HEALTH IS WEALTH

MEDWELL

Reva Gupta
Harsh Sharma
XII Aquila

Session: 2024-25

CERTIFICATE

This is to certify that Reva Gupta and Harsh Sharma of Class XII have successfully completed their project ‘MedWell’ under the guidance of Teacher Ms. Stuti Gupta in accordance with the guidelines provided by the Central Board of Secondary Education 2024-25.

Ms. Stuti Gupta
Venkateshwar Global School

ACKNOWLEDGEMENT

We would like to express our sincere gratitude and appreciation to our Computer Science teacher, Ms. Stuti Gupta for guiding us immensely through the course of our project, “MedWell”. Her constructive motivation has played a major role in the successful completion of our project. We give our sincere thanks to our school’s principal, Ms. Namita Singhal, for extending us every possible support or resource we needed for the completion of our project. We would also like to thank our parents for their help and understanding throughout the duration of the compilation of the project.

INTRODUCTION OF THE PROJECT

Medwell: Your Health, Simplified:

In today's fast-paced world, managing your health can often feel overwhelming. Juggling appointments, medications, and finding the right healthcare provider can be a daunting task. Medwell is here to change that.

Imagine a world where healthcare is accessible, personalized, and effortless. Medwell is your personal health companion, designed to simplify every aspect of your well-being. From connecting with trusted doctors to managing your medications, Medwell empowers you to take control of your health journey.

With a focus on patient-centric care, Medwell offers a range of features to support your holistic health. Discover how Medwell can transform the way you experience healthcare.

Goal of the app: Medwell is your dedicated healthcare partner, designed to simplify and improve your overall well-being. With a focus on patient empowerment, Medwell provides a comprehensive suite of tools to manage your health effectively.

Connect with a Doctor-

Experience the convenience of virtual care with Medwell. Easily connect with qualified doctors from the comfort of your home. Whether you have a minor ailment or require expert advice, our platform facilitates seamless communication with healthcare professionals.

Customize User ID-

Take control of your health data with a personalized user ID. Create a secure profile to store and manage your medical information. Enjoy the peace of mind knowing that your sensitive details are protected.

Doctor Directory-

Finding the right doctor for your needs has never been easier. Medwell's doctor directory offers a comprehensive list of healthcare providers specializing in various medical fields. Filter by location, expertise, and availability to find the perfect match for your healthcare needs.

Send Feedback(MESSAGING SYSTEM)

Your opinion matters. Medwell empowers you to share your experiences with your connected doctor. Provide valuable feedback to help improve the quality of care and enhance the overall patient experience.

Medication Management

Stay on top of your medication regimen with Medwell's medication management tool. Easily input and track your prescriptions, set reminders, and monitor your medication history. Take charge of your health by ensuring you're taking the right medications at the right time. With Medwel, you have the tools to actively participate in your healthcare journey.

CODES

Python code { FILES }

(python file)

MAIN FILE.py

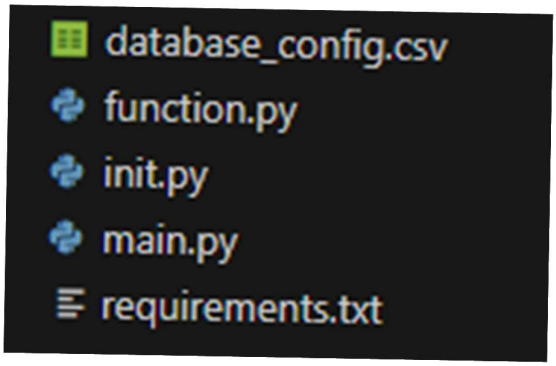
FUNCTIONS.py

INIT.py

(Txt and csv file)

