

# Course Based Learning

(Via Projects)

Project is Based on the Language **Python**

**“Patient Management System(PMS)”**

Submitted By

NIZBA IQRA

-

2020051506

Under the Guidance of

**Dr Vishwa Kiran S**

**Mr K B Hemanth Raj**



**Future Vision**

## Future Vision BIE

Future Vision Beginners Intermediate Experts (BIE)

Kaveri Nagar, R T Nagar Post, Bengaluru – 560032

# PATIENT MANAGEMENT SYSTEM

## Abstract:

The project "**Patient management system**" is aimed to develop to maintain the day-to-day state of admission/discharge of patients , list of patient, reports, generation and etc

It is used to design to achieve the following objectives:

1. To computerize all details regarding patient details and hospital details.
2. Scheduling the appointment of patient with doctors to make it convenient for both.
3. Scheduling the serviced of specialized doctors and emergency properly so that facilities provided by hospital are fully utilized in effective and efficient manner.
4. if the medical store issues medicines to patient and also any charges it should keep tracking of all bills.
5. it should be able to handle the ward in which the patient is to be taken to.
6. the information of the patients should be kept up to date and there record should be kept in the system for historical purpose.

## Introduction:-

The project Patient 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. 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. Patient Management System is powerful, flexible, and easy to use and is designed and developed to deliver real conceivable benefits to hospitals. Patient Management System is designed for multi speciality hospitals, to cover a wide range of hospital administration and management processes. It is an integrated end-to-end Patient 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.

Patient 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.

## **Motivation :**

1. Most important reason for keeping medical records is to provide information on a patient's care to other health care.
2. Well documented medical record Provides support for the Physicians defence in the event of medical malpractice action.

## **ANALYSIS:**

### **Cost Estimation and Scheduling**

Project estimation and scheduling were carried out together by the project leader as per the norms of the hospital. Some cost estimation was done at the early stage before the schedules were drawn up. Once the project was underway, estimates were updated.

### **Existing software:**

Hospitals currently use a manual system for the management and maintenance of critical information. The current system requires numerous paper forms, with data stores spread throughout the hospital management infrastructure. Often information is incomplete or does not follow management standards. Forms are often lost in transit between departments requiring a comprehensive auditing process to ensure that no vital information is lost. Multiple copies of the same information exist in the hospital and may lead to inconsistencies in data in various datastores

### **Alphabetical System :**

Files are arranged in alphabetical order according to the patients surname or business name.

## **Limitations of Existing software:**

Filing by discharge number, alphabetical filing is generally unsatisfactory because other important records or register in the facility are concerned exclusively with medical record numbers & using PMS (Patient Management System) searching of Patient information along with billing details is comparatively easy.

## **Proposed system:**

The Patient Management System is designed for any hospital to replace their existing manual paper based system. The new system is to control the information of patients. Room availability, staff and operating room schedules and patient invoices. These services are to be provided in an efficient, cost effective manner, with the goal of reducing the time and resources currently required for such tasks.

The Hospital Management System is designed for any hospital to replace their existing manual paper based system. The new system is to control the information of patients. Room availability, staff and operating room schedules and patient invoices. These services are to be provided in an efficient, cost effective manner, with the goal of reducing the time and resources currently required for such tasks .

## **Feasibility Study:**

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. Three key considerations involved in the feasibility analysis are:

### **Economic Feasibility:**

This study is carried out to check the economic impact will have on the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customised products have to be purchased.

### **Technical Feasibility:**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes for the implementing this system.

### **Operational Feasibility:**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

## **SYSTEM REQUIREMENT SPECIFICATION**

### **INTRODUCTION:**

To be used efficiently, all computer software needs certain hardware components or the other software resources to be present on a computer. These pre-requisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that this trend plays a bigger part in driving upgrades to existing computer systems than technological advancements.

### **PURPOSE:**

The main purpose of software requirement Specifications document is to describe in a precise manner all the capabilities that will be provided by the software applications, "Patient Management System". These are to be exposed to the

development, testing team and end users of the software.

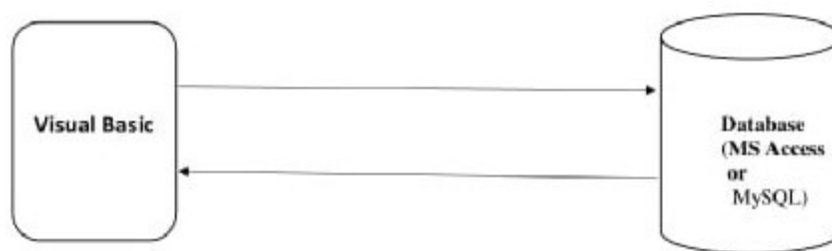
## SCOPE:

The purposed software product is the Patient Management System. The System will be used in an hospital, clinic, dispensary to get information from the patients and then storing that data for future usage.

## OVERALL DESCRIPTION OF THE PROPOSED SYSTEM:

### Product Perspective:

The application will be window -based, Self-contained and independent software product



## HARDWARE REQUIREMENTS:

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatibility and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

## HARDWARE REQUIREMENTS FOR PRESENT PROJECT:

- ✓ PROCESSOR : Intel dual Core ,i3
- ✓ RAM : 1 GBHARD

✓ DISK : 80

## **SOFTWARE REQUIREMENTS:**

Software Requirements deal with defining software resource requirements and pre-requisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or pre-requisites are generally not included in the software installation package and need to be installed separately before the software is installed

### **SOFTWARE REQUIREMENTS FOR PRESENT PROJECT:**

- ✓ OPERATING SYSTEM : Windows 7/ XP/8/10.
- ✓ FRONT END : Python
- ✓ DATABASE : My SQL

## **Proposed Methodology:**

### **By Testing Process:**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

## **Types Of Testing:**

### **1.Unit testing:**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an

individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

## **2.Functional test:**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centred on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures: interfacing systems or procedures must be invoked.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page

## **3.Integration Testing:**



Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications.

e.g., components in a software system or one step up software applications at the company level interact without error.

### **Test Results:**

All the test cases mentioned above passed successfully. No defects encountered.

### **References :**

- Sample template and guidance given by our mentor Mr. K.B Hemanth Raj during our course session.
- The course textbook, "Python for Everybody" by Charles R. Severance.
- The textbook, "Software Engineering" by IAN Sommerville.

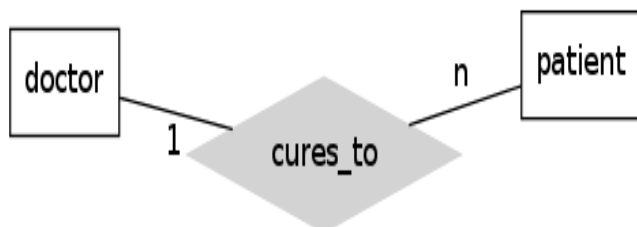
### **Abstract:**

#### **Database Using Software Engineering Concepts:**

##### **1.E-R Diagrams:**

Developing databases is a very important task to develop a system. Before going to form exact database tables and establishing relationships between them, we conceptually or logically can model our database using ER diagrams.

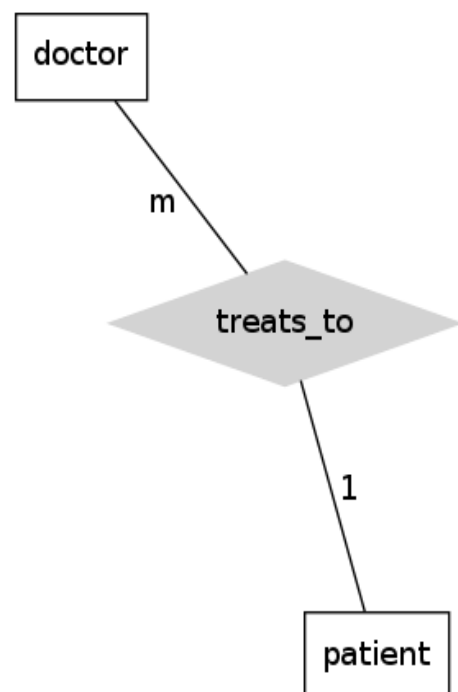
Showing relation (1 to n), (n to 1) between doctor and patient

