ASSIGNMENT – 2

Name	T. RAJDEEP TUKARAM PATIL
Batch	B7-1A3E

AIM

1. Create User table with user with email, username, roll number, password.
2. Perform UPDATE, DELETE Queries with user table 3. Connect python code to db2.
4. Create a flask app with registration page, login page and welcome page. By default, load the registration page once the user enters all the fields store the data in database and navigate to login page authenticate user username and password. If the user is valid show the welcome page.

DIRECTORY STRUCTURE

- static/styles
  - home.css
  - login.css
  - signup.css
  - users_list.css
  - user_update.css
- templates
  - home.html
  - login.html
  - signup.html
  - users_list.html
  - user_update.html
- app.py

CODE 1. HTML FILES

a) home.html

```
<!DOCTYPE html>

<html>

<link rel="stylesheet" href="{{url_for('static', filename='styles/home.css')}}">

<body>

    <h2>Welcome, {{name}}</h2>
```

```
</body>
</html>
```

b) login.html

```
<!DOCTYPE html>

<html>

<link rel="stylesheet" href="{{url_for('static', filename='styles/login.css')}}">

<body>
    <form action="/" method="POST">
        <h1>CRUD APPLICATION !!!</h1>

        <h3>Login</h3>

        <div class="error">
            {% if error %}
               {{ error }}
            {% endif %}
       </div>


        <label>
            Username
        </label>

        <br>

        <input type="text" name="username" required>

        <br><br>

        <label>
            Password
        </label>

        <br>

        <input type="password" name="password" required>
        <br><br>

        <button type="submit">SIGN IN</button>
```

```
        <br><br><br>

        Don't have an account? <a href="signup">Sign up</a>
        <br><br><br>
    </form>
</body>
</html>
```

c) signup.html

```
<!DOCTYPE html>

<html>

<link rel="stylesheet" href="{{url_for('static', filename='styles/signup.css')}}">

<body>
    <form action="/signup" method="POST">
<h1>CRUD APPLICATION !!!</h1>

        <h3>Signup</h3>

        <div class="error">
            {% if error %}
               {{ error }}
            {% endif %}
       </div>

       <div class="msg">
            {% if msg %}
                {{ msg }}
            {% endif %}
        </div>

        <label>
            Email
        </label>

        <br>

        <input type="email" name="email">

        <br><br>
```

```
        <label>
            Username
        </label>

        <br>

        <input type="text" name="username" required>

        <br><br>


        <label>
            Roll Number
        </label>

        <br>

        <input type="text" name="roll_no" required>

        <br><br>


        <label>
            Password
        </label>

        <br>

        <input type="password" name="password" required>

        <br><br>

        <button type="submit">SIGN UP</button>

        <br><br><br>

        Already have an account? <a href="/">Sign in</a>

        <br><br><br>
    </form>
</body>

</html>
```

d) users_list.html

```
<!DOCTYPE html>

<html>

<link rel="stylesheet" href="{{url_for('static', filename='styles/users_list.css')}}">

<body>

    <h1>Users List</h1>

    <table>
        <thead>
            <th>USERNAME</th>
            <th>EMAIL</th>
            <th>ROLL_NO</th>
            <th>OPTIONS</th>
        </thead>

        <tbody>
            {% for user in users %}
                <tr>
                    <td>{{user[0]}}</td>
                    <td>{{user[1]}}</td>
                    <td>{{user[2]}}</td>
                    <td>
                        <a href="/user/{{user[0]}}/update">Update</a>
                        <a href="/user/{{user[0]}}/delete">Delete</a>
                    </td>
                </tr>
            {% endfor %}
        </tbody>
    </table>

</body>
</html>
```

e) user_update.html

```
<!DOCTYPE html>

<html>

<link rel="stylesheet" href="{{url_for('static', filename='styles/user_update.css')}}">

<body>
    <form action="/user/{{username}}/update" method="POST">
        <h1>UPDATE USER!</h1>

        <div class="error">
            {% if error %}
               {{ error }}
            {% endif %}
       </div>



        <label>
            Username
        </label>

        <br>

        <input type="text" name="username" value="{{username}}" required>

        <br><br>


        <label>
            Email
        </label>

        <br>

        <input type="email" name="email" value="{{email}}" required>
        <br><br>


        <label>
            Roll No
        </label>

        <br>

        <input type="text" name="roll_no" value="{{roll_no}}" required>
```

```
        <br><br>

        <button type="submit">UPDATE</button>

        <br><br><br>
    </form>
</body>
</html>
```

