HTTP API

Author: Vu Trung Kien

Last update: 10/03/2021

# Terminologies

## API

An application programming interface (API) is a computing interface that defines interactions between multiple software or mixed hardware-software intermediaries. It defines the kinds of calls or requests that can be made, how to make them, the data formats that should be used, the conventions to follow, etc. It can also provide extension mechanisms so that users can extend existing functionality in various ways and to varying degrees. An API can be entirely custom, specific to a component, or designed based on an industry-standard to ensure interoperability. Through information hiding, APIs enable modular programming, allowing users to use the interface independently of the implementation.

## Client – Server Model

Client–server model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients.

## RESTful API

Representational state transfer (REST) is a software architectural style which uses a subset of HTTP. It is commonly used to create interactive applications that use Web services. A Web service that follows these guidelines is called RESTful. Such a Web service must provide its Web resources in a textual representation and allow them to be read and modified with a stateless protocol and a predefined set of operations. This approach allows interoperability between the computer systems on the Internet that provide these services. REST is an alternative to, for example, SOAP as way to access a Web service.

# Required Reading Materials

## Books

**REST API Design Rulebook by Mark Massé**

Link Download: <https://1drv.ms/b/s!AvI3kUptqznWldRIMFsATvtVxUnoEw?e=sCrBIq>

## Articles

<https://github.com/NationalBankBelgium/REST-API-Design-Guide/wiki>

# Highlights

## GET và POST method

GET và POST method là hai method được sử dụng phổ biến nhất. Trong đó, GET thường được client sử dụng để lấy dữ liệu từ server, và POST được sử dụng để truyền dữ liệu lên server.

## Request Parameters

Có ba phương thức truyền tham số phổ biến trong một HTTP request: in path, in query, và in body.

### In Path

Tham số được truyền vào thông qua API Path. Ví dụ:

[GET] [http://localhost:1111/books/{book-id}](http://localhost:1111/books/%7bbook-id%7d)

Trong đó {book-id} là tham số truyền vào, chẳng hạn gọi: <http://localhost:1111/books/rest-api-book>

Truyền tham số in path thường được sử dụng khi truy vấn thông tin của một resource khi biết ID.

### In Query

Ví dụ:

[GET] <http://localhost:1111/events?year=2013&month=3>

Ở đây truyền vào hai tham số sau dấu ? là year và month với giá trị 2013 và 3.

Truyền tham số in query thường được sử dụng khi truy vấn thông tin trả về là một danh sách, và tham số in query được dùng để lọc các giá trị trong list.

### In Body

Truyền tham số trong body của request **phải** sử dụng method POST. Các tham số được truyền qua “form-data” chứa một cặp key và value. Trong đó value của một tham số thường là kiểu text hoặc binary stream.

Truyền thàm số trong body của request thường được sử dụng khi số lượng tham số nhiều, phức tạp hoặc cần truyền file lên server.

## Routing

Routing là quá trình ghép một đường dẫn (path) của API đến hàm (phương thức) được lập trình trước.

# Example

Phần này trình bày một ví dụ sử dụng ngôn ngữ Python và Flask framework để viết server cung cấp một API đơn giản. Và dùng Postman để đóng vai trò client gửi request.

## Cài đặt các thư viện

pip install flask flask\_cors flask\_restful numpy opencv-python

## Sample Code

Import required libraries:

from flask import Flask

from flask\_cors import cross\_origin

from flask\_restful import Api, Resource, reqparse

import werkzeug

import numpy as np

import cv2

Create ImageFileChecker class to check whether a byte stream is an image or not:

class ImageFileChecker():

def check(self, file\_bytes):

try:

npimg = np.frombuffer(file\_bytes, np.uint8)

img = cv2.imdecode(npimg, cv2.IMREAD\_COLOR |

cv2.IMREAD\_IGNORE\_ORIENTATION)

except:

img = None

if img is None:

return False

return True

Create a Flask Resource named CheckImageFileService to handle HTTP request from clients:

class CheckImageFileService(Resource):

def \_\_init\_\_(self, parser: reqparse.RequestParser, checker: ImageFileChecker):

self.parser = parser

self.checker = checker

@cross\_origin(supports\_credentials=True)

def get(self):

return "Please use POST method"

@cross\_origin(supports\_credentials=True)

def post(self):

args = self.parser.parse\_args()

file = args.file

ok = False

if file:

file\_bytes = file.read()

ok = self.checker.check(file\_bytes)

return {"isImage": ok}

Create a Flask App and route resources:

def create\_app():

app = Flask(\_\_name\_\_)

api = Api(app)

parser = reqparse.RequestParser()

parser.add\_argument(

"file",

required=True,

type=werkzeug.datastructures.FileStorage,

location="files",

)

checker = ImageFileChecker()

api.add\_resource(

CheckImageFileService,

"/check-image-file",

resource\_class\_kwargs={"parser": parser, "checker": checker},

)

return app

app = create\_app()

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

app.run(host="0.0.0.0", port=3012)

