

Assignment -2

Assignment Date	19 November 2022
Team ID	PNT2022TMID32569
Project Name	Personal Expense Tracker Application
Maximum Marks	2 Marks

Question-1:

Create User table with email, username, roll number, password.

Insert value into table

The screenshot displays the IBM Db2 on Cloud web interface. The top navigation bar includes options like 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The 'Tables' tab is active, showing a search bar and a 'New table' button. Below this, a table lists existing tables, with 'EXAMPLE' selected. The 'Table definition' panel on the right shows the schema for 'EXAMPLE' with columns: EMAIL (CHAR, Nullable Y, Length 5, Scale 0), USERNAME (CHAR, Nullable Y, Length 5, Scale 0), roll no (INTEGER, Nullable Y, Length 10, Scale 0), and PASSWORD (CHAR, Nullable Y, Length 5, Scale 0). A 'View data' button is visible at the bottom of the definition panel.

Name	Data type	Nullable	Length	Scale
EMAIL	CHAR	Y	5	0
USERNAME	CHAR	Y	5	0
roll no	INTEGER	Y	10	0
PASSWORD	CHAR	Y	5	0

IBM Db2 on Cloud

Data objects Saved objects

Filter objects

KCD96709

*Untitled - 1

```
1 insert into EXAMPLE values('1','vel','5105','vel');
2 insert into EXAMPLE values('2','sri','50965','sri');
3 insert into EXAMPLE values('3','srii','5096','srii');
4 insert into EXAMPLE values('4','yogi','5112','yogi');
```

History

Script	Date	Status	Runtime
Untitled - 1	Nov 19, 2022 2:10:19 AM	4	0.022 s
insert into EXAMPLE values('1','vel','5105','vel')			0.007 s
insert into EXAMPLE values('2','sri','50965','sri')			0.005 s
insert into EXAMPLE values('3','srii','5096','srii')			0.005 s
insert into EXAMPLE values('4','yogi','5112','yogi')			0.005 s

IBM Db2 on Cloud

Load Data Load History Tables Views Indexes Aliases MQTs Sequences Application objects

KCD96709.EXAMPLE

Back

Export to CSV

EMAIL	USERNAME	roll no	PASSWORD
1	vel	5105	vel
2	sri	50965	sri
3	srii	5096	srii
4	yogi	5112	yogi

Question-2:

Perform UPDATE, DELETE Queries with user table

Update the table

The screenshot displays the IBM Db2 on Cloud web interface. The top section shows the 'Data objects' tab with a search bar and a list of objects, including 'KCD96709'. The 'SQL' tab is active, showing a query editor with the following SQL code:

```
1 update EXAMPLE
2 set username = 'yo'
3 where username= 'yogi';
4
```

The 'History' tab is also visible, showing a table of executed queries:

Script	Date	Status	Runtime
untitled - 1	Nov 19, 2022 2:28:05 AM	1	0.016 s
update EXAMPLE set username = 'yo' where username= 'yogi'			0.016 s

The bottom section shows the 'Tables' tab with a table named 'KCD96709.EXAMPLE'. The table has four columns: EMAIL, USERNAME, roll no, and PASSWORD. The data is as follows:

EMAIL	USERNAME	roll no	PASSWORD
1	vel	5105	vel
2	sri	50965	sri
3	srii	5096	srii
4	yo	5112	yogi

Delete a row in the table

The screenshot shows the IBM Db2 on Cloud console. On the left, under 'Data objects', the table 'KCD96709' is selected. The main area shows a SQL script in a text editor:

```
1 delete from EXAMPLE
2 where username= 'yo';
3
```

Below the editor, the 'History' tab is active, showing a table of executed scripts:

Script	Date	Status	Runtime
Untitled - 1	Nov 19, 2022 2:21:39 AM	✓ 1	0.009 s
delete from EXAMPLE where username= 'yo'		✓	0.009 s
Untitled - 1	Nov 19, 2022 2:28:05 AM	✓ 1	0.016 s

The screenshot shows the IBM Db2 on Cloud console displaying the table 'KCD96709.EXAMPLE'. The table has four columns: EMAIL, USERNAME, roll no, and PASSWORD. The data is as follows:

EMAIL	USERNAME	roll no	PASSWORD
1	vel	5105	vel
2	sri	50965	sri
3	srii	5096	srii

Question 3:

Connect python code to db2

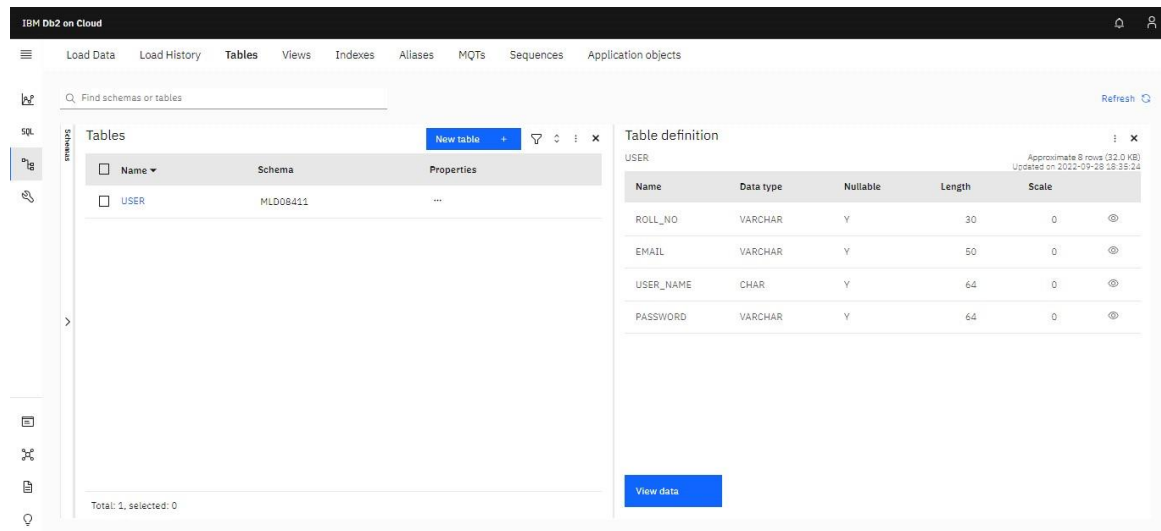
Program:

```
from flask import Flask
import ibm_db
app=Flask(__name__)
app.secret_key='a' try:
    conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=2f3279a5-73d1-4859-
88f0a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;PORT=30756;SECURI
TY
=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=mld08411;PWD=mCn9EnGnLG
Gbg2JH", "", "") except:
    print("Unable to connect: ",ibm_db.conn_error())
@app.route("/") def
dashboard():
    return "welcome" if
__name__=='__main__':
app.run(debug=True)
```

Question 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 username and password. If the user is valid show the welcome page.

Database:



Program:

User.py

```
from flask import Flask, flash, render_template, request, redirect, url_for, session
import ibm_db
app=Flask(__name__)
app.secret_key='a'
try:
    conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=2f3279a5-73d1-4859-
88f0a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;PORT=30756;SECUTY
=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=mld08411;PWD=mCn9EnGnLG
Gbg2JH","","")
except:
    print("Unable to connect: ",ibm_db.conn_error())

@app.route("/")
def dash():
    return render_template('register.html',msg=" ")
@app.route("/register",methods=['GET','POST'])
def register():
    error = None
    if request.method=='POST':
        email=request.form['email']
        password=request.form['password']
        sql="SELECT * FROM user WHERE email=?"
```

```

    prep_stmt=ibm_db.prepare(conn,sql)
    ibm_db.bind_param(prepare_stmt,1,email)
    ibm_db.execute(prepare_stmt)
    account=ibm_db.fetch_assoc(prepare_stmt)
    print(account)
    if account:
        error="Account already exists! Log in to continue !"
    else:
        insert_sql="INSERT INTO user values(?,?,?,?)"
        prep_stmt=ibm_db.prepare(conn,insert_sql)
        ibm_db.bind_param(prepare_stmt,2,email)
        ibm_db.bind_param(prepare_stmt,4,password)
        ibm_db.execute(prepare_stmt)
        flash(" Registration successfull. Log in to continue !")
    else:
        pass
        return render_template('login.html',error=error)

@app.route('/login',methods=['GET','POST'])
def login():
    error = None
    if request.method=='POST':
        email=request.form['email']
        password=request.form['password']
        sql="SELECT * FROM user WHERE user_name=? 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['USER_NAME']
            session["username"]=account["USER_NAME"]
            flash("Logged in successfully!")
            return redirect(url_for("welcome_page"))
        else:
            error="Incorrect username / password"

