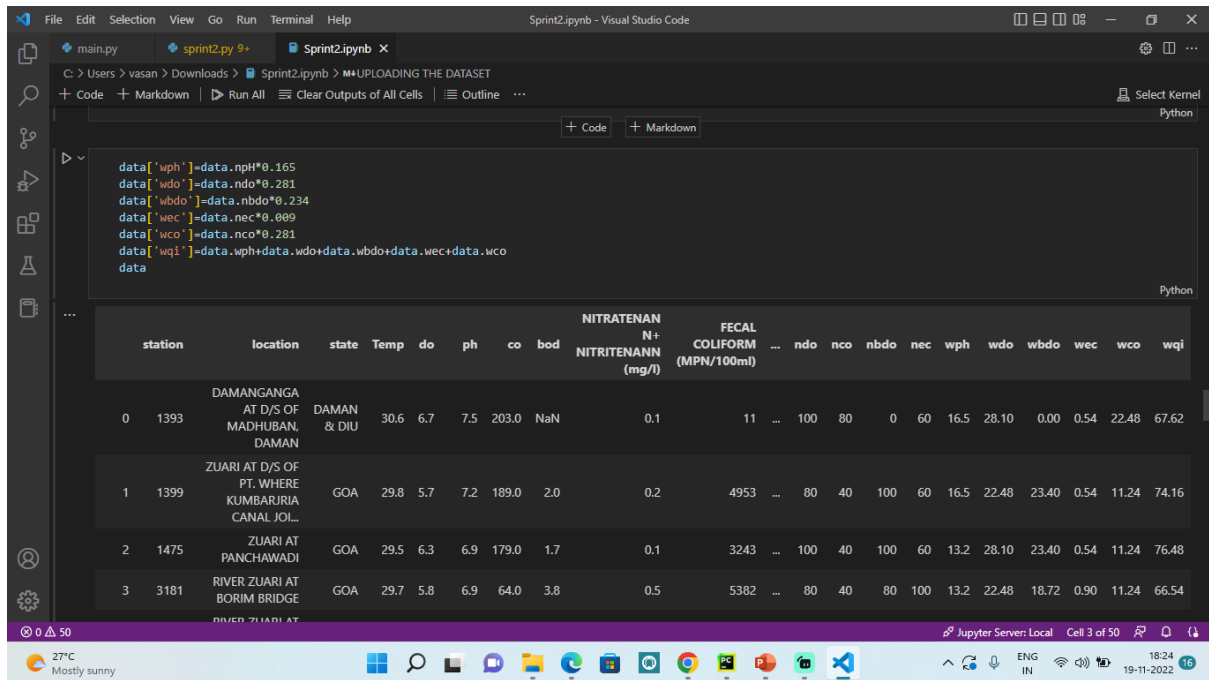


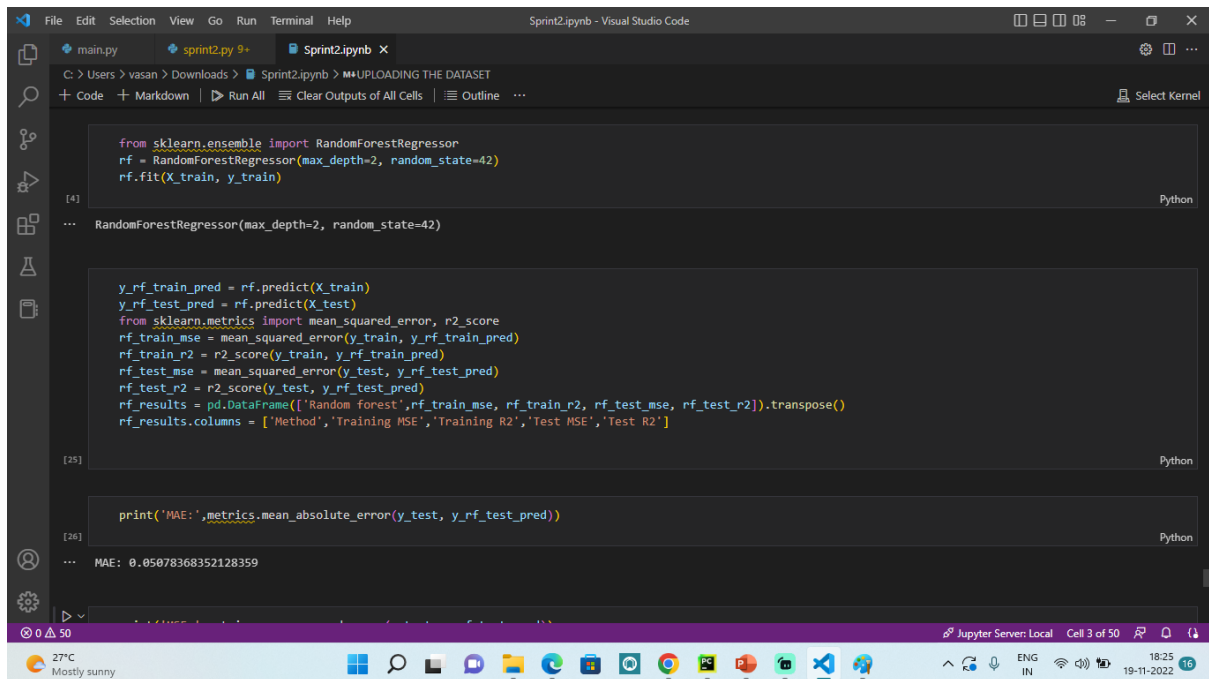
INTEGRATE FLASK WITH SCORING END POINT



The screenshot shows a Jupyter Notebook cell in Visual Studio Code. The code defines a function to preprocess data by scaling certain features. Below the code, a preview of the dataset is shown as a table.

```
data['wph'] = data.nph * 0.165
data['wdo'] = data.ndo * 0.281
data['wbdo'] = data.nbdo * 0.234
data['wec'] = data.nec * 0.009
data['wco'] = data.nco * 0.281
data['wqi'] = data.wph + data.wdo + data.wbdo + data.wec + data.wco
data
```

	station	location	state	Temp	do	ph	co	bod	NITRATENAN N+ NITRITENANN (mg/l)	FECAL COLIFORM (MPN/100ml)	...	ndo	nco	nbdo	nec	wph	wdo	wbdo	wec	wco	wqi
0	1393	DAMANGANGA AT D/S OF MADHUBAN, DAMAN	DAMAN & DIU	30.6	6.7	7.5	203.0	NaN	0.1	11	...	100	80	0	60	16.5	28.10	0.00	0.54	22.48	67.62
1	1399	ZUARI AT D/S OF PT. WHERE KUMBARJIA CANAL JOIL	GOA	29.8	5.7	7.2	189.0	2.0	0.2	4953	...	80	40	100	60	16.5	22.48	23.40	0.54	11.24	74.16
2	1475	ZUARI AT PANCHAWADI	GOA	29.5	6.3	6.9	179.0	1.7	0.1	3243	...	100	40	100	60	13.2	28.10	23.40	0.54	11.24	76.48
3	3181	RIVER ZUARI AT BORIM BRIDGE	GOA	29.7	5.8	6.9	64.0	3.8	0.5	5382	...	80	40	80	100	13.2	22.48	18.72	0.90	11.24	66.54



