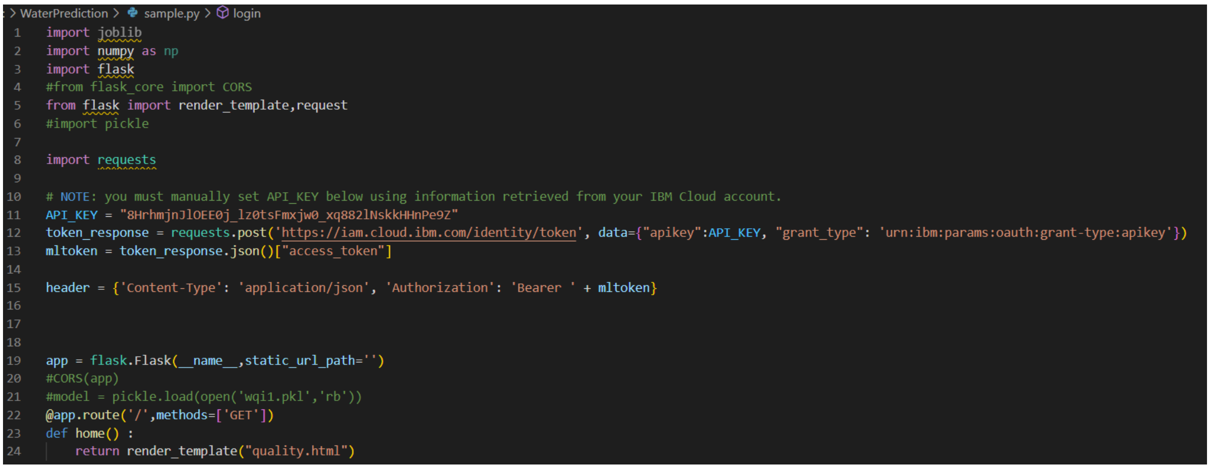
Project Development Phase
Project Development – Delivery Of Sprint-4

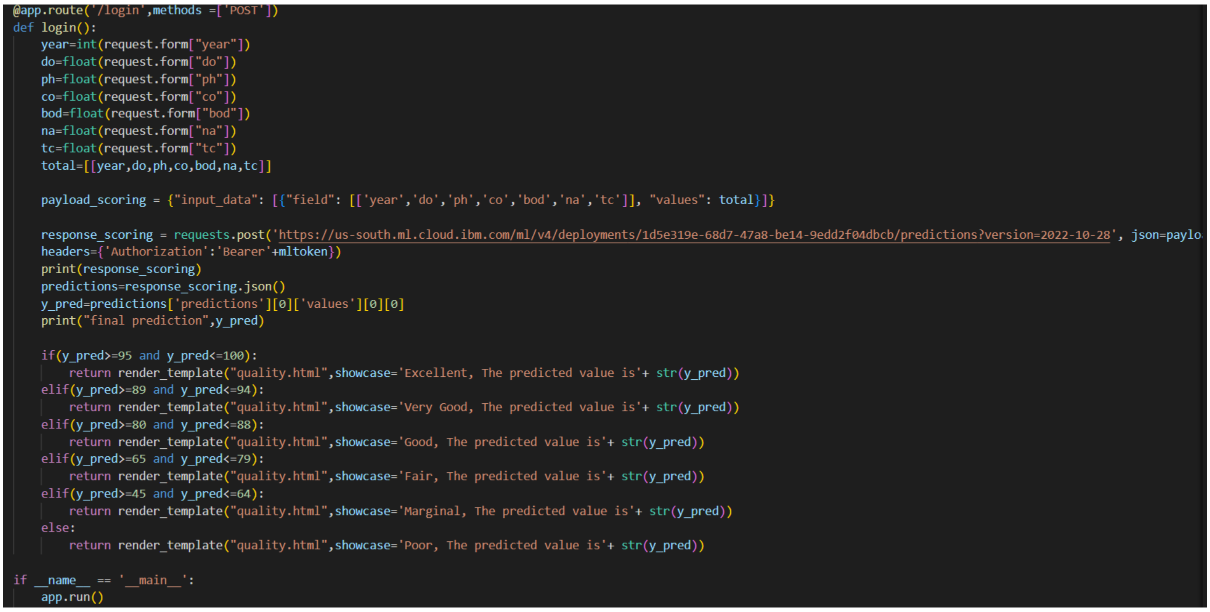
Team ID	PNT2022TMID17967
Project Name	Project – Efficient Water Quality Analysis and Prediction using Machine Learning

Finding Water Quality Level:

Finding the water quality level based on the predicted output of the model.



