Model Deployment

Name: Motamen MohammedAhmed

Batch Code: LISUM03

Submission Date: 12/09/2021

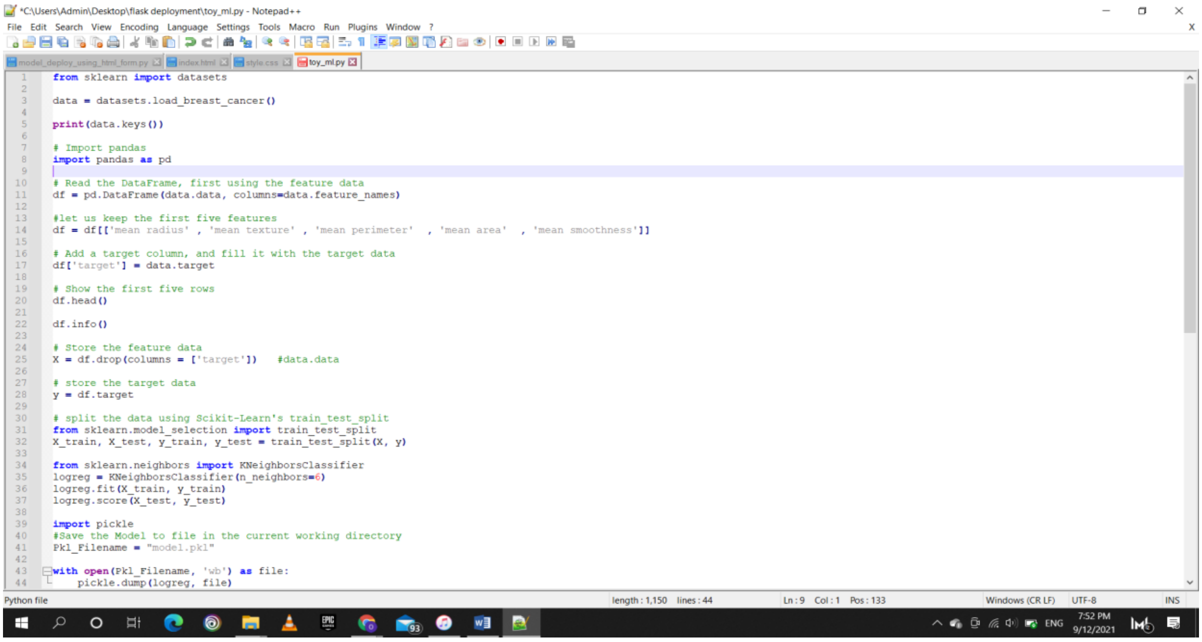
Submitted to: Data Glacier

Introduction:

These are the steps required to deploy a machine learning model on the web using Flask micro-webserver. Each step is provided with a screenshot.

Step 1:

Find a toy dataset and build your model. We chose the breast cancer Wisconsin dataset.