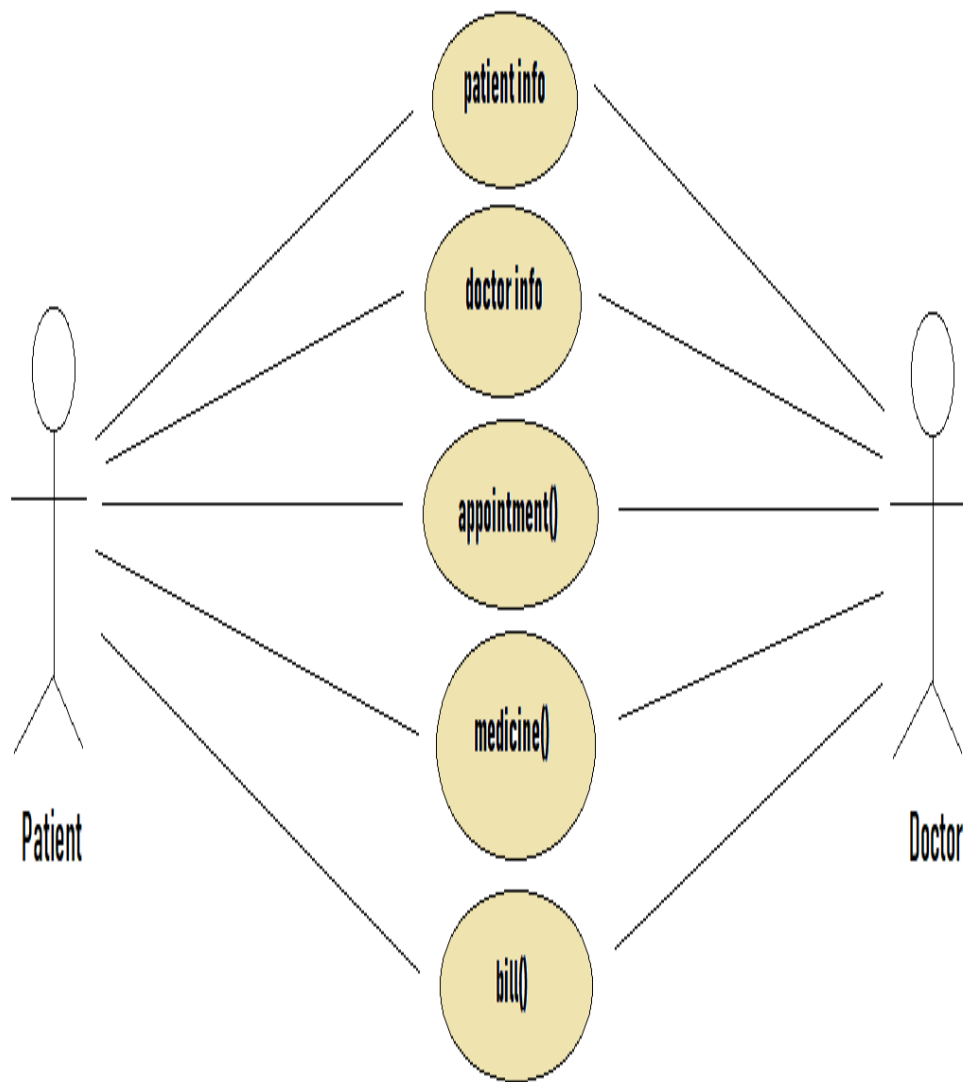


## 2.USE CASE

### DIAGRAM:

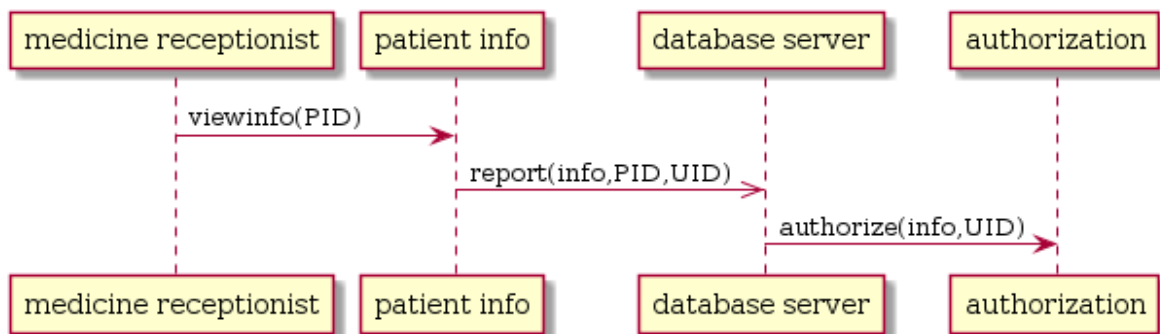
A use case diagram in the Unified Modelling Language(UML) is a type of behavioural diagram defined by and created from a use-case analysis. its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals(represented as use cases),and any dependencies between those use cases. Use case diagrams are formally included in two modelling languages defined by the OMG: the Unified Modelling Language(UML) and the systems Modelling Language(sys ML)





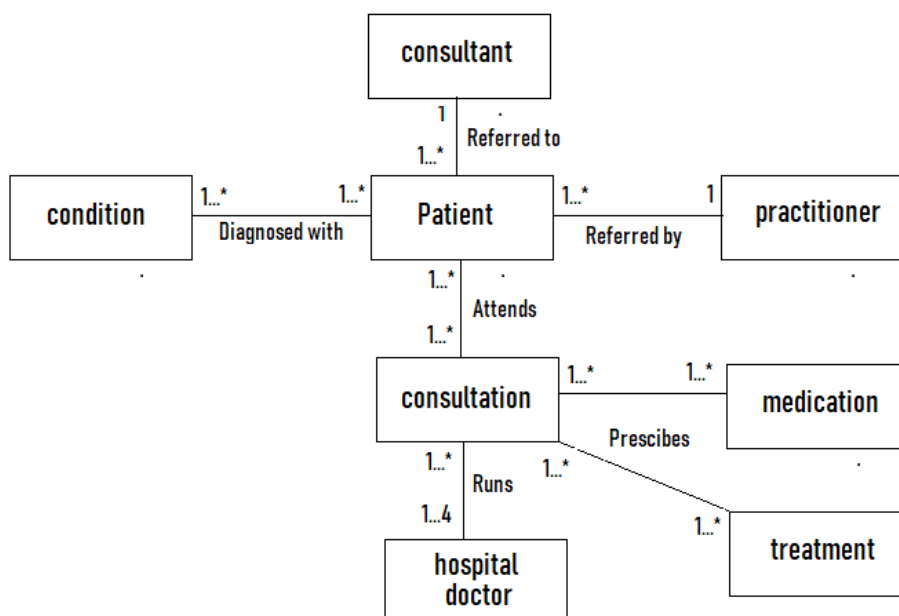
### 3.Sequence diagram:

A Sequence Diagram is an interaction diagram that emphasis the time ordering of messages; a collaboration diagram is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Sequence diagrams and collaboration diagrams are isomorphic, meaning that you can take one and transform it into the other.



### 4.Classes and associations in the PMS:

A Class is a category or group of things that has similar attributes and common behaviour. A

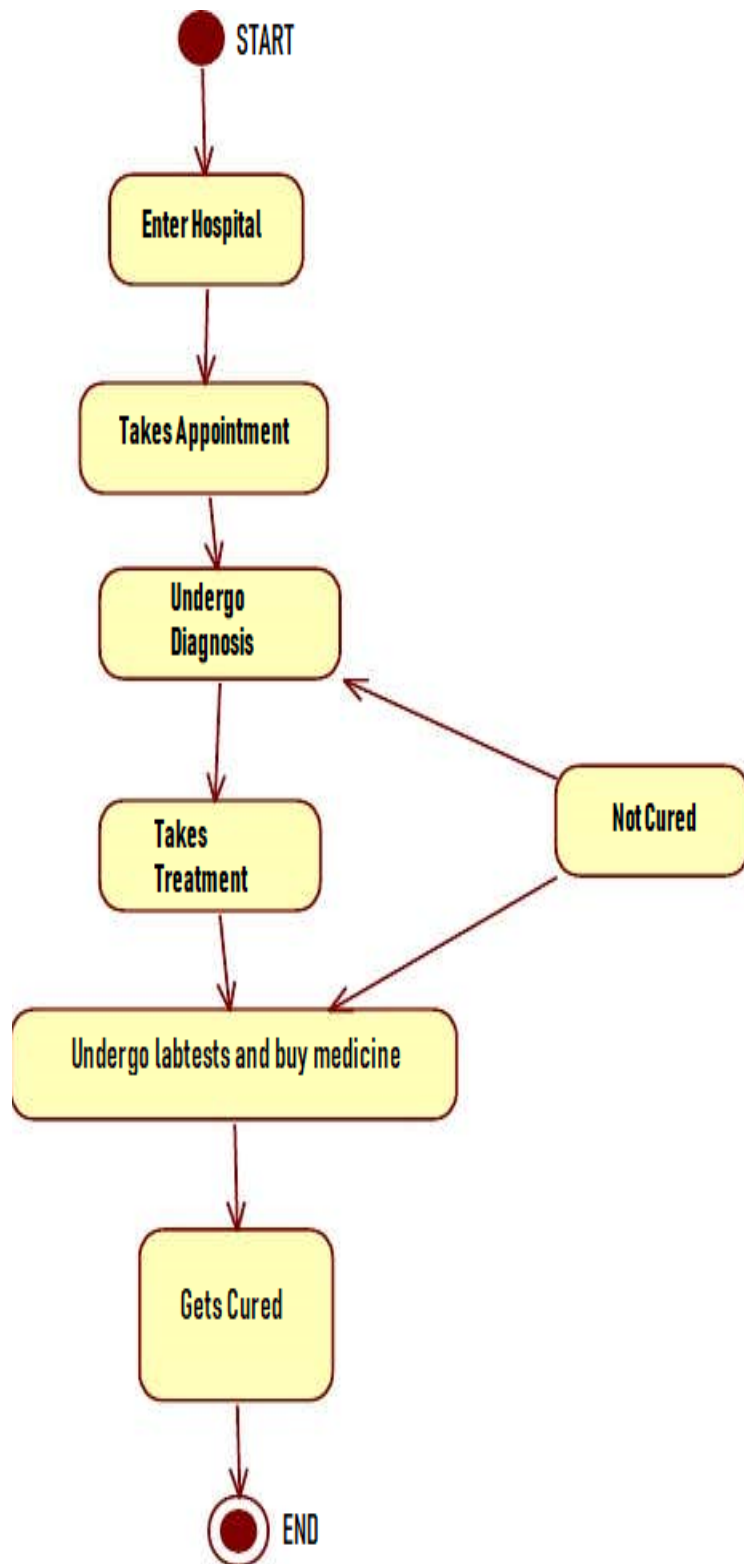


Rectangle is the icon that represents the class it is divided into three areas. The upper most area contains the name, the middle; area contains the attributes and the

lowest areas show the operations. Class diagrams provides the representation that developers work from.

## STATECHART DIAGRAMS:

The state diagram shows the states of an object and represents activities as arrows connecting the states. The Activity Diagram highlights the activities. Each activity is represented by a rounded rectangle—narrower and more oval-shaped than the state icon. An arrow represents the transition from the one activity to the next. The activity diagram has a starting point represented by a filled-in circle, and an end point represented by a bullseye.



## SOFTWARE LIFE CYCLE MODEL

The phases and the outputs of the waterfall model at each phase are summarized below in tabular format.

PHASE	OUTPUT
Problem Description	Feasibility Report
Analysis	SRS
Design	Detailed Design Report
Coding	Complete Source Code
Testing	Test plan, report and manual
Installation	Installation Report

## OBJECT ORIENTED CONCEPTS– JAVA:

**Java** is a general computer programming language developed by Sun Microsystems. Originally called "Oak", by its inventor James Gosling, **Java** was designed with several innovative features. These include a language that is entirely object oriented, and the ability to write an application once and move it to virtually any platform.

