

**Testing POST method.**

http://127.0.0.1:5000/predict

[

        {

            "sepal length (cm)": 5.1,

            "sepal width (cm)": 3.5,

            "petal length (cm)": 1.1,

            "petal width (cm)": 0.2

        },

        {

            "sepal length (cm)": 6.3,

            "sepal width (cm)": 2.9,

            "petal length (cm)": 6.3,

            "petal width (cm)": 1.8

        },

        {

            "sepal length (cm)": 6.7,

            "sepal width (cm)": 2.8,

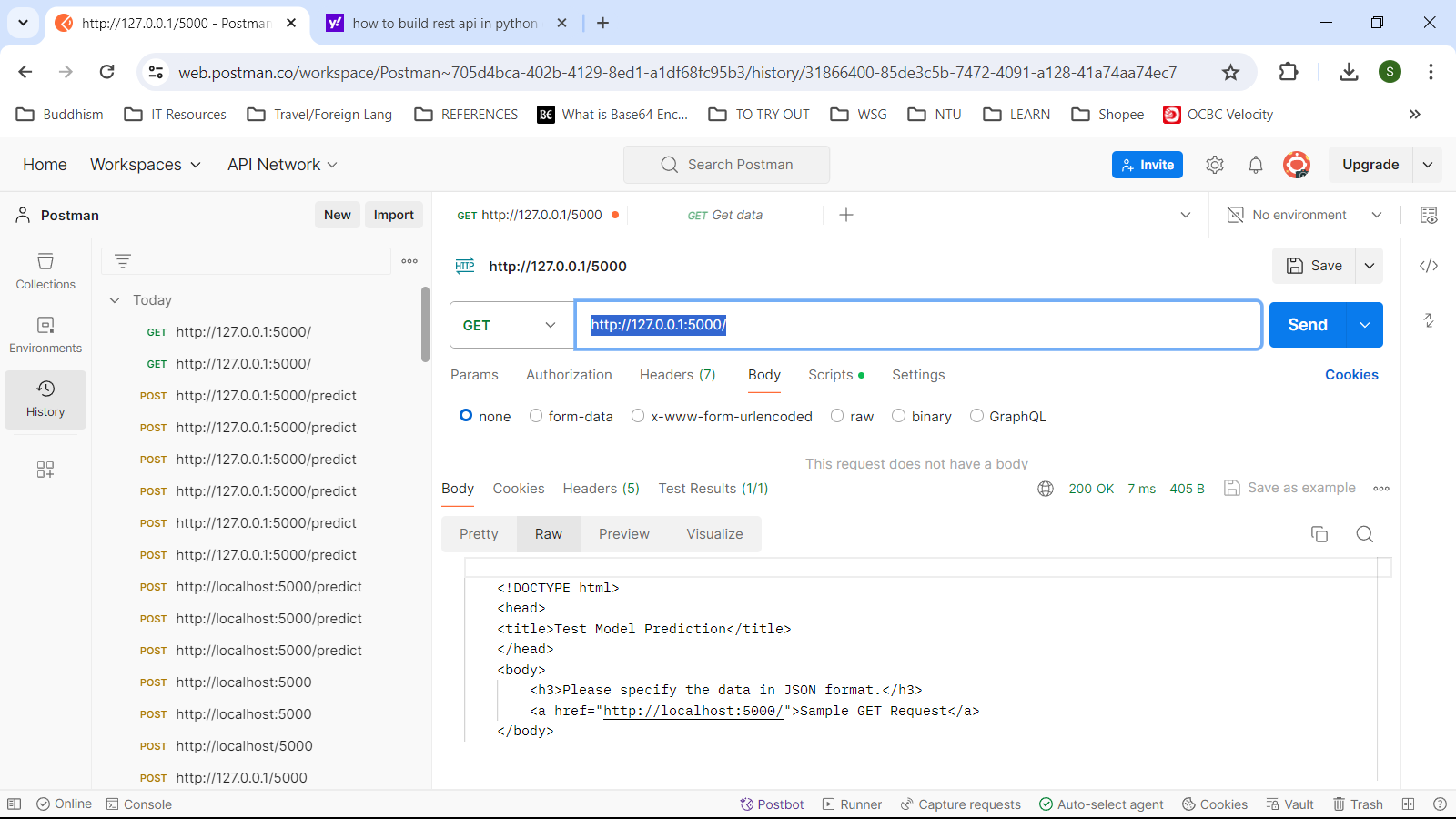
            "petal length (cm)": 4.4,

            "petal width (cm)": 1.4

        }

    ]

