# MRA DAV Public School

**(BY PASS ROAD, SOLAN)**

# HIMACHAL PRADESH


## INFORMATICS PRACTICES

## Class-XII

# PROJECT REPORT

PROJECT BY:
NAME: MANN UPADHYAY
ROLL NO: 25

# CERTIFICATE

This is to certify that this Project Report on Hospital Management System using Python and MySQL is submitted by Mann Upadhyay (XII-Alpha) (R.No.25) to the Computer Department of MRA DAV PUBLIC SCHOOL, SOLAN, Himachal Pradesh, carried out by her/him towards partial completion of Practical Exam for class XII during academic year 2023-2024.

**Internal Examiner** **External Examiner** **H.O.D** **PRINCIPAL**

# AKNOWLEDMENT

I would want to convey my gratitude to everyone who has assisted me in finishing my assignment successfully.

First and first, I want to thank CBSE from the bottom of my heart for giving me such a wonderful opportunity to develop a project and learn more about this fascinating subject.

Second, I want to express my gratitude to our school's principal, Ms. Masooma Singha, for providing us with the inspiration and amazing support we needed to finish the project.

Thirdly, I would like to thank Mr. Rajesh, my IP teacher, who guided me through every step of the Project Report preparation.

Finally, I'd want to thank everyone who has helped me along the way, including my teachers, parents, and whose support has made this effort possible

# STUDENT PROFILE

**Name:** : Mann Upadhyay

**Date of Birth:** : 28 August 2006

**Father Name:** : Mr. Rohit

**Mother Name:** : Mrs. Chandan Dhar

**Class:** : $12^{th}$

**Section:** : Alpha

**Roll No:** : 25

**Address:** : C/O Sh. Deepak Thakur, Near Rajeshwari Temple, Kandaghat, Distt- Solan, Himachal Pradesh, 173215

**Hobbies:** : Drawing Sketches, Coding and Listening Music

# PROJECT REPORT ON HOSPITAL MANAGEMENT SYSTEM USING PYTHON AND MYSQL

## Table of Contents


| S.No | Topic | Page No |
|---|---|---|
| 1 | General Introduction of the Project | 6 |
| 2 | Hardware and Software Requirements | 7 |
| 3 | Python Code for the Project | 8-21 |
| 4 | MySQL used for the Project | 22-28 |
| 5 | Output of the Project | 29-36 |
| 6 | Further scopes of the Project | 37 |
| 7 | Bibliography | 38 |

# GENERAL INTRODUCTION OF THE PROJECT

Hospital Management project is developed using python and Mysql. In this project there are three categories, that can use this project:

- **Admin** : Can create new tables, updated records, enter records, browse the data, search using wildcard.
- **Doctor**: Can browse records, update records
- **Patient**: Can buy the medicine, check his/her disease history, browse his/her detail.

# INTRODUCTION TO PYTHON & MYSQL

# HARDWARE REQUIREMENT
# SOFTWARE REQUIREMENT

Python is a programming language that lets you work more quickly and integrate your systems more effectively.

Pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.
Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.
MySQL is the world's most popular open source database

This project is developed on following hardware and software requirements:
A) Hardware Requirements:

- Processor: Intel(R) Core(TM) i3 CPU @2,40GHz
- Installed RAM: 4.00 GB
- System type: 64-bit operating system
- Operating System: MS Windows 10

B) Software Requirements:

- PyCharm
- MySQL workbench and phpmyadmin

# PYTHON CODE FOR PROJECT


**Figure 1: PyCharm tool is used for developing the application**

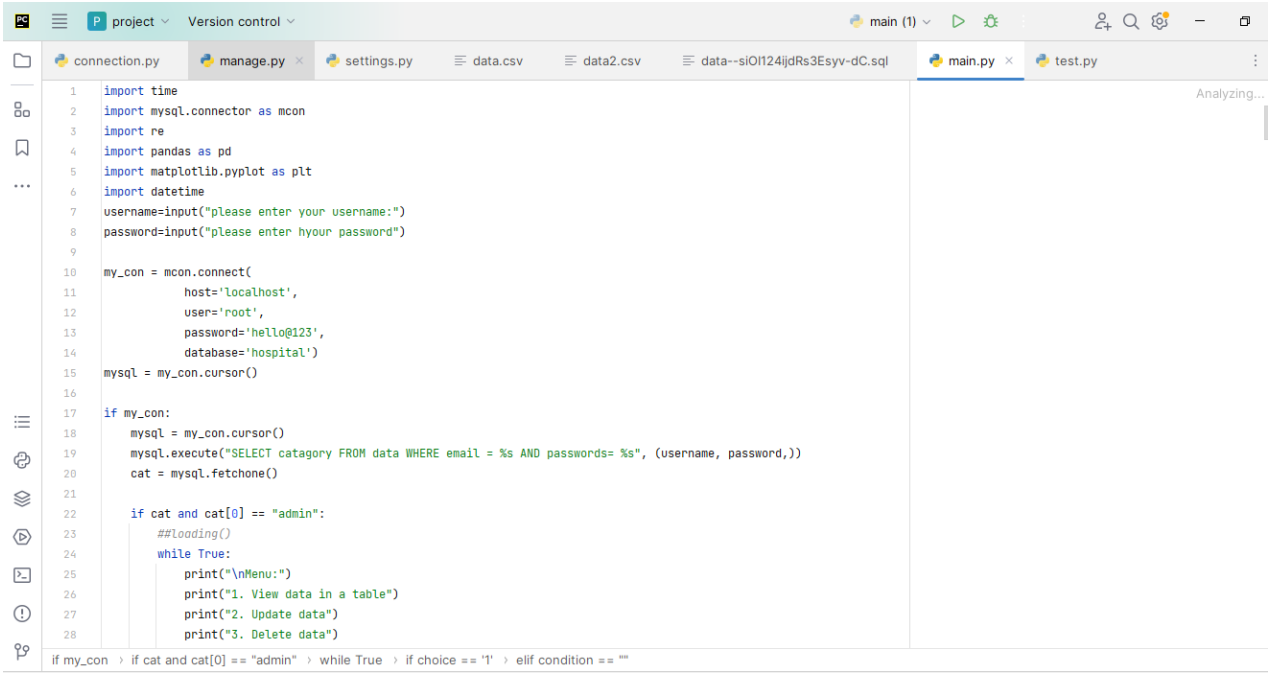



**Figure 2: Code in python in PyCharm**

# Python Code:

```
import time
import mysql.connector as mcon
import re
import pandas as pd
import matplotlib.pyplot as plt
import datetime
username=input("please enter your username:")
password=input("please enter hyour password")

my_con = mcon.connect(
            host='localhost',
            user='root',
            password='',
            database='hospital')
mysql = my_con.cursor()

if my_con:
    mysql = my_con.cursor()
    mysql.execute("SELECT catagory FROM data WHERE email = %s AND
passwords= %s", (username, password,))
    cat = mysql.fetchone()

    if cat and cat[0] == "admin":
        ##loading()
        while True:
            print("\nMenu:")
            print("1. View data in a table")
            print("2. Update data")
            print("3. Delete data")
            print("4. Create a new table")
            print("5. Delete a table")
            print("6. Alter a table")
            print("7. Search for wildcard character")
            print("8. Graphs")
            print("9. Exit")

            choice = input("Enter your choice: ")

            if choice == '1':
                mysql.execute("SHOW TABLES")
                tables = mysql.fetchall()

                print("Available Tables:")
                for i, table in enumerate(tables, start=1):
                    print(f"{i}. {table[0]}")


                table_choice = int(input("Select a table (enter
the number): "))
```

```
                if 1 <= table_choice <= len(tables):
                    tbname = tables[table_choice - 1][0]
                else:
                    print("Invalid choice. Please select a valid
table.")

                mysql.execute(f"SHOW COLUMNS FROM {tbname}")
                columns = mysql.fetchall()

                print(f"Columns in {tbname}:")
                for i, column in enumerate(columns, start=1):
                    print(f"{i}. {column[0]}")
                clminp = input("Enter the names of the columns
you want (comma-separated): ")
                if clminp=="":
                    clminp="*"
                condition = input("Enter the condition (e.g.,
'column_name = value'): ")
                order_direction = input("Enter 'ASC' for ascend-
ing or 'DESC' for descending: ")
                order_direction = order_direction.upper()
                ordcl= columns[0]
                ordcl=ordcl[0]

                if order_direction not in ["ASC", "DESC"]:
                    order_direction = "ASC"

                if condition == "" and clminp == "":
                    sql_query = "SELECT * FROM " + tbname + " OR-
DER BY "+ordcl +" "+ order_direction + ";"
                elif condition == "":
                    sql_query = "SELECT " + clminp + " FROM " +
tbname + " ORDER BY "+ordcl +" "+ order_direction + ";"
                elif clminp == "":
                    sql_query = "SELECT * FROM " + tbname + "
WHERE " + condition + " ORDER BY "+ordcl +" "+ order_direction +
";"
                else:
                    sql_query = "SELECT " + clminp + " FROM " +
tbname + " WHERE " + condition + " ORDER BY "+ordcl +" "+ or-
der_direction + ";"

                print(sql_query)

                try:
                    mysql.execute(sql_query)
                    result = mysql.fetchall()

                    if result:
                        print("Selected Data:")
                        for row in result:
                            print(row)
```

```
                    else:
                        print("No data matching the condition.")
                except mcon.Error as err:
                    print(f"Error executing SELECT query: {err}")



            elif choice == '2':
                mysql.execute("SHOW TABLES")
                tables = mysql.fetchall()

                print("Available Tables:")
                for i, table in enumerate(tables, start=1):
                    print(f"{i}. {table[0]}")

                table_choice = int(input("Select a table (enter
the number): "))
                if 1 <= table_choice <= len(tables):
                    tbname = tables[table_choice - 1][0]
                else:
                    print("Invalid choice. Please select a valid
table.")
                condition = input("Enter the condition (e.g.,
'column_name = value'): ")
                new_data = input("Enter the new data (e.g., 'col-
umn_name = new_value'): ")

                # Construct the SQL query
                sql_query = "UPDATE "+ tbname +" SET "+ new_data
+" WHERE "+ condition +";"

                try:
                    mysql.execute(sql_query)
                    mysql.connection.commit()
                    print("Data updated successfully.")

                except mcon.Error as err:
                    print(f"Error updating data: {err}")

            elif choice == '3':
                ##loading()
                mysql.execute("SHOW TABLES")
                tables = mysql.fetchall()

                print("Available Tables:")
                for i, table in enumerate(tables, start=1):
                    print(f"{i}. {table[0]}")

                table_choice = int(input("Select a table (enter
the number): "))
                if 1 <= table_choice <= len(tables):
                    tbname = tables[table_choice - 1][0]
```

```
                else:
                    print("Invalid choice. Please select a valid
table.")
                condition = input("Enter the condition (e.g.,
'column_name = value'): ")
                sql_query = "DELETE FROM "+tbname+" WHERE
"+condition+";"

                try:
                    mysql.execute(sql_query)
                    mysql.connection.commit()
                    print("Data deleted successfully.")

                except mcon.Error as err:
                    print(f"Error deleting data: {err}")

            elif choice == '4':

                tbname = input("Enter the name of the new table:
")
                num_columns = int(input("Enter the number of col-
umns: "))

                columns = []
                for i in range(num_columns):
                    column_name = input(f"Enter name for column
{i + 1}: ")
                    column_type = input(f"Enter data type for
column {i + 1}: ")
                    columns.append(f"{column_name} {col-
umn_type}")

                # Construct the SQL query
                sql_query = f"CREATE TABLE {tbname} ({',
'.join(columns)});"

                try:
                    mysql.execute(sql_query)
                    print(f"Table '{tbname}' created success-
fully.")

                except mcon.Error as err:
                    print(f"Error creating table: {err}")
            elif choice == '5':
                table_name = input("Enter the name of the table
to delete: ")

                # Construct the SQL query
                sql_query = f"DROP TABLE {table_name};"

                try:
                    mysql.execute(sql_query)
```

```
                    print(f"Table '{table_name}' deleted success-
fully.")

                except mcon.Error as err:
                    print(f"Error deleting table: {err}")

            elif choice == '6':
                mysql.execute("SHOW TABLES")
                tables = mysql.fetchall()

                print("Available Tables:")
                for i, table in enumerate(tables, start=1):
                    print(f"{i}. {table[0]}")

                table_choice = int(input("Select a table (enter
the number): "))
                if 1 <= table_choice <= len(tables):
                    tbname = tables[table_choice - 1][0]
                else:
                    print("Invalid choice. Please select a valid
table.")

                print("Available options for altering the ta-
ble:")
                print("1. edit column")
                print("2. Delete column")
                option = input("Enter your choice: ")

                if option == '1':
                    column_name = input("Enter the name of the
new column: ")
                    column_type = input("Enter data type for the
new column: ")

                    # Construct the SQL query
                    sql_query = f"ALTER TABLE {tbname} ADD COLUMN
{column_name} {column_type};"

                elif option == '2':
                    mysql.execute(f"SHOW COLUMNS FROM {tbname}")
                    columns = mysql.fetchall()
                    print(columns)
                    old_column_name = input("Enter the name of
the column to modify: ")
                    new_column_name = input("Enter the new name
for the column: ")
                    new_column_type = input("Enter the new data
type for the column: ")

                    # Construct the SQL query
                    sql_query = f"ALTER TABLE {tbname} CHANGE
COLUMN {old_column_name} {new_column_name} {new_column_type};"
```

```
                elif option == '3':
                    column_name = input("Enter the name of the
column to delete: ")

                    # Construct the SQL query
                    sql_query = f"ALTER TABLE {tbname} DROP COL-
UMN {column_name};"

                else:
                    print("Invalid option.")


                try:
                    mysql.execute(sql_query)
                    print(f"Table '{tbname}' altered success-
fully.")

                except mcon.Error as err:
                    print(f"Error altering table: {err}")


            elif choice == '7':
                wildcard = input("Enter the character you want to
search for: ")
                mysql.execute("SHOW TABLES")
                tables = [table[0] for table in mysql.fetchall()]

                for tbname in tables:
                    # Get a list of all columns in the table.
                    mysql.execute(f"SHOW COLUMNS FROM {tbname}")
                    columns = [column[0] for column in
mysql.fetchall()]

                    for column in columns:
                        # Construct and execute a query to search
for the wildcard character in the column.
                        query = f"SELECT * FROM {tbname} WHERE
{column} LIKE %s"
                        params = (f"%{wildcard}%",)  # Add '%'
before and after the wildcard character.

                        mysql.execute(query, params)
                        results = mysql.fetchall()

                        if results:
                            print(f"Table: {tbname}, Column:
{column}")
                            print("Matching Rows:")
                            for row in results:
                                print(row)
            elif choice=='8':
```

```
                print("Possible graphs:"
                      "1. Comparison of prices of medicines"
                      "2. Observed diseases"
                      "3. Exit")
                choice = int(input("Enter your choice (1/2/3):
"))

                if choice == 1:
                    mysql.execute("SELECT NAME, Price FROM medi-
cation_prices")
                    data = mysql.fetchall()
                    if data:
                        df = pd.DataFrame(data, col-