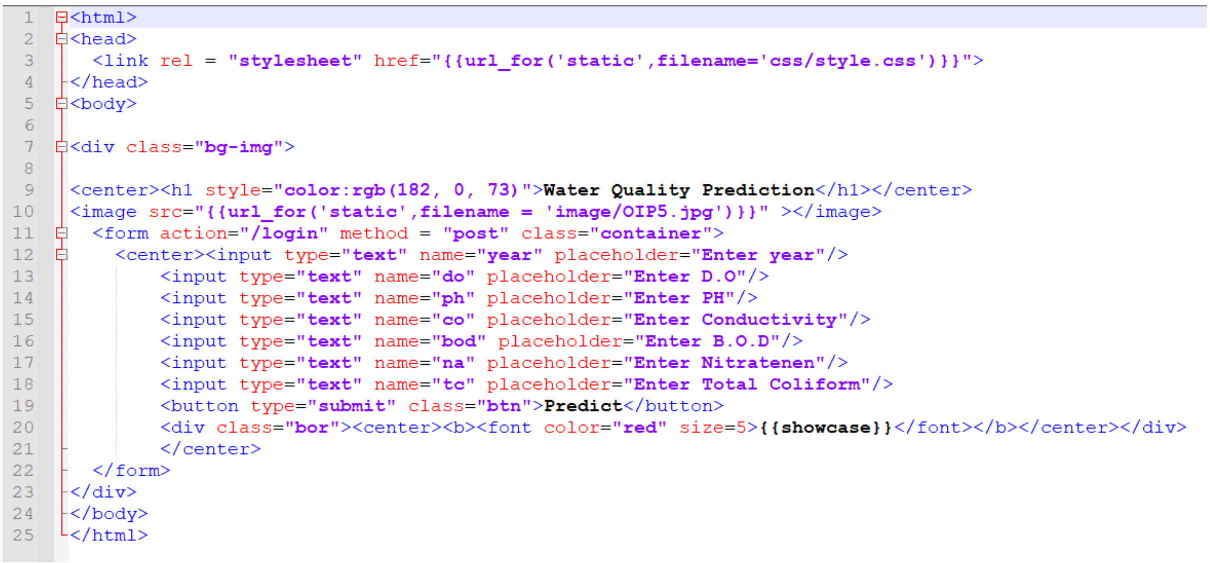
```python
import joblib
import numpy as np
import flask
#from flask_core import CORS
from flask import render_template,request
#import pickle

import requests

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
API_KEY = "8HrhmjnJlOEE0j_lz0tsFmxjw0_xq882lNskkHHnPe9Z"
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}



app = flask.Flask(__name__,static_url_path='')
#CORS(app)
#model = pickle.load(open('wqi1.pkl','rb'))
@app.route('/',methods=['GET'])
def home() :
    return render_template("quality.html")
```



```python
@app.route('/login',methods =['POST'])
def login():
    year=int(request.form["year"])
    do=float(request.form["do"])
    ph=float(request.form["ph"])
    co=float(request.form["co"])
    bod=float(request.form["bod"])
    na=float(request.form["na"])
    tc=float(request.form["tc"])
    total=[[year,do,ph,co,bod,na,tc]]

    payload_scoring = {"input_data": [{"field": [['year','do','ph','co','bod','na','tc']], "values": total}]}

    response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/1d5e319e-68d7-47a8-be14-9edd2f04dbcb/predictions?version=2022-10-28', json=paylo
    headers={'Authorization':'Bearer'+mltoken})
    print(response_scoring)
    predictions=response_scoring.json()
    y_pred=predictions['predictions'][0]['values'][0][0]
    print("final prediction",y_pred)

    if(y_pred>=95 and y_pred<=100):
        return render_template("quality.html",showcase='Excellent, The predicted value is'+ str(y_pred))
    elif(y_pred>=89 and y_pred<=94):
        return render_template("quality.html",showcase='Very Good, The predicted value is'+ str(y_pred))
    elif(y_pred>=80 and y_pred<=88):
        return render_template("quality.html",showcase='Good, The predicted value is'+ str(y_pred))
    elif(y_pred>=65 and y_pred<=79):
        return render_template("quality.html",showcase='Fair, The predicted value is'+ str(y_pred))
    elif(y_pred>=45 and y_pred<=64):
        return render_template("quality.html",showcase='Marginal, The predicted value is'+ str(y_pred))
    else:
        return render_template("quality.html",showcase='Poor, The predicted value is'+ str(y_pred))

if __name__ == '__main__':
    app.run()
```

