

Assignment -2
User Table Creation

Assignment Date	12 October 2022
Student Name	Aswini S
Student Roll Number	962319104026
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.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

```

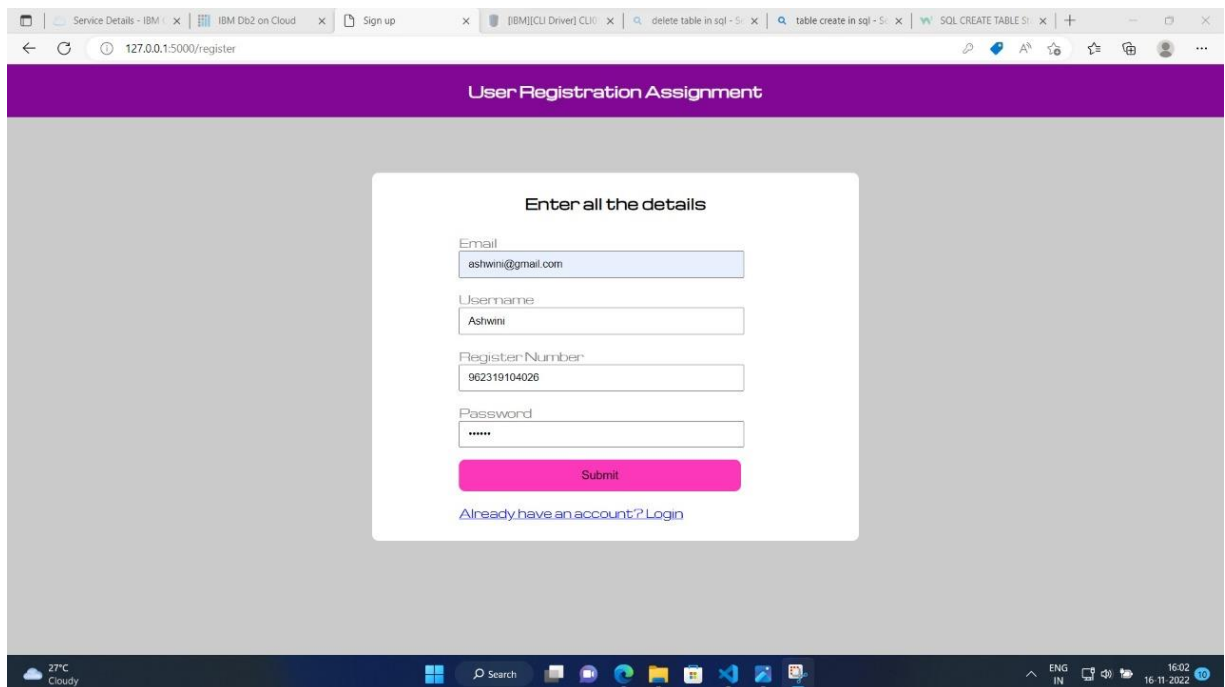


