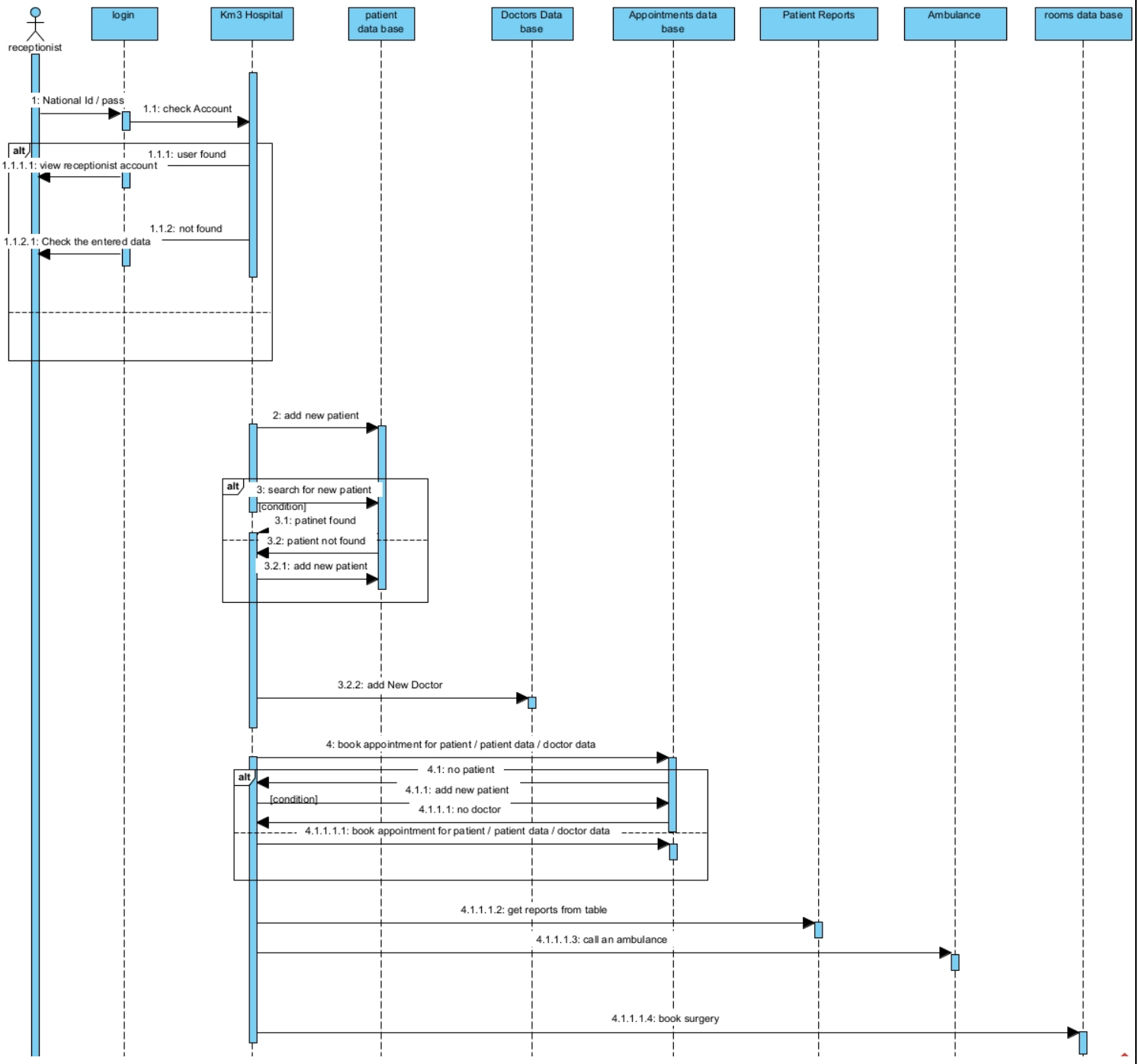
Managers


Sequence diagram
Mohamed Magdy | January 2, 2023
Log in
Hospital Database
Managers
Manager
UserName/Password
Verify User
Alternative
if user found
View Manager Profile
User found
if user found
Error message
User not found
ImportPatientData ( )
ImportPatientData ( )
Doctors Analysis and Diagraming ( )
Patient Analysis and Diagraming ( )
View doctors Data and analysis
View Patient Data and analysis

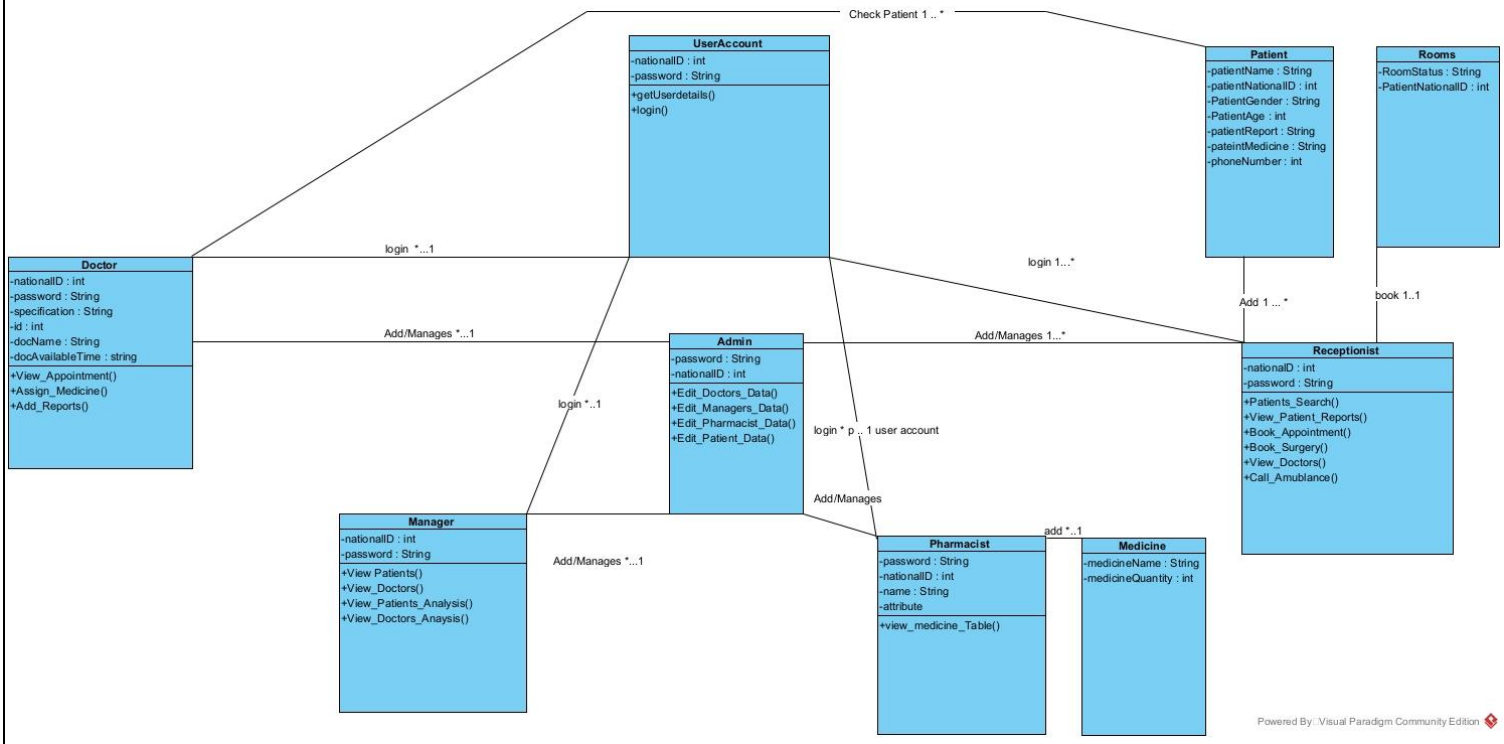
Doctors


Doctor
Log in
KMCube
Doctor1
ViewPatient
Hospital Database
Reports
1: login
2: call data
3: go to(View Patients Data menu)
1.1: Check Account
1.1.1: User found
4: Add Report
1.1.2: User not found
1.1.3: go to(Doctor1 menu)
4.1: View data
1.1.3.1: go to (Reports menu)