The screenshot shows a Jupyter Notebook cell in Visual Studio Code. The code trains a Random Forest Regressor model and evaluates its performance using Mean Squared Error (MSE) and Mean Absolute Error (MAE).

```
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor(max_depth=2, random_state=42)
rf.fit(X_train, y_train)

RandomForestRegressor(max_depth=2, random_state=42)

y_rf_train_pred = rf.predict(X_train)
y_rf_test_pred = rf.predict(X_test)
from sklearn.metrics import mean_squared_error, r2_score
rf_train_mse = mean_squared_error(y_train, y_rf_train_pred)
rf_train_r2 = r2_score(y_train, y_rf_train_pred)
rf_test_mse = mean_squared_error(y_test, y_rf_test_pred)
rf_test_r2 = r2_score(y_test, y_rf_test_pred)
rf_results = pd.DataFrame(['Random forest', rf_train_mse, rf_train_r2, rf_test_mse, rf_test_r2]).transpose()
rf_results.columns = ['Method', 'Training MSE', 'Training R2', 'Test MSE', 'Test R2']

print('MAE:', metrics.mean_absolute_error(y_test, y_rf_test_pred))
```

MAE: 0.05078368352128359

File Edit Selection View Go Run Terminal Help Sprint2.ipynb - Visual Studio Code

C:\Users> vasan > Downloads > Sprint2.ipynb > M4UPLOADING THE DATASET

+ Code + Markdown + Run All Clear Outputs of All Cells Outline ... Select Kernel

```
[25]
```

```
[26] print('MAE:', metrics.mean_absolute_error(y_test, y_rf_test_pred))
```

```
... MAE: 0.05078368352128359
```

```
[24] print('MSE:', metrics.mean_squared_error(y_test, y_rf_test_pred))
```

```
... MSE: 0.00709044752030462
```

+ Code + Markdown

```
[29] import numpy as np
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_rf_test_pred)))
```

```
... RMSE: 0.08428479511467634
```

ACCURACY OF THE MODEL

```
[30] metrics.r2_score(y_test, y_rf_test_pred)
```

Python

0 50

27°C Mostly sunny

Jupyter Server: Local Cell 3 of 50

18:25 19-11-2022

File Edit Selection View Go Run Terminal Help Sprint2.ipynb - Visual Studio Code

C:\Users> vasan > Downloads > Sprint2.ipynb > M4UPLOADING THE DATASET

+ Code + Markdown + Run All Clear Outputs of All Cells Outline ... Select Kernel

```
[54]
```

SAVE THE MODEL

```
import joblib
from sklearn.ensemble import RandomForestRegressor
from sklearn.datasets import make_regression
X, y = make_regression(n_features=4, n_informative=2, random_state=0, shuffle=False)
rfr = RandomForestRegressor(max_depth=3)
rfr.fit(X, y)
print(rfr.predict([[0, 1, 0, 1]]))
joblib.dump(rfr, "my_random_forest.joblib")
loaded_rfr = joblib.load("my_random_forest.joblib")
```

```
[53]
```

```
... [34.83545299]
```