umns=["Medication", "Price"])
                        df.plot.bar(x="Medication", y="Price",
title="Comparison of Medication Prices")
                        plt.show()
                    else:
                        print("No data found for medication
prices.")

                elif choice == 2:
                    while True:
                        lst = []
                        mysql.execute("select diseases from pa-
tients")
                        y = mysql.fetchall()
                        df = pd.DataFrame({"": y})
                        df.columns = ["diseases"]
                        print(df)
                        x = 0
                        while len(df) != x:
                            y = df.iloc[x, 0]
                            input_tuple = y
                            # Extract the string from the tuple
                            input_str = input_tuple[0]

                            # Split the string into a list of
substrings using ',' as the delimiter
                            integers_as_strings = in-
put_str.split(',')

                            # Convert each substring to an inte-
ger and print it
                            for num_str in integers_as_strings:
                                lst.append(int(num_str))
                            x += 1
                        df = pd.DataFrame({"dis": lst})

                        df =
df['dis'].value_counts().reset_index()
```

```
                        mysql.execute("SELECT scientific_name
FROM disease LIMIT 10;")
                        namedis = mysql.fetchall()
                        # Step 8: Plot the disease names and
their counts
                        print(df)
                        plt.bar(df['dis'], df['count'])
                        plt.ylim(300)

                        # Show the plot
                        plt.show()
                elif choice == 3:
                    pass
                else:
                    print("it appears the data you entered is
wrong,kindly re-enter it")
            elif choice == '9':
                my_con.close()
                print("Exiting the program.")
                break
            else:
                print("it appears the data you entered is
wrong,kindly re-enter it")


    elif cat and cat[0] == "doctors":
        while True:
            print("\nMenu:")
            print("1. View data in a table")
            print("2. Update data")
            print("3. Exit")

            choice = int(input("Enter your choice: "))

            if choice == '1':
                mysql.execute("SHOW TABLES")
                tables = mysql.fetchall()

                print("Available Tables:")
                for i, table in enumerate(tables, start=1):
                    print(f"{i}. {table[0]}")

                table_choice = int(input("Select a table (enter
the number): "))
                if 1 <= table_choice <= len(tables):
                    tbname = tables[table_choice - 1][0]
                else:
                    print("Invalid choice. Please select a valid
table.")

                mysql.execute(f"SHOW COLUMNS FROM {tbname}")
                columns = mysql.fetchall()
```

```
                print(f"Columns in {tbname}:")
                for i, column in enumerate(columns, start=1):
                    print(f"{i}. {column[0]}")
                clminp = input("Enter the names of the columns
you want (comma-separated): ")
                if clminp == "":
                    clminp = "*"
                condition = input("Enter the condition (e.g.,
'column_name = value'): ")
                order_direction = input("Enter 'ASC' for ascend-
ing or 'DESC' for descending: ")
                order_direction = order_direction.upper()
                if order_direction not in ["ASC", "DESC"]:
                    order_direction = "ASC"
                if condition == "" and clminp == "":
                    sql_query = "SELECT * FROM " + tbname + " OR-
DER BY " + order_direction + ";"
                elif condition == "":
                    sql_query = "SELECT " + clminp + " FROM " +
tbname + "+ ORDER BY " + order_direction + ";"
                elif clminp == "":
                    sql_query = "SELECT * FROM " + tbname +
"WHERE" + condition + " ORDER BY " + order_direction + ";"
                else:
                    sql_query = "SELECT " + clminp + " FROM " +
tbname + " WHERE " + condition + " ORDER BY " + order_direction +
";"
                print(sql_query)

                try:
                    mysql.execute(sql_query)
                    result = mysql.fetchall()

                    if result:
                        print("Selected Data:")
                        for row in result:
                            print(row)
                    else:
                        print("No data matching the condition.")

                except mcon.Error as err:
                    print(f"Error executing SELECT query: {err}")


            elif choice == '2':
                mysql.execute("SHOW TABLES")
                tables = mysql.fetchall()

                print("Available Tables:")
                for i, table in enumerate(tables, start=1):
                    print(f"{i}. {table[0]}")
```

```
                table_choice = int(input("Select a table (enter
the number): "))
                if 1 <= table_choice <= len(tables):
                    tbname = tables[table_choice - 1][0]
                else:
                    print("Invalid choice. Please select a valid
table.")
                condition = input("Enter the condition (e.g.,
'column_name = value'): ")
                new_data = input("Enter the new data (e.g., 'col-
umn_name = new_value'): ")

                # Construct the SQL query
                sql_query = "UPDATE " + tbname + " SET " +
new_data + " WHERE " + condition + ";"

                try:
                    mysql.execute(sql_query)
                    mysql.connection.commit()
                    print("Data updated successfully.")

                except mcon.Error as err:
                    print(f"Error updating data: {err}")

            elif choice == '3':
                my_con.close()
                print("Exiting the program.")
                break
            else:
                print("it appears the data you entered is
wrong,kindly re-enter it")

    elif cat and cat[0] == "user":
        while True:
            print("\nMenu:")
            print("1.view personal data")
            print("2.view diseases encountered by now")
            print("3.purchase medicines")
            print("4.exit")

            choice = int(input("Enter your choice: "))
            if choice == 1:

                try:
                    sql = "SELECT * FROM patients WHERE email =
%s"
                    params = (username,)
                    mysql.execute(sql, params)
                    result = mysql.fetchall()

                    if result:
```

```
                        print("Selected Data:")
                        for row in result:
                            print(row)
                    else:
                        print("No data matching the condition.")

                except mcon.Error as err:
                    print(f"Error executing SELECT query: {err}")



            if choice == 2:
                ##loading()
                mysql.execute("SELECT diseases FROM patients
WHERE email = %s", (username,))

                # Fetch the results
                myresult = mysql.fetchall()

                # Print the diagnosed diseases and recommended
medicines
                print("You have been diagnosed with the following
diseases (with the names of recommended medicines):")
                for row in myresult:
                    diseases = row[0].split(",")  # Split dis-
eases if they are comma-separated
                    for disease in diseases:
                        disease = disease.strip()  # Remove lead-
ing/trailing spaces (like trim)
                        disease_id = f"d_{disease.replace(' ',
'_')}"  # Create the disease ID
                        disease_id = disease_id.replace(",", "")
# Remove commas from disease ID
                        # Execute a query to fetch details of the
disease and recommended medicines
                        mysql.execute("SELECT * FROM disease
WHERE disease_id = %s", (disease_id,))
                        disease_info = mysql.fetchall()

                        # Print disease information
                        if disease_info:
                            print(
                                f"Disease: {disease_info[0][1]}")
# f for strings and [0][1]Means first row 3rd coloumn
                            print(
                                f"Recommended Medicines: {dis-
ease_info[0][2], disease_info[0][3], disease_info[0][4]}")
                            print()


            if choice == 3:
                print("Available medicines:")
```

```
                mysql.execute("SELECT * FROM medication_prices")
                med = mysql.fetchall()
                for item in med:
                    print(f"ID: {item[3]}, Name: {item[0]},
Price: ${item[1]}, Quantity: {item[2]}")
                mysql.execute("SELECT user_id FROM patients WHERE
email = %s", (username,))
                userid = mysql.fetchall()
                userid= userid[0]
                userid = userid[0]
                item_id = int(input("Enter the ID of the med you
want to buy: "))
                quantity = int(input("Enter the quantity you want
to buy: "))
                dt = datetime.datetime.now()
                mysql.execute("SELECT name, price, qty FROM medi-
cation_prices WHERE med_id = %s", (item_id,))
                item = mysql.fetchone()


                if item:
                    item_name, item_price, item_quantity = item
                    if item_quantity >= quantity:
                        total_cost = item_price * quantity
                        print(f"Item: {item_name}, Quantity:
{quantity}, Total Cost: ${total_cost}")

                        confirm = input("Confirm purchase
(yes/no): ").strip().lower()
                        if confirm == "yes":
                            # Deduct the purchased quantity from
the item's quantity
                            mysql.execute("UPDATE medica-
tion_prices SET qty = qty - %s WHERE med_id = %s",(quantity,
item_id))
                            print(userid, dt, item_id, quantity)

                            sql = "INSERT INTO med_purchased
(uid, date, med_id, qty_pur) VALUES (%s, %s, %s, %s)"


                            params = (userid, dt, item_id, quan-
tity)


                            mysql.execute(sql, params)
                            print("Purchase successful!")
                        else:
                            print("Purchase canceled.")
                    else:
                        print("Insufficient quantity available.")
                else:
```

```
                    print("Item not found.")

            elif choice == 4:
                pass
            else:
                print("it appears the data you entered is
wrong,kindly re-enter it")

    else:
        print("It seems that an unexpected error has occurred.
Please inform the staff about it.")
```