User Interface (HTML Page):

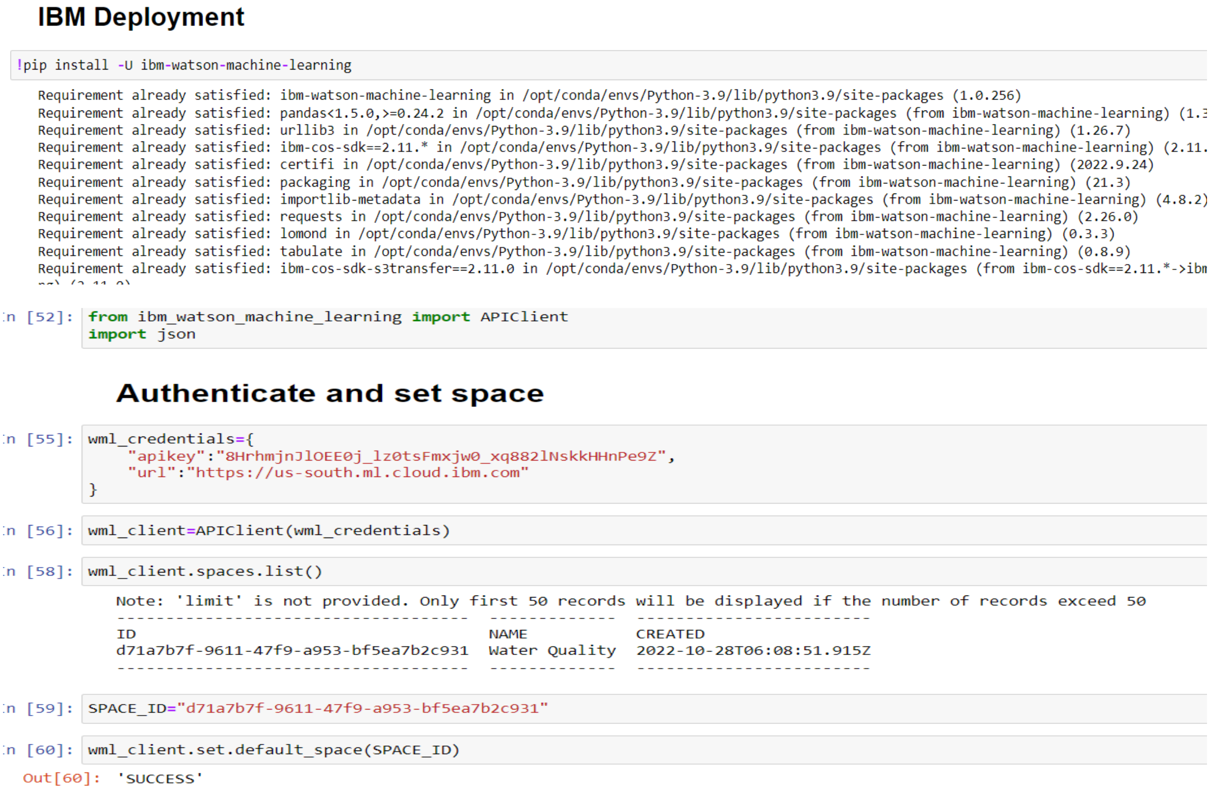
Designing the user interface to get input from user and show the WQI and water quality level.



```html
<html>
<head>
  <link rel = "stylesheet" href="{{url_for('static',filename='css/style.css')}}">
</head>
<body>

<div class="bg-img">

<center><h1 style="color:rgb(182, 0, 73)">Water Quality Prediction</h1></center>
<image src="{{url_for('static',filename = 'image/OIP5.jpg')}}" ></image>
  <form action="/login" method = "post" class="container">
    <center><input type="text" name="year" placeholder="Enter year"/>
        <input type="text" name="do" placeholder="Enter D.O"/>
        <input type="text" name="ph" placeholder="Enter PH"/>
        <input type="text" name="co" placeholder="Enter Conductivity"/>
        <input type="text" name="bod" placeholder="Enter B.O.D"/>
        <input type="text" name="na" placeholder="Enter Nitratenen"/>
        <input type="text" name="tc" placeholder="Enter Total Coliform"/>
        <button type="submit" class="btn">Predict</button>
        <div class="bor"><center><b><font color="red" size=5>{{showcase}}</font></b></center></div>
        </center>
  </form>
</div>
</body>
</html>
```