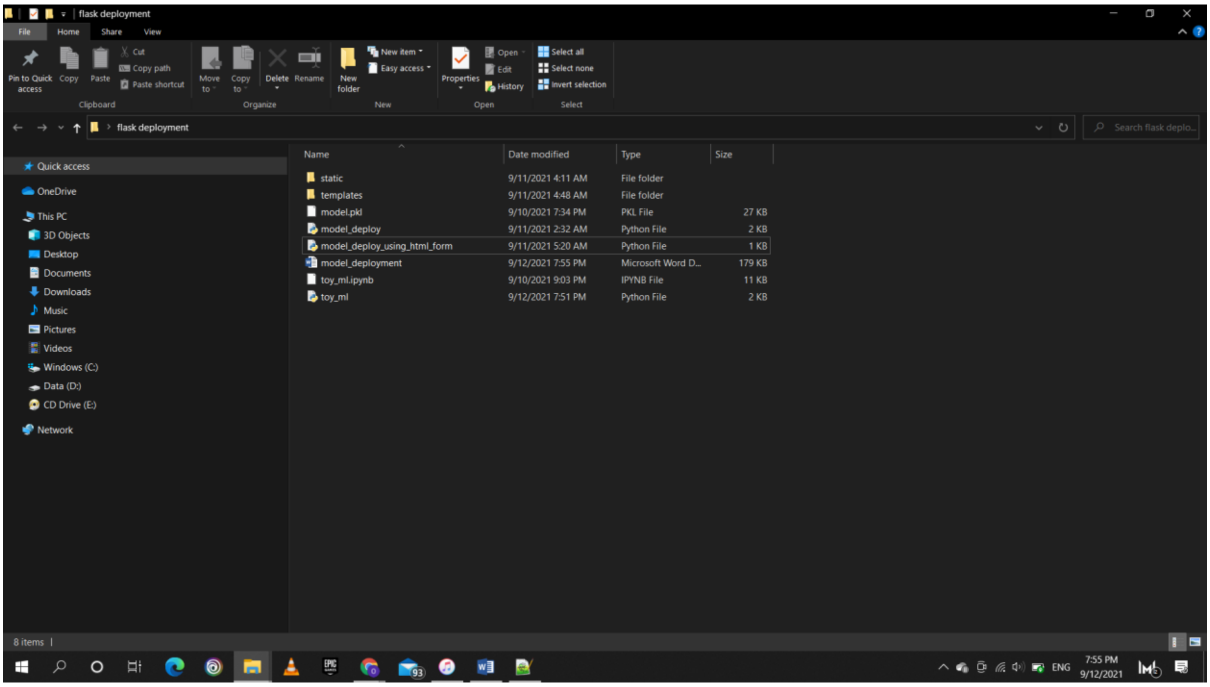


*C:\Users\Admin\Desktop\flask deployment\toy_ml.py - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
model_deploy_using_html_form.py
index.html
style.css
toy_ml.py
Python file
length : 1,150 lines : 44
Ln : 9 Col : 1 Pos : 133
Windows (CR LF)
UTF-8
INS
7:52 PM
9/12/2021

```python
from sklearn import datasets

data = datasets.load_breast_cancer()

print(data.keys())

# Import pandas
import pandas as pd

# Read the DataFrame, first using the feature data
df = pd.DataFrame(data.data, columns=data.feature_names)

#let us keep the first five features
df = df[['mean radius' , 'mean texture' , 'mean perimeter'  , 'mean area'  , 'mean smoothness']]

# Add a target column, and fill it with the target data
df['target'] = data.target

# Show the first five rows
df.head()

df.info()

# Store the feature data
X = df.drop(columns = ['target'])   #data.data

# store the target data
y = df.target

# split the data using Scikit-Learn's train_test_split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y)

from sklearn.neighbors import KNeighborsClassifier
logreg = KNeighborsClassifier(n_neighbors=6)
logreg.fit(X_train, y_train)
logreg.score(X_test, y_test)

import pickle
#Save the Model to file in the current working directory
Pkl_Filename = "model.pkl"

with open(Pkl_Filename, 'wb') as file:
    pickle.dump(logreg, file)
```

Step 2:

Save the model using pickle module. The model is saved as model.pkl file.



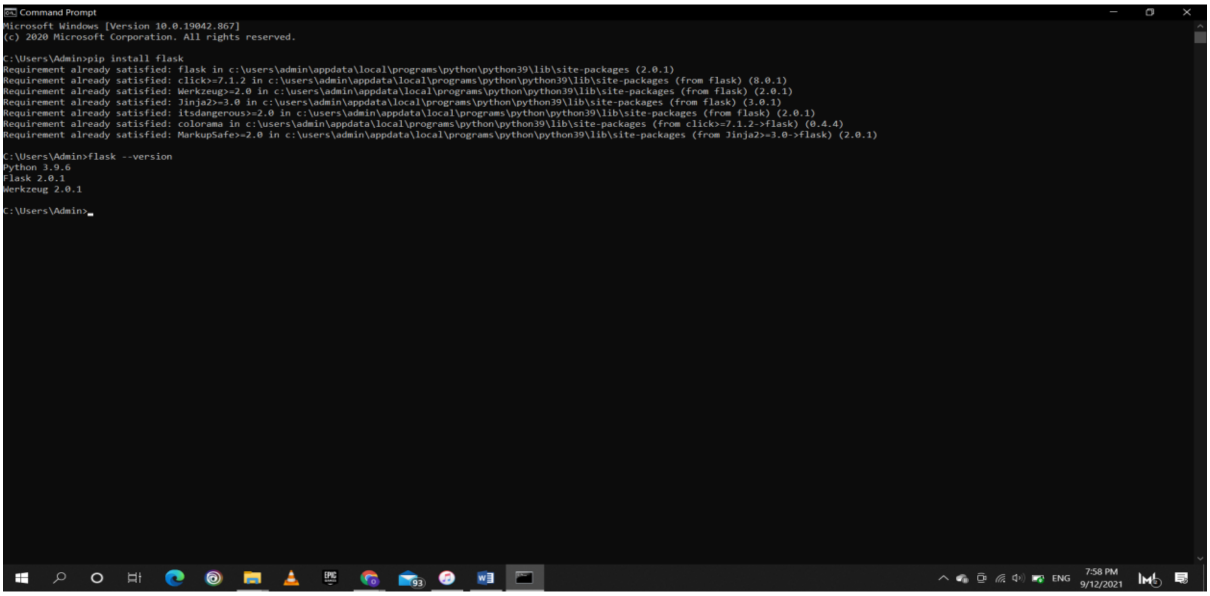
flask deployment

Name	Date modified	Type	Size
static	9/11/2021 4:11 AM	File folder	
templates	9/11/2021 4:48 AM	File folder	
model.pkl	9/10/2021 7:34 PM	PKL File	27 KB
model_deploy	9/11/2021 2:32 AM	Python File	2 KB
model_deploy_using_html_form	9/11/2021 5:20 AM	Python File	1 KB
model_deployment	9/12/2021 7:55 PM	Microsoft Word D...	179 KB
toy_ml.ipynb	9/10/2021 9:03 PM	IPYNB File	11 KB
toy_ml	9/12/2021 7:51 PM	Python File	2 KB

8 items

Step 3:

Install Flask and make sure it is installed successfully. Flask is the core for this deployment to work properly.



