

ASSIGNMENT-2

Date	10 October 2022
Team ID	PNT2022TMID30856
Project Name	Personal Expense Tracker
Batch number	B8-2A4E

Question 1: Create User table with username , email , roll number and password.

QUERY:

```
CREATE TABLE USERS(  
empId INTEGER PRIMARY KEY,  
ROLL NO TEXT NOT NULL,  
USERNAME TEXT NOT NULL,  
EMAIL ID TEXT NOT NULL,  
PASSWORD TEXT NOT NULL  
);
```

```
INSERT INTO USERS VALUES(012,'Bhanuupriya S','bhanuupriya@gmail.com','bhanuupriya@012');
```

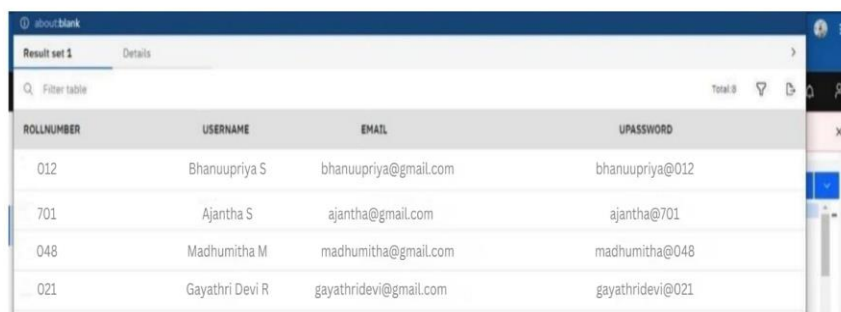
```
INSERT INTO USERS VALUES(701,'Ajantha S','ajantha@gmail.com','ajantha@701');
```

```
INSERT INTO USERS VALUES(048,'Madhumitha
```

```
M','madhumitha@gmail.com','madhumitha@048');INSERT INTO USERS
```

```
VALUES(021,'Gayathri Devi
```

```
R','gayathridevi@gmail.com','gayathridevi@021');SELECT * FROM USERS;
```



The screenshot shows a web application interface with a table displaying user data. The table has four columns: ROLLNUMBER, USERNAME, EMAIL, and UPASSWORD. There are four rows of data, corresponding to the users created in the SQL query. The interface includes a search bar, a filter button, and a details button.

ROLLNUMBER	USERNAME	EMAIL	UPASSWORD
012	Bhanuupriya S	bhanuupriya@gmail.com	bhanuupriya@012
701	Ajantha S	ajantha@gmail.com	ajantha@701
048	Madhumitha M	madhumitha@gmail.com	madhumitha@048
021	Gayathri Devi R	gayathridevi@gmail.com	gayathridevi@021

2. Perform UPDATE, DELETE Queries with user table.

Updation:

UPDATE USERS

SET Email='sbhanuu@gmail.com'

WHERE RollNumber=012;

UPDATE USERS

SET Password='bhanuu@26'

WHERE RollNumber=012,

SELECT * FROM USERS;



ROLLNUMBER	USERNAME	EMAIL	UPASSWORD
012	Bhanupriya S	sbhanuu@gmail.com	bhanuu@26
701	Ajantha S	ajantha@gmail.com	ajantha@701
048	Madhumitha M	madhumitha@gmail.com	madhumitha@048
021	Gayathri Devi R	gayathridevi@gmail.com	gayathridevi@021

Deletion:

DELETE FROM USERS

WHERE RollNumber=021;



ROLLNUMBER	USERNAME	EMAIL	UPASSWORD
012	Bhanupriya S	sbhanuu@gmail.com	bhanuu@26
701	Ajantha S	ajantha@gmail.com	ajantha@701
048	Madhumitha M	madhumitha@gmail.com	madhumitha@048

3. Connect python code to db2.

```
import ibm_db

conn=ibm_db.connect("DATABASE=bludb;AUTHENTICATION=SERVER;HOSTNAME=19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30699;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=lzl34218;PWD=3XYDnCQAbEBZIDLb",",")
print(conn)

print("connection successfull...")
```

