**GitHub Copilot, Flask, OpenAPI and Swagger Editor**

**Prerequisites**

Swagger Editor

<https://editor.swagger.io>

Just make sure you can access this tool, this is a workflow step in the process that converts APIFlask generated JSON to YAML. Yaml is consumed by the OpenAPI VS Code extensions

Python/Pip:

$ npm install apiflask

VSCode

Python VS Code Extension

A screenshot of a computer

Description automatically generated

OpenAPI VS Code Extension

A screenshot of a computer

Description automatically generated

Install these extensions as they will be used in this lab

API Flask

* Website: [https://apiflask.com](https://apiflask.com/)
* Documentation: <https://apiflask.com/docs>
* PyPI Releases: <https://pypi.python.org/pypi/APIFlask>

APIFlask is a lightweight Python web API framework based on [Flask](https://github.com/pallets/flask) and [marshmallow-code](https://github.com/marshmallow-code) projects. It's easy to use, highly customizable, ORM/ODM-agnostic, and 100% compatible with the Flask ecosystem.

With APIFlask, you will have:

* More sugars for view function (@app.input(), @app.output(), @app.get(), @app.post() and more)
* Automatic request validation and deserialization
* Automatic response formatting and serialization
* Automatic [OpenAPI Specification](https://github.com/OAI/OpenAPI-Specification) (OAS, formerly Swagger Specification) document generation
* Automatic interactive API documentation
* API authentication support (with [Flask-HTTPAuth](https://github.com/miguelgrinberg/flask-httpauth))
* Automatic JSON response for HTTP errors

## **Requirements**

* Python 3.8+
* Flask 2.0+

## **Prompting – Getting Started**

Create app.py in VS Code

Prompt string: Be sure to experiment with your own

# Generate APIFlask app

# add a GET method called "SayHello" that accepts input as a query string using a string as a parameter for the name to say a greeting

Press CTRL – [Return]

A screenshot of a computer

Description automatically generated

Coaxing CoPilot to generate code happens with a few [Return] keys, I

Also entered:

import apiflask [space]

After Copilot generated the Imports and application Object, I did have to prompt it for Schema Object:

# Schema for input

From here It generated the Input and Output Schema objects

A computer code with text

Description automatically generated with medium confidence

Prompt for entry point:

# GET method - Use Query string parameters

You should have something that resembles:

import apiflask as api

from apiflask import Schema, abort

from apiflask.fields import Integer, String

from apiflask.validators import Length, OneOf

app = api.APIFlask(\_\_name\_\_)

# Schema for input

class SayHelloIn(Schema):

name = String(required=True, validate=Length(0, 10))

# Schema for output

class SayHelloOut(Schema):

message = String()

# GET method - Use Query string parameters

@app.get('/SayHello')

@app.input(SayHelloIn)

@app.output(SayHelloOut)

def say\_hello(json\_data):

return {'message': 'Hello ' + json\_data['name'] + '!'}

Be sure to save this: [CTRL] + s or [CMD] + s

Make sure your vnev is set up, I use conda:

conda info –env

conda activate snow-3.8

Make sure you get the right Python Interpreter in your VS Code session.

Click VS-Code menu

View | Command Palette, then enter Python: Select Interpreter

A screenshot of a computer program

Description automatically generated

This should match your virtual environment

I open a terminal and run the code, you can optionally create a Launch Profile for Flask in VS Code

Click the Debug Icon -- 

Then Choose “Add Configuration”

A screenshot of a computer program

Description automatically generated

Add this configuration to your .vscode/launch.json

{

"name": "Python: Flask",

"type": "python",

"request": "launch",

"module": "flask",

"env": {

"FLASK\_APP": "app.py",

"FLASK\_DEBUG": "1"

},

"args": [

"run",

"--no-debugger",

"--no-reload"

],

"jinja": true,

"justMyCode": true

}

Click on the file: app.py

To enter a breakpoint, click in the margin, left of the line numbers. It will look like this:

A screen shot of a computer code

Description automatically generated

From the VS Code Menu:

Run | | Start Debugging

Bring up a browser: <http://127.0.0.1:5000/SayHello?name=John>

Error ????? – This is an error handler response but why?

A screenshot of a computer

Description automatically generated

Lets take advantage of OpenAPI and use the executable Documentation

[APIFlask 0.1.0 - Swagger UI](http://127.0.0.1:5000/docs)

The problem is evident, the data is in the request body (which you cant do with a GET), not being passed as query parameters

A screenshot of a computer

Description automatically generated

We will need to modify the generated code -- <https://apiflask.com/request/>

A screenshot of a computer program

Description automatically generated

Change from the body to query string

A screen shot of a computer code

Description automatically generated

This works

A screenshot of a computer

Description automatically generated

Add one more method to add 2 numbers and return the sum

Prompt:

# GET Method: Add two query string parameters and return the sum

This is great, CoPilot learned from the last method and from appearances did this method properly

A screenshot of a computer code

Description automatically generated

Let’s go back to our OpenAPI documentation to test this:

<http://127.0.0.1:5000/docs>

A white and blue line

Description automatically generated

We see this works

A screenshot of a computer

Description automatically generated

So far, this has been a Code First API approach, now we will pull our design documentation into VS Code

Click the Blue “/openapi.json” link A close up of a logo

Description automatically generated

Copy the JSON into the Clipboard

No add the Swagger URL to your browser: <https://editor.swagger.io>

From the menu: File | clear editor

Next paste the JSON into the editor, then you will see:

A screenshot of a computer

Description automatically generated

Select OK

If all goes well you will see

A screenshot of a computer

Description automatically generated

The VS Code OpenAPI extension works from the command palette, and all commands are prefixed with: OpenAPI

A screenshot of a computer

Description automatically generated

Paste in our API specification in JSON and save it

A screenshot of a computer

Description automatically generated

Next run the Swagger Interface:

Click: View | Command Palette

Enter: OpenAPI

A screenshot of a computer program

Description automatically generated

To make the Swagger/OpenAPI editor work, I ran the Flask API in another terminal, set your default directory where your python/flask app is located:

A screenshot of a computer

Description automatically generated