HOSPITAL MANAGEMENT SYSTEM

Mini-Project Final Report

By: Sathvik Chanda (200953078)

Sushant Raj (200953062)

Vaishnavi (200953025)

Adithya Rao Kalathur (200953015)

NOVEMBER 10, 2022

ADVANCED PROGRAMMING LAB(ICT3166)

Index

Abstract:

This Hospital Management System project is a computerized hospital front desk management that produces user-friendly, quick, and cost-effective software. It handles and secures patient information, diagnosis data, and so on. This was done by hand and its' principal job is to register and maintain patient and doctor information and to access and update the information when needed. Patient information and diagnosis are entered into the system, then the output is used to display these details on the screen. A username and password are required to access the Hospital Management System. It can be accessed by a receptionist or an administrator. They are the only ones who have access to the database. The information is easily accessible. For personal usage, the data is well-protected, and the data processing is quick.

Introduction:

The project Hospital Management system includes registration of patients, storing their details into the system, and also computerized billing in the pharmacy, and labs. The software has the facility to give a unique id for every patient and stores the details of every patient and the staff automatically. It includes a search facility to know the current status of each room. User can search availability of a doctor and the details of a patient using the id. The Hospital Management System can be entered using a username and password. It is accessible either by an administrator or receptionist. Only they can add data into the database. The data can be retrieved easily. The interface is very user-friendly. The data are well protected for personal use and makes the data processing very fast. Hospital Management System is powerful, flexible, and easy to use and is designed and developed to deliver real conceivable benefits to hospitals. Hospital Management System is designed for multispeciality hospitals, to cover a wide range of hospital administration and management processes. It is an integrated end-toend Hospital Management System that provides relevant information across the hospital to support effective decision making for patient care, hospital administration and critical financial accounting, in a seamless flow. Hospital Management System is a software product suite designed to improve the quality and management of hospital management in the areas of clinical process analysis and activity-based costing. Hospital Management System enables you to develop your organization and improve its effectiveness and quality of work. Managing the key processes efficiently is critical to the success of the hospital helps you manage your processes.

Background Information:

In this busy world we don’t have the time to wait in infamously long hospital queues. The problem is, queuing at hospital is often managed manually by administrative staff, then take a token there and then wait for our turn then ask for the doctor and the most frustrating thing - we went there by traveling a long distance and then we come to know the doctor is on leave or the doctor can’t take appointments.

HMS will help us overcome all these problems because now patients can book their appointments at home, they can check whether the doctor they want to meet is available or not. Doctors can also confirm or decline appointments, this help both patient and the doctor because if the doctor declines’ appointment then patient will know this in advance and patient will visit hospital only when the doctor confirms’ the appointment this will save time and money of the patient.

Patients can also pay the doctor’s consultant fee online to save their time.

HMS is essential for all healthcare establishments, be it hospitals, nursing homes, health clinics, rehabilitation centers, dispensaries, or clinics. The main goal is to computerize all the details regarding the patient and the hospital. The installation of this healthcare software results in improvement in administrative functions and hence better patient care, which is the prime focus of any healthcare unit.

Methodology:

The Hospital Management System can be entered using a username and password. It is accessible either by an administrator or receptionist. Only they can add data into the database. The data can be retrieved easily. The data are well protected for personal use and makes the data processing very fast.

SOFTWARE REQUIREMENTS FOR THE PROJECT:

- FRONT END: Tkinter
- SERVER-SIDE SCRIPT: Python
- DATABASE: Sqlite (DBrowser)

Hospital Management System Project Modules:

- **Patient Management:** This module covers from the process of intake until discharge of an account of the patient's engagement with the health-care team. Communication, empathy, examination, evaluation, diagnosis, prognosis, and intervention are all part of the process.
- **Doctor/Physician Management:** The management of the physicians would be included in creating this system. Through this process, the admin will have the information and transactions made by the doctors with the patients.
- **Medicine and Prescription Management:** This module will handle the process of monitoring a patient's medications to verify that they are taken correctly and that the intended therapeutic outcome is achieved.
- **Online Appointment Management:** This process is a tool that helps hospital admin manage their appointments. Internet booking is one of the tools available in an appointment management solution. Booking with a mobile app.
- **Medical and Transaction Management:** Medical and transaction management modules aims to secure every transaction made by the patients and physicians in order to enhance healthcare quality and outcomes.
- **Payment and Expense Management:** Payment and expense management module id meant to assist the admin in the payment

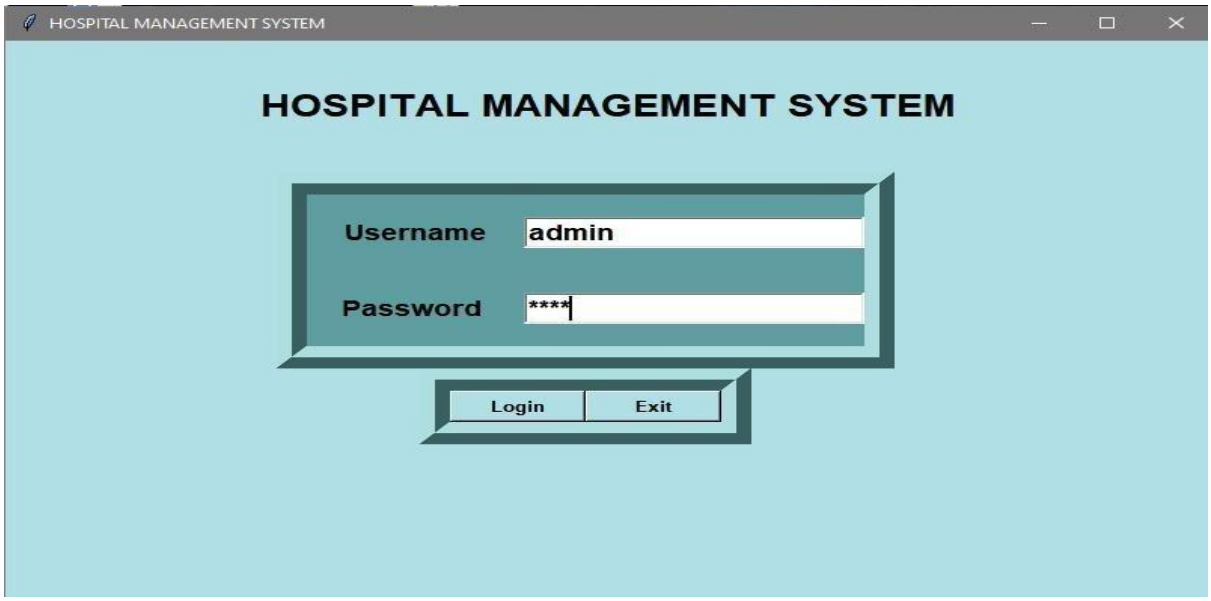
management process. This will help the hospital with the full payment processing and accounts payable process.