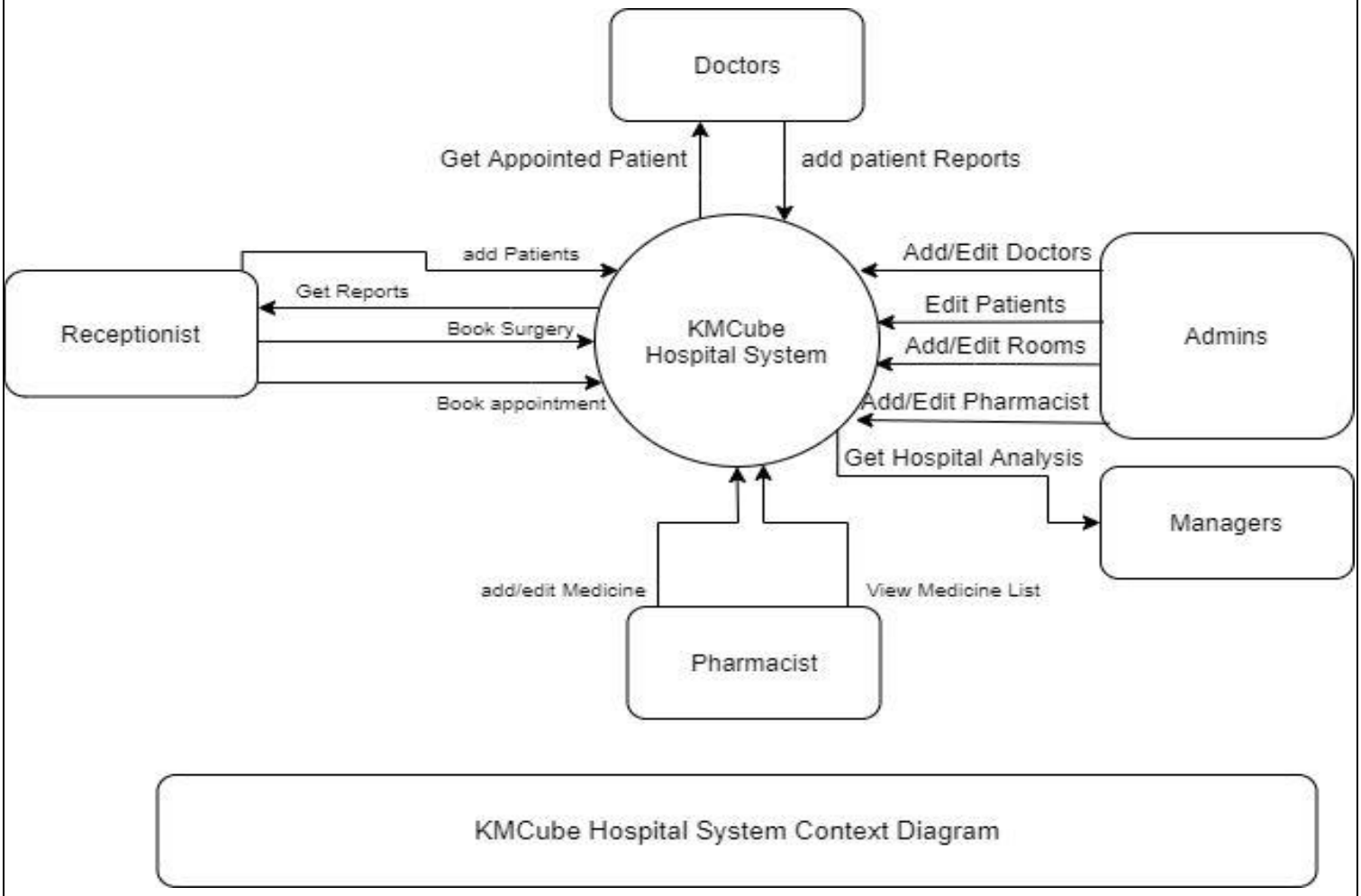
Reception


receptionist
login
Km3 Hospital
patient data base
Doctors Data base
Appointments data base
Patient Reports
Ambulance
rooms data base
1: National Id / pass
1.1: check Account
alt
1.1.1: user found
1.1.1.1: view receptionist account
1.1.2: not found
1.1.2.1: Check the entered data
2: add new patient
alt
3: search for new patient
[condition]
3.1: patinet found
3.2: patient not found
3.2.1: add new patient
3.2.2: add New Doctor
4: book appointment for patient / patient data / doctor data
alt
4.1: no patient
4.1.1: add new patient
[condition]
4.1.1.1: no doctor
4.1.1.1.1: book appointment for patient / patient data / doctor data
4.1.1.1.2: get reports from table
4.1.1.1.3: call an ambulance
4.1.1.1.4: book surgery

