Project-Report

PL/SQL

Submitted to

Mr. RAVI RAJ (Mentor)

Submitted by

Name: Pratyush Kashyap
UID: 25MCA20228
Section/Group: MCA-4-B

MASTER OF COMPUTER APPLICATION

Hospital Management System

Acknowledgment

I would like to express my sincere gratitude to my faculty and guide for their valuable guidance, motivation, and support throughout the completion of this project. Their encouragement and insightful feedback helped me in understanding the concepts of database management and software development more effectively. I am also thankful to my friends and classmates for their cooperation and assistance during the preparation and testing phases of this project.

Abstract

The *Hospital Management System (HMS)* is a desktop-based software application designed to manage hospital operations efficiently. It helps medical staff, administrators, and receptionists manage patient records, doctor information, billing, and appointments within a single platform.

This project is developed using **Python (Tkinter for GUI)** and **MySQL** as the backend database. It simplifies the day-to-day administrative tasks of a hospital by automating operations like adding patient details, viewing doctor lists, generating bills, and updating or deleting records.

The system provides a user-friendly graphical interface that does not require users to have advanced technical knowledge. The HMS ensures data accuracy, easy access, and secure record maintenance. It not only saves time but also enhances hospital efficiency, providing better healthcare service delivery.

Introduction

Hospitals handle a vast amount of data daily, from patient admissions to doctor schedules and billing.

Manual record-keeping often leads to inefficiency, data loss, and human error.

The *Hospital Management System* aims to solve these issues by digitizing hospital processes.

The system allows hospital staff to register new patients, assign doctors, track treatments, and manage billing information easily. By integrating all this data into one central system, HMS ensures better coordination among departments. The main goal is to enhance hospital productivity, ensure accurate record maintenance, and improve the overall patient experience.

Aim / Objectives

The main aim of the *Hospital Management System* is to automate and streamline hospital operations for better management and faster service.

Objectives:

1. To maintain accurate and updated records of patients and doctors.
2. To reduce manual errors and paperwork through an automated digital system.
3. To manage billing and payments efficiently.
4. To design an easy-to-use graphical interface for hospital staff.
5. To ensure data security and quick retrieval of patient records when required.

Technologies Used

1. **Frontend (GUI):** Python Tkinter
2. **Backend (Database):** MySQL
3. **Connector Library:** mysql-connector-python
4. **Development Tool:** Visual Studio Code
5. **Operating System:** Windows 10 / 11

Algorithm / Flow of the Project

1. **Start the Application**

2. **Home Interface Opens** – provides options for Add, View, Update, Delete records.

3. **Add Patient:**
   - Input details → Name, Age, Gender, Disease, Doctor Assigned.
   - Data stored in the Patients table.

4. **View Records:**
   - Display patient and doctor details using MySQL queries.

5. **Update / Delete:**
   - Modify or remove patient details if required.

6. **Generate Bill:**
   - Fetch patient ID → calculate total → display and store payment status.

7. **Exit the Application**

Flowchart (Textual Representation):

```
Start
  ↓
Open GUI
  ↓
Select Operation → [Add | View | Update | Delete | Bill]
  ↓
Perform MySQL Query (INSERT / SELECT / UPDATE / DELETE)
  ↓
Display Result in Tkinter Window
  ↓
End
```

Dataset Used

The project uses MySQL tables for data storage:

Table 1: Doctors

Column Name	Data Type	Description
Doctor_ID	INT (PK)	Unique ID for doctor
Doctor_Name	VARCHAR	Name of doctor
Specialization	VARCHAR	Area of expertise
Contact_Number	VARCHAR	Doctor's contact info

Table 2: Patients

Column Name	Data Type	Description
Patient_ID	INT (PK)	Unique patient ID
Patient_Name	VARCHAR	Name of patient
Age	INT	Patient age
Gender	VARCHAR	Gender
Disease	VARCHAR	Diagnosed disease
Doctor_ID	INT (FK)	Assigned doctor

Table 3: Billing

Column Name	Data Type	Description
Bill_ID	INT (PK)	Bill number
Patient_ID	INT (FK)	Related patient
Amount	DECIMAL	Total bill amount
Payment_Status	VARCHAR	Paid / Unpaid