Connecting with Cloud:

The completed module is shifted into cloud.



IBM Deployment

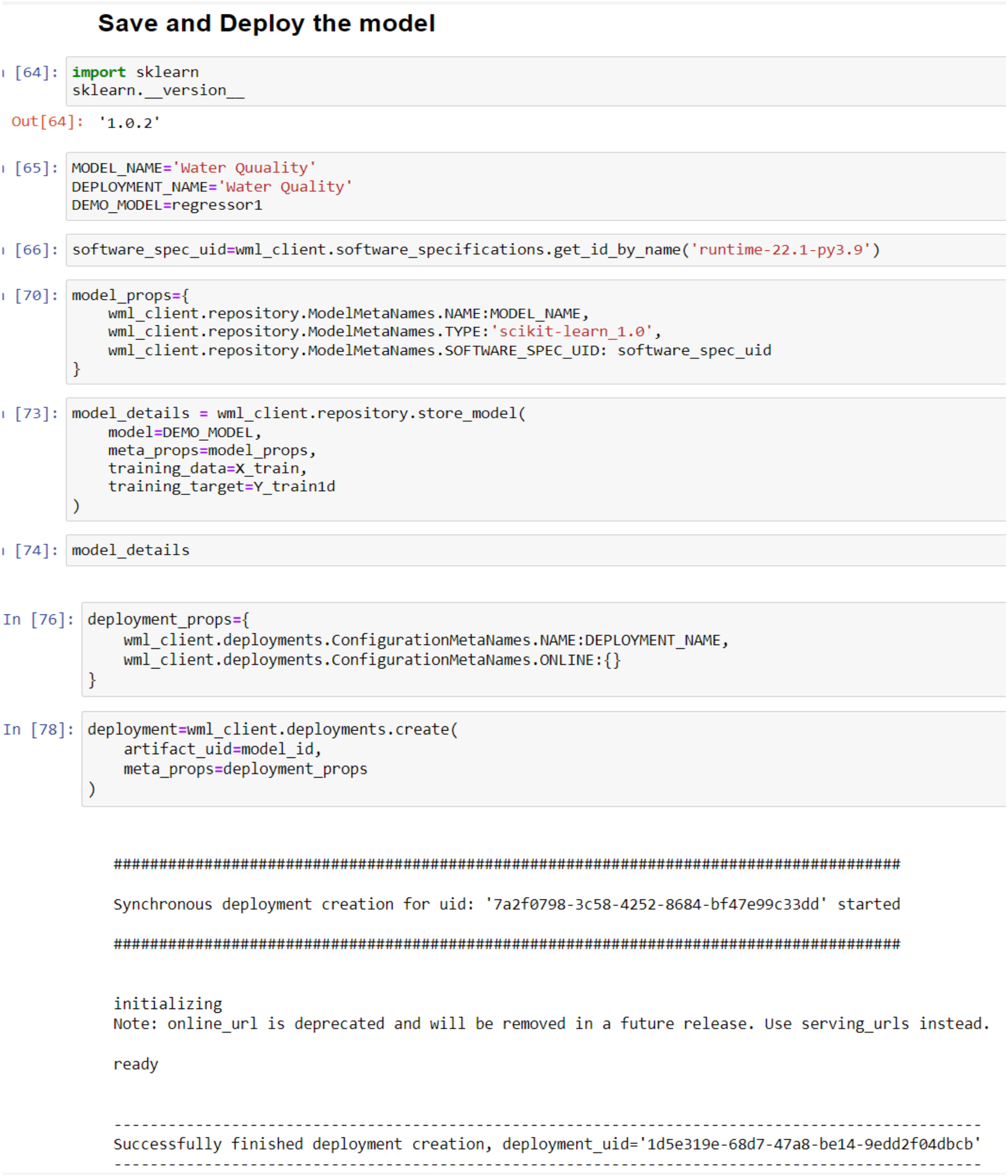
```
!pip install -U ibm-watson-machine-learning
```

```
Requirement already satisfied: ibm-watson-machine-learning in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.0.256)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (1.3
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (1.26.7)
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (2.11.
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (2022.9.24)
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (21.3)
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (4.8.2)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (2.26.0)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (0.3.3)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (0.8.9)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm
```

```
:n [52]: from ibm_watson_machine_learning import APIClient
         import json
```

