Database Management System (COCSC05)

HOSPITAL MANAGEMENT SYSTEM

Submitted To:- Dr. Veenu

Submitted By:-

Sumit Dhote (2022UCS1602)

Anshul Goyal (2022UCS1597)

Ojas Nagdawane (2022UCS1618)

Project Report

Overview:

The Hospital Management System (HMS) is a comprehensive solution designed to streamline and digitalize various processes within a hospital environment. This project aims to enhance efficiency, accuracy, and accessibility of information across different departments.

Background:

Healthcare institutions often face challenges in managing patient information, appointments, billing, and other administrative tasks. The Hospital Management System is developed to address these challenges by introducing a centralized digital platform.

The current hospital management system is plagued by inefficiencies and limitations that hinder the delivery of optimal healthcare services. These issues range from administrative bottlenecks to patient care challenges, and they impede the overall effectiveness of the healthcare institution. As such, there is an urgent need to develop and implement an advanced hospital management system.

Objectives:

- Automate hospital processes for improved efficiency.
- Enhance patient care by facilitating quick access to medical records.
- Improve resource management, including staff scheduling and inventory control.
- Ensure accurate billing and financial tracking.

Scope :

The system covers the following functionalities:

- It provides the hospital management to add, view and delete data of the patients, staff and various facilities available in a hospital.

- Using this the hospital can manage all their available data in an organised manner

Technologies and Tools Used:

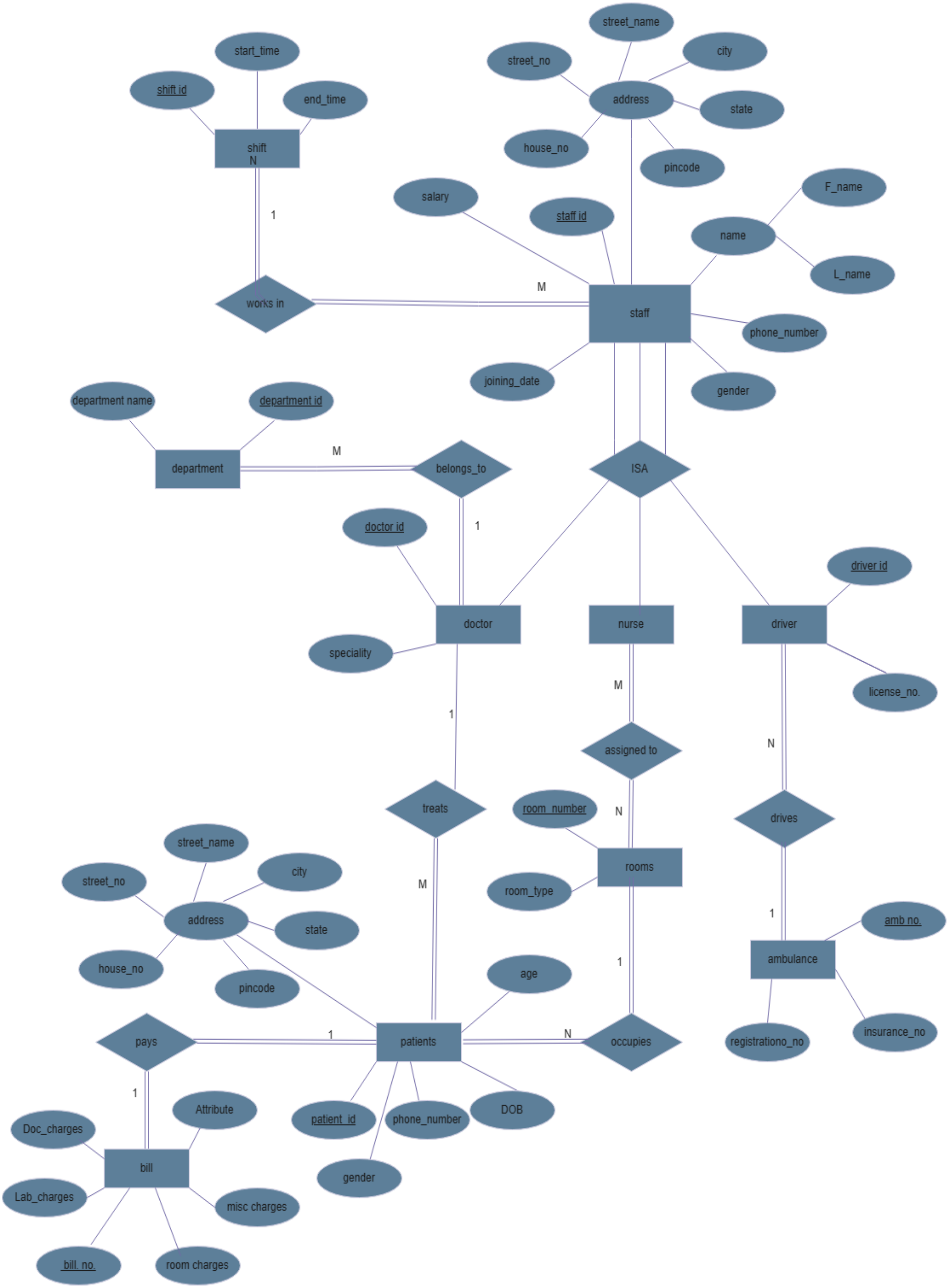
The implementation of the Hospital Management System leverages a combination of python packages and use their power to access database and show them to the user :

- Frontend:
 - python library tkinter / Tk is used.