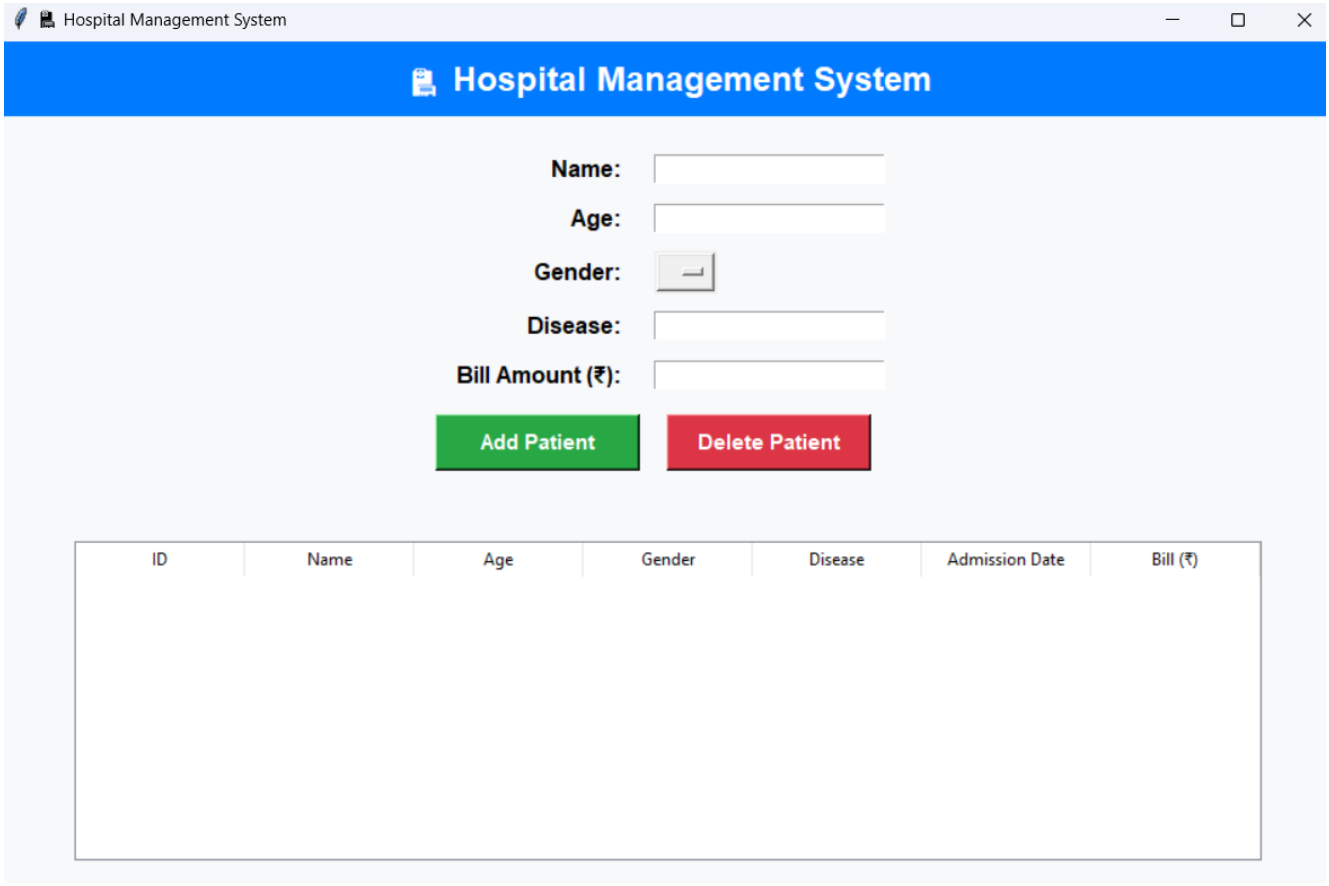
Code for Experiment:

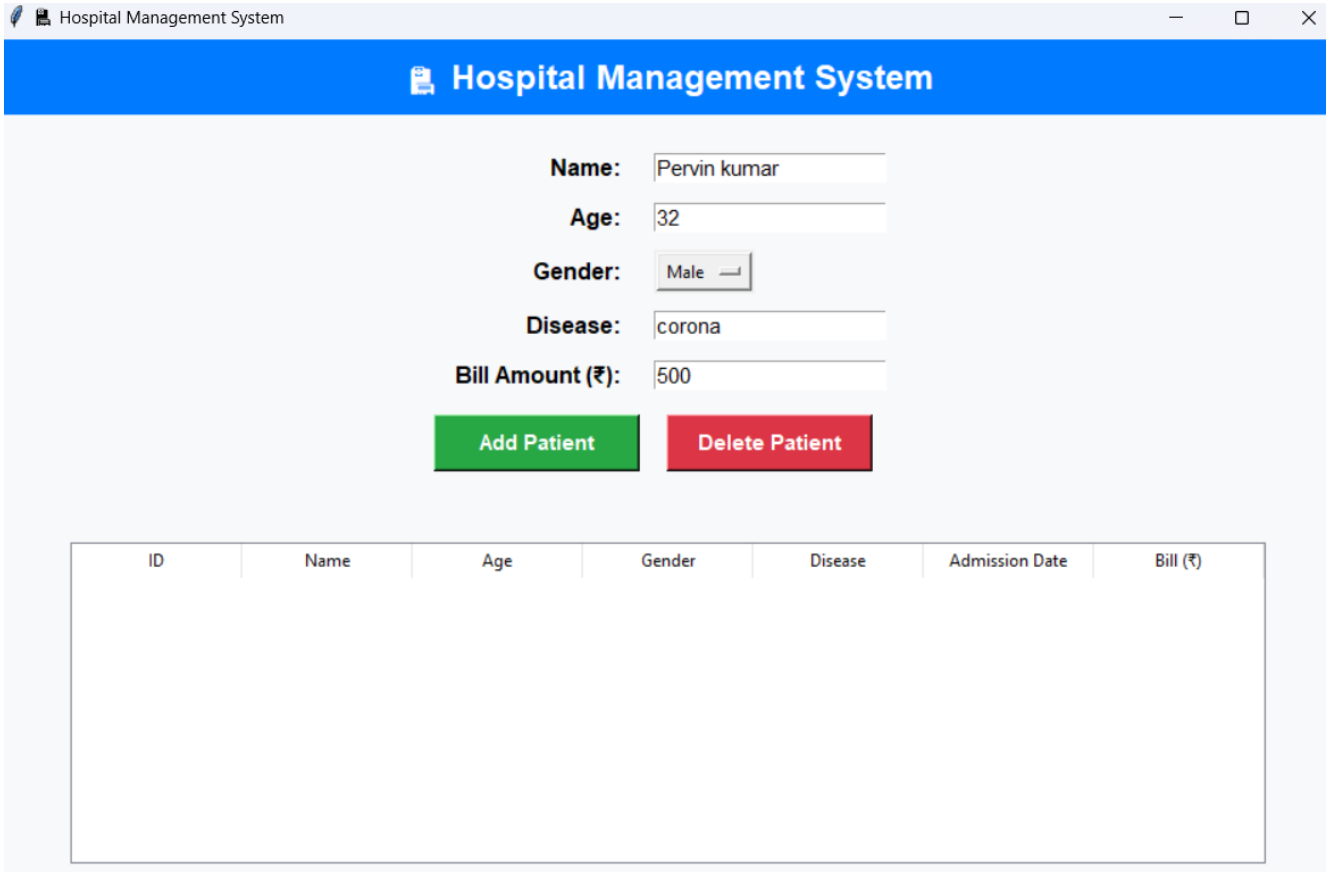
```
import tkinter as tk
from tkinter import messagebox
import mysql.connector

# Database connection
con = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="hospital_db"
)
cursor = con.cursor()

# Create tables if not exist
cursor.execute("""
CREATE TABLE IF NOT EXISTS Doctors (
    Doctor_ID INT PRIMARY KEY,
    Doctor_Name VARCHAR(50),
    Specialization VARCHAR(50),
    Contact_Number VARCHAR(15)
)
""")
cursor.execute("""
CREATE TABLE IF NOT EXISTS Patients (
    Patient_ID INT PRIMARY KEY,
    Patient_Name VARCHAR(50),
    Age INT,
    Gender VARCHAR(10),
    Disease VARCHAR(50),
    Doctor_ID INT,
    FOREIGN KEY (Doctor_ID) REFERENCES Doctors(Doctor_ID)
)
""")
cursor.execute("""
CREATE TABLE IF NOT EXISTS Billing (
```