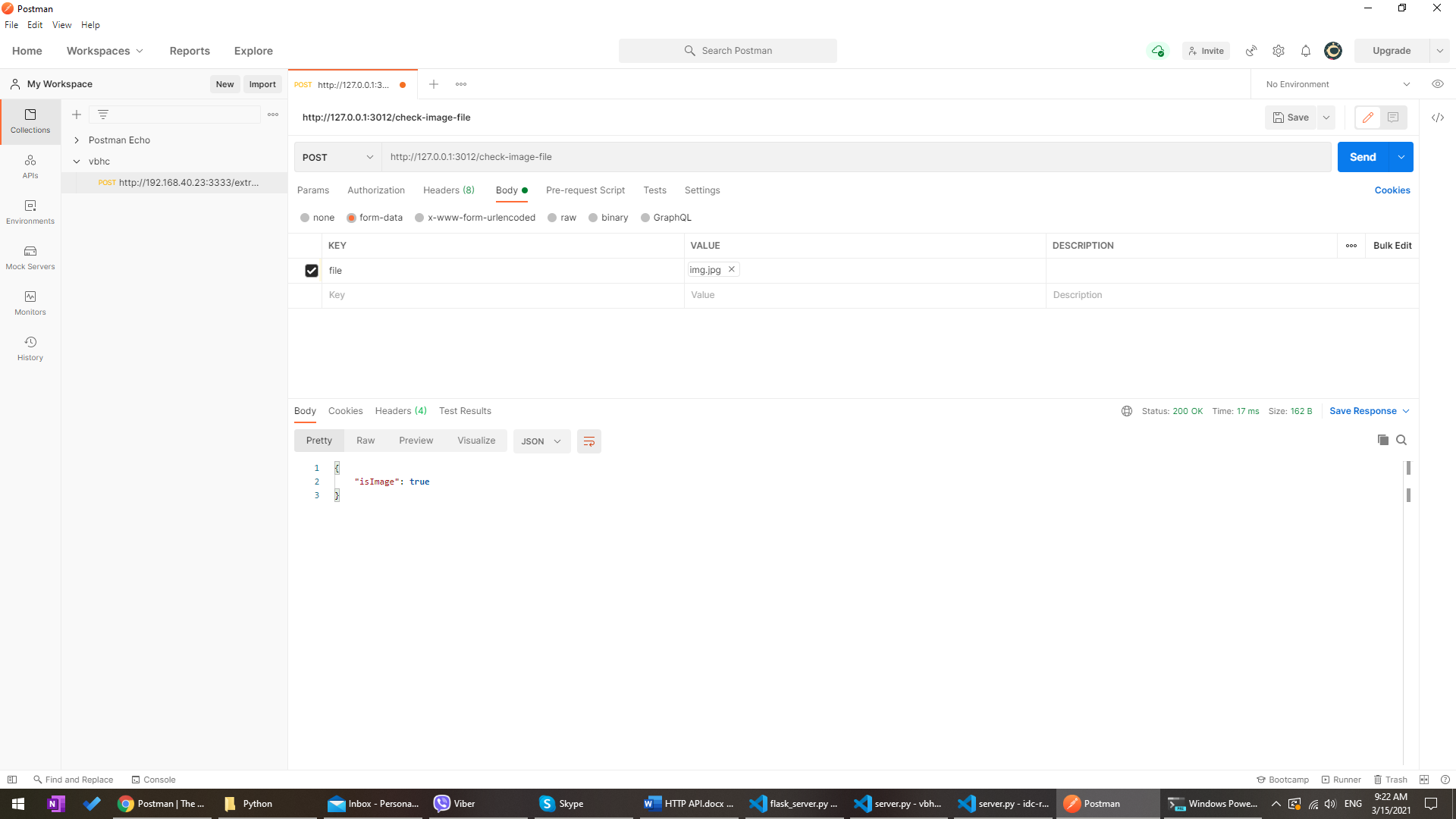
Put all things together: <https://1drv.ms/u/s!AvI3kUptqznWldROnKU-LDL9X7x5_A?e=km07iK>

Chạy server:

python flask\_server.py

## Dùng Postman Để Gửi Request

Download Postman: <https://www.postman.com/>



# Exercise

Bạn cần thể hiện khả năng sử dụng các framework deep learning để huấn luyện model giải quyết một bài toán về Computer Vision và triển khai model đã huấn luyện cho mọi người sử dụng. Bài toán được chọn là bài toán phân loại hình ảnh sử dụng dataset Fashion MNIST (<https://github.com/zalandoresearch/fashion-mnist>). Nhiệm vụ của bạn bao gồm hai phần:

1. Huấn luyện model phân loại hình ảnh cho tập dữ liệu Fashion MNIST
2. Triển khai RestfulAPI tại localhost. Người dùng có thể gọi API này với tham số là tệp ảnh truyền lên và server trả về JSON chứa tên class được phân loại của ảnh đầu vào.

Ví dụ gọi: [http://127.0.0.1:1234/classify](http://localhost:1234/classify) trả về response:

{

"class": "Shirt"

}