- Backend:
 - to access database python mysql-connector package is used.

- Database:
 - MySQL for data storage.

ER - diagram


start_time
shift id
end_time
shift
N
1
works in
M
street_name
street_no
city
address
state
house_no
pincode
salary
staff id
F_name
name
L_name
staff
phone_number
joining_date
gender
department name
department id
department
M
belongs_to
ISA
1
doctor id
driver id
doctor
nurse
driver
speciality
license_no.
M
1
N
assigned to
treats
room_number
N
drives
rooms
street_name
street_no
city
M
room_type
address
state
1
amb no.
house_no
pincode
age
1
ambulance
pays
1
patients
N
occupies
registrationo_no
insurance_no
1
Attribute
Doc_charges
patient_id
phone_number
DOB
bill
gender
Lab_charges
misc charges
bill_no.
room charges

Relational Schema

- Staff (staffID, F_name, L_name, Gender, Phone_number, House_number, street_number, street_name, city, state, pincode, joining_date, salary, Shift_ID)

- Department(DepartmentID, DepartmentName)

- Shift (Shift ID, Start_time, End_time)

- Doctor(Doctor ID, Speciality, DepartmentID, StaffID)

- Patients (Patient ID,, F_name,L_name, Date_of_Birth, Age, Gender, Phone_number, House_number, street_number, street_name, city, state, pincode, Doctor_ID,RoomNumber,)

- Nurse(Staff ID, RoomNumber)

- Driver(Staff_ID, DriverID, License_Number, AmbulanceID, StaffID)

- Rooms(RoomNumber, RoomType)

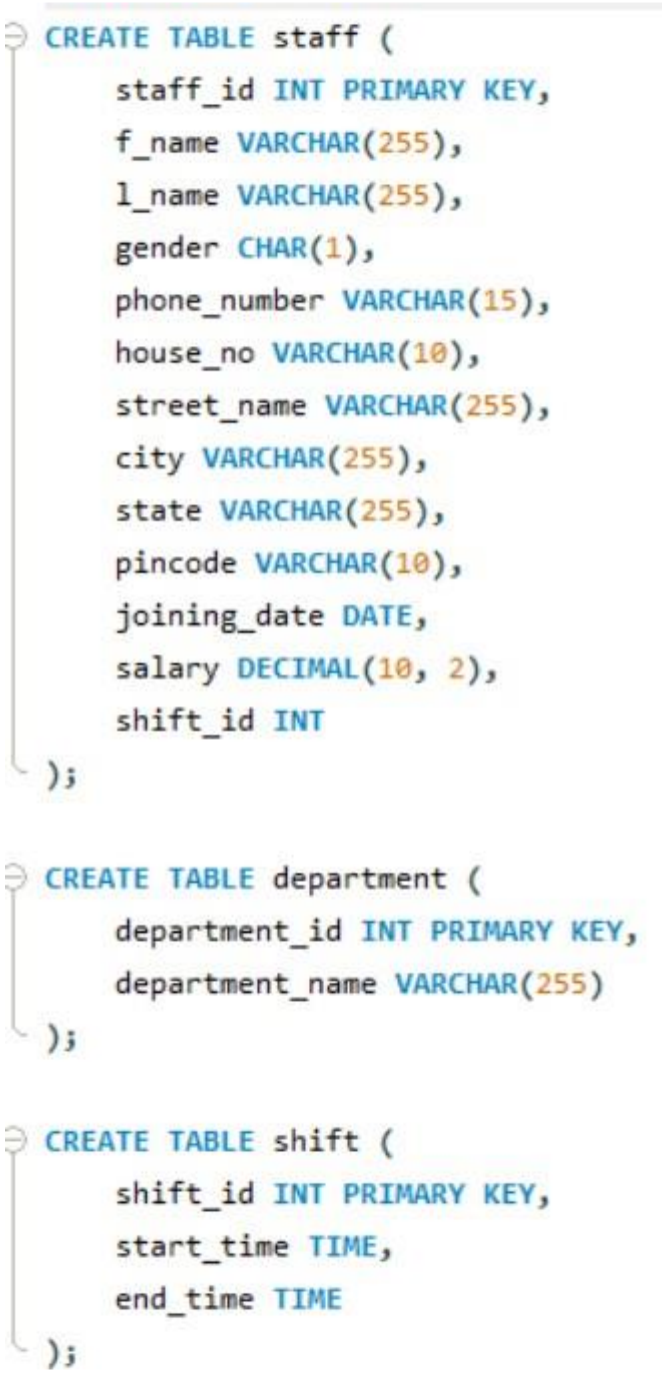
- Ambulance(AmbulanceNumber,Registration Number, License Number)

- Records(Record ID, Date_admitted, Date_discharged, Diagnosis, Patient_ID)

