INFO3180 Lab 1 (20 Marks)

Due Date: **February 5, 2023 at 11:55pm**

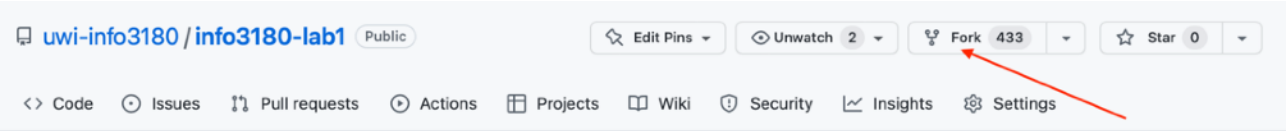
Flask

In this exercise you will install flask, create some routes, customize a template.

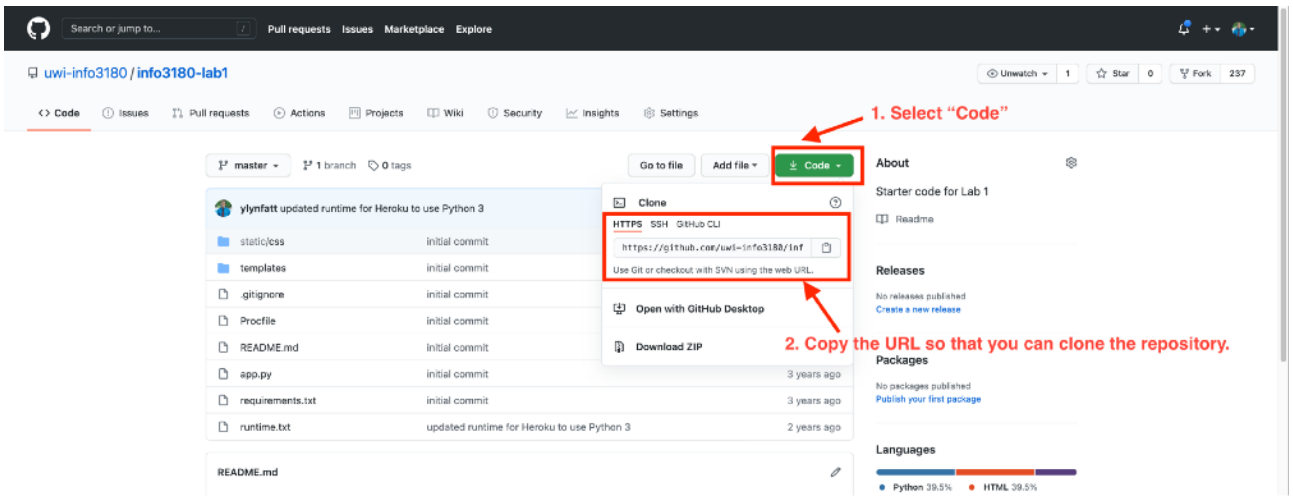
Fork and Clone the repository

A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. Most commonly, forks are used to either propose changes to someone else's project or to use someone else's project as a starting point for your own idea. In our case for this lab we are using another project on Github as the starting point for this Lab.

Login to your Github account and fork the repository by visiting the link below and clicking on the "**Fork**" button:
https://github.com/uwi-info3180/info3180-lab1/



Next, to clone it to your local machine you will first need to get the URL of the new repository that you just forked to your account:



Then do the following on your own local computer from the command line/prompt in a folder of your choice. Ensure you change **{yourusername}** to your actual Github username:

```
git clone https://github.com/{yourusername}/info3180-lab1
info3180-lab1
```

Step 1 - Setup and activate a virtual environment

Navigate to the folder with the code you just cloned and setup a virtual environment.

```
cd info3180-lab1
python -m venv venv
or
python3 -m venv venv (if you have both Python 2 and Python 3)
```

Activate the environment

```
source venv/bin/activate
or
.\venv\Scripts\activate (if using Windows)
```