**{"prediction":"[0, 2, 1]"}**



**Testing GET method.**

http://127.0.0.1:5000/

**<!DOCTYPE html>**

**<head>**

**<title>Test Model Prediction</title>**

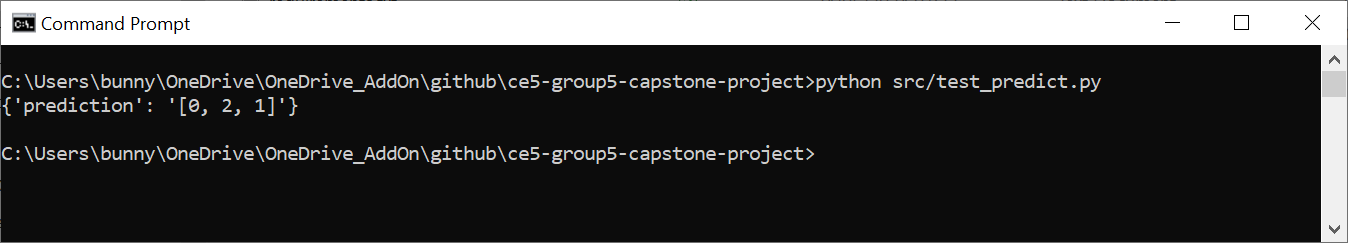
**</head>**

**<body>**

**<h3>Please specify the data in JSON format.</h3>**

**<a href="http://localhost:5000/">Sample GET Request</a>**

**</body>**



**C:\Users\bunny\OneDrive\OneDrive\_AddOn\github\ce5-group5-capstone-project>python src/test\_predict.py**

**{'prediction': '[0, 2, 1]'}**

**test\_predict.py**

# https://sebhastian.com/no-module-named-requests/

# pip install requests

# For pip3:

# pip3 install requests

import requests

def test\_predict():

# Define the URL of your Flask API

url = 'http://127.0.0.1:5000/'

# Send a GET request to the API.

response = requests.get(url)

# Define the URL of your Flask API

url = 'http://127.0.0.1:5000/predict'

# Define the input data as a dictionary.

# This is (0, setosa).

# This is (2, virginica).

# This is (1, versicolor).

test\_data = [

{

"sepal length (cm)": 5.1,

"sepal width (cm)": 3.5,

"petal length (cm)": 1.1,

"petal width (cm)": 0.2

},

{

"sepal length (cm)": 6.3,

"sepal width (cm)": 2.9,

"petal length (cm)": 6.3,

"petal width (cm)": 1.8

},

{

"sepal length (cm)": 6.7,

"sepal width (cm)": 2.8,

"petal length (cm)": 4.4,

"petal width (cm)": 1.4

}

]

# Send a POST request to the API with the input data

response = requests.post(url, json=test\_data)

# Check the HTTP response status code

# For a full list of response codes, go to https://www.guru99.com/testing-rest-api-manually.html.

if response.status\_code == 200:

# Parse and print the JSON response (assuming it contains the prediction)

prediction = response.json()

print(prediction)

else:

# Handle the case where the API request failed

print(f'API Request Failed with Status Code: {response.status\_code}')

print(f'Response Content: {response.text}')

if \_\_name\_\_ == "\_\_main\_\_":

test\_predict()

**# WORKDIR = C:\Users\bunny\OneDrive\OneDrive\_AddOn\github\ce5-group5-capstone-project>**

**# docker build -t ml-model:latest .**

**# docker run -dp 5000:5000 ml-model:latest**

**#**

**# python src/test\_predict.py**

**# {'prediction': '[0, 2, 1]'}**

**#**

**# When you test the POST method using postman.com, please enter the data as raw/JSON. The keys must be enclosed by double and not single quotes.**

**# [**

**# {**

**# "sepal length (cm)": 5.1,**

**# "sepal width (cm)": 3.5,**

**# "petal length (cm)": 1.1,**

**# "petal width (cm)": 0.2**

**# }**

**# ]**

**# Sample curl commands for testing.**

**# curl -i -X GET http://rest-api.io/items**

**# curl -i -X GET http://rest-api.io/items/5069b47aa892630aae059584**

**# curl -i -X DELETE http://rest-api.io/items/5069b47aa892630aae059584**

**# This is (1, versicolor).**

**# curl -i -X POST -H 'Content-Type: application/json' -d '{'sepal length (cm)': 6.7, 'sepal width (cm)': 2.8, 'petal length (cm)': 4.4, 'petal width (cm)': 1.4}' http://127.0.0.1:5000/predict**

**# curl -i -X PUT -H 'Content-Type: application/json' -d '{"name": "Updated item", "year": "2010"}' http://rest-api.io/items/5069b47aa892630aae059584**