Command Prompt

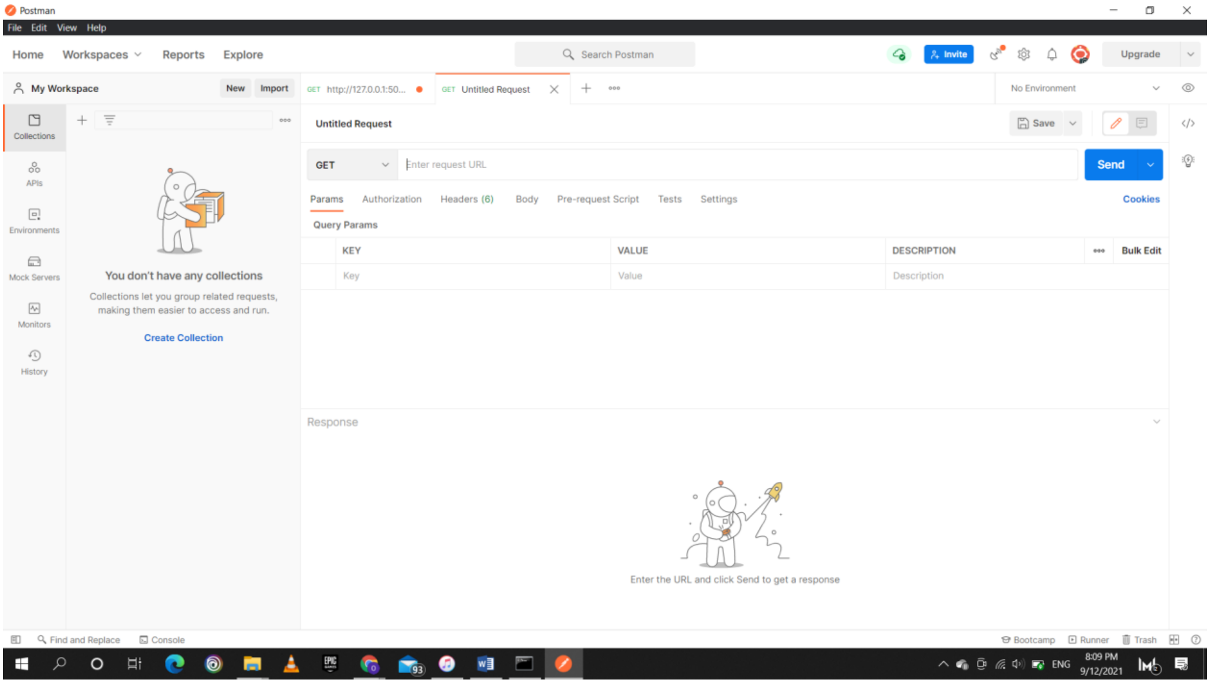
```
Microsoft Windows [Version 10.0.19042.867]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Admin>pip install flask
Requirement already satisfied: flask in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (2.0.1)
Requirement already satisfied: click>=7.1.2 in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (from flask) (8.0.1)
Requirement already satisfied: Werkzeug>=2.0 in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (from flask) (2.0.1)
Requirement already satisfied: Jinja2>=3.0 in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (from flask) (3.0.1)
Requirement already satisfied: itsdangerous>=2.0 in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (from flask) (2.0.1)
Requirement already satisfied: colorama in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (from click>=7.1.2->flask) (0.4.4)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (from Jinja2>=3.0->flask) (2.0.1)

C:\Users\Admin>flask --version
Python 3.9.6
Flask 2.0.1
Werkzeug 2.0.1

C:\Users\Admin>_
```

Deployment Using Postman:

These steps are needed when deploying the model using Postman.

Step 4:

Install and run Postman. Then, navigate to workspace menu.



You don't have any collections

Collections let you group related requests, making them easier to access and run.

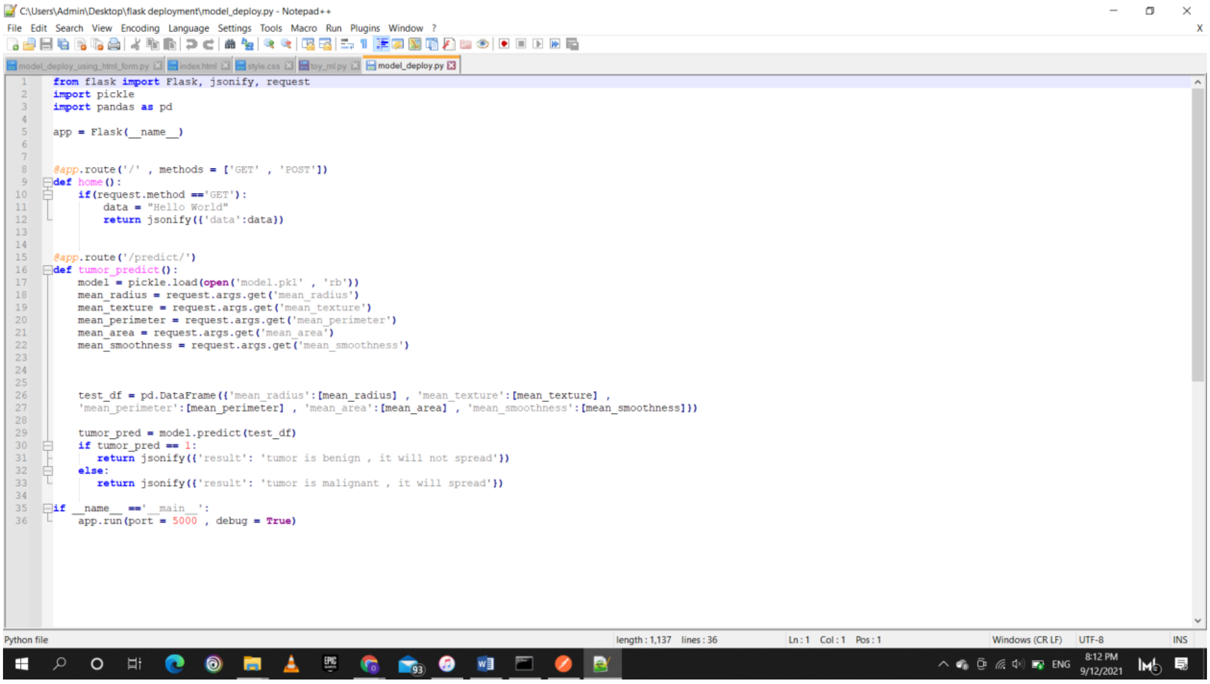
Enter the URL and click Send to get a response