Step 3 - Install Flask

Install Flask by running the following command:

```
pip install Flask
```

You'll see output similar to this:

```
(venv) ➜ lab1 pip install Flask
Collecting Flask
  Using cached Flask-0.12-py2.py3-none-any.whl
Collecting Jinja2>=2.4 (from Flask)
  Using cached Jinja2-2.9.4-py2.py3-none-any.whl
Collecting Werkzeug>=0.7 (from Flask)
  Using cached Werkzeug-0.11.15-py2.py3-none-any.whl
Collecting click>=2.0 (from Flask)
  Using cached click-6.7-py2.py3-none-any.whl
Collecting itsdangerous>=0.21 (from Flask)
Collecting MarkupSafe>=0.23 (from Jinja2>=2.4->Flask)
Installing collected packages: MarkupSafe, Jinja2, Werkzeug, click, itsdangerous
, Flask
Successfully installed Flask-0.12 Jinja2-2.9.4 MarkupSafe-0.23 Werkzeug-0.11.15
click-6.7 itsdangerous-0.24
```

Step 3 - Create a Route and View Function

The main application code is located at **app.py**. You will notice that we have already imported the Flask library and initialized the application:

```
from flask import Flask, render_template
app = Flask(__name__)
```

However, we have not yet created a route and it's respective *view function*. So let's do this now. Our first route will be **'/'** and will represent our home page. We do this by using the **@app.route** decorator from Flask. Next we'll create a view function called **'home'** and we'll simply return the message **'My home page'**.

```
@app.route('/')
def home():
    return 'My home page'
```

Now let us start the Flask development server:

```
$ flask --app app --debug run
```

This tells Flask how to start the development server and to and also to turn on **debug mode**.

You will see output similar to this:

Now open your web browser and browse to http://127.0.0.1:5000.

You should simply see '**My home page**' being displayed.

Step 4 - Push your code to your Github repository

Now that you've created your first route, it's time to add, commit and push your code to your Github repository.

```
git add .
git commit -m 'created my first Flask app'
git push origin main
```

Step 5 - Create another route and create a template to render.

We will now create another route and its respective view function, but this time we will render a Jinja2 template using the **`render_template()`** method. The new route is '**`/about`**' and the view function should be called '**`about`**':

```
@app.route('/about')
def about():
    return render_template('about.html')
```

Now create a new template file in your ***templates*** directory called '**about.html**'. In that file, create a basic HTML page with the HTML code below. Ensure that you put your name within the level one heading (**<h1></h1>)** and write a short paragraph within a paragraph tag (** **) about yourself.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>My Website - About</title>
</head>
<body>
    <h1>Place Your Name Here</h1>
    <p>Write a short paragraph about yourself here.</p>
</body>
</html>
```

Now browse to your new route in your web browser to see your new page. Feel free to customize this some more if you'd like but ensure you have the <h1> heading and the paragraph.

Step 6 - Push your code to your Github repository

Now that you've created your second route, it's time to add, commit and push your code to your Github repository.

```
git add .
git commit -m 'created my first template file'
git push origin main
```

Step 7 - Style your Page and add an Image

Now that you know how to render and use a template, let us add a CSS file to our template, style the page and also add an image.

Open your **about.html** template file and in the **<head></head>** tags add the following:

```
<link rel="stylesheet" href="{{ url_for('static', filename='css/app.css') }}" />
```

Update your **app.css** stylesheet and add some CSS to change the font-size, font-family and colour of the text in your web page or any other styles that you would like to make your webpage look like your own.

Next, put an image of your choosing in your **static** folder. Now in your **about.html** template file load that image by using the **url_for()** flask function in an **<img />** tag.

Now check your webpage again and ensure that your image is showing and also that the styles you added to your CSS file are also working.

Step 8 - Push your code to your Github repository

Now that you've added your image and styled your page, it's time to add, commit and push your code to your Github repository.

```
git add .
git commit -m 'Added an image and styled the webpage.'
git push origin main
```

Submission

Submit your code via the "Lab 1 Submission" link on OurVLE. You should submit the following links:

1. Github repository URL for your Flask Exercise e.g. https://github.com/{yourusername}/info3180-lab1

Grading

1. Define the route and created associated view function for home page which returns the string **'My home page'.** (3 marks)
2. Defined the route and created associated view function for the About page. The view function should utilize the appropriate function to render the template for the about page. (5 marks)
3. The template for the about page should be created in the **templates** folder with HTML specified in the example and your name should be in the **<h1></h1>** heading and paragraph about yourself in a ** ** tag. (5 marks)
4. Your webpage should load the **app.css** stylesheet using the **url_for()** function and your web page should be styled. (2 marks)
5. You should have an image in your **static** folder and that image should be loaded using the **url_for()** function in an **<img />** tag and display on your about page. (3 marks)
6. You should be able to successfully view the about page in web browser. (2 marks)