Requirments.txt

Database_config.csv



MAIN FILE

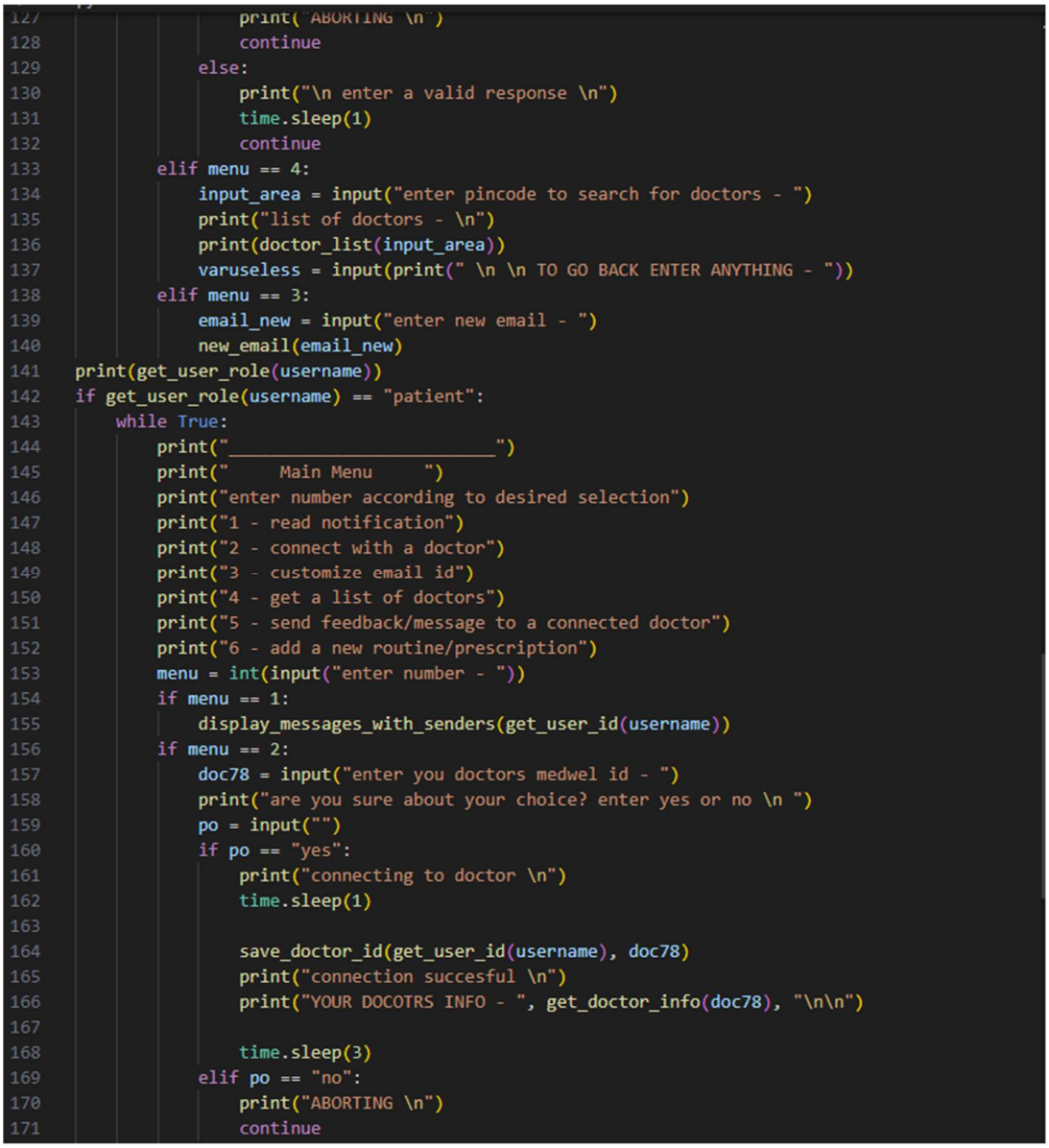
The main file is where all the code is sequenced and run.