These modules must be present in creating the Hospital Management to satisfy the needs in managing Hospital transactions. Through this, the management and monitoring of patients would be much easier for both hospital admin and physicians.

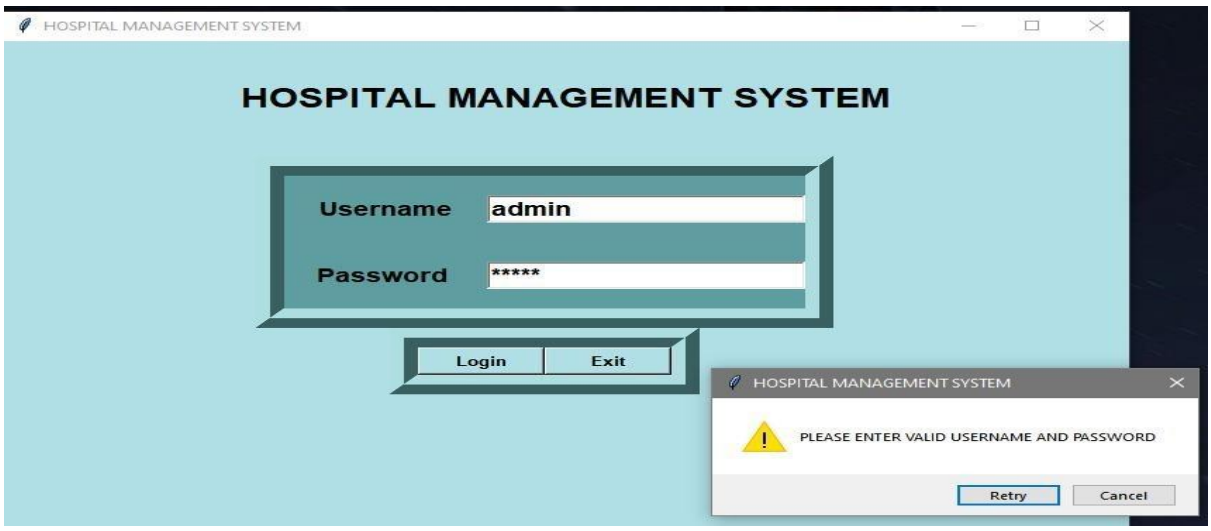
Sample Screenshots:

Login window

- **If username and password are valid.**



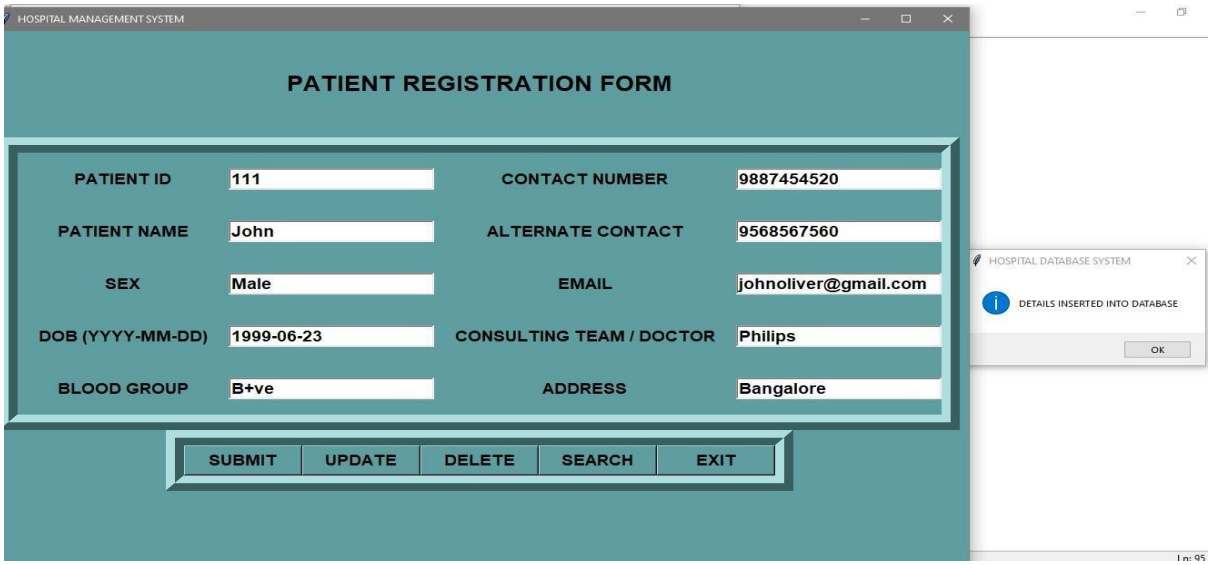
- **If username and password are invalid.**