## **CLASSES**

A class is the blueprint from which individual objects are created.

## **OBJECTS**

Objects are key to understanding object-oriented technology

**Bundling code into individual software objects provides a number of benefits, including:**

- 1) Modularity
- 2) Information-hiding
- 3) Code re-use
- 4) Pluggability and debugging ease

## **INHERITANCE**

Object-oriented programming allows classes to inherit commonly used state and behaviour from other classes. In the Java programming language, each class is allowed to have one direct superclass, and each superclass has the potential for an unlimited number of subclasses.

## **INTERFACE**

An interface is a group of related methods with empty bodies.

## **PACKAGE**

A package is a namespace that organizes a set of related classes and interfaces.



**SOURCE CODE:**

```
import sqlite3

#it connects to the database

def connectDatabase():

    global conn

    conn=sqlite3.connect('DRPBAM.db')

    print("opened database successfully")


#This would create a table

def CreateTable():

#DOCTOR DETAILS

    conn.execute("CREATE TABLE IF NOT EXISTS DDETAILS

        (ID INT[20],

        DNAME CHAR(50) NOT NULL,

        ADDRESS CHAR(50),

        PHONE CHAR(15) NOT NULL,

        QUALIFICATION CHAR(20) NOT NULL,

        GENDER CHAR(10) NOT NULL);")

#ROOM DETAILS

    conn.execute("CREATE TABLE IF NOT EXISTS RDETAILS

        (PATIENTID INTEGER PRIMARY KEY AUTOINCREMENT,
```

PATIENTNAME CHAR[50],

ROOMID INT[20],

ROOMTYPE CHAR(50) NOT NULL;''')#ROOMTYPE MEANS:GENERAL/PRIVATE

#### #PATIENT DETAILS

```
conn.execute("CREATE TABLE IF NOT EXISTS PDETAILS
(PATIENTID INTEGER PRIMARY KEY AUTOINCREMENT,
PATIENTNAME CHAR(50) NOT NULL,
AGE INT(10),
GENDER CHAR(50),
ADDRESS CHAR(50),
DATE OF ADMISSION INT(20),
CONTACT NUMBER CHAR(15) NOT NULL);''')
```

#### #BILL DETAILS

```
conn.execute("CREATE TABLE IF NOT EXISTS BDETAILS
(BILLNO INT[20],
GENERATED BILLDATE INT(20) NOT NULL,
PATIENTID INTEGER PRIMARY KEY AUTOINCREMENT,
PNAME CHAR(50) NOT NULL,
AGE INT(10),
ADMISSIONDATE INT(20),
DISCHARGEDATE INT(20),
ROOM CHARGES INT(20),
PATHOLOGY FEES INT(50) NOT NULL,
DOCTOR CHECKUP FEE INT(50) NOT NULL,
MISCELLANEOUS FEE INT(50) NOT NULL,
TOTAL AMOUNT OF BILL INT(80) NOT NULL);''')
```

## #appointment details

```
conn.execute("CREATE TABLE IF NOT EXISTS ADETAILS
(PATIENTID INTEGER PRIMARY KEY AUTOINCREMENT,
PATIENTNAME VARCHAR[50],
DOCTORNAME VARCHAR[50],
CONSULTFEES VARCHAR[50],
APPOINTMENTDATE VARCHAR[50],
APPOINTMENTTIME INT[20],
TIMEPREFERRED INT[20],
APPOINTMENT NUMBER INT[20],
PAYMENTMODE CHAR[50]);")
```

## #MEDICINE DETAILS

```
conn.execute("CREATE TABLE IF NOT EXISTS MDETAILS
(PATIENTID INTEGER PRIMARY KEY AUTOINCREMENT,
PATIENTNAME VARCHAR[50],
DISEASEOFPATIENT VARCHAR[50],
MEDICINERECOMMENDED VARCHAR[50],
CONSULTEDDOCTORNAME VARCHAR[50]);")
```

```
print("table created successfully")
```

```
#this will close a connection to Database
```

```
def CloseDatabase():
```

```
    conn.close()
```

```
#this would insert a new record to database
```

```
def NewRecord():
```

```
#DDETAILS
```

```
    print("\n Doctor Details:")
```

```
    id=int(input("enter doctor's id:"))
```

```
    dname=input("enter the doctor's name:")
```

```
    address=input("enter the address:")
```

```
    phone=input("enter the phone number:")
```

```
    qualification=input("enter the qualification:")
```

```
    gender=input("enter gender:")
```

```
    conn.execute("INSERT INTO  
DDETAILS(ID,DNAME,ADDRESS,PHONE,QUALIFICATION,GENDER)VALUES(?,?,?,?,?)  
",(id,dname,address,phone,qualification,gender));
```

```
#RDETAILS
```

```
    print("\n Room Details:")
```

```
    patientid=int(input("enter patientid:"))
```

```
    patientname=input("enter patient name:")
```

```
    roomid=int(input(" enter roomid number"))
```

```
    roomtype=input("enter roomtype[general/private]")
```

```
    conn.execute("INSERT INTO RDETAILS\  
    (PATIENTID,PATIENTNAME,ROOMID,ROOMTYPE)\  
    VALUES(?,?,?,?)",(patientid,patientname,roomid,roomtype));
```

```
#PDETAILS
```

```
    print("\n Patient Details:")
```

```
    patientid=int(input("enter patient's id:"))
```

```
    patientname=input("enter patient's name:")
```

```
age=int(input("enter the patient's age:"))
gender=input("enter the gender:")
address=input("enter the address of patient:")
dateofadmission=int(input("enter date of admission:"))
contactnumber=input("enter contact number:")

conn.execute("INSERT INTO
PDETAILS(PATIENTID,PATIENTNAME,AGE,GENDER,ADDRESS,DATE,CONTACT)\

VALUES(?,?,?,?,?,?,?)",(patientid,patientname,age,gender,address,dateofadmission,co
ntactnumber));
```

#### #BDETAILS

```
print("\n Billing Details:")
billno=int(input("enter the bill num:"))
generatedbilldate=int(input("enter generated bill date:"))
patientid=int(input("enter patient's id:"))
patientname=input("enter patient's name:")
age=int(input("enter age"))
admissiondate=int(input("enter admission date:"))
dischargedate=int(input("enter discharge date:"))
roomcharges=int(input("enter room charges:"))
pathologyfees=int(input("enter pathology fees:"))
doctorcheckupfees=int(input("enter doctor checkup fees"))
miscellaneousfees=int(input("enter miscellaneous fees"))

totalamount=roomcharges+pathologyfees+doctorcheckupfees+miscellaneousfees
print("TOTAL=",totalamount)
conn.execute("INSERT INTO
```

```
BDETAILS(BILLNO,GENERATED,PATIENTID,PNAME,AGE,ADMISSIONDATE,DISCHARGE  
DATE,ROOM,PATHOLOGY,DOCTOR,MISCELLANEOUS,TOTAL)VALUES(?,?,?,?,?,?,?  
?,?,?,?,?),(billno,generatedbilldate,patientid,patientname,age,admissiondate,discharge  
date,roomcharges,pathologyfees,doctorcheckupfees,miscellaneousfees,totalamount)  
);
```

#### #ADETAILS

```
print("\n Appointment Details:")  
  
patientid=int(input("enter patient id:"))  
  
patientname=input("\n enter the name of patient:")  
  
doctorname=input("enter the name of doctor:")  
  
consultfees=input("enter the consult fees:")  
  
appointmentdate=input("enter the date of appointment:")  
  
appointmenttime=int(input("enter the appointmenttime:"))  
  
timepreferred=int(input("enter the preferred time:"))  
  
appointmentnumber=int(input("enter the appointment number:"))  
  
paymentmode=input("enter the mode of payment:")  
  
conn.execute("INSERT INTO  
ADETAILS(PATIENTID,PATIENTNAME,DOCTORNAME,CONSULTFEES,APPOINTMEN  
TDATE,APPOINTMENTTIME,TIMEPREFERRED,APPOINTMENT,PAYMENTMODE)VAL  
UES(?,?,?,?,?,?,?,?)",(patientid,patientname,doctorname,consultfees,appointmentdat  
e,appointmenttime,timepreferred,appointmentnumber,paymentmode));
```

#### #MDETAILS

```
print("\n Medicine Details:")  
  
patientid=int(input("enter patient id:"))  
  
patientname=input("enter name of patient:")  
  
diseaseofpatient=input("enter name of disease:")  
  
medicinerecommended=input("enter name of medicine:")
```

```
consulteddoctorname=input("enter name of recommended doctor:")

conn.execute("INSERT INTO
MDETAILS(PATIENTID,PATIENTNAME,DISEASEOFPATIENT,MEDICINERECOMMEND
ED,CONSULTEDDOCTORNAME)VALUES(?,?,?,?), (patientid,patientname,diseaseofp
atient,medicinerecommended,consulteddoctorname));

conn.commit()
```

#this would display doctor details separately

```
def getDataDDetails():
```

```
    cursor=conn.execute("SELECT
DNAME,ADDRESS,PHONE,QUALIFICATION,GENDER FROM DDETAILS")
```

```
    for row in cursor:
```

```
        print("DNAME=",row[0])
```

```
        print("ADDRESS=",row[1])
```

```
        print("PHONE=",row[2])
```

```
        print("QUALIFICATION=",row[3])
```

```
        print("GENDER=",row[4],"\n")
```

#this would display patient's room details

```
def getDataRDetails():
```

```
    cursor=conn.execute("SELECT PATIENTID,PATIENTNAME, ROOMID,ROOMTYPE
FROM RDETAILS")
```

```
    for row in cursor:
```

```
        print("PATIENTID=",row[0])
```

```
        print("PNAME=",row[1])
```

```
        print("ROOMID=",row[2])
```

```
        print("ROOMTYPE=",row[3],"\n")
```

