

Document

Week4: Deployment on Flask

Name: [Fatimah Asiri](#)

Username: alassirifatima@gmail.com

Batch code: LISUM14 30 Sep – 30 Dec 2022

Submission Date: 10/23/2022

Submitted to: [Data Glacier](#).

This document contains a snapshot of each step
of the deployment

(Deployment Machine Learning on Flask)

ML Model Deployment using Flask

1. Build ML Model.
2. Deploy using Flask.

Files to be created

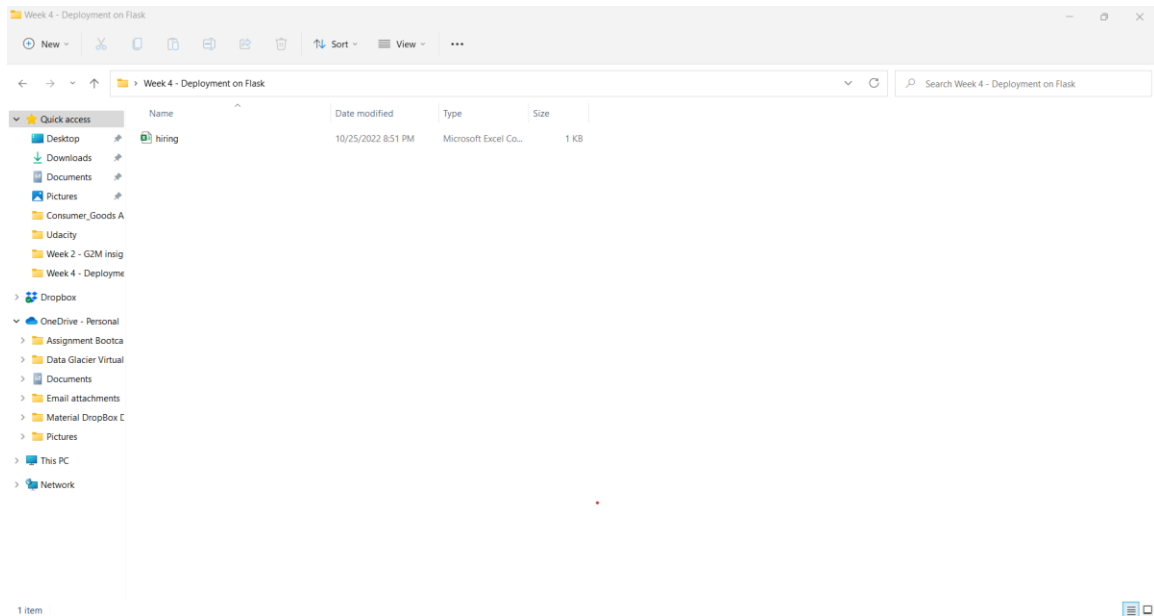
1. model.py (ML model)
2. model.pkl (Pickle file of ML model)
3. app.py (Flask Application)
4. index.html (inside the folder templates)
5. hiring dataset (data to build ML model)
6. request.py ()