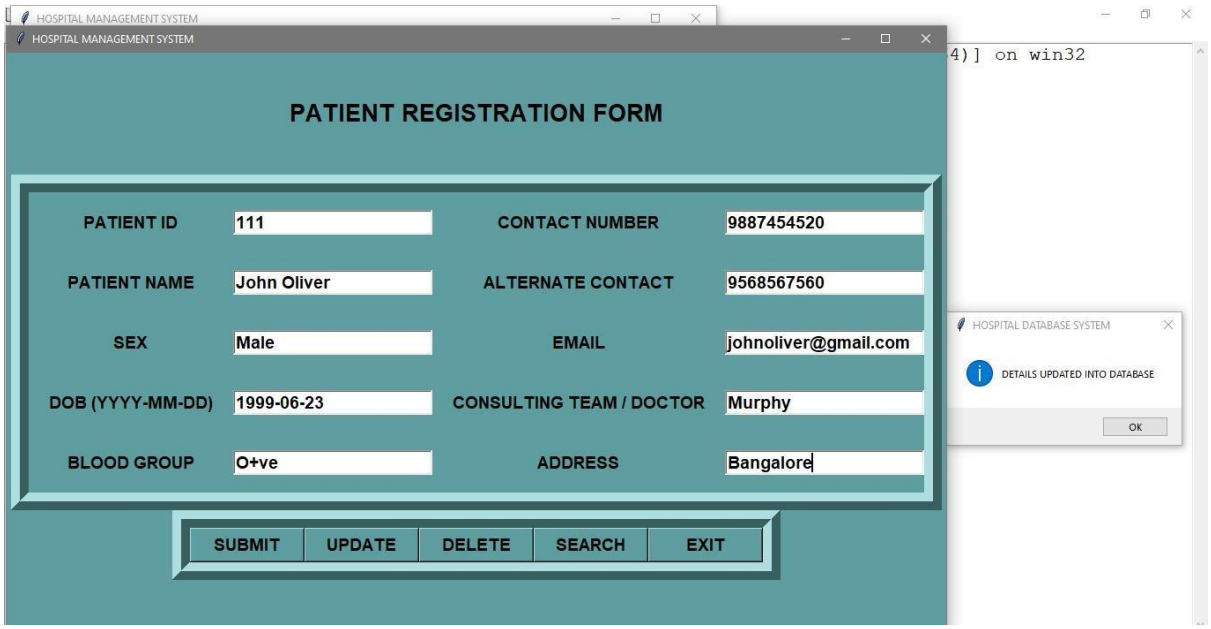
Menu

Patient Registration form:

- Patient registration: Submit

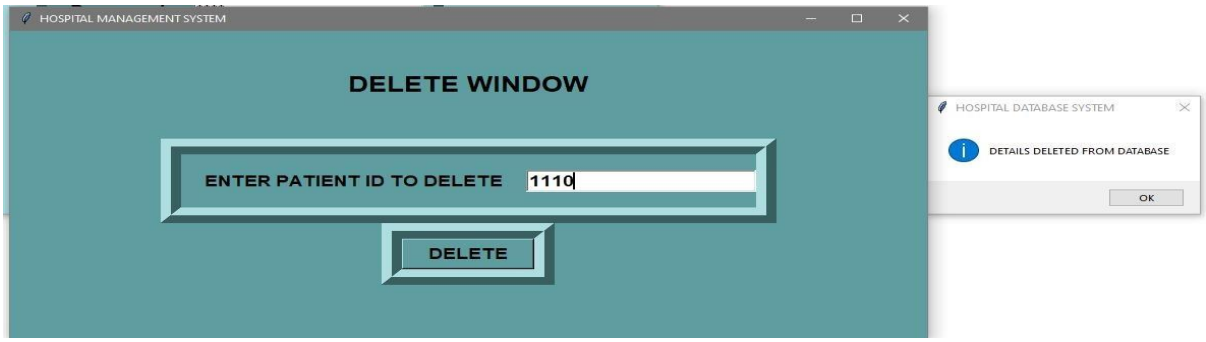


- Patient registration: Update



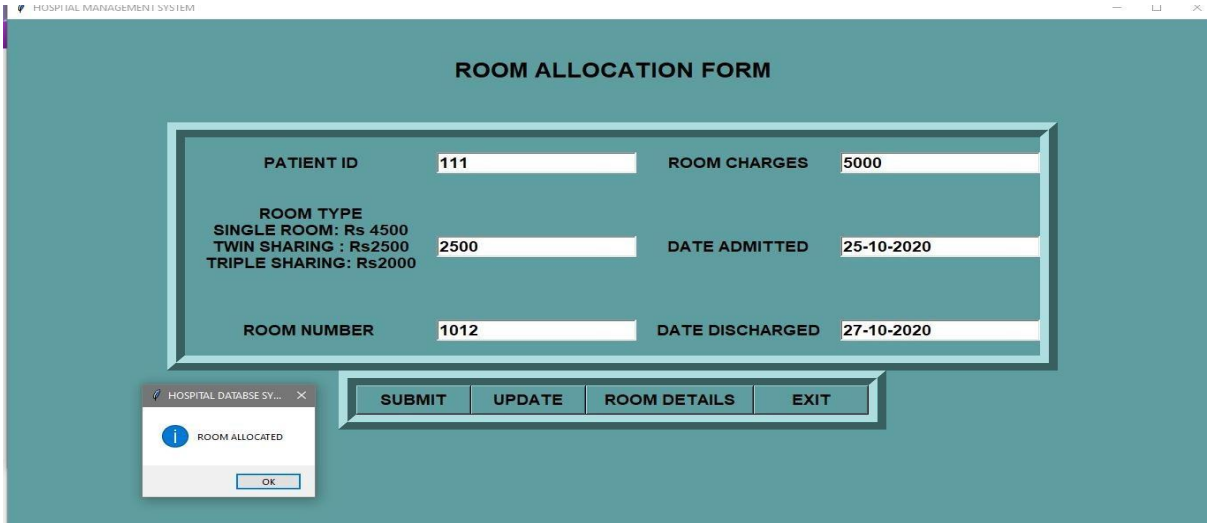
- Patient registration: Search

- Patient registration: Delete

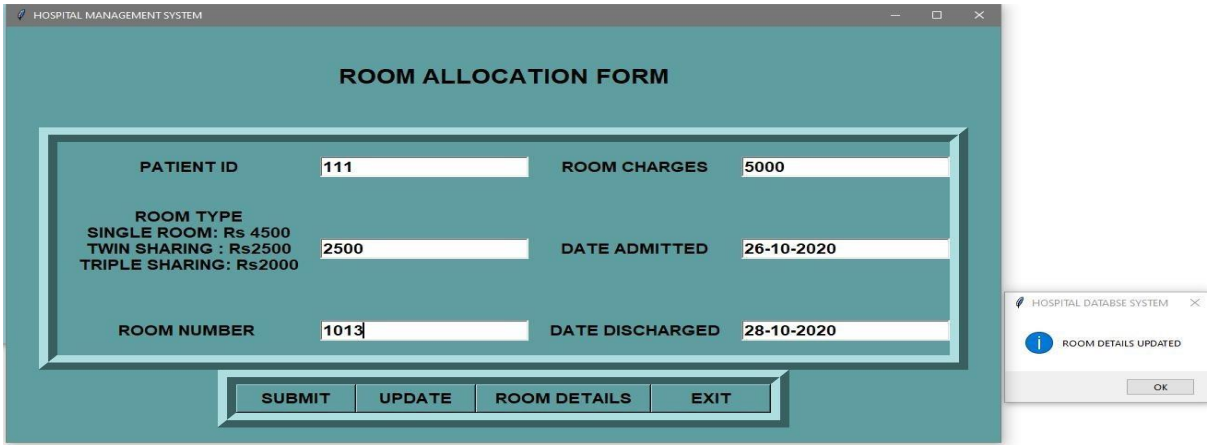


Room Allocation:

- Room Allocation: Submit



- Room Allocation: Update



- Room Allocation: Room details

Employee Registration:

- Employee Registration: Save

- Employee Registration: Delete

Book Appointment:

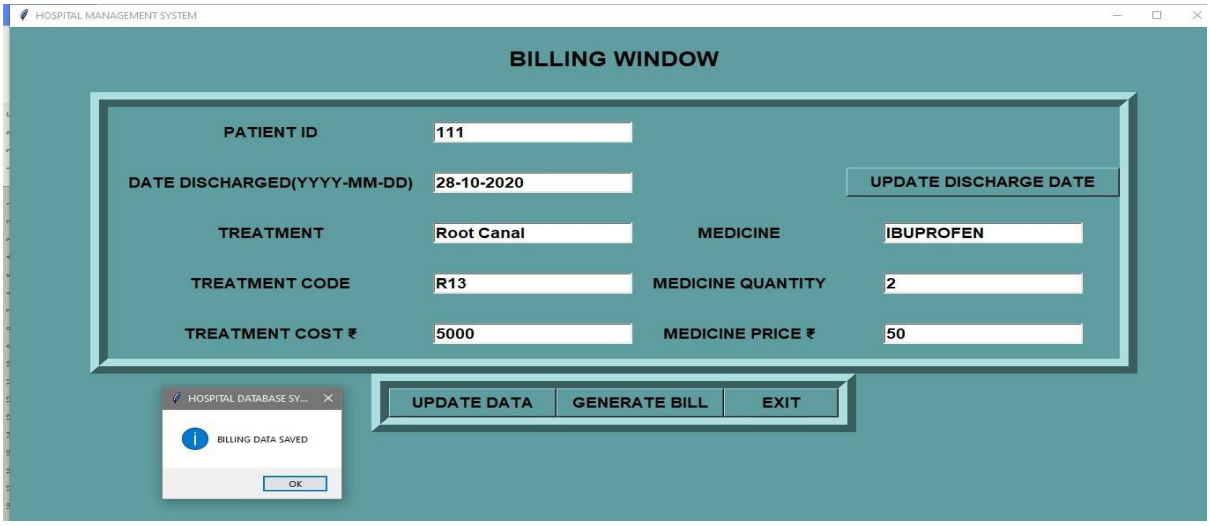
- Book Appointment: Save

- Book Appointment: Delete

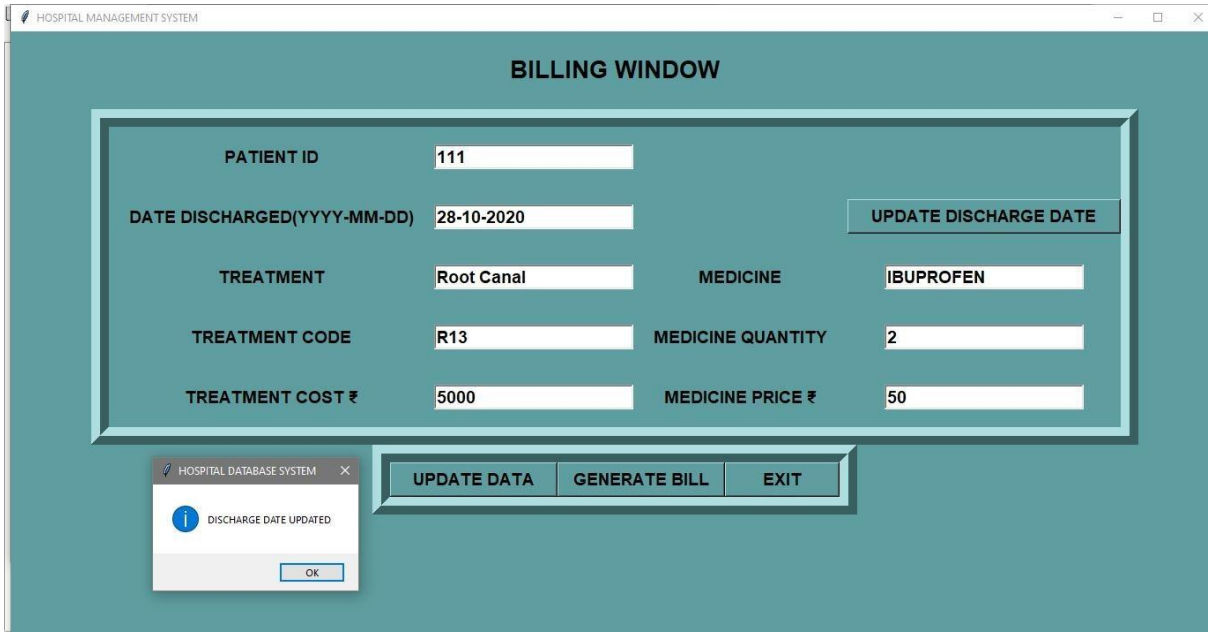
- Book Appointment: Search Appointments

Patient Bill:

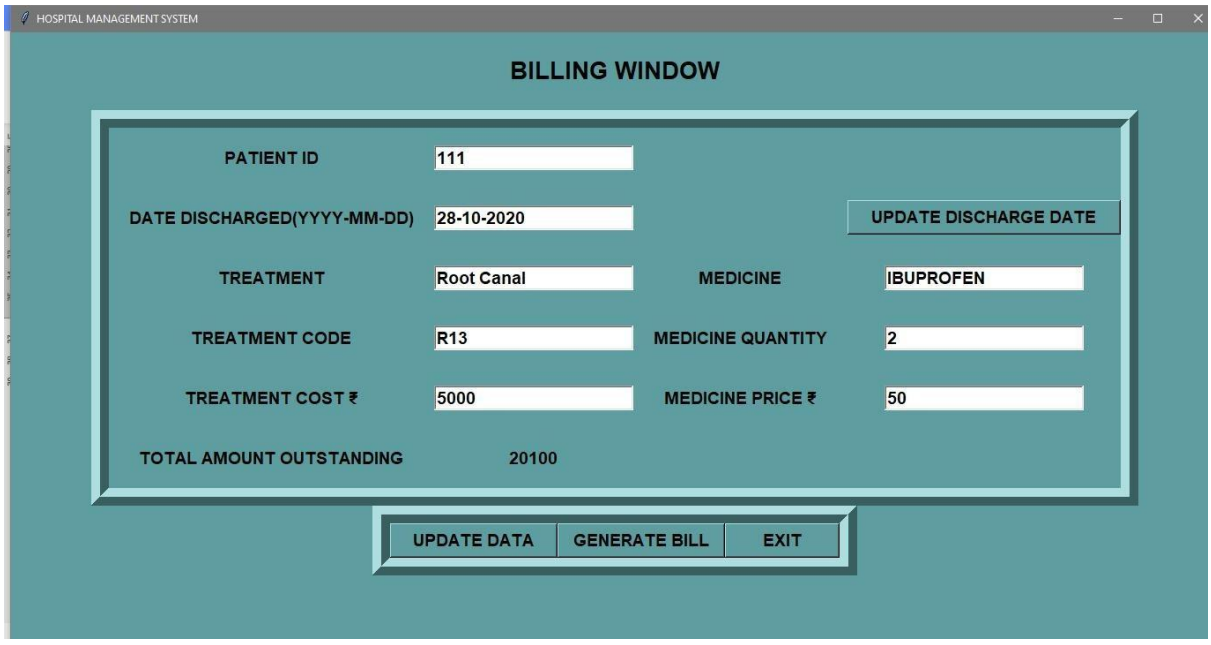
- Patient Bill: Update Data



- Patient Bill: Update Discharge Date

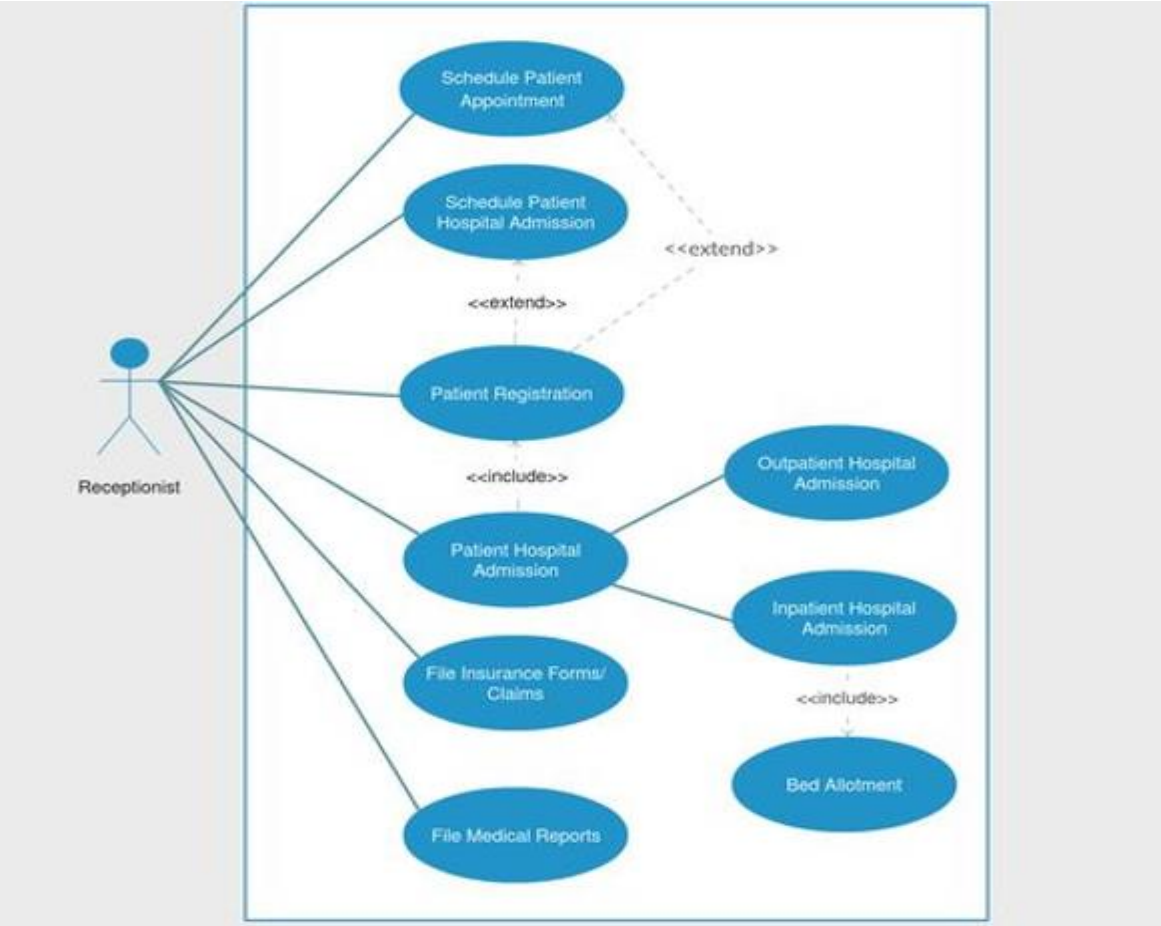