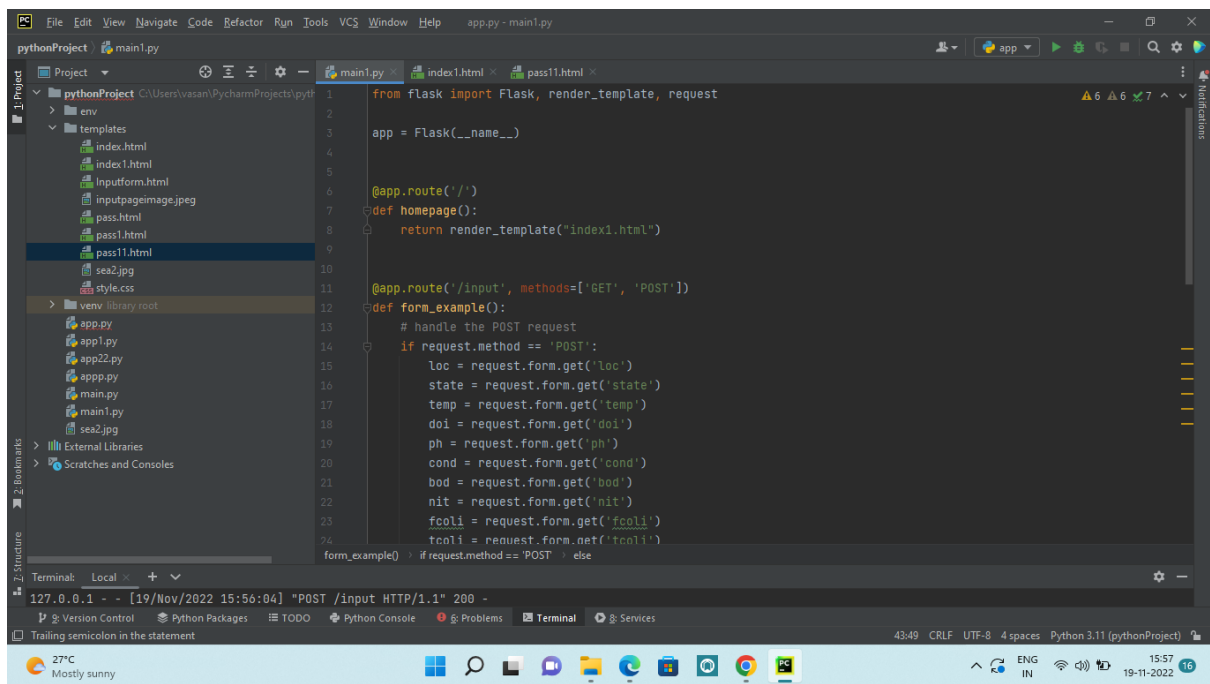
Python

0 50

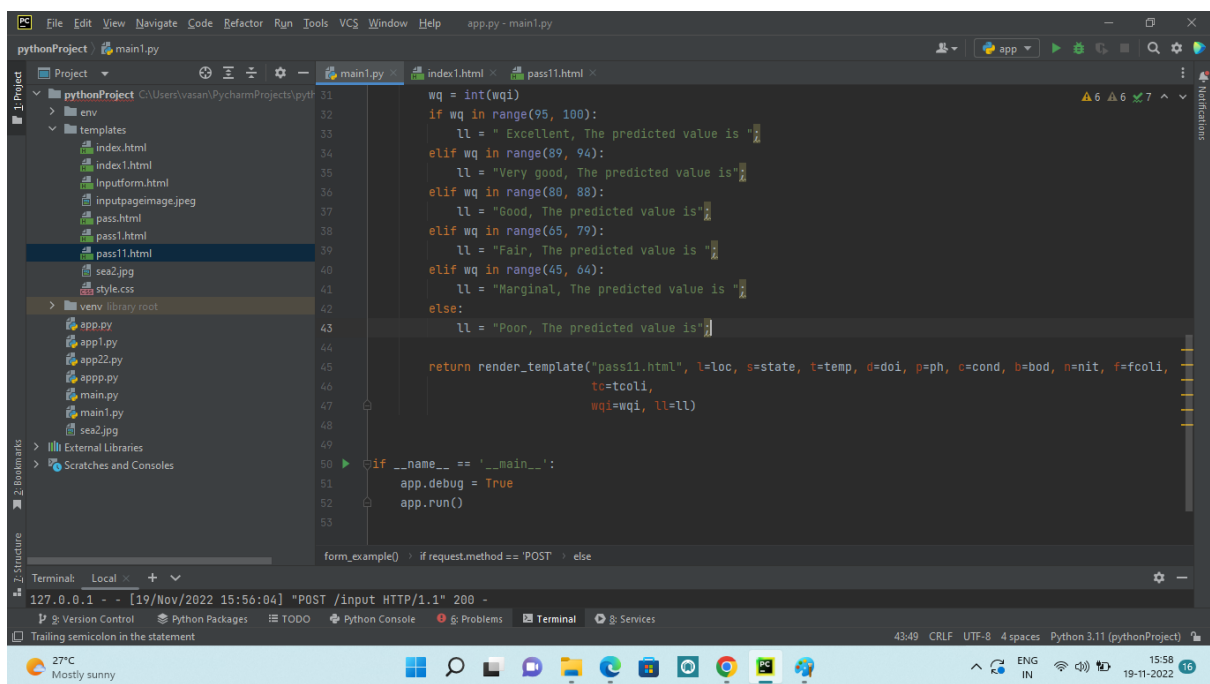
27°C Mostly sunny

Jupyter Server: Local Cell 3 of 50

18:26 19-11-2022



```
1 from flask import Flask, render_template, request
2
3 app = Flask(__name__)
4
5 @app.route('/')
6 def homepage():
7     return render_template("index1.html")
8
9
10 @app.route('/input', methods=['GET', 'POST'])
11 def form_example():
12     # handle the POST request
13     if request.method == 'POST':
14         loc = request.form.get('loc')
15         state = request.form.get('state')
16         temp = request.form.get('temp')
17         doi = request.form.get('doi')
18         ph = request.form.get('ph')
19         cond = request.form.get('cond')
20         bod = request.form.get('bod')
21         nit = request.form.get('nit')
22         fcoli = request.form.get('fcoli')
23         fcoli = request.form.get('fcoli')
24     if request.method == 'POST' else
```



```
31 wqi = int(wqi)
32 if wqi in range(95, 100):
33     ll = "Excellent, The predicted value is "
34 elif wqi in range(89, 94):
35     ll = "Very good, The predicted value is "
36 elif wqi in range(80, 88):
37     ll = "Good, The predicted value is "
38 elif wqi in range(65, 79):
39     ll = "Fair, The predicted value is "
40 elif wqi in range(45, 64):
41     ll = "Marginal, The predicted value is "
42 else:
43     ll = "Poor, The predicted value is "
44
45 return render_template("pass11.html", l=loc, s=state, t=temp, d=doi, p=ph, c=cond, b=bod, n=nit, f=fcoli,
46                        tc=tccli,
47                        wqi=wqi, ll=ll)
48
49
50 if __name__ == '__main__':
51     app.debug = True
52     app.run()
53
54 form_example() > if request.method == 'POST' > else
```

S

