Let's build an API to hack - Part 1: The basics

Introduction
Requirements
Let's set up
Hello World
Extra: Full code

Introduction

To get started with properly hacking an API and learning how they can be vulnerable, I think it's wise that we build our own API so that we can at least be sure we will not be sued unless we truely do not like ourselves 🙂. We will be building several vulnerable APIs in python and to do this we will be using Flask. Flask is a python webframework which will allow us to easily start a web application. The beauty of flask is that it's bearbones and does not require any external libraries. This means that pretty much any of our systems can run it. I would personally advice you to rent a VPS to do this on because that will stop you from worrying about all the networking and logistics of hosting a vulnerable API on your home network.

In this chapter you will be learning about what an API is, how to build one in a virtual python environment and how to hack it.

Requirements

- A patato ... Seriously though, a small VPS or spare computer with the minimal amount of RAM and disk space will do. The APIs we will be building do not require much.
- Python 3.x (https://www.python.org/downloads/)

- Flask (pip install Flask but hold off if you did not do it yet, we will be creating a python virtual env)

Let's set up

To start with, we will need to set up a virtual environment first. This is a place we can install our dependencies of a certain project on and keep them seperate from the other projects. This is very useful to keep oversight but also if you have one project that requires a certain version of an import while another project might need a much older and non-compatible version of that library.

```
mkdir "GoudAPI"
cd GoudAPI
python3 -m venv GoudAPI
```

```
mkdir myproject
cd myproject
py -3 -m venv GoudAPI
```

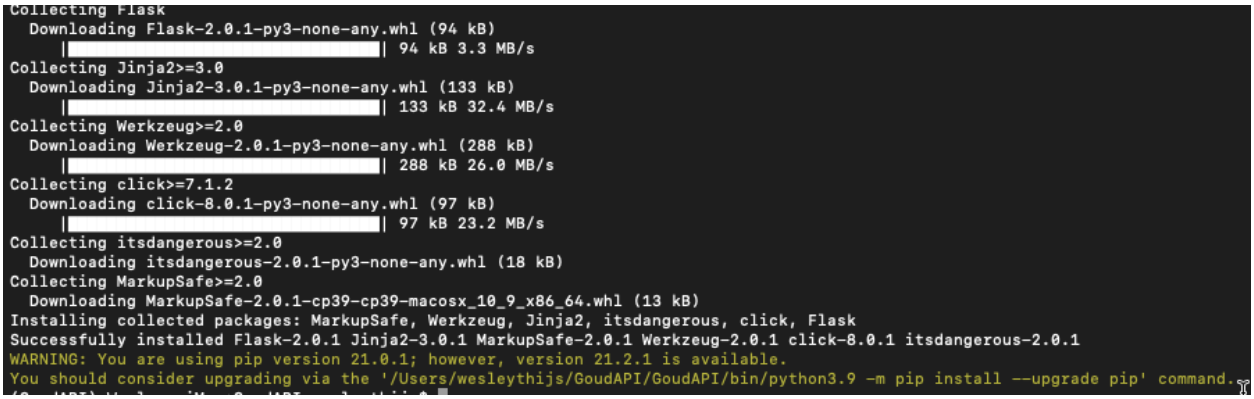
With these commands we are creating a venv (virtual enviornment) called GoudAPI which is marked by a new folder, now we have to swith to it.

```
. GoudAPI/bin/activate
```

```
GoudAPI\Scripts\activate
```

And now we can easily use pip to install flask

```
pip install Flask
```



Now that flask is installed, we can easily create our first vulnerable API.

Hello World

We will first need to write a few lines of code to tell python it should start a flask web application.

```
import flask
from flask import request, jsonify

app = flask.Flask(__name__)
app.config["DEBUG"] = True
```

These lines will tell python to import flask and to set it up with debugging enabled. app.config["DEBUG"] = True refers to a variable from the flask config. We could set up a flask config file seperatly but let's keep it all in one place for now.

We also want to import request and jsonify for later since our first API will only return static data so let's build up that static data with the following code.

```
# Create some test data for our catalog in the form of a list of dictionaries.
cheeseType = [
    {'id': 0,
     'title': 'Gouda',
     'author': 'The XSS Rat',
     'description': 'Dutch: Goudse kaas, "cheese from Gouda") is a sweet, creamy, yellow c
ows milk cheese originating from the Netherlands. ...',
     'year': '1992'},
{'id': 1,
     'title': 'Casu marzu',
     'author': 'The XSS Rat',
```

```
     'description': 'Casu martzu[1] (Sardinian pronunciation; literally rotten/putrid chee
se)is a traditional Sardinian sheep milk cheese that contains live insect larvae (maggot
s)',
     'year': '2020'},
{'id': 2,
     'title': 'Roquefort',
     'author': 'The XSS Rat',
     'description': 'Roquefort is a popular French cheese, reported to be a favourite of E
mperor Charlemagne. In France, it is called the cheese of kings and popes.',
     'year': '2020'}
]
```

So now that we have some data, we need to define a way to get that data. In the next step we will define the path /api/v2/resources/cheese/all and the possible methods. When we make a GET request on "/api/v2/resources/cheese/all" we will return the data from the previous step.

```
# A route to return all of the available entries in our catalog.
@app.route('/api/v2/resources/cheese/all', methods=['GET'])
def api_all():
    return jsonify(cheeseType)
```

all that is left is for us to start up our application but as of now flask does not know how to start yet so let's tell it.

```
app.run(host="0.0.0.0")
```