```

```
        return render_template('login.html',error=error)
@app.route('/welcome')
def welcome_page():
    return render_template("welcome.html",user=session['id'])
if __name__=='__main__':
    app.run(debug=True)
```

register.html

```
<html>
<head>
    <title>Registration page</title>
    <link rel="stylesheet" href="{ {url_for('static', filename='style.css')}}">
</head>
<body>
    <div class="center"> <br /><br />
    <form action="/register" method="POST">
        <h2>Student Registration Portal</h2> <br /><br />
        <p> Enter Email ID:</p>
        <input type="email" name="email"/>
        <p> Enter Username:</p>
        <input type="text" name= "username" />
        <p> Enter Password:</p>
        <input type="password" name="password"/>
        <p> Enter Roll number:</p>
        <input type="text" name="rollnumber"/>
        <br><br><br>
        <input type="submit" value="submit" />
    </form>
    </div>
</body>
</html>
```


login.html

```
<html>
  <head>
    <title>Login page</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
  </head>
  <body>
    {% if error %}
      <p><strong style="color:red">Error</strong>: {{ error }}</p>
    {% endif %}
    {% with messages = get_flashed_messages() %}
      {% if messages %}
        {% for message in messages %}
          <p style="color:green">{{ message }}</p>
        {% endfor %}
      {% endif %}
    {% endwith %}
    <div class="center"> <br /><br />
    <form action="/login" method="POST">
      <h1>Login</h1> <br /><br />
      <p> Enter Username:</p>
      <input type="text" name="username" />
      <p> Enter Password:</p>
      <input type="password" name="password"/>
      <br><br><br>
      <input type="submit" value="submit" />
    </form>
  </div>
</body>
</html>
```

welcome.html

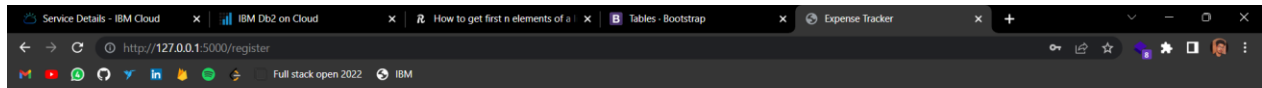
```
<html>
  <head>
    <title>Login page</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
  </head>
```

```
<body>
{% with messages = get_flashed_messages() %}
    {% if messages %}
        {% for message in messages %}
            <p style="color:green">{{ message }}</p>
        {% endfor %}
    {% endif %}
{% endwith %}
<br /><br />
<center>
    <h2 > Welcome {{user}}</h2>
</center>
</body>
</html>
```

style.css

```
.center{
margin: auto;
width: 15%;
} h2{
text-align:center;
} input{ border-
radius:5px;
padding:7px;
}
```

Output:



REGISTER

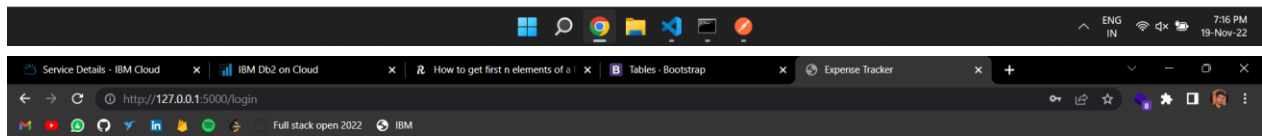
Name

Email

Password

Confirm Password

Already have an account? [Login Here](#)



LOGIN

Email Address

Password

Don't have an account? [Register Here](#)



Welcome ,Sridharan S !