Coursework Report

Niko Triantafillou

40405829@napier.ac.uk

Edinburgh Napier University – Advanced Web Tech (SET09103)

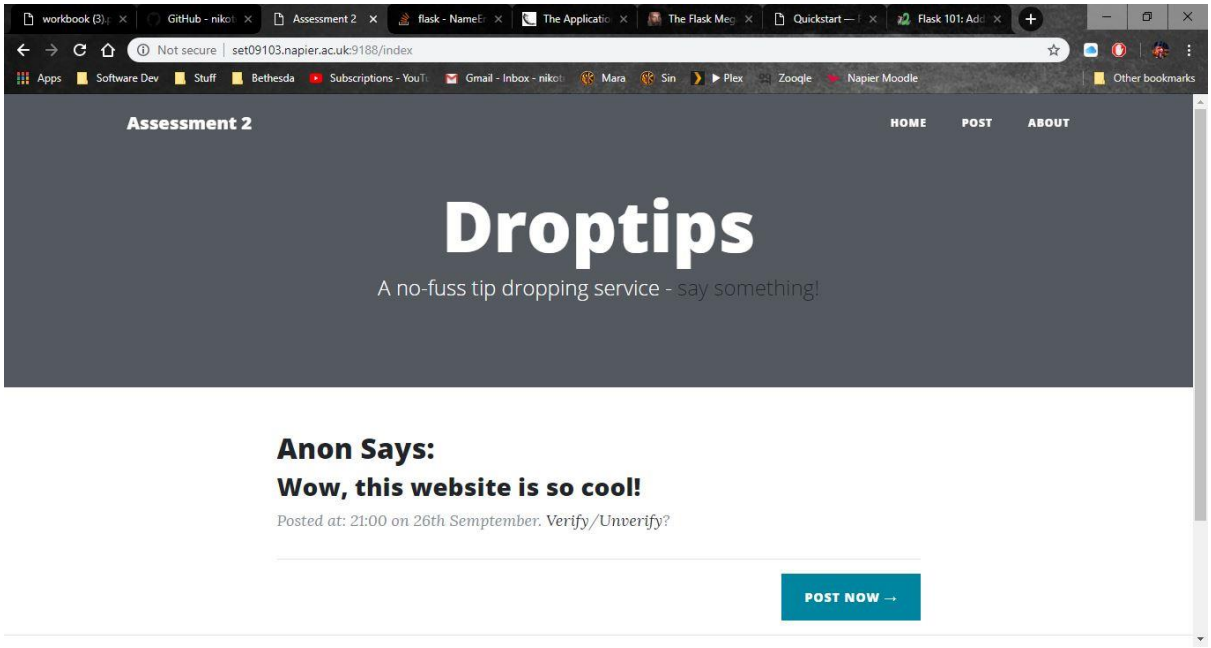
Assessment 2: Droptips

Introduction

Assessment 2: Droptips is a web application that utilises the technologies of Python, Flask and Jinja2 to create a pseudo-Twitter clone without the need to log-in or sign-up to use the service! The concept behind the service is to allow a self-validating community to police itself using a simple verify/unverified token system. If a user has posted something they have seen, an other user can verify the authenticity of the tip!

The web application takes advantage of Flask routing to ensure the traversal of the application is as smooth as possible for the user, whilst operating Flask static files for hosting the Bootstrap framework. In addition, I have incorporated the Flask extensions of WTForms for form validation and, to a lesser extent, some SQLAlchemy/SQLite3, but this was unfortunately cut out of the prototype web application due to server permission issues using an excess backlog of development time.

The specific content found in the web application revolves around a user-generated, user-driven environment where anyone can post, from anywhere, about anything.



