Week 5 – API based deployment using Postman

Name: Archana Devi Ramesh

Batch code: LISUM16

Submission date: 5th January 2023

Submitted to: Data Glacier

Submission Link: https://github.com/ArchanaDeviRamesh/Data-Glacier-Week5

API based Deployment steps

1. Since **Postman** accepts input in the form of **json** objects, the data returned from the flask app is converted to json, instead of returning a web page.

app_api.py

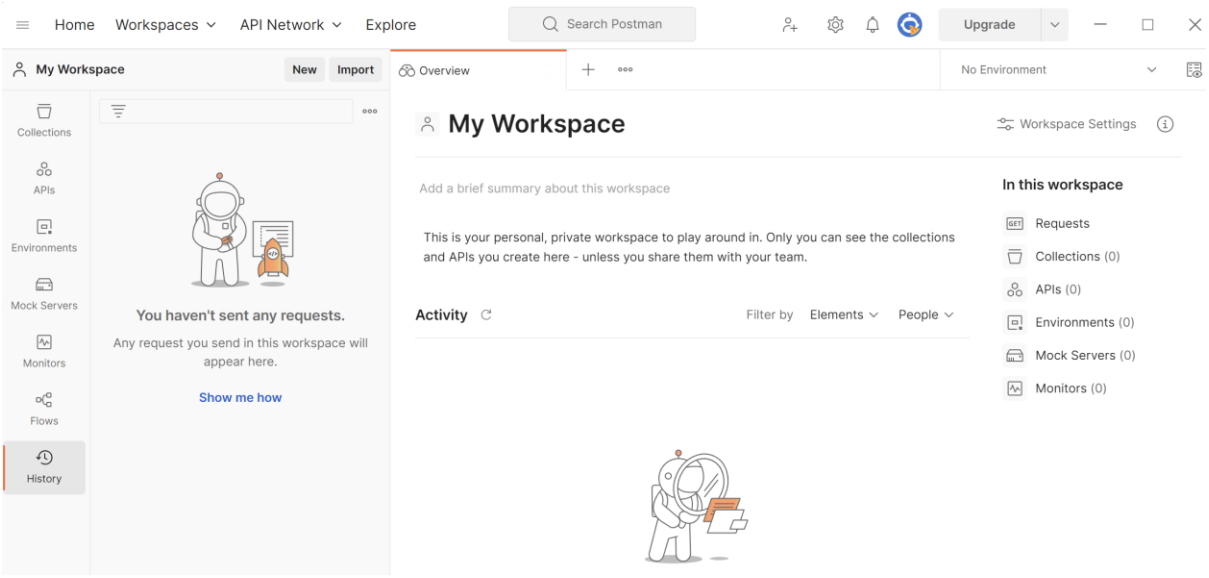
```python
import numpy as np
import pandas as pd
from flask import Flask, request, jsonify
import pickle

# Create flask app
app_api = Flask(__name__)
model = pickle.load(open("iris_model.pkl", "rb"))

@app_api.route("/", methods = ['GET', 'POST'])
def Home():
    if request.method == 'GET':
        data = 'Testing with API'
        return jsonify({'data':data})

@app_api.route("/predict/", methods = ['GET', 'POST'])
def predict():
    sepal_length = request.args.get("sepal_length")
    sepal_width = request.args.get("sepal_width")
    petal_length = request.args.get("petal_length")
    petal_width = request.args.get("petal_width")
    test_df = pd.DataFrame({'sepal.length': [sepal_length], 'sepal.width':[sepal_width], 'petal.length':[petal
    prediction = model.predict(test_df)
    return jsonify({'Iris Flower':str(prediction)})

if __name__ == "__main__":
    app_api.run(debug=True)
```