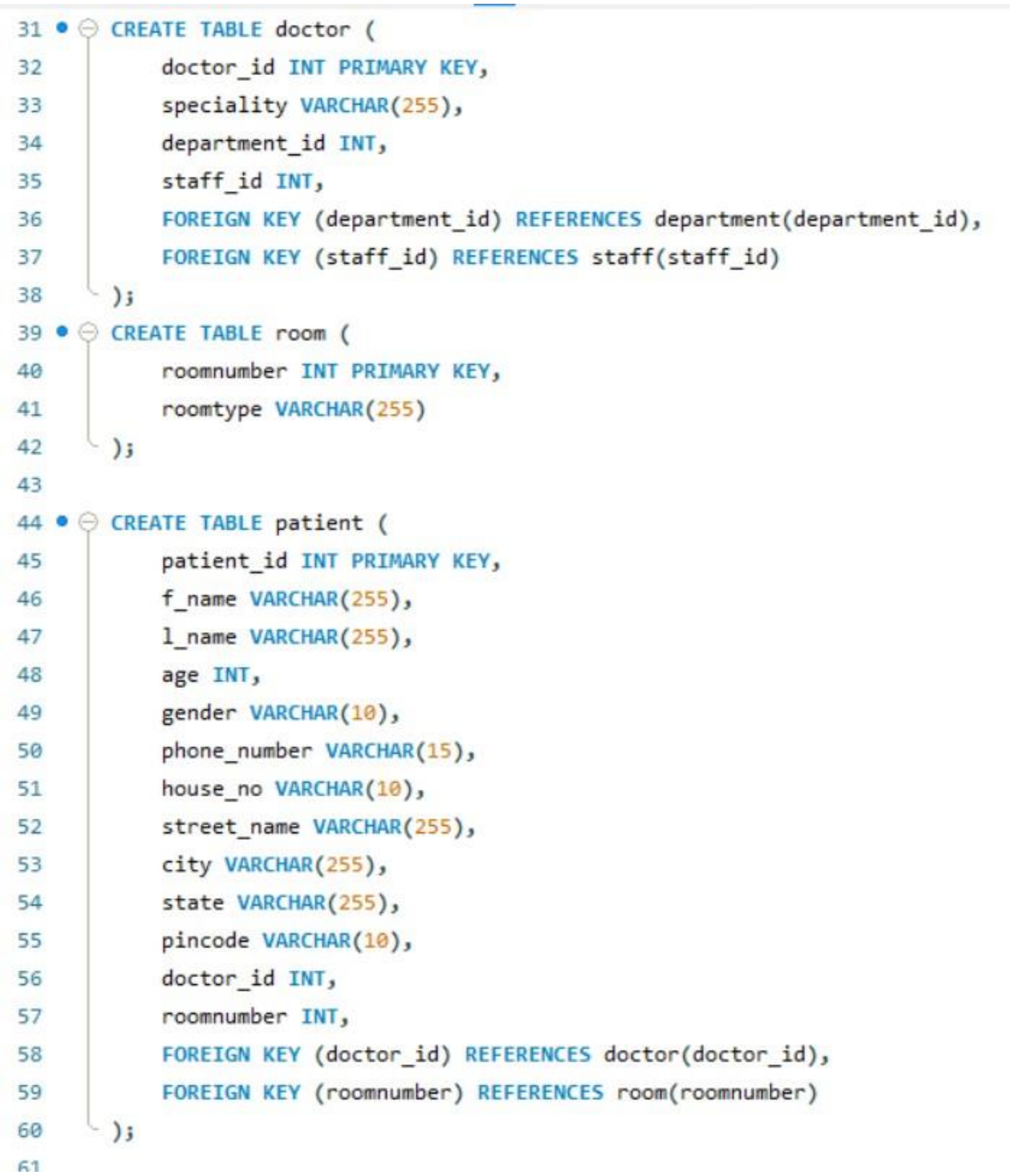
- Bill Bill number, Doc_charges, Lab_charges, Room_charges, TotalAmount, BillingDate, PatientID)

Database Design:

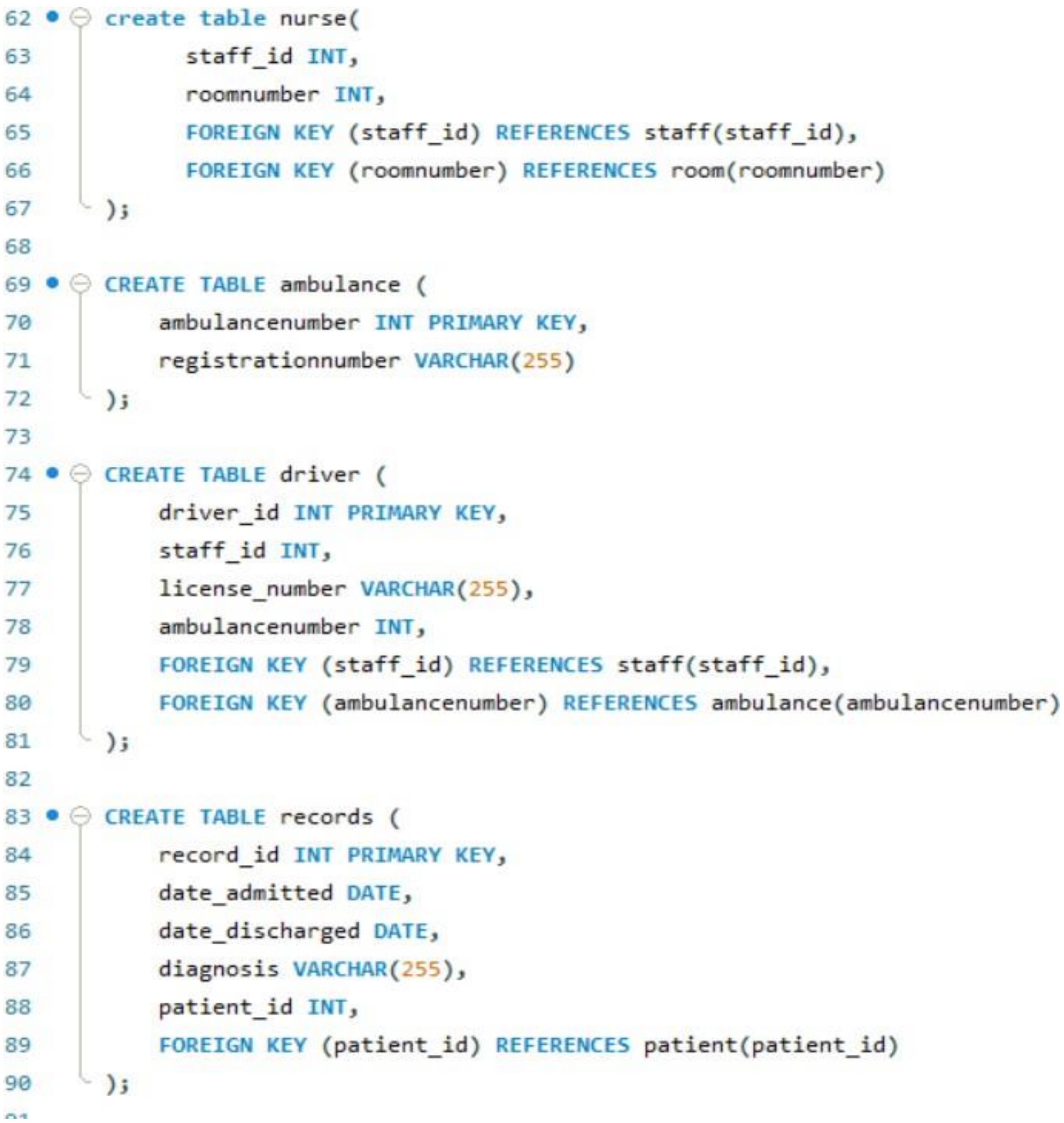
Table and relationships :
Tables are created using following DDL :-



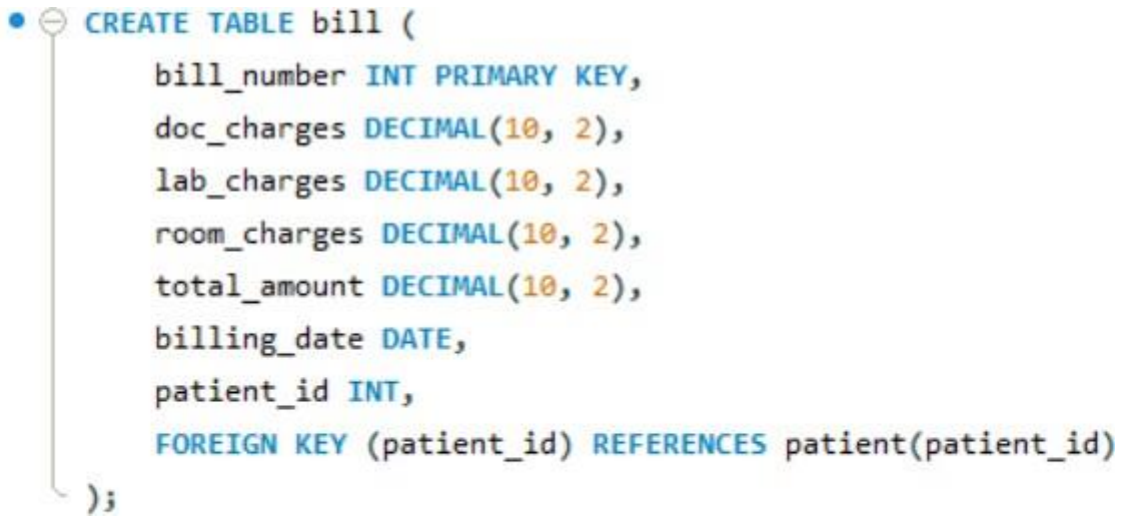
```
CREATE TABLE staff (
    staff_id INT PRIMARY KEY,
    f_name VARCHAR(255),
    l_name VARCHAR(255),
    gender CHAR(1),
    phone_number VARCHAR(15),
    house_no VARCHAR(10),
    street_name VARCHAR(255),
    city VARCHAR(255),
    state VARCHAR(255),
    pincode VARCHAR(10),
    joining_date DATE,
    salary DECIMAL(10, 2),
    shift_id INT
);

CREATE TABLE department (
    department_id INT PRIMARY KEY,
    department_name VARCHAR(255)
);

CREATE TABLE shift (
    shift_id INT PRIMARY KEY,
    start_time TIME,
    end_time TIME
);
```



```sql
CREATE TABLE doctor (
    doctor_id INT PRIMARY KEY,
    speciality VARCHAR(255),
    department_id INT,
    staff_id INT,
    FOREIGN KEY (department_id) REFERENCES department(department_id),
    FOREIGN KEY (staff_id) REFERENCES staff(staff_id)
);
CREATE TABLE room (
    roomnumber INT PRIMARY KEY,
    roomtype VARCHAR(255)
);

CREATE TABLE patient (
    patient_id INT PRIMARY KEY,
    f_name VARCHAR(255),
    l_name VARCHAR(255),
    age INT,
    gender VARCHAR(10),
    phone_number VARCHAR(15),
    house_no VARCHAR(10),
    street_name VARCHAR(255),
    city VARCHAR(255),
    state VARCHAR(255),
    pincode VARCHAR(10),
    doctor_id INT,
    roomnumber INT,
    FOREIGN KEY (doctor_id) REFERENCES doctor(doctor_id),
    FOREIGN KEY (roomnumber) REFERENCES room(roomnumber)
);
```



```sql
create table nurse(
      staff_id INT,
      roomnumber INT,
      FOREIGN KEY (staff_id) REFERENCES staff(staff_id),
      FOREIGN KEY (roomnumber) REFERENCES room(roomnumber)
);

CREATE TABLE ambulance (
    ambulancenumber INT PRIMARY KEY,
    registrationnumber VARCHAR(255)
);

CREATE TABLE driver (
    driver_id INT PRIMARY KEY,
    staff_id INT,
    license_number VARCHAR(255),
    ambulancenumber INT,
    FOREIGN KEY (staff_id) REFERENCES staff(staff_id),
    FOREIGN KEY (ambulancenumber) REFERENCES ambulance(ambulancenumber)
);

CREATE TABLE records (
    record_id INT PRIMARY KEY,
    date_admitted DATE,
    date_discharged DATE,
    diagnosis VARCHAR(255),
    patient_id INT,
    FOREIGN KEY (patient_id) REFERENCES patient(patient_id)
);
```



```sql
CREATE TABLE bill (
    bill_number INT PRIMARY KEY,
    doc_charges DECIMAL(10, 2),
    lab_charges DECIMAL(10, 2),
    room_charges DECIMAL(10, 2),
    total_amount DECIMAL(10, 2),
    billing_date DATE,
    patient_id INT,
    FOREIGN KEY (patient_id) REFERENCES patient(patient_id)
);
```

