# CSCD 330 - Computer Networks

Lab 3, Flask Server

## Overview

Write an API server using the flask library. The server will have at least 2 routes. 1 for converting from a domain name into a business address. 1 for converting from a domain name into the weather.

## Instructions

Complete the above program following the steps in the next section. You must use the same APIs as used in lab 2. Call this file lab3.py and include your name at the top of the file – #author: YOUR NAME. Your output should match the provided example output. You must also create a README explaining how to run your program and answering the questions below. Furthermore, you must create a test script in Bash. Use `curl` for testing. Call this file test.sh.

You may only use the following imports:

from flask import Flask
from json import loads
from requests import get
from socket import gethostbyname
from subprocess import getstatusoutput

### Steps:

#### 1. Create the API server:

The flask server must support 2 API calls to the following specifications:

1. Accepts a URL in the format "/address/<domain_name>" and returns the physical address from the `whois` entry.
2. Accepts a URL in the format "/weather/<domain_name>" and returns the weather using the same APIs as in lab 2.

**2. Cache requests:**

To save on network traffic, cache requests such that duplicate calls are not rerun. Instead return the original response prefaced with "Cached:" so that we know the cache is working. Python Dictionaries work well for this.

### Questions:

1. Identify the following in the URL: http://localhost:5000/weather/google.com
    - Domain:
    - Path:
    - Port:
    - Protocol:
2. Identify the following in the URL: https://translate.google.com/
    - Domain:
    - Subdomain:
    - TLD:
    - Path:
    - Protocol:
    - Port:
3. What is a Python decorator?
4. Is there any problem with your cache implementation? Would your cache implemenation work in production?

## Turn in:

Submit a tarball with the following:
- Your source code (in Python) called lab3.py
- Your test script (in Bash) called test.sh – test at least 3 inputs.
- Your README answering the above questions and explaining how to run your program.

In case you have forgotten:

```
tar -czvf lab2_YOURNAME.tar.gz *.py *.sh README
```

## Example output:

**curl localhost:5000/address/google.com**

```
1600 Amphitheatre Parkway, Mountain View, CA, 94043
```

**curl localhost:5000/address/google.com**

```
Cached: 1600 Amphitheatre Parkway, Mountain View, CA, 94043
```

**curl localhost:5000/weather/google.com**

```
Sunny, with a high near 74. North wind 2 to 10 mph.
```

**curl localhost:5000/weather/google.com**

```
Cached: Sunny, with a high near 74. North wind 2 to 10 mph.
```