

## Assignment -2

### Python Programming

Assignment Date	23 October 2022
Student Name	Srinidhi S
Student Roll Number	2019115105
Maximum Marks	2 Marks

#### Question-1:

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

#### Solution-1:

The screenshot displays the IBM Db2 on Cloud web interface. The 'Tables' tab is active, showing a list of tables. A 'New table' button is visible. The 'Table definition' panel on the right shows the structure of the 'USER' table:

Name	Data type	Nullable	Length	Scale
EMAIL	VARCHAR	N	32	0
USERNAME	VARCHAR	N	32	0
ROLLNO	VARCHAR	N	32	0
PASSWORD	VARCHAR	N	32	0

The interface also shows a search bar for schemas or tables, a 'Refresh' button, and a 'View data' button at the bottom of the table definition panel.

#### Question-2:

Perform UPDATE, DELETE Queries with user table.

#### Solution-2:

##### Insert

```
insert into USER values ('tosrini11@gmail.com','srinidhi11',2019115105, '4853sri');
```

```
insert into USER values ('kmona200181@gmail.com','itsmona14', 2019115055,'123456');
```

```
insert into USER values ('varshaaks@gmail.com','varshaa',2019115116,'asdfghjkl');
```

```
insert into USER values ('harinik@gmail.com', '_harini_',2019115035,'152427');
```

IBM Db2 on Cloud

Data objects | My script

Filter objects

JFY60687

- Tables
- Views
- MQTs
- Aliases
- Nicknames

\*insert x \*update \*delete +

Syntax assistant Run all

```

1 insert into USER values ('tosrini11@gmail.com','srinidhi11',2019115105,'4853sri');
2 insert into USER values ('kmona200181@gmail.com','itsmona14', 2019115055,'123456');
3 insert into USER values ('varshaaks@gmail.com','varshaa',2019115116,'asdfghjkl');
4 insert into USER values ('harinik@gmail.com', '_harini_',2019115035,'152427');
5

```

History Results

Find history

Script	Date	Status	Runtime
insert	Oct 23, 2022 12:29:55 PM	4	0.024 s
insert into USER values ('tosrini11@gmail.com','srinidhi11',2019115105, '48...			0.008 s
insert into USER values ('kmona200181@gmail.com','itsmona14', 2019115055,'1...			0.005 s
insert into USER values ('varshaaks@gmail.com','varshaa',2019115116,'asdfgh...			0.005 s
insert into USER values ('harinik@gmail.com', '_harini_',2019115035,'152427...			0.006 s

IBM Db2 on Cloud

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

JFY60687.USER

Back

Export to CSV

EMAIL	USERNAME	ROLLNO	PASSWORD
harinik@gmail.com	_harini_	2019115035	152427
kmona200181@gmail.com	itsmona14	2019115055	123456
tosrini11@gmail.com	srinidhi11	2019115105	4853sri
varshaaks@gmail.com	varshaa	2019115116	asdfghjkl

## Update

update user set email='tosri12345@gmail.com' where rollno=2019115105;

IBM Db2 on Cloud

Data objects | My script

Filter objects

JFY60687

- Tables
- Views
- MQTs
- Aliases
- Nicknames

insert update x \*delete +

Save SQL script to server Syntax assistant Run all

```

1 update user set email='tosri12345@gmail.com' where rollno=2019115105;
2

```

History Results

Find history

Script	Date	Status	Runtime
update	Oct 23, 2022 12:32:03 PM	1	0.008 s
update user set email='tosri12345@gmail.com' where rollno=2019115105			0.008 s

IBM Db2 on Cloud				
Load Data Load History Tables Views Indexes Aliases MQTs Sequences Application objects				
JFY60687.USER				
<div>Export to CSV</div>				
EMAIL	USERNAME	ROLLNO	PASSWORD	
harinik@gmail.com	_harini_	2019115035	152427	
kmona200181@gmail.com	itsmona14	2019115055	123456	
tosri12345@gmail.com	srinidhi11	2019115105	4853sri	
varshaaks@gmail.com	varshaa	2019115116	asdfghjkl	

## Delete

delete from user where rollno = 2019115035;

IBM Db2 on Cloud				
Data objects My script				
<div>Filter objects</div> <div> <div>JFY60687</div> <div>Tables</div> <div>Views</div> <div>MQTs</div> <div>Aliases</div> <div>Nicknames</div> </div>				
<div>insert update delete</div> <div> <div>delete from user where rollno = 2019115035;</div> </div>				
<div>History Results</div> <div> <div>Script</div> <div>Date</div> <div>Status</div> <div>Runtime</div> </div>				
<div>delete</div> <div>Oct 23, 2022 12:36:00 PM</div> <div>1</div> <div>0.009 s</div>				
<div>delete from user where rollno = 2019115035</div> <div></div> <div>0.009 s</div>				