# MySQL USED FOR THE PROJECT

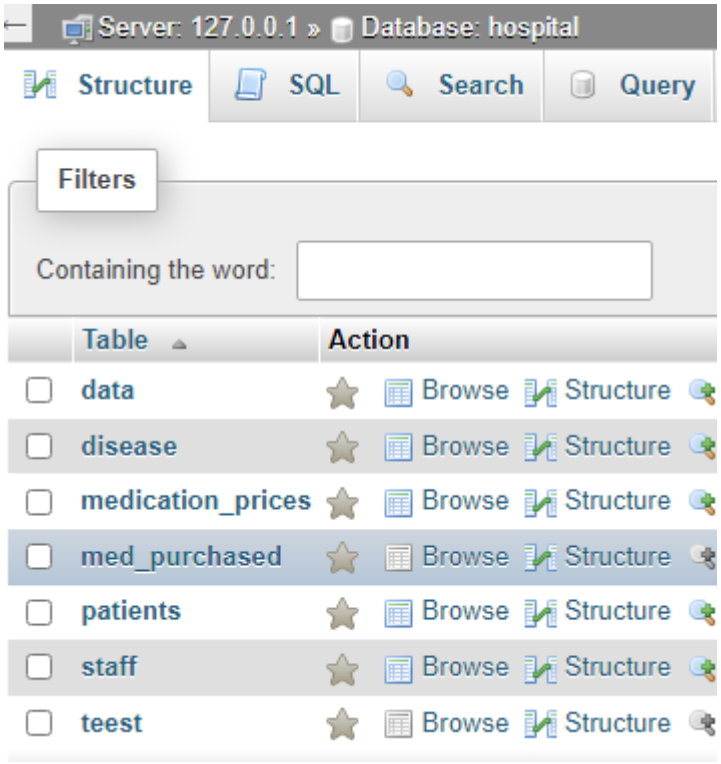



**Figure 3 : Created Database with Name Hospital**

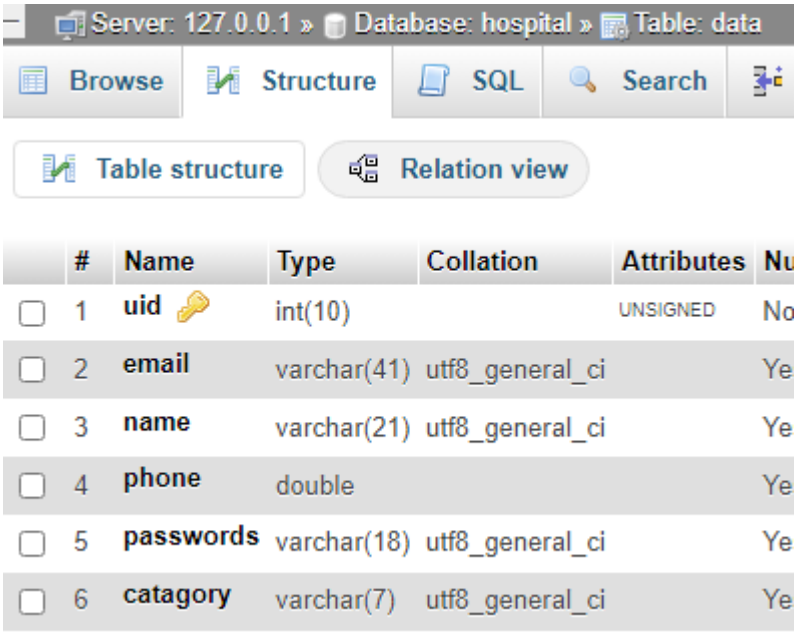



**Figure 4 : Created Table with Name data**

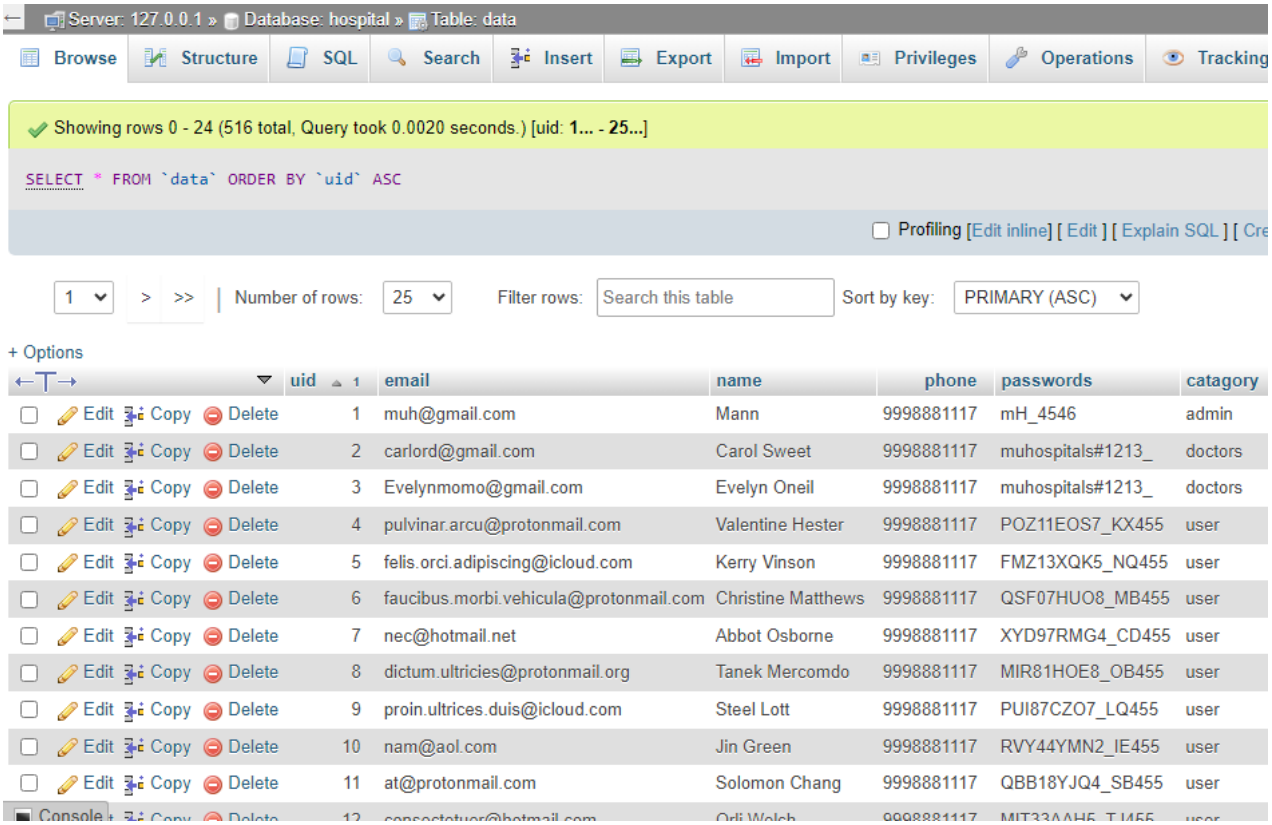



**Figure 5 : Inserted rows in table data**

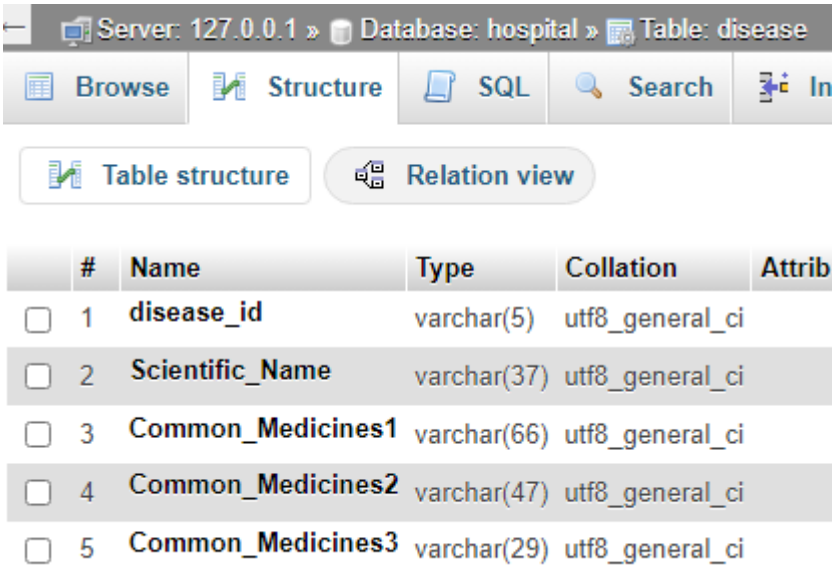
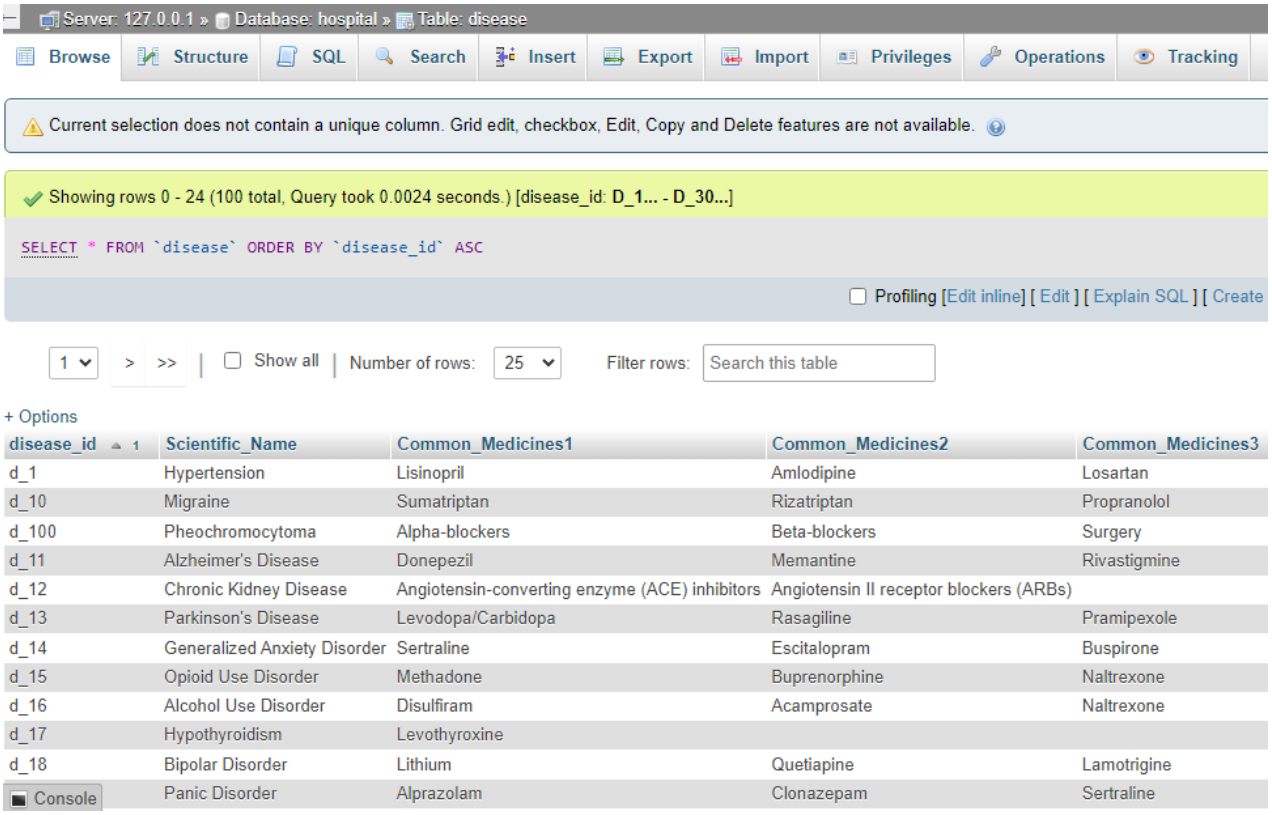



**Figure 6 : Created Table with Name disease**




| disease_id | Scientific_Name | Common_Medicines1 | Common_Medicines2 | Common_Medicines3 |
|---|---|---|---|---|
| d_1 | Hypertension | Lisinopril | Amlodipine | Losartan |
| d_10 | Migraine | Sumatriptan | Rizatriptan | Propranolol |
| d_100 | Pheochromocytoma | Alpha-blockers | Beta-blockers | Surgery |
| d_11 | Alzheimer's Disease | Donepezil | Memantine | Rivastigmine |
| d_12 | Chronic Kidney Disease | Angiotensin-converting enzyme (ACE) inhibitors | Angiotensin II receptor blockers (ARBs) | |
| d_13 | Parkinson's Disease | Levodopa/Carbidopa | Rasagiline | Pramipexole |
| d_14 | Generalized Anxiety Disorder | Sertraline | Escitalopram | Buspirone |
| d_15 | Opioid Use Disorder | Methadone | Buprenorphine | Naltrexone |
| d_16 | Alcohol Use Disorder | Disulfiram | Acamprosate | Naltrexone |
| d_17 | Hypothyroidism | Levothyroxine | | |
| d_18 | Bipolar Disorder | Lithium | Quetiapine | Lamotrigine |
| | Panic Disorder | Alprazolam | Clonazepam | Sertraline |

