Data Analytics - Ironhack Madrid

QUICK GUIDE FLASK CHEATSHEET

Create your own flask based APIs!

FLASK OBJECT

```
from flask import Flask

app = Flask("my_api")
```

Import and Create

Any string can be the name.

```
app.run()

app.run(debug=True)
```

Run at the end of file, with or without debug mode

ROUTE DECORATOR

```
@app.route("/hello")
def hello_world():
    return {"hello":"world"}
```

This is how you connect an endpoint (route) with a python function. On this function you can do anything you can do on any python function. Just remember it should return a JSON object and that functions decorated should have unique names.

RETURN JSON

```
from bson import json_util

@app.route("/endpoint")
def function():
    return json_util.dumps(response)
```

Flask requires us to return a JSON file at the end of all functions. **json_util.dumps** is a nice tool to make sure our response is properly converted.

ARGUMENTS AS PART OF THE ENDPOINT

```
@app.route("/user/<name>")
def welcome_user(name):
    return {"Welcome":name}
```

To have your arguments as part of the endpoint, just enclose the variable name in **<>** on the route and make sure to use the same name on the definition of the function.

ARGUMENTS AS QUERY PARAMETERS

```
from flask import request

@app.route("/user")
def welcome_user():
    name = request.args["name"]
    return {"Welcome":name}
```

If, however your arguments will be query parameters, you must use the **request** module. **request.args** will be a dictionary of all the query params from said request.

Query Parameters

http://127.0.0.1/user?name=John&last_name=Doe

Designing the API

Endpoints (routes)

List all the endpoints you will need, which parameters they need and what they will do.

endpoint	args	function
/help		Shows available endpoints and how to use them.
/hello/<name>	name (endpoint)	Welcomes user with name given on the url.
/user/add	name (request)	Adds user to database with name sent as query parameter

endpoint

A request to a specific endpoint will call the function it decorates.
A request may or not contain query parameters.

JSON

decorated function

return

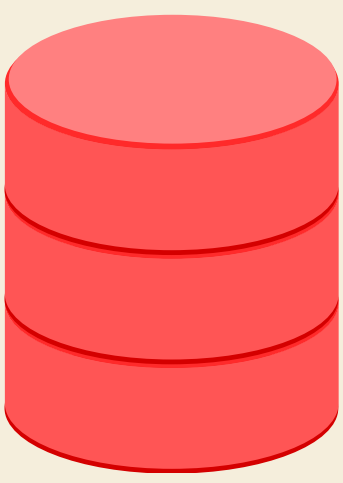
Endpoint arguments will be set on the endpoint (**<name>**) and on the definition of the function. They are your local variables and ready to use. Query params are stored on a dictionary **request.args**

After all is said and done, your endpoint function should return a JSON object, that will be your API response.

Secondary functions

algorithms

database connection



Database

Your function assigned to an endpoint may call on how many other functions you may need, processing data, using databases, etc.

Requesting to the API

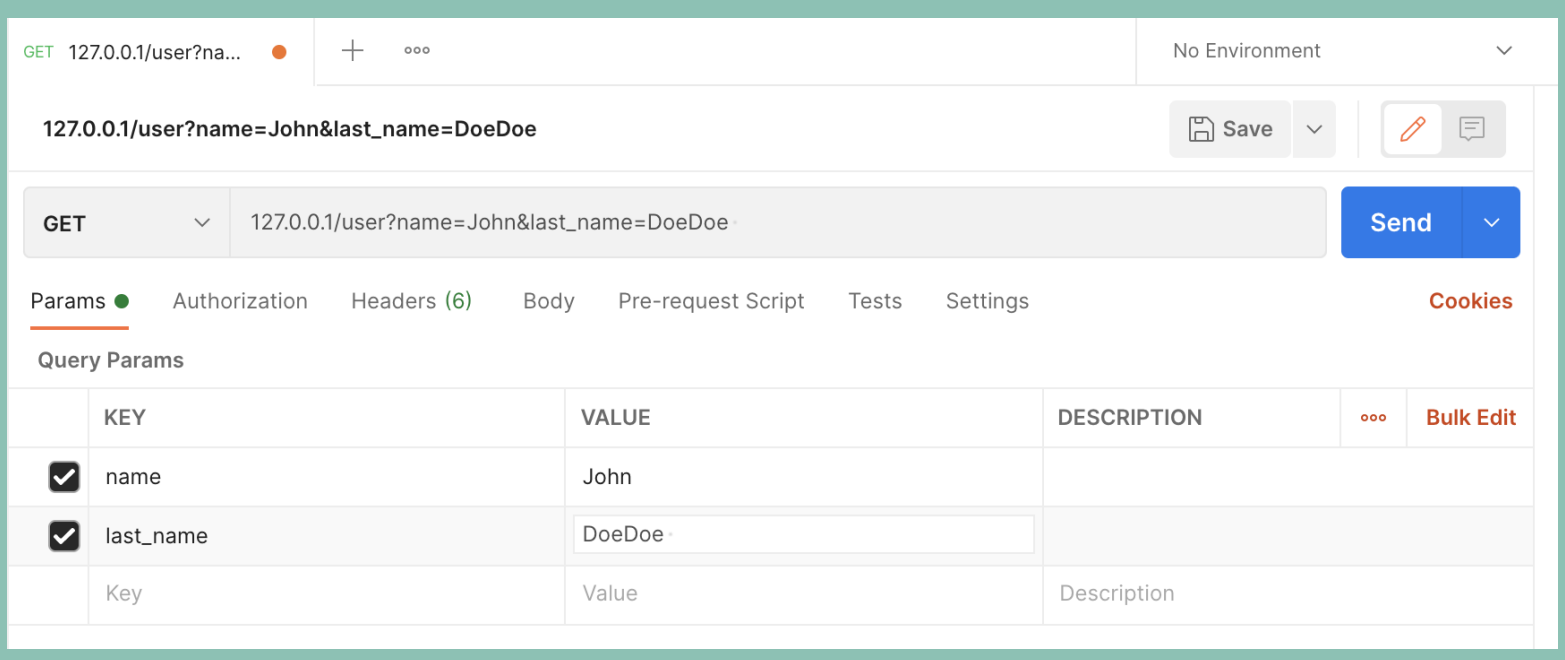
On a browser

Just type the url on the address bar with the proper endpoint and query params.

http://127.0.0.1:5000/user?name=John&last_name=Doe

On a program

On a dedicated program such as Postman, you will need to find the requests tab, fill the url and the key:values for the params. The url will be automatically updated to include the params.



On Python

Use the **requests** module, make sure to have your url properly set as a string and the query params as a dictionary

```
import requests

url = "http://127.0.0.1:5000/user/"
query_params = {
                  "name" : "John",
                  "last_name" : "Doe"
                }

response = requests.get(url, params=query_params)
data = response.json()
```