2. CSS FILES

a) home.css

```
h2 {     text-align:
center; }
```

b) login.css

```
.error {
color:
red;
}
```

c) signup.css

```
.error {
color: red;
}


.msg {
color: green;
}
```

d) users_list.css

```
table {     border: 1px solid black;     text-align: center;
}

th, td {     border: 1px solid black;     text-align: center;
}
```

e) user_update.css

```
.error {
color:
red;
}
```

3. app.py

NOTE: AS IBM CLOUD ACCOUNT IS NOT ACCESSIBLE, POSTGRES DB IS USED

```
import psycopg2
 from flask import Flask from
flask import render_template
from flask import request from
flask import redirect


conn = psycopg2.connect(database="crudapp", user="postgres", password="password",
host="127.0.0.1", port="5432")
 app =
Flask(__name__)

@app.route("/",methods=["GET","POST"])
def login():
if(request.method=="GET"):
        return render_template("login.html", error=None)
elif(request.method=="POST"):
        username = request.form["username"]
password = request.form["password"]
         cursor =
conn.cursor()


        query = f'select username from users where username = \'{username}\' and password =
\'{password}\''
cursor.execute(query)         info
= cursor.fetchone()
```

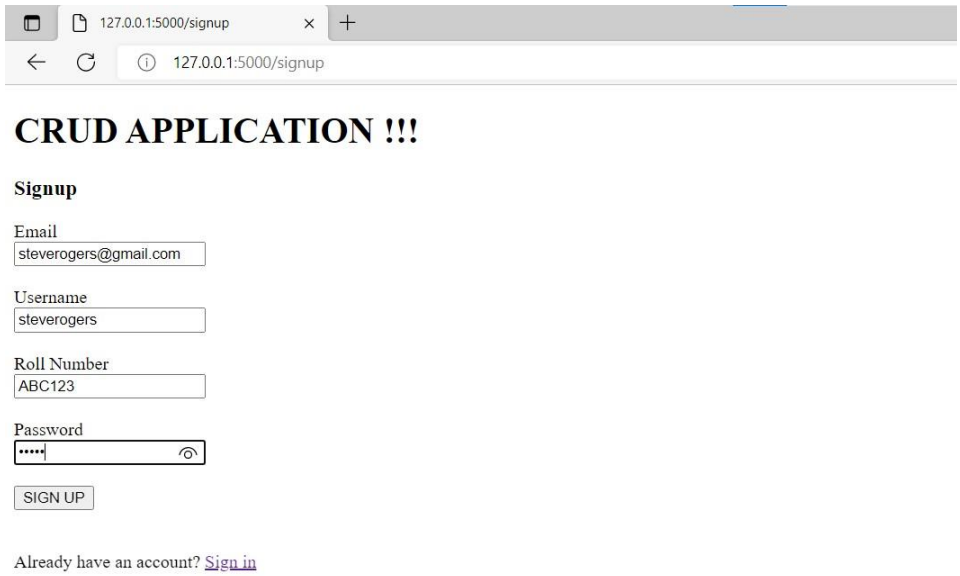
```
        if(info == None):
            return render_template("login.html", error="INVALID CREDENTIALS !!!")


        return render_template("home.html", name=username)

@app.route("/signup",methods=["GET","POST"])
def signup():     if(request.method=="GET"):
        return render_template("signup.html", error=None, msg=None)
elif(request.method=="POST"):
        username = request.form["username"]
email = request.form["email"]         roll_no
= request.form["roll_no"]         password =
request.form["password"]
         cursor =
conn.cursor()
         query = f'select email from users where email =
\'{email}\''         cursor.execute(query)         rows =
cursor.fetchall()

if(len(rows)!=0):
            return render_template("signup.html", error="EMAIL ALREADY EXISTS !!!")


        query = f'select username from users where username = \'{username}\''
cursor.execute(query)         rows = cursor.fetchall()

if(len(rows)!=0):
            return render_template("signup.html", error="USERNAME ALREADY EXISTS !!!")
         query = "insert into users values(%s, %s, %s, %s)"
cursor.execute(query, (email, username, roll_no, password))
conn.commit()
         return render_template("signup.html", error=None,msg="ACCOUNT CREATED
SUCCESSFULLY
!!! PLEASE LOGIN")



@app.route("/users",methods=["GET","POST"])
def users():     if(request.method=="GET"):
cursor = conn.cursor()
         query = f'select username, email, roll_no from
users'
```

