SPRINT 3

Date	11-10-2022
Team ID	PNT2022TMID35567
Project Name	Efficient water quality analysis and prediction using Machine Learning

CODE :

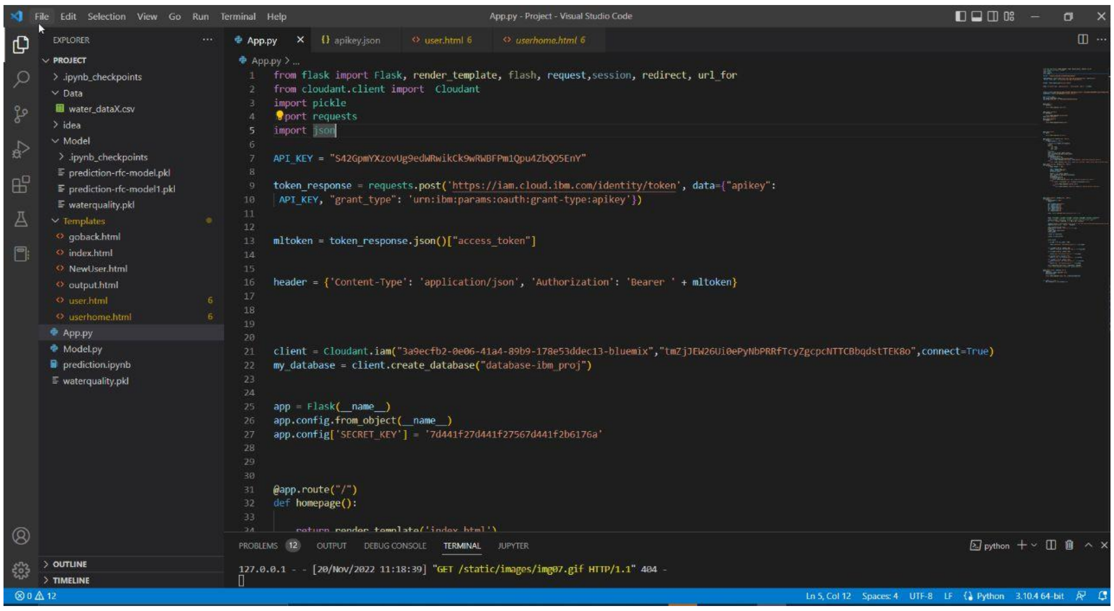
```
from flask import Flask, render_template, flash, request,session, redirect, url_for

from cloudant.client import  Cloudant

import pickle

import requests

import json

API_KEY = "S42GpmYXzovUg9edWRwikCk9wRWBFPm1Qpu4ZbQO5EnY"

token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":

 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})

mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

client = Cloudant.iam("3a9ecfb2-0e06-41a4-89b9-178e53ddec13-
bluemix","tmZjJEW26Ui0ePyNbPRRfTcyZgcpcNTTCBbqdstTEK8o",connect=True)

my_database = client.create_database("database-ibm_proj")

app = Flask(_name_)

app.config.from_object(_name_)

app.config['SECRET_KEY'] = '7d441f27d441f27567d441f2b6176a'


@app.route("/")

def homepage():

   return render_template('index.html')


@app.route("/userhome")

def userhome():

   return render_template('userhome.html')
```

```
@app.route("/addamount")


@app.route("/NewUser")
def NewUser():
   return render_template('NewUser.html')


@app.route("/user")
def user():
   return render_template('user.html')


@app.route("/newuse",methods=['GET','POST'])
def newuse():
    if request.method == 'POST':#
        x = [x for x in request.form.values()]
        print(x)
        data = {
            '_id': x[1],
            'name': x[0],
            'psw': x[2]
        }
        print(data)
        query = {'_id': {'Seq': data['_id']}}
        docs = my_database.get_query_result(query)
        print(docs)
        print(len(docs.all()))
        if (len(docs.all()) == 0):
            url = my_database.create_document(data)
            return render_template('goback.html', data="Register, please login using your
details")
        else:
            return render_template('goback.html', data="You are already a member, please
login using your details")
```

```
@app.route("/userlog", methods=['GET', 'POST'])
def userlog():
        if request.method == 'POST':
            user = request.form['_id']
            passw = request.form['psw']
            print(user, passw)
            query = {'_id': {'$eq': user}}
            docs = my_database.get_query_result(query)
            print(docs)
            print(len(docs.all()))
            if (len(docs.all()) == 0):
                return render_template('goback.html', pred="The username is not found.")
            else:
                if ((user == docs[0][0]['_id'] and passw == docs[0][0]['psw'])):
                    return render_template("userhome.html")
                else:
                    return render_template('goback.html',data="user name and password
incorrect")

@app.route("/predict", methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':
        outttt =""
        year = request.form["year"]
        do = request.form["do"]
        ph = request.form["ph"]
        co = request.form["co"]
        bod = request.form["bod"]
        na = request.form["na"]
        tc = request.form["tc"]
        model = pickle.load(open('Model/waterquality.pkl','rb'))
        total = [[int(year), float(do), float(ph), float(co), float(bod), float(na), float(tc)]]
        #total = int(year)+ float(do)+ float(ph)+ float(co)+float(bod)+float(na)+ float(tc)
        payload_scoring = {"input_data": [{"fields": ['Year', 'do', 'ph', 'co', 'bod',
       'na', 'tc'], "values": [[2014,6.7, 7.5, 203, 2, 0.1, 27.0]]}]}
        response_scoring = requests.post('https://eu-
de.ml.cloud.ibm.com/ml/v4/deployments/d95fbefa-4503-49cf-b870-
1348309c3bdc/predictions?version=2022-11-16', json=payload_scoring,
        headers={'Authorization': 'Bearer ' + mltoken})
        pred = response_scoring.json()
        print("Scoring response...\n-----------------\n")
        print(pred)
        y_pred = model.predict(total)
        print(y_pred)
        y_pred1 = y_pred[[0][0]]
        y_pred2 = y_pred1[[10][0]]
        print(y_pred2)
        if (y_pred2 >= 95 and y_pred2 <= 100):
           outttt ="Excellent, the Predicted value is " + str(y_pred2)
        elif (y_pred2 >= 89 and y_pred2 <= 94):
           outttt = "Very good, the Predicted value is " + str(y_pred2)
        elif (y_pred2 >= 80 and y_pred2 <= 88):
           outttt="Good, the Predicted value is " + str(y_pred2)
        elif(y_pred2 >= 65 and y_pred2 <= 79):
           outttt = "Fair, the Predicted value is " + str(y_pred2)
        elif (y_pred2 >= 45 and y_pred2 <= 64):
           outttt ="Marginal, the Predicted value is " + str(y_pred2)
        else:
           outttt="Poor, the Predicted value is " + str(y_pred2)
        return redirect(url_for('output', pred=outttt), code=307)
        #return render_template('userhome.html', prediction=outttt)
```