Step 5:

Run the below code. This code does the following:

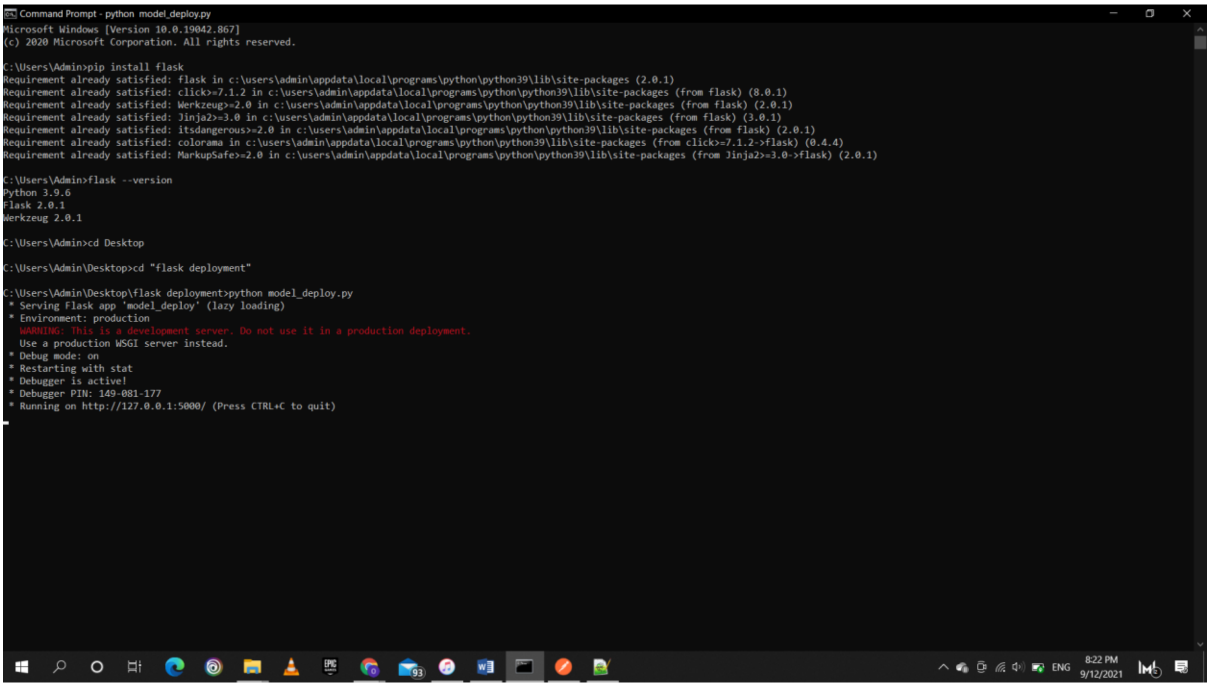
1. Load the model.
2. Create (/) endpoint for the API. This shows only a hello world message.
3. Create (/predict/) endpoint. This endpoint receives the features for predictions from the HTTP request.
4. Run the model and display the message of prediction on the web.

To run the code, type python *file_name* in the CMD. You will see the Flask server up and running.



C:\Users\Admin\Desktop\flask deployment\model_deploy.py - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
model_deploy_using_html_form.py index.html style.css toy_ml.py model_deploy.py
Python file
length : 1,137 lines : 36 Ln : 1 Col : 1 Pos : 1 Windows (CR LF) UTF-8 INS
8:12 PM 9/12/2021

