

## ASSIGNMENT – 2

DIKSHA KRISHNAN

311519104015

[dikshakrish5@gmail.com](mailto:dikshakrish5@gmail.com)

PNT2022TMID27807

### 1. CREATE USER TABLE WITH USER WITH EMAIL, USERNAME, ROLL NUMBER, PASSWORD

#### TABLE CREATION:

The screenshot shows the IBM Db2 on Cloud web interface. The 'Tables' tab is selected, and the 'USERS' table is highlighted. The 'Table definition' panel on the right shows the table structure:

Name	Data type	Nullable	Length	Scale
EMAIL	VARCHAR	Y	50	0
USERNAME	VARCHAR	Y	50	0
Roll_No	INTEGER	Y		0
PASSWORD	VARCHAR	Y	50	0

The screenshot shows the IBM Db2 on Cloud web interface with the 'Data objects' tab selected. The 'My script' panel shows the following SQL commands:

```
1 insert into Users values ('dikshakrish5@gmail.com', 'diksha_krishnan', 4013, 'lockmat123');
2 insert into Users values ('abcd@gmail.com', 'abcd', 4028, 'abcd@123');
3 insert into Users values ('pqrs@gmail.com', 'pqrs', 4012, 'pqrs123');
4 insert into Users values ('ibm@gmail.com', 'ibm', 4009, 'ibm123');
```

The 'History' panel shows the execution results:

Script	Date	Status	Runtime
Untitled - 1	Oct 10, 2022 1:08:03 AM	✓ 4	0.024 s
insert into Users values ('dikshakrish5@gmail.com', 'diksha_krishnan', 4013,...		✓	0.008 s
insert into Users values ('abcd@gmail.com', 'abcd', 4028, 'abcd@123')		✓	0.005 s
insert into Users values ('pqrs@gmail.com', 'pqrs', 4012, 'pqrs123')		✓	0.006 s
insert into Users values ('ibm@gmail.com', 'ibm', 4009, 'ibm123')		✓	0.005 s

The screenshot shows the IBM Db2 on Cloud console interface. The top navigation bar includes 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The 'Tables' tab is selected, displaying a table named 'BKZ99436.USERS'. A 'Back' button is in the top right. Below the table name, there is an 'Export to CSV' button. The table has four columns: EMAIL, USERNAME, ROLLNO, and PASSWORD. The data rows are as follows:

EMAIL	USERNAME	ROLLNO	PASSWORD
abcd@gmail.com	abcd	4028	abcd@123
dikshakrish5@gmail.com	diksha_krishnan	4013	lockmat123
ibm@gmail.com	ibm	4009	ibm123
pqrs@gmail.com	pqrs	4012	pqrs123

## 2. PERFORM UPDATE, DELETE QUERIES WITH USER TABLE

### UPDATE QUERY:

The screenshot shows the IBM Db2 on Cloud console with a SQL script being executed. The script is in a text editor with a 'Run selected' button. Below the script, the 'History' tab is active, showing a table of executed queries.

**SQL Script:**

```

1 insert into Users values ('dikshakrish5@gmail.com', 'diksha_krishnan', 4013, 'lockmat123');
2 insert into Users values ('abcd@gmail.com', 'abcd', 4028, 'abcd@123');
3 insert into Users values ('pqrs@gmail.com', 'pqrs', 4012, 'pqrs123');
4 insert into Users values ('ibm@gmail.com', 'ibm', 4009, 'ibm123');
5
6 update Users set ROLLNO = 4015 where username = 'diksha_krishnan';

```

**History Table:**

Script	Date	Status	Runtime
Untitled - 1	Oct 10, 2022 1:11:42 AM	✓ 1	0.010 s
update Users set ROLLNO = 4015 where username = 'diksha_krishnan'		✓	0.010 s
Untitled - 1	Oct 10, 2022 1:08:03 AM	✓ 4	0.024 s
insert into Users values ('dikshakrish5@gmail.com', 'diksha_krishnan', 4013_		✓	0.008 s
insert into Users values ('abcd@gmail.com', 'abcd', 4028, 'abcd@123')		✓	0.005 s

IBM Db2 on Cloud

bpe61bfd0365e9u4psdglite.db2.cloud.ibm.com/crm%3Av1%3Abluemix%3Apublic%3Adashdb-for-transactions%3Aus-south%3Aa%2Fcb741807a1524b0d9c02c29c1159e97f%3...

