CPSC 476 - Back-End Engineering - Spring 2018

Project 2, due March 20

Introduction

In this project, you will refactor MiniTwit to use the Web Service API that you created in Project 1, then run multiple instances of both the front- and back-end servers behind a load balancer.

Test Environment

You may use any platform for development and testing, and NGINX is partially supported on Windows, but note that per the Syllabus the test environment for projects in this course is the Ubuntu MATE VM for available from http://michael.shafae.com/#resources.

Refactoring MiniTwit

Remove all references to `sqlite3` and all database queries from `minitwit.py`, replacing them with requests to `mt_api.py` via the Requests library. Replace the `DATABASE` configuration setting with a new `API_BASE_URL` setting.

In order to run `minitwit.py` and `mt_api.py` at the same time, you'll need to run them on different ports (e.g., 5000 and 5001).

(Tip: When installing Requests, you can safely ignore Python 3. The library is officially supported on Python 2.7. Also note that you can use `pip` without switching to `pipenv`.)

It's not impossible that you will run into a design mistake in your API from Project 1, resulting an an action in the MiniTwit front-end that you cannot accomplish directly using the back-end API. If so, don't panic: modify the API as necessary and document the required changes.

Running Multiple Application Servers

To simulate running MiniTwit in production, you will need to run three instances of `minitwit.py` and three instances of `mt_api.py`. To do this, you can create a `Procfile` and use the `foreman` command-line utility.

To install foreman on the Ubuntu MATE VM, use the following shell commands:

```
$ sudo apt update
$ sudo apt install -y ruby-foreman
```

(Tip: in recent versions of Foreman, the `-c` option for concurrency has been replaced by the `-m` or `--formation` option. The version of foreman installed with Ubuntu 16.04 continues to use `-c`. Note also that `bundle exec` is a command for running Ruby applications. You'll want to use `flask run` instead.)

Load Balancing

Install NGINX and verify that you can see the *Welcome to nginx!* page on http://localhost/.

Now configure two different NGINX server blocks:

- One block, listening on port 80, using the three `minitwit.py` instances as upstream servers.
- Another block, listening on port 8080, using the three `mt_api.py` instances as upstream servers.

Set the `API_BASE_URL` for `minitwit.py` to the load-balanced URL `http://localhost:8080`.

Testing

You will know that the application is working when you can log into and use MinitTwit on http://localhost/ and see activity logged by foreman for all three front-end and all three back-end processes.

Submission

Turn in the code for your project by placing `minitwit.py`, `Procfile`, modified configuration files from `/etc/nginx`, and any other relevant files in the `project2/` subdirectory of the folder that was you on Dropbox. If you needed to modify `mt_api.py`, include the updated code and documentation. You may work alone, or make a single submission for a team of 2-3 students. If you work in a team, make only one submission.

To complete your submission, print the following sheet, fill out the spaces below, and submit it to the professor in class by the deadline. Failure to follow the instructions exactly will incur a **10%** penalty on the grade for this project for all students on the team.

Project Submission

CPSC 476, **Section 1**

Project Number ____________________

Names of up to three students for this submission

1. ____
2. ____
3. ____

CSUF email of the Dropbox account containing the project files for this submission

____@csu.fullerton.edu

Comments on your submission

____

____

____

____

____

____

____

____

____

____

____

____

____

____