IBM Db2 on Cloud				
Load Data Load History Tables Views Indexes Aliases MQTs Sequences Application objects				
JFY60687.USER				
<div>Export to CSV</div>				
EMAIL	USERNAME	ROLLNO	PASSWORD	
kmona200181@gmail.com	itsmona14	2019115055	123456	
tosri12345@gmail.com	srinidhi11	2019115105	4853sri	
varshaaks@gmail.com	varshaa	2019115116	asdfghjkl	

**Question-3:**

Connect python code to db2.

**Solution-3:**

```
# -*- coding: utf-8 -*-
```

```
''''
```

```
@author: Srinidhi
```

```
''''
```

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

```
import ibm_db
```

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

```
app.secret_key = 'a'
```

```
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30699;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=jfy60687;PWD=XVHyqh5GQJBW5LFF","")
```

```
if(conn):
```

```
    print("CONNECTED SUCCESSFULLY")
```

```
    print("Connection : "+str(conn))
```

```
    sql="SELECT * FROM USER WHERE rollno=2019115105"
```

```
    email="tosri12345@gmail.com"
```

```
    stmt = ibm_db.prepare(conn,sql)
```

```
    ibm_db.execute(stmt)
```

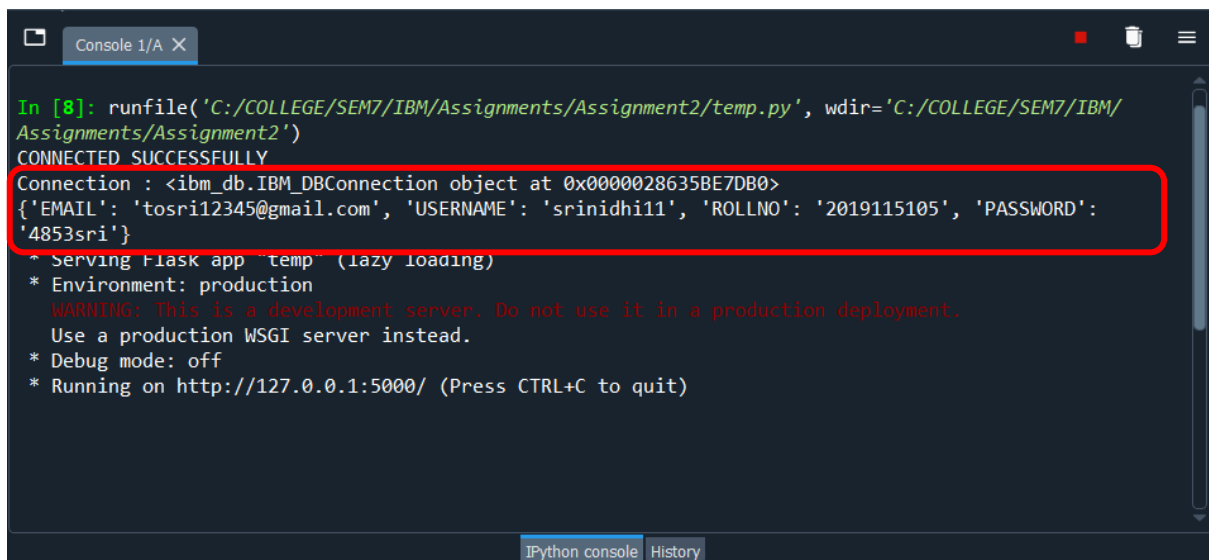
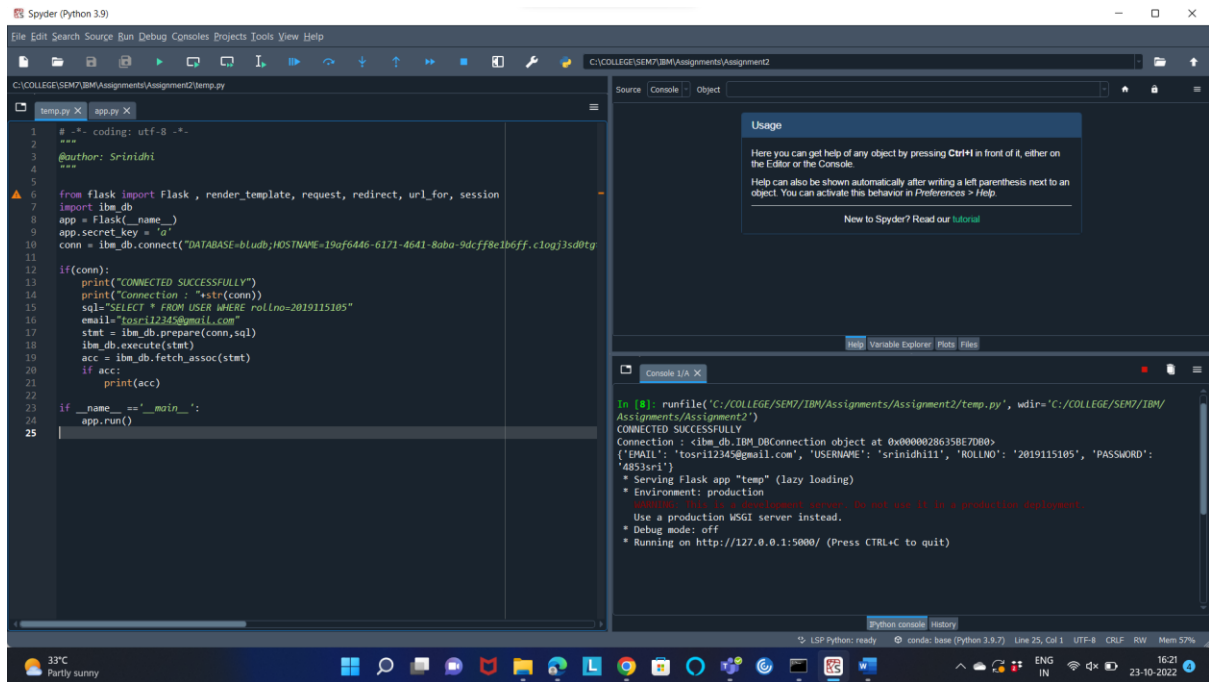
```
    acc = ibm_db.fetch_assoc(stmt)
```

```
    if acc:
```

```
        print(acc)
```

```
if __name__ == '__main__':
```

```
    app.run()
```