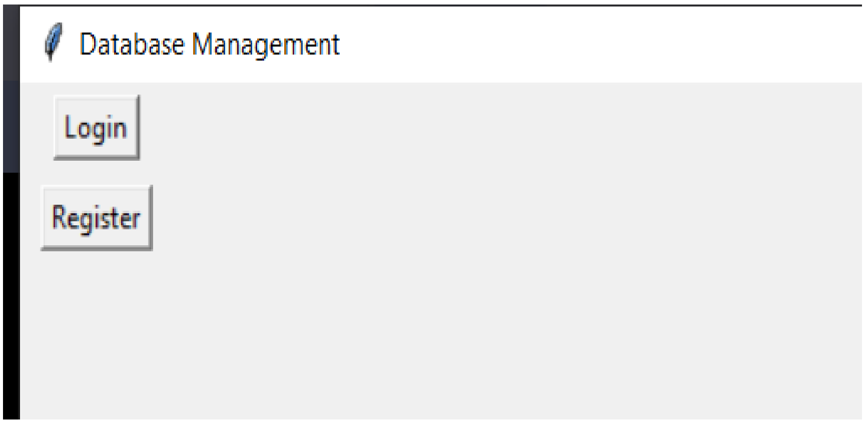
Front End:

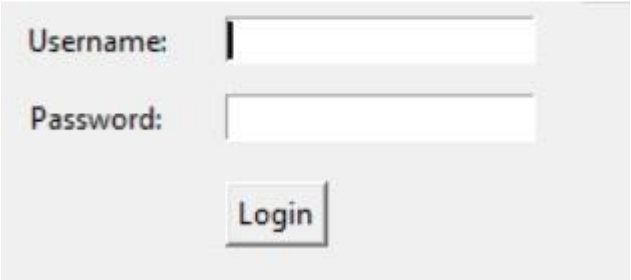
Python is used to connected to database using mysql-connector package using following syntax :

```
import mysql.connector as c

con=c.connect(host="localhost",user="root",password="kart@mysql19",database="hospital")

cur=con.cursor()
```

The cursor created is used to access data from database and to transfer query to the database.

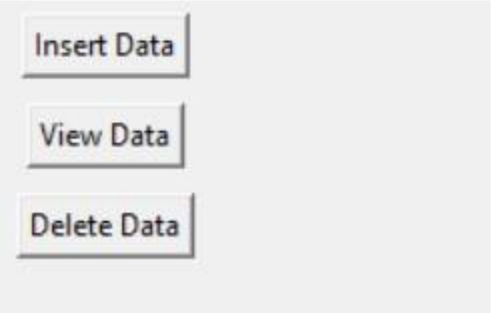
The first page is a page for Login or Register, the user can login and can then continue :