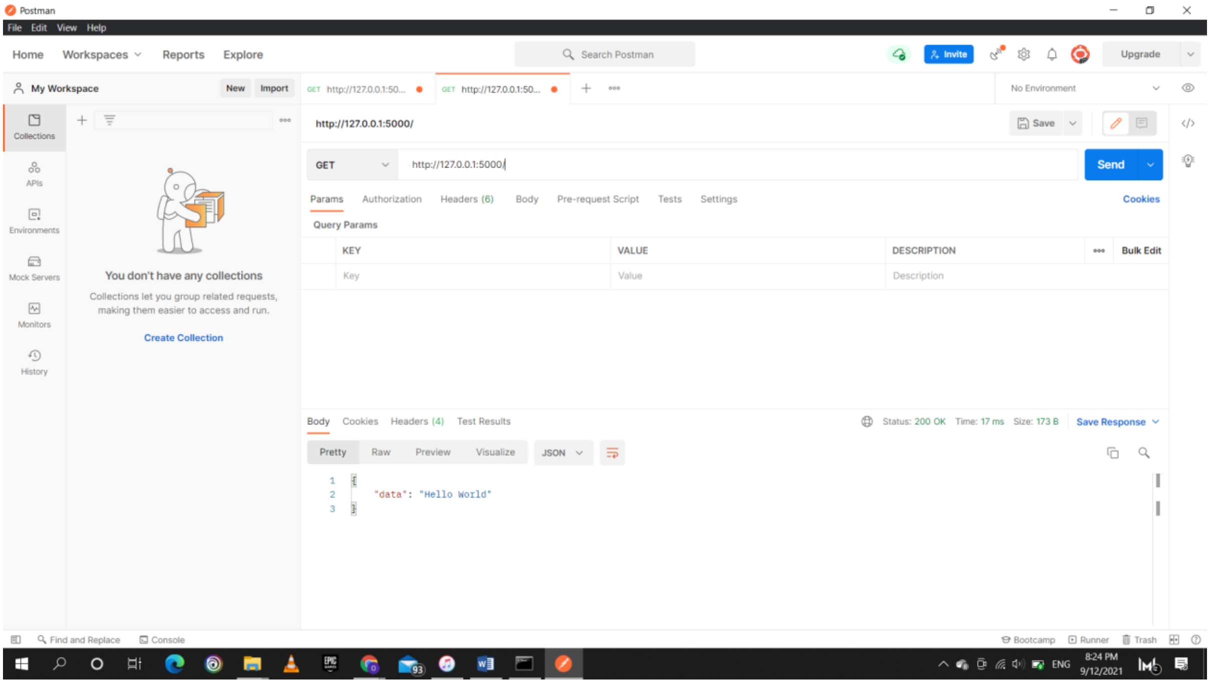
```python
from flask import Flask, jsonify, request
import pickle
import pandas as pd

app = Flask(__name__)


@app.route('/' , methods = ['GET' , 'POST'])
def home():
    if(request.method =='GET'):
        data = "Hello World"
        return jsonify({'data':data})


@app.route('/predict/')
def tumor_predict():
    model = pickle.load(open('model.pkl' , 'rb'))
    mean_radius = request.args.get('mean_radius')
    mean_texture = request.args.get('mean_texture')
    mean_perimeter = request.args.get('mean_perimeter')
    mean_area = request.args.get('mean_area')
    mean_smoothness = request.args.get('mean_smoothness')



    test_df = pd.DataFrame({'mean_radius':[mean_radius] , 'mean_texture':[mean_texture] ,
    'mean_perimeter':[mean_perimeter] , 'mean_area':[mean_area] , 'mean_smoothness':[mean_smoothness]})

    tumor_pred = model.predict(test_df)
    if tumor_pred == 1:
      return jsonify({'result': 'tumor is benign , it will not spread'})
    else:
      return jsonify({'result': 'tumor is malignant , it will spread'})

if __name__ =='__main__':
    app.run(port = 5000 , debug = True)
```



```
Microsoft Windows [Version 10.0.19042.867]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Admin>pip install flask
Requirement already satisfied: flask in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (2.0.1)
Requirement already satisfied: click>=7.1.2 in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (from flask) (8.0.1)
Requirement already satisfied: Werkzeug>=2.0 in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (from flask) (2.0.1)
Requirement already satisfied: Jinja2>=3.0 in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (from flask) (3.0.1)
Requirement already satisfied: itsdangerous>=2.0 in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (from flask) (2.0.1)
Requirement already satisfied: colorama in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (from click>=7.1.2->flask) (0.4.4)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\admin\appdata\local\programs\python\python39\lib\site-packages (from Jinja2>=3.0->flask) (2.0.1)

C:\Users\Admin>flask --version
Python 3.9.6
Flask 2.0.1
Werkzeug 2.0.1

C:\Users\Admin>cd Desktop

C:\Users\Admin\Desktop>cd "flask deployment"

C:\Users\Admin\Desktop\flask deployment>python model_deploy.py
 * Serving Flask app 'model_deploy' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 149-081-177
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Step 6:

In Postman, enter the flask server URL http://127.0.0.1:5000/. This will show the hello world message on web. This is equivalent to running the (/) endpoint.



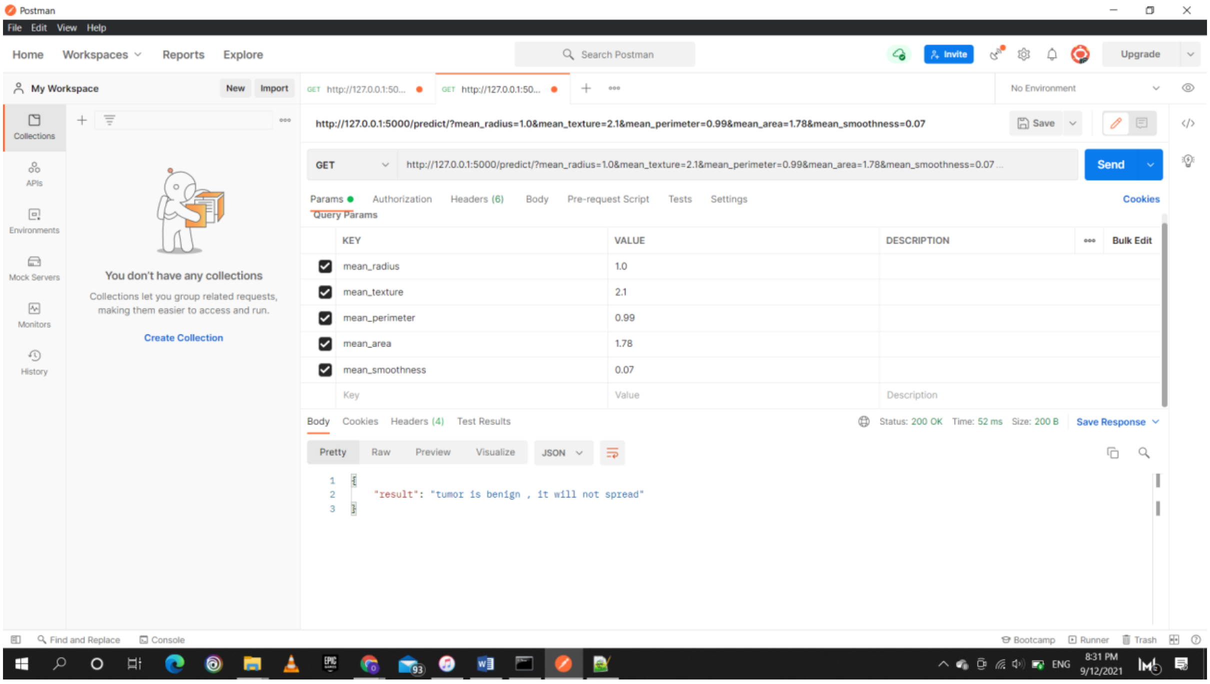
GET http://127.0.0.1:5000/

Status: 200 OK Time: 17 ms Size: 173 B

