

Assignment -2
User Table Creation

Assignment Date	12 October 2022
Student Name	Binoy Paul
Student Roll Number	962319104034
Maximum Marks	2 Marks

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-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width,
initialscale=1.0">
  <title>{% block title %}{% endblock %}</title>

  <style>
    @import
url('https://fonts.googleapis.com/css2?family=Michroma&display=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(with_categories=true) %}
  {% if messages %}
    {% 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>
            </table>
        </div>
    </div>

```

```

        </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

```

Views.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-5c0f-46df-8954-
7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32733;SECURITY=SSL;SSLS
erverCertificate=DigiCertGlobalRootCA.crt;UID=tyb34892;PWD=Qq5GdhZKREQ!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 * 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)

        # 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.prepares(stmt,
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

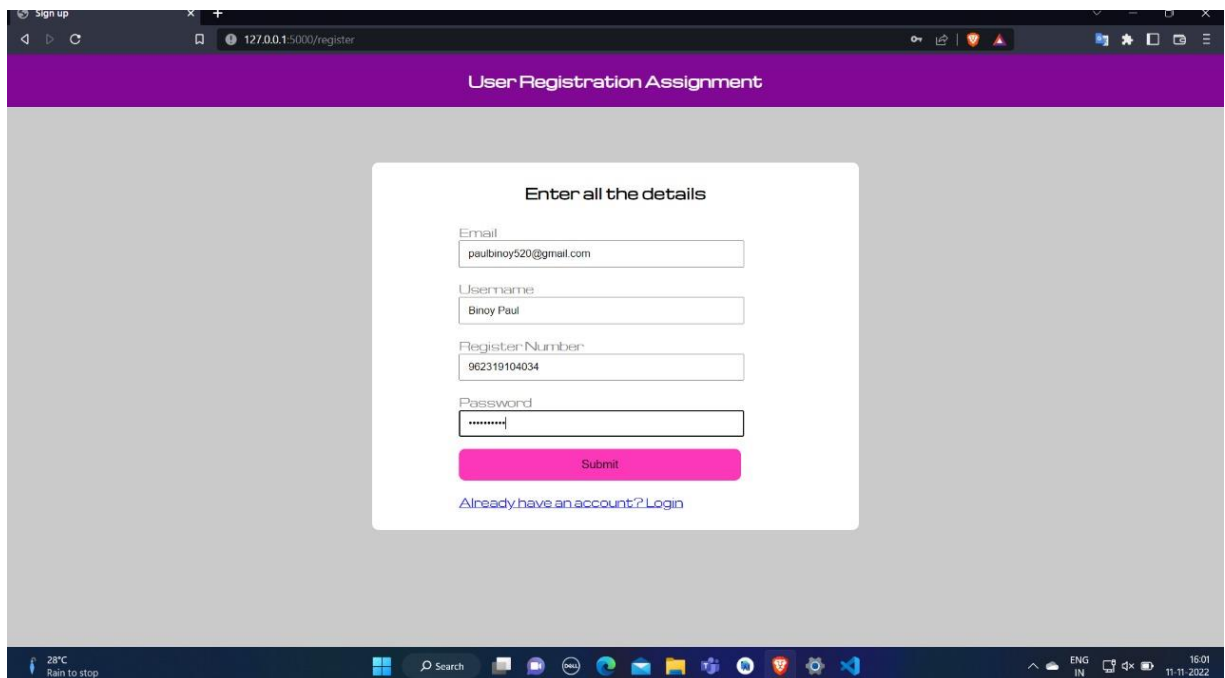
```



```
app = create_app()
```

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

Output:



The screenshot shows a web browser window with the address bar displaying "127.0.0.1:5000/register". The page has a purple header with the text "User Registration Assignment". The main content area is a light gray background with a white registration form in the center. The form is titled "Enter all the details" and contains four input fields: "Email" (with the value "paulbinoy520@gmail.com"), "Username" (with the value "Binoy Paul"), "Register Number" (with the value "962319104034"), and "Password" (with masked characters "*****"). Below the fields is a pink "Submit" button. At the bottom of the form, there is a link that says "Already have an account? Login". The Windows taskbar is visible at the bottom of the screen, showing the time as 11:11, 2022.

sign up

127.0.0.1:5000/register

User Registration Assignment

Enter all the details

Email
paulbinoy520@gmail.com

Username
Binoy Paul

Register Number
962319104034

Password

Submit

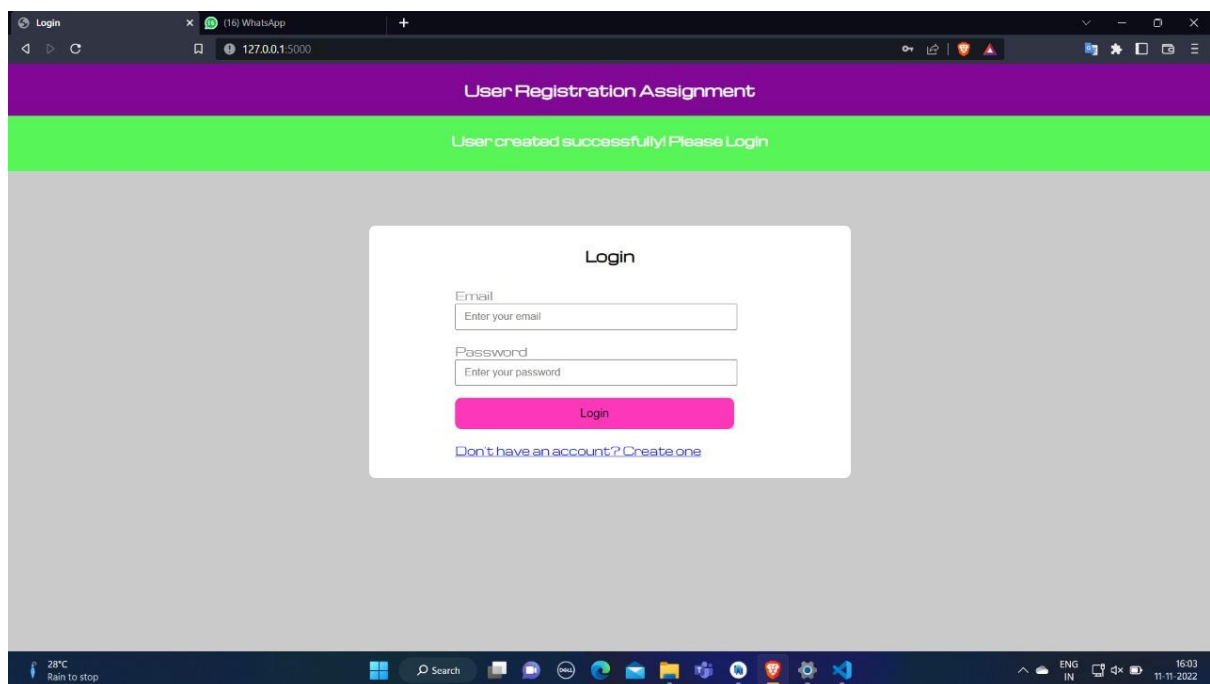
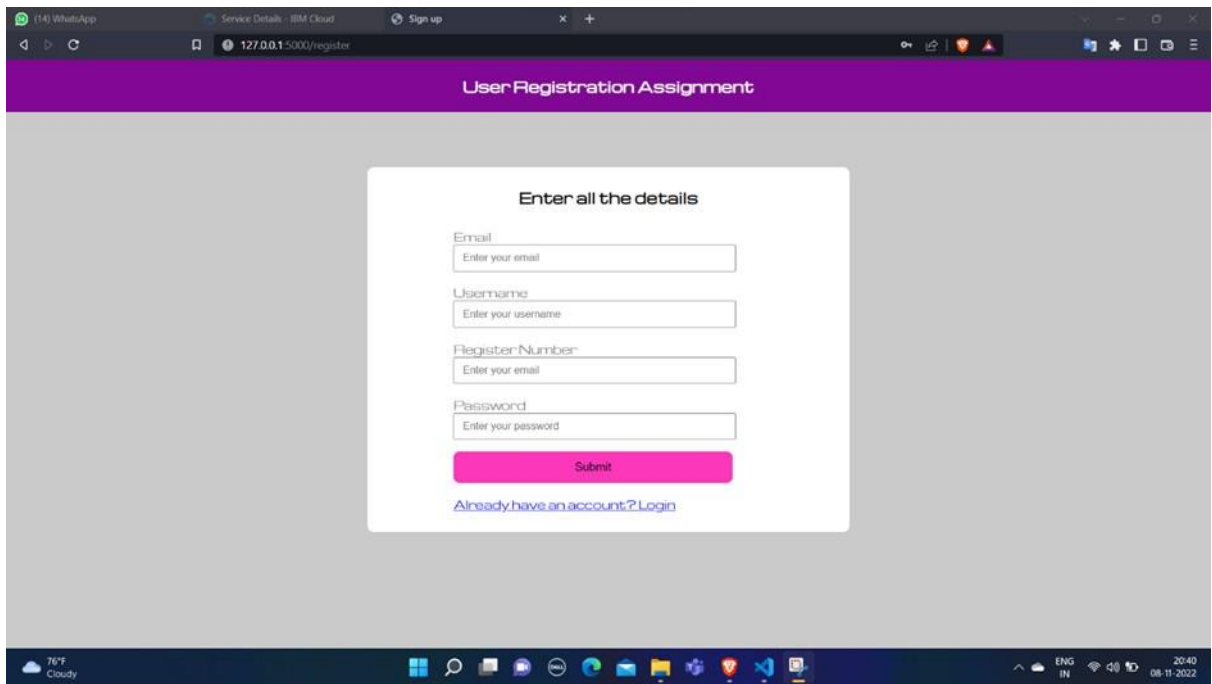
[Already have an account? Login](#)

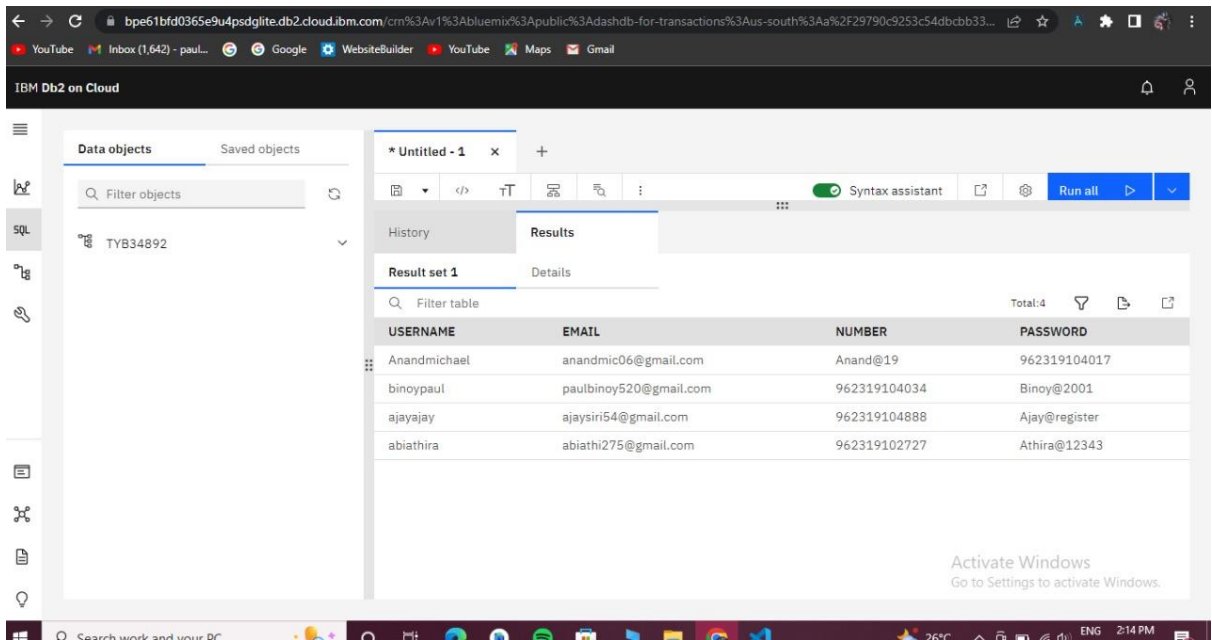
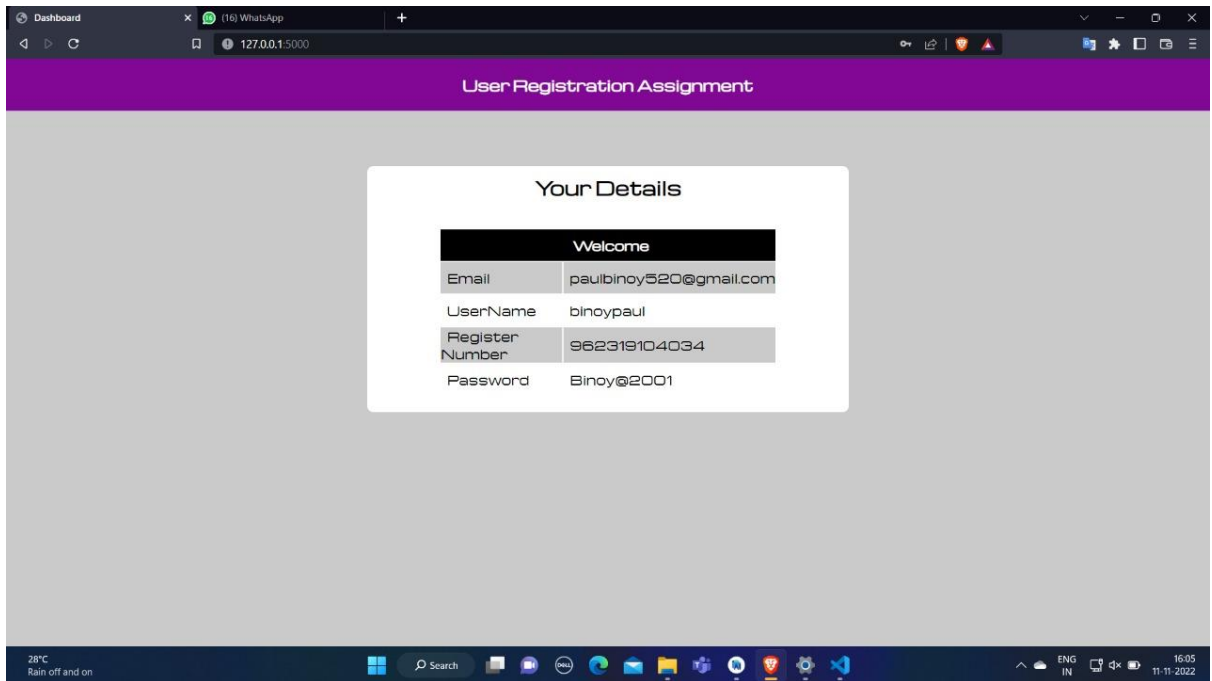
28°C
Rain to stop

Search

ENG
IN

16:01
11-11-2022





IBM Db2 on Cloud

Data objects Saved objects

Filter objects

TYB34892

SQL

*Untitled - 1 x *Untitled - 2 +

Syntax assistant

Run all

```
1 UPDATE user
2 SET password = '123456'
3 WHERE NUMBER = '962319104034';
```

History Results

Result set 1 Details

Filter table

Total: 4

USERNAME	EMAIL	NUMBER	PASSWORD
Anandmichael	anandmic06@gmail.com	Anand@19	962319104017
binoypaul	paulbinoy520@gmail.com	962319104034	123456
ajayajay	ajaysiri54@gmail.com	962319104888	Ajay@register
abiathira	abiathi275@gmail.com	962319102727	Athira@12343

Activate Windows
Go to Settings to activate Windows.

Search work and your PC

30°C

ENG 2:48 PM 11/12/2022

IBM Db2 on Cloud

Data objects Saved objects

Filter objects

TYB34892

SQL

*Untitled - 1 x *Untitled - 2 +

Syntax assistant

Run all

```
1 DELETE FROM User WHERE Username='binoypaul';
```

History Results

Result set 1 Details

Filter table

Total: 3

USERNAME	EMAIL	NUMBER	PASSWORD
Anandmichael	anandmic06@gmail.com	Anand@19	962319104017
ajayajay	ajaysiri54@gmail.com	962319104888	Ajay@register
abiathira	abiathi275@gmail.com	962319102727	Athira@12343

Activate Windows
Go to Settings to activate Windows.

Search work and your PC

28°C

ENG 4:51 PM 11/12/2022