Combine all of this into one script and it will start up when we ask it, one last thing i want to highlight here is the host="0.0.0.0" argument. We do not need this but if we do not give it this argument, the API will only be accesible from localhost. The default port for flask will be 5000 so make sure nothing else is running on that port.

```
 GNU nano 2.9.3                                  training.py

import flask
from flask import request, jsonify

app = flask.Flask(__name__)
app.config["DEBUG"] = True

# Create some test data for our catalog in the form of a list of dictionaries.
cheeseType = [
    {'id': 0,
     'title': 'Gouda',
     'author': 'The XSS Rat',
     'description': 'Dutch: Goudse kaas, "cheese from Gouda") is a sweet, creamy, yellow cows milk cheese originating from the Netherlands. ...',
     'year': '1992'},
{'id': 1,
     'title': 'Casu marzu',
     'author': 'The XSS Rat',
     'description': 'Casu martzu[1] (Sardinian pronunciation; literally rotten/putrid cheese)is a traditional Sardinian sheep milk cheese that contains live insect
     'year': '2020'},
{'id': 2,
     'title': 'Roquefort',
     'author': 'The XSS Rat',
     'description': 'Roquefort is a popular French cheese, reported to be a favourite of Emperor Charlemagne. In France, it is called the cheese of kings and pope
     'year': '2020')
]

# A route to return all of the available entries in our catalog.
@app.route('/api/v2/resources/cheese/all', methods=['GET'])
def api_all():
    return jsonify(cheeseType)

app.run(host="0.0.0.0")
```

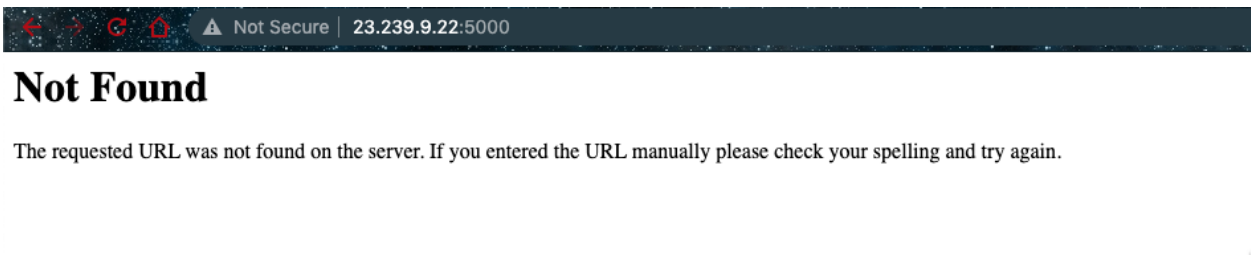
Combine all of that into a file and we are ready to go.

```
python training1.py
```

replace training1.py with whatever you named your file.

```
root@localhost:~/projects/api# python training.py
 * Serving Flask app "training" (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
 * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 641-322-905
```

And with that, we should be able to surf to our VPS' IP adres on port 5000 and see what our flask app is saying.



To get some data, we need to surf to the path we defined earlier.
"/api/v2/resources/cheese/all"


```
[
  {
    "author": "The XSS Rat",
    "description": "Dutch: Goudse kaas, \"cheese from Gouda\") is a sweet, creamy, yellow cows milk cheese originating from the Netherlands. ...",
    "id": 0,
    "title": "Gouda",
    "year": "1992"
  },
  {
    "author": "The XSS Rat",
    "description": "Casu martzu[1] (Sardinian pronunciation; literally rotten/putrid cheese)is a traditional Sardinian sheep milk cheese that contains live insect larvae (maggots)",
    "id": 1,
    "title": "Casu marzu",
    "year": "2020"
  },
  {
    "author": "The XSS Rat",
    "description": "Roquefort is a popular French cheese, reported to be a favourite of Emperor Charlemagne. In France, it is called the cheese of kings and popes.",
    "id": 2,
    "title": "Roquefort",
    "year": "2020"
  }
]
```


Make sure this works as this will be our basefile for the rest of the exercises.

Extra: Full code

```
import flask
from flask import request, jsonify

app = flask.Flask(__name__)
app.config["DEBUG"] = True

# Create some test data for our catalog in the form of a list of dictionaries.
cheeseType = [
    {'id': 0,
     'title': 'Gouda',
     'author': 'The XSS Rat',
     'description': 'Dutch: Goudse kaas, "cheese from Gouda") is a sweet, creamy, yellow c
ows milk cheese originating from the Netherlands. ...',
     'year': '1992'},
{'id': 1,
     'title': 'Casu marzu',
     'author': 'The XSS Rat',
     'description': 'Casu martzu[1] (Sardinian pronunciation; literally rotten/putrid chee
se)is a traditional Sardinian sheep milk cheese that contains live insect larvae (maggot
s)',
     'year': '2020'},
{'id': 2,
     'title': 'Roquefort',
     'author': 'The XSS Rat',
     'description': 'Roquefort is a popular French cheese, reported to be a favourite of E
mperor Charlemagne. In France, it is called the cheese of kings and popes.',
     'year': '2020'}
]

# A route to return all of the available entries in our catalog.
```

```
@app.route('/api/v2/resources/cheese/all', methods=['GET'])
def api_all():
    return jsonify(cheeseType)

app.run(host="0.0.0.0")
```