Class Diagram



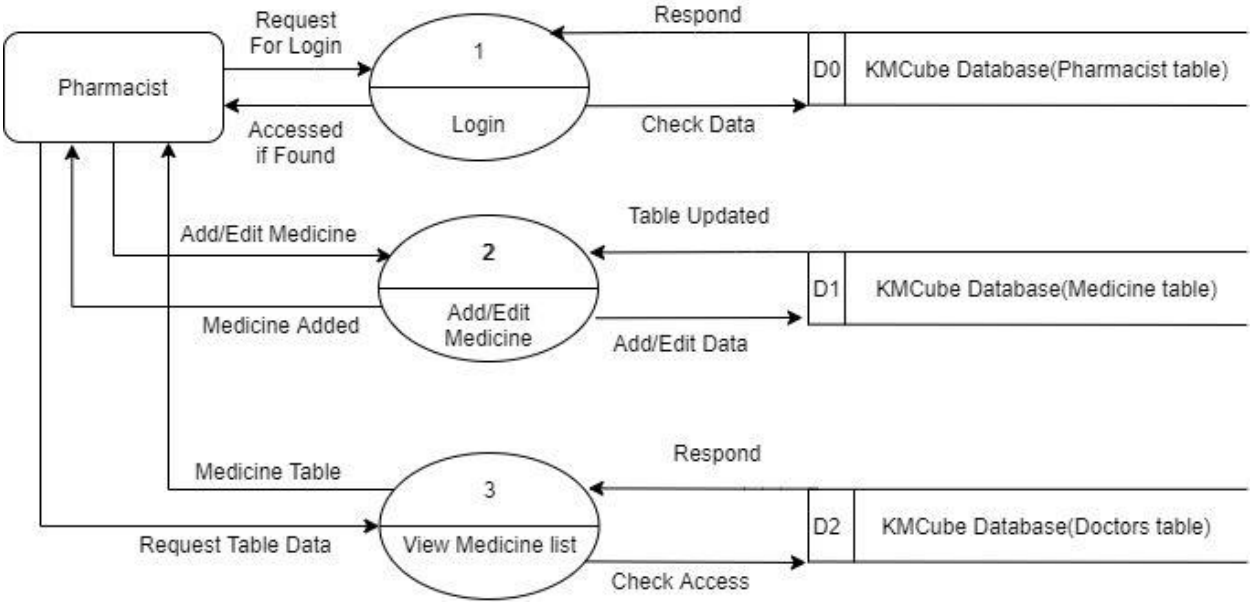
DFD Diagram context

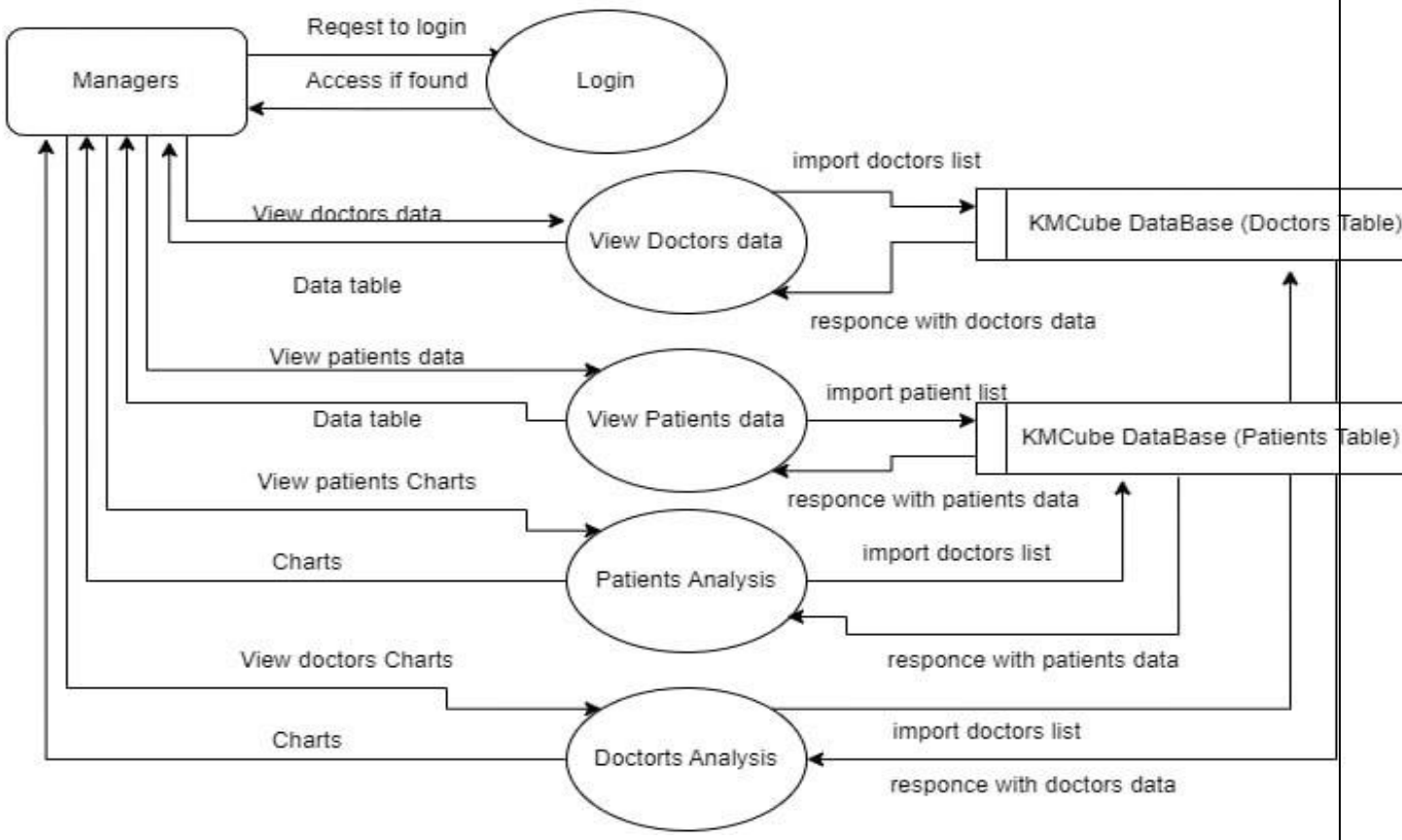


KMCube Hospital System Context Diagram

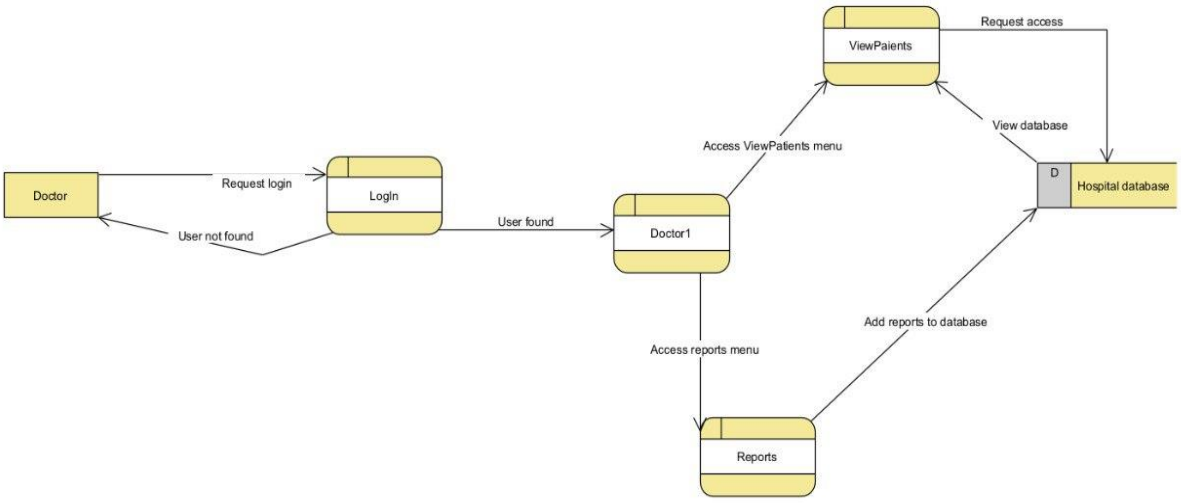
Admins DFD

Pharmacist DFD



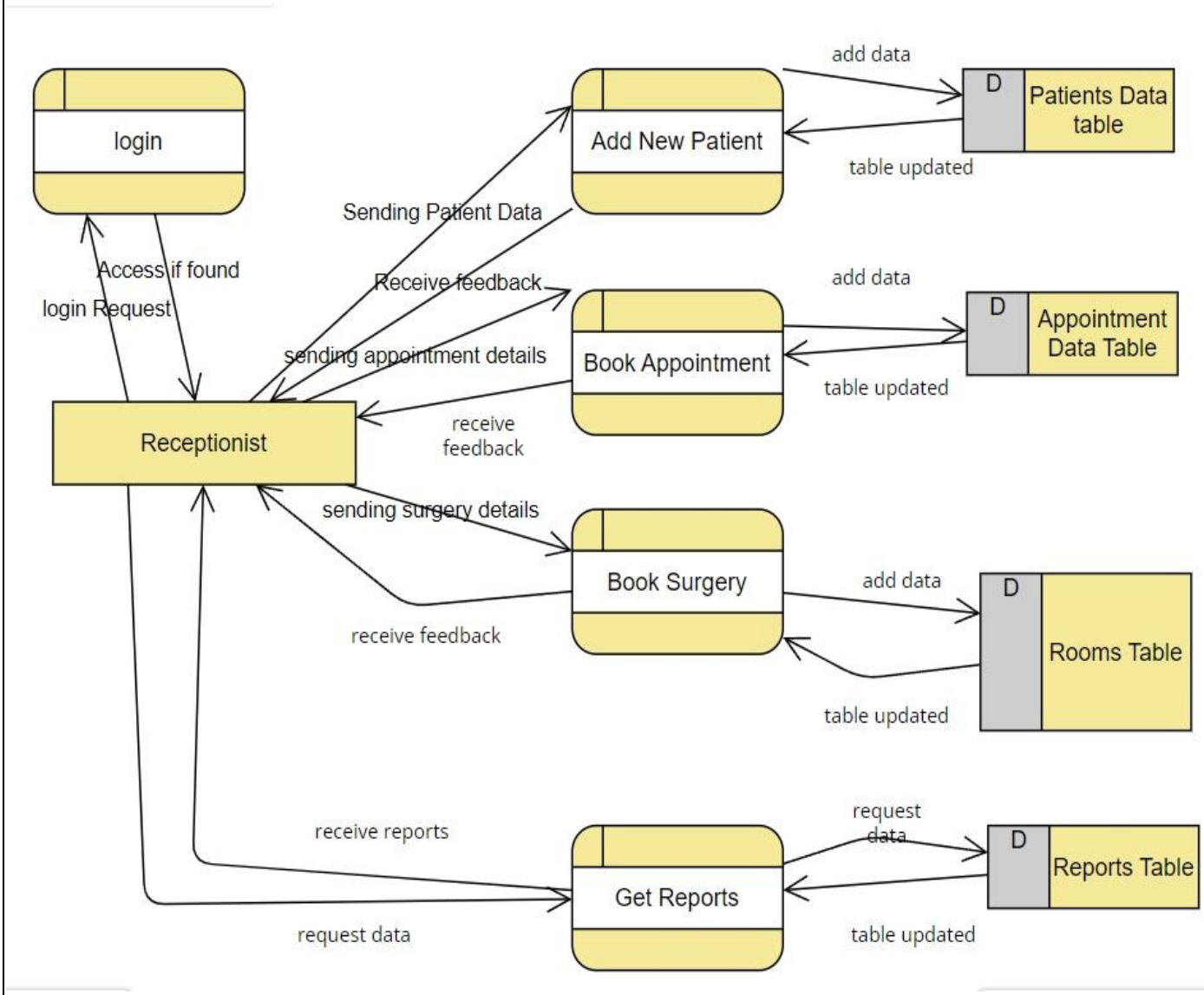


Managers DFD

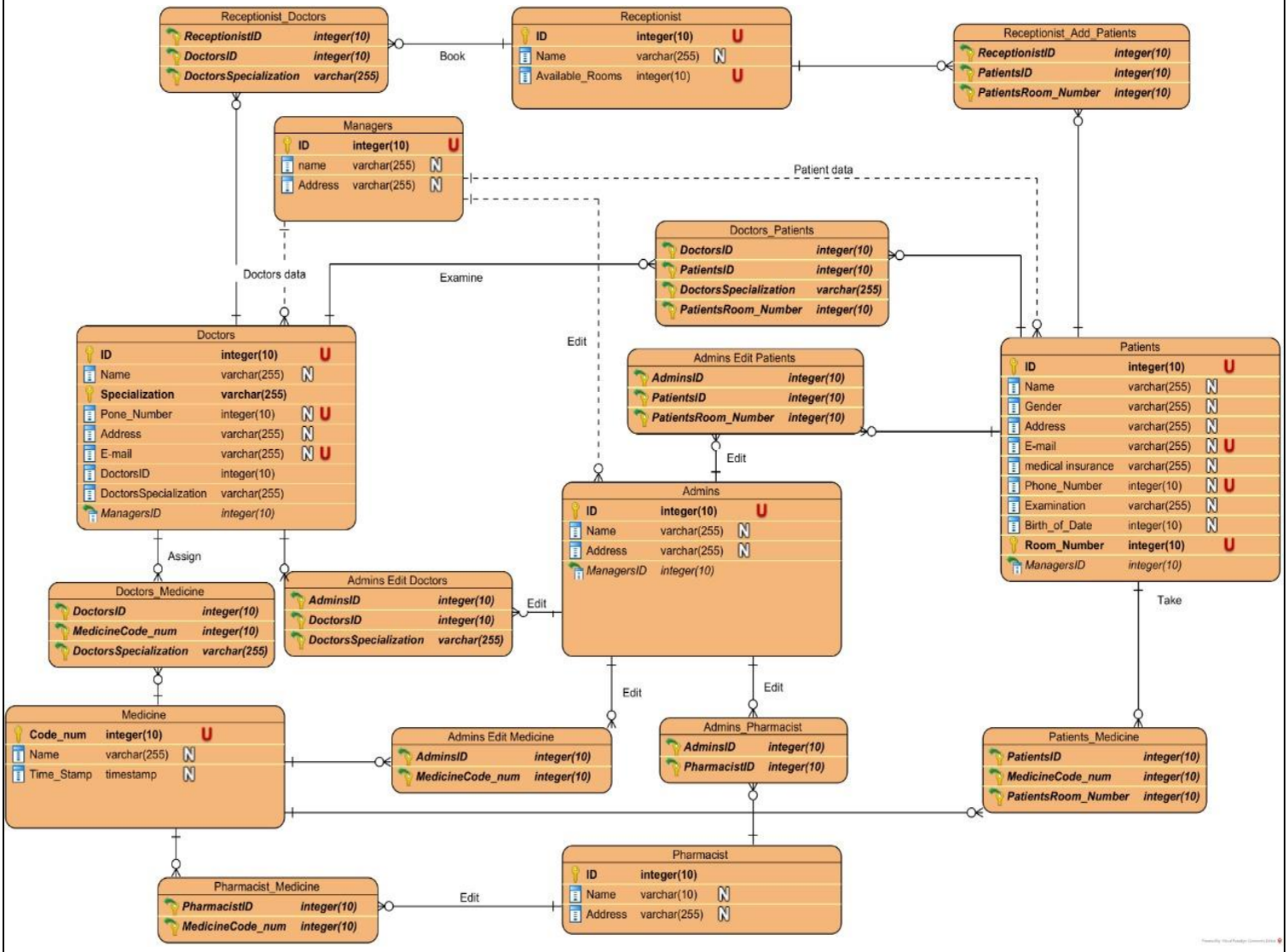


Doctors DFD

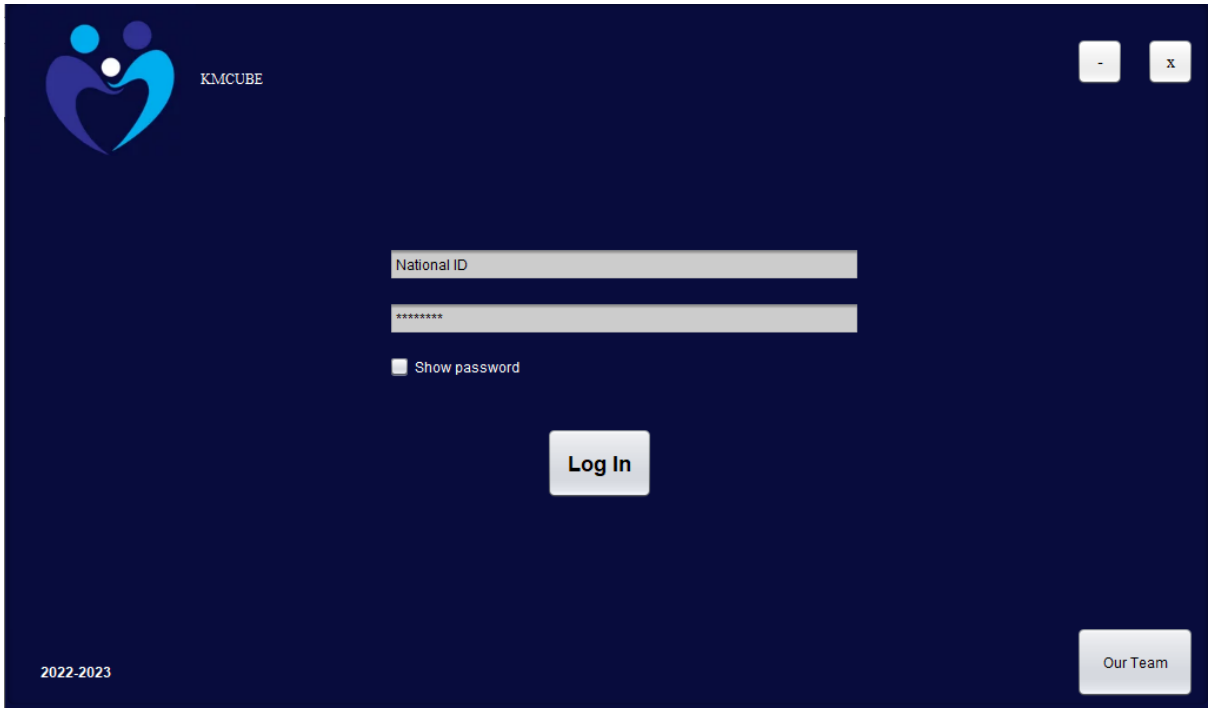
Reception DFD



ERD


Receptionist_Doctors
ReceptionistID integer(10)
DoctorsID integer(10)
DoctorsSpecialization varchar(255)
Book
Receptionist
ID integer(10) U
Name varchar(255) N
Available_Rooms integer(10) U
Receptionist_Add_Patients
ReceptionistID integer(10)
PatientsID integer(10)
PatientsRoom_Number integer(10)
Managers
ID integer(10) U
name varchar(255) N
Address varchar(255) N
Patient data
Doctors_Patients
DoctorsID integer(10)
PatientsID integer(10)
DoctorsSpecialization varchar(255)
PatientsRoom_Number integer(10)
Doctors data
Examine
Doctors
ID integer(10) U
Name varchar(255) N
Specialization varchar(255)
Pone_Number integer(10) N U
Address varchar(255) N
E-mail varchar(255) N U
DoctorsID integer(10)
DoctorsSpecialization varchar(255)
ManagersID integer(10)
