Jonathan Smoley
Professor Bowers
CPSC 324
22 April 2024

Project Check-In

Research

Due to early explorations into web scraping, GCP tools, and community projects dealing with cloud-based scraping pipelines, I have found ample resources detailing what services are needed and what libraries are available for this project. Within the GCP environment, Cloud Scheduler can be used to publish an action to Cloud Pub/Sub. Once the action is seen by Pub/Sub, a subscribed Cloud Function to run a web scraper can be notified. From this stage there are a few options: the Cloud Function could parse through the HTML documents received and produce the desired output, or intermediate HTML documents could be saved to Cloud Storage. If the former is chosen, the only other step would be to deliver the result… more on this later. Should the latter option be taken, an additional Pub/Sub step would be required to scan for updates to the bucket in Cloud Storage. This may be a better option as the data retrieval and data parsing steps can be split up into separate Cloud Functions.

Implementation

As for actual development of this pipeline, I have since web scraped my dataset with a number of different tools to test which may work best. With wget and curl, I can get the entire contents of Gonzaga's public webpage, but these need to be run using a shell script or using the subprocess library in Python. Python does open the opportunity to learn how to use Beautiful Soup and Scrapy; however, in testing both of these I found no end to the data (called 'items') that could be scraped. Further testing will be needed to find a finite amount of data to collect. On the GCP side, I have spun up test Pub/Sub jobs and timed triggers in Scheduler. I have not yet tested a full pipeline from scheduler to storing a document, but I have plans to begin with a basic web scrape of accessible urls. Currently, the largest stress points in this project align with parsing text from the collected HTML documents and delivering the resulting data. Parsing text from HTML is primarily concerning due to inexperience with the process. In contrast, delivering the result brings up concerns over the best course of action. Meaning, a "modelfile" with the context generated from this pipeline could be sent to a server where the project is deployed or pushed to the GitHub repository that hosts the project.

In sum, there has been a lot of exploration work and not as much real implementation. I estimate that this project is around 45-50% complete.