```
{
    "data": "Hello World"
}
```

Step 7:

In the URL, add /predict/ to the end of it to use the (/predict/) endpoint. This endpoint requires the features, which can be created using the key field. To add values to each key, use the value field. Finally press the send and you will see the result of the prediction written below.



GET http://127.0.0.1:5000/predict/?mean_radius=1.0&mean_texture=2.1&mean_perimeter=0.99&mean_area=1.78&mean_smoothness=0.07

Query Params

	KEY	VALUE	DESCRIPTION
☑	mean_radius	1.0	
☑	mean_texture	2.1	
☑	mean_perimeter	0.99	
☑	mean_area	1.78	
☑	mean_smoothness	0.07	

Status: 200 OK Time: 52 ms Size: 200 B

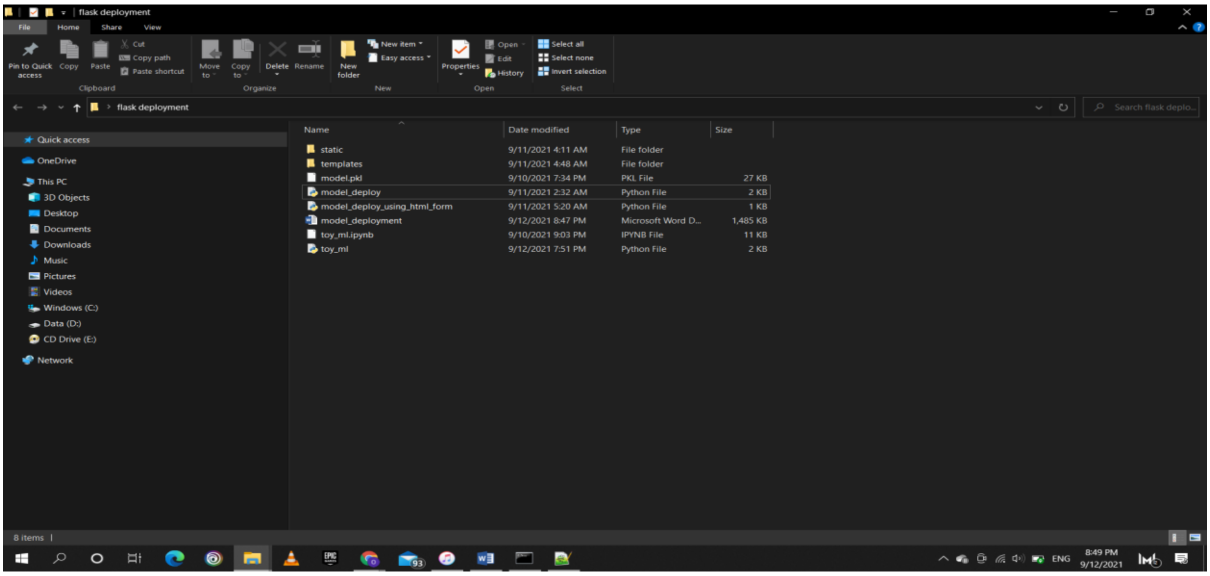
```
{
    "result": "tumor is benign , it will not spread"
}
```

Deployment Using HTML Form:

We can deploy the model using HTML forms. In this way, the model will receive the features from the form fields.

Step 4:

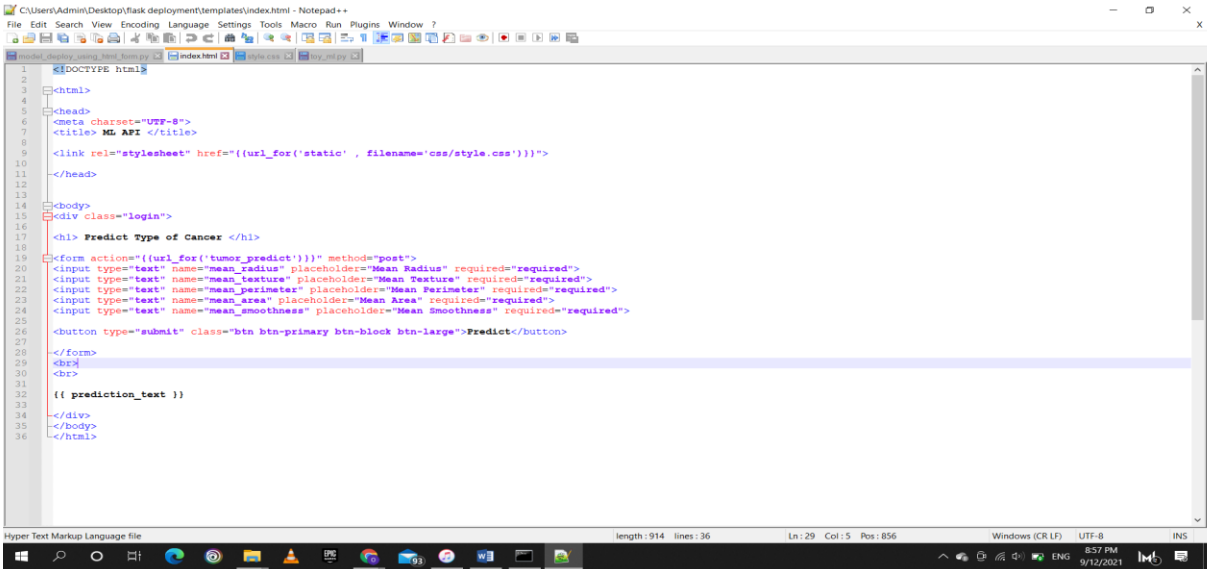