#this would display patient details

def getDatsPDetails():

```
    cursor=conn.execute("SELECT
PATIENTID,PATIENTNAME,AGE,GENDER,ADDRESS,DATE OFADMISSION,CONTACT
NUMBER FROM PDETAILS")
```

for row in cursor:

```
        print("PATIENTID=",row[0])
        print("PNAME=",row[1])
        print("AGE=",row[2])
        print("GENDER=",row[3])
        print("ADDRESS=",row[4])
        print("DATE OF ADMISSION=",row[5])
        print("CONTACT NUMBER=",row[6],"\n")
```

#this would display patient's billing details

def getDataBDetails():

```
    cursor=conn.execute("SELECT
BILLNO,GENERATED,PATIENTID,PNAME,AGE,ADMISSIONDATE,DISCHARGEDATE,R
OOM,PATHOLOGY,DOCTOR,MISCELLANEOUS,TOTAL FROM BDETAILS")
```

for row in cursor:

```
        print("BILL NO=",row[0])
        print("GENERATED BILL DATE=",row[1])
        print("PATIENT ID=",row[2])
        print("PNAME=",row[3])
        print("AGE=",row[4])
        print("DATE OF ADMISSION=",row[5])
```



```
print("DATE OF DISCHARGE=",row[6])
print("ROOM CHARGES=",row[7])
print("PATHOLOGY FEES=",row[8])
print("DOCTOR CHECKUP FEES=",row[9])
print("MISCELLANEOUS FEES=",row[10])
print("TOTAL AMOUNT OF BILL=",row[11],"\n")
```

#this would display patient's appointment details

def getDataADetails():

```
cursor=conn.execute("SELECT PATIENTID,
PATIENTNAME,DOCTORNAME,CONSULTFEES,APPOINTMENTDATE,APPOINTMENT
TIME,TIMEPREFERRED,APPOINTMENT,PAYMENTMODE FROM ADETAILS")
```

for row in cursor:

```
print("PATIENTID=",row[0])
print("PATIENTNAME=",row[1])
print("DOCTORNAME=",row[2])
print("CONSULTFEES=",row[3])
print("APPOINTMENTDATE=",row[4])
print("APPOINTMENTTIME=",row[5])
print("TIMEPREFERRED=",row[6])
print("APPOINTMENTNUMBER=",row[7])
print("PAYMENTMODE=",row[8],"\n")
```

#this would display patient's medicine details

def getDataMDetails():

```
cursor=conn.execute("SELECT  
PATIENTID,PATIENTNAME,DISEASEOFPATIENT,MEDICINERECOMMENDED,CONSUL  
TEDDOCTORNAME FROM MDETAILS")
```

```
for row in cursor:
```

```
    print("PATIENTID=",row[0])
```

```
    print("PATIENTNAME=",row[1])
```

```
    print("DISEASEOFPATIENT=",row[2])
```

```
    print("MEDICINERECOMMENDED=",row[3])
```

```
    print("CONSULTEDDOCTORNAME=",row[4],"\\n")
```

```
#this would display all information about the patient
```

```
def getData():
```

```
    getDataDDetails()
```

```
    getDataRDetails()
```

```
    getDatsPDetails()
```

```
    getDataBDetails()
```

```
    getDataADetails()
```

```
    getDataMDetails()
```

```
#this would update patient's information
```

```
def UpdatePDetails():
```

```
#PDETAILS
```

```
    print("\\n Patient Details:")
```

```
    patientid=int(input("enter patient's id:"))
```

```
    patientname=input("enter patient's name:")
```

```
    age=int(input("enter the patient's age:"))
```

```
gender=input("enter the gender:")
address=input("enter the address of patient:")
dateofadmission=int(input("enter date of admission:"))
contactnumber=input("enter contact number:")
```

```
conn.execute("UPDATE PDETAILS SET
PATIENTNAME=?,AGE=?,GENDER=?,ADDRESS=?,DATE=?,CONTACT=? WHERE
PATIENTID=?", (patientname,age,gender,address,dateofadmission,contactnumber,patientid))

conn.commit()

print("total number of rows updated:",conn.total_changes)
```

#this would update patient's doctor information

```
def UpdateDDetails():
```

```
#DDETAILS
```

```
print("\n Doctor Details:")
id=int(input("enter doctor's id:"))
dname=input("enter the doctor's name:")
address=input("enter the address:")
phone=input("enter the phone number:")
qualification=input("enter the qualification:")
gender=input("enter gender:")
```

```
conn.execute("UPDATE DDETAILS SET
DNAME=?,ADDRESS=?,PHONE=?,QUALIFICATION=?,GENDER=? WHERE
ID=?", (dname,address,phone,qualification,gender,id))

conn.commit()

print("total number of rows updated:",conn.total_changes)
```

#this would update patient's room details

def UpdateRDetails():

#RDETAILS

print("\n Room Details:")

patientid=int(input("enter patientid:"))

patientname=input("enter patient name:")

roomid=int(input(" enter roomid number"))

roomtype=input("enter roomtype[general/private]")

conn.execute("UPDATE RDETAILS SET PATIENTNAME=?,ROOMID=?,ROOMTYPE=?  
WHERE PATIENTID=?", (patientname,roomid,roomtype,patientid))

conn.commit()

print("total number of rows updated:",conn.total\_changes)

#this would update patient's billing details

def UpdateBDetails():

#BDETAILS

print("\n Billing Details:")

billno=int(input("enter the bill num:"))

generatedbilldate=int(input("enter generated bill date:"))

patientid=int(input("enter patient's id:"))

patientname=input("enter patient's name:")

age=int(input("enter age"))

admissiondate=int(input("enter admission date:"))

dischargedate=int(input("enter discharge date:"))

roomcharges=int(input("enter room charges:"))

```
pathologyfees=int(input("enter pathology fees:"))
```

```
doctorcheckupfees=int(input("enter doctor checkup fees"))
```

```
miscellaneousfees=int(input("enter miscellaneous fees"))
```

```
totalamount=roomcharges+pathologyfees+doctorcheckupfees+miscellaneousfees
```

```
print("TOTAL=",totalamount)
```

```
conn.execute("UPDATE BDETAILS SET  
BILLNO=?,GENERATED=?,PNAME=?,AGE=?,ADMISSIONDATE=?,DISCHARGEDATE=?,  
ROOM=?,PATHOLOGY=?,DOCTOR=?,MISCELLANEOUS=?,TOTAL=? WHERE  
PATIENTID=?", (billno,generatedbilldate,patientname,age,admissiondate,dischargedate,roomcharges,pathologyfees,doctorcheckupfees,miscellaneousfees,totalamount,patientid))
```

```
conn.commit()
```

```
print("total number of rows updated:",conn.total_changes)
```

```
#this would update patient's appointment details
```

```
def UpdateADetails():
```

```
#ADetails
```

```
print("\n Appointment Details:")
```

```
patientid=int(input("enter patient id:"))
```

```
patientname=input("enter the name of patient:")
```

```
doctorname=input("enter the name of doctor:")
```

```
consultfees=input("enter the consult fees:")
```

```
appointmentdate=input("enter the date of appointment:")
```

```
appointmenttime=int(input("enter the appointmenttime:"))
```

```
timepreferred=int(input("enter the preferred time:"))
```

```
appointmentnumber=int(input("enter the appointment number:"))
```

```
paymentmode=input("enter the mode of payment:")
```

```
conn.execute("UPDATE ADETAILS SET
PATIENTNAME=?,DOCTORNAME=?,CONSULTFEES=?,APPOINTMENTDATE=?,APPOI
NTMENTTIME=?,TIMEPREFERRED=?,APPOINTMENT=?,PAYMENTMODE=? WHERE
PATIENTID=?", (patientname,doctorname,consultfees,appointmentdate,appointmentt
ime,timepreferred,appointmentnumber,paymentmode,patientid))
```

```
conn.commit()
```

```
print("total number of rows updated:",conn.total_changes)
```

```
#this would update patient's medicine details
```

```
def UpdateMDetails():
```

```
#MDETAILS
```

```
print("\n Medicine Details:")
```

```
patientid=int(input("enter patient id:"))
```

```
patientname=input("\n enter name of patient:")
```

```
diseaseofpatient=input("enter name of disease:")
```

```
medicinerecommended=input("enter name of medicine:")
```

```
consulteddoctorname=input("enter name of recommended doctor:")
```

```
conn.execute("UPDATE MDETAILS SET
PATIENTNAME=?,DISEASEOFPATIENT=?,MEDICINERECOMMENDED=?,CONSULTED
DOCTORNAME=? WHERE
PATIENTID=?", (patientname,diseaseofpatient,medicinerecommended,consulteddoct
orname,patientid))
```

```
print("total number of rows updated:",conn.total_changes)
```

```
conn.commit()
```

```
print("total number of rows updated:",conn.total_changes)
```

```
#this would update all information
```

```
def UpdateData():
```

UpdatePDetails()

UpdateDDetails()

UpdateRDetails()

UpdateBDetails()

UpdateADetails()

UpdateMDetails()

def DeleteData():

    patientid=int(input("enter the patientid to be deleted:"))

    conn.execute("DELETE FROM DDETAILS WHERE ID=?", (patientid,))

    conn.execute("DELETE FROM RDETAILS WHERE PATIENTID=?", (patientid,))

    conn.execute("DELETE FROM PDETAILS WHERE PATIENTID=?", (patientid,))

    conn.execute("DELETE FROM BDETAILS WHERE PATIENTID=?", (patientid,))

    conn.execute("DELETE FROM ADETAILS WHERE PATIENTID=?", (patientid,))

    conn.execute("DELETE FROM MDETAILS WHERE PATIENTID=?", (patientid,))

    conn.commit()

    print("Total number of rows deleted:", conn.total\_changes)

#this would search patient's information-ALL

def SearchPatientInfo():

    patientid=int(input("enter the patient id:"))

    cursor=conn.execute("SELECT  
PD.PATIENTID,PD.PATIENTNAME,PD.AGE,PD.GENDER,PD.ADDRESS,PD.DATE,PD.C  
ONTACT,RD.ROOMID,RD.ROOMTYPE,\n