#### 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-4:

```
# -*- coding: utf-8 -*-
```

```
=====
```

Created on Sun Oct 23 16:04:42 2022

@author: tosri

"""

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

```
import ibm_db
```

```
import re
```

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

```
app.secret_key = 'a'
```

```
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30699;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=jfy60687;PWD=XVHyqh5GQJBW5LFF","")
```

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

```
def login():
```

```
    global userid
```

```
    msg = "
```

```
    if request.method == 'POST':
```

```
        email = request.form['email']
```

```
        password = request.form['password']
```

```
        sql = "SELECT * FROM USERS WHERE EMAIL=? AND PASSWORD=?"
```

```
        stmt = ibm_db.prepare(conn,sql)
```

```
        ibm_db.bind_param(stmt,1,email)
```

```
        ibm_db.bind_param(stmt,2,password)
```

```
        ibm_db.execute(stmt)
```

```
        acc = ibm_db.fetch_assoc(stmt)
```

```
        print(acc)
```

```
        if acc:
```

```
            session['loggedin'] = True
```

```
            session['id'] = acc['USERNAME']
```

```
            userid = acc['USERNAME']
```

```
            session['username'] = acc['USERNAME']
```

```

        msg = acc['USERNAME']

        return render_template('dashboard.html', msg = msg)
else:
    msg = "Incorrect username/password!!"
    return render_template('login.html', msg = msg)

@app.route('/register',methods=['GET', 'POST'])
def register():
    msg = ""
    if request.method == 'POST':
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        sql = "SELECT * FROM USERS WHERE USERNAME=?"
        stmt = ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        acc = ibm_db.fetch_assoc(stmt)
        print(acc)
        if acc:
            msg = "Account already exists !!"
        elif not re.match(r'^@+@[^@]+\.[^@]+',email):
            msg = "Invalid Email address"
        elif not re.match(r'[A-Za-z0-9]+',username):
            msg = "Name must contain only characters and numbers !!"
        else:
            sql = "INSERT INTO USERS VALUES (?,?,?)"
            stmt = ibm_db.prepare(conn,sql)
            ibm_db.bind_param(stmt,1,username)
            ibm_db.bind_param(stmt,2,email)
            ibm_db.bind_param(stmt,3,password)