Edit
Admins Edit Patients
AdminsID integer(10)
PatientsID integer(10)
PatientsRoom_Number integer(10)
Patients
ID integer(10) U
Name varchar(255) N
Gender varchar(255) N
Address varchar(255) N
E-mail varchar(255) N U
medical insurance varchar(255) N
Phone_Number integer(10) N U
Examination varchar(255) N
Birth_of_Date integer(10) N
Room_Number integer(10) U
ManagersID integer(10)
Edit
Admins
ID integer(10) U
Name varchar(255) N
Address varchar(255) N
ManagersID integer(10)
Assign
Doctors_Medicine
DoctorsID integer(10)
MedicineCode_num integer(10)
DoctorsSpecialization varchar(255)
Admins Edit Doctors
AdminsID integer(10)
DoctorsID integer(10)
DoctorsSpecialization varchar(255)
Edit
Take
Edit
Edit
Medicine
Code_num integer(10) U
Name varchar(255) N
Time_Stamp timestamp N
Admins Edit Medicine
AdminsID integer(10)
MedicineCode_num integer(10)
Admins_Pharmacist
AdminsID integer(10)
PharmacistID integer(10)
Patients_Medicine
PatientsID integer(10)
MedicineCode_num integer(10)
PatientsRoom_Number integer(10)
Pharmacist
ID integer(10)
Name varchar(10) N
Address varchar(255) N
Pharmacist_Medicine
PharmacistID integer(10)
MedicineCode_num integer(10)
Edit

