CS421/621 Lab05

Python & API

Objectives:

- Python programming and retrieving data using API

In this lab tutorial on APIs in Python, you will learn how to retrieve data from remote websites. Most of the popular websites make some part of the their data available for others through their **A**pplication **P**rogramming **I**nterface.

You have learned the importance and the practical aspects of the API usage in lecture 5. Today, we will be using APIs to retrieve data. I recommend you to use Anaconda& Jupyter notebook for this lab. If you haven't tried iPython before, I suggest you to do so. (You can use any IDE or editor for this lab, anaconda and Jupyter iPython is just a suggestion)
Here are the links for installation;

Anaconda https://www.anaconda.com/products/individual

Jupyter Notebook. https://jupyter.readthedocs.io/en/latest/install.html#installing-jupyter-using-anaconda-and-conda

A useful tutorial about jupyter
https://www.youtube.com/watch?v=HW29067qVWk&feature=youtu.be

What is CURL?

It is a tool to transfer data from or to a server without user interaction. CURL stands for "Client URL" and it provides a library and command-line tool. It is a widely used tool because of its ability to be flexible and complete complex tasks.

Syntax:

curl [options] [URL...]

Some examples:

curl https://www.google.com
curl GET https://www.google.com
curl --location --request GET google.com

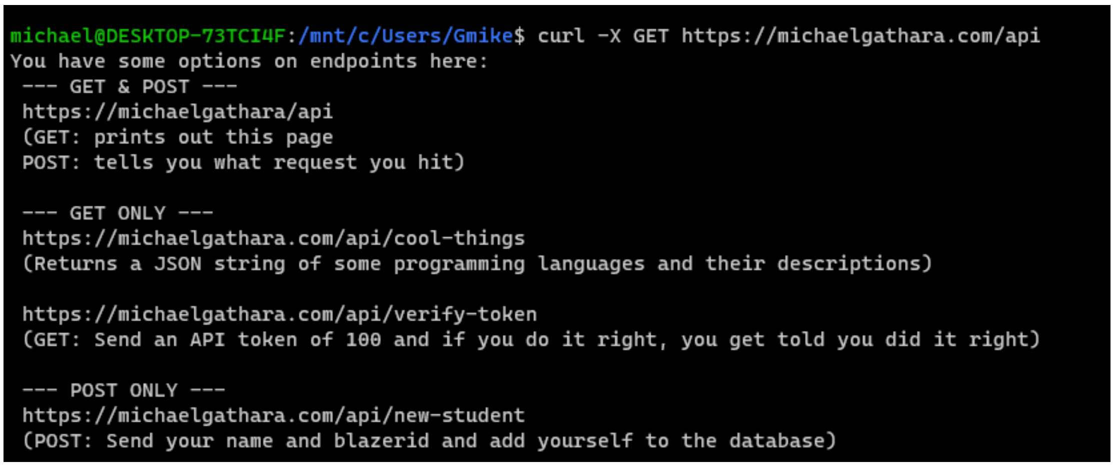
Attempt `curl --location --request GET google.com` vs `curl google.com`

Exercise 1

Let us get started with the API. Each API is different but in general, they all have some form of documentation. To acquire some documentation for our API, you need to make a GET request to the API itself.

```
curl -X GET https://michaelgathara.com/api
Red = Options
Blue = URL
```

You should get back a small set of docs that looks like this:



```
michael@DESKTOP-73TCI4F:/mnt/c/Users/Gmike$ curl -X GET https://michaelgathara.com/api
You have some options on endpoints here:
 --- GET & POST ---
 https://michaelgathara/api
 (GET: prints out this page
 POST: tells you what request you hit)

 --- GET ONLY ---
 https://michaelgathara.com/api/cool-things
 (Returns a JSON string of some programming languages and their descriptions)

 https://michaelgathara.com/api/verify-token
 (GET: Send an API token of 100 and if you do it right, you get told you did it right)

 --- POST ONLY ---
 https://michaelgathara.com/api/new-student
 (POST: Send your name and blazerid and add yourself to the database)
```

We can see that we have some options for endpoints to hit whether it be GET requests, POST requests or both. Say we wanted to send a POST request instead to the same endpoint as above

```
curl -X POST https://michaelgathara.com/api
Red = Options
Blue = URL
```

(Attach a screenshot of the output in your submission)

You can also add more options to your request, such as an API key if the API endpoint requires API keys for authentication. The **/verify-token** endpoint requires you to send an API key of “100” with your request.

```
curl -H "x-api-key: 100" https://michaelgathara.com/api/verify-token
```

(Attach a screenshot of the output in your submission)

Try the above CURL request with the wrong API token, and **attach a screenshot with your submission**

You can also send different types of data with requests, for example you can send JSON objects as well. The endpoint **/api/new-student** allows for you to send JSON objects with your request. It also only supports POST requests.