Authenticate and set space

```
:n [55]: wml_credentials={
             "apikey":"8HrhmjnJlOEE0j_lz0tsFmxjw0_xq882lNskkHHnPe9Z",
             "url":"https://us-south.ml.cloud.ibm.com"
         }

:n [56]: wml_client=APIClient(wml_credentials)

:n [58]: wml_client.spaces.list()

            Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50
            ------------------------------------  -------------  ------------------------
            ID                                    NAME           CREATED
            d71a7b7f-9611-47f9-a953-bf5ea7b2c931  Water Quality  2022-10-28T06:08:51.915Z
            ------------------------------------  -------------  ------------------------

:n [59]: SPACE_ID="d71a7b7f-9611-47f9-a953-bf5ea7b2c931"

:n [60]: wml_client.set_default_space(SPACE_ID)
  Out[60]: 'SUCCESS'
```



Save and Deploy the model

```
ı [64]: import sklearn
        sklearn.__version__

Out[64]: '1.0.2'
```

```
ı [65]: MODEL_NAME='Water Quuality'
        DEPLOYMENT_NAME='Water Quality'
        DEMO_MODEL=regressor1
```

```
ı [66]: software_spec_uid=wml_client.software_specifications.get_id_by_name('runtime-22.1-py3.9')
```

```
ı [70]: model_props={
            wml_client.repository.ModelMetaNames.NAME:MODEL_NAME,
            wml_client.repository.ModelMetaNames.TYPE:'scikit-learn_1.0',
            wml_client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
        }
```

```
ı [73]: model_details = wml_client.repository.store_model(
            model=DEMO_MODEL,
            meta_props=model_props,
            training_data=X_train,
            training_target=Y_train1d
        )
```

```
ı [74]: model_details
```

```
In [76]: deployment_props={
             wml_client.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,
             wml_client.deployments.ConfigurationMetaNames.ONLINE:{}
         }
```

```
In [78]: deployment=wml_client.deployments.create(
             artifact_uid=model_id,
             meta_props=deployment_props
         )
```

```
#######################################################################################

Synchronous deployment creation for uid: '7a2f0798-3c58-4252-8684-bf47e99c33dd' started

#######################################################################################


initializing
Note: online_url is deprecated and will be removed in a future release. Use serving_urls instead.

ready


------------------------------------------------------------------------------------------------
Successfully finished deployment creation, deployment_uid='1d5e319e-68d7-47a8-be14-9edd2f04dbcb'
------------------------------------------------------------------------------------------------
```