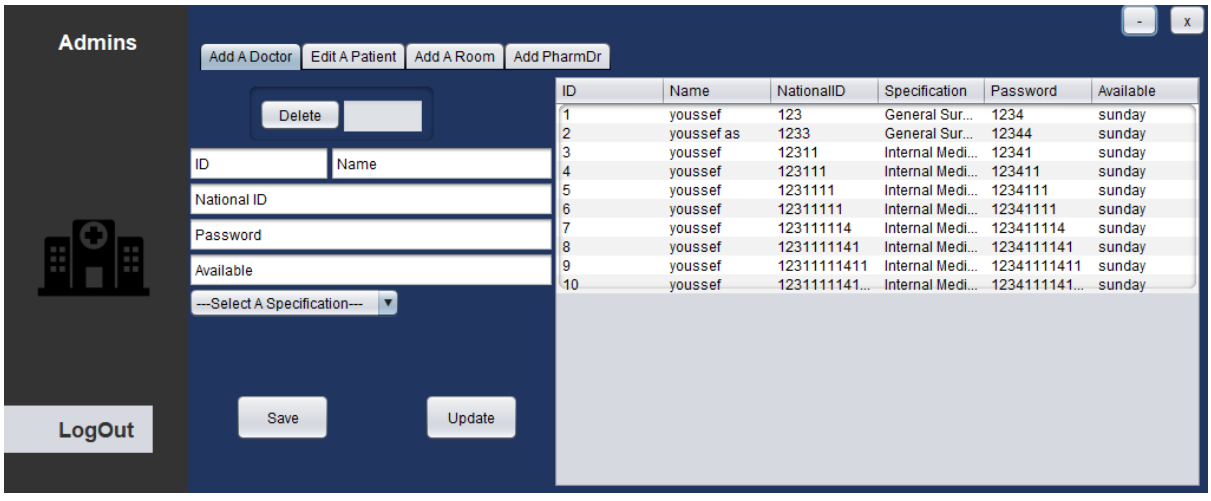
Project's Snapshots



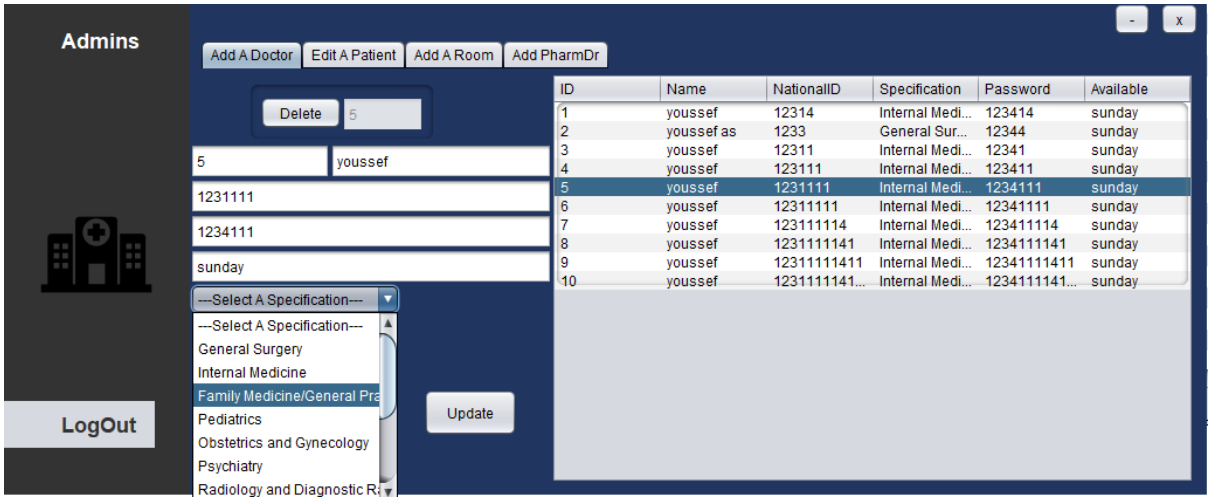
Main Page

Team Page

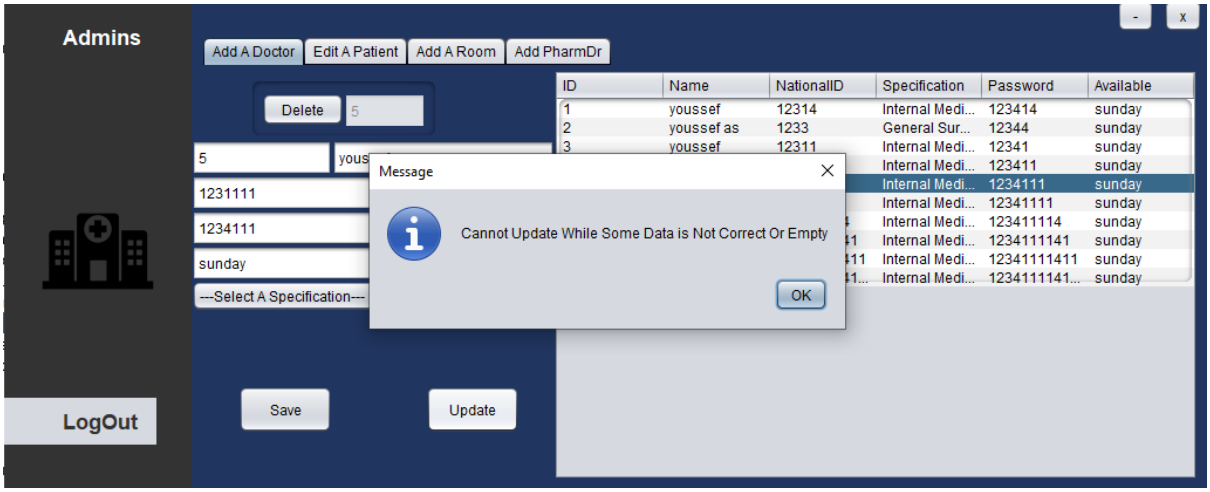
Logo Page



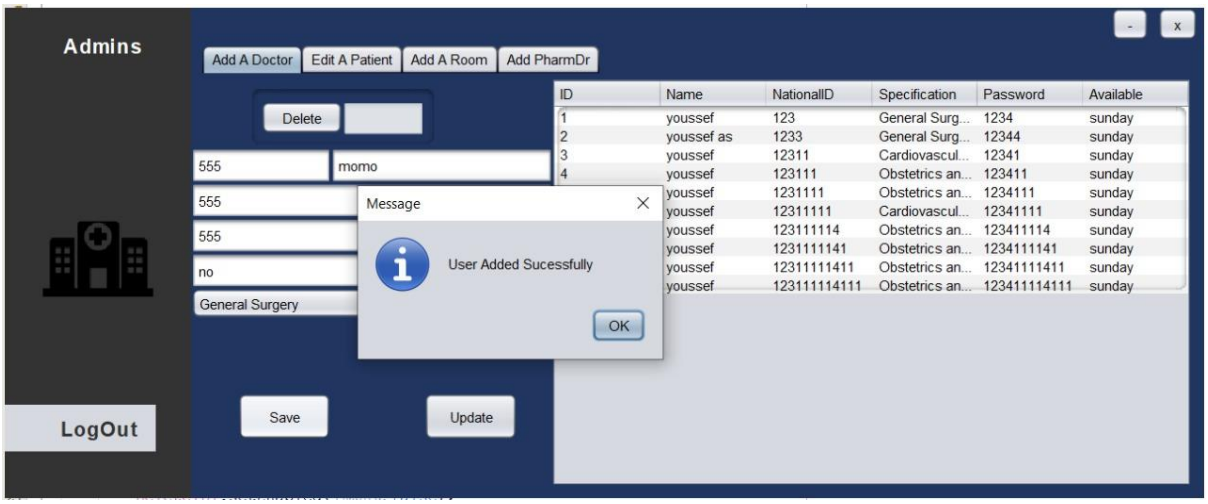
Admins Page 1



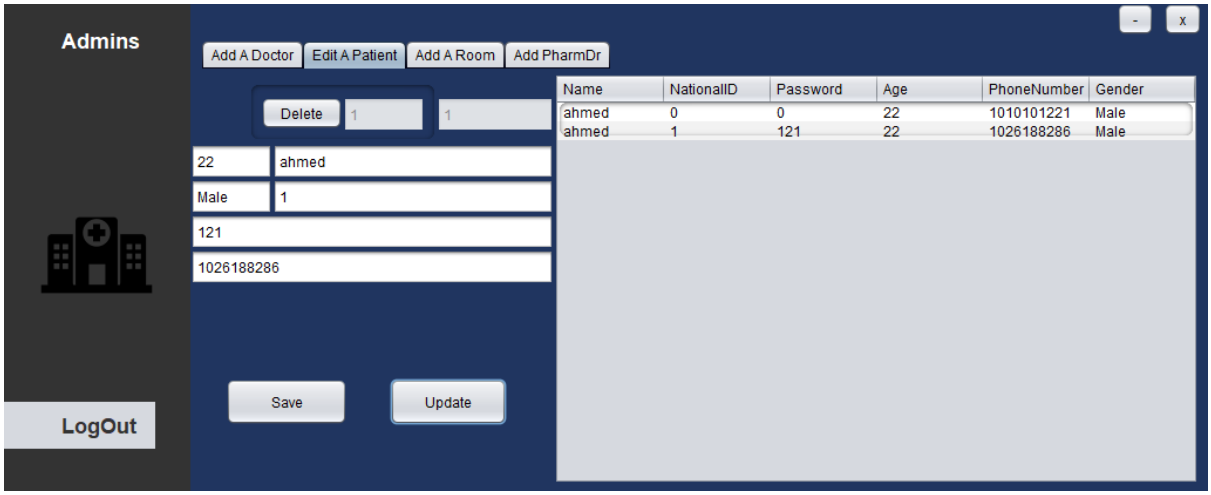
Admins Page 1



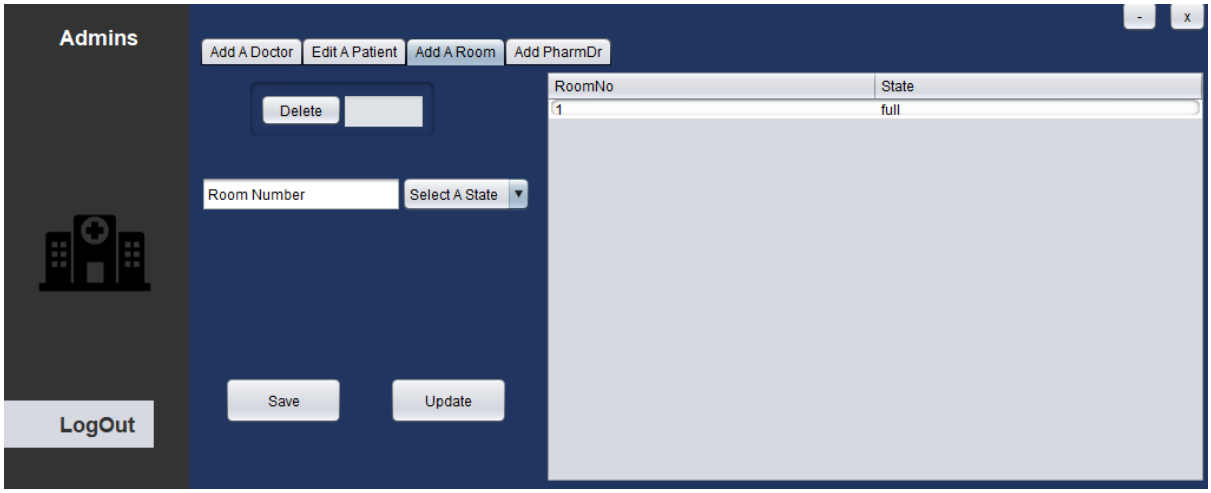
Admin Page 1 – adding doctor



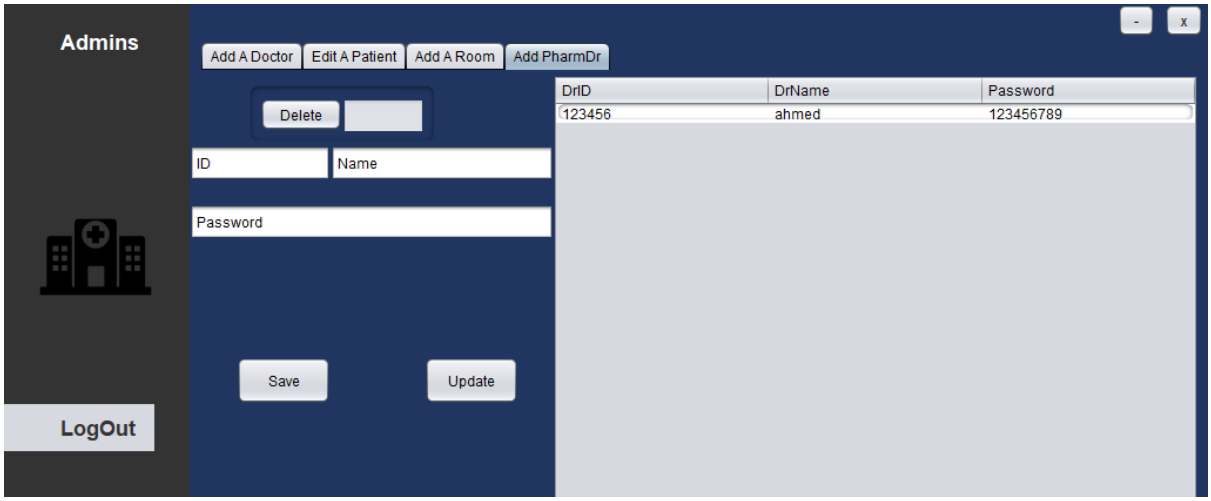
Admin Page 1 – adding doctor



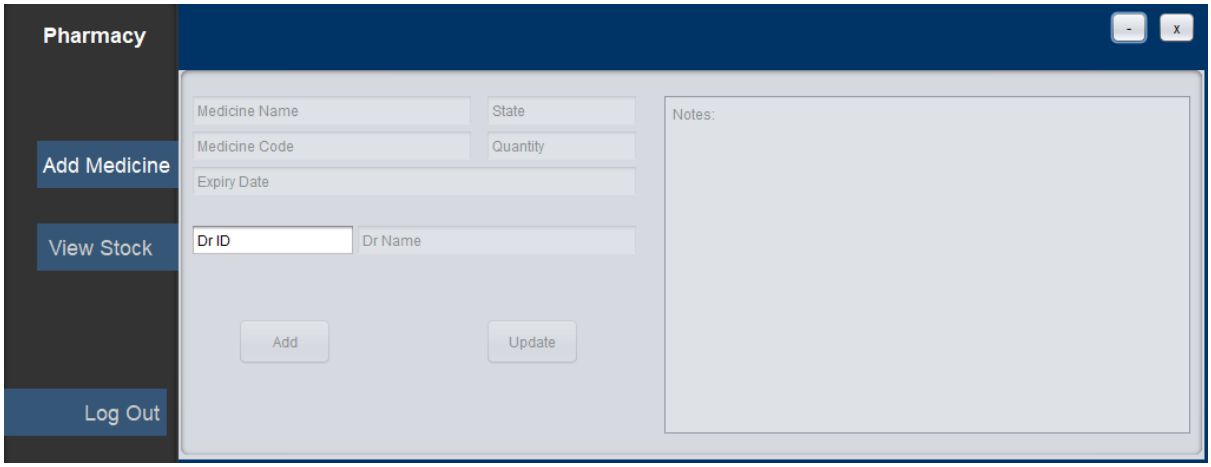
Admins Page 2



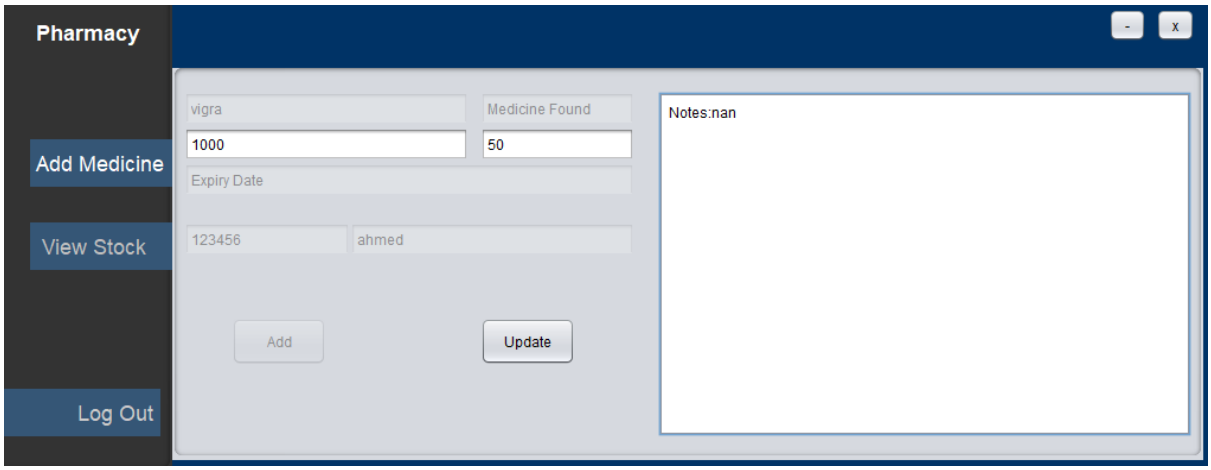
Admins Page 3



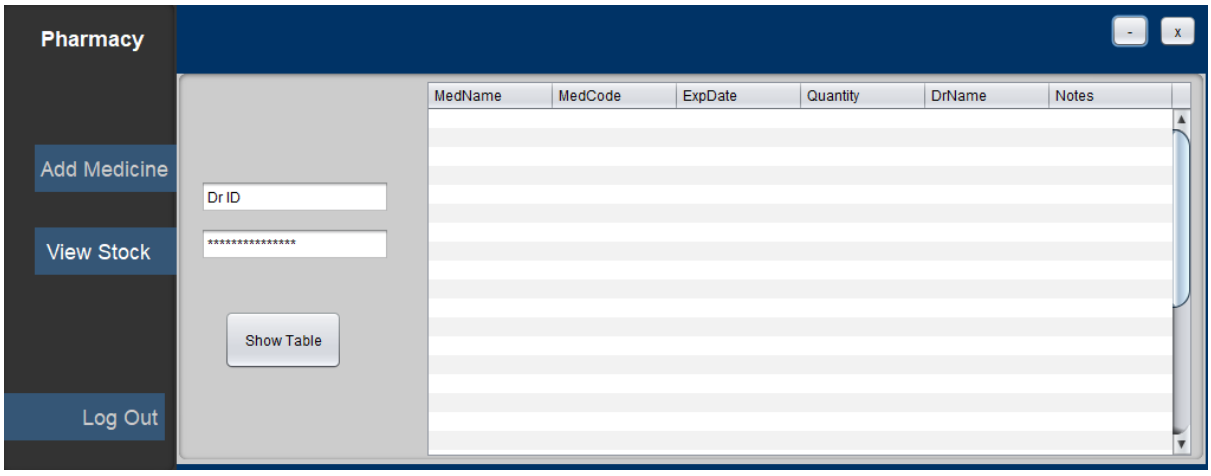