GoogleInbox (9) - dikshakri...Welcome To Colab...(475) How to create...VIPBox Sports Strea...

IBM Db2 on Cloud

Load DataLoad HistoryTablesViewsIndexesAliasesMQTsSequencesApplication objects

BKZ99436.USERS

Back

Export to CSV

EMAIL	USERNAME	ROLLNO	PASSWORD
abcd@gmail.com	abcd	4028	abcd@123
dikshakrish5@gmail.com	diksha_krishnan	4015	lockmat123
ibm@gmail.com	ibm	4009	ibm123
pqrs@gmail.com	pqrs	4012	pqrs123

DELETE QUERY:

IBM Db2 on Cloud

bpe61bfd0365e9u4psdglite.db2.cloud.ibm.com/crm%3Av1%3Abluemix%3Apublic%3Adashdb-for-transactions%3Aus-south%3Aa%2Fcb741807a1524b0d9c02c29c1159e97f%3...

GoogleInbox (9) - dikshakri...Welcome To Colab...(475) How to create...VIPBox Sports Strea...

IBM Db2 on Cloud

Data objectsMy script

Filter objects

BKZ99436

\*Untitled - 1

Syntax assistantRun selected

```
1 insert into Users values ('dikshakrish5@gmail.com', 'diksha_krishnan', 4013, 'lockmat123');
2 insert into Users values ('abcd@gmail.com', 'abcd', 4028, 'abcd@123');
3 insert into Users values ('pqrs@gmail.com', 'pqrs', 4012, 'pqrs123');
4 insert into Users values ('ibm@gmail.com', 'ibm', 4009, 'ibm123');
5
6 update Users set ROLLNO = 4015 where username = 'diksha_krishnan';
7
8 delete from Users where ROLLNO = 4009;
```

History

Results

Find history

Script	Date	Status	Runtime
Untitled - 1	Oct 10, 2022 1:18:52 AM	1	0.011 s
delete from Users where ROLLNO = 4009			0.011 s
Untitled - 1	Oct 10, 2022 1:17:08 AM	1	0.007 s
update Users set ROLLNO = 4015 where username = 'diksha_krishnan'			0.007 s
Untitled - 1	Oct 10, 2022 1:16:06 AM	4	0.029 s

The screenshot shows the IBM Db2 on Cloud web interface. The top navigation bar includes 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The 'Tables' tab is selected, displaying a table named 'BKZ99436.USERS'. The table has four columns: 'EMAIL', 'USERNAME', 'ROLLNO', and 'PASSWORD'. There are three data rows. A 'Back' button is in the top right, and an 'Export to CSV' button is in the top right of the table area.

EMAIL	USERNAME	ROLLNO	PASSWORD
abcd@gmail.com	abcd	4028	abcd@123
dikshakrish5@gmail.com	diksha_krishnan	4015	lockmat123
pqrs@gmail.com	pqrs	4012	pqrs123

### 3. CONNECT PYTHON TO DB2

#### CONNECTING TO DB:

##### **reg2.html (Registration Page):-**

```
<html>
<body>
  <center>
    <h3><b>REGISTRATION</b></h3>
    <form action = "http://localhost:5000/reg2" method = "POST">
      <p>Enter Register No.: <input type = "text" name="rollno"/></p>
      <p>Enter Email ID: <input type = "text" name="email"/></p>
      <p>Enter Username: <input type = "text" name="uname"/></p>
      <p>Enter Password: <input type = "password" name="pwd"/></p>
      <p><input type = "submit" value="SUBMIT"/></p>
    </form>
  </center>
</body>
</html>
```

##### **login.html (Login Page):-**

```
<html>
<body>
  <center>
    <h3><b>LOGIN</b></h3>
    <form action = "http://localhost:5000/login" method = "POST">
      <p>Enter Username: <input type = "text" name="uname"/></p>
      <p>Enter Password: <input type = "password" name="pwd"/></p>
      <p><input type = "submit" value="SUBMIT"/></p>
    </form>
  </center>
```

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

### ass2.py:-

```
from flask import Flask,render_template, redirect, url_for, request, session
import ibm_db
import re

app=Flask(__name__)
app.secret_key='a'

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=98538591-7217-4024-b027-
8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;PORT=30875;SECURITY=SSL;SSLSe
rverCertificate=DigiCertGlobalRootCA.crt;UID=bkz99436;PWD=w4b5WZw6Dj9eVXLB;",",")

@app.route('/')
def home():
    return render_template('reg2.html')

@app.route('/login',methods=["GET","POST"])
def login():
    global userid
    msg=" "

    if request.method=="POST":
        username = request.form['uname']
        password = request.form['pwd']
        sql = "SELECT * FROM Users WHERE USERNAME=? AND PASSWORD=?"
        stmt = ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            session['loggedin'] = True
            session['id'] = account['USERNAME']
            userid = account["USERNAME"]
            session['username'] = account["USERNAME"]
            msg = 'Logged in successfully!'
            return redirect(url_for('welcome', username=username))
        else: msg = "Incorrect Username/Password"
    return render_template('login.html', msg = msg)

@app.route('/reg2',methods=["GET","POST"])
def registration():
    msg = " "

    if request.method=="POST":
        username = request.form['uname']
        email = request.form['email']
```

```

password = request.form['pwd']
rollno = request.form['rollno']
sql="SELECT * FROM USERS WHERE USERNAME=?"
stmt=ibm_db.prepare(conn,sql)
ibm_db.bind_param(stmt,1,username)
ibm_db.execute(stmt)
account = ibm_db.fetch_assoc(stmt)
print(account)
if account:
    msg = "Account already exists!"
elif not re.match(r'^[a-zA-Z0-9]+@[a-zA-Z0-9]+\.[a-zA-Z0-9]+$', email):
    msg = "Invalid Email Address."
elif not re.match(r'[A-Za-z0-9]+', username):
    msg = "Username must contain only alphabets and numbers."
else:
    insert_sql = "INSERT INTO USERS VALUES(?,?,?,?)"
    prep_stmt = ibm_db.prepare(conn,insert_sql)
    ibm_db.bind_param(prepare_stmt,1,email)
    ibm_db.bind_param(prepare_stmt,2,username)
    ibm_db.bind_param(prepare_stmt,3,rollno)
    ibm_db.bind_param(prepare_stmt,4,password)
    ibm_db.execute(prepare_stmt)
    msg = "You have successfully registered."
    return render_template('login.html', msg=msg)

```

```

elif request.method == 'POST': msg="Please fill out the form."
return render_template('reg2.html',msg=msg)

```

```

@app.route('/welcome/<username>')
def welcome(username):
    return "Welcome %s!" %username

```

```

if __name__=="__main__":
    app.run(host='0.0.0.0')

