GCP FaaS

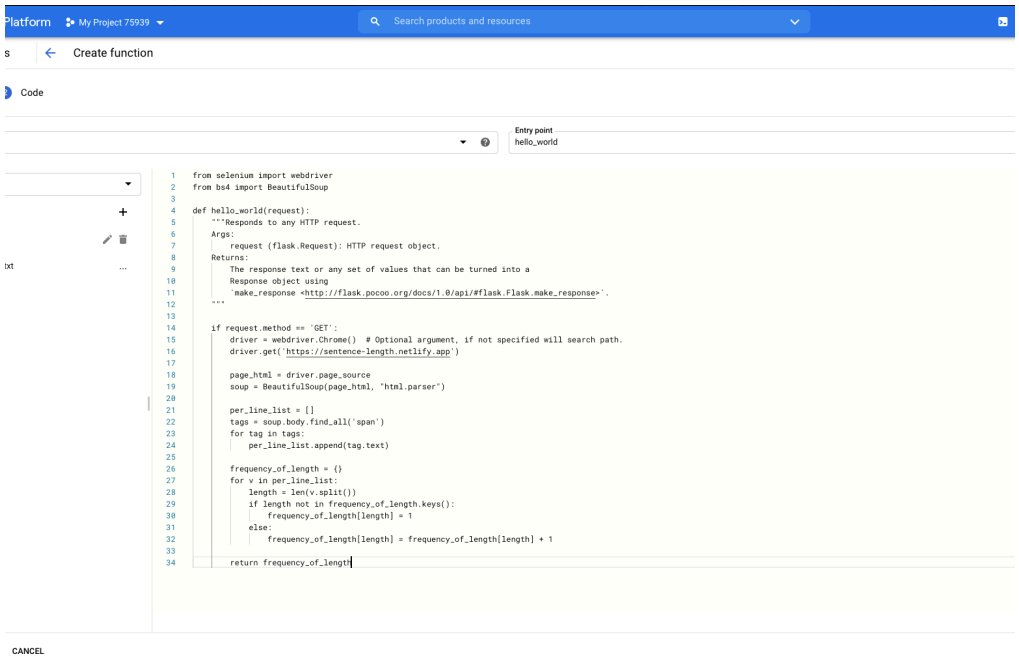
FaaS link: **https://funky1-xoya5vvlsq-uc.a.run.app/input?link=*any url***

Example: **https://funky1-xoya5vvlsq-uc.a.run.app/input?link=https://sentence-length.netlify.app**

Output format: *JSON where key is the the number of words in a line and the value is the count of such lines with the same length.*
To note: I implement my solution in Python.

INITIAL SOLUTION (THAT DIDN'T WORK!):

Initially, I built a simple cloud function in my project to parse html, and calculate the distribution in the output format mentioned above. I used python to do this, with urllib3 and beautifulsoup4 as my main dependencies. The dependencies were added in ***requirements.txt*** while the main code was written in ***main.py***.



FaaS with GCP Cloud functions

However, the above cloud function did not work because even though it succeeded in calculating frequency distribution in static webpages, most webpages are dynamic generated at runtime (like with javascript). Typical HTML parsers and web-scrapers don't work with that because they're not able to fetch the actual content of the webpage that we see in the browser. For example, in the link shared, we don't see any content of the Christmas Carol with our static crawler.

SOLUTION THAT WORKS:

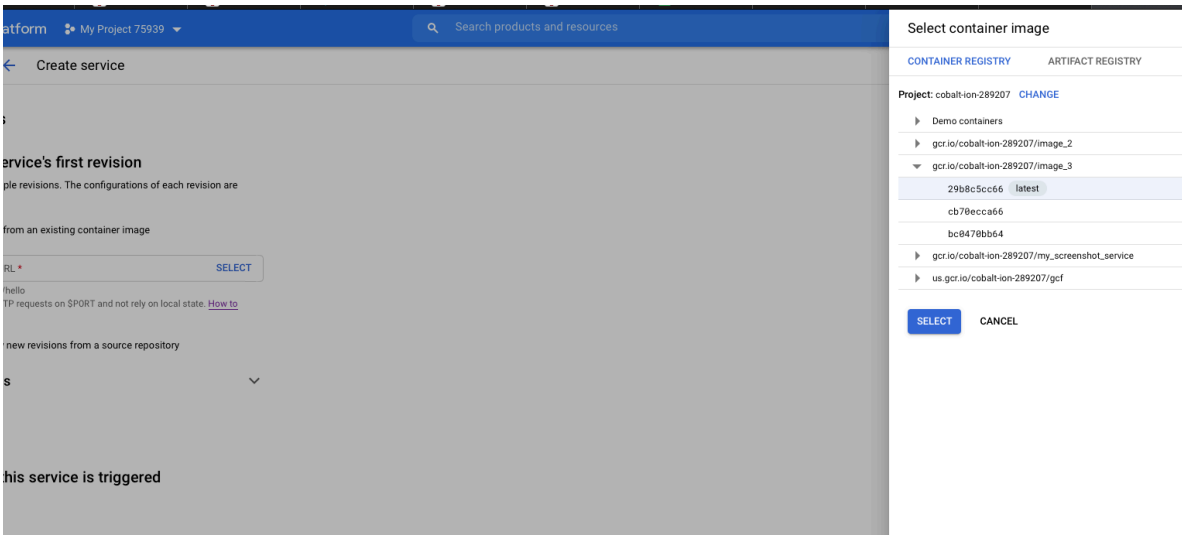
To capture the content of javascript generated webpages, I use Selenium's Chrome driver to crawl and parse dynamic webpages. **However, using selenium and chrome driver requires a chrome binary to be installed with an executable path in the system**. **That's a problem if we want to run serverless cloud functions!** Headless Chrome with Selenium in general doesn't work with GCP Cloud functions. Out of all the languages supported for GCP Cloud functions, the only compatible solution is using *Node.js* with *puppeteer.*
Alternatively, I used Google Cloud Run to pre-build an image with Google Chrome Driver and all the python dependencies with Docker and push the image with my code (main.py).
The Dockerfile is attached in the submission and looks like:

```
ll the missing libraries

-y gconf-service libasound2 libatk1.0-0 libcairo2 libcups2 libfontconfig1 libgdk-pixbuf2.0-0 libgtk-3-0 libnspr4 lib
appindicator1 libnss3 lsb-release xdg-utils


google.com/linux/direct/google-chrome-stable_current_amd64.deb
hrome-stable_current_amd64.deb; apt-get -fy install

omedriver.storage.googleapis.com/2.41/chromedriver_linux64.zip
er_linux64.zip

/usr/bin/chromedriver
t:root /usr/bin/chromedriver
n/chromedriver

endencies.
t requirements.txt
equirements.txt

 the container image.



e on container startup. Here we use the gunicorn
e worker process and 8 threads.
ith multiple CPU cores, increase the number of workers
 cores available.
bind :$PORT --workers 4 --threads 8 main:app
```

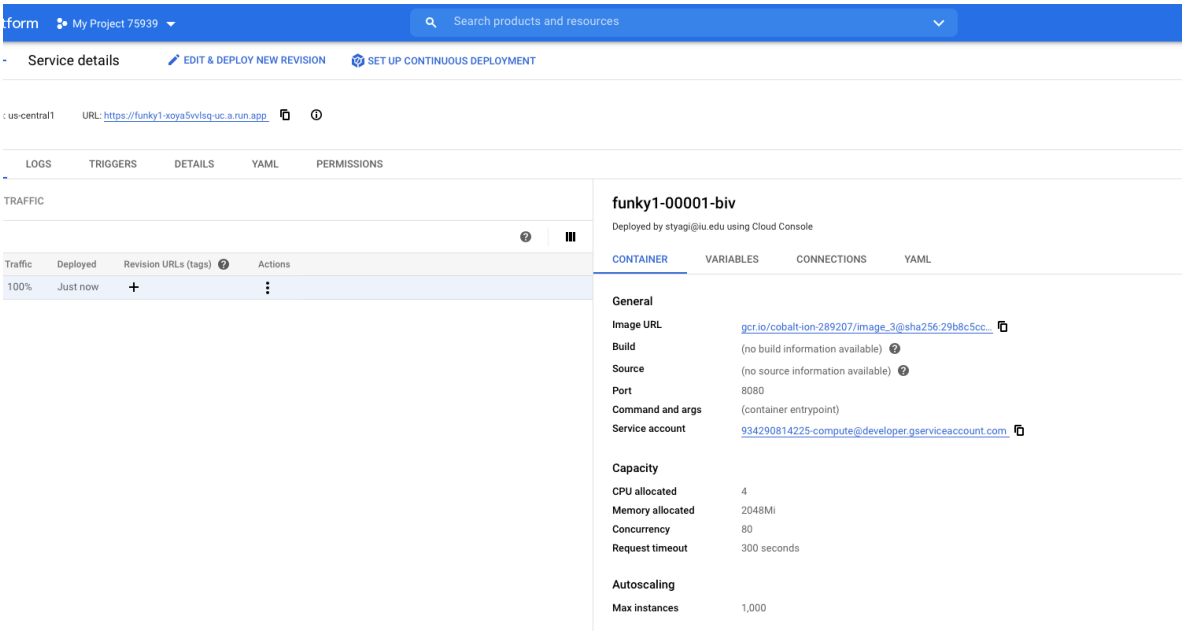
Dockerfile

I build and push my image as named ***image_3*** under project ***cobalt-ion-289207*** with:

```
gcloud builds submit --tag gcr.io/cobalt-ion-289207/image_3
```



Creating our Function with custom image



FaaS running successfully