The screenshot shows an IDE with a project named 'pythonProject'. The file explorer on the left shows a 'templates' folder containing 'index.html', 'inputform.html', 'inputpageimage.jpeg', 'pass1.html', 'pass11.html', 'sea2.jpg', and 'style.css'. The 'pass11.html' file is open in the editor, showing HTML code for a form with fields for TEMPERATURE, D.O., pH, CONDUCTIVITY, B.O.D., NITRATENANN, and FECALCOLIFORM. A button with id='ll' is also present. The terminal at the bottom shows the command 'python main1.py' being executed, and the Flask server is running on http://127.0.0.1:5000. The status bar at the bottom indicates the file is encoded in UTF-8 with 4 spaces, using Python 3.11.

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help app.py - pass11.html
pythonProject templates pass11.html
Project
pythonProject C:\Users\vasan\PycharmProjects\pythonProject
env
templates
index.html
index1.html
inputform.html
inputpageimage.jpeg
pass.html
pass1.html
pass11.html
sea2.jpg
style.css
venv library root
main1.py index1.html pass11.html
11 <h1>TEMPERATURE = {{t}}</h1>
12 <h1>D.O = {{d}}</h1>
13 <h1>pH = {{p}}</h1>
14 <h1>CONDUCTIVITY = {{c}}</h1>
15 <h1>B.O.D = {{b}} </h1>
16 <h1>NITRATENANN = {{n}}</h1>
17 <h1>FECALCOLIFORM = {{f}}</h1>
18 <h1 onclick="funcctig()">TOTALCOLIFORM = {{tc}}</h1>
19
20 <h1 id="ll"> </h1>
21
html body center h1
Terminal Local +
127.0.0.1 - - [19/Nov/2022 15:56:04] "POST /input HTTP/1.1" 200 -
PS C:\Users\vasan\PycharmProjects\pythonProject> python main1.py
* Serving Flask app 'main1'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 363-249-780
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download onc... (38 minutes ago) 22:5 CRLF UTF-8 4 spaces Python 3.11 (pythonProject)
27°C Mostly sunny 16:00 19-11-2022
```