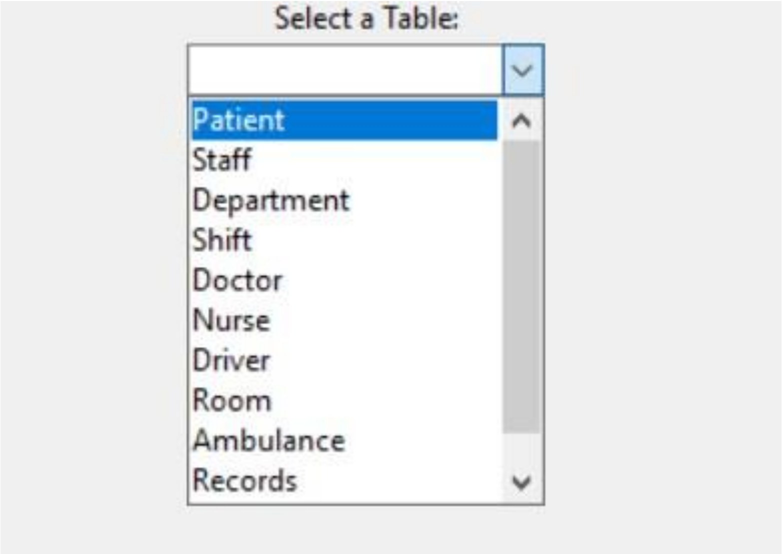
User login

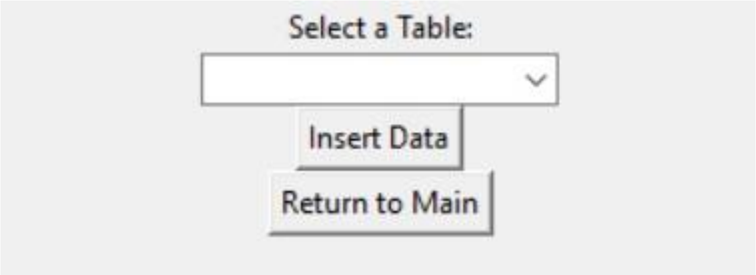
After logging in user can insert, delete or view database,



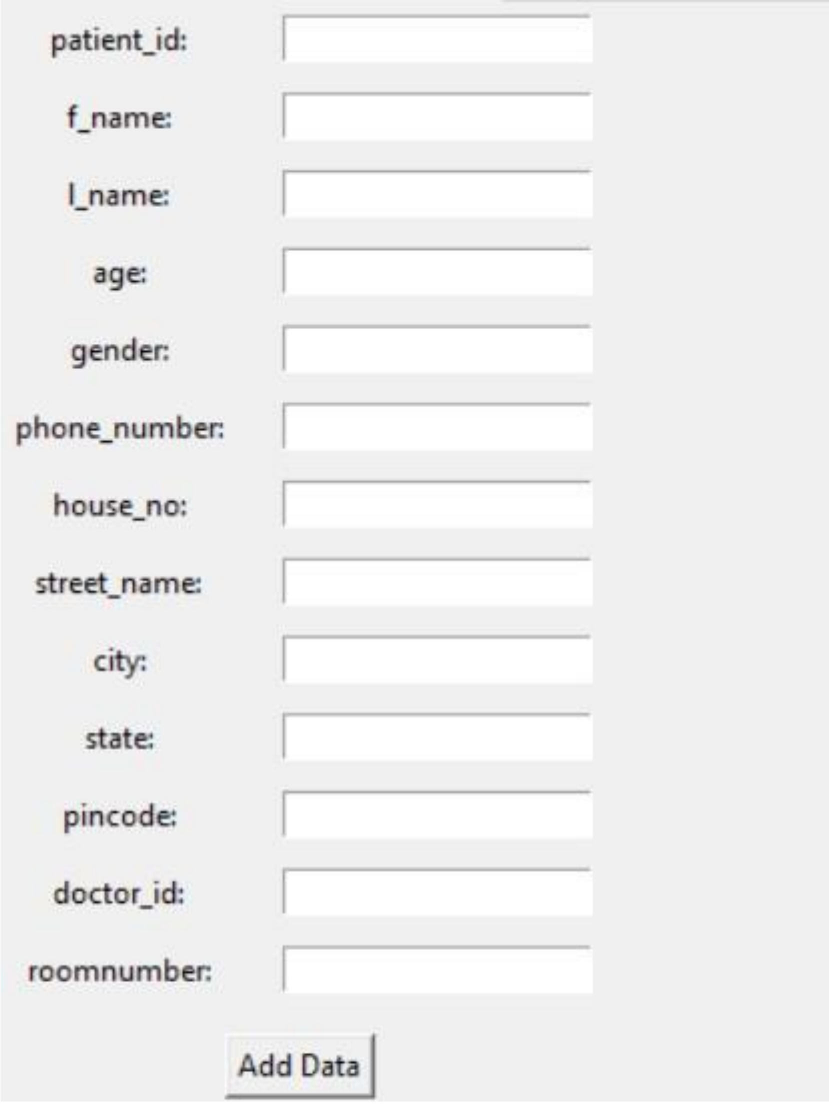
Inserting in database :

First user has to select in which table data is to be inserted :