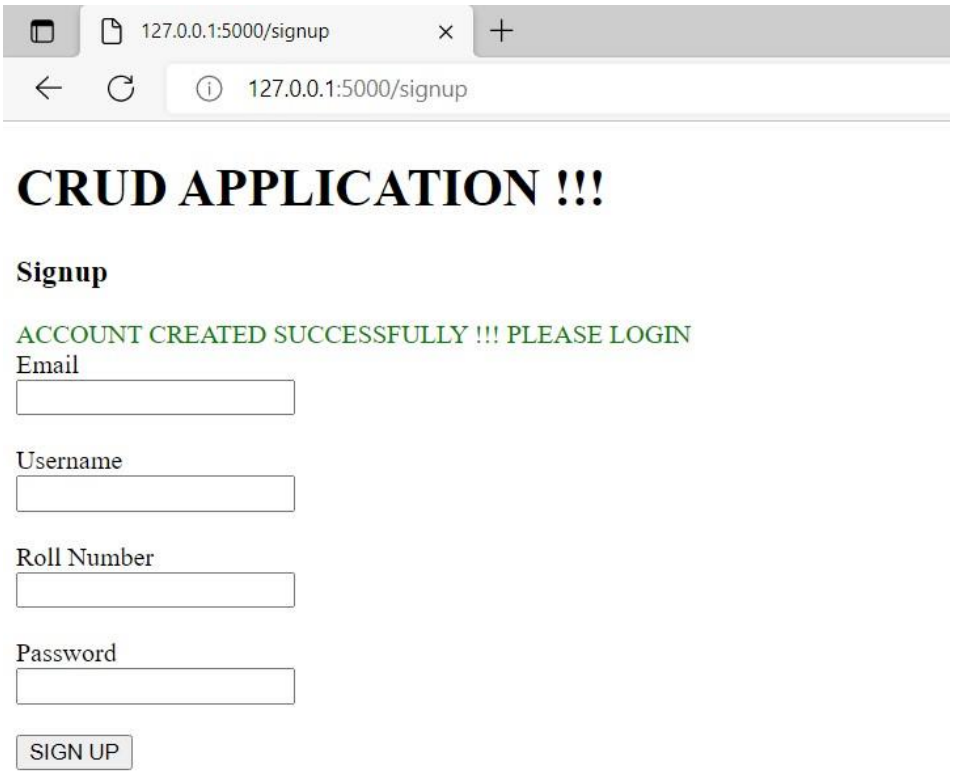
```
        cursor.execute(query)
rows = cursor.fetchall()
         return render_template("users_list.html",
users=rows)     elif(request.method=="POST"):
        username = request.form["username"]
email = request.form["email"]         roll_no
= request.form["roll_no"]         password =
request.form["password"]
         cursor =
conn.cursor()
         query = f'select email from users where email =
\'{email}\''         cursor.execute(query)         rows =
cursor.fetchall()

if(len(rows)!=0):
            return render_template("signup.html", error="EMAIL ALREADY EXISTS !!!")


        query = f'select username from users where username = \'{username}\''
cursor.execute(query)         rows = cursor.fetchall()

if(len(rows)!=0):
            return render_template("signup.html", error="USERNAME ALREADY EXISTS !!!")
         query = "insert into users values(%s, %s, %s, %s)"
cursor.execute(query, (email, username, roll_no, password))
conn.commit()
         return render_template("signup.html", error=None,msg="ACCOUNT CREATED
SUCCESSFULLY
!!! PLEASE LOGIN")


@app.route("/user/<string:username>/update",methods=["GET","POST"])
def userUpdate(username):     if(request.method=="GET"):
cursor = conn.cursor()
         query = f'select username, email, roll_no from users where username
=
\'{username}\''
cursor.execute(query)         info
= cursor.fetchone()
         return
render_template("user_update.html",username=info[0],email=info[1],roll_no=info[2],error=Non
e)     elif(request.method=="POST"):
```

```
        new_username =
request.form["username"]         email =
request.form["email"]         roll_no =
request.form["roll_no"]
         cursor =
conn.cursor()
         query1 = f'select email from users where email = \'{email}\' and username
<>
\'{username}\''
cursor.execute(query1)         rows1
= cursor.fetchall()


        query2 = f'select username from users where username = \'{new_username}\' and
username <> \'{username}\''         cursor.execute(query2)         rows2 =
cursor.fetchall()


        if(len(rows1) == 0 and len(rows2) == 0):
            query = "update users set email=%s, username=%s, roll_no=%s where username=%s"
cursor.execute(query, (email, new_username, roll_no, username))             conn.commit()
         return
redirect("/users")

@app.route("/user/<string:username>/delete")
def userDelete(username):     cursor =
conn.cursor()
     query = f'delete from users where
username=\'{username}\''     cursor.execute(query)
conn.commit()
     return
redirect("/users")


if __name__=="__main__":
    app.run()
conn.close()
```