Create an HTML page. It does not have to be fancy, but it should have a form and its fields to enter the features' values. If you want to add a style to the page, you can create a CSS file and link it to the page. Make sure that the HTML page is saved in templates folder, and CSS file is saved in /static/css folder. These are the default settings for the Flask (can be changed).



flask deployment

Name	Date modified	Type	Size
static	9/11/2021 4:11 AM	File folder	
templates	9/11/2021 4:48 AM	File folder	
model.pkl	9/10/2021 7:34 PM	PKL File	27 KB
model_deploy	9/11/2021 2:32 AM	Python File	2 KB
model_deploy_using_html_form	9/11/2021 5:20 AM	Python File	1 KB
model_deployment	9/12/2021 8:47 PM	Microsoft Word D...	1,485 KB
toy_ml.ipynb	9/10/2021 9:03 PM	IPYNB File	11 KB
toy_ml	9/12/2021 7:51 PM	Python File	2 KB

8 items



C:\Users\Admin\Desktop\flask deployment\templates\index.html - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
model_deploy_using_html_form.py
index.html
style.css
toy_ml.py

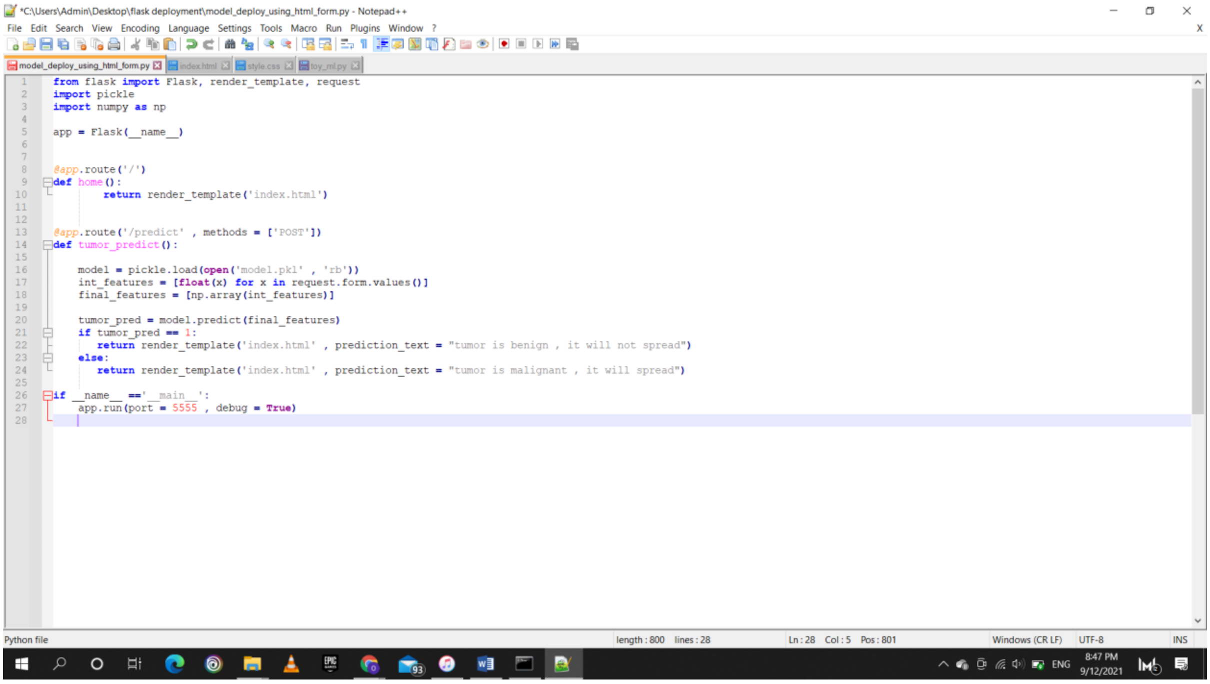
```
<!DOCTYPE html>