**Figure 7 : Inserted rows in table disease**

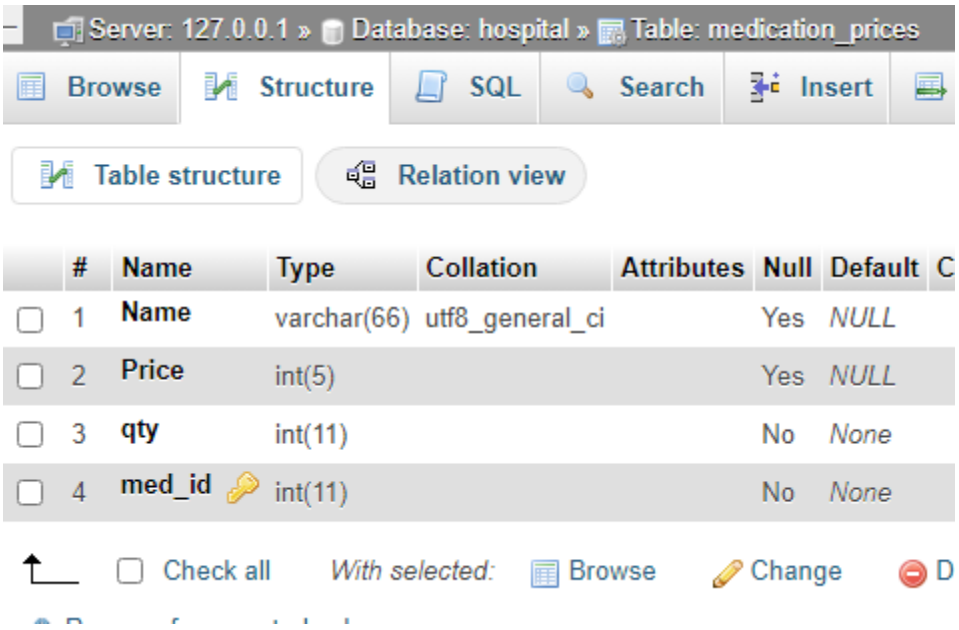



**Figure 8 : Created Table with Name medication_prices**

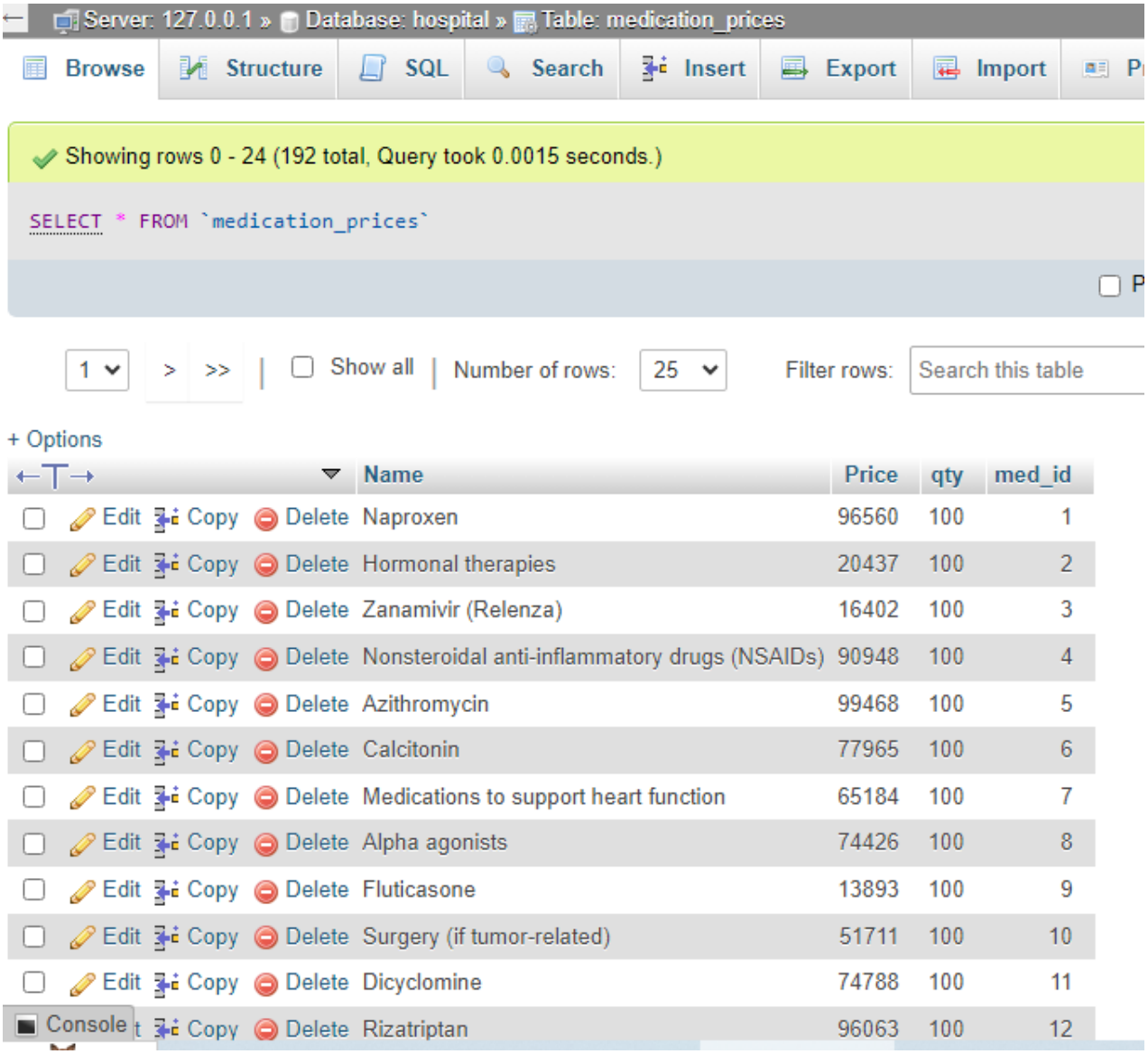



**Figure 9 : Inserted rows in table medication_prices**

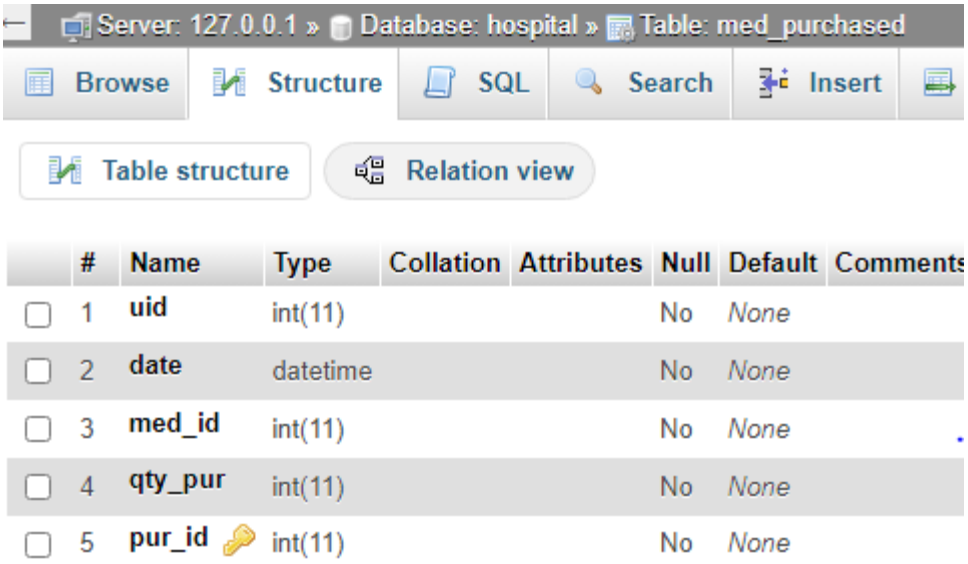



**Figure 10 : Created Table with Name med_purchased**

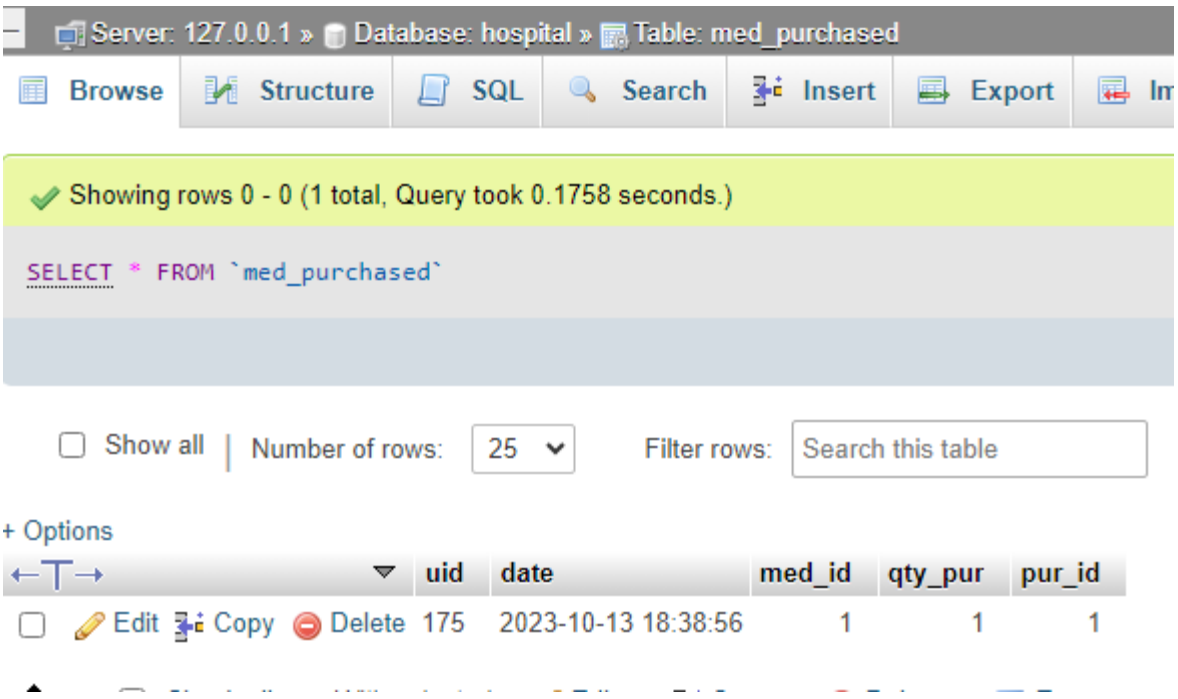
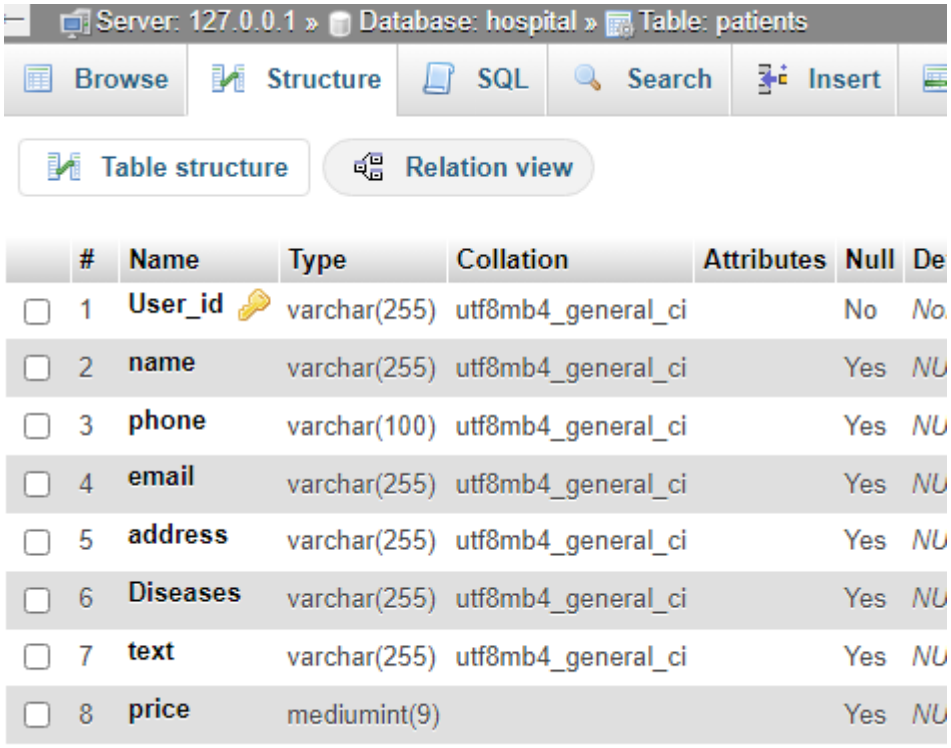
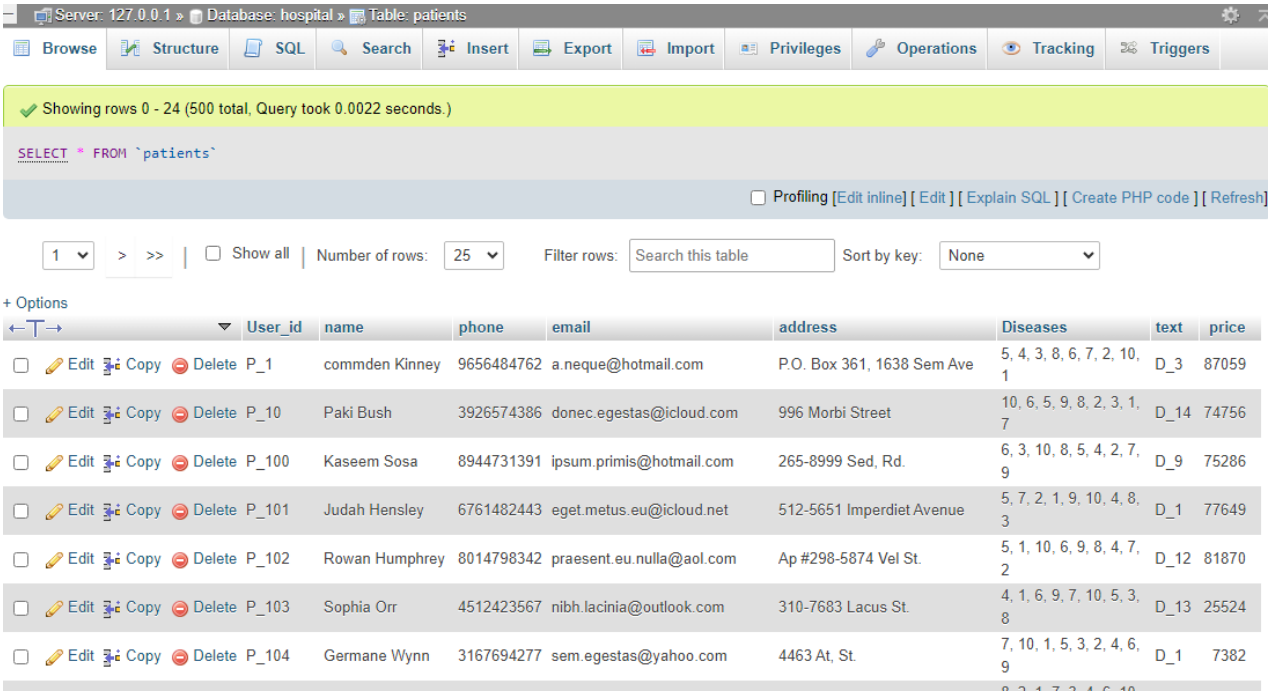



**Figure 11 : Inserted rows in table med_purchased**



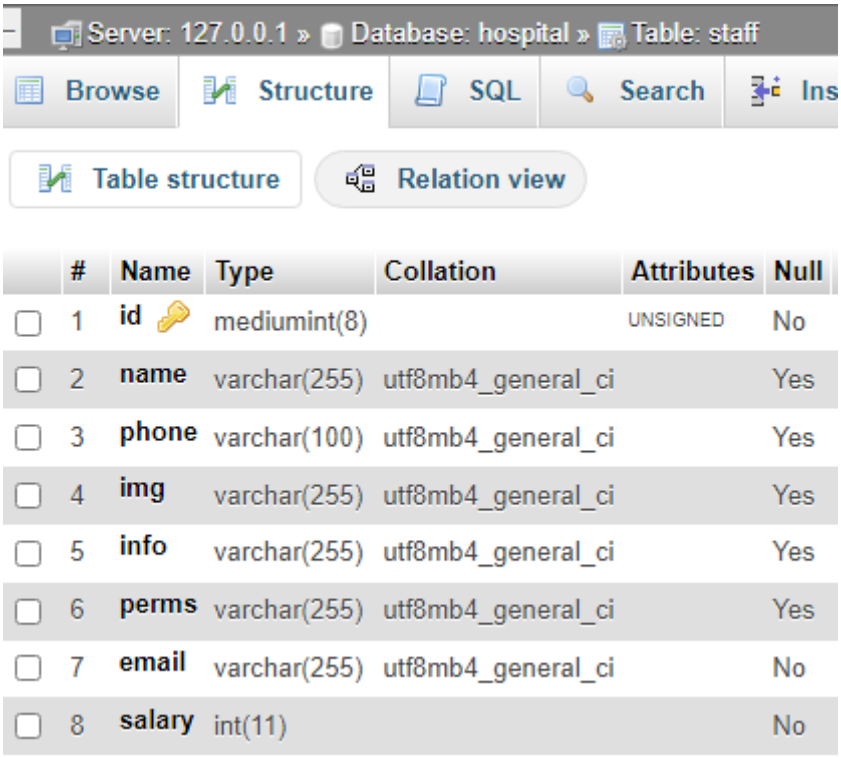

| # | Name | Type | Collation | Attributes | Null | De |
|---|---|---|---|---|---|---|
| 1 | User_id | varchar(255) | utf8mb4_general_ci | | No | No |
| 2 | name | varchar(255) | utf8mb4_general_ci | | Yes | NU |
| 3 | phone | varchar(100) | utf8mb4_general_ci | | Yes | NU |
| 4 | email | varchar(255) | utf8mb4_general_ci | | Yes | NU |
| 5 | address | varchar(255) | utf8mb4_general_ci | | Yes | NU |
| 6 | Diseases | varchar(255) | utf8mb4_general_ci | | Yes | NU |
| 7 | text | varchar(255) | utf8mb4_general_ci | | Yes | NU |
| 8 | price | mediumint(9) | | | Yes | NU |

**Figure 12 : Created Table with Name patients**




| | User_id | name | phone | email | address | Diseases | text | price |
|---|---|---|---|---|---|---|---|---|
| Edit Copy Delete | P_1 | commden Kinney | 9656484762 | a.neque@hotmail.com | P.O. Box 361, 1638 Sem Ave | 5, 4, 3, 8, 6, 7, 2, 10, 1 | D_3 | 87059 |
| Edit Copy Delete | P_10 | Paki Bush | 3926574386 | donec.egestas@icloud.com | 996 Morbi Street | 10, 6, 5, 9, 8, 2, 3, 1, 7 | D_14 | 74756 |
| Edit Copy Delete | P_100 | Kaseem Sosa | 8944731391 | ipsum.primis@hotmail.com | 265-8999 Sed, Rd. | 6, 3, 10, 8, 5, 4, 2, 7, 9 | D_9 | 75286 |
| Edit Copy Delete | P_101 | Judah Hensley | 6761482443 | eget.metus.eu@icloud.net | 512-5651 Imperdiet Avenue | 5, 7, 2, 1, 9, 10, 4, 8, 3 | D_1 | 77649 |
| Edit Copy Delete | P_102 | Rowan Humphrey | 8014798342 | praesent.eu.nulla@aol.com | Ap #298-5874 Vel St. | 5, 1, 10, 6, 9, 8, 4, 7, 2 | D_12 | 81870 |
| Edit Copy Delete | P_103 | Sophia Orr | 4512423567 | nibh.lacinia@outlook.com | 310-7683 Lacus St. | 4, 1, 6, 9, 7, 10, 5, 3, 8 | D_13 | 25524 |
| Edit Copy Delete | P_104 | Germane Wynn | 3167694277 | sem.egestas@yahoo.com | 4463 At, St. | 7, 10, 1, 5, 3, 2, 4, 6, 9 | D_1 | 7382 |