OUTPUT 1. REGISTRATION

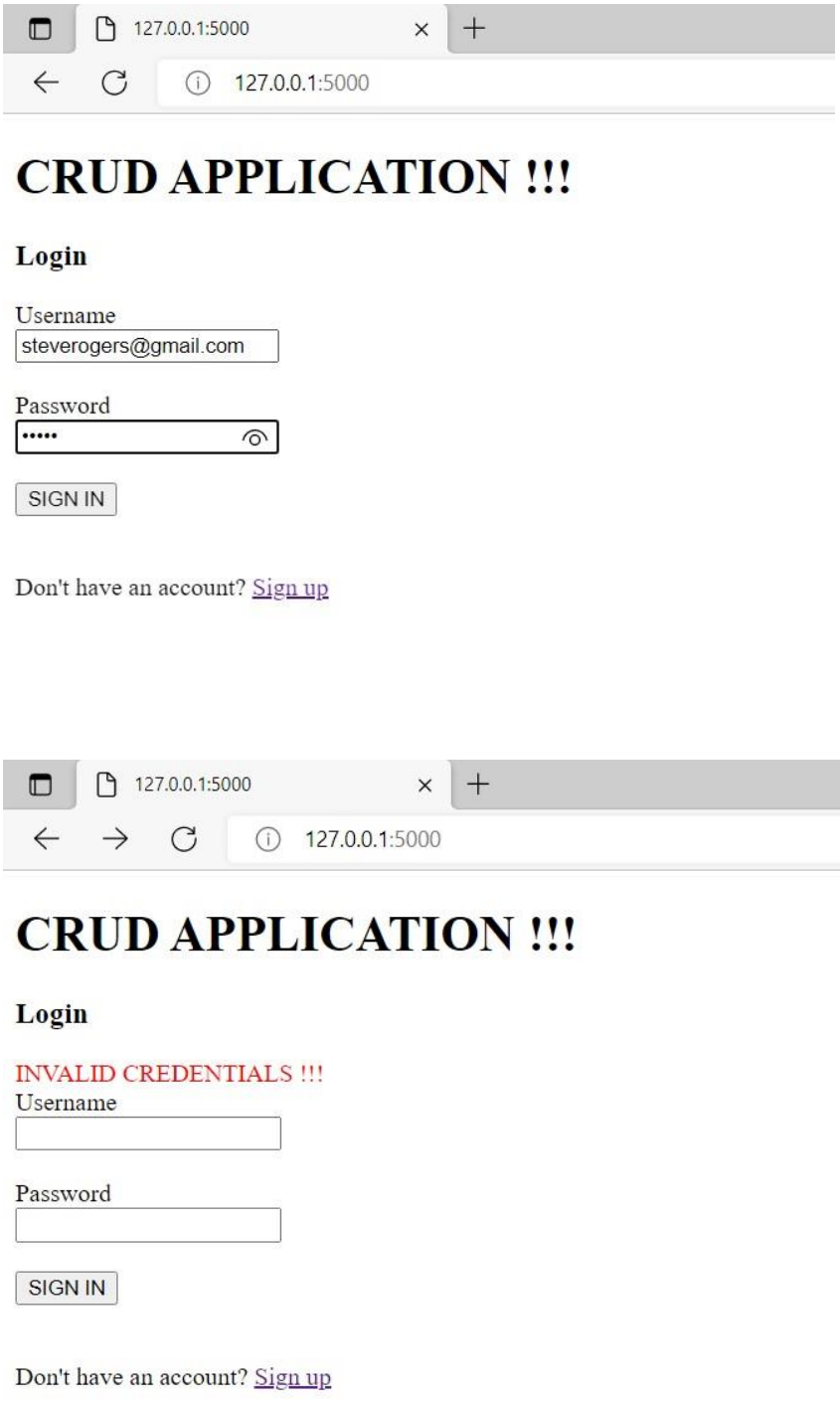
a) SIGNUP





b) LOGIN

- **INVALID**


127.0.0.1:5000
127.0.0.1:5000
CRUD APPLICATION !!!
Login
Username
steverogers@gmail.com
Password
SIGN IN
Don't have an account? Sign up
127.0.0.1:5000
127.0.0.1:5000
CRUD APPLICATION !!!
Login
INVALID CREDENTIALS !!!
Username
Password
SIGN IN
Don't have an account? Sign up

- **VALID**

CRUD APPLICATION !!!

Login

Username
steverogers

Password
•••••

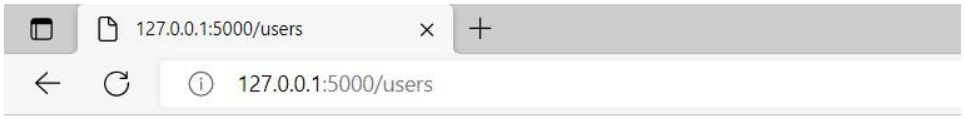
SIGN IN

Don't have an account? Sign up

Welcome, steverogers

2. READ/UPDATE/DELETE

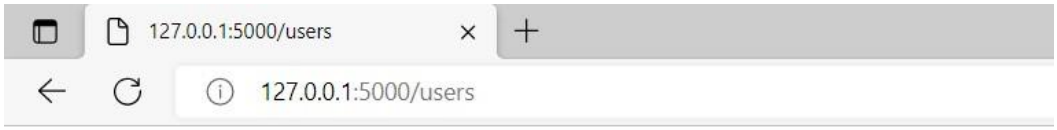
a) READ



Users List

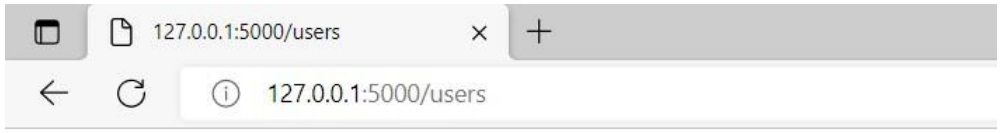
USERNAME	EMAIL	ROLL_NO	OPTIONS
user2	user1@gmail.com	12345	Update Delete
steverogers	steverogers@gmail.com	ABC123	Update Delete

b) DELETE



Users List

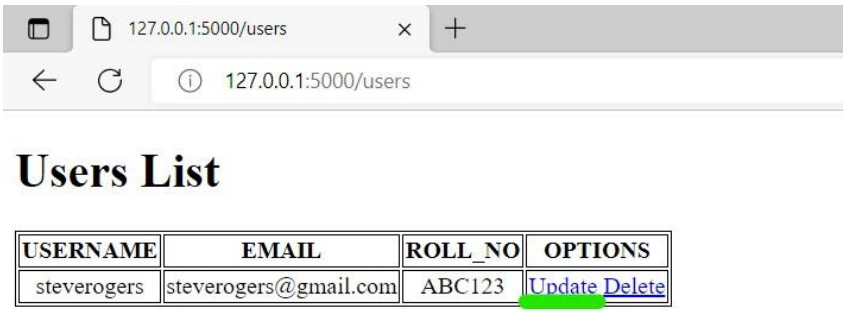
USERNAME	EMAIL	ROLL_NO	OPTIONS
user2	user1@gmail.com	12345	Update Delete
steverogers	steverogers@gmail.com	ABC123	Update Delete

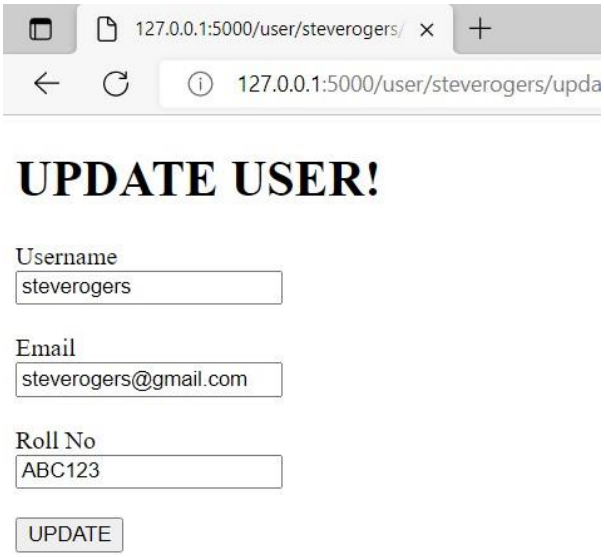


Users List

USERNAME	EMAIL	ROLL_NO	OPTIONS
steverogers	steverogers@gmail.com	ABC123	Update Delete

c) UPDATE


127.0.0.1:5000/users
127.0.0.1:5000/users
Users List
USERNAME
EMAIL
ROLL_NO
OPTIONS
steverogers
steverogers@gmail.com
ABC123
Update Delete


127.0.0.1:5000/user/steverogers/
127.0.0.1:5000/user/steverogers/upda
UPDATE USER!
Username
steverogers
Email
steverogers@gmail.com
Roll No
ABC123
UPDATE

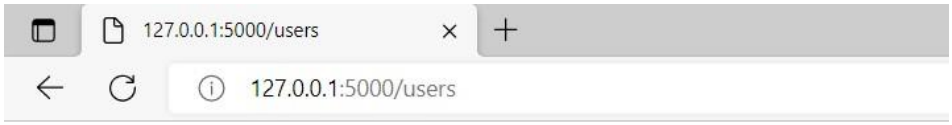
UPDATE USER!

Username
steverogers

Email
steverogers@gmail.com

Roll No
XYZ123

UPDATE



Users List

USERNAME	EMAIL	ROLL_NO	OPTIONS
steverogers	steverogers@gmail.com	XYZ123	Update Delete