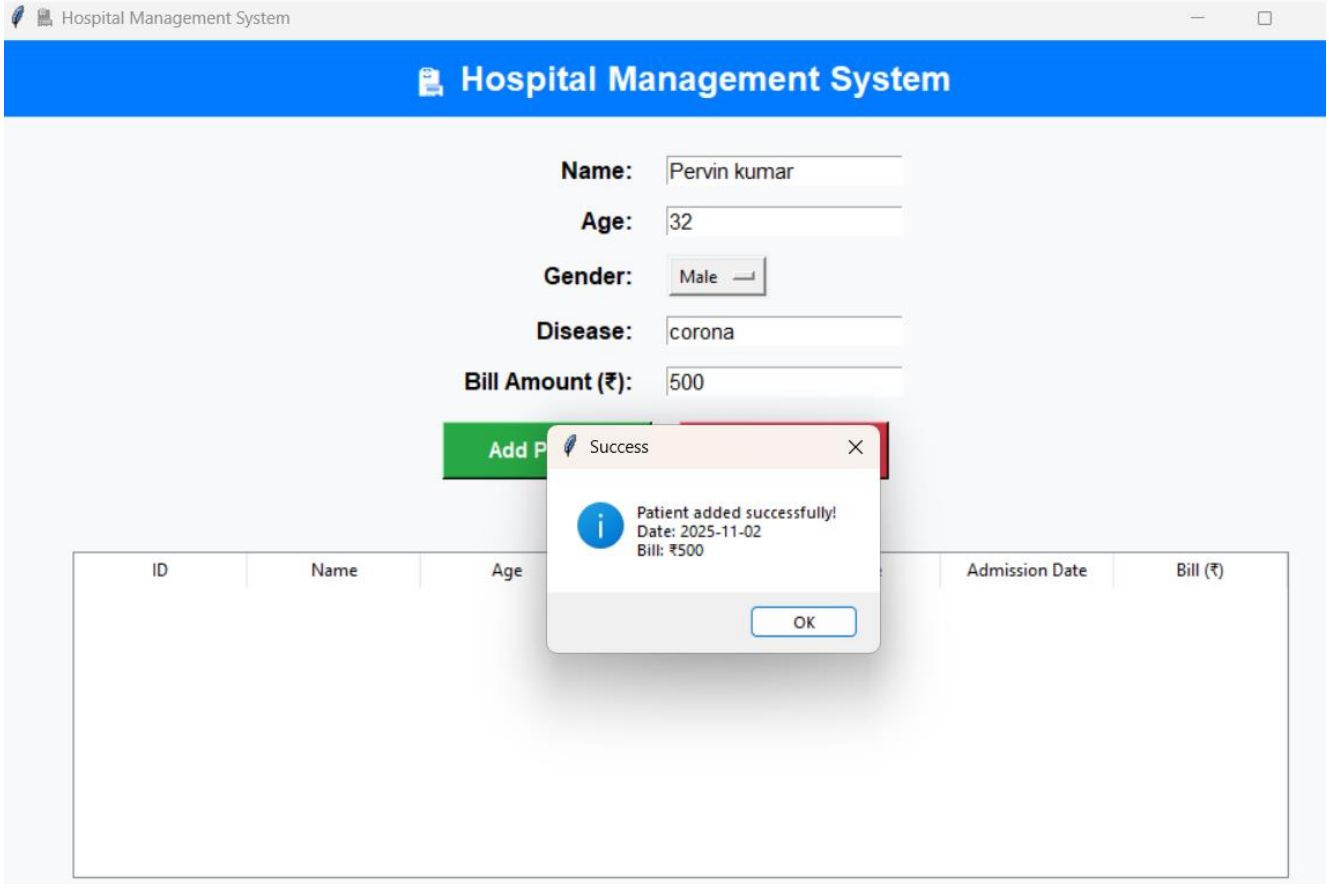
```
Bill_ID INT PRIMARY KEY,
    Patient_ID INT,
    Amount DECIMAL(10,2),
    Payment_Status VARCHAR(20),
    FOREIGN KEY (Patient_ID) REFERENCES Patients(Patient_ID)
)
""")

# GUI setup
root = tk.Tk()
root.title("Hospital Management System")
root.geometry("600x500")
root.config(bg="#e8f4f8")

# Labels & Entries
tk.Label(root, text="Patient ID").grid(row=0, column=0)
pid = tk.Entry(root); pid.grid(row=0, column=1)
tk.Label(root, text="Patient Name").grid(row=1, column=0)
pname = tk.Entry(root); pname.grid(row=1, column=1)
tk.Label(root, text="Age").grid(row=2, column=0)
page = tk.Entry(root); page.grid(row=2, column=1)
tk.Label(root, text="Gender").grid(row=3, column=0)
pgender = tk.Entry(root); pgender.grid(row=3, column=1)
tk.Label(root, text="Disease").grid(row=4, column=0)
pdisease = tk.Entry(root); pdisease.grid(row=4, column=1)
tk.Label(root, text="Doctor ID").grid(row=5, column=0)
pdoc = tk.Entry(root); pdoc.grid(row=5, column=1)

# Functions
def add_patient():
    cursor.execute("INSERT INTO Patients VALUES
(%s,%s,%s,%s,%s,%s)",
                   (pid.get(), pname.get(), page.get(),
pgender.get(), pdisease.get(), pdoc.get()))
    con.commit()
    messagebox.showinfo("Success", "Patient added successfully")

def view_patients():
    cursor.execute("SELECT * FROM Patients")
    records = cursor.fetchall()
    output = ""
    for r in records:
        output += str(r) + "\n"
    messagebox.showinfo("All Patients", output)

# Buttons
btn_style = {"width": 18, "height": 1}
tk.Button(root, text="Add Patient", command=add_patient,