1. Choose a toy dataset.

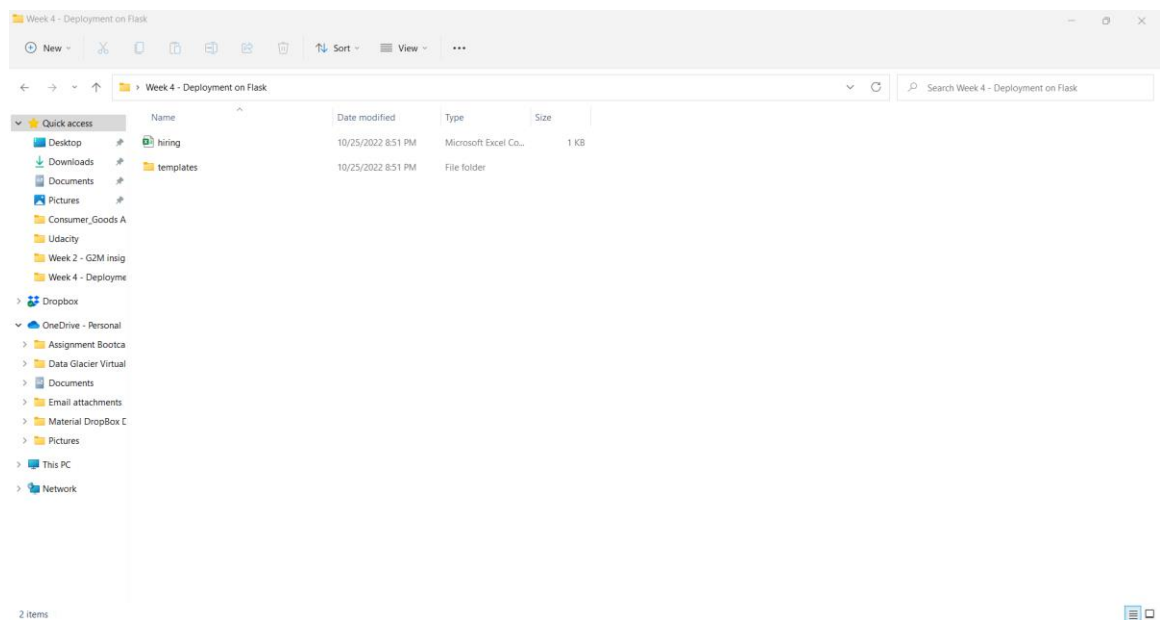
Select the hiring dataset.

| experience | test_score | interview | salary |
|------------|------------|-----------|--------|
| 8 | 9 | | 50000 |
| 8 | 6 | | 45000 |
| five | 6 | 7 | 60000 |
| two | 10 | 10 | 65000 |
| seven | 9 | 6 | 70000 |
| three | 7 | 10 | 62000 |
| ten | 7 | 7 | 72000 |
| eleven | 7 | 8 | 80000 |