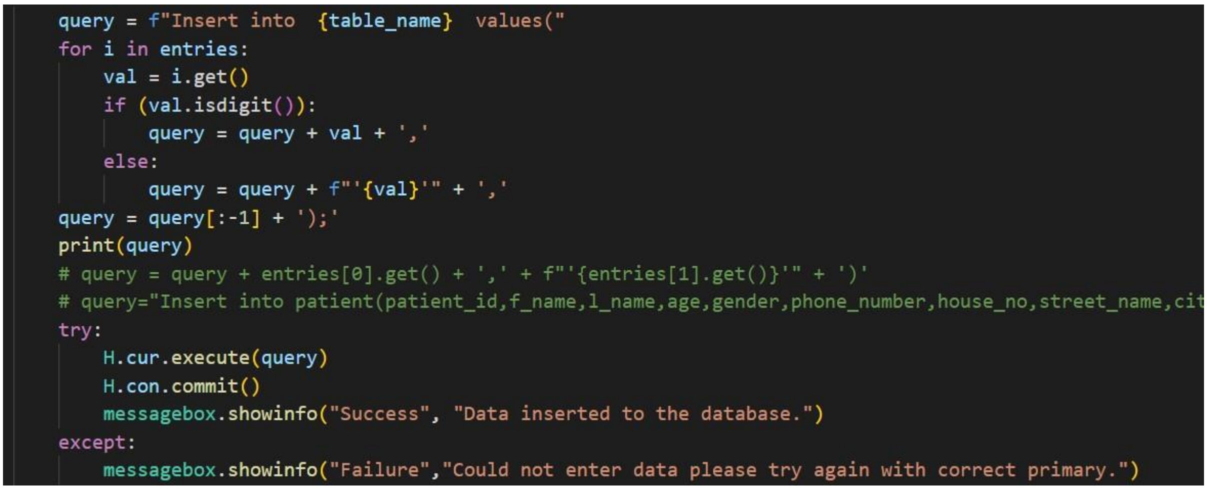




- Then user has to fill the form to insert data



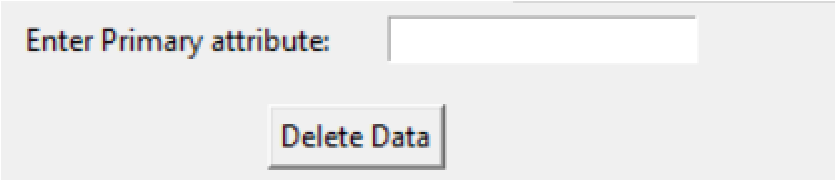
- On clicking add data following python script inserts data into database by firing the insert query :



```python
query = f"Insert into  {table_name}  values("
for i in entries:
    val = i.get()
    if (val.isdigit()):
        query = query + val + ','
    else:
        query = query + f"'{val}'" + ','
query = query[:-1] + ');'
print(query)
# query = query + entries[0].get() + ',' + f"'{entries[1].get()}'" + ')'
# query="Insert into patient(patient_id,f_name,l_name,age,gender,phone_number,house_no,street_name,cit
try:
    H.cur.execute(query)
    H.con.commit()
    messagebox.showinfo("Success", "Data inserted to the database.")
except:
    messagebox.showinfo("Failure","Could not enter data please try again with correct primary.")
```

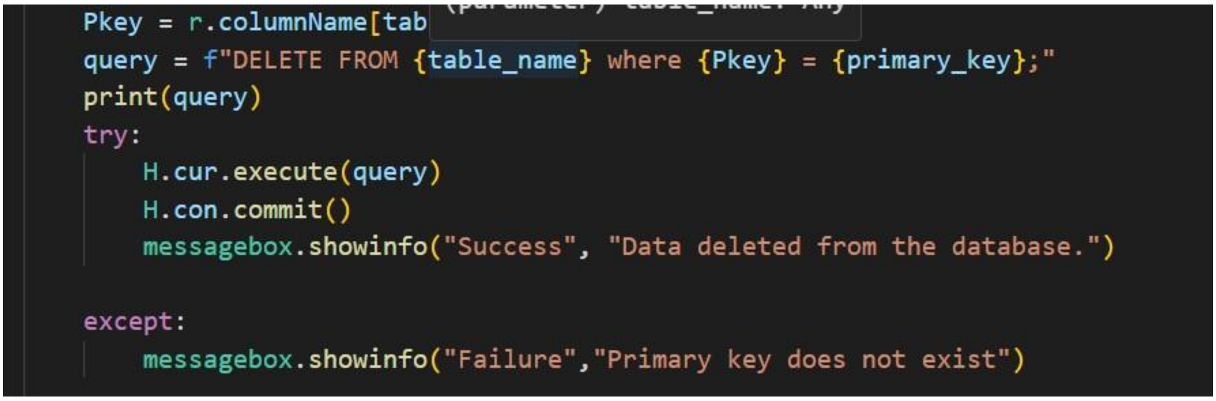
Deleting from database :

Similar to inserting user has to select the table, and then enter the primary key of the table from which data has to be deleted :



Enter Primary attribute:

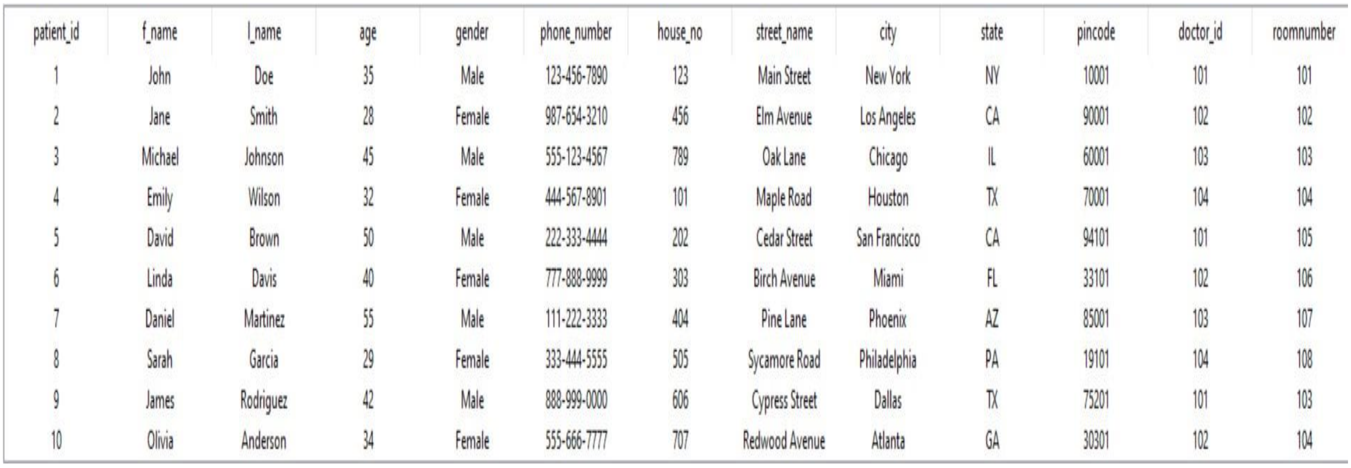
On clicking delete data, following python script deleted data from database,



```
Pkey = r.columnName[tab
query = f"DELETE FROM {table_name} where {Pkey} = {primary_key};"
print(query)
try:
    H.cur.execute(query)
    H.con.commit()
    messagebox.showinfo("Success", "Data deleted from the database.")

except:
    messagebox.showinfo("Failure","Primary key does not exist")
```

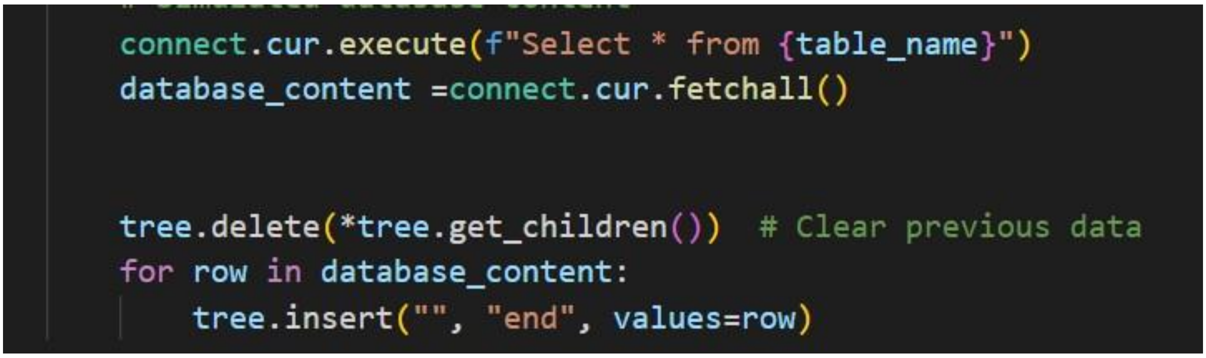
Viewing the database :

The user can view tables by going in the view menu and then selecting the table to be viewed ;



patient_id	f_name	l_name	age	gender	phone_number	house_no	street_name	city	state	pincode	doctor_id	roomnumber
1	John	Doe	35	Male	123-456-7890	123	Main Street	New York	NY	10001	101	101
2	Jane	Smith	28	Female	987-654-3210	456	Elm Avenue	Los Angeles	CA	90001	102	102
3	Michael	Johnson	45	Male	555-123-4567	789	Oak Lane	Chicago	IL	60001	103	103
4	Emily	Wilson	32	Female	444-567-8901	101	Maple Road	Houston	TX	70001	104	104
5	David	Brown	50	Male	222-333-4444	202	Cedar Street	San Francisco	CA	94101	101	105
6	Linda	Davis	40	Female	777-888-9999	303	Birch Avenue	Miami	FL	33101	102	106
7	Daniel	Martinez	55	Male	111-222-3333	404	Pine Lane	Phoenix	AZ	85001	103	107
8	Sarah	Garcia	29	Female	333-444-5555	505	Sycamore Road	Philadelphia	PA	19101	104	108
9	James	Rodriguez	42	Male	888-999-0000	606	Cypress Street	Dallas	TX	75201	101	103
10	Olivia	Anderson	34	Female	555-666-7777	707	Redwood Avenue	Atlanta	GA	30301	102	104

The query running in background to fetch data,



```
connect.cur.execute(f"Select * from {table_name}")
database_content =connect.cur.fetchall()


tree.delete(*tree.get_children())  # Clear previous data
for row in database_content:
    tree.insert("", "end", values=row)
```

Future Scope of project

The future scope of a Hospital Management System (HMS) project involves potential improvements, expansions, and additional features that can be implemented to enhance the system's capabilities. Here are some possible future enhancements for an HMS:

1. Scaling the project where patient and other staff too could login and access their records.

2. Develop a mobile application for both patients and healthcare providers, enabling easy access to appointments, medical records, and other relevant information.

3. Enhance billing processes with automated invoicing, online payment options, and improved insurance management for a smoother financial workflow.

4. Implement features for better staff communication, training modules, and performance analytics to optimize workforce management.

5. Regularly update and strengthen the system's security measures to protect sensitive patient information from cybersecurity threats.