bg="#4CAF50", fg="white", **btn_style).grid(row=6, column=0, pady=5)
tk.Button(root, text="View Patients", command=view_patients,
```

```
bg="#2196F3", fg="white", **btn_style).grid(row=6, column=1, pady=5)
tk.Button(root, text="Exit", command=root.quit, bg="#f44336",
fg="white", **btn_style).grid(row=7, column=0, columnspan=2,
pady=10)

root.mainloop()
```

Output:




Hospital Management System
Hospital Management System
Name:
Pervin kumar
Age:
32
Gender:
Male
Disease:
corona
Bill Amount (₹):
500
Add Patient
Delete Patient
ID
Name
Age
Gender
Disease
Admission Date
Bill (₹)


Hospital Management System
Hospital Management System
Name:
Pervin kumar
Age:
32
Gender:
Male
Disease:
corona
Bill Amount (₹):
500
Success
Patient added successfully!
Date: 2025-11-02
Bill: ₹500
OK
ID
Name
Admission Date
Bill (₹)

Conclusion

The *Hospital Management System* provides an efficient and user-friendly way to manage hospital operations digitally. By integrating Python with MySQL, the project achieves reliable data handling and smooth GUI interaction. The system reduces manual effort, saves time, and enhances accuracy in record maintenance. It can be expanded further to include modules for appointment scheduling, doctor login, and advanced billing.

Learning Outcomes

From this project, I have learned:

1. How to connect Python applications with a MySQL database.
2. The use of Tkinter for creating attractive and functional GUIs.
3. How CRUD operations are performed in real-world applications.
4. The importance of database normalization and system design in software projects.

Reference: https://github.com/kashyappratyush69-oss/Hospital-management-system.git