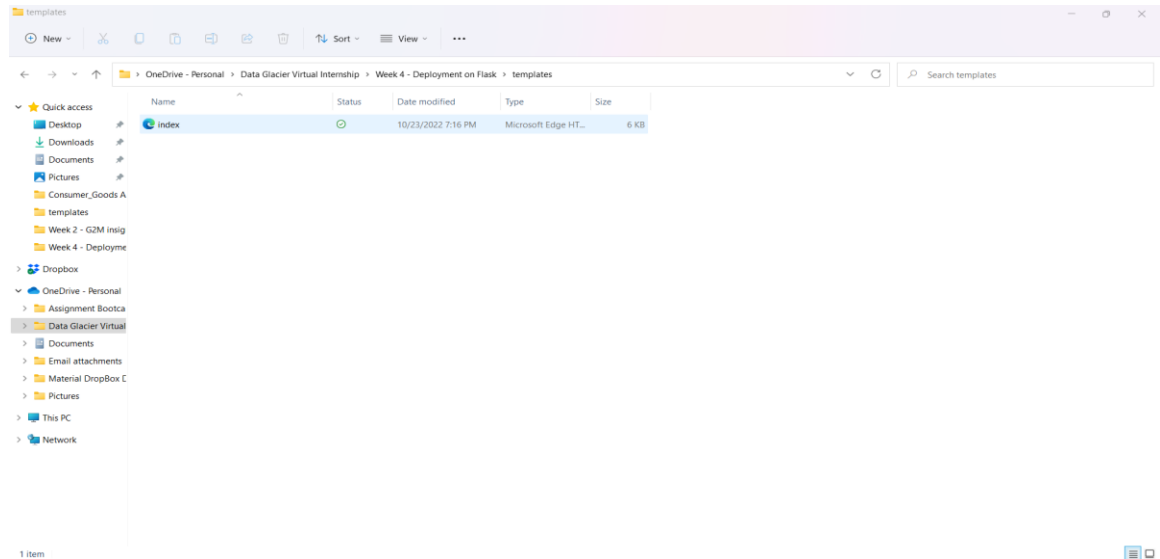
2. Download csv file inside the folder



3. Create new folder (templates).

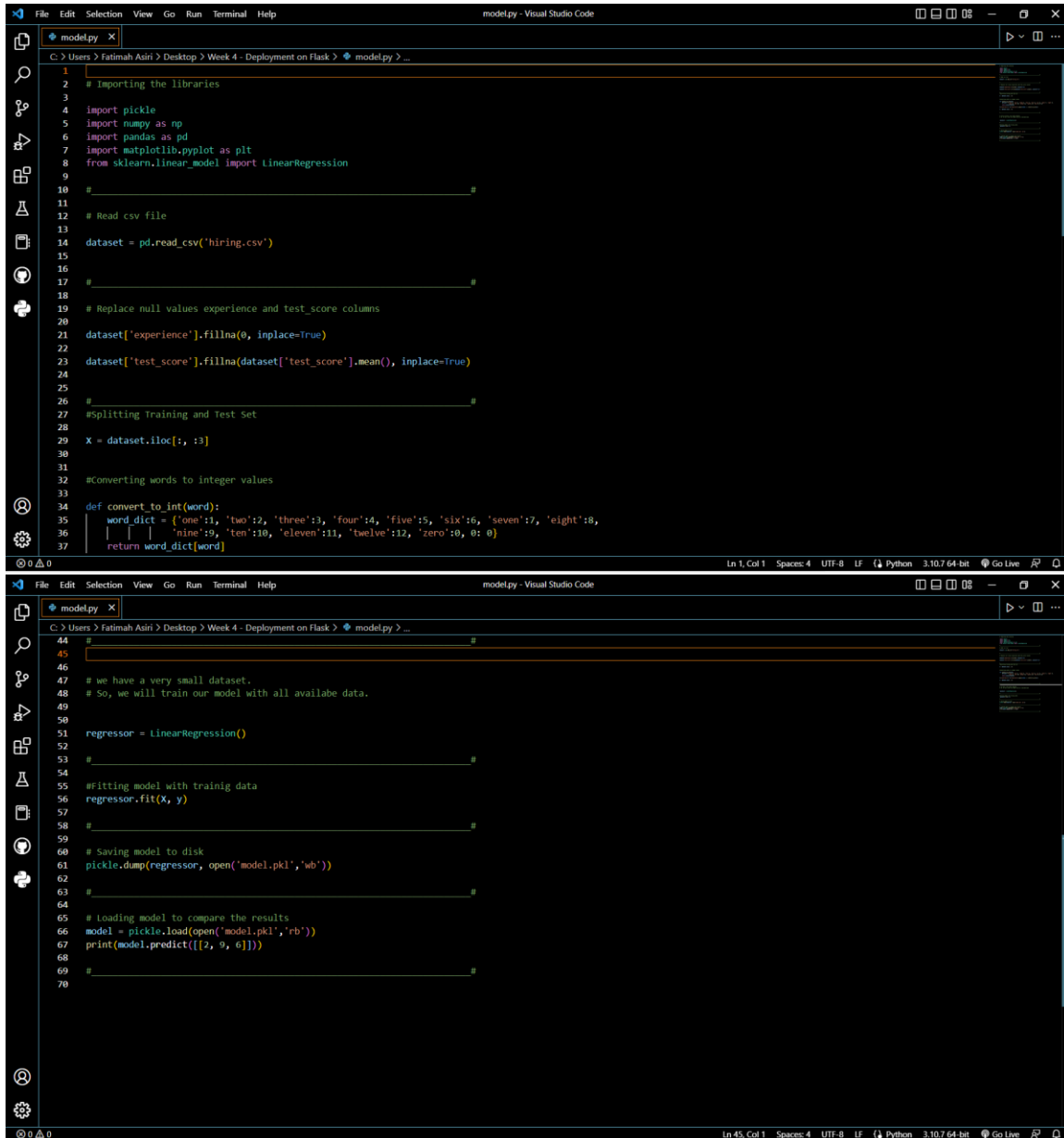


4. Create index.html file in the templates folder.



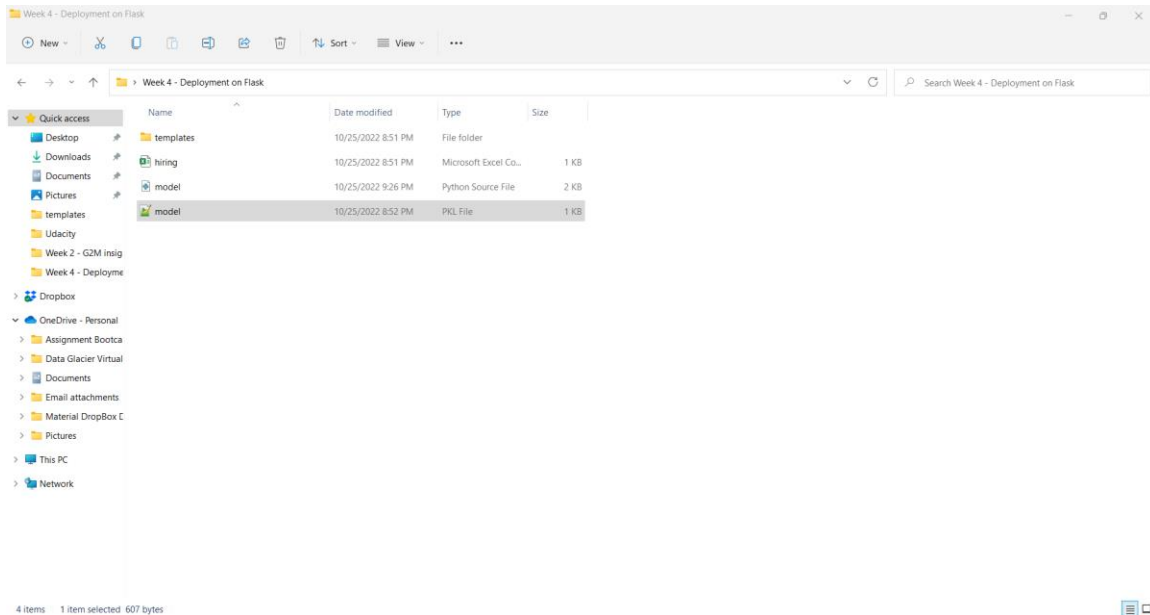
```
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
C:\Users\Fatimah Asiri\OneDrive\Data Glacier Virtual Internship\Week 4 - Deployment on Flask\templates\index.html - Notepad++
index.html
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5
6
7 <title>Predict Salary Analysis on Flask</title>
8
9
10 </head>
11
12 <body bgcolor="#F0F0F0">
13 <div>
14
15 <center>
16 <h1>Predict Salary Analysis</h1>
17 </center>
18
19 <hr>
20
21 <!-- Main Input For Receiving Query to our ML -->
22
23 <center>
24 <form action="{{ url_for('predict')}}" method="post">
25
26 <br>
27 <br>
28 <input type="text" name="experience" placeholder="Experience" required="required" />
29
30 <br>
31 <br>
32 <input type="text" name="test_score" placeholder="Test Score" required="required" />
33
34 <br>
35 <br>
36 <input type="text" name="interview_score" placeholder="Interview Score" required="required" />
37
38 <br>
39 <br>
40 <br>
41 <br>
42 <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
43
44
```