```

```

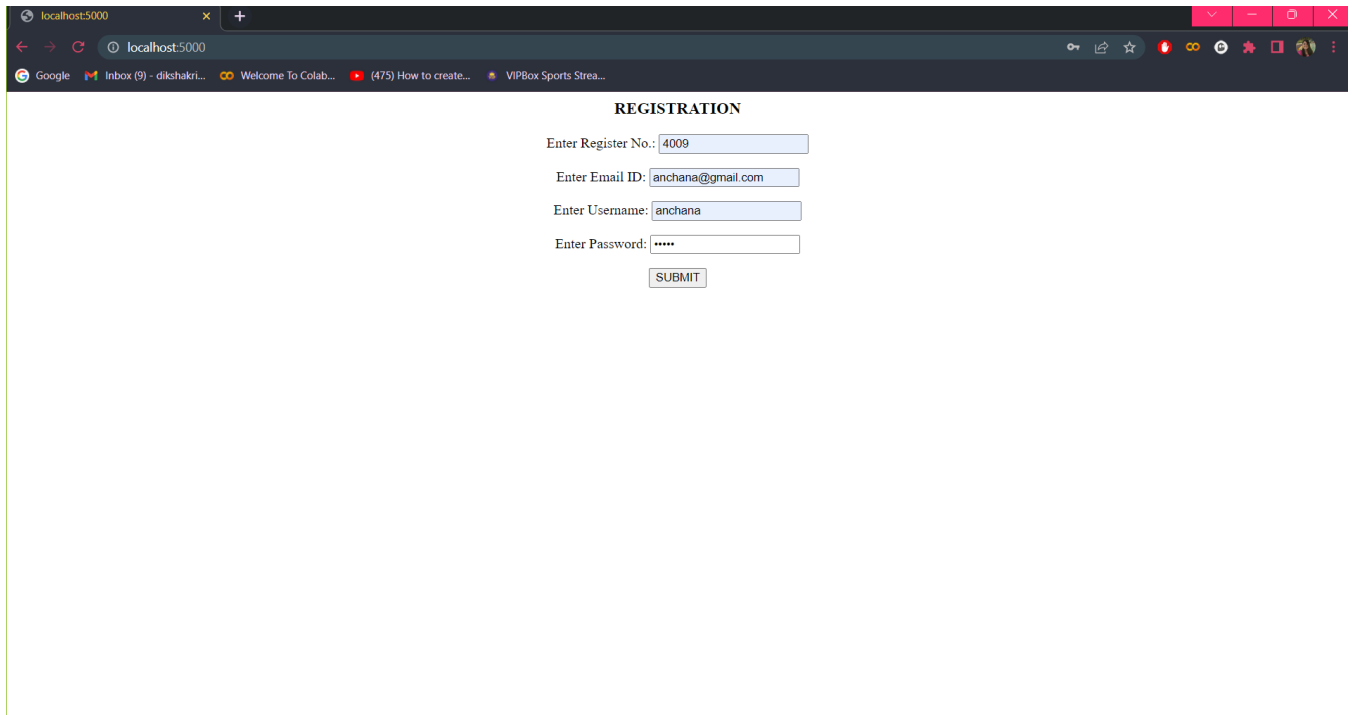
C:\Windows\System32\cmd.exe - py ass2.py
Microsoft Windows [Version 10.0.22621.674]
(c) Microsoft Corporation. All rights reserved.

D:\Diksha\VII Sem\IBM\Assignment Programs\CAD>py ass2.py
* Serving Flask app 'ass2'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://192.168.0.6:5000
Press CTRL+C to quit
127.0.0.1 - - [15/Oct/2022 23:29:57] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [15/Oct/2022 23:29:58] "GET /favicon.ico HTTP/1.1" 404 -
False
127.0.0.1 - - [15/Oct/2022 23:32:59] "POST /reg2 HTTP/1.1" 200 -
False
127.0.0.1 - - [15/Oct/2022 23:35:15] "POST /login HTTP/1.1" 200 -
{'EMAIL': 'anchana@gmail.com', 'USERNAME': 'anchana', 'ROLLNO': 4009, 'PASSWORD': '12345'}
127.0.0.1 - - [15/Oct/2022 23:35:33] "POST /login HTTP/1.1" 302 -
127.0.0.1 - - [15/Oct/2022 23:35:33] "GET /welcome/anchana HTTP/1.1" 200 -