BD.BILLNO,BD.GENERATED,BD.ADMISSIONDATE,BD.DISCHARGEDATE,BD.ROOM,BD.  
PATHOLOGY,BD.DOCTOR,BD.MISCELLANEOUS,BD.TOTAL,\

AD.DOCTORNAME,AD.CONSULTFEES,AD.APPOINTMENTDATE,AD.APPOINTMENTTI  
ME,AD.TIMEPREFERRED,AD.APPOINTMENT,AD.PAYMENTMODE,\

MD.DISEASEOFPATIENT,MD.MEDICINERECOMMENDED,MD.CONSULTEDDOCTORN  
AME\

FROM PDETAILS AS PD,RDETAILS AS RD,BDETAILS AS BD,ADETAILS AS  
AD,MDETAILS AS MD\

WHERE

PD.PATIENTID=RD.PATIENTID=BD.PATIENTID=AD.PATIENTID=MD.PATIENTID AND  
PD.PATIENTID=?",(patientid,))

for row in cursor:

print("PATIENT DETAILS:")

print("PATIENTID=",row[0])

print("PNAME=",row[1])

print("AGE=",row[2])

print("GENDER=",row[3])

print("ADDRESS=",row[4])

print("DATE OF ADMISSION=",row[5])

print("CONTACT NUMBER=",row[6],"\n")

print("ROOM DETAILS:")

print("ROOMID=",row[7])

print("ROOMTYPE=",row[8],"\n")

print("BILLING DETAILS:")

print("BILL NO=",row[9])

print("GENERATED BILL DATE=",row[10])



```
print("DATE OF ADMISSION=",row[11])
print("DATE OF DISCHARGE=",row[12])
print("ROOM CHARGES=",row[13])
print("PATHOLOGY FEES=",row[14])
print("DOCTOR CHECKUP FEES=",row[15])
print("MISCELLANEOUS FEES=",row[16])
print("TOTAL AMOUNT OF BILL=",row[17],"\n")
print("APPOINTMENT DETAILS:")
print("DOCTORNAME=",row[18])
print("CONSULTFEES=",row[19])
print("APPOINTMENTDATE=",row[20])
print("APPOINTMENTTIME=",row[21])
print("TIMEPREFERRED=",row[22])
print("APPOINTMENTNUMBER=",row[23])
print("PAYMENTMODE=",row[24],"\n")
print("MEDICINE DETAILS:")
print("DISEASEOFPATIENT=",row[25])
print("MEDICINERECOMMENDED=",row[26])
print("CONSULTEDDOCTORNAME=",row[27],"\n")
```

#this would search only patient details

```
def SearchPDetails():
```

```
    patientid=int(input("enter the patient's id to know his/her details:"))
```

```
    cursor=conn.execute("SELECT
PATIENTID,PATIENTNAME,AGE,GENDER,ADDRESS,DATE,CONTACT FROM
PDETAILS\
```

```
        WHERE PATIENTID=?",(patientid,))
```

```
for row in cursor:
```

```
    print("PATIENT DETAILS:")
    print("PATIENTID=",row[0])
    print("PNAME=",row[1])
    print("AGE=",row[2])
    print("GENDER=",row[3])
    print("ADDRESS=",row[4])
    print("DATE OF ADMISSION=",row[5])
    print("CONTACT NUMBER=",row[6],"\n")
```

```
connectDatabase()
```

```
CreateTable()
```

```
while True:
```

```
    print("choose the options:\n1.New Record\n 2.view Individual Data-All")
    print("3.view All Data\n 4.Update Data\n")
    print("5.Search information\n 6.Search PatientDetails")
    print("7.delete data")
    print("8.exit")
```

```
    data=int(input())
```

```
    if data==8:
```

```
        break;
```

```
    elif data==1:
```

```
        NewRecord()
```

```
    elif data==2:
```

```
        while True:
```

```
            print("choose the options:\n 1.view Doctor Details\n 2.view Room Details\n\n 3.view patient details\n 4.view Bill Details\n 5.view Appointment Details\n 6.view
```

Medicine Details\n 7.exit")

data=int(input())

if data==7:

break;

elif data==1:

getDataDDetails()

elif data==2:

getDataRDetails()

elif data==3:

getDatsPDetails()

elif data==4:

getDataBDetails()

elif data==5:

getDataADetails()

elif data==6:

getDataMDetails()

else:

print("enter a valid option")

elif data==3:

getData()

elif data==4:

while True:

print("choose options:\n 1.update individual records\n 2.update all data\n 3.exit")

data=int(input())

```
    if data==3:
        break;
    elif data==1:
        while True:
            print("choose options:\n 1.update patient details\n 2.update doctor
details\n 3.update room details\n 4.update bill details\n 5.update appointment
details\n 6.update medicine details\n 7.exit")
            data=int(input())
            if data==7:
                break;
            elif data==1:
                UpdatePDetails()
            elif data==2:
                UpdateDDetails()
            elif data==3:
                UpdateRDetails()
            elif data==4:
                UpdateBDetails()
            elif data==5:
                UpdateADetails()
            elif data==6:
                UpdateMDetails()
            else:
                print("enter a valid option")

elif data==2:
    UpdateData()
```

```
elif data==5:
```

```
    SearchPatientInfo()
```

```
elif data==6:
```

```
    SearchPDetails()
```

```
elif data==7:
```

```
    DeleteData()
```

```
else:
```

```
    print("enter a valid option:")
```

```
CloseDatabase()
```

```
print("THANK YOU")
```

## OUTPUT:

### Database Design:

#### 1) Doctor Details Database Details:

DB Browser for SQLite - C:\Users\prema a\Downloads\DRPBAM.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save

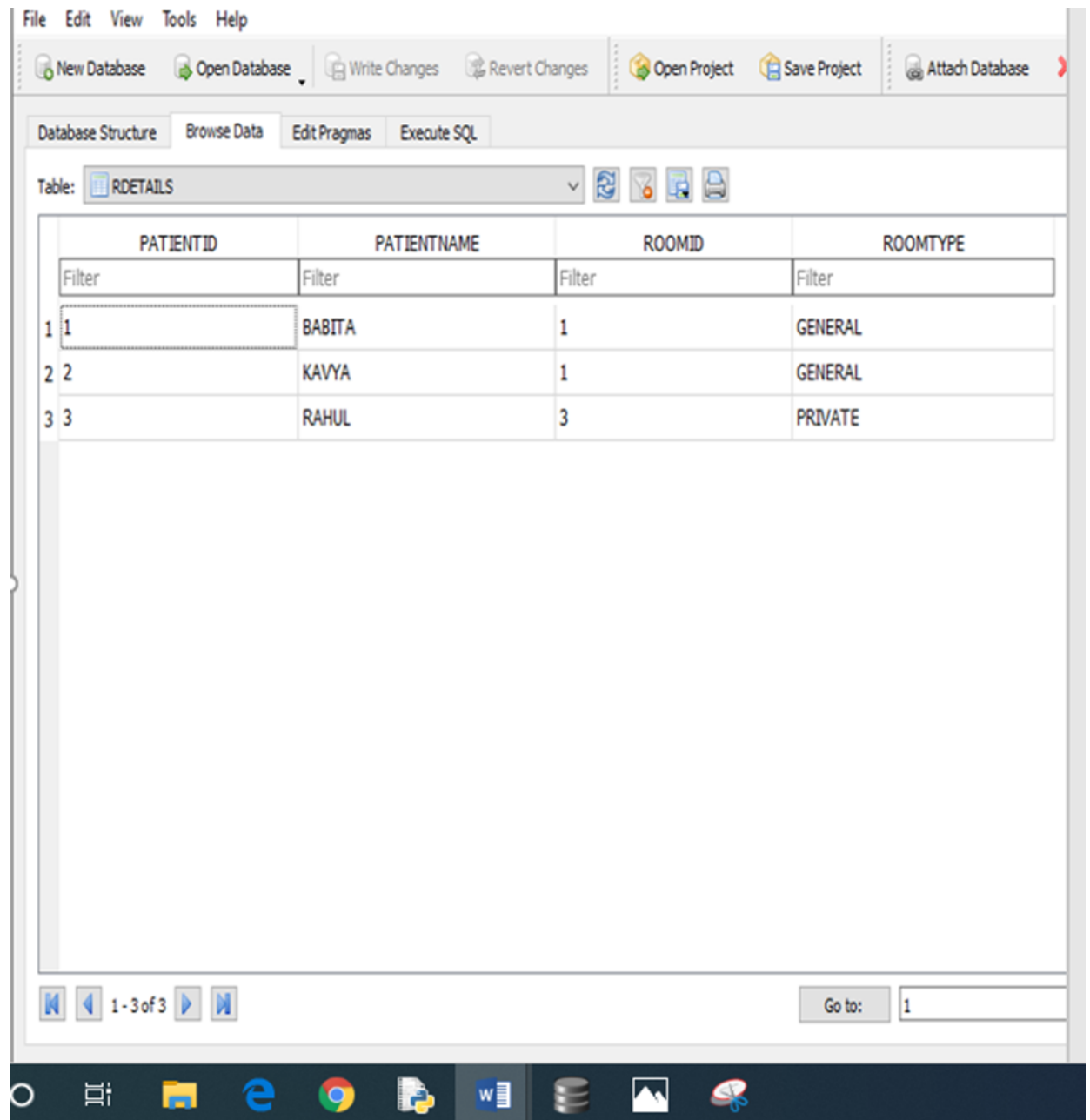
Database Structure Browse Data Edit Pragmas Execute SQL

Table: DDETAILS

	ID	DNAME	ADDRESS	PHONE	QUALIFICATION	GENDER
	Filter	Filter	Filter	Filter	Filter	Filter
1	1	CHARAN	MUMBAI	8296319657	MBBS	MALE
2	11	ROHIT	BLORE	8762147502	MBBS	MALE
3	111	IQRA	BPET	8296319523	MD	FEMALE