5. Create model.py file

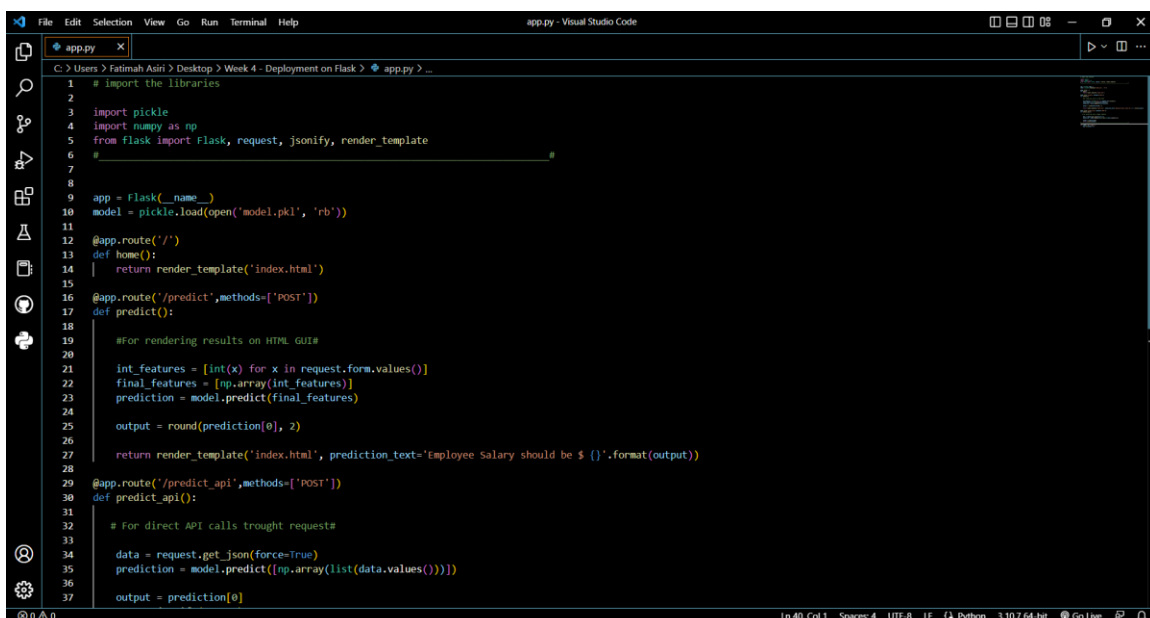


```
1 # Importing the libraries
2 import pickle
3 import numpy as np
4 import pandas as pd
5 import matplotlib.pyplot as plt
6 from sklearn.linear_model import LinearRegression
7
8 # Read csv file
9 dataset = pd.read_csv('hiring.csv')
10
11 # Replace null values experience and test_score columns
12 dataset['experience'].fillna(0, inplace=True)
13 dataset['test_score'].fillna(dataset['test_score'].mean(), inplace=True)
14
15 # Splitting Training and Test Set
16 X = dataset.iloc[:, :3]
17
18 # Converting words to integer values
19 def convert_to_int(word):
20     word_dict = {'one':1, 'two':2, 'three':3, 'four':4, 'five':5, 'six':6, 'seven':7, 'eight':8,
21                 'nine':9, 'ten':10, 'eleven':11, 'twelve':12, 'zero':0, 0: 0}
22     return word_dict[word]
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47 # we have a very small dataset.
48 # So, we will train our model with all available data.
49
50 regressor = LinearRegression()
51
52
53
54 # Fitting model with training data
55 regressor.fit(X, y)
56
57
58
59 # Saving model to disk
60 pickle.dump(regressor, open('model.pkl', 'wb'))
61
62
63
64
65 # Loading model to compare the results
66 model = pickle.load(open('model.pkl', 'rb'))
67 print(model.predict([[2, 9, 6]]))
68
69
70
```

6. Create Pickle file of our model model.pkl




7. Create app.py files





```
28 | return request_json['test_score'], prediction_text, employee_salary, insurance_cost, format(output)
29 |
30 | @app.route('/predict_api', methods=['POST'])
31 | def predict_api():
32 |     # For direct API calls through request#
33 |
34 |     data = request.get_json(force=True)
35 |     prediction = model.predict([np.array(list(data.values()))])
36 |
37 |     output = prediction[0]
38 |     return jsonify(output)
39 |
40 |
41 | if __name__ == "__main__":
42 |     app.run(debug=True)
```

8. Create request.py



```
1 | import requests
2 |
3 | url = 'http://localhost:5000/predict_api'
4 | r = requests.post(url, json={'experience':2, 'test_score':9, 'interview_score':6})
5 |
6 | print(r.json())
```

9. Read me file for GitHub

10. The Result of running app

The image displays two screenshots of a web application titled "Predict Salary Analysis".

The top screenshot shows the initial state of the application. It features three input fields labeled "Experience", "Test Score", and "Interview Score", each with a corresponding "Predict" button below them. The output section displays the name "Fatimah Asiri" and the date range "LISUM 14 30 Sep - 30 Dec 2022".

The bottom screenshot shows the application after a prediction. The output section now includes the text "Employee Salary should be \$ 56770.15" above the name "Fatimah Asiri" and the date range "LISUM 14 30 Sep - 30 Dec 2022".