**Figure 13 : Inserted rows in table patients**




| | # | Name | Type | Collation | Attributes | Null |
|---|---|---|---|---|---|---|
| ☐ | 1 | id | mediumint(8) | | UNSIGNED | No |
| ☐ | 2 | name | varchar(255) | utf8mb4_general_ci | | Yes |
| ☐ | 3 | phone | varchar(100) | utf8mb4_general_ci | | Yes |
| ☐ | 4 | img | varchar(255) | utf8mb4_general_ci | | Yes |
| ☐ | 5 | info | varchar(255) | utf8mb4_general_ci | | Yes |
| ☐ | 6 | perms | varchar(255) | utf8mb4_general_ci | | Yes |
| ☐ | 7 | email | varchar(255) | utf8mb4_general_ci | | No |
| ☐ | 8 | salary | int(11) | | | No |

**Figure 14 : Created Table with Name staff**

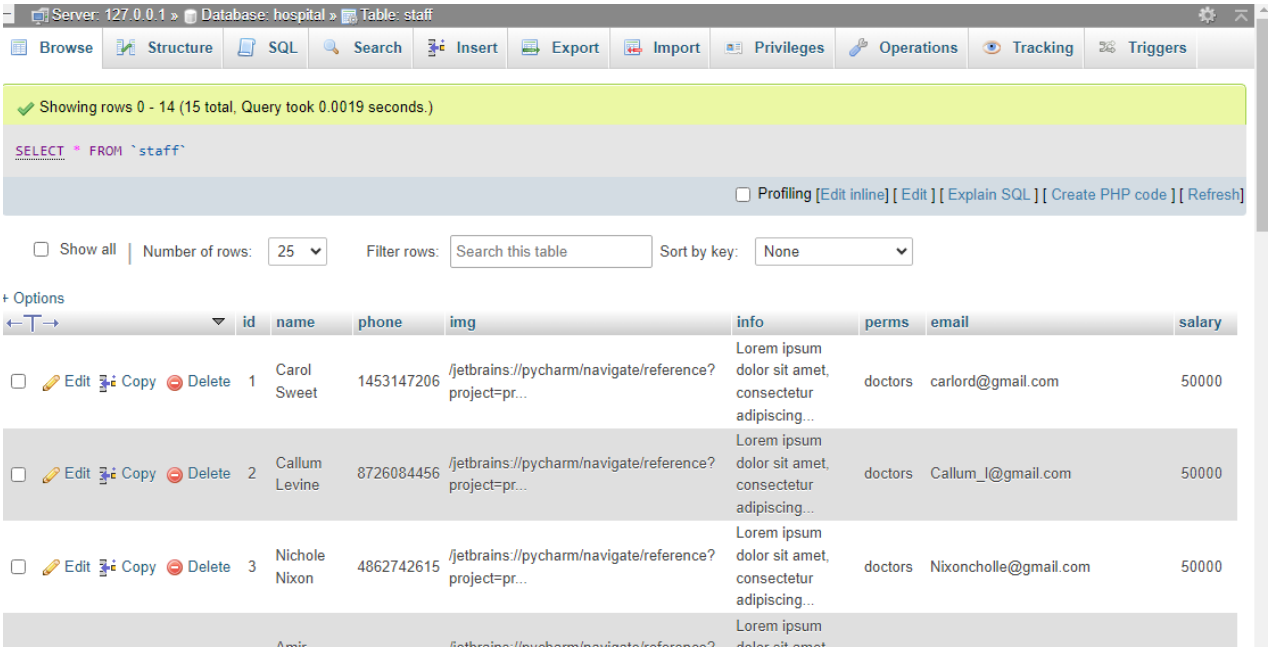



**Figure 15 : Inserted rows in table staff**

# Output of the project:

## ADMIN PANEL

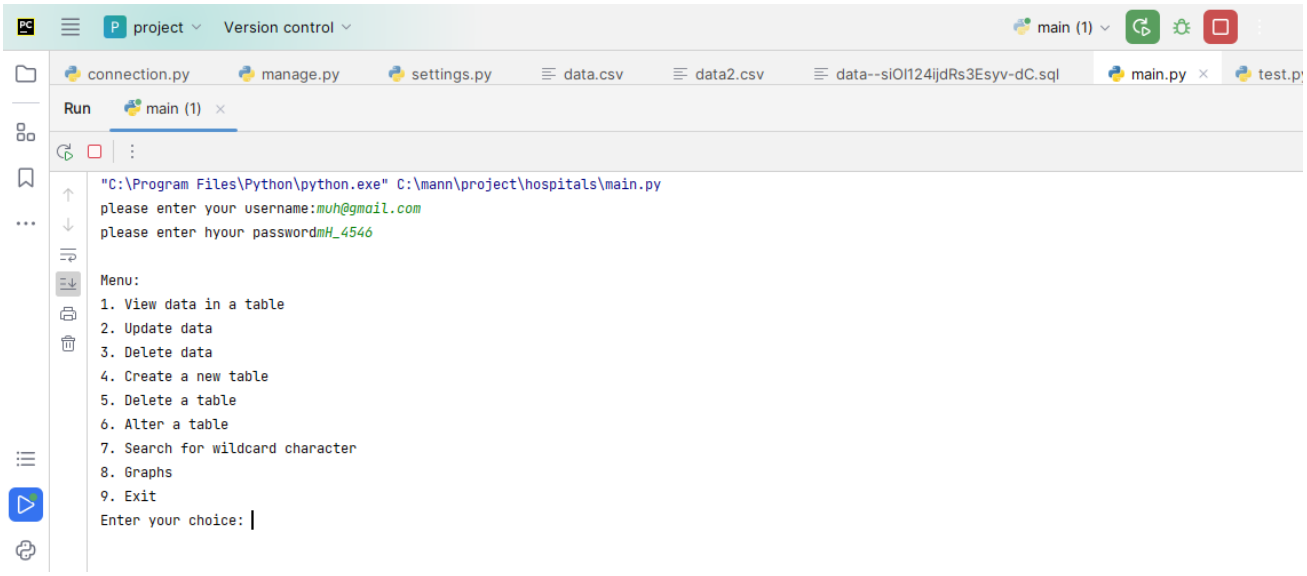



**Figure 16: Execution of the project**

```
"C:\Program Files\Python\python.exe" C:\mann\project\hospitals\main.py
please enter your username:muh@gmail.com
please enter hyour passwordmH_4546
```

## LOGIN AS ADMIN:

```
Menu:
1. View data in a table
2. Update data
3. Delete data
4. Create a new table
5. Delete a table
6. Alter a table
7. Search for wildcard character
8. Graphs
9. Exit
Enter your choice: 1
Available Tables:
1. data
2. disease
3. med_purchased
4. medication_prices
5. patients
6. staff
Select a table (enter the number): 1
Columns in data:
1. uid
2. email
3. name
4. phone
5. passwords
6. catagory
```

```
Enter the names of the columns you want (comma-separated): uid, name
Enter the condition (e.g., 'column_name = value'): uid<10
Enter 'ASC' for ascending or 'DESC' for descending:
SELECT uid, name  FROM data WHERE uid<10 ORDER BY uid ASC;
Selected Data:
(1, 'Mann')
(2, 'Carol Sweet')
(3, 'Evelyn Oneil')
(4, 'Valentine Hester')
(5, 'Kerry Vinson')
(6, 'Christine Matthews')
(7, 'Abbot Osborne')
(8, 'Tanek Mercomdo')
(9, 'Steel Lott')
```

## SELECT DATA FROM TABLE:

```
Menu:
1. View data in a table
2. Update data
3. Delete data
4. Create a new table
5. Delete a table
6. Alter a table
7. Search for wildcard character
8. Graphs
9. Exit
Enter your choice: 2
Available Tables:
1. data
2. disease
3. med_purchased
4. medication_prices
5. patients
6. staff
Select a table (enter the number): 1
Enter the condition (e.g., 'column_name = value'): uid=1
Enter the new data (e.g., 'column_name = new_value'): phone=9817079797
Data updated successfully
```

## UDATE TABLE:

```
Enter your choice: 4
Enter the name of the new table: teest
Enter the number of columns: 1
Enter name for column 1: col1
Enter data type for column 1: int
Table 'teest' created successfully.
```

## CREATE TABLE:

```
Select a table (enter the number): 7
Available options for altering the table:
1. edit column
2. Delete column
Enter your choice: 1
Enter the name of the new column: k
Enter data type for the new column: char
Table 'teest' altered successfully.
```

## ALTER:

```
Enter the character you want to search for: lio
Table: disease, Column: Scientific_Name
Matching Rows:
('d_82', 'Poliomyelitis', 'Vaccination (Prevention)', '', '')
```

## WILDCARD:

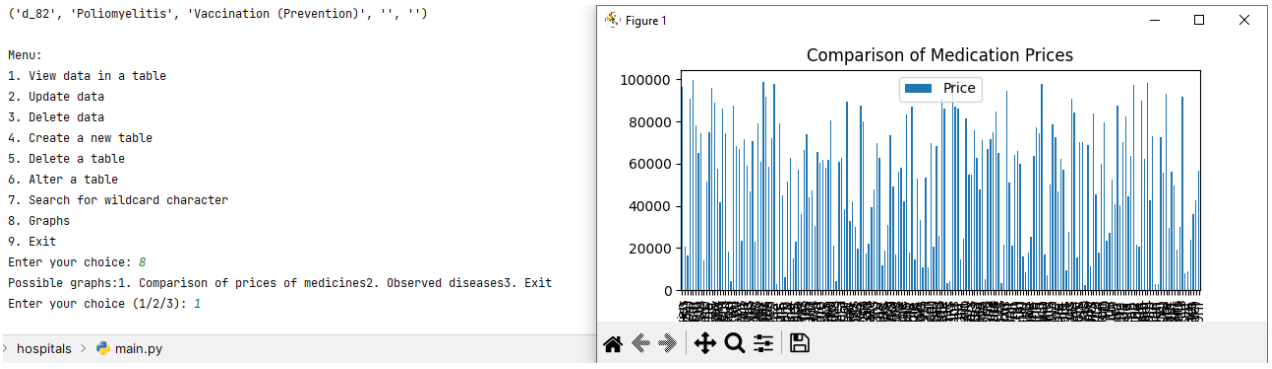



**Figure 5 : Comparison of Medication Prices**

```
Enter your choice: 8
Possible graphs:1. Comparison of prices of medicines2. Observed diseases3. Exit
Enter your choice (1/2/3): 2
```