```
app = create_app()
```

```
if __name__ == "__main__":
```

```
    app.run(debug=True)
```

Output:



The screenshot shows a web browser window with multiple tabs. The active tab is titled "127.0.0.1:5000/register". The browser's address bar shows the URL "127.0.0.1:5000/register". The page has a purple header with the text "User Registration Assignment". Below the header, there is a white registration form with the title "Enter all the details". The form contains four input fields: "Email" (with the value "ashwini@gmail.com"), "Username" (with the value "Ashwini"), "Register Number" (with the value "962319104026"), and "Password" (with masked characters "*****"). Below the input fields is a pink "Submit" button. At the bottom of the form, there is a link that says "Already have an account? Login". The browser's taskbar at the bottom shows the Windows logo, a search bar, and several application icons. The system tray on the right shows the date and time as "16:02 16-11-2022".

Service Details - IBM x IBM Db2 on Cloud x Sign up x [IBM] CLI Driver CLI x delete table in sql - S x table create in sql - S x SQL CREATE TABLE S x +

127.0.0.1:5000/register

User Registration Assignment

Enter all the details

Email
ashwini@gmail.com

Username
Ashwini

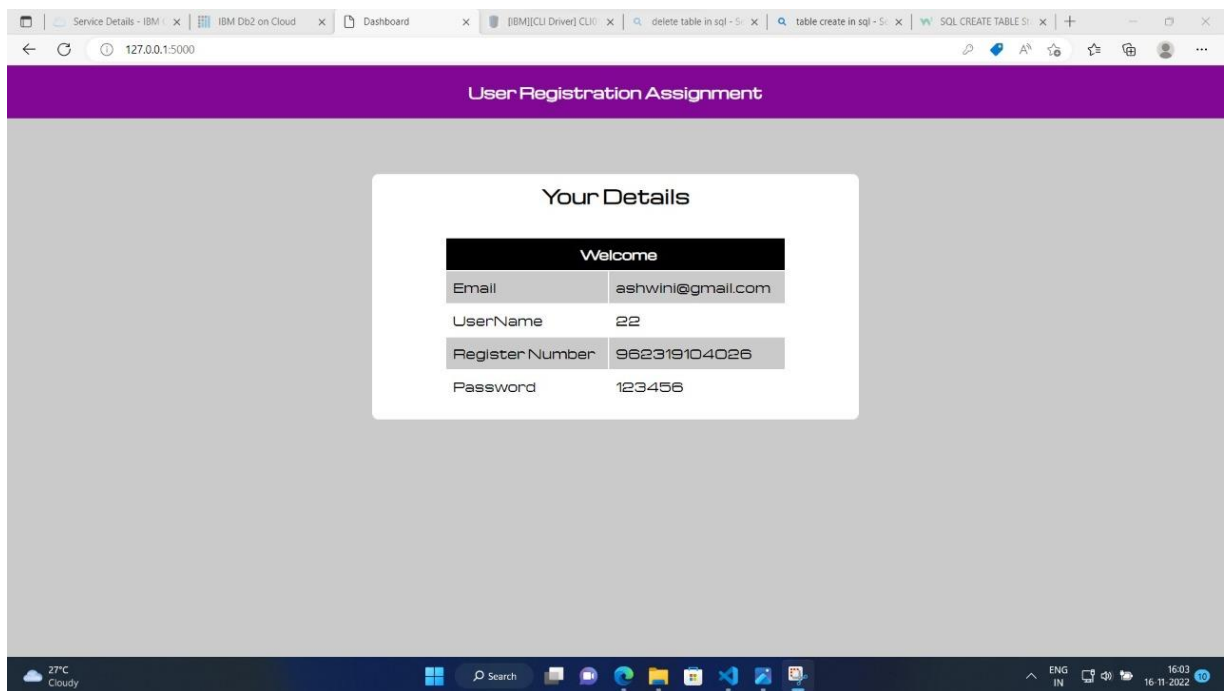
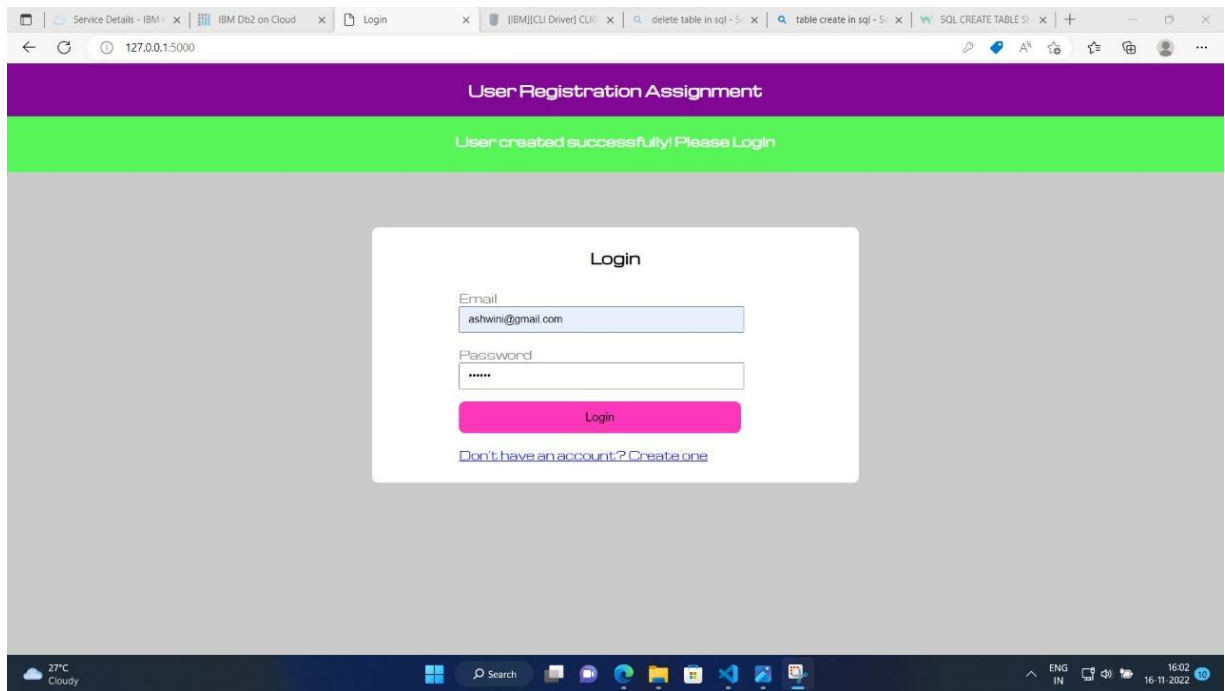
Register Number
962319104026

Password

Submit

[Already have an account? Login](#)

27°C Cloudy Search ENG IN 16:02 16-11-2022



IBM Db2 on Cloud

Data objects

Filter objects

SQL

ZNJ04989

*Untitled - 1

1 SELECT * FROM user;

History

Results

Result set 1

Filter table

Total:4

USERNAME	EMAIL	NUMBER	PASSWORD
JayaHariPrasad	jayahariprasath2001@gmail.com	962319104007	123456
Ashwini	ashwini@gmail.com	962319104026	123456
Bharathi	bharathi@gmail.com	962319104033	123456
Ajithkanna	ajithkanna@gmail.com	962319104010	123456

Market +0.17%

22:14 16-11-2022

IBM Db2 on Cloud

Data objects

Filter objects

SQL

ZNJ04989

*Untitled - 1

1 UPDATE user

2 SET number = '962319104017'

3 WHERE email = 'jeyahariprasath2001@gmail.com';

History

Results

Result set 1

Filter table

Total:8

USERNAME	EMAIL	NUMBER	PASSWORD
22	jayahariprasath2001@gmail.com	962319104047	123456
22	ashwini@gmail.com	962319104026	123456
22	bharathi@gmail.com	962319104033	123456
22	ajithkanna@gmail.com	962319104010	123456
Worldhell	Lifeislikeahell@gmail.com	962319104210	123456
allthebest	abc@gmail.com	962319104245	123456
wonders	def@gmail.com	962319104250	123456
alliswell	xyz@gmail.com	962319104255	123456

27°C Haze

21:54 16-11-2022

IBM Db2 on Cloud

Data objects

Filter objects

SQL

ZNJ04989

*Untitled - 1

```
1 UPDATE user
2 SET number = '962319104007'
3 WHERE email = 'jayahariprasath2001@gmail.com';
```

History

Find history

Script	Date	Status	Runtime
Untitled - 1	Nov 16, 2022 9:57:43 PM	1	0.011 s
UPDATE user SET number = '962319104007' WHERE email = 'jayahariprasath2001@gn...			0.011 s

27°C Haze

IBM Db2 on Cloud

Data objects

Filter objects

SQL

ZNJ04989

*Untitled - 1

```
1 SELECT * FROM user;
```

History

Results

Result set 1

Filter table

USERNAME	EMAIL	NUMBER	PASSWORD
22	jayahariprasath2001@gmail.com	962319104007	123456
22	ashwini@gmail.com	962319104026	123456
22	bharathi@gmail.com	962319104033	123456
22	ajithkanna@gmail.com	962319104010	123456
Worldhell	Lifeislikeahell@gmail.com	962319104210	123456
allthebest	abc@gmail.com	962319104245	123456
wonders	def@gmail.com	962319104250	123456
alliswell	xyz@gmail.com	962319104255	123456

27°C Haze

IBM Db2 on Cloud

Data objects

Filter objects

ZNJ04989

*Untitled - 1

1

DELETE FROM user WHERE email='Lifeislikeahell@gmail.com';

History

Results

Find history

Script	Date	Status	Runtime
Untitled - 1	Nov 16, 2022 10:03:00 PM	1	0.011 s
DELETE FROM user WHERE email='Lifeislikeahell@gmail.com'			0.011 s
Untitled - 1	Nov 16, 2022 10:02:36 PM	1	0.010 s
DELETE FROM user WHERE email='Lifeislikeahell@gmail.com'			0.010 s
Untitled - 1	Nov 16, 2022 10:02:15 PM	1	0.010 s
DELETE FROM user WHERE EMAIL='Lifeislikeahell@gmail.com'			0.010 s
Untitled - 1	Nov 16, 2022 9:58:49 PM	1	0.010 s
SELECT * FROM user			0.010 s
Untitled - 1	Nov 16, 2022 9:57:43 PM	1	0.011 s
UPDATE user SET number = '962319104007' WHERE email = 'jayahariprasath2001@g-			0.011 s

27°C

Haze

Search

ENG

IN

22:03

16-11-2022

IBM Db2 on Cloud

Data objects

Filter objects

ZNJ04989

*Untitled - 1

1

SELECT * FROM user;

History

Results

Filter table

USERNAME	EMAIL	NUMBER	PASSWORD
22	jayahariprasath2001@gmail.com	962319104007	123456
22	ashwini@gmail.com	962319104026	123456
22	bharathi@gmail.com	962319104033	123456
22	ajithkanna@gmail.com	962319104010	123456
allthebest	abc@gmail.com	962319104245	123456
wonders	def@gmail.com	962319104250	123456
alliswell	xyz@gmail.com	962319104255	123456

27°C

Haze

Search

ENG

IN

22:04

16-11-2022