Output:



```
Run: main x
C:\Users\HI\PycharmProjects\Assignment2\venv\Scripts\python.exe C:/Users/HI/PycharmProjects/Assignment2/main.py
HostName:19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud
Port:30699
connection successfull...

Process finished with exit code 0
```

4. Create flask app with registration page, login page and welcome page.

```
app.py from flask import Flask, render_template,
request,redirect import ibm_db import dbconn app =
Flask(__name__) @app.route('/')
def register():

    return render_template("register.html")
@app.route('/login') def
login():

    return render_template("login.html")
#welcome page code
@app.route('/welcome')
def welcome():

    if dbconn.con: sql="SELECT *
FROM authentication

    smt= ibm_db.exec_immediate(dbconn.con,sql)
    res = ibm_db.fetch_both(smt) result=[]
    while res != False:
        result.append(res)
        row=len(result)
        res = ibm_db.fetch_both(smt)

    return render_template("welcome.html",r=result,rows=row)
#registration page code
```

```

@app.route("/signup", methods=['POST'])
def signup(): if request.method == 'POST':

    name = request.form.get('name')
    email = request.form.get('email')
    pwd = request.form.get('pwd') roll
    = request.form.get('roll')

    sql = "INSERT INTO authentication (username,email,password,rollnumber) VALUES
    ('{0}','{1}','{2}','{3}')"

    res = ibm_db.exec_immediate(dbconn.con, sql.format(name, email, pwd, roll))

    if      sql:      return
    redirect("/login")    else:
    return redirect("/") else:
    print("Could'nt store anything...")

#login page code

@app.route("/signin", methods=['POST']) def
signin():

    if request.method == 'POST':

        email = request.form.get('email') pwd
        = request.form.get('pwd')

    sql = "SELECT * FROM authentication WHERE EMAIL='{0}' AND PASSWORD='{1}'"

    smt = ibm_db.prepare(dbconn.con, sql.format(email,pwd)) ibm_db.execute(smt)

    res=ibm_db.fetch_assoc(smt)

    if res:

        return redirect("/welcome")

    else: return redirect("/login") else:
    print("Could'nt store anything...")

#update page code

@app.route("/update/<string:id>", methods=['POST','GET']) def
update(id):

    if request.method == 'POST':

```

```

name = request.form.get('name')

email = request.form.get('email')

pwd = request.form.get('pwd') roll
= request.form.get('roll')

sql="UPDATE authentication SET username='{0}',email='{1}',password='{2}',rollnumber='{3}'
WHERE      id='{4}'"      res      =      ibm_db.exec_immediate(dbconn.con,

sql.format(name,email,pwd,roll,id))

if sql:

    return redirect("/welcome")

else: return redirect("/") else:

    print("Could'nt store anything...") sql="SELECT *
FROM      authentication      WHERE      id='{0}'"

smt=ibm_db.exec_immediate(dbconn.con,sql.format(id))

res      =      ibm_db.fetch_both(smt)      return

render_template("update.html",update=res)

#delete page code

@app.route("/delete/<string:id>",methods=['POST','GET']) def
delete(id):

    sql="DELETE FROM authentication WHERE id='{0}'"

smt=ibm_db.exec_immediate(dbconn.con,sql.format(id))

return redirect("/welcome") if __name__ == '__main__':

    app.run(debug=True)

```

Output:

Register

Username

Enter the name

Email

Enter the email

Password

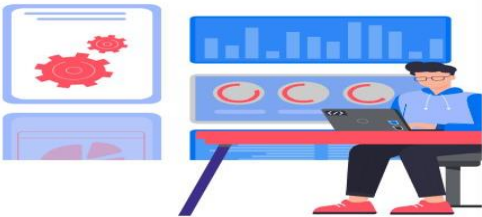
Enter the Password

Roll number

Enter the roll number

sign up

You have an account
[login](#)



Login

Email

Enter the email

Password

Enter the password

login

Don't have an account

[Register](#)



User details

{% for x in range(0,rows, 1):%} {%for y in range(0,5,1):%} {%endfor%} {%endfor%}

UserID	Username	Email	Password	Roll number	Update	Delete
{{r[x][y]}}	Edit	Delete				