Final project report

Andrew ID: haichuax

0.How to run these applications

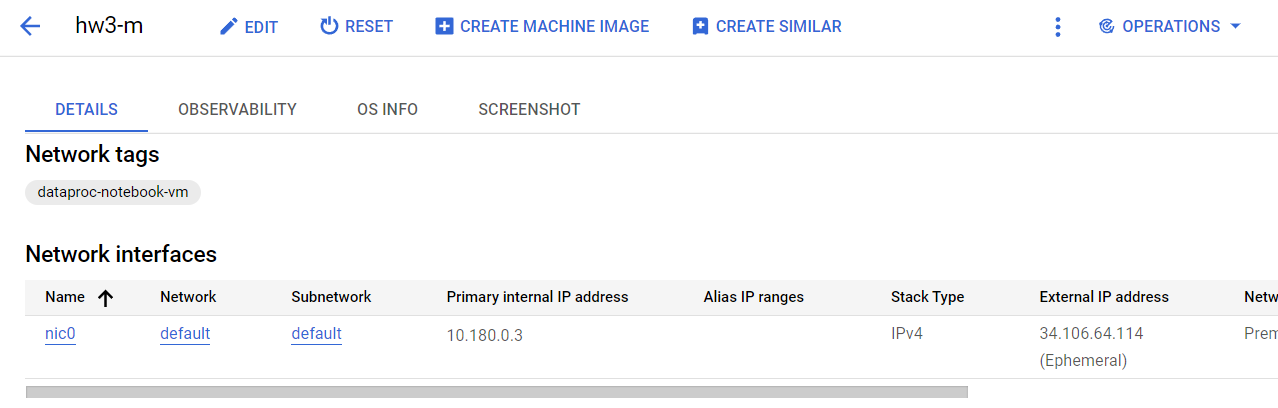
Step 1: Change external IP: 34.106.64.114 (it will change every time I restart the clusters) in:

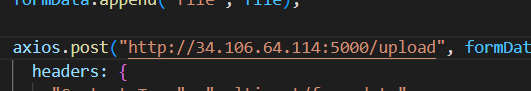
sample\src\pages\landing.js

sample\src\component\TopN.js

sample\src\component\MiniSearchClient.js

Example:





Step2: Start the first application on your PC:

*cd sample  
docker run -it --rm -v %cd%:/app -v /app/node\_modules -p 3001:3000 -e CHOKIDAR\_USEPOLLING=true tonyrays/dockerhub:projtimagepush2*

Step3: Start the second application on GCP:

Connect to compute engine via SSH, move folder flask\_second\_app to the compute engine.

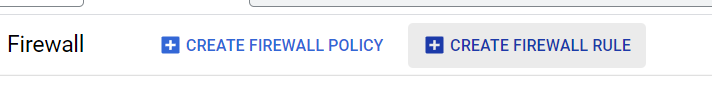
*pip install Flask*

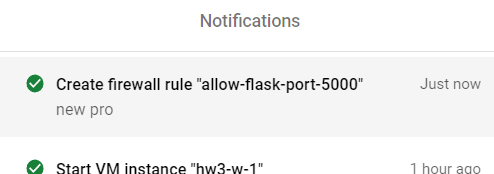
*pip install flask\_cors*

*python app.py*

**# Note1 If TCP connection is blocked:**

Create a firewall rule to allow traffic to port 5000:





**# Note2 Data folder:**

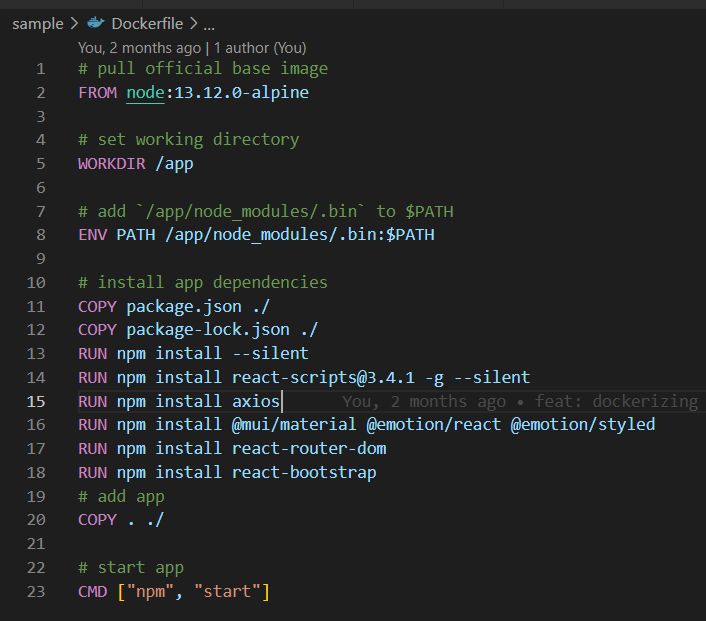
Create a 'Data' folder and insert the necessary data into it. Then, compress the folder into a zip file for uploading. Alternatively, you can use the zip file provided in the extra credit quiz.("./test\_data/

Data.zip")

1. Brief introduction to the first application

The first application contains four frontend pages:

1. Landing.js: where you can upload a zip file to construct Inverted Indicies
2. MiniSearchIndex.js: index page, proceed to (3) or (4), or go back to (1)
3. MiniSearchClient.js: [Search For Term](http://localhost:3001/miniSearchIndex/miniSearchClient)
4. Top\_N.js: TOP-N Frequent Terms
5. dropdown.js: implement a dropdown for better user experience
6. Docker image on dockerhub



Watch the video for more details.

1. How I create the second application step by step

Since I have completed extra credit quiz, so I just modify the mapper.py and reducer.py I used for extra credits.