It is the starting point of medwell

```python
import time
import os
import subprocess
import sys


def check_requirements():
    if not os.path.isfile("requirements.txt"):
        print("Error: requirements.txt not found. Exiting program.")
        sys.exit(1)

    with open("requirements.txt") as f:
        packages = f.read().splitlines()

    for package in packages:
        try:
            __import__(package.split('==')[0])
        except ImportError:
            print(f"{package} not found. Installing...")
            subprocess.check_call([sys.executable, "-m", "pip", "install", package])

    print("All requirements are satisfied.")

check_requirements()
from function import *


while True:
    print("\n|MEDWELL| - the future of health communication\n")
    print("1. Create an account")
    print("2. Login")
    print("3. connect custom database host")
    choice = input("Enter your choice: ")

    if choice == "1":
        connect_to_database()
        create_account()
```

```
38
39      elif choice == "2":
40          connect_to_database()
41          user = login()
42          from function import username
43          if user == None:
44              print("Invalid username or password")
45              time.sleep(2)
46              print("\n\nwait 5 seconds to return to sign up screen")
47              time.sleep(3)
48              continue
49          else:
50              break
51      elif choice == "2007special":
52          deluser = input("enter username to delete - ")
53          delete_account(deluser)
54      elif choice == "3":
55                  host = input("Enter the database host (e.g., localhost): ")
56                  port = int(input("enter port for your host(e.g, 3306): "))
57                  user = input("Enter the database user: (eg, root): ")
58                  password = input("Enter the database password: (eg, admin): ")
59                  database = input("Enter the database name (leave blank if not needed): ")
60                  save_database_config(host,user,password,port,database)
61      else:
62          print("Invalid choice. Please try again.")
63
64  print("user found - ",)
65  print("loading data - .....")
66
67  if get_user_first_value(username) == 1:
68      p = input("Doctor enter 2 , patient enter 1    -->  ")
69      if p == "2":
70          cursor.execute(
71              "UPDATE users SET role = 'doctor' WHERE username = %s", (username,)
72          )
73          print("your account has succesfully been changed to doctors list")
74          time.sleep(2)
75          print("succesfully finished account initialization")
76          update_user_first_value(username, "2")
77      else:
78          print("role changed to patient")
79          update_user_first_value(username, "2")
```

```
time.sleep(0.5)
print("loading user complete")
notif = notif_grab(username)
time.sleep(0.5)
print("loading functionality")
time.sleep(0.5)
print("_____")
print("Welcome back to medwel")
time.sleep(0.6)
if notif >= 1:

    print(f"\n \n  You have {notif-1} new notifications \n \n ")

if get_user_role(username) == "doctor":
    while True:
        print("_____")
        print("     Main Menu     ")
        print("enter number according to desired selection")
        print("1 - read notification")
        print("2 - enter patient medwel id")
        print("3 - change email")
        print("4 - get a list of doctors")
        print("5 - send feedback to a connected patient")
        menu = int(input("enter number - "))
        if menu == 1:
            display_messages_with_senders(get_user_id(username))
        if menu == 5:
            doctor_id = input("Enter your patient's Medwel ID: ")
            message_text = input("Enter your message: ")
            sender_id = get_user_id(username)  # Get the sender's ID
            recipient_id = doctor_id  # Assuming the doctor ID is the recipient ID
            send_message(sender_id, recipient_id, message_text)
        if menu == 2:
            doc78 = input("enter you patients medwel id - ")
            print("are you sure about your choice? enter yes or no \n ")
            po = input("")
            if po == "yes":
                print("connecting to doctor \n")
                time.sleep(1)

                save_doctor_id(doc78, get_user_id(username))
                print("connection succesful \n")
                print("YOUR PATIENTS INFO - ", get_doctor_info(doc78), "\n\n")

                time.sleep(2)
```



```
                print("ABORTING \n")
                continue
            else:
                print("\n enter a valid response \n")
                time.sleep(1)
                continue
        elif menu == 4:
            input_area = input("enter pincode to search for doctors - ")
            print("list of doctors - \n")
            print(doctor_list(input_area))
            varuseless = input(print(" \n \n TO GO BACK ENTER ANYTHING - "))
        elif menu == 3:
            email_new = input("enter new email - ")
            new_email(email_new)
print(get_user_role(username))
if get_user_role(username) == "patient":
    while True:
        print("_____")
        print("     Main Menu     ")
        print("enter number according to desired selection")
        print("1 - read notification")
        print("2 - connect with a doctor")
        print("3 - customize email id")
        print("4 - get a list of doctors")
        print("5 - send feedback/message to a connected doctor")
        print("6 - add a new routine/prescription")
        menu = int(input("enter number - "))
        if menu == 1:
            display_messages_with_senders(get_user_id(username))
        if menu == 2:
            doc78 = input("enter you doctors medwel id - ")
            print("are you sure about your choice? enter yes or no \n ")
            po = input("")
            if po == "yes":
                print("connecting to doctor \n")
                time.sleep(1)

                save_doctor_id(get_user_id(username), doc78)
                print("connection succesful \n")
                print("YOUR DOCOTRS INFO - ", get_doctor_info(doc78), "\n\n")

                time.sleep(3)
            elif po == "no":
                print("ABORTING \n")
                continue
```

```
        if menu == 3:
            email_new = input("enter new email")
        if menu == 4:
            input_area = input("enter pincode to search for doctors -")
            print("list of doctors - \n")
            print(doctor_list(input_area))
            varuseless = input(print(" \n \n TO GO BACK ENTER ANYTHING - "))

            new_email(email_new)
        if menu == 5:
            doctor_id = input("Enter your doctor's Medwel ID: ")
            message_text = input("Enter your message: ")
            sender_id = get_user_id(username)  # Get the sender's ID
            recipient_id = doctor_id  # Assuming the doctor ID is the recipient ID
            send_message(sender_id, recipient_id, message_text)
        if menu == 6:
            patient_id = get_user_id(username)
            add_routine(patient_id)
```

Function file

The Function file is where all the functions are stored for a clean code

```
import mysql.connector
import time
import csv
import os

CONFIG_FILE = 'database_config.csv'

def connect_to_database():
        global conn
        global cursor
        config = load_database_config()
        conn = mysql.connector.connect(
        host=config['host'],
        user=config['user'],
        password=config['password'],
        port=config['port'],
        database=config['database']
        )
        print("Connection to database successful!")
        cursor = conn.cursor()
        import init
        time.sleep(1)
        return conn
```

```python
def save_database_config(host, user, password, port, database):

    with open(CONFIG_FILE, 'w', newline='') as f:
        writer = csv.writer(f)
        writer.writerow(['host', 'user', 'password', 'port', 'database'])
        writer.writerow([host, user, password, port, database])

def load_database_config():

    if os.path.exists(CONFIG_FILE):
        with open(CONFIG_FILE, 'r') as f:
            reader = csv.reader(f)
            next(reader)  # Skip header row
            row = next(reader, None)
            if row:
                return {
                    'host': row[0],
                    'user': row[1],
                    'password': row[2],
                    'port': int(row[3]),
                    'database': row[4]
                }
    return None


def save_doctor_id(user_id, doctor_id):
    query = "INSERT INTO medical (id, doctors) VALUES (%s, %s) ON DUPLICATE KEY UPDATE doctors = %s"
    cursor.execute(query, (user_id, doctor_id, doctor_id))
    conn.commit()


def send_message(sender_id, recipient_id, message_text):
    query = "INSERT INTO messages (sender_id, recipient_id, message_text) VALUES (%s, %s, %s)"
    cursor.execute(query, (sender_id, recipient_id, message_text))
    conn.commit()


def get_messages(user_id):
    query = "SELECT * FROM messages WHERE recipient_id = %s"
    cursor.execute(query, (user_id,))
    messages = cursor.fetchall()
    return messages
```

```python
def get_user_role(username):
    user_id = get_user_id(username)
    query = "SELECT role FROM users WHERE id = %s"
    cursor.execute(query, (user_id,))
    result = cursor.fetchone()
    if result:
        return result[0]
    else:
        return None


def notif_grab(username):
    try:
        user_id = get_user_id(username)
        cursor.execute("SELECT notifs FROM users WHERE id = %s", (user_id,))
        result = cursor.fetchone()
        return result[0] if result else 0
    except Exception as e:
        print(f"Error in notif_grab: {e}")
        return 0

def update_user_first_value(username, new_value):

    query = "UPDATE users SET first = %s WHERE username = %s"
    cursor.execute(query, (new_value, username))

    conn.commit()
    print(f"Updated first value for user {username} to {new_value}")


def get_user_first_value(username):
    query = "SELECT first FROM users WHERE username = %s"
    cursor.execute(query, (username,))
    result = cursor.fetchone()
    if result:

        return result[0]  # Return the value of the "1st" column
    else:
        return None
```

```python
def check_username_exists(username):
    query = "SELECT COUNT(*) FROM users WHERE username = %s"
    cursor.execute(query, (username,))
    count = cursor.fetchone()[0]
    return count > 0


def check_email_exists(email):
    query = "SELECT COUNT(*) FROM users WHERE email = %s"
    cursor.execute(query, (email,))
    count = cursor.fetchone()[0]
    return count > 0


def create_account():
    username = input("Enter a username: ")
    if check_username_exists(username):
        print("Username already exists. Please choose a different username.")
        time.sleep(1)
        print("\n \n")
        time.sleep(1)
        return
    password = input("Enter a password: ")
    email = input("Enter an email: ")
    if check_email_exists(email):
        print("Email already exists. Please choose a different email.")
        time.sleep(1)
        print("\n \n")
        time.sleep(1)
        return
    area = input("enter your pincode")
    time.sleep(1)
    print("\n registering user \n")
    time.sleep(2)
    cursor.execute(
        """
        INSERT INTO users (username, password, email, area)
        VALUES (%s, %s, %s, %s)
    """,
        (username, password, email, area),
    )
    print(" succesful \n")
    conn.commit()
```

```python
def login():
    global username
    username = input("Confirm your username: ")
    password = input("Enter your password: ")
    cursor.execute(
        """
        SELECT * FROM users
        WHERE username = %s AND password = %s
    """,
        (username, password),
    )
    user = cursor.fetchone()
    return username


def delete_account(username):
    query = "DELETE FROM users WHERE username = %s"
    cursor.execute(query, (username,))

    conn.commit()
    print(f"Account for {username} deleted successfully!")


def get_user_id(username):
    cursor.execute("SELECT id FROM users WHERE username = %s", (username,))
    result = cursor.fetchone()
    if result:
        return result[0]


def get_doctor_info(doctor_id):
    query = "SELECT username,role,email FROM users WHERE id = %s AND role = 'doctor'"
    cursor.execute(query, (doctor_id,))
    doctor_info = cursor.fetchone()
    return doctor_info


def doctor_list(pin):
    query = "SELECT username,role,email,area FROM users WHERE area = %s AND role = 'doctor' "
    cursor.execute(query, (pin,))
    doctor_info = cursor.fetchall()
    return doctor_info
```

```python
def new_email(new):
    print("feature still in development ")


def display_messages_with_senders(user_id):
    query = """
    SELECT m.message_text, m.created_at, u.username
    FROM messages m
    JOIN users u ON m.sender_id = u.id
    WHERE m.recipient_id = %s
    ORDER BY m.created_at DESC
    """
    cursor.execute(query, (user_id,))
    messages = cursor.fetchall()

    if not messages:
        print("\nNo messages found.\n")
        time.sleep(2)
        return

    for message in messages:
        message_text, created_at, sender_username = message
        print(f"[{created_at}] {sender_username}: {message_text}")


def send_message(sender_id, recipient_id, message_text):
    query = "INSERT INTO messages (sender_id, recipient_id, message_text) VALUES (%s, %s, %s)"
    cursor.execute(query, (sender_id, recipient_id, message_text))
    conn.commit()
    print("Message sent successfully!")

def add_routine(patient_id):
    routine_description = input("Enter routine description: ")
    routine_type = input("Enter routine type (e.g., medication, exercise): ")
    start_date = input("Enter start date (YYYY-MM-DD): ")
    end_date = input("Enter end date (YYYY-MM-DD) or leave blank: ")

    # If end_date is not provided, set it to None
    if end_date == "":
        end_date = None
```

```python
  # Insert the routine into the database
  cursor.execute(
      """
      INSERT INTO Routines (patient_id, routine_description, routine_type, start_date, end_date)
      VALUES (%s, %s, %s, %s, %s)
      """,
      (patient_id, routine_description, routine_type, start_date, end_date)
  )
  conn.commit()  # Commit the changes to the database
  print("Routine added successfully!")
```

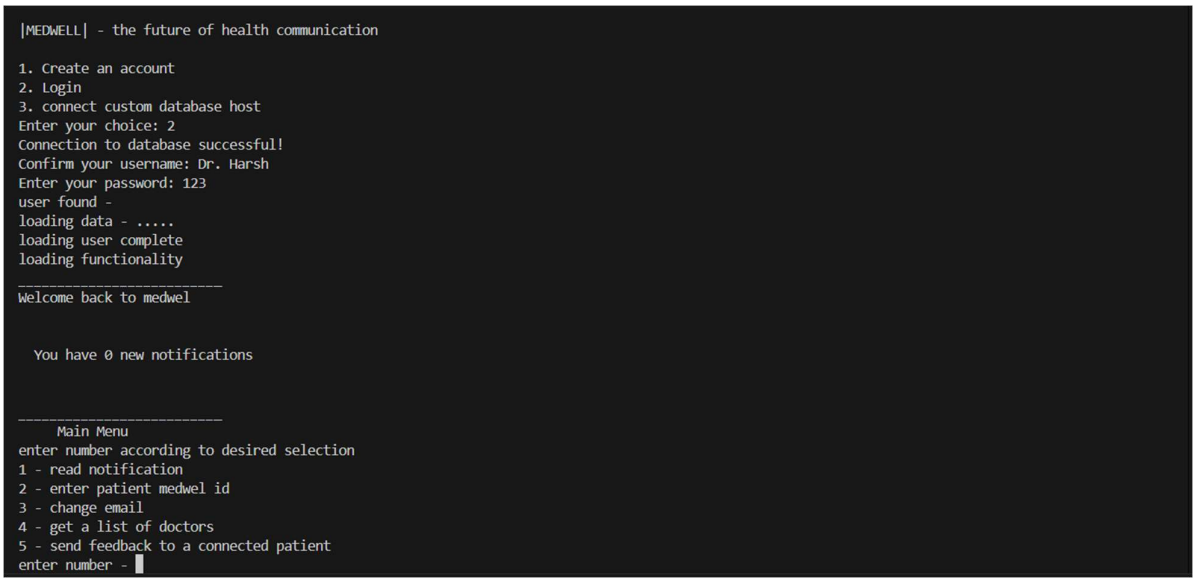
INIT FILE

```
import function as fn


cursor =  fn.cursor

cursor.execute("CREATE DATABASE IF NOT EXISTS user_ids")
cursor.execute("USE user_ids")
cursor.execute(
    """
    CREATE TABLE IF NOT EXISTS users (
                    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
                    username VARCHAR(50) NOT NULL,
                    role VARCHAR(20) NOT NULL DEFAULT 'patient',
                    password VARCHAR(255) NOT NULL,
                    first INT(20) DEFAULT 1,
                    area VARCHAR(20) NOT NULL,
                    email VARCHAR(100) ,
                    notifs INT DEFAULT 1
        )

"""
)
cursor.execute(
    """
    CREATE TABLE IF NOT EXISTS medical (
                id INT references users.id ,
                med_data varchar(100) ,
                doctors varchar(50) ,
                feedback varchar(50)

        )

"""
)
cursor.execute(
    """
    CREATE TABLE IF NOT EXISTS messages (
        id INT AUTO_INCREMENT,
        sender_id INT,
        recipient_id INT,
        message_text TEXT,
        created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
        PRIMARY KEY (id)
    )
"""
```

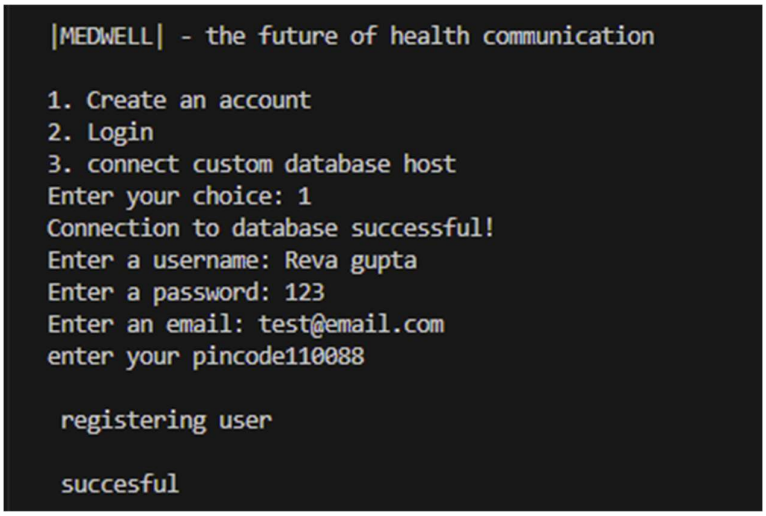
```
cursor.execute(
    """
    CREATE TABLE IF NOT EXISTS Routines (
        patient_id INT NOT NULL,
        routine_description TEXT NOT NULL,
        routine_type VARCHAR(50),
        start_date DATE,
        end_date DATE,
        created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
        FOREIGN KEY (patient_id) REFERENCES users(id)
    )
"""
)
```

OUTPUT CODE

Interface



Creating a account



MAIN MENU (patient)

```
-------------------------
     Main Menu
enter number according to desired selection
1 - read notification
2 - connect with a doctor
3 - customize email id
4 - get a list of doctors
5 - send feedback/message to a connected doctor
6 - add a new routine/prescription
```

MAIN MENU (doctor)

```
-------------------------
     Main Menu
enter number according to desired selection
1 - read notification
2 - enter patient medwel id
3 - change email
4 - get a list of doctors
5 - send feedback to a connected patient
```

MESSAGING

Sending a message

```
enter number - 5
Enter your doctor's Medwel ID: 6
Enter your message: Hello doctor please update my prescription.
Message sent successfully!
-------------------------
```

Reading a message

```
enter number - 1
[2024-11-05 16:57:48] Reva gupta: Hello doctor please update my prescription.
```

Adding a prescription(patient)

```
enter number - 6
Enter routine description: 3 x crocin 5mg tablets,1x melatoonin 5mg
Enter routine type (e.g., medication, exercise): medication
Enter start date (YYYY-MM-DD): 2024-07-09
Enter end date (YYYY-MM-DD) or leave blank:
Routine added successfully!
```

Searching for doctor

```
enter number - 4
enter pincode to search for doctors -110088
list of doctors -

[('Rauhak', 'doctor', 'test@test.com', '110088')]
```

Customizing email

```
enter number - 3
enter new email TEST@TEST.COM
```

SQL Table

User table

```
mysql> select * from users;
+----+-----------+---------+----------+-------+--------+-----------------+--------+
| id | username  | role    | password | first | area   | email           | notifs |
+----+-----------+---------+----------+-------+--------+-----------------+--------+
|  1 | harsh     | patient | 123      |     2 | 110088 | Stud#ent@123    |      1 |
|  2 | Rauhak    | doctor  | 123      |     2 | 110088 | test@test.com   |      1 |
|  3 | reva      | patient | 123      |     1 | 110088 | reva@test.com   |      1 |
|  5 | Reva gupta | patient | 123      |     2 | 110088 | test@email.com  |      1 |
|  6 | Dr. Harsh | doctor  | 123      |     2 | 110034 | harsh@gmail.com |      1 |
+----+-----------+---------+----------+-------+--------+-----------------+--------+
5 rows in set (0.00 sec)
```

Messaging table

```
mysql> select * from messages;
+----+-----------+--------------+-----------------------------------------+---------------------+
| id | sender_id | recipient_id | message_text                            | created_at          |
+----+-----------+--------------+-----------------------------------------+---------------------+
|  1 |         5 |            6 | Hello doctor please update my prescription. | 2024-11-05 16:57:48 |
+----+-----------+--------------+-----------------------------------------+---------------------+
1 row in set (0.00 sec)
```

Connection table

```
mysql> select * from medical;
+------+----------+---------+----------+
| id   | med_data | doctors | feedback |
+------+----------+---------+----------+
|    4 | NULL     | 2       | NULL     |
+------+----------+---------+----------+
1 row in set (0.00 sec)
```

Routines table

```
+------------+-----------------------------------------+--------------+------------+------------+---------------------+
| patient_id | routine_description                     | routine_type | start_date | end_date   | created_at          |
+------------+-----------------------------------------+--------------+------------+------------+---------------------+
|          5 | 5mg melatonin daily 2 hours before bed  | medication   | 2024-07-08 | 2024-08-08 | 2024-11-05 16:58:43 |
|          5 | 3 x crocin 5mg tablets,1x melatoonin 5mg | medication   | 2024-07-09 | NULL       | 2024-11-05 17:20:22 |
+------------+-----------------------------------------+--------------+------------+------------+---------------------+
```

REFERENCES

1. https://dev.mysql.com/
2. www.python.org
3. https://www.w3schools.com/python/python_mysql_getstarted.asp
4. Code Academy