Project#3 Proposal: FCC Reporting Efficiency

Kurt Stoneburner

All television broadcast stations, both commercial and noncommercial, must prepare a list of important issues facing their communities of license and the programs aired during July, August, and September dealing with those issues as part of their broadcast license requirements.

Full-Stack Delivery

This project will consist of front-end and back-end processes. The backend will include

- Adobe Analystics
- Python: Web Scraping, push content to Firebase
- Firebase: Hosted Online Database

Front-End:

- Browser based user interface hosted on github.io

Backend Overview

This project streamlines the gathering of important stories and topics and reduces the overall workload of report generation. I've already created a basic version of this project, that scrapes the company website and generates a spreadsheet of stories applicable for reporting to the FCC. My boss selects stories from the spreadsheet and pastes them into a final document for submission to the FCC.

The primary goal of this project is to further streamline the reporting process and reduce the overall time commitment for generating these reports.

The current version generates content by scraping a fixed list of URLs that associated with Tagged Keywords. The selenium based scraper clicks a show more button until a quarters worth of stories is displayed, then scrapes all of them. Results are output to an excel spreadsheet.

The biggest issue is the tags are arbitrary and are tied to a synthetic image the company wishes to project. There are no metrics tied to the current process.

The first step is to integrate with Adobe Analytics. Reports can be created that show the top 100 URLs by Unique Visitors in a given quarter. These results can be exported to a CSV.

Each URL will be downloaded using Selenium

As part of basic Processing, A Dictionary of Tags will be created

Each Tag, will be summed with the Unique visitor count for each story. This will generate a weighted list of popular tags. The weighted tags will eventually be used to display the URLs by popularity of tag names. I have a rough of idea of this process, but it's a bit squishy at the moment.

The Downloaded and Processed stories will be converted to a dictionary and uploaded to a free Firebase account. Firebase is Google's platform for managing mobile and web applications. The free tier supports API based database access.

Client Side Overview

A big appeal for Firebase is it can be accessed using client-side (browser) code. This means the webpages hosting the Firebase connectors can be hosted on Github Pages. This completely side-steps backend webserver management.

There is will a user interface to select a current quarter. This will pull a list of URLs based on weighted story tag values. The interface will consist of a list of Story Titles with the ability to click on any story to expand the whole text. Ideally, the user will be able to select a number of items to export. Once all items are selected, there will be an export process. Most likely, formatted text will be placed in the clipboard for pasting into a final document. Alternatively, formatted text can be exported to a file. The export process and ultimate user interface will be heavily dictated by consulting with the user (my boss). If I'm going to the trouble to build a tool, it should be a bespoke tool.

The basic protoype should take 20-40 hours to construct.