

Assignment -2

DB and Flask

Assignment Date	12 October 2022
Student Name	Mr P Sathiyananth
Student Roll Number	621319104051
Maximum Marks	2 Marks

Question-1:

Create user table with email, username, roll number, password

The screenshot shows a database IDE interface. On the left, there's a sidebar with 'Data objects' and 'My script'. The main area displays a SQL script for creating a table named 'user'. The script is as follows:

```

1 create table user(
2     S1_No int generated by default as identity not null,
3     User_Name varchar(200) not null,
4     User_Email varchar(200) not null,
5     Roll_No int not null,
6     password varchar(200),
7     primary key(S1_No)
8 );

```

Below the script editor, there's a 'History' tab showing a table with columns: Script, Date, Status, and Runtime. The table contains two entries:

Script	Date	Status	Runtime
Assignment-2	Oct 18, 2022 3:11:13 PM	✓ 1	0.111 s
create table user(S1_No int generated by default as identity n...		✓	0.111 s

Insert Values:

The screenshot shows the same database IDE interface, but now with a SQL script for inserting values into the 'user' table. The script is as follows:

```

5 Roll_No int not null,
6 Password varchar(200),
7 primary key(S1_No)
8 );
9
10 insert into user(User_Name,User_Email,Roll_No>Password) values('Sathiyananth','sakthiyananth@gmail
11 insert into user(User_Name,User_Email,Roll_No>Password) values('Karthikeyan','karthi@gmail.com',22
12 insert into user(User_Name,User_Email,Roll_No>Password) values('Naveen','Naveen@gmail.com',37,'nav
13 insert into user(User_Name,User_Email,Roll_No>Password) values('Salman','salman@gmail.com',47,'sal

```

Below the script editor, the 'History' tab shows a table with columns: Script, Date, Status, and Runtime. The table contains four entries:

Script	Date	Status	Runtime
insert into user(User_Name,User_Email,Roll_No>Password) value...		✓	0.007 s
Assignment-2	Oct 18, 2022 3:18:38 PM	✓ 1	0.010 s
insert into user(User_Name,User_Email,Roll_No>Password) value...		✓	0.010 s
Assignment-2	Oct 18, 2022 3:18:25 PM	✓ 1	0.099 s
create table user(S1_No int generated by default as identity...		✓	0.099 s

User table:

The screenshot shows the IBM Db2 on Cloud interface. At the top, there's a navigation bar with 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The 'Tables' tab is selected. Below the navigation bar, there's a search bar with 'LHK92942.USER' and a 'Back' button. To the right of the search bar, there's an 'Export to CSV' button. Below the search bar, there's a table with the following data:

SL_NO	USER_NAME	USER_EMAIL	ROLL_NO	PASSWORD
1	Sathiyananth	sakthiyananth@gmail.com	51	Ibm1234
2	Karthikeyan	karthi@gmail.com	22	karthi@22
3	Naveen	Naveen@gmail.com	37	naveen@37
4	Salman	salman@gmail.com	47	salman@47

Question-2:

Perform update & delete queries with the table.

Update values:

The screenshot shows the IBM Db2 on Cloud interface. On the left, there's a 'Data objects' panel with a search bar and a list of objects. The 'My script' panel is active, showing a SQL script. The script contains the following queries:

```
7 primary key (SL_NO)
8 );
9
10 insert into user(User_Name,User_Email,Roll_No>Password) values('Sathiyananth','sakthiyananth@gmail
11 insert into user(User_Name,User_Email,Roll_No>Password) values('Karthikeyan','karthi@gmail.com',22
12 insert into user(User_Name,User_Email,Roll_No>Password) values('Naveen','Naveen@gmail.com',37,'nav
13 insert into user(User_Name,User_Email,Roll_No>Password) values('Salman','salman@gmail.com',47,'sal
14
15 update user set User_Name = 'abinash' where Roll_No=47;
```

Below the script, there's a 'History' panel showing the execution history of the script. The history table has the following data:

Script	Date	Status	Runtime
Assignment-2	Oct 18, 2022 3:25:51 PM	✓ 1	0.008 s
update user set User_Name = 'abinash' where Roll_No=47		✓	0.008 s
Assignment-2	Oct 18, 2022 3:21:22 PM	✓ 3	0.022 s
insert into user(User_Name,User_Email,Roll_No>Password) value...		✓	0.009 s
insert into user(User_Name,User_Email,Roll_No>Password) value...		✓	0.006 s

Delete values:

The screenshot shows the IBM Db2 on Cloud SQL editor interface. The left sidebar displays the 'Data objects' tab with a search filter 'LHK92942'. The main editor area shows a script titled '* Assignmenten...' with the following SQL statements:

```
9
10 insert into user(User_Name,User_Email,Roll_No>Password) values('Sathiyananth','sakthiyananth@gmail
11 insert into user(User_Name,User_Email,Roll_No>Password) values('Karthikeyan','karthi@gmail.com',22
12 insert into user(User_Name,User_Email,Roll_No>Password) values('Naveen','Naveen@gmail.com',37,'nav
13 insert into user(User_Name,User_Email,Roll_No>Password) values('Salman','salman@gmail.com',47,'sal
14
15 update user set User_Name = 'abinash' where Roll_No=47;
16
17 delete from user where User_Name='abinash';
```

The 'delete from user where User_Name='abinash';' statement is highlighted. Below the script editor, the 'History' tab is active, showing a table of script execution history:

Script	Date	Status	Runtime
Assignment-2	Oct 18, 2022 3:28:00 PM	✓ 1	0.007 s
delete from user where User_Name='abinash'		✓	0.007 s
Assignment-2	Oct 18, 2022 3:25:51 PM	✓ 1	0.008 s
update user set User_Name = 'abinash' where Roll_No=47		✓	0.008 s
Assignment-2	Oct 18, 2022 3:21:22 PM	✓ 3	0.022 s

The screenshot shows the IBM Db2 on Cloud 'Tables' view for the 'LHK92942.USER' table. The table structure is as follows:

SL_NO	USER_NAME	USER_EMAIL	ROLL_NO	PASSWORD
1	Sathiyananth	sakthiyananth@gmail.com	51	Ibm1234
2	Karthikeyan	karthi@gmail.com	22	karthi@22
3	Naveen	Naveen@gmail.com	37	naveen@37

Question-3:

Connect python code to db2.

```
import ibm_db
```

```
import bcrypt
```

```
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32731;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;PROTOCOL=TCP;UID=lhk92942;PWD=Cht7cZes9VeaXQ5N", "", "")
```

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

Solution:**app.py**

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

```
import ibm_db
```

```
import bcrypt
```

```
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32731;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;PROTOCOL=TCP;UID=lhk92942;PWD=Cht7cZes9VeaXQ5N",",")
```

```
# url_for('static', filename='style.css')
```

```
app = Flask(__name__)
```

```
app.secret_key = b'_5#y2L"F4Q8z\n\xec]/'
```

```
@app.route("/", methods=['GET'])
```

```
def home():
```

```
    if 'email' not in session:
```

```
        return redirect(url_for('login'))
```

```
    return render_template('home.html', name='Home')
```

```
@app.route("/register", methods=['GET', 'POST'])
```

```
def register():
```

```

if request.method == 'POST':

    email = request.form['email']

    username = request.form['username']

    rollNo = request.form['rollNo']

    password = request.form['password']

    if not email or not username or not rollNo or not password:

        return render_template('register.html',error='Please fill all fields')

    hash=bcrypt.hashpw(password.encode('utf-8'),bcrypt.gensalt())

    query = "SELECT * FROM user_details WHERE email=? OR rollNo=?"

    stmt = ibm_db.prepare(conn, query)

    ibm_db.bind_param(stmt,1,email)

    ibm_db.bind_param(stmt,2,rollNo)

    ibm_db.execute(stmt)

    isUser = ibm_db.fetch_assoc(stmt)

    if not isUser:

        insert_sql = "INSERT INTO user_details(EMAIL, USERNAME, ROLLNO,
PASSWORD) 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, hash)

```

```

        ibm_db.execute(prepare_stmt)

        return render_template('register.html',success="You can login")

    else:

        return render_template('register.html',error='Invalid Credentials')

    return render_template('register.html',name='Home')

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

    if request.method == 'POST':

        email = request.form['email']

        password = request.form['password']

        if not email or not password:

            return render_template('login.html',error='Please fill all fields')

        query = "SELECT * FROM user_details WHERE email=?"

        stmt = ibm_db.prepare(conn, query)

        ibm_db.bind_param(stmt,1,email)

        ibm_db.execute(stmt)

        isUser = ibm_db.fetch_assoc(stmt)

        print(isUser,password)

        if not isUser:

            return render_template('login.html',error='Invalid Credentials')

```

```
        isPasswordMatch = bcrypt.checkpw(password.encode('utf-8'),isUser['PASSWORD'].encode('utf-8'))

        if not isPasswordMatch:

            return render_template('login.html',error='Invalid Credentials')

        session['email'] = isUser['EMAIL']

        return redirect(url_for('home'))

    return render_template('login.html',name='Home')

@app.route('/logout')

def logout():

    session.pop('email', None)

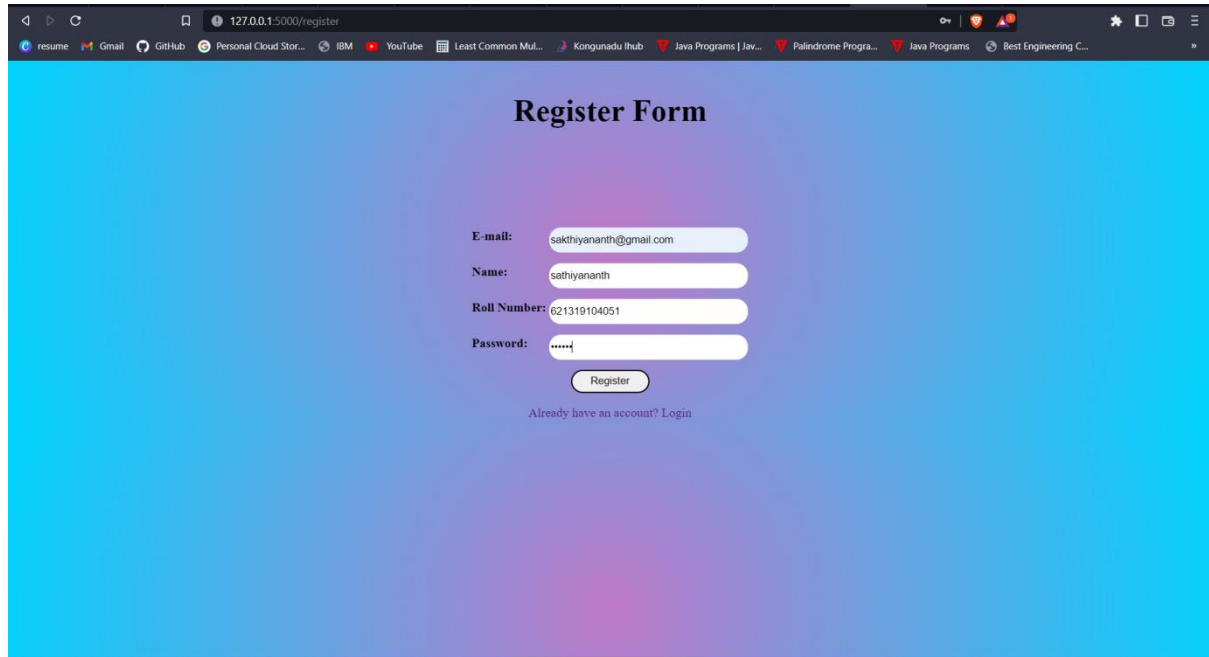
    return redirect(url_for('login'))

if __name__ == "__main__":

    app.run(debug=True)
```

Output:

Registration Page:



127.0.0.1:5000/register

Register Form

E-mail:

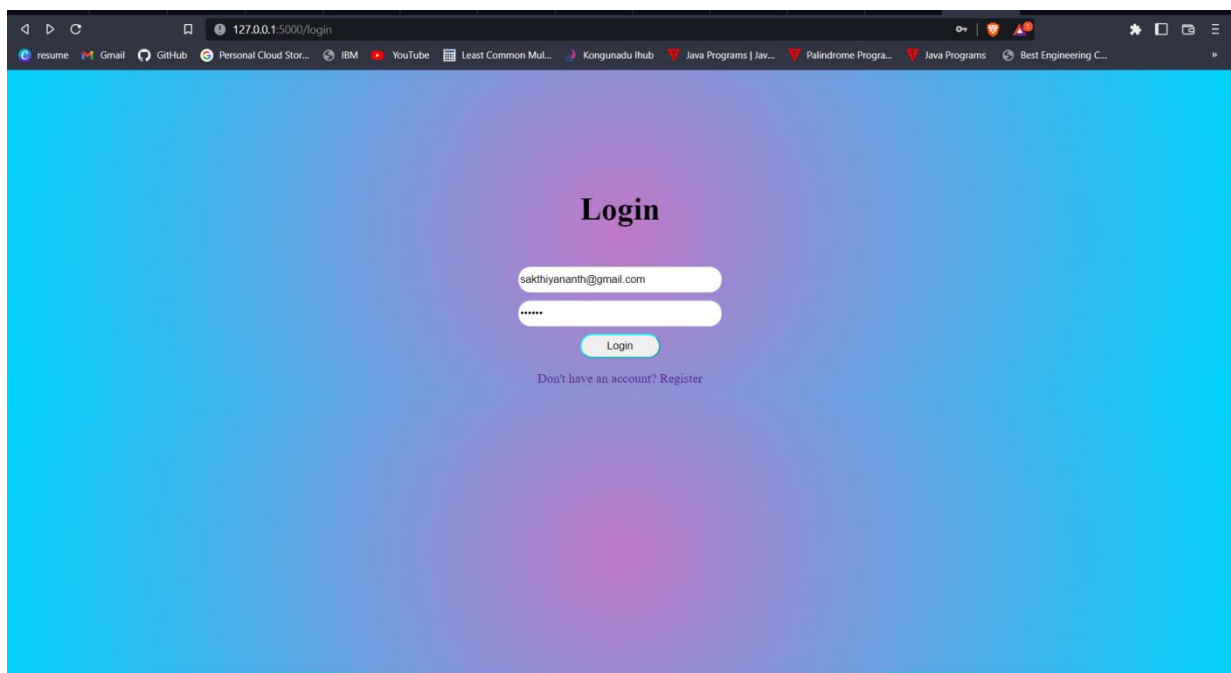
Name:

Roll Number:

Password:

[Already have an account? Login](#)

Login Page:

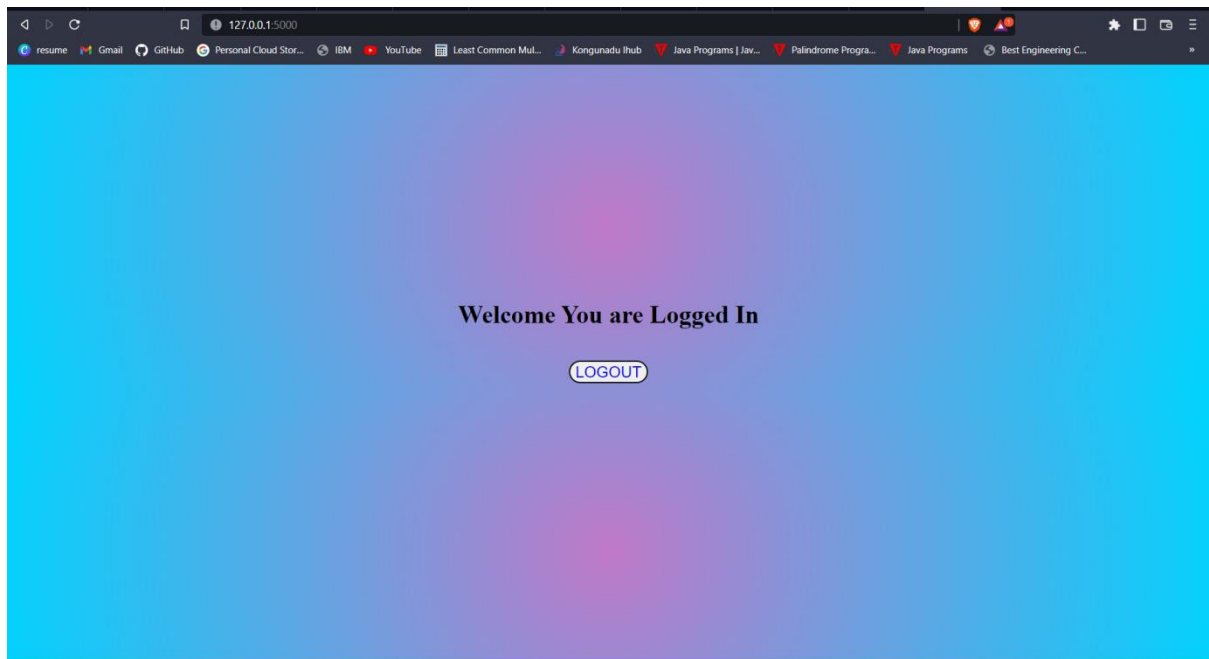


127.0.0.1:5000/login

Login

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

Home Page:



Database:

A screenshot of the IBM Db2 on Cloud console. The interface shows a table named 'LHK92942.USER_DETAILS'. The table has four columns: EMAIL, USERNAME, ROLLNO, and PASSWORD. The data row shows the email 'sakthiyananth@gmail.com', the username 'sathiyananth', the roll number '51', and a hashed password. The console includes a navigation menu on the left with options like 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. There are also buttons for 'Back' and 'Export to CSV'.

EMAIL	USERNAME	ROLLNO	PASSWORD
sakthiyananth@gmail.com	sathiyananth	51	\$2b\$12\$JhZ90QB1epbYBu4qt5ZICOBq3cYqu5NTfzc0JuejZmATFk7WKaEri