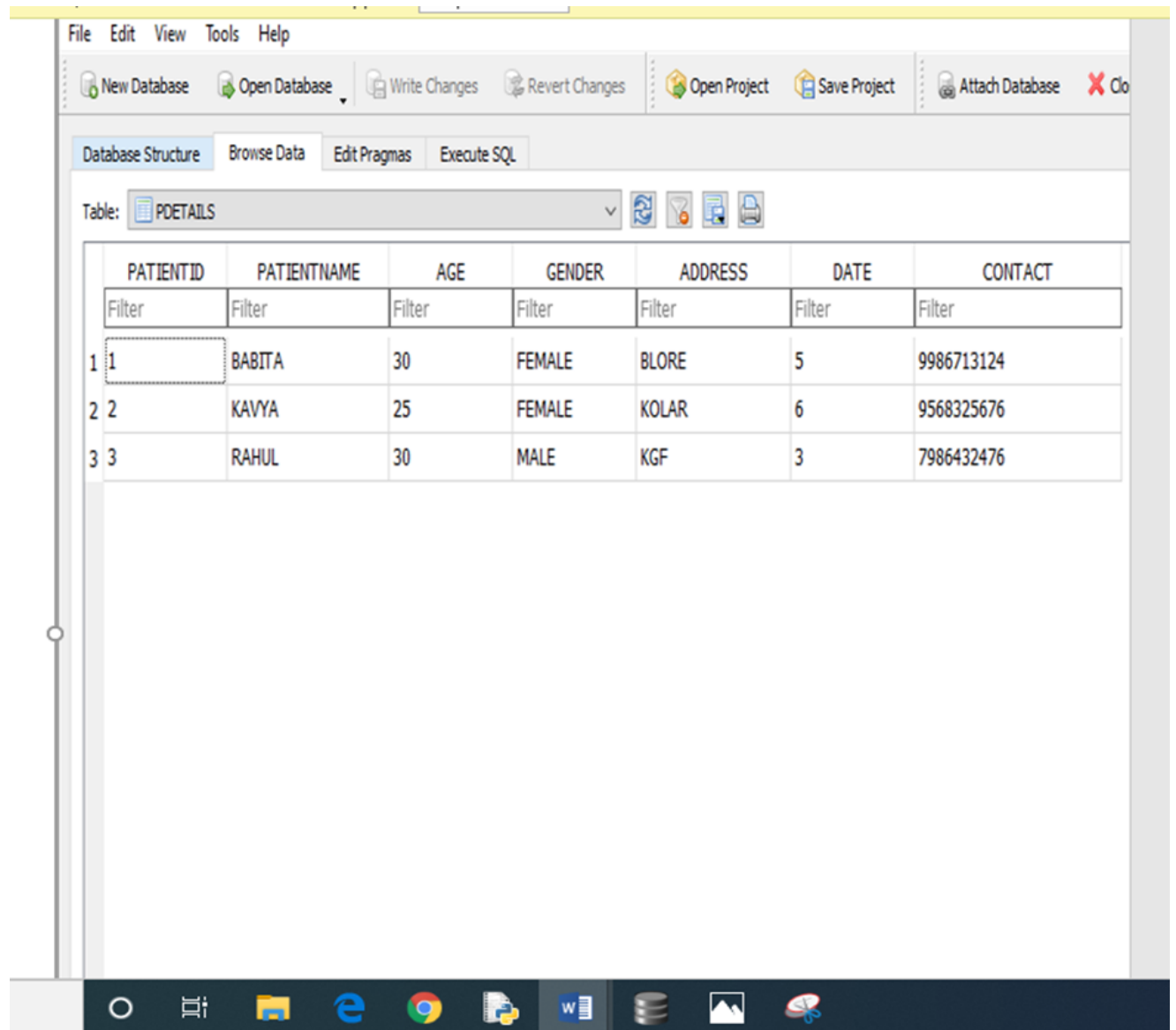
1 - 3 of 3

## **2) Room Details Database Design:**



### 3) Patient Details Database Design:





#### 4) Billing Details Database Design:

DB Browser for SQLite - C:\Users\prema a\Downloads\DRPBAM.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

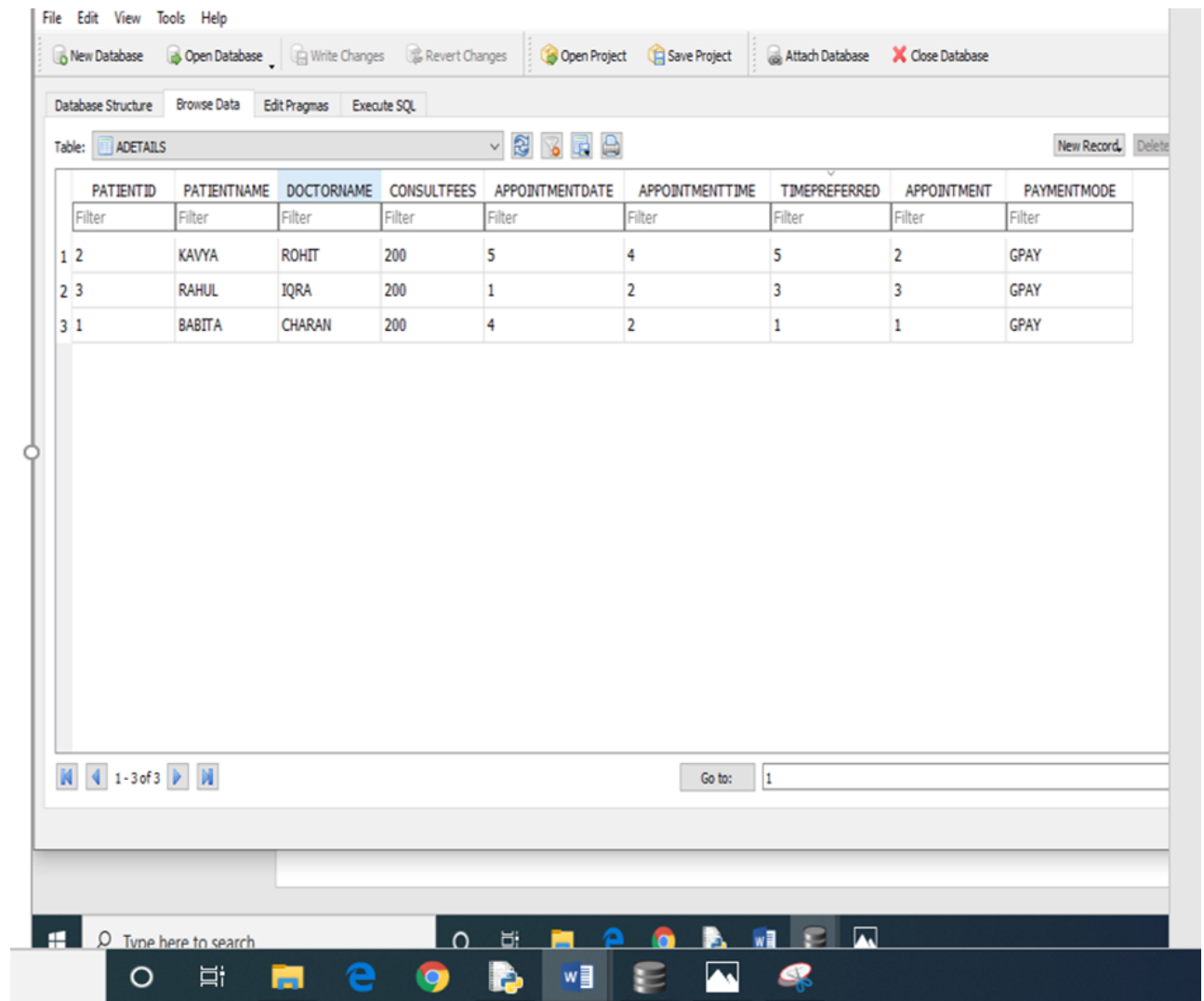
Table: BDETAILS

	BILLNO	GENERATED	PATIENTID	PNAME	AGE	ADMISSIONDATE	DISCHARGEDATE	ROOM	PATHOLOGY	DOCTOR	MISCELLANEOUS	TOTAL
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	6	1	BABITA	30	5	6	100	100	150	50	400
2	2	7	2	KAVYA	25	6	7	100	50	150	50	350
3	3	4	3	RAHUL	30	3	5	150	150	200	50	550