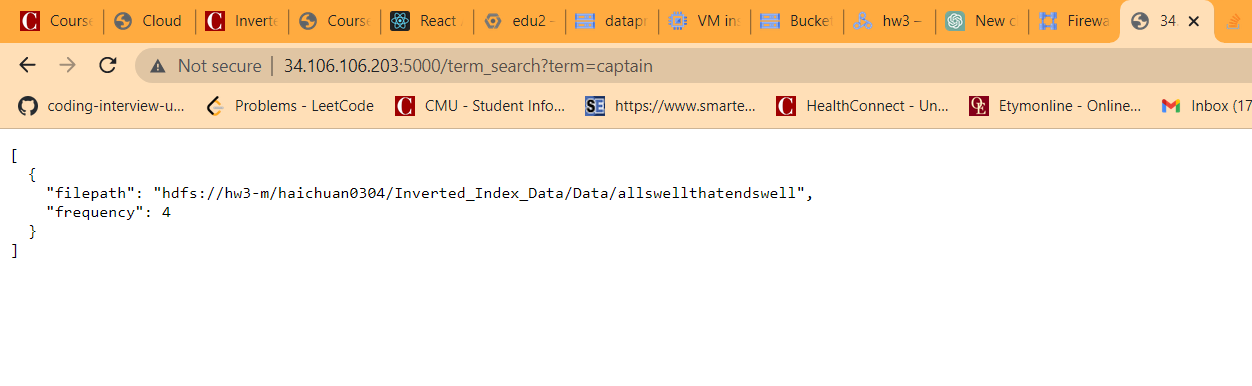
The second application contains:

1. App.py
2. Mapper\_q.py
3. Reducer\_q.py
4. Term\_search.py
5. Top\_n\_search.py

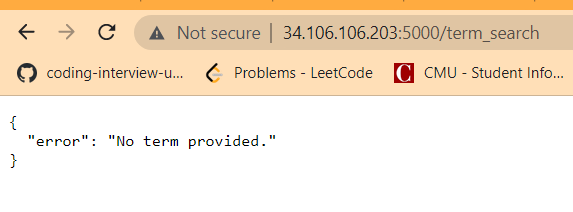
**2.1 Test manually generate json file:**

hadoop jar /usr/lib/hadoop/hadoop-streaming.jar -file mapper\_q.py -mapper 'python mapper\_q.py' -file reducer\_q.py -reducer 'python reducer\_q.py' -input /haichuan0304/Inverted\_Index\_Data/Data/ -output /haichuan0304/output\_inverted\_final6

hadoop fs -getmerge /haichuan0304/output\_inverted\_final6 ./flask/inverted\_index.json

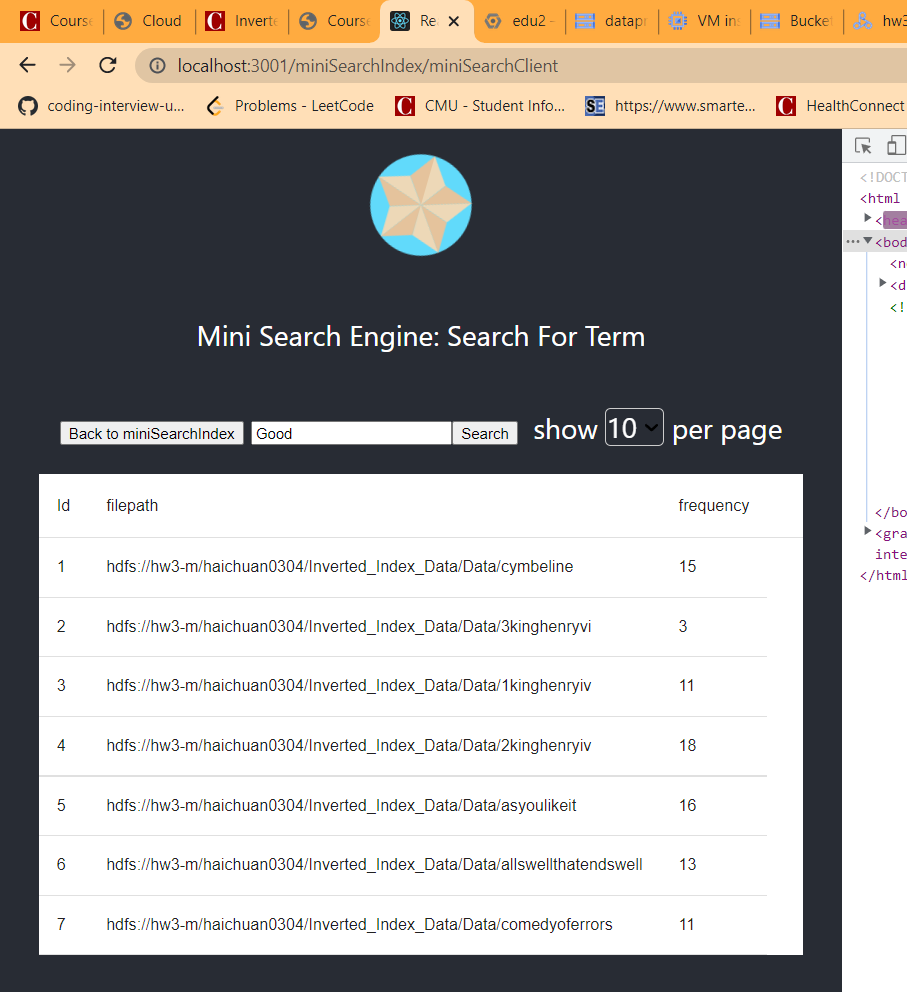


I also handle edge cases like’ empty params’:

