Assignment -2

User Table Creation

Project Date	**OCTOBER 2022**
Team ID	**PNT2022TMID08073**
Project Name	**News Tracker Application**

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

Solution:

Table Creation:

Base.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compati ble" content=" IE=edge">
   <meta name="viewport" content="width=device-width, initial-
scale=l.0">
   <title>{% block title %H% endblock %}</title>

   <style>
      @import
url('https://fonts.googleapis.com/css2?family=Michroma&di    splay=swap');
      </style>
   <link rel="stylesheet" href="{{ url_for('static', filename='css/style.css')
}}"/>
</head>
<body>
   <!-- Nav Bar -->
   <nav>
      <div>
         <h3>User Registration Assignment</h3>
      </div>
   </nav>

   {% with messages = get_flashed_messages(wi th_categories=true) %}
      {% ifmessages %}
```

```
        {% for category, message in messages %}
          {% if category == "error" %}
            <div class="flash-div">
              <h4>{{ message }}</h4>
            </div>
          {% else %}
            <div class="flash-div success">
              <h4>{{ message }}</h4>
            </div>
          {% endif %}
        {% endfor %}
      {% endif %}
    {% endwith %}

    <div class=" main-div">
      {% block main %}
      {% endblock %}
    </div>
  </body>
</html>
```

Dashboard.html:

```
{% extends 'base.html' %}

{% block title %}
    Dashboard
{% endblock %}

{% block main %}
    <div class="form-main-div">
      <div class="table-div">
        <h2>Your Details</h2>
        <table>
          <th colspan="2">
            Welcome
          </th>
          <tr>
            <td>Email</td>
            <td>{{ account['EMAIL'] }}</td>
          </tr>
          <tr>
            <td>UserName</td>
            <td>{{ account['USERNAME'] }}</td>
          </tr>
          <tr>
            <td>Register Number</td>
```

```
            <td>{{ account['NUMBER'] }}</td>
          </tr>
          <tr>
            <td>Password</td>
            <td>{{ account['PASSWORD'] }}</td>
          </tr>
        </table>
      </div>
    </div>
{% endblock %}
```

Login.html:

```
{% extends 'base.html' %}

{% block title %}
    Login
{% endblock %}

{% block main %}
    <div class="form-main-div">
      <div  class="form-div">
        <h3>Login</h3>
        <form method="POST">
          <label>Email</label> <br>
          <input class=" inputs" type="text" placeholder=" Enter your
email"  name="email"/>

          <label>Password</label> <br>
          <input class=" inputs" type=" password" placeholder=" Enter your
password" name=" password"/>

          <button  class="submit">Login</button>

          <div>
            <a href="/register">Don't  have an account? Create one</a>
          </div>
        </form>
      </div>
    </div>
{% endblock %}
```

Register.html:

```
{% extends 'base.html' %}
```

```
{% block title %}
    Sign up
{% endblock %}

{% block main %}
    <div class="form-main-div">
        <div class="form-div">
            <h3>Enter all the details</h3>
            <form method="POST">
                <label>Email</label> <br>
                <input class=" inputs" type="text" placeholder=" Enter your email"
name=" email"/>

                <label>Username</label> <br>
                <input class=" inputs" type="text" placeholder=" Enter your
username" name="username"/>

                <label>Register Number</label> <br>
                <input class="inputs" type="number" placeholder=" Enter your
email"  name="number"/>

                <label>Password</label> <br>
                <input class=" inputs" type=" password" placeholder=" Enter your
password" name="password"/>

                <input class="submit"  type="submit"/>

                <div>
                    <a href="/">Already have an account? Login</a>
                </div>
            </form>
        </div>
    </div>
{% endblock %}
```

init.py:

```
from flask import Flask


def create_app():
    app =  Flask(_name_)
```

```
    app.config['SECRET_KEY'] = "PHqtYfAN2v"

    # registering the blue print witg the app
    from .views import blue_print
    app.register_blueprint(blue_print,    url_prefix="/")

    return app
```

View s.py:

```
from flask import Blueprint, redirect, render_template, request, flash
import ibm_db
import re # regular expression

blue_print = Blueprint("blue_print", "_name_")
conn = ibm_db.connect(' DATABASE=bludb; HOSTNAME=54a2f15b-Sc0f-46df-8954-
7e38e612c2bd.clogj3sd0tgtu01qde00.databases.appdomai      n.cloud;PORT=32733;SECU RITY=SSL;SSLS
erverCertificate=DigiCertGlobal RootCA.crt; UID=tyb34892; PWD=QqSGdhZKREQI 1Vrc', '', '')

@blue_print.route('/', methods = ['GET', 'POST'])
def home():
    if request.method == 'POST':
        # getting the data entered by the user
        email = request.form.get('email')
        password = request.form.get('password')

        # validating the inputs
        if len(email) < 10:
            flash(" Email must be atleast 10 characters long", category="error")

        elif len(password) < 6:
            flash(" Password must be atleast 6 characters long", category="error")
```

```
    else:
        # checking whether the user with the email exists in the database
        sql_check_query = "SELECT * FROM user WHERE email = ?"
        stmt = ibm_db.prepare(conn, sql_check_query)
        ibm_db.bind_param(stmt, 1,email)
        ibm_db.execute(stmt)

        account = ibm_db.fetch_assoc(stmt)
        print(account)

        if account:

            # email id exists
            # checking if the password is correct
            if not account['PASSWORD'] == password:
                flash('Invalid password', category='error')

            else:
                # user entered the correct password
                # redirecting the user to the dashboard
                return render_template('dashboard.html', account=account)

        else:
            # email id does not exist in the database
            flash('Email invalid... Try Again', category='error')

    return render_template('login.html')

return render_template('login.html')
```

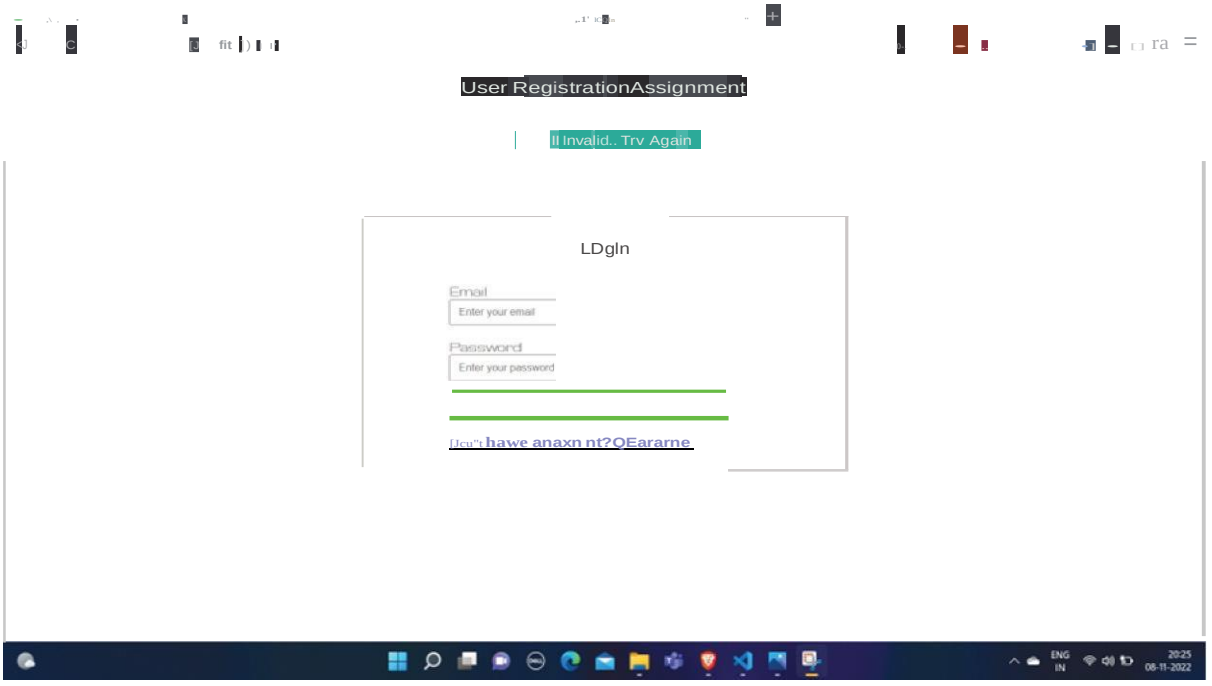
```
@blue_print.route('/register', methods = ['GET', 'POST'])
def register():
    if request.method == 'POST':
        # getting the data entered by the user
        username = request.form.get('username')
        email = request.form.get('email')
        number = request.form.get('number')
        password = request.form.get('password')

        # validating the data entered by the user
        if(len(number) < 12):
            flash("Reg. No must be 12 numbers long", category="error")

        elif not re.match(r"[a-zA-Z]*$", username):
            flash("Use only alphabets in username", category="error")

        elif len(username) < 6:
            flash("Username must be atleast 6 characters long", category="error")

        elif len(password) < 6:
            flash("Password must be atleast 6 characters long", category="error")

        elif len(email) < 10:
            flash("Email must be atleast 10 characters long", category="error")

        else:
            # checking whether the user table contains an entry with the email already
            sql_check_query="SELECT*FROMuserWHEREemail=?"
            stmt = ibm_db.prepare(conn, sql_check_query)
            ibm_db.bind_param(stmt, 1,email)
            ibm_db.execute(stmt)
```

```
            account = ibm_db.fetch_assoc(stmt)

            # email id does not exist in the database
            if not account:
                # inserting the data into the database
                sql_insert_query = "INSERT INTO user (username, email, password, number) VALUES (?, ?,
?, ?)"
                stmt = ibm_db.prepare(conn, sql_insert_query)
                ibm_db.bind_param(stmt, 1, username)
                ibm_db.bind_param(stmt, 2, email)
                ibm_db.bind_param(stmt, 3, password)
                ibm_db.bind_param(stmt, 4, str(number))
                ibm_db.execute(stmt)

                # user data has been inserted into the database
                # showing login page to the user
                flash(' User created successfully! Please Login', category='success')
                return redirect('/')

            else:
                flash('Email id already exists! Try another one', category='error')

        return render_template('register.html')

    return render_template('register.html')

@blue_print.route('/dashboard')
def dashboard():
    return render_template('dashboard.html')
```

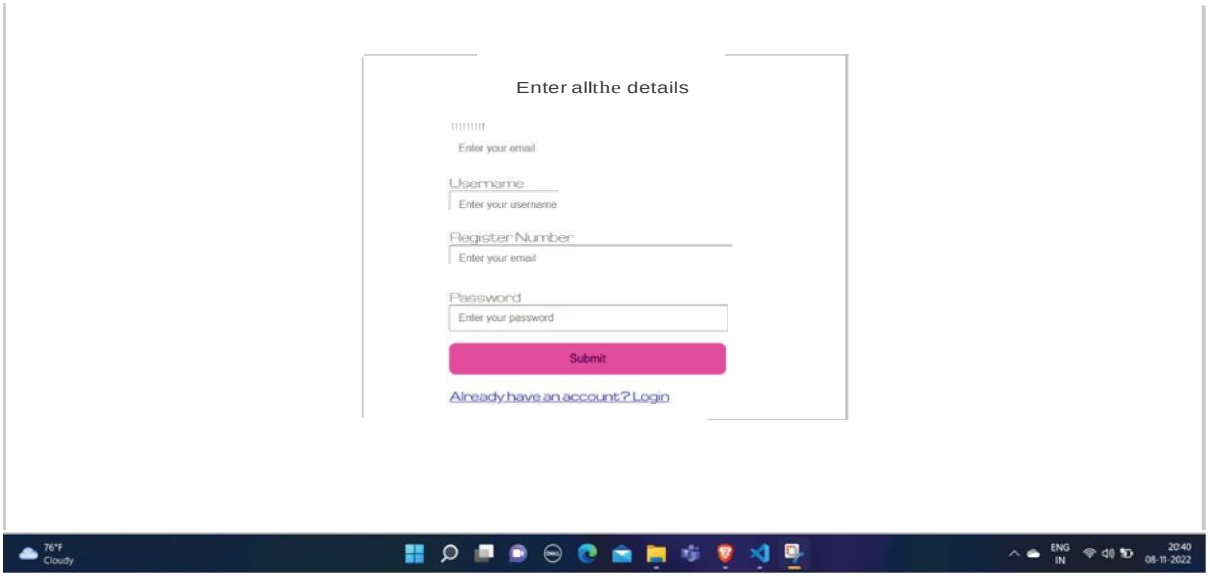
App.py:

```
from registration import create_app

app = create_app()

if __name__ == '__main__':
    app.run(debug=True)
```

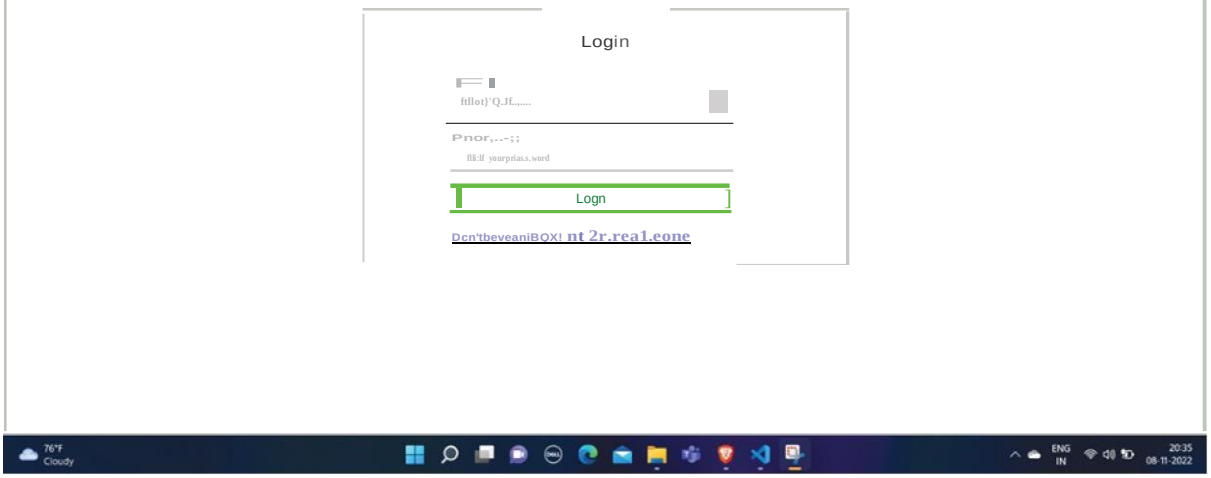
Output:



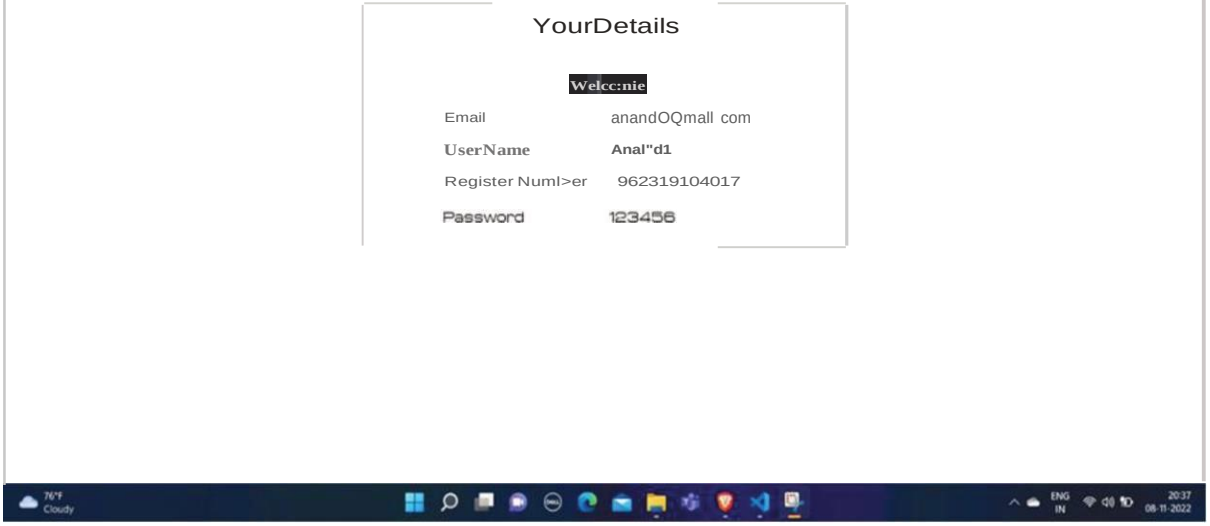
UserReg1strat1 on A ssignmen



User RegistrationAssignment



UserReg strationAssignment

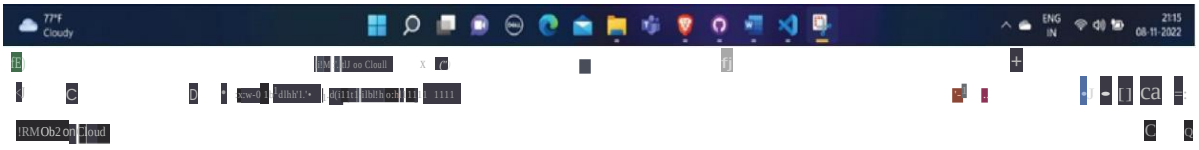



'Untllled • 1
Filter objects
TYB34892
1 SELECT • F u..r;
Syntax assistant
Run all
H5C0fY
Results
Result set 1
Details
0. Fil1tlllt!lt
IOI :I '? [:. /'!
PAIIWORD
AnandmicNel
anandmlc i .tom
%2319104017
;111n@1.mil1 .com
123456
902319104009
Anandh
anand@gmail.com
%2ll9!04CI?
12l-4So
IBM Db2 011 Cloud
• URltltled· t
Filter objects
TYBI4892
l'!i - '> <' '1' 1T a Z 'a
I UPDATE user SET HUI1BU•'91B193378355' IjVERE USERllAl1E•'Anandh';
Find history
S<rlpt
Dat1
Stalus
Runlirno
Untitled • 1
Nov 8, 2022 9:13:14 I
01
9.11111 I
UPDATE user SET NUMBER='918903378355' WHERE USERNAME='Anandh'
0
Ili.Bil» i
un.UUid · 1
Nov 8. 2022 9:04:43 PM
0 1
111.IM& I
SE!.EtT · fUH?
0
B.BIM

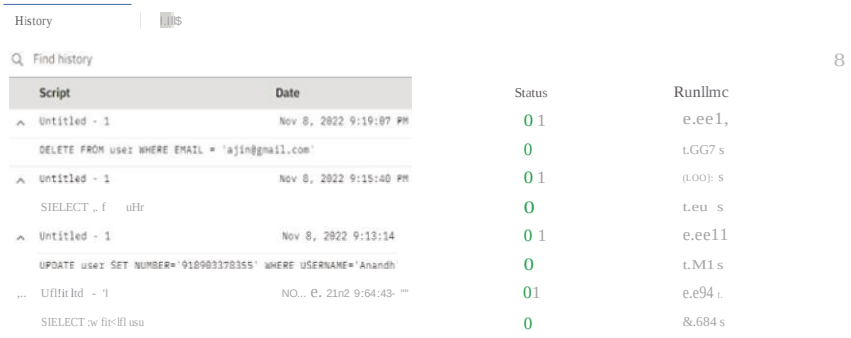

Filter objects
'1 TYlll4892
•ur1tltled ·t
1 SUECT • rROH uti&r;
Syntax assistant
$


History
Results
Result set 1
Details
Filter table
totalc3
USERItA:ME
EMAIL
NUMBER
PASSWORD
aD.1ndmic06@.pnail.com
AAand01t
t6231t104017
ajin@gmail.com
12.145(>
01231'110400'!
anand@gmail.com
918W337BJSS
123451

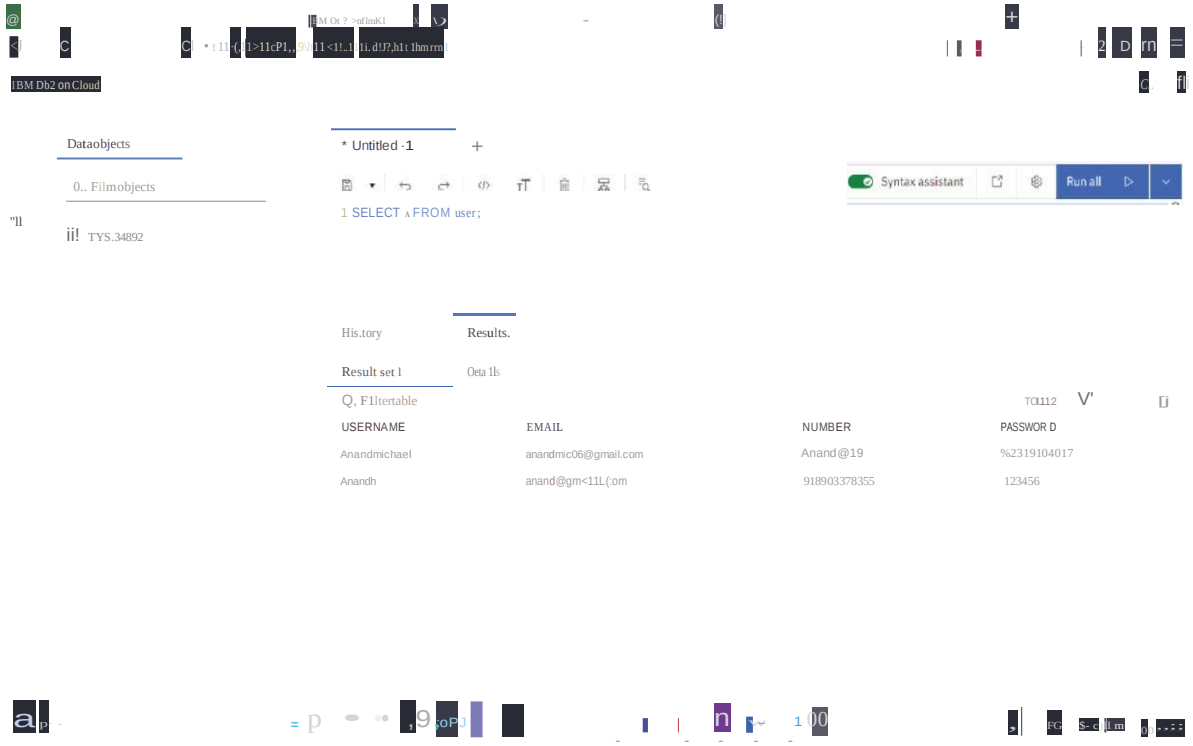
0


77°f
Cloudy
ENG
IN
2115
08-11-2022


Filter objects
TYB34892
'* Untilled ·1
1 DELE1E fROH us&r "11ERE EHAIL • 'ajin@pa:il.c
Syntax assistant
Run all


History
Find history
Script
Date
Status
Runllmc
Untitled - 1
Nov 8, 2022 9:19:07 PM
0 1
e.ee1,
DELETE FROM user WHERE EMAIL = 'ajin@gmail.com'
0
t.GG7 s
Untitled - 1
Nov 8, 2022 9:15:40 PM
0 1
Untitled - 1
Nov 8, 2022 9:13:14
0 1
e.ee11
UPDATE user SET NUMBER='918903178355' WHERE USERNAME='Anandh'
0
t.M1 s
SIELECT :w fit<lfl usu
0
&.684 s

0


IBM Db2 on Cloud
Dataobjects
0.. Filmobjects
"ll
ii! TYS.34892
* Untitled · 1
1 SELECT ∧ FROM user;
Syntax assistant
Run all
His.tory
Results.
Result set l
Oeta 1ls
Q, F1ltertable
USERNAME
EMAIL
NUMBER
PASSWOR D
Anandmichael
anandmic06@gmail.com
Anand@19
%2319104017
Anandh
anand@gm<11L(:om
918903378355
123456