- Patient Bill: Generate Bill

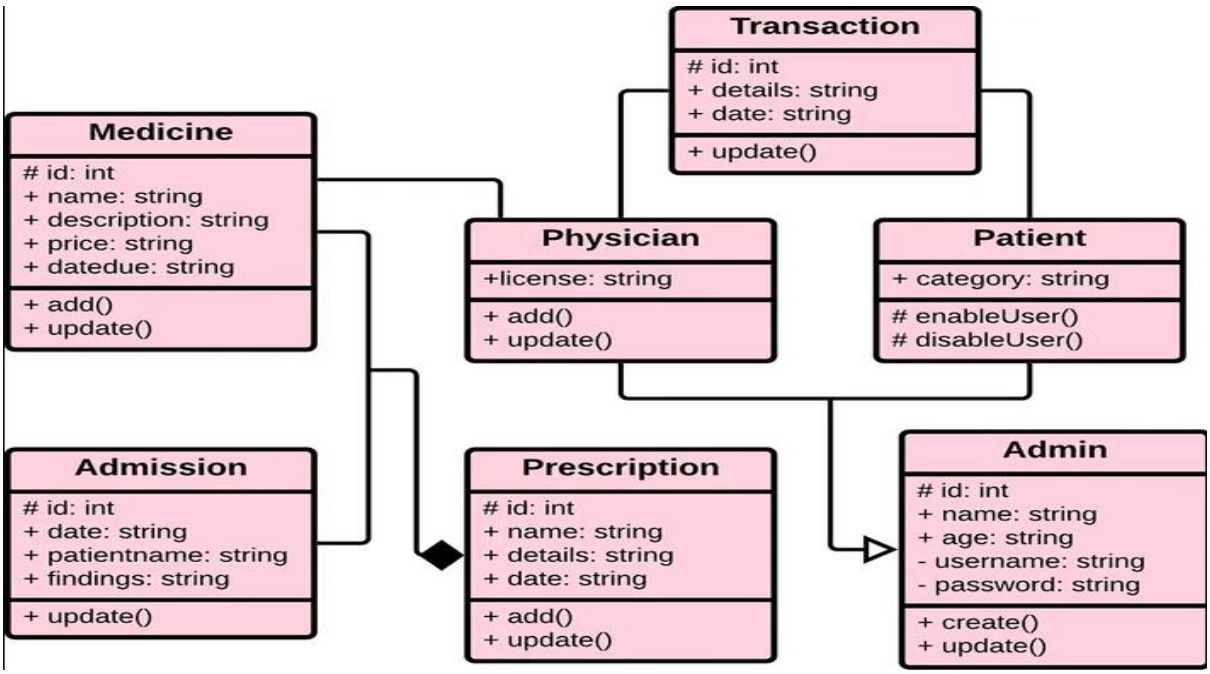


UML Diagrams for our Project:

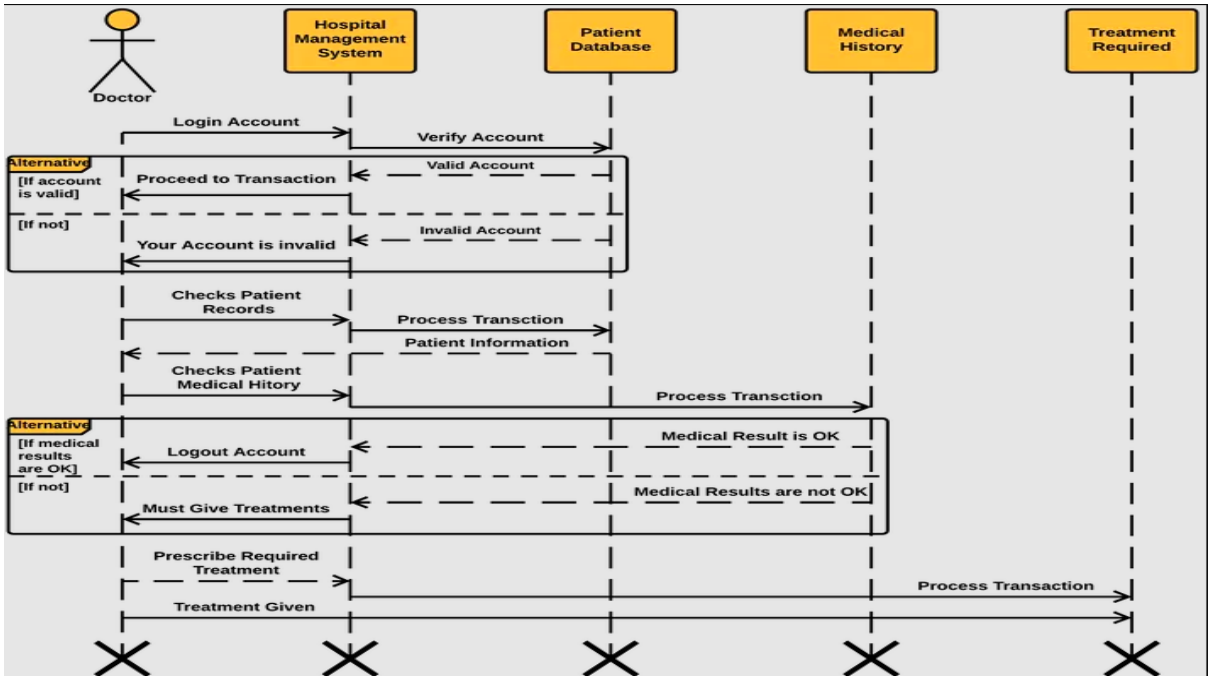
Use Case Diagram:



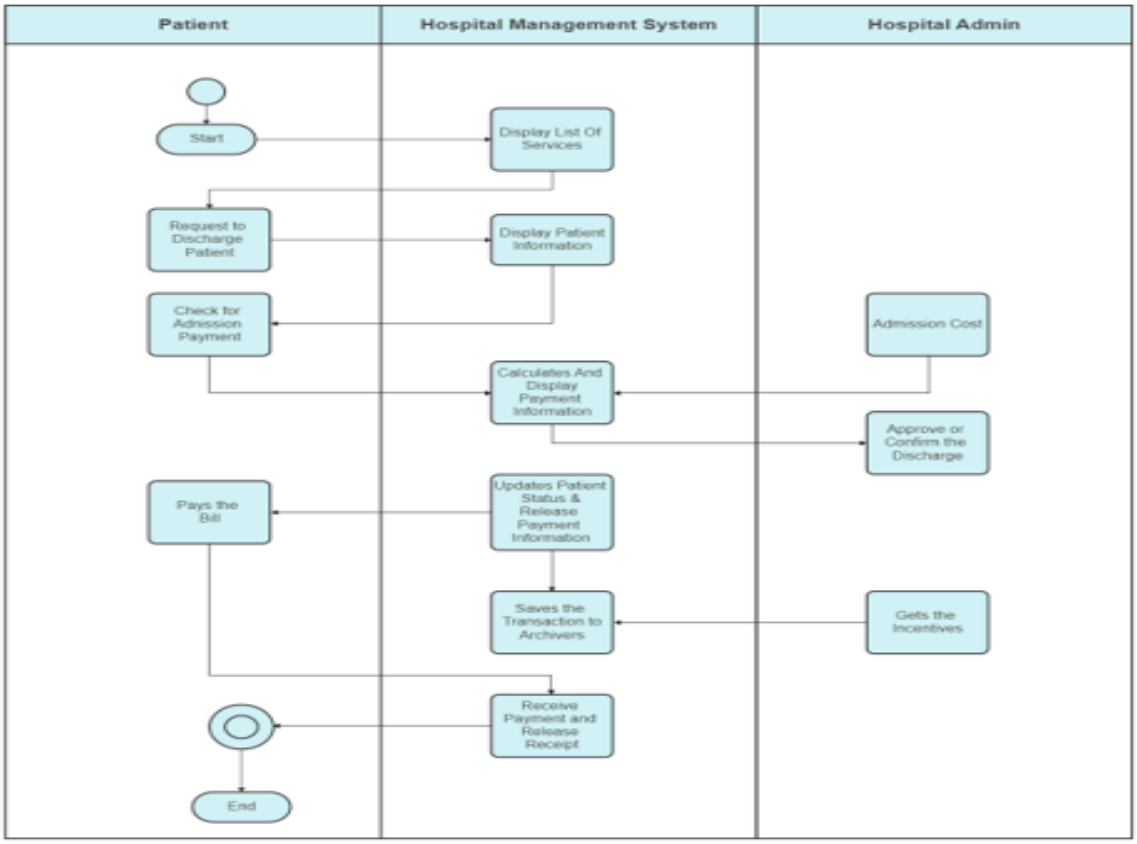
Class Diagram:



Sequence Diagram:



Activity Diagram:



Implementation:

Sample codes:

Login.py

```
from tkinter import *
import tkinter.messagebox
from tkinter import ttk
from tkinter import font
from menu import Menu
def main():
    root = Tk()
    app= MainWindow(root)
#MAIN WINDOW FOR LOG IN
class MainWindow:
    # constructor
    def __init__(self,master):
        # public data mambers
        self.master = master
        self.master.title("HOSPITAL MANAGEMENT SYSTEM")
        self.master.geometry("800x500+0+0")
        self.master.config(bg="powder blue")
        self.frame = Frame(self.master,bg="powder blue")
        self.frame.pack()
        self.Username = StringVar()
        self.Password = StringVar()
        self.lblTitle = Label(self.frame,text = "HOSPITAL MANAGEMENT SYSTEM",
font="Helvetica 20 bold",bg="powder blue",fg="black")
        self.lblTitle.grid(row =0 ,column = 0,columnspan=2,pady=40)
        #=====================
```

```
        self.LoginFrame1 = Frame(self.frame,width=400,height=80,relief="ridge",bg="cadet
blue",bd=20)

        self.LoginFrame1.grid(row=1,column=0)

        self.LoginFrame2 = Frame(self.frame,width=400,height=80,relief="ridge",bg="cadet
blue",bd=20)

        self.LoginFrame2.grid(row=2,column=0)

        #======LABEL AND ENTRY=========

        self.lblUsername = Label(self.LoginFrame1,text="Username",font="Helvetica 14
bold",bg="cadet blue",bd=22)

        self.lblUsername.grid(row=0,column=0)

        self.lblUsername = Entry(self.LoginFrame1,font="Helvetica 14 bold",textvariable=
self.Username,bd=2)

        self.lblUsername.grid(row=0,column=1)

        self.lblPassword = Label(self.LoginFrame1,text="Password ",font="Helvetica 14
bold",bg="cadet blue",bd=22)

        self.lblPassword .grid(row=1,column=0)

        self.lblPassword  = Entry(self.LoginFrame1,font="Helvetica 14
bold",show="*",textvariable= self.Password,bd=2)

        self.lblPassword .grid(row=1,column=1)

        #===========BUTTONS====

        self.btnLogin = Button(self.LoginFrame2,text = "Login" ,font="Helvetica 10 bold",
width =10 ,bg="powder blue",command = self.Login_system)

        self.btnLogin.grid(row=3,column=0)

        self.btnExit = Button(self.LoginFrame2,text = "Exit" ,font="Helvetica 10 bold", width
=10 ,bg="powder blue",command = self.Exit)

        self.btnExit.grid(row=3,column=1)

    # public member function

    #Function for LOGIN

    def Login_system(self):

        S1=(self.Username.get())

        S2=(self.Password.get())

        if(S1=='admin' and S2=='1234'):

            self.newWindow = Toplevel(self.master)
```

```
            self.app = Menu(self.newWindow)
        elif(S1=='root' and S2=='4321'):
            self.newWindow = Toplevel(self.master)
            self.app = Menu(self.newWindow)
        else:
            tkinter.messagebox.askretrycancel("HOSPITAL MANAGEMENT SYSTEM" ,
"PLEASE ENTER VALID USERNAME AND PASSWORD")
    #Function for Exit
    def Exit(self):
        self.master.destroy()
if __name__ == "__main__":
    main()
```

Database.py

```
import sqlite3
conn=sqlite3.connect("HospitalDB.db")
print("DATABASE CONNECTION SUCCESSFUL")
#conn.execute("Drop table if EXISTS PATIENT")
#c = conn.cursor()
#conn.execute("""Create table PATIENT
 #          (PATIENT_ID int(10) primary key,
  #          NAME VARCHAR(20) not null,
   #         SEX varchar(10) not null,
    #        BLOOD_GROUP varchar(5) not null,
     #       DOB date not null,
      #      ADDRESS varchar(100) not null,
       #     CONSULT_TEAM varchar(50) not null,
        #    EMAIL varchar(20) not null
         #   )""")
print("PATIENT TABLE CREATED SUCCESSFULLY")
#conn.execute("Drop table if EXISTS CONTACT_NO")
#c = conn.cursor()
```

```
#conn.execute("""CREATE TABLE CONTACT_NO
 #          (PATIENT_ID int(10) PRIMARY KEY,
  #          CONTACTNO int(15) not null,
   #         ALT_CONTACT int(15),
    #        FOREIGN KEY(PATIENT_ID) REFERENCES PATIENT(PATIENT_ID))
     #      """)
print("CONTACT_NO TABLE CREATED SUCCESSFULLY")
#conn.execute("Drop table if EXISTS employee")
#c = conn.cursor()
#conn.execute("""create table employee
 #          (EMP_ID varchar(10) primary key,
  #         EMP_NAME varchar(20)not null,
   #         SEX varchar(10) not null,
    #        AGE int(5) not null,
     #       DESIG varchar(20) not null,
      #      SAL int(10) not null,
       #    EXP varchar(100) not null,
        #     EMAIL varcahr(20) not null,
         #    PHONE int(12))""")

print("EMPLOYEE TABLE CREATED SUCCESSFULLY")
#conn.execute("Drop table if EXISTS TREATMENT")
#c = conn.cursor()
#conn.execute("""CREATE TABLE TREATMENT
 #          (PATIENT_ID int(10) primary key,
  #          TREATMENT varchar(100) not null,
   #         TREATMENT_CODE varchar(30) not null,
    #        T_COST int(20) not null,
     #       FOREIGN KEY(PATIENT_ID) REFERENCES PATIENT(PATIENT_ID));
      #      """)
```

```
print("TREATMENT TABLE CREATED SUCCESSFULLY")
#conn.execute("Drop table if EXISTS MEDICINE")
#c = conn.cursor()
#conn.execute("""CREATE TABLE MEDICINE
 #          (PATIENT_ID int(10) primary key,
  #          MEDICINE_NAME varchar(100) not null,
   #          M_COST int(20) not null,
    #          M_QTY int(10) not null,
     #          FOREIGN KEY(PATIENT_ID) REFERENCES PATIENT(PATIENT_ID));
      #          """)
print("MEDICINE TABLE CREATED SUCCESSFULLY")
#conn.execute("Drop table if EXISTS ROOM")
#c = conn.cursor()
#conn.execute("""Create table ROOM
        # (PATIENT_ID int(10)not NULL ,
         #  ROOM_NO varchar(20) PRIMARY KEY ,
         #ROOM_TYPE varchar(10) not null,
         #  RATE int(10) not null,
          # DATE_ADMITTED date,
           # DATE_DISCHARGED date NULL,
          # FOREIGN KEY(PATIENT_ID) REFERENCES PATIENT(PATIENT_ID)
           #);
          # """)
print("ROOM TABLE CREATED SUCCESSFULLY")
conn.execute("Drop table if EXISTS APPOINTMENT")
c = conn.cursor()
c.execute("""create table appointment
          (
           PATIENT_ID int(20) not null,
           EMP_ID varchar(10) not null,
```

```
        AP_NO varchar(10) primary key,
        AP_TIME time,
        AP_DATE date,
        description varchar(100),
        FOREIGN KEY(PATIENT_ID) references PATIENT(PATIENT_ID),
        FOREIGN KEY(EMP_ID) references employee(EMP_ID));""")
print("APPOINTMENT TABLE CREATED SUCCESSFULLY")
conn.commit()
conn.close()
```

Results :

- **Improves interaction between the patient and the hospital**
  With faster, secure, and easy data retrieval, a hospital/healthcare facility would be able to provide better and efficient care to the patients. With every department interconnected and integrated into the HMS, the quality of patient care can be enhanced, leading to greater customer satisfaction and lowered turnovers. Today there is severe competition even in the realm of healthcare, and patients and their kin prefer to visit a facility that is efficient, cost-effective, and secure.
- **Boosts productivity and establishes a routine**
  An effective HMS will be based on the standard operating procedures (SOPs) and best practices, leaving little or no room for error. The HMS enables your health facility to care for its patients better through faster processes, storing and analysis of patient history, preparation and access of real time reports, appointment scheduling and tracking, and many more such processes. Employees, too, find it easier to manage the huge numbers of patients, records, and other jobs critical to the smooth functioning of a healthcare facility.
- **Makes patient's data easily retrievable**
  An effective HMS will be based on the standard operating procedures (SOPs) and best practices, leaving little or no room for error. The HMS enables your health facility to care for its patients better through faster processes, storing and analysis of patient history, preparation and access of real time reports, appointment scheduling and tracking, and many more such processes. Employees, too, find it easier to manage the huge numbers of patients, records, and other jobs critical to the smooth functioning of a healthcare facility.
- **Keep the hospital data secure**
  The cloud-based medical records software in a Hospital Management System, ensures that all data remains interlinked and with high security. This translates to ease of data storage and retrieval by authorized personnel only. Faster access

to accurate data has a significant impact on the speed and efficiency of the overall operational and administrative processes of a hospital/healthcare facility.

- **Cost-effective solution**
  The risk of improper billing, financial management, fraud, and other financial related issues are kept to the minimum with an effective HMS. Further costs are reduced since the requirement of a larger human workforce is lowered. Manual tasks such as record maintenance and storage can easily be managed by the system, thereby freeing humans to focus on higher level tasks. In addition to better utilization of human capital, healthcare facilities also cut costs with relation to storage and maintenance of a large number of records. Only mandatory physical records would need storing in order to remain compliant with regulation standards.

Conclusion:

The Hospital Management System has today become an indispensable part of any hospital/clinic/healthcare facility. In order to create a differentiated, efficient, speedy, and thoughtful healthcare model, it would make sense to invest in a comprehensive HMS.

References:

1. A detailed view of Hospital Management System (HMS) https://mocdoc.in/blog/a-detailed-view-of-hospital-management-system-hms
2. Critical Elements and Lessons Learnt from the Implementation of an RFID-enabled Healthcare Management System in a Medical Organization. https://pubmed.ncbi.nlm.nih.gov/20703523/
3. REDUCING THE LITERAL AND HUMAN COST OF CHILD ABUSE: IMPACT OF A NEW HOSPITAL MANAGEMENT SYSTEM . https://www.researchgate.net/publication/18463711_Reducing_the_literal_and_human_cost_of_child_abuse_impact_of_a_new_hospital_management_system
4. Design of an RFID-based Healthcare Management System using an Information System Design Theory. https://link.springer.com/article/10.1007/s10796-009-9154-3
5. Internet Of Things (IOT) enabled smart autonomous hospital management system - A real world health care use case with the technology drivers. https://www.researchgate.net/publication/304416566_Internet_Of_Things_IOT_enabled_smart_autonomous_hospital_management_system_-_A_real_world_health_care_use_case_with_the_technology_drivers