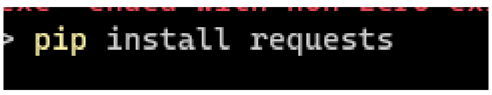
```
curl -X POST
-H "Content-Type: application/json"
-d '{"name": "Michael Gathara", "blazerid": "mikegtr"}'
https://michaelgathara.com/api/new-student
```

(Attach a screenshot of the output in your submission)

Exercise 2

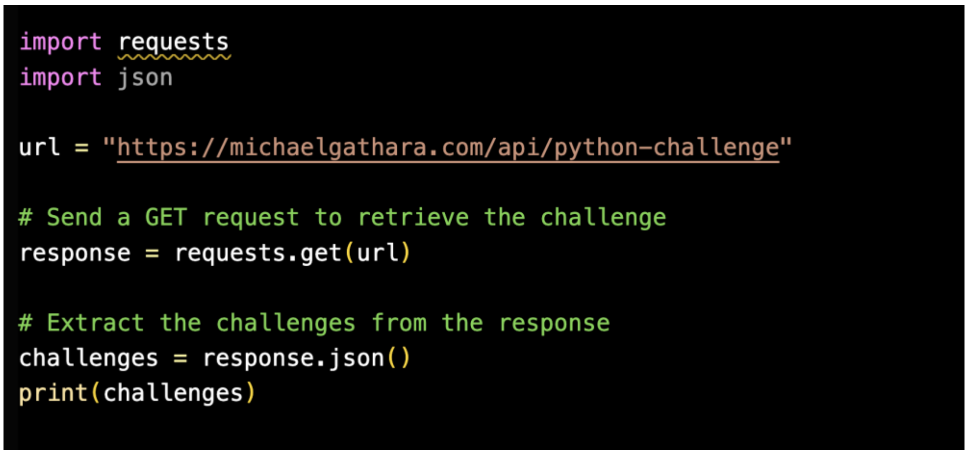
We are going to use Python to send automatic GET requests based on the data sent by the GET requests.

You will start by defining a Python program that uses the requests library to request the Python challenge problems. First pip install the requests and json libaries if you do not already have them.



```
> pip install requests
```

To retrieve the data, you will send a GET request to the URL and the data will be returned as a JSON object



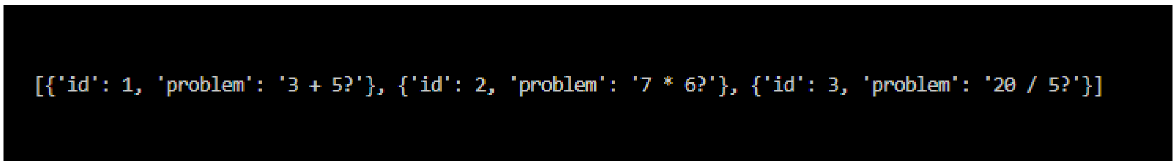
```python
import requests
import json

url = "https://michaelgathara.com/api/python-challenge"

# Send a GET request to retrieve the challenge
response = requests.get(url)

# Extract the challenges from the response
challenges = response.json()
print(challenges)
```

Printing the data returns



```
[{'id': 1, 'problem': '3 + 5?'}, {'id': 2, 'problem': '7 * 6?'}, {'id': 3, 'problem': '20 / 5?'}]
```

Write a Python program that will parse the response from the URL and solve the problems given. Print the problem answers and turn them in alongside the screenshots of your answer. Your answer should include your name and Blazer ID at the top. Your solution should work regardless of how many problems are returned. Do not hardcode your answer

Deliverables:

- **Your screenshots**
    - Make sure you name your screenshots appropriately
    - These should be turned in on GitHub
- **Your Python Program**
    - On GitHub
- **ICF (on canvas)**
    - On Canvas
- **A link to your GitHub repository**
    - **(You should be using the same repo for all your labs in this class do not create a new repository for this lab.)**
    - **On Canvas**

References & Recommended Links

https://rapidapi.com/blog/how-to-use-an-api-with-python/

https://documenter.getpostman.com/view/10808728/SzS8rjbc?version=latest

https://towardsdatascience.com/covid-19-data-collection-a-python-api-story-347aafa95e69

https://github.com/ExpDev07/coronavirus-tracker-api/blob/master/README.md#picking-data-source

https://github.com/Kamaropoulos/COVID19Py

https://curl.haxx.se/

https://man7.org/linux/man-pages/man1/curl.1.html