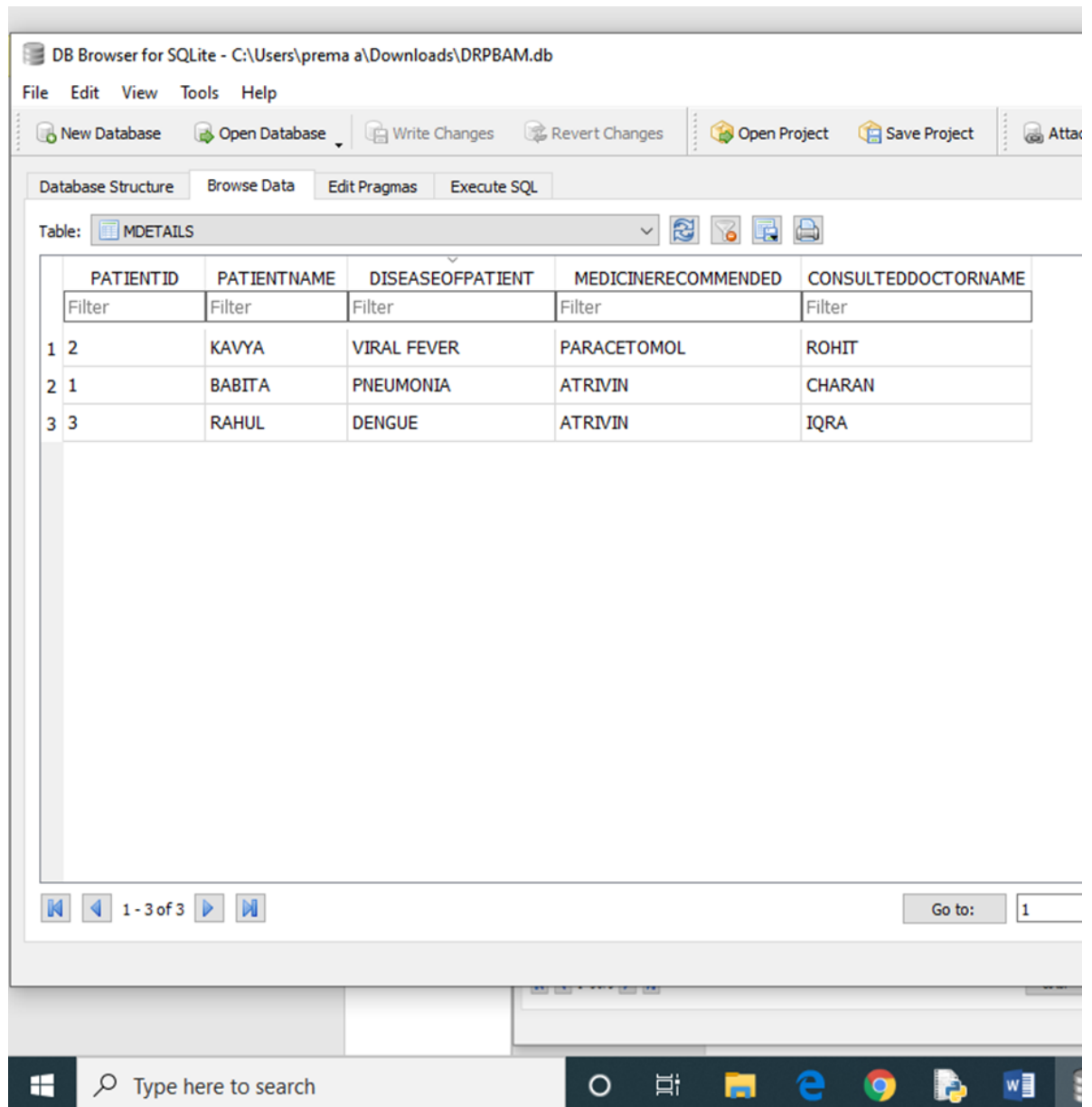
1 - 3 of 3

Go to: 1

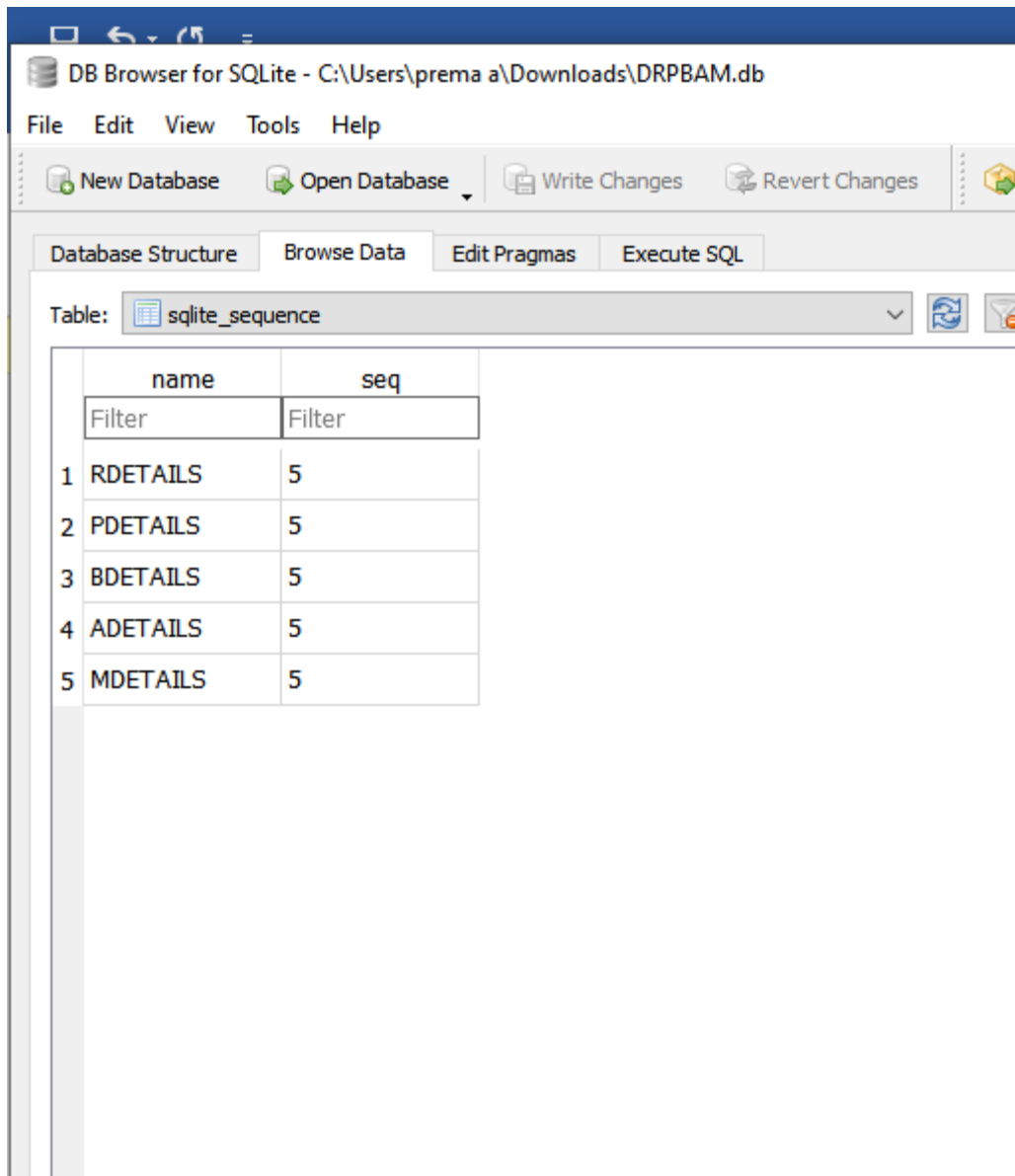
### **5) Appointment Details Database Design:**



## 6) Medicine Details Database Design:



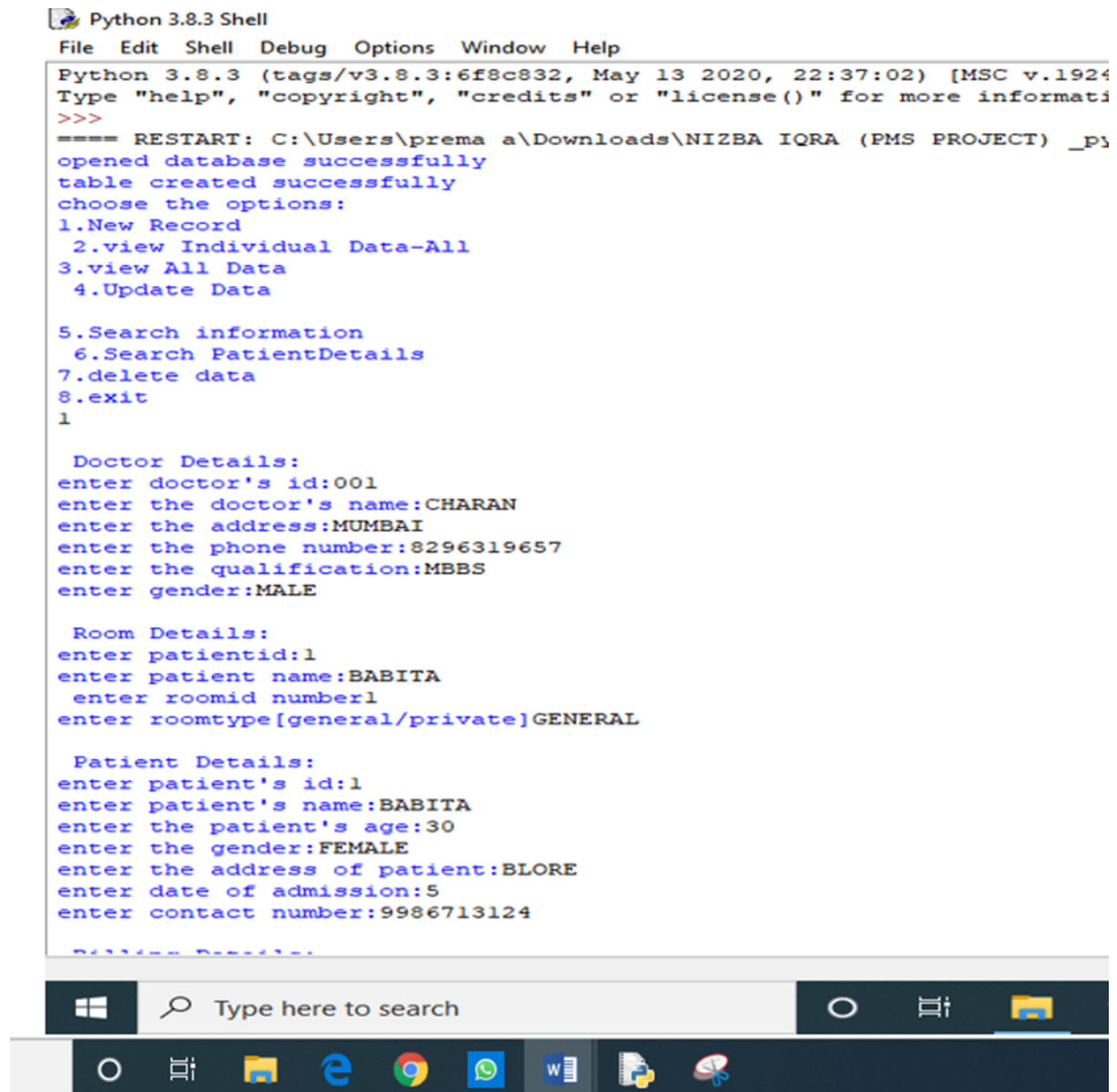
**Sqlite sequence:**



The screenshot shows the DB Browser for SQLite application window. The title bar reads "DB Browser for SQLite - C:\Users\prema a\Downloads\DRPBAM.db". The menu bar includes "File", "Edit", "View", "Tools", and "Help". The toolbar contains icons for "New Database", "Open Database", "Write Changes", and "Revert Changes". The main interface has four tabs: "Database Structure", "Browse Data", "Edit Pragas", and "Execute SQL". The "Browse Data" tab is active, showing a table named "sqlite\_sequence". The table has two columns: "name" and "seq". The data is as follows:

	name	seq
	Filter	Filter
1	RDETAILS	5
2	PDETAILS	5
3	BDETAILS	5
4	ADETAILS	5
5	MDETAILS	5