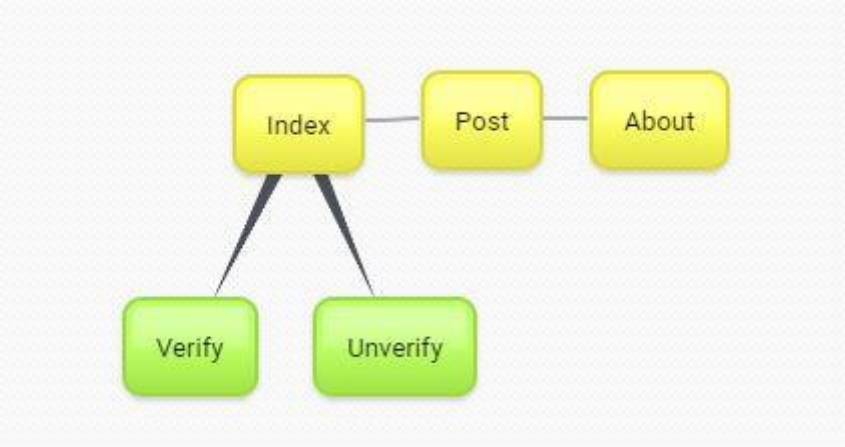
Design

The layout is a basic template layout using Bootstrap version 4.1.3, straight-out-of-the-box. The layout was heavily influenced from start-bootstrap.com, a free Bootstrap resource, that allows anyone to use the content found on their website royalty-free.

Whilst the prototype does not feature any imagery, I feel the design does invite users to get involved. Consistency aids in user accessibility and familiarity, and the purpose of posting without requiring a user account creates a nice, content driven network of dynamically relevant information based on user interaction.

For the body content, I have used the Bootstrap container wrapper. The user posts featured on the index are neatly laid out using custom CSS specifically tailored to message posting, but the content is displayed directly from Python Flask's micro-framework, that provide the user with a seamless user experience.

As I ran into server issues, I was unable to fully test and develop the database system to allow for proper user-driven content, however I was able to utilise WTForms to handle the form validation from within the Flask views. It allows for CSRF protection right away, which is cross-site request forgery. This aids in preventing unauthorised commands that specifically target state-changing requests.



Enhancements

The features that I would add would be an interactive administration panel that would allow full hierarchal control of the entire web application straight from an admin's login; this would also require a fully fleshed out login/logout functionality, making use of a powerful relational database resource such as PHPMyAdmin to facilitate the necessary database needs of a large and powerful Twitter contender.

In addition, I would look to incorporate a member log-in functionality for more avid users of the website, whilst also maintaining the values instilled from the landing page of a free, no fuss tip dropping service. Furthermore, I would look to allow image posting as part of the tip dropping service, which would only enhance the user experience and breathe more life into the application.

Critical Evaluation

I have structured the architecture so that every file has a proper namespace and location. From the top-level folder, the "run.py" file can be utilised in Linux to launch the application. Within the "source" folder, the "__init__.py" file creates an instance of the Flask module, whilst the "views.py" file controls the Flask routes and Jinja2 is used to import the Bootstrap CSS and JS files into the application from the "header.html" and "footer.html" files respectively.

I am particularly happy with the usage of the Flask extension WTForms, as it adds its own validation and inherent CSRF protection.

What didn't work too well was the inclusion of SQAlchemy; whilst I understood the basis of the extension, and the powerful Python SQL toolkit it provided, I was unable to utilise it due to the apparent server issues blocking my database development.

Personal Evaluation

I feel like I performed quite well. Using the available resources throughout the module workbook and additional resources found on the Internet, I was able to create a simple web application pseudo-Twitter clone.

Using my personal experience in software development (I am a self-taught VB/C# .NET junior developer with 2 years of experience in the industry), I was able to quickly understand the way Python, Flask and Jinja2 were presented and make connections on how these technologies worked on a fundamental level.

The challenges I faced were specifically tailored to learning the Linux operating system; I found it incredibly difficult using a command-line based system for the entirety of the applications development, but I eventually found ways to circumvent or make things easier when re-iterating testing cycles of the application (the Up arrow is your best friend to repeat past commands).

Whilst I endeavoured to flesh out the database modules to incorporate a truly dynamic pseudo-Twitter clone like feel, I could not get passed a seemingly crippling server permission error whereby the database could not be located, or the pre-rendered SQLite3 table could not be loaded. I have come to this conclusion based off past experiences on working out-with an environment you do not own, and because of my own personal research into the specific error messages thrown at me by the Linux shell and other user reports of similar issues.

References

- To gain a better understanding of web forms and databases in Python: https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-i-hello-world
- To understand SQLAlchemy: http://flask-sqlalchemy.pocoo.org/2.3/quickstart/#a-minimal-application
- To understand Python Flask better and generating dynamic data: http://www.blog.pythonlibrary.org/2017/12/14/flask-101-adding-editing-and-displaying-data/
- For general Python questions: http://www.stackoverflow.com
- For Bootstrap development: https://getbootstrap.com/docs/
- Free Bootstrap themes: https://startbootstrap.com/
- For Navigation Map: https://bubbl.us