```

```

ibm_db.execute(stmt)

msg = "Successgully registered !!Login to continue"

return render_template('login.html', msg = msg)

elif request.method == 'POST':

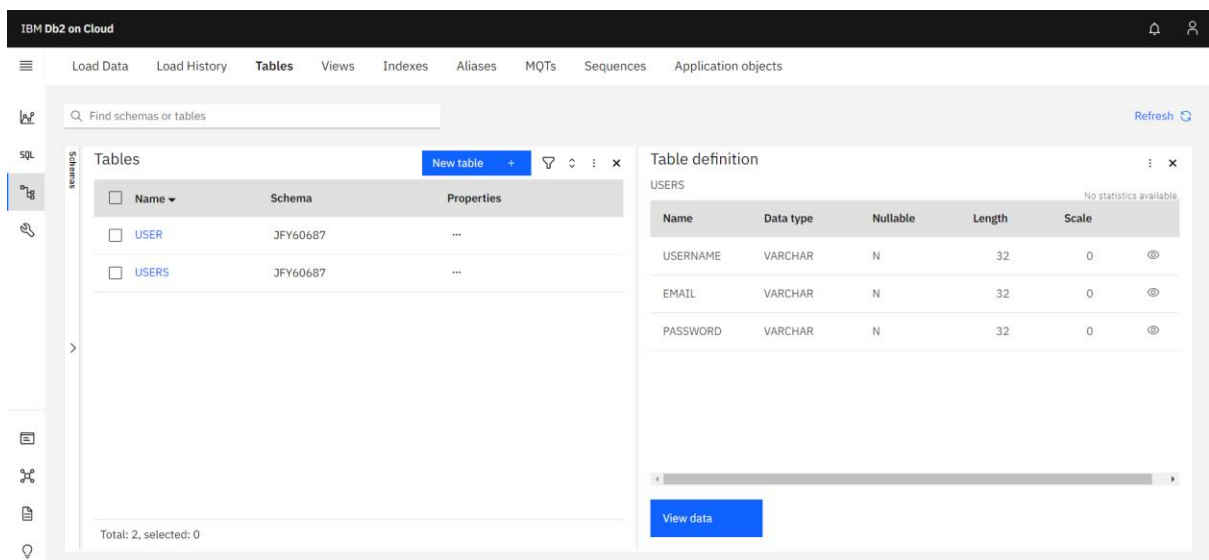
    msg = "Please fill out the form !"

    return render_template('register.html', msg = msg)

if __name__ == '__main__':

    app.run()

```



IBM Db2 on Cloud

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

Find schemas or tables Refresh

**Tables** New table

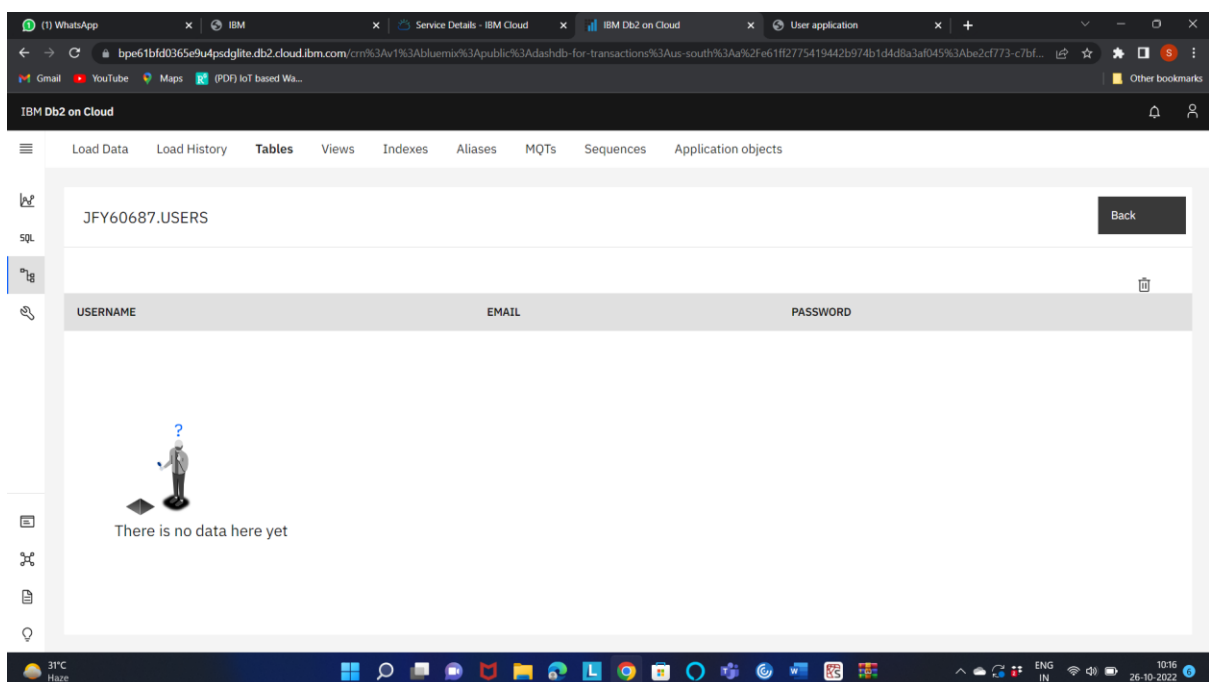
Name	Schema	Properties
USER	JFY60687	...
USERS	JFY60687	...

Total: 2, selected: 0

**Table definition** USERS

Name	Data type	Nullable	Length	Scale
USERNAME	VARCHAR	N	32	0
EMAIL	VARCHAR	N	32	0
PASSWORD	VARCHAR	N	32	0

View data



IBM Db2 on Cloud

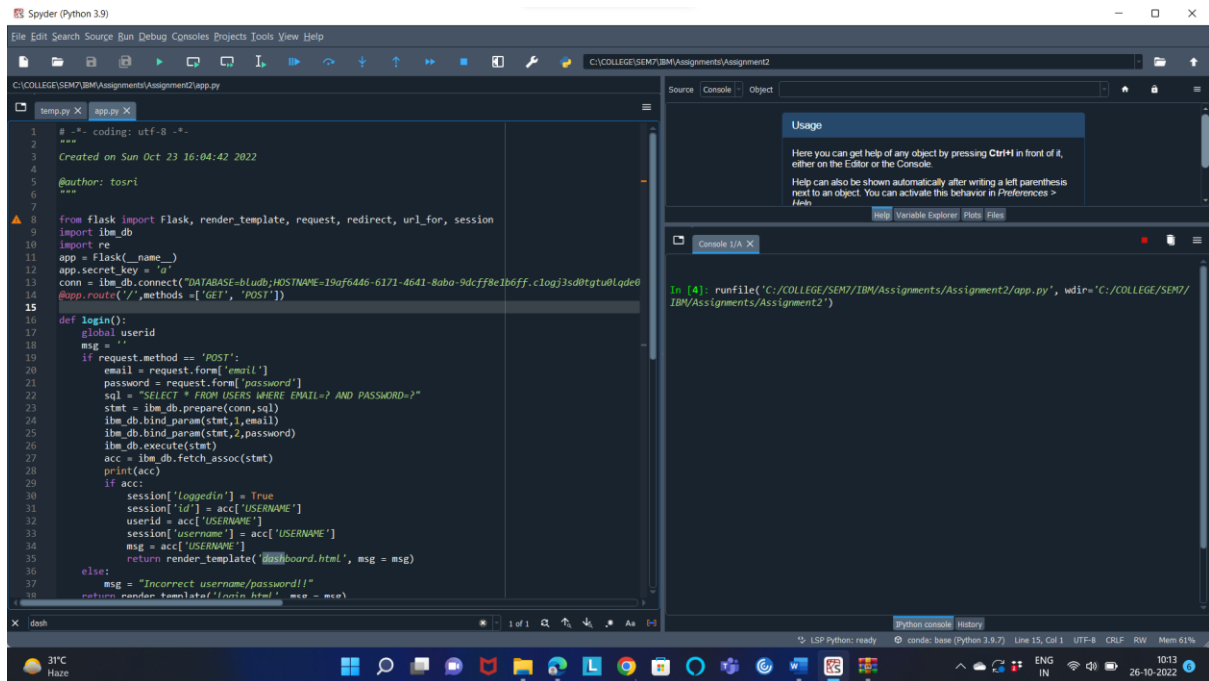
Load Data Load History **Tables** Views Indexes Aliases MQTs Sequences Application objects

JFY60687.USERS Back

USERNAME	EMAIL	PASSWORD
There is no data here yet		



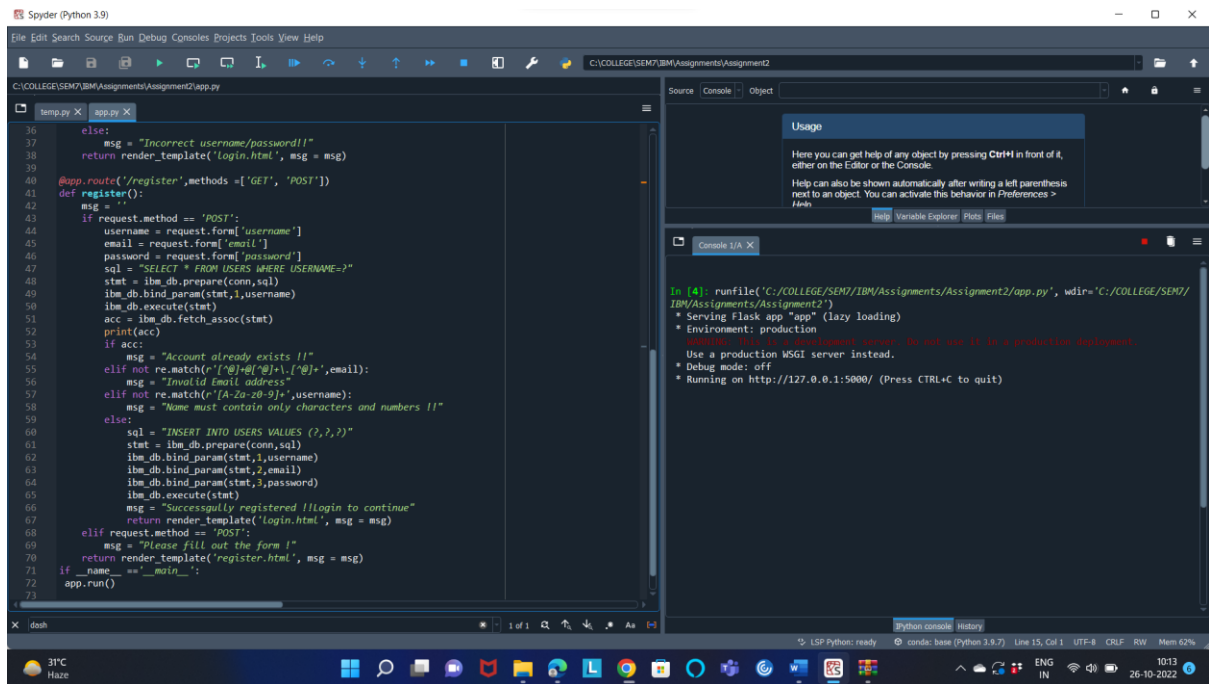
# Code



The screenshot shows the Spyder Python IDE interface. The main editor displays a Python file named `app.py` with the following code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Sun Oct 23 16:04:42 2022
4
5 @author: tostri
6 """
7
8 from flask import Flask, render_template, request, redirect, url_for, session
9 import ibm_db
10 import re
11 app = Flask(__name__)
12 app.secret_key = "z"
13 conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=19df6446-6171-4641-8aba-9dcff8e1b6ff.clog3sdtgtuqlde0
14 @app.route('/', methods = ['GET', 'POST'])
15
16 def login():
17     global userid
18     msg = ""
19     if request.method == 'POST':
20         email = request.form['email']
21         password = request.form['password']
22         sql = "SELECT * FROM USERS WHERE EMAIL=? AND PASSWORD=?"
23         stmt = ibm_db.prepare(conn, sql)
24         ibm_db.bind_param(stmt, 1, email)
25         ibm_db.bind_param(stmt, 2, password)
26         ibm_db.execute(stmt)
27         acc = ibm_db.fetch_assoc(stmt)
28         print(acc)
29         if acc:
30             session['logged_in'] = True
31             session['id'] = acc['USERNAME']
32             userid = acc['USERNAME']
33             session['username'] = acc['USERNAME']
34             msg = acc['USERNAME']
35             return render_template('dashboard.html', msg = msg)
36         else:
37             msg = "Incorrect username/password!!"
38             return render_template('login.html', msg = msg)
```

The right-hand pane shows the 'Usage' section with a message: "Here you can get help of any object by pressing **Ctrl+H** in front of it, either on the Editor or the Console." Below this, the 'Console I/O' pane shows the command: `runfile('C:/COLLEGE/SEM7/IBM/Assignments/Assignment2/app.py', wdir='C:/COLLEGE/SEM7/IBM/Assignments/Assignment2')`.



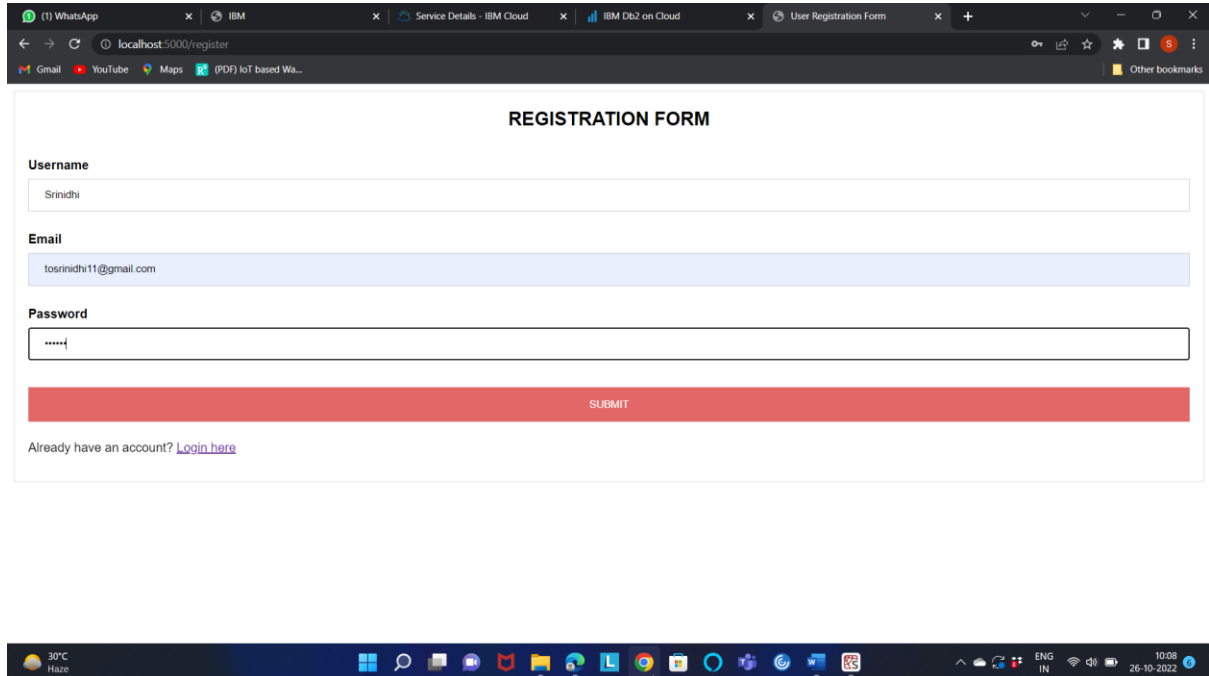
The screenshot shows the Spyder Python IDE interface. The main editor displays a Python file named `app.py` with the following code:

```
36 else:
37     msg = "Incorrect username/password!!"
38     return render_template('login.html', msg = msg)
39
40 @app.route('/register', methods = ['GET', 'POST'])
41 def register():
42     msg = ""
43     if request.method == 'POST':
44         username = request.form['username']
45         email = request.form['email']
46         password = request.form['password']
47         sql = "SELECT * FROM USERS WHERE USERNAME=?"
48         stmt = ibm_db.prepare(conn, sql)
49         ibm_db.bind_param(stmt, 1, username)
50         ibm_db.execute(stmt)
51         acc = ibm_db.fetch_assoc(stmt)
52         print(acc)
53         if acc:
54             msg = "Account already exists!!"
55         elif not re.match(r'([a-zA-Z0-9]+@[a-zA-Z0-9]+\.[a-zA-Z0-9]+)', email):
56             msg = "Invalid Email address"
57         elif not re.match(r'[A-Za-z0-9]*', username):
58             msg = "Name must contain only characters and numbers!!"
59         else:
60             sql = "INSERT INTO USERS VALUES (?, ?, ?)"
61             stmt = ibm_db.prepare(conn, sql)
62             ibm_db.bind_param(stmt, 1, username)
63             ibm_db.bind_param(stmt, 2, email)
64             ibm_db.bind_param(stmt, 3, password)
65             ibm_db.execute(stmt)
66             msg = "Successfully registered !!! Login to continue"
67             return render_template('login.html', msg = msg)
68         elif request.method == 'POST':
69             msg = "Please fill out the form!"
70             return render_template('register.html', msg = msg)
71     if __name__ == '__main__':
72         app.run()
```

The right-hand pane shows the 'Usage' section with a message: "Here you can get help of any object by pressing **Ctrl+H** in front of it, either on the Editor or the Console." Below this, the 'Console I/O' pane shows the command: `runfile('C:/COLLEGE/SEM7/IBM/Assignments/Assignment2/app.py', wdir='C:/COLLEGE/SEM7/IBM/Assignments/Assignment2')`. The console output shows the following messages:

- \* Serving Flask app "app" (lazy loading)
- \* Environment: production
- \* Use a production WSGI server instead.
- \* Debug mode: off
- \* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)

# Register



REGISTRATION FORM

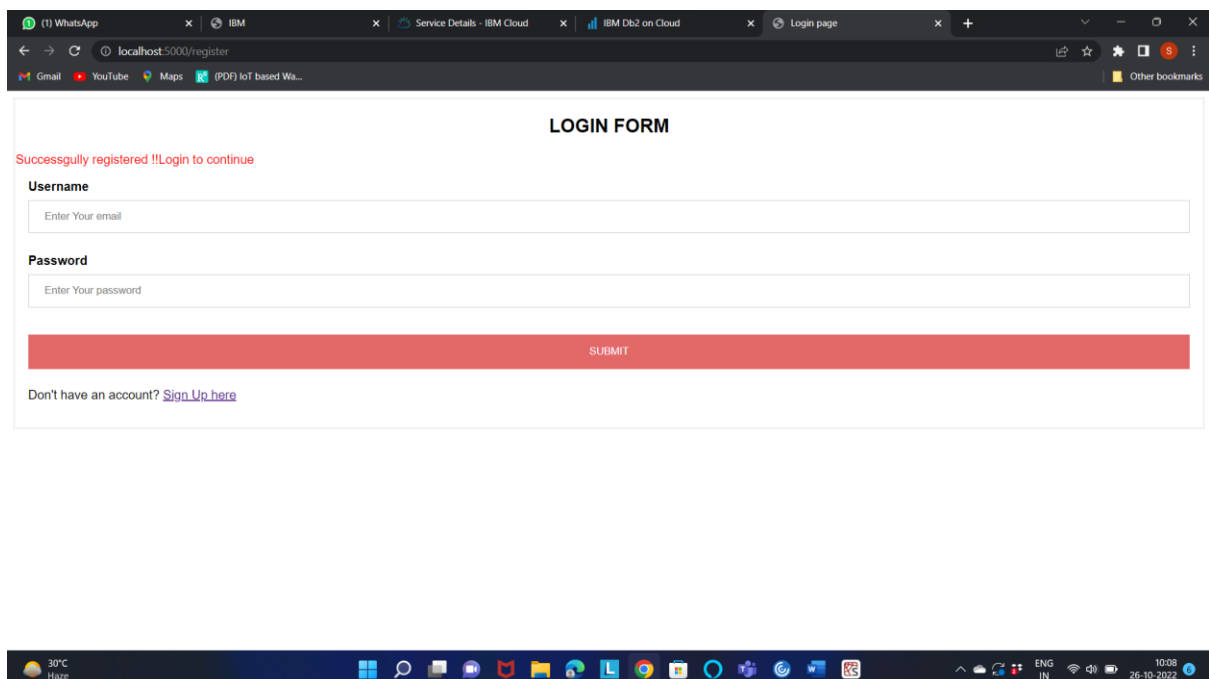
**Username**

**Email**

**Password**

**SUBMIT**

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



LOGIN FORM

Successfully registered !!Login to continue

**Username**

**Password**

**SUBMIT**

Don't have an account? [Sign Up here](#)

# Login

LOGIN FORM

Username

Srinidhi

Password

\*\*\*\*\*

SUBMIT

Don't have an account? [Sign Up here](#)

Welcome Srinidhi

## Table

The screenshot displays the IBM Db2 on Cloud web interface. At the top, there's a navigation bar with tabs for 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The 'Tables' tab is currently selected. Below this, the schema 'JFY60687.USERS' is shown. A table with the name 'USERS' is displayed, containing three columns: 'USERNAME', 'EMAIL', and 'PASSWORD'. A single data row is visible with the values 'Srinidhi', 'tosrinidhi11@gmail.com', and '123456'. To the right of the table, there are buttons for 'Back' and 'Export to CSV'. The bottom of the image shows a Windows taskbar with various icons and a system tray indicating the date and time as 26-10-2022, 10:09.

USERNAME	EMAIL	PASSWORD
Srinidhi	tosrinidhi11@gmail.com	123456