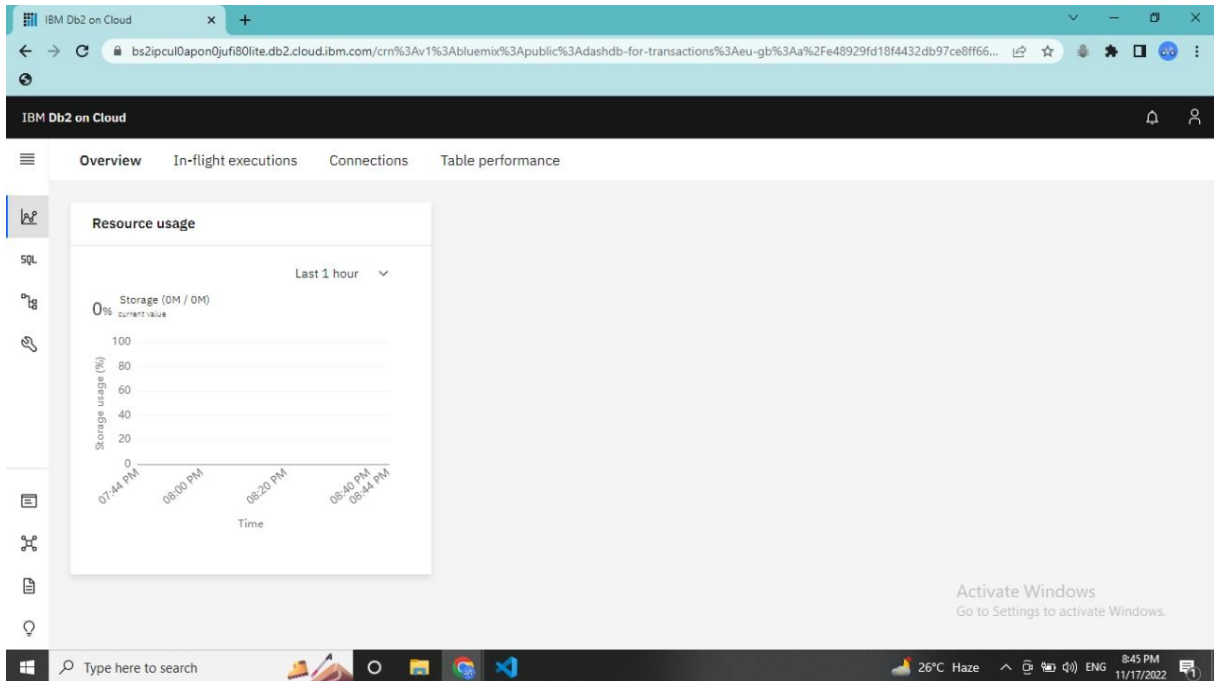


# IMPLEMENTING WEB APPLICATION

## Create IBM DB2 And Connect With Python:



The screenshot shows a Visual Studio Code editor window with a Python script named 'app.py' open. The script is a Flask application that connects to an IBM Db2 database. The code includes imports for Flask, render\_template, request, redirect, url\_for, session, ibm\_db, re, os, and SendGridAPIClient. It defines a Flask app and sets the secret key to 'ibm'. The database connection details are hardcoded, including the hostname, uid, pwd, driver, db, port, protocol, and cert. The DSN (Data Source Name) is formatted using these details. The script prints the DSN and connects to the database using ibm\_db.connect().

```
1 from flask import Flask, render_template, request, redirect, url_for, session
2 import ibm_db
3 import re
4 import os
5 from sendgrid import SendGridAPIClient
6 from sendgrid.helpers.mail import *
7
8 app = Flask(__name__)
9 app.secret_key = "ibm"
10
11 hostname = "ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8l1cg.databases.appdomain.cloud"
12 uid = "jbd44427"
13 pwd = "bbyhN0MedJ6mIj7t"
14 driver = "{IBM DB2 ODBC DRIVER}"
15 db = "bludb"
16 port = "31505"
17 protocol = "TCPIP"
18 cert = "DigiCertGlobalRootCA.crt"
19
20
21 dsn = (
22     "DATABASE={0};"
23     "HOSTNAME={1};"
24     "PORT={2};"
25     "UID={3};"
26     "SECURITY=SSL;"
27     "SSLServerCertificate={4};"
28     "Pwd={5};"
29 ).format(db, hostname, port, uid, cert, pwd)
30
31 print(dsn)
32
33 conn = ibm_db.connect(dsn, "", "")
```

```
app.py - IBM MY EXPENCE - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
  OPEN EDITORS
  IBM MY EXPENCE
    projectMainDummy
      static
      templates
      DigiCertGlobalRootCA...
      Dockerfile
      requirements.txt
      static
        home4.css
        index.css
        style.css
      templates
        home.html
        index.html
      app.py
      DigiCertGlobalRootCA...
      Dockerfile
      requirements.txt
      sendEmail.py
  OUTLINE
  TIMELINE
  app.py x home.html 1 # home4.css # index.css # index.html # style.css
  app.py > tracker
32 conn = ibm_db.connect(dsn, "", "")
33
34 message = ""
35
36
37 @app.route('/', methods=['GET', 'POST'])
38 def home():
39     print(session)
40     print("Message - " + message)
41     if session:
42         if session["loggedin"]:
43             return redirect(url_for('tracker'))
44         else:
45             login_page = True
46             print(request.values.get('page'))
47             if request.values.get('page') == "register":
48                 login_page = False
49             return render_template('index.html', login=login_page, message=message)
50
51
52 @app.route('/login', methods=['GET', 'POST'])
53 def login():
54     if request.method == "POST":
55         global message
56
57         user = request.form
58         print(user)
59         email = user["email"]
60         passwd = user["passwd"]
61
62         print("Email - " + email + ", Password - " + passwd)
63
64         sql = "SELECT * FROM users WHERE email = ? AND pass = ?"
```

```

65         stmt = ibm_db.prepare(conn, sql)
66         ibm_db.bind_param(stmt, 1, email)
67         ibm_db.bind_param(stmt, 2, passwd)
68         ibm_db.execute(stmt)
69
70         account = ibm_db.fetch_assoc(stmt)
71         print("Account - ")
72         print(account)
73
74         if account:
75             session['loggedin'] = True
76             session['id'] = account['EMAIL']
77             user_email = account['EMAIL']
78             session['email'] = account['EMAIL']
79             session['name'] = account['NAME']
80             return redirect(url_for('tracker'))
81
82         else:
83             message = "Incorrect Email or Password"
84             return redirect(url_for('home'))
85
86
87
88 @app.route('/register', methods=['GET', 'POST'])
89 def register():
90     if request.method == "POST":
91         global message
92
93         user = request.form
94         print(user)
95         name = user["name"]
```

