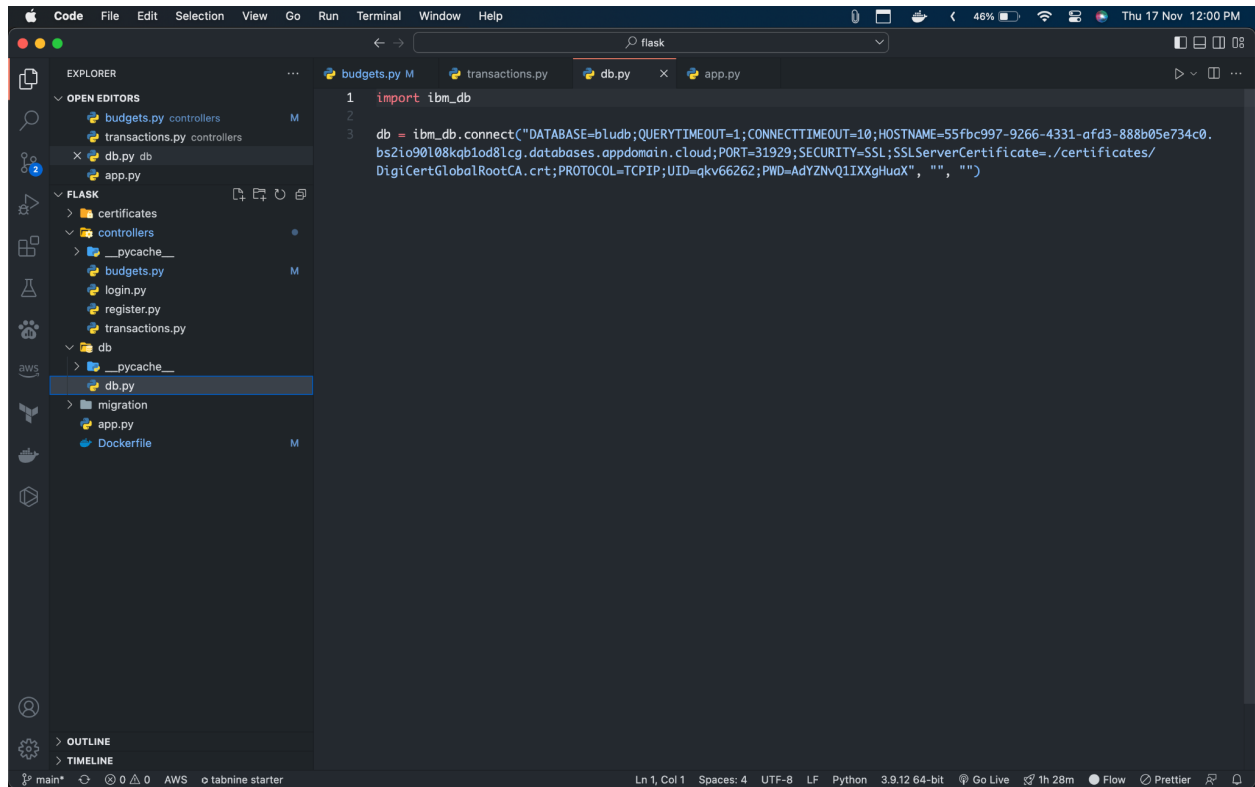


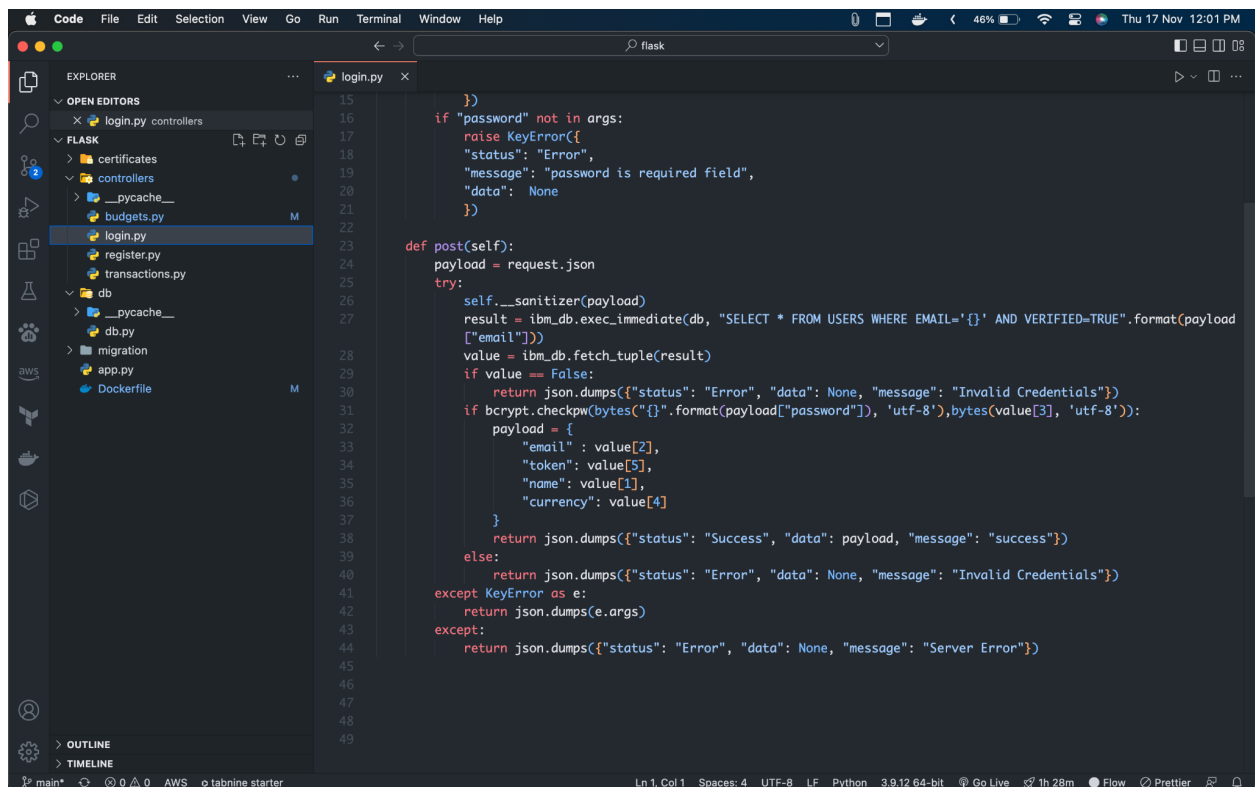
# Create IBM DB2 and connect with python

## Flask Code



This screenshot shows the Visual Studio Code editor with the `db.py` file open. The Explorer sidebar on the left shows the project structure, including `controllers`, `certificates`, `__pycache__`, `budgets.py`, `login.py`, `register.py`, `transactions.py`, `db`, `__pycache__`, `migration`, `app.py`, and `Dockerfile`. The `db.py` file contains the following code:

```
1 import ibm_db
2
3 db = ibm_db.connect("DATABASE=bludb;QUERYTIMEOUT=1;CONNECTTIMEOUT=10;HOSTNAME=55Fbc997-9266-4331-afd3-888b05e734c0.
bs2io90108kqb1od8lcg.databases.appdomain.cloud;PORT=31929;SECURITY=SSL;SSLServerCertificate=./certificates/
DigiCertGlobalRootCA.crt;PROTOCOL=TCPIP;UID=qkv6626Z;PWD=AdyZnvQ1IXXghuAX", "", "")
```



This screenshot shows the Visual Studio Code editor with the `login.py` file open. The Explorer sidebar on the left shows the project structure, including `controllers`, `certificates`, `__pycache__`, `budgets.py`, `login.py`, `register.py`, `transactions.py`, `db`, `__pycache__`, `migration`, `app.py`, and `Dockerfile`. The `login.py` file contains the following code:

```
15 })
16
17 if "password" not in args:
18     raise KeyError({
19         "status": "Error",
20         "message": "password is required field",
21         "data": None
22     })
23
24 def post(self):
25     payload = request.json
26     try:
27         self.__sanitizer(payload)
28         result = ibm_db.exec_immediate(db, "SELECT * FROM USERS WHERE EMAIL='{0}' AND VERIFIED=TRUE".format(payload["email"]))
29         value = ibm_db.fetch_tuple(result)
30         if value == False:
31             return json.dumps({"status": "Error", "data": None, "message": "Invalid Credentials"})
32         if bcrypt.checkpw(bytes("{}").format(payload["password"]), 'utf-8'), bytes(value[3], 'utf-8')):
33             payload = {
34                 "email": value[2],
35                 "token": value[5],
36                 "name": value[1],
37                 "currency": value[4]
38             }
39             return json.dumps({"status": "Success", "data": payload, "message": "success"})
40         else:
41             return json.dumps({"status": "Error", "data": None, "message": "Invalid Credentials"})
42     except KeyError as e:
43         return json.dumps(e.args)
44     except:
45         return json.dumps({"status": "Error", "data": None, "message": "Server Error"})
46
47
48
49
```