Admins Page 4



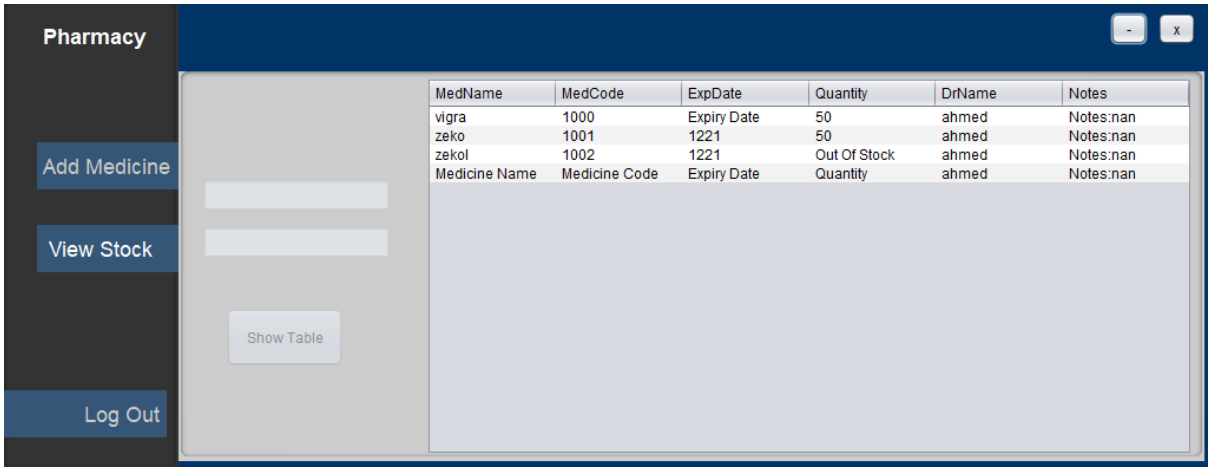
Pharmacy Page 1 - Add medicine



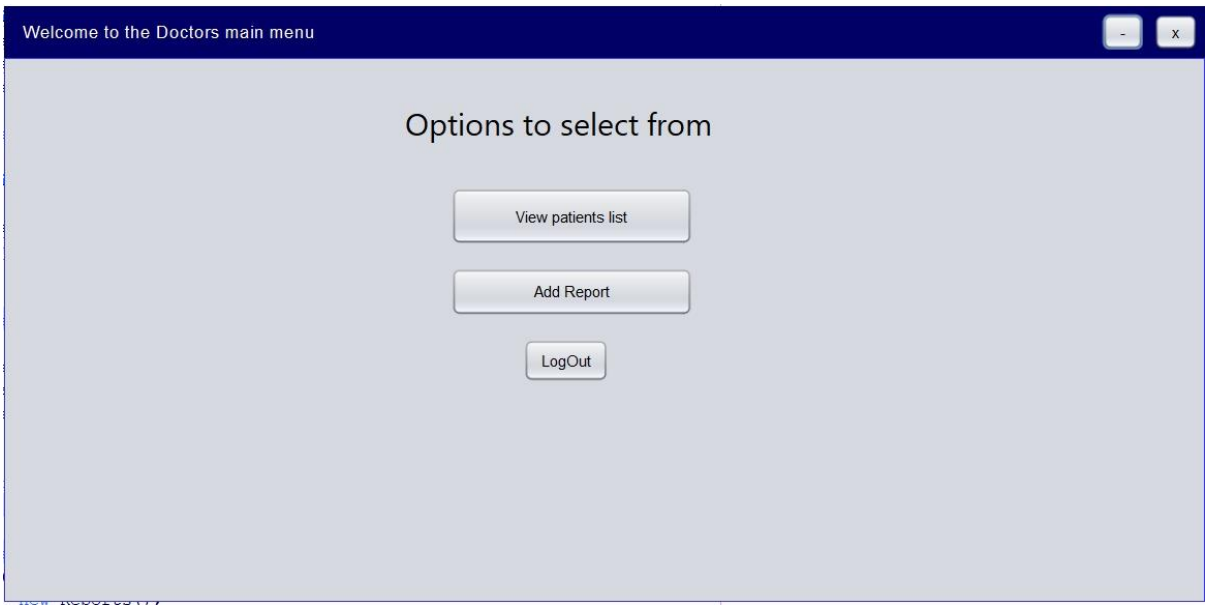
Pharmacy Page 1 - Add medicine



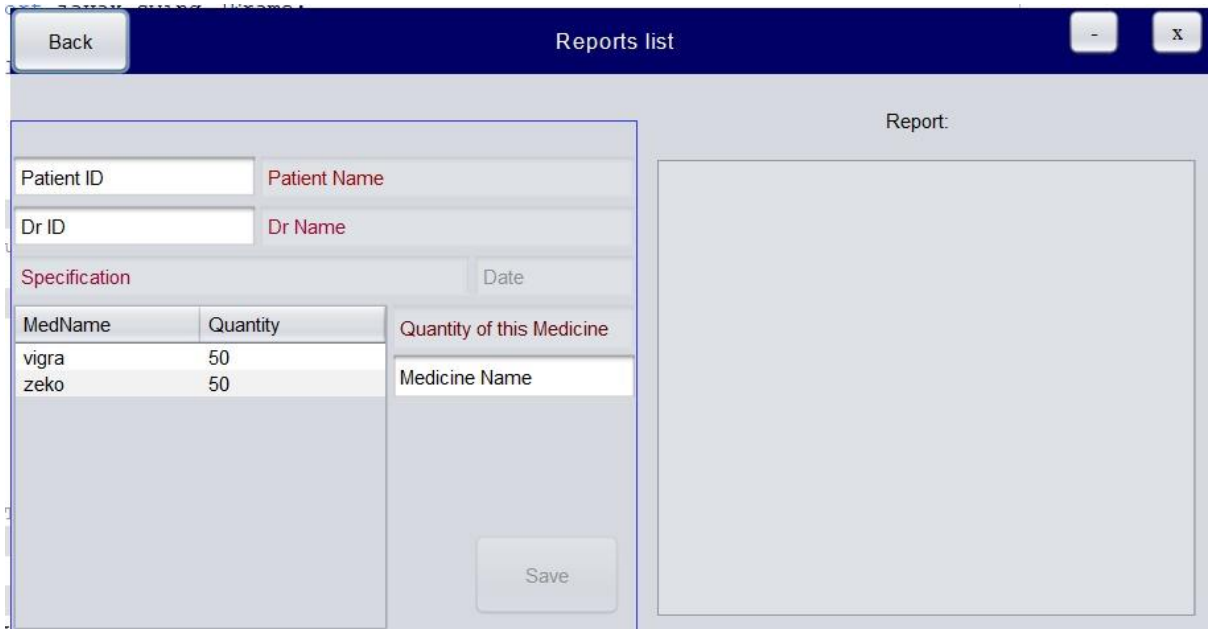
Pharmacy Page 2 – View Stock



Pharmacy Page 2 – View Stock



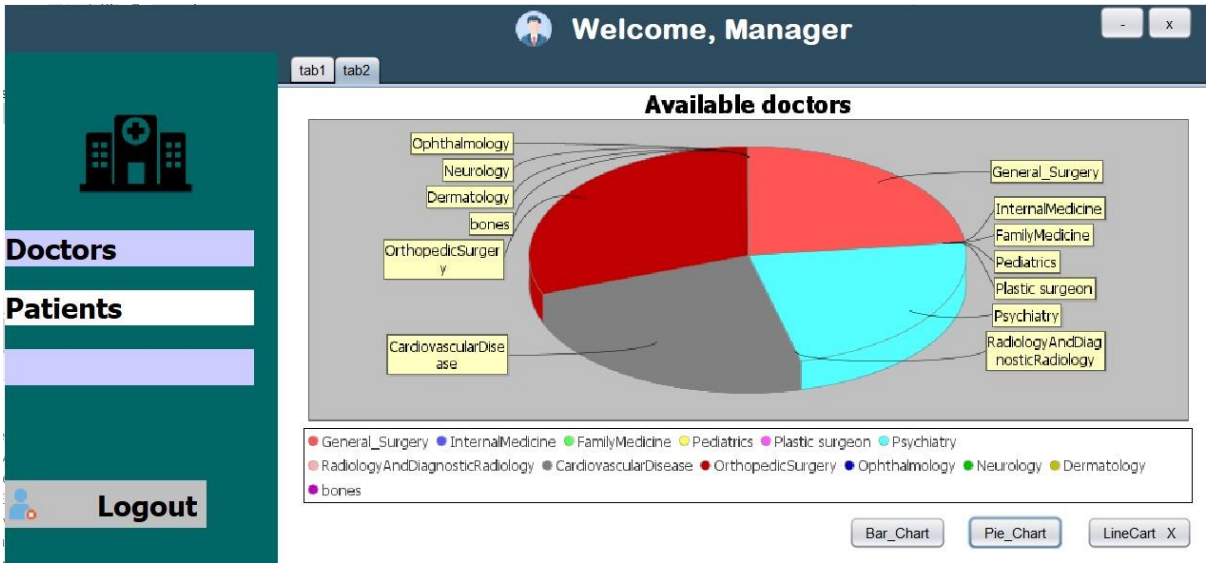
Doctors Page 1



Doctors Page 2

Managers page 1

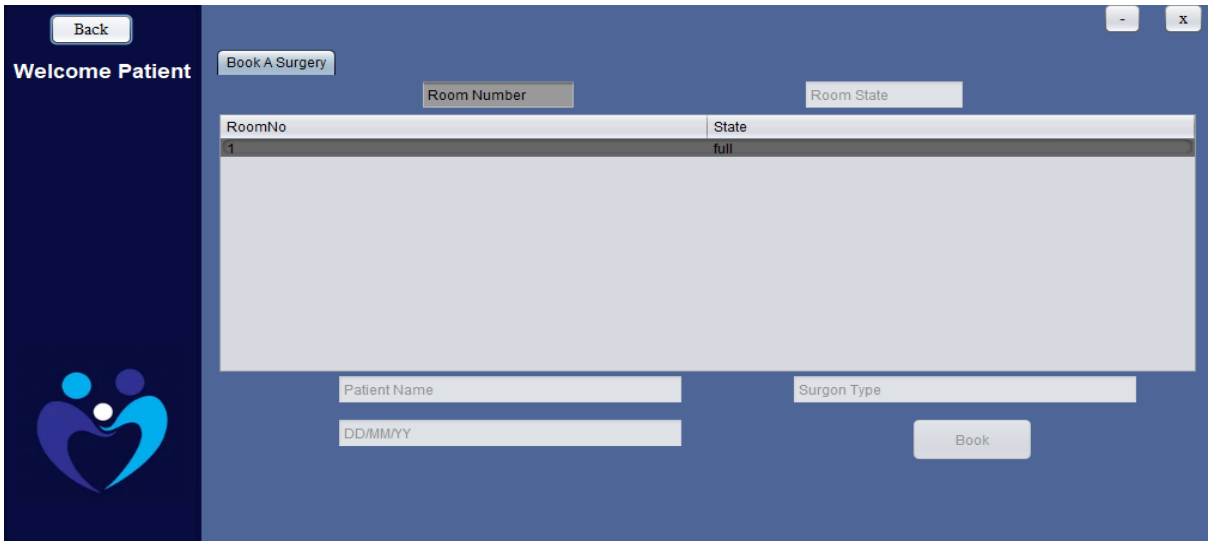
Managers page 2



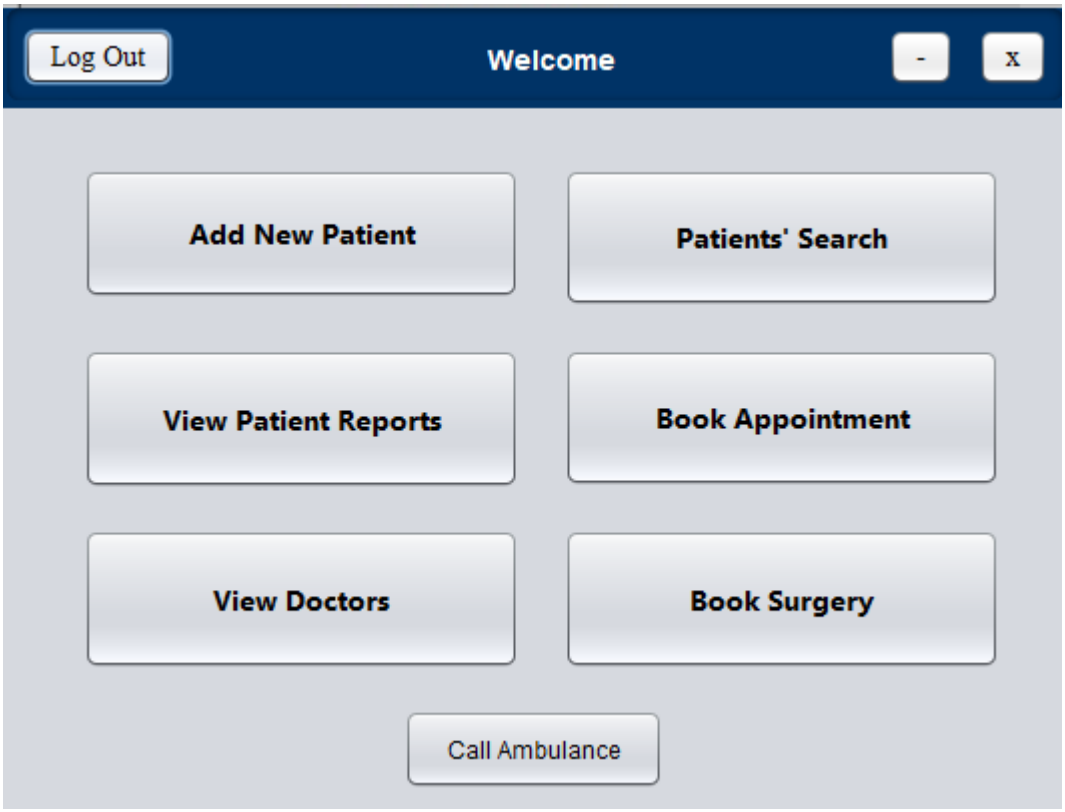
Managers page 3

Call An Ambulance Page

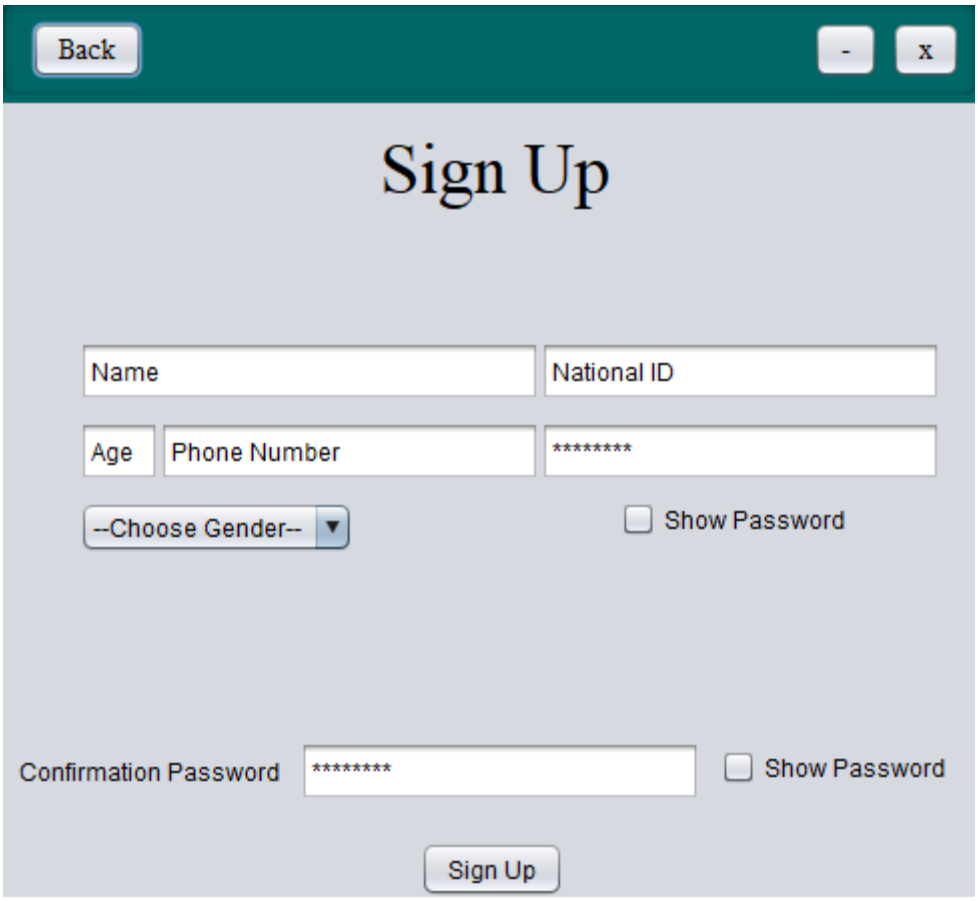
Reception Page 1



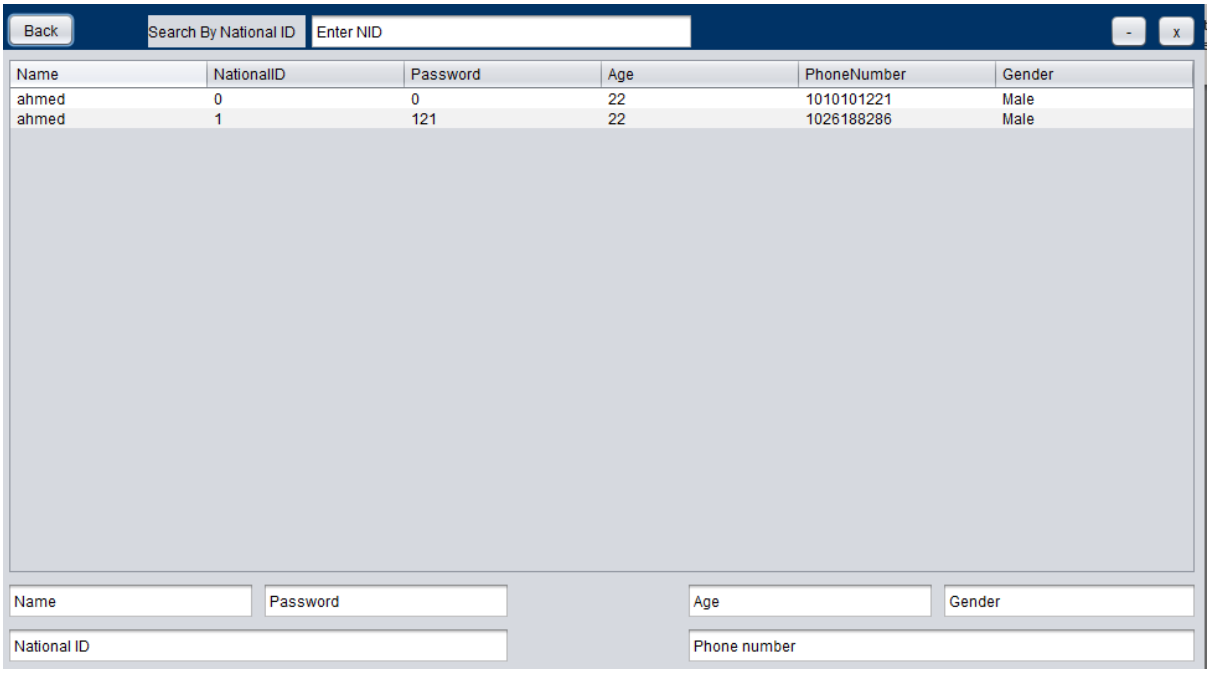