2. Run the above python file as python app_api.py, copy the URL http://127.0.0.1:5000 from the terminal

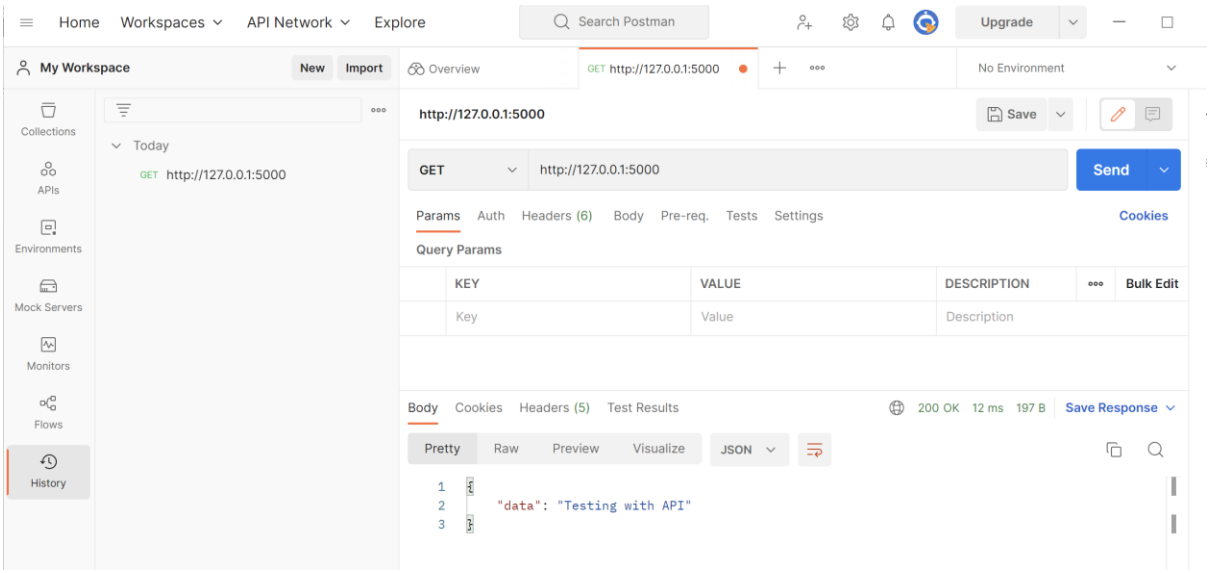
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
(D:\mydocuments\CANADA\Internship\DataGlacier\Week4\myvenv) D:\mydocuments\CANADA\Internship\DataGlacier\Week4>python a
pp_api.py
 * Serving Flask app 'app_api'
 * 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: 528-755-425
```

3. Open Postman and click on **Requests**, under **In this workspace**



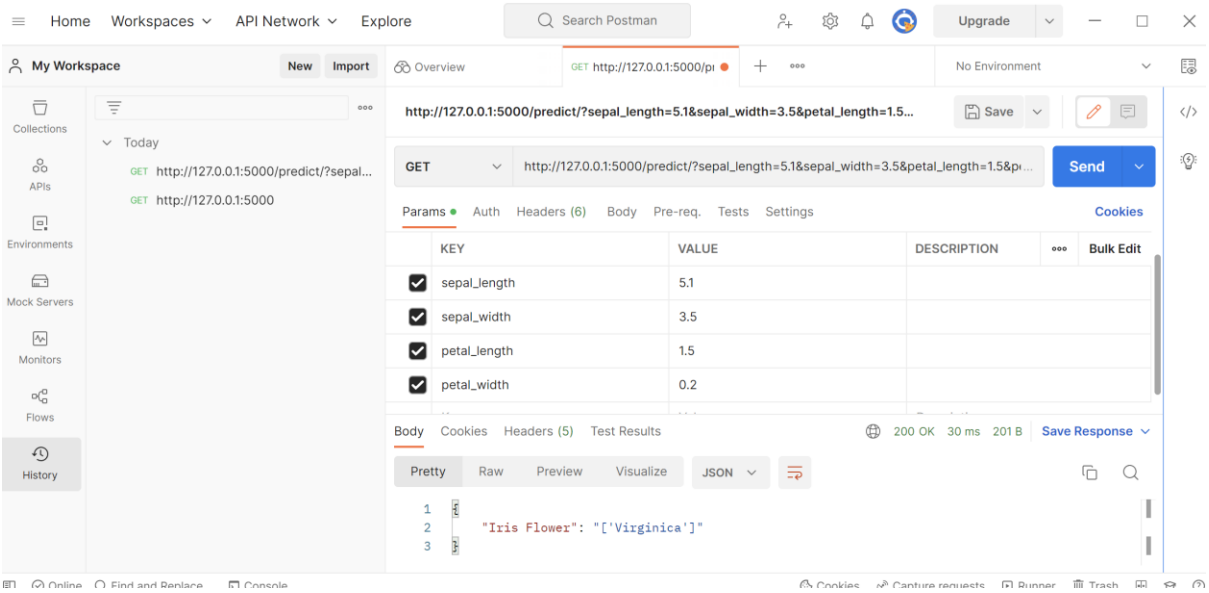
4. Enter the URL in the tab near **GET** and click **Send.**
   Under GET request, an object **data** was passed with the value **Testing with API** which is getting displayed under the results



5. For the prediction, enter the URL http://127.0.0.1:5000/predict and give key and value for the four features **(sepal_length, sepal_width, petal_length, petal_width).** Make sure to have the same key names given under predict method for these features.

```
@app_api.route("/predict/", methods = ['GET', 'POST'])
def predict():
    sepal_length = request.args.get("sepal_length")
    sepal_width = request.args.get("sepal_width")
    petal_length = request.args.get("petal_length")
    petal_width = request.args.get("petal_width")
```

6. After entering the key and value for all four features, click Send. The prediction will be displayed in the results section



7. If any of the parameters is not passed, it will throw an error as the model expects four parameters