<html>

<head>
<meta charset="UTF-8">
<title> ML API </title>

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

</head>


<body>
<div class="login">

<h1> Predict Type of Cancer </h1>

<form action="{{url_for('tumor_predict')}}" method="post">
<input type="text" name="mean_radius" placeholder="Mean Radius" required="required">
<input type="text" name="mean_texture" placeholder="Mean Texture" required="required">
<input type="text" name="mean_perimeter" placeholder="Mean Perimeter" required="required">
<input type="text" name="mean_area" placeholder="Mean Area" required="required">
<input type="text" name="mean_smoothness" placeholder="Mean Smoothness" required="required">

<button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>

</form>
<br>
<br>

{{ prediction_text }}

</div>
</body>
</html>
```

Hyper Text Markup Language file
length : 914 lines : 36
Ln : 29 Col : 5 Pos : 856
Windows (CR LF) UTF-8 INS

Step 5:

Run this code by typing python *file_name* in the CMD. It does the following:

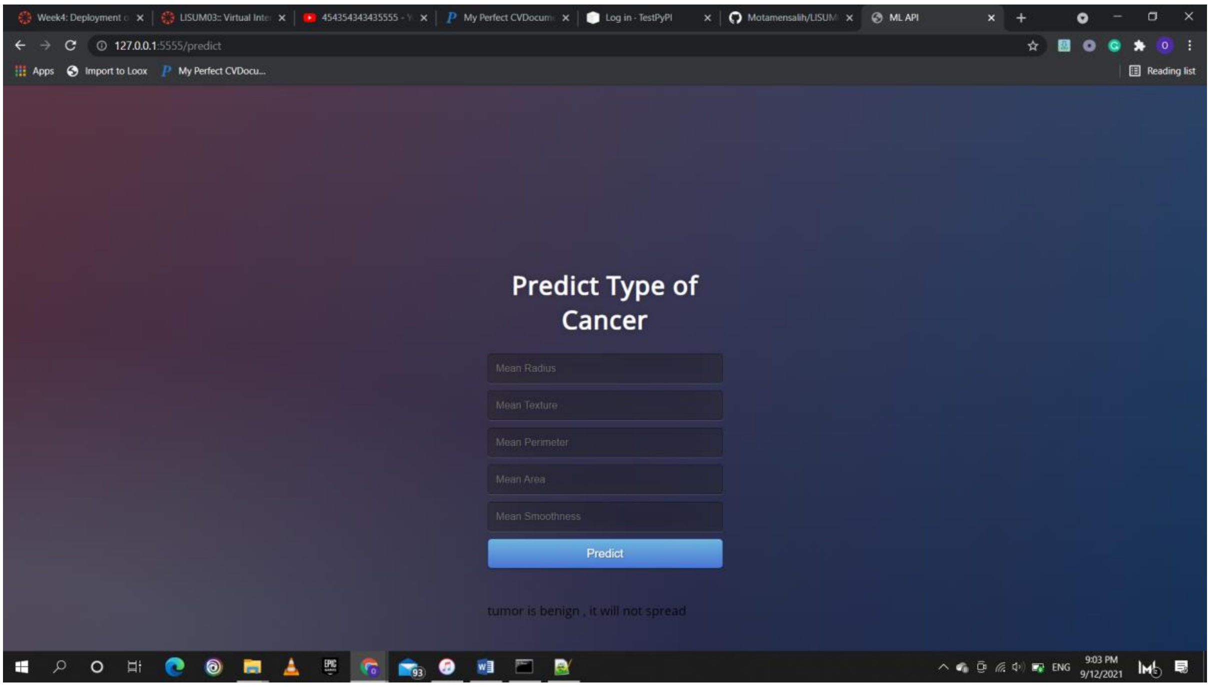
1. Load the model.
2. Create (/) endpoint to load the HTML page.
3. Create (/predict) endpoint to receive the features from the HTML form and use them for prediction.
4. Add the prediction result to the HTML and load it.


*C:\Users\Admin\Desktop\flask deployment\model_deploy_using_html_form.py - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
model_deploy_using_html_form.py
index.html
style.css
toy_ml.py
Python file
length : 800 lines : 28
Ln : 28 Col : 5 Pos : 801
Windows (CR LF)
UTF-8
INS
8:47 PM
9/12/2021
ENG

```
from flask import Flask, render_template, request
import pickle
import numpy as np

app = Flask(__name__)


@app.route('/')
def home():
        return render_template('index.html')


@app.route('/predict' , methods = ['POST'])
def tumor_predict():

    model = pickle.load(open('model.pkl' , 'rb'))
    int_features = [float(x) for x in request.form.values()]
    final_features = [np.array(int_features)]

    tumor_pred = model.predict(final_features)
    if tumor_pred == 1:
       return render_template('index.html' , prediction_text = "tumor is benign , it will not spread")
    else:
       return render_template('index.html' , prediction_text = "tumor is malignant , it will spread")

if __name__ =='__main__':
    app.run(port = 5555 , debug = True)
    
```

Step 6:

Open the browser and type http://127.0.0.1:5555/. This will display the HTML page with the features' fields. Type the values in each field and press predict button. The result of prediction will be displayed.



Predict Type of Cancer

tumor is benign , it will not spread