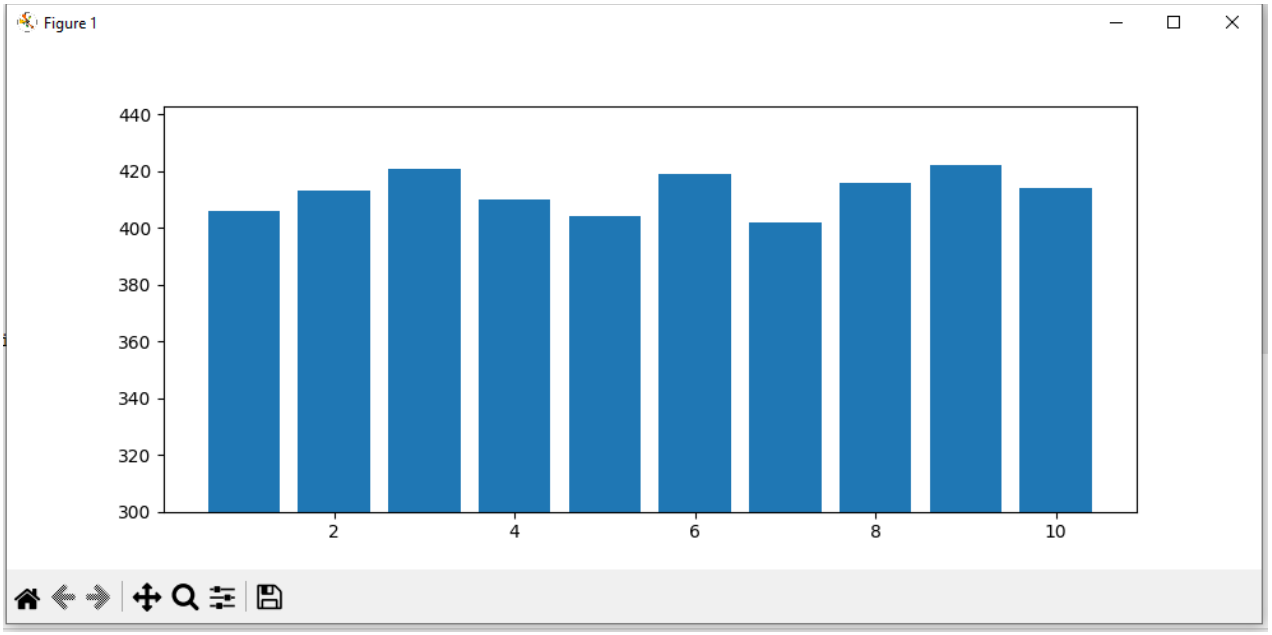



**Figure 5 : Comparison of observed diseases**

## LOGIN AS DOCTOR:

```
please enter your username:carlord@gmail.com
please enter hyour passwordmuhospitals#1213_

Menu:
1. View data in a table
2. Update data
3. Exit
Enter your choice:
```

```
Enter the names of the columns you want (comma-separated): uid, name
Enter the condition (e.g., 'column_name = value'): uid<10
Enter 'ASC' for ascending or 'DESC' for descending:
SELECT uid, name  FROM data WHERE uid<10 ORDER BY uid ASC;
Selected Data:
(1, 'Mann')
(2, 'Carol Sweet')
(3, 'Evelyn Oneil')
(4, 'Valentine Hester')
(5, 'Kerry Vinson')
(6, 'Christine Matthews')
(7, 'Abbot Osborne')
(8, 'Tanek Mercomdo')
(9, 'Steel Lott')
Enter your choice: 2
Available Tables:
1. data
2. disease
3. med_purchased
4. medication_prices
5. patients
6. staff
Select a table (enter the number): 1
Enter the condition (e.g., 'column_name = value'): uid=1
Enter the new data (e.g., 'column_name = new_value'): phone=9817079797
Data updated successfully
```

## LOGIN AS PATIENT:

```
please enter your username:pulvinar.arcu@protonmail.com
please enter hyour passwordPOZ11EOS7_KX455

Menu:
1.view personal data
2.view diseases encountered by now
3.purchase medicines
4.exit
Enter your choice: 1
Selected Data:
('P_175', 'Valentine Hester', '5919125687', 'pulvinar.arcu@protonmail.com', 'Ap #242-2026 Arcu. Ave', '4, 8, 3, 1, 10, 9, 2, 6, 7', 'D_14', 6955)
```

## VIEW OWN DATA:

```
Enter your choice: 2
You have been diagnosed with the following diseases (with the names of recommended medicines):
Disease: Osteoarthritis
Recommended Medicines: ('Acetaminophen', ' Ibuprofen', ' Naproxen')

Disease: Chronic Obstructive Pulmonary Disease
Recommended Medicines: ('Albuterol', ' Tiotropium', ' Fluticasone/Salmeterol')

Disease: Influenza
Recommended Medicines: ('Oseltamivir (Tamiflu)', ' Zanamivir (Relenza)', '')

Disease: Hypertension
Recommended Medicines: ('Lisinopril', ' Amlodipine', ' Losartan')

Disease: Migraine
Recommended Medicines: ('Sumatriptan', ' Rizatriptan', ' Propranolol')

Disease: Epileptic Seizures
Recommended Medicines: ('Levetiracetam', ' Carbamazepine', ' Valproic Acid')

Disease: Diabetes Mellitus Type 2
Recommended Medicines: ('Metformin', ' Insulin', ' Glipizide')

Disease: Gastroesophageal Reflux Disease
Recommended Medicines: ('Omeprazole', ' Esomeprazole', ' Ranitidine')
```

## DISEASES HISTORY:

```
Enter the ID of the med you want to buy: 1
Enter the quantity you want to buy: 1
Item: Naproxen, Quantity: 1, Total Cost: $96560
Confirm purchase (yes/no): YES
P_175 2023-10-16 20:23:42.010899 1 1
Purchase successful!
```

```
Enter your choice: 3
Available medicines:
ID: 1, Name: Naproxen, Price: $96560, Quantity: 100
ID: 2, Name: Hormonal therapies, Price: $20437, Quantity: 100
ID: 3, Name: Zanamivir (Relenza), Price: $16402, Quantity: 100
ID: 4, Name: Nonsteroidal anti-inflammatory drugs (NSAIDs), Price: $90948, Quantity: 100
ID: 5, Name: Azithromycin, Price: $99468, Quantity: 100
ID: 6, Name: Calcitonin, Price: $77965, Quantity: 100
ID: 7, Name: Medications to support heart function, Price: $65184, Quantity: 100
ID: 8, Name: Alpha agonists, Price: $74426, Quantity: 100
ID: 9, Name: Fluticasone, Price: $13893, Quantity: 100
ID: 10, Name: Surgery (if tumor-related), Price: $51711, Quantity: 100
ID: 11, Name: Dicyclomine, Price: $74788, Quantity: 100
ID: 12, Name: Rizatriptan, Price: $96063, Quantity: 100
ID: 13, Name: Riluzole, Price: $89084, Quantity: 100
ID: 14, Name: Hormone replacement therapy, Price: $57517, Quantity: 100
ID: 15, Name: Surgical interventions, Price: $41583, Quantity: 100
ID: 16, Name: Pain management, Price: $86092, Quantity: 100
ID: 17, Name: Anticoagulants (Warfarin  Apixaban), Price: $74501, Quantity: 100
ID: 18, Name: Prednisone, Price: $18103, Quantity: 100
ID: 19, Name: Proton pump inhibitors (PPIs), Price: $4201, Quantity: 100
ID: 20, Name: Surgery (Cataract removal), Price: $87527, Quantity: 100
ID: 21, Name: Blood pressure control, Price: $68251, Quantity: 100
ID: 22, Name: Growth hormone therapy, Price: $66844, Quantity: 100
ID: 23, Name: Blood transfusions, Price: $23408, Quantity: 100
ID: 24, Name: Glucagon, Price: $71808, Quantity: 100
```

## PURCHASE MEDICINES:

```
Enter the ID of the med you want to buy: 1
Enter the quantity you want to buy: 1
Item: Naproxen, Quantity: 1, Total Cost: $96560
Confirm purchase (yes/no): YES
P_175 2023-10-16 20:23:42.010899 1 1
Purchase successful!
```

## ENTERING WRONG EMAIL OR PASSWORD:

```
please enter your username:jotaro@kujo.com
please enter hyour passwordjolynecujoh_3456
It seems that an unexpected error has occurred. Please inform the staff about it.
```

# FURTHER SCOPES OF THE PROJECT

This project can be provided with the GUI functionality. Report facility can also be implemented. Some security implementation like encryption of the password can be incorporated in the project.

# BIBLIOGRAPHY


- https://www.w3schools.com/
- https://ncert.nic.in/
- https://techterms.com/
- https://www.geeksforgeeks.org/
- A book on Informatics Practices by Sumita Arora, Dhanpat Rai Publication
- https://stackoverflow.com/
- https://bugs.python.org/file47781/Tutorial_EDIT.pdf