**Output:**



```
Python 3.8.3 Shell
File Edit Shell Debug Options Window Help
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924
Type "help", "copyright", "credits" or "license()" for more informati
>>>
==== RESTART: C:\Users\prema a\Downloads\NIZBA IQRA (PMS PROJECT) _ps
opened database successfully
table created successfully
choose the options:
1.New Record
2.view Individual Data-All
3.view All Data
4.Update Data

5.Search information
6.Search PatientDetails
7.delete data
8.exit
1

    Doctor Details:
enter doctor's id:001
enter the doctor's name:CHARAN
enter the address:MUMBAI
enter the phone number:8296319657
enter the qualification:MBBS
enter gender:MALE

    Room Details:
enter patientid:1
enter patient name:BABITA
    enter roomid number1
enter roomtype[general/private]GENERAL

    Patient Details:
enter patient's id:1
enter patient's name:BABITA
enter the patient's age:30
enter the gender:FEMALE
enter the address of patient:BLORE
enter date of admission:5
enter contact number:9986713124

    Patient Details:
```

```
File Edit Shell Debug Options Window Help
Billing Details:
enter the bill num:1
enter generated bill date:6
enter patient's id:1
enter patient's name:BABITA
enter age30
enter admission date:5
enter discharge date:6
enter room charges:100
enter pathology fees:100
enter doctor checkup fees150
enter miscellaneous fees50
TOTAL= 400

Appointment Details:
enter patient id:1
enter the name of patient:BABITA
enter the name of doctor:CHARAN
enter the consult fees:200
enter the date of appointment:4
enter the appointmenttime:2
enter the preferred time:1
enter the appointment number:1
enter the mode of payment:GPAY

Medicine Details:
enter patient id:1
enter name of patient:BABITA
enter name of disease:PNEUMONIA
enter name of medicine:ATRIVIN
enter name of recommended doctor:CHARAN
choose the options:
1.New Record
2.view Individual Data-All
3.view All Data
4.Update Data

5.Search information
6.Search PatientDetails
7.delete data
8.exit
,
```





ed, but first we need to close some apps.

Update now

Python 3.8.3 Shell

File Edit Shell Debug Options Window Help

6.Search PatientDetails

7.delete data

8.exit

1

Doctor Details:

enter doctor's id:011

enter the doctor's name:ROHIT

enter the address:BLORE

enter the phone number:8762147502

enter the qualification:MBBS

enter gender:MALE

Room Details:

enter patientid:2

enter patient name:KAVYA

enter roomid number1

enter roomtype[general/private]GENERAL

Patient Details:

enter patient's id:2

)enter patient's name:KAVYA

enter the patient's age:25

enter the gender:FEMALE

enter the address of patient:KOLAR

enter date of admission:6

enter contact number:9568325676

Billing Details:

enter the bill num:2

enter generated bill date:7

enter patient's id:2

enter patient's name:KAVYA

enter age25

enter admission date:6

enter discharge date:7

enter room charges:100

enter pathology fees:50

enter doctor checkup fees150

enter miscellaneous fees50

TOTAL= 350



```
Python 3.8.3 Shell
File Edit Shell Debug Options Window Help

Appointment Details:
enter patient id:2
enter the name of patient:KAVYA
enter the name of doctor:ROHIT
enter the consult fees:200
enter the date of appointment:5
enter the appointmenttime:4
enter the preferred time:5
enter the appointment number:2
enter the mode of payment:GPAY

Medicine Details:
enter patient id:2
enter name of patient:KAVYA
enter name of disease:VIRAL FEVER
enter name of medicine:PARACETOMOL
enter name of recommended doctor:ROHIT
choose the options:
1.New Record
2.view Individual Data-All
3.view All Data
4.Update Data

5.Search information
6.Search PatientDetails
7.delete data
8.exit
1

Doctor Details:
enter doctor's id:111
enter the doctor's name:IQRA
enter the address:BPET
enter the phone number:8296319523
enter the qualification:MD
enter gender:FEMALE

Room Details:
enter patientid:3
enter patient name:RAHUL
enter roomid number3
enter roomtype (general/private) PRIVATE
```

lled, but first we need to close some apps.

Update now

```
Room Details:
enter patientid:3
enter patient name:RAHUL
enter roomid number3
enter roomtype[general/private]PRIVATE
```

```
Patient Details:
enter patient's id:3
enter patient's name:RAHUL
enter the patient's age:30
enter the gender:MALE
enter the address of patient:KGF
enter date of admission:3
enter contact number:7986432476
```

```
Billing Details:
enter the bill num:3
enter generated bill date:4
enter patient's id:3
enter patient's name:RAHUL
enter age30
enter admission date:3
enter discharge date:5
enter room charges:150
enter pathology fees:150
enter doctor checkup fees200
enter miscellaneous fees50
TOTAL= 550
```

```
Appointment Details:
enter patient id:3
enter the name of patient:RAHUL
enter the name of doctor:IQRA
enter the consult fees:200
enter the date of appointment:1
enter the appointementtime:2
enter the preferred time:3
enter the appointment number:3
enter the mode of payment:GPAY
```



led, but first we need to close some apps.

Update now

```
BILL NO= 2
GENERATED BILL DATE= 7
PATIENT ID= 2
PNAME= KAVYA
AGE= 25
DATE OF ADMISSION= 6
DATE OF DISCHARGE= 7
ROOM CHARGES= 100
PATHOLOGY FEES= 50
DOCTOR CHECKUP FEES= 150
MISCELLANEOUS FEES= 50
TOTAL AMOUNT OF BILL= 350
```

```
BILL NO= 3
GENERATED BILL DATE= 4
PATIENT ID= 3
PNAME= RAHUL
AGE= 30
DATE OF ADMISSION= 3
DATE OF DISCHARGE= 5
ROOM CHARGES= 150
PATHOLOGY FEES= 150
DOCTOR CHECKUP FEES= 200
MISCELLANEOUS FEES= 50
TOTAL AMOUNT OF BILL= 550
```

```
choose the options:
1.view Doctor Details
2.view Room Details
3.view patient details
4.view Bill Details
5.view Appointment Details
6.view Medicine Details
7.exit
```

```
7
choose the options:
1.New Record
2.view Individual Data-All
3.view All Data
4.Update Data
```



```
5.Search information
6.Search PatientDetails
7.delete data
8.exit
5
enter the patient id:3
PATIENTID= 3
PNAME= RAHUL
AGE= 30
GENDER= MALE
ADDRESS= KGF
DATE OF ADMISSION= 3
CONTACT NUMBER= 7986432476

ROOMID= 3
ROOMTYPE= PRIVATE

BILL NO= 1
GENERATED BILL DATE= 6
DATE OF ADMISSION= 5
DATE OF DISCHARGE= 6
ROOM CHARGES= 100
PATHOLOGY FEES= 100
DOCTOR CHECKUP FEES= 150
MISCELLANEOUS FEES= 50
TOTAL AMOUNT OF BILL= 400

DOCTORNAME= CHARAN
CONSULTFEES= 200
APPOINTMENTDATE= 4
APPOINTMENTTIME= 2
TIMEPREFERRED= 1
APPOINTMENTNUMBER= 1
PAYMENTMODE= GPAY

DISEASEOFPATIENT= PNEUMONIA
MEDICINERECOMMENDED= ATRIVIN
CONSULTEDOCTORNAME= CHARAN
```



```
ed, but first we need to close some apps. Update now
APPOINTMENTTIME= 2
TIMEPREFERRED= 1
APPOINTMENTNUMBER= 1
PAYMENTMODE= GPAY

DISEASEOFFPATIENT= PNEUMONIA
MEDICINERECOMMENDED= ATRIVIN
CONSULTEDDOCTORNAME= CHARAN

choose the options:
1.New Record
2.view Individual Data-All
3.view All Data
4.Update Data

5.Search information
6.Search PatientDetails
7.delete data
8.exit
6
enter the patient's id to know his/her details:1
PATIENTID= 1
PNAME= BABITA
AGE= 30
GENDER= FEMALE
ADDRESS= BLORE
DATE OF ADMISSION= 5
CONTACT NUMBER= 9986713124

choose the options:
1.New Record
2.view Individual Data-All
3.view All Data
4.Update Data

5.Search information
6.Search PatientDetails
7.delete data
8.exit
8
THANK YOU
>>> |
```

## CONCLUSION:

The Project Patient Management System is for computerizing the working in hospital. It is a great improvement over the manual system. The computerization of the system has speed up the process. In the current system, the front office managing is very slow. The Patient managing system was thoroughly checked and tested with dummy data and thus is found to be very reliable. The software takes care of all the requirements of an average hospital and is capable to provide easy and effective storage of information related to patients that can be to the hospital.

It provides effective storage of information related to patient. It also provides billing facility on the basis of patient's status.

Since we are entering details of the patients electronically in the "Patient Management System", data will be secured. Using this application we can retrieve patient's history with a single click. Thus processing information will be faster. It guarantees accurate maintenance of Patient details. It easily reduces the book keeping task and thus reduces the human effort and increases accuracy speed.

## FURTHER ENHANCEMENTS

The proposed system is Patient Management System. We can enhance this system by including more facilities like pharmacy system for the stock details of medicines in the pharmacy.

Providing such features enable the users to include more comments into the system.