Reception Page 2



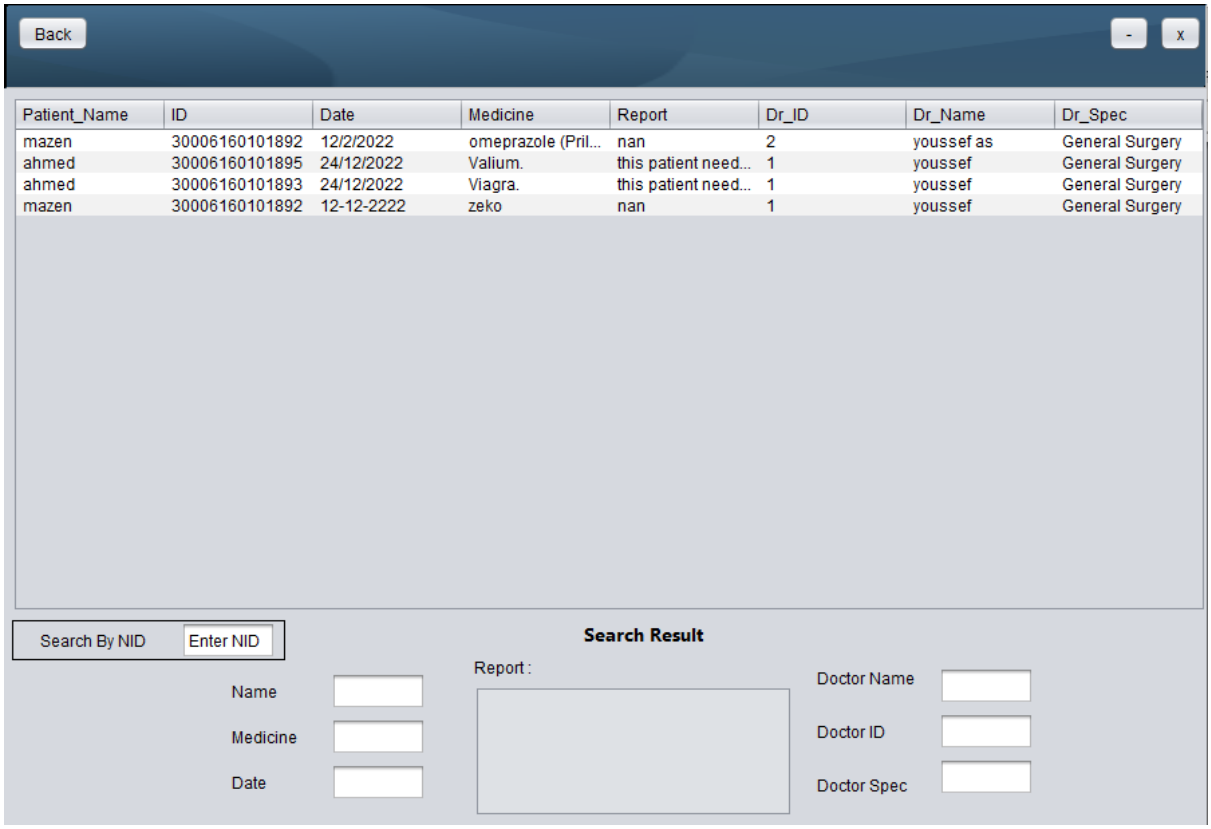
Reception Page 3



Reception Page 4

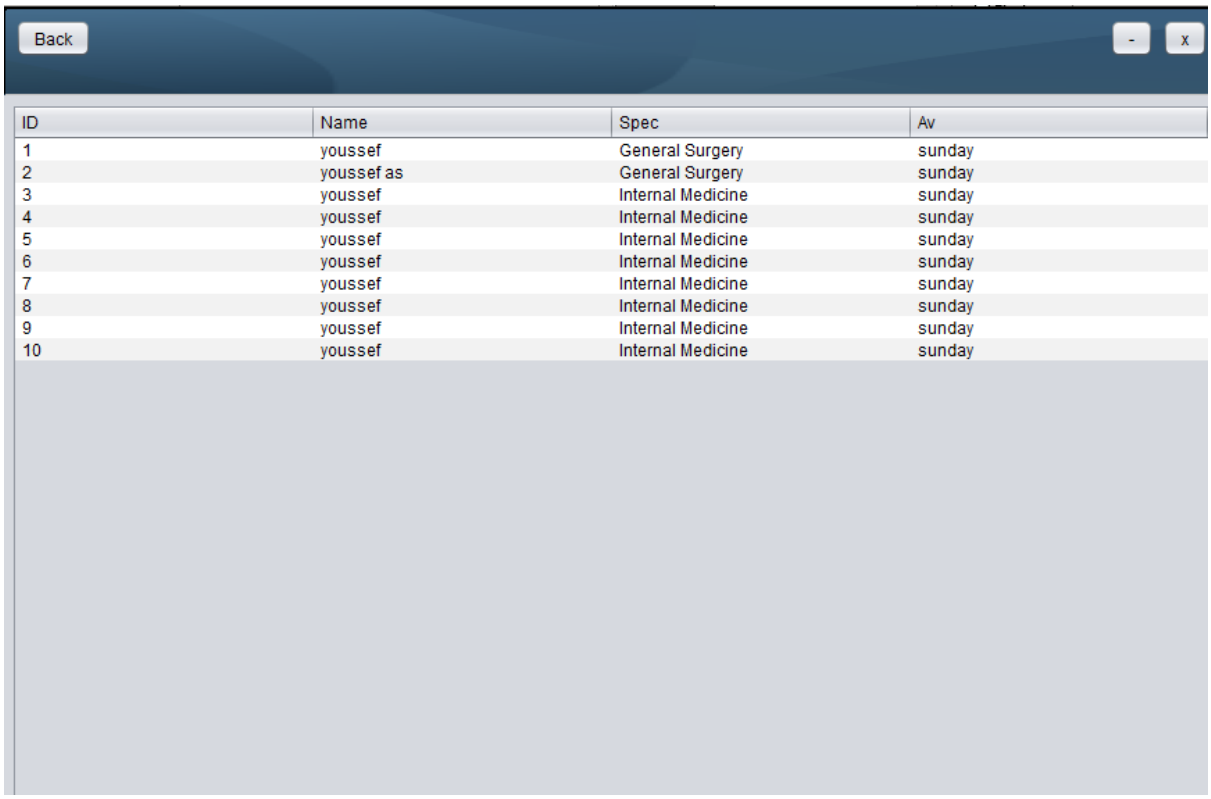


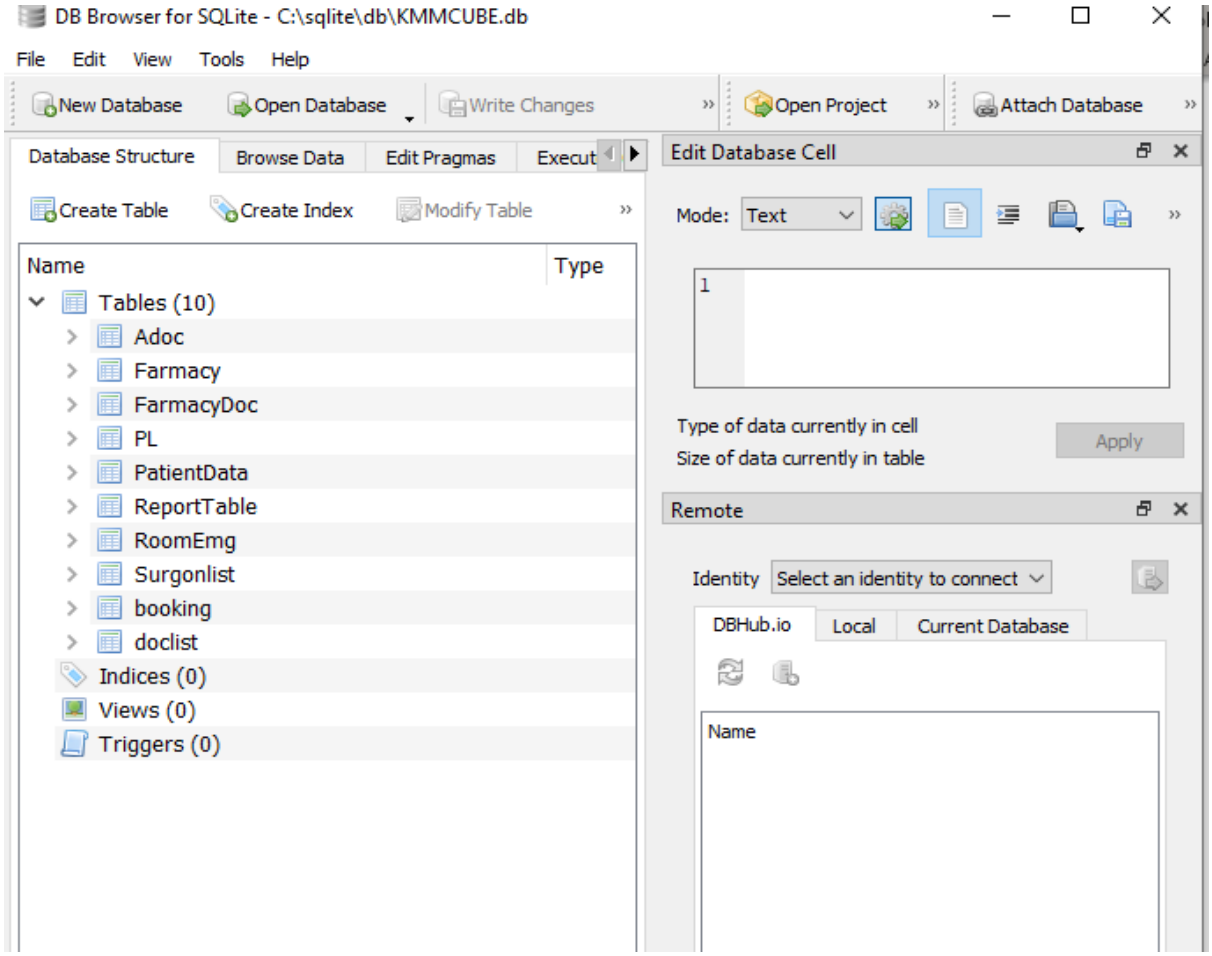
Reception Page 5



Reception Page 6

Reception Page 7


Back
-
x
ID
Name
Spec
Av
1 youssef General Surgery sunday
2 youssef as General Surgery sunday
3 youssef Internal Medicine sunday
4 youssef Internal Medicine sunday
5 youssef Internal Medicine sunday
6 youssef Internal Medicine sunday
7 youssef Internal Medicine sunday
8 youssef Internal Medicine sunday
9 youssef Internal Medicine sunday
10 youssef Internal Medicine sunday


DB Browser for SQLite - C:\sqlite\db\KMMCUBE.db
File
Edit
View
Tools
Help
New Database
Open Database
Write Changes
Open Project
Attach Database
Database Structure
Browse Data
Edit Pragmas
Execut
Create Table
Create Index
Modify Table
Name
Type
Tables (10)
Adoc
Farmacy
FarmacyDoc
PL
PatientData
ReportTable
RoomEmg
Surgonlist
booking
doclist
Indices (0)
Views (0)
Triggers (0)
Edit Database Cell
Mode:
Text
1
Type of data currently in cell
Size of data currently in table
Apply
Remote
Identity
Select an identity to connect
DBHub.io
Local
Current Database
Name

Data Base