```

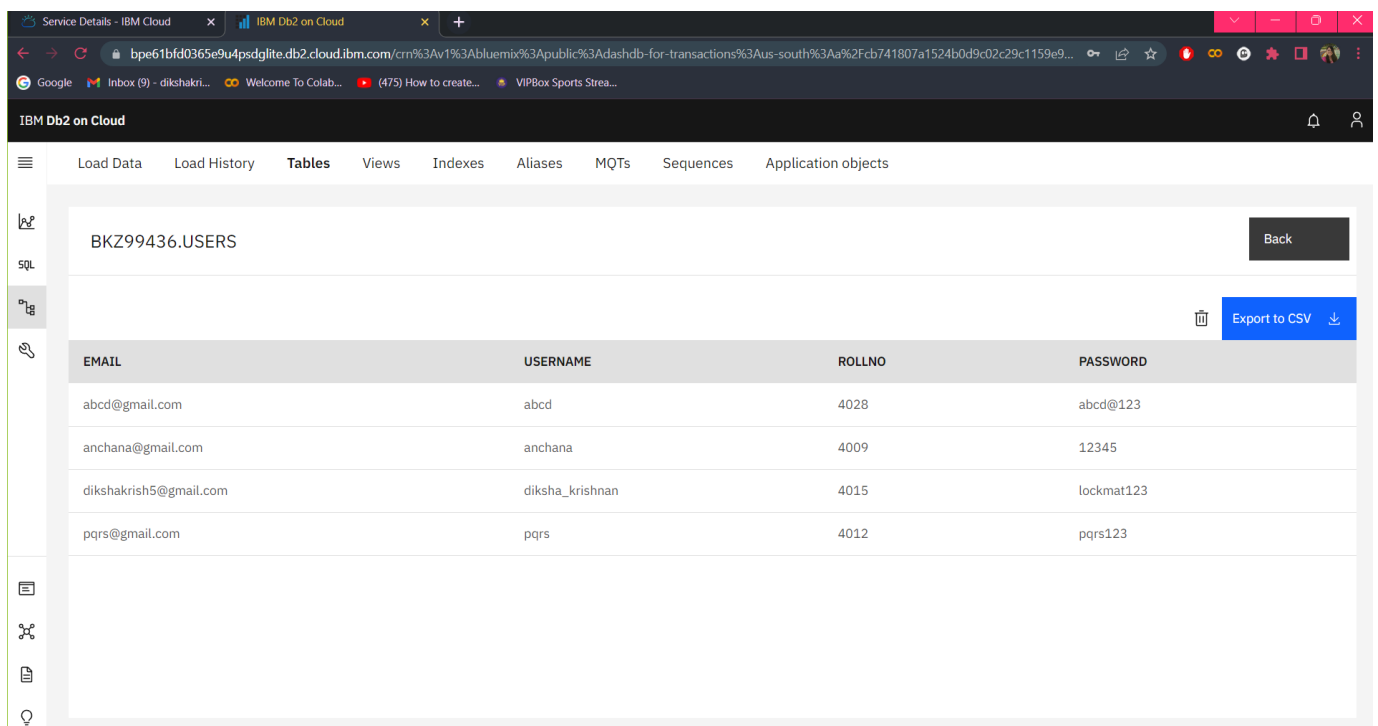
#### 4. CREATE A FLASK APP WITH REGISTRATION PAGE, LOGIN PAGE AND WELCOME PAGE. BY DEFAULT LOAD THE REGISTRATION PAGE ONCE THE USER ENTERS ALL THE FIELDS STORE THE DATA IN DATABASE AND NAVIGATE TO LOGIN PAGE AUTHENTICATE USER USERNAME AND PASSWORD. IF THE USER IS VALID SHOW THE WELCOME PAGE

##### REGISTRATION PAGE:



The screenshot shows a web browser window with the address bar displaying 'localhost:5000'. The page content is a registration form titled 'REGISTRATION'. The form contains four input fields: 'Enter Register No.' with the value '4009', 'Enter Email ID' with the value 'anchana@gmail.com', 'Enter Username' with the value 'anchana', and 'Enter Password' with masked characters '\*\*\*\*'. Below these fields is a 'SUBMIT' button.

##### DATABASE INSERTION:



The screenshot shows the IBM Db2 on Cloud console. The 'Tables' tab is selected, and the table 'BKZ99436.USERS' is displayed. The table has four columns: EMAIL, USERNAME, ROLLNO, and PASSWORD. The table contains four rows of data.

EMAIL	USERNAME	ROLLNO	PASSWORD
abcd@gmail.com	abcd	4028	abcd@123
anchana@gmail.com	anchana	4009	12345
dikshakrish5@gmail.com	diksha_krishnan	4015	lockmat123
pqrs@gmail.com	pqrs	4012	pqrs123

LOGIN PAGE:

localhost/reg2

localhost:5000/reg2

GoogleInbox (9) - dikshakri...Welcome To Colab...(475) How to create...VIPBox Sports Strea...

LOGIN

Enter Username:anchana

Enter Password:\*\*\*\*\*

SUBMIT

WELCOME PAGE:

localhost/welcome/anchana

localhost:5000/welcome/anchana

GoogleInbox (9) - dikshakri...Welcome To Colab...(475) How to create...VIPBox Sports Strea...

Welcome anchana!