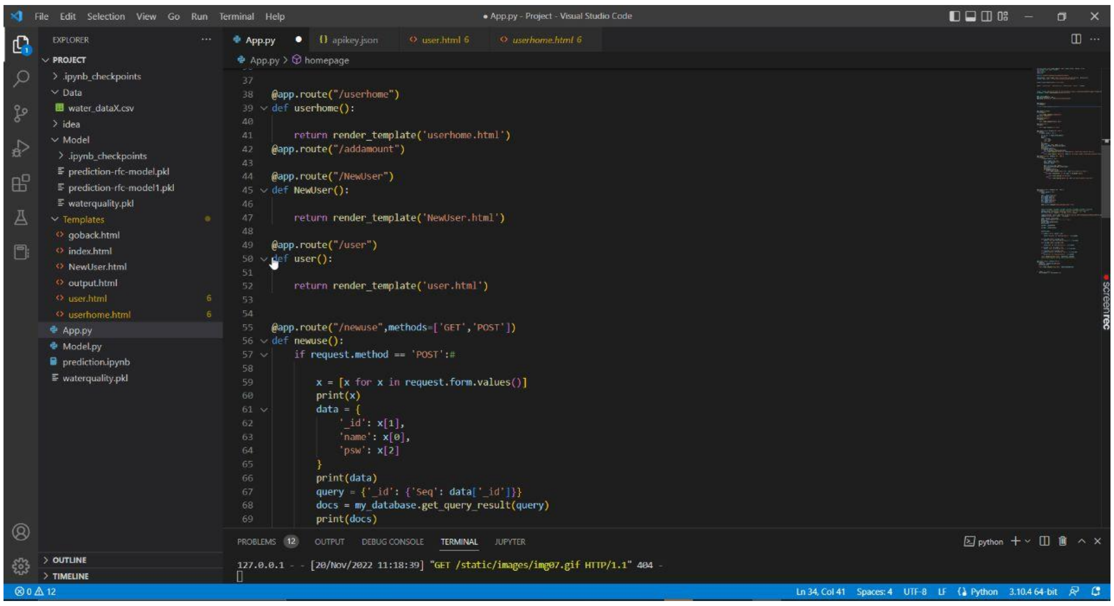
```
@app.route("/output", methods=['POST'])
def output():
    prediction = request.args.get('pred')
    print(prediction)
    return render_template('output.html', prediction=prediction)
if _name_ == '_main_':
   app.run(debug=True, use_reloader=True)
```

SCREENSHOT :



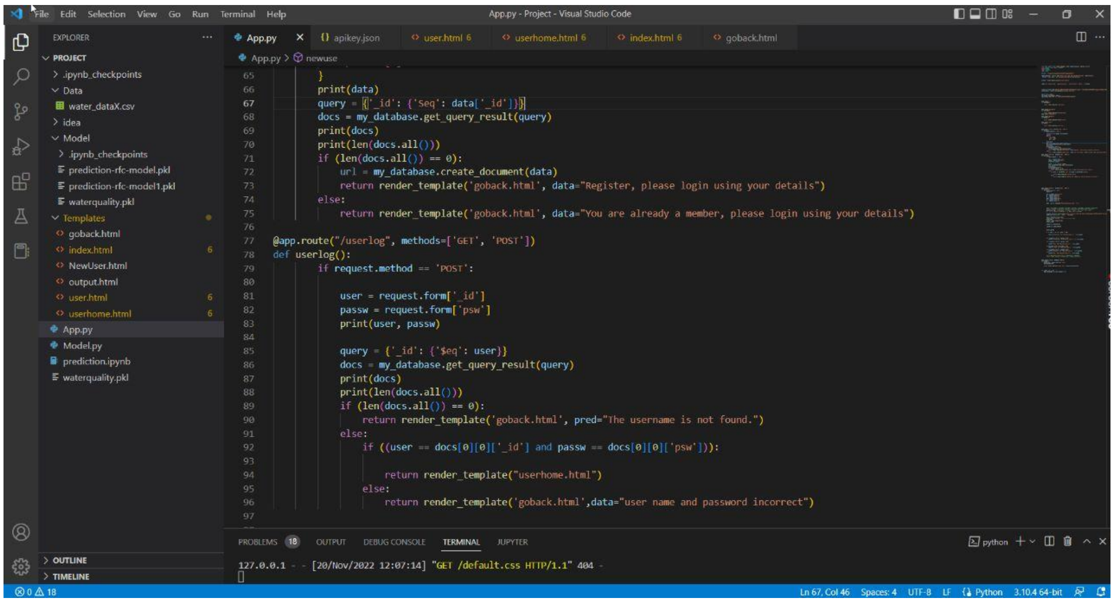
```
from flask import Flask, render_template, flash, request,session, redirect, url_for
from cloudant.client import  Cloudant
import pickle
  port requests
import json

API_KEY = "S42GpmYXzovUg9edWRwikCk9wRWBFPm1Qpu4ZbQO5EnY"

token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})


mltoken = token_response.json()["access_token"]


header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}




client = Cloudant.iam("3a9ecfb2-0e06-41a4-89b9-178e53ddec13-bluemix","tmZjJEW26Ui0ePyNbPRRfTcyZgcpcNTTCBbqdstTEK8o",connect=True)
my_database = client.create_database("database-ibm_proj")


app = Flask(__name__)
app.config.from_object(__name__)
app.config['SECRET_KEY'] = '7d441f27d441f27567d441f2b6176a'



@app.route("/")
def homepage():

    return render_template('index.html')
```

```
127.0.0.1 - - [20/Nov/2022 11:18:39] "GET /static/images/img07.gif HTTP/1.1" 404 -
```



```

@app.route("/userhome")
def userhome():

    return render_template('userhome.html')
@app.route("/addamount")

@app.route("/NewUser")
def NewUser():

    return render_template('NewUser.html')

@app.route("/user")
def user():

    return render_template('user.html')


@app.route("/newuse",methods=['GET','POST'])
def newuse():
    if request.method == 'POST':#

        x = [x for x in request.form.values()]
        print(x)
        data = {
            '_id': x[1],
            'name': x[0],
            'psw': x[2]
        }
        print(data)
        query = {'_id': {'Seq': data['_id']}}
        docs = my_database.get_query_result(query)
        print(docs)
```

```
127.0.0.1 - - [20/Nov/2022 11:18:39] "GET /static/images/img07.gif HTTP/1.1" 404 -
```



```python
        }
        print(data)
        query = {'_id': {'Seq': data['_id']}}
        docs = my_database.get_query_result(query)
        print(docs)
        print(len(docs.all()))
        if (len(docs.all()) == 0):
            url = my_database.create_document(data)
            return render_template('goback.html', data="Register, please login using your details")
        else:
            return render_template('goback.html', data="You are already a member, please login using your details")

@app.route("/userlog", methods=['GET', 'POST'])
def userlog():
        if request.method == 'POST':

            user = request.form['_id']
            passw = request.form['psw']
            print(user, passw)

            query = {'_id': {'$eq': user}}
            docs = my_database.get_query_result(query)
            print(docs)
            print(len(docs.all()))
            if (len(docs.all()) == 0):
                return render_template('goback.html', pred="The username is not found.")
            else:
                if ((user == docs[0][0]['_id'] and passw == docs[0][0]['psw'])):

                    return render_template("userhome.html")
                else:
                    return render_template('goback.html',data="user name and password incorrect")

```

```
127.0.0.1 - - [20/Nov/2022 12:07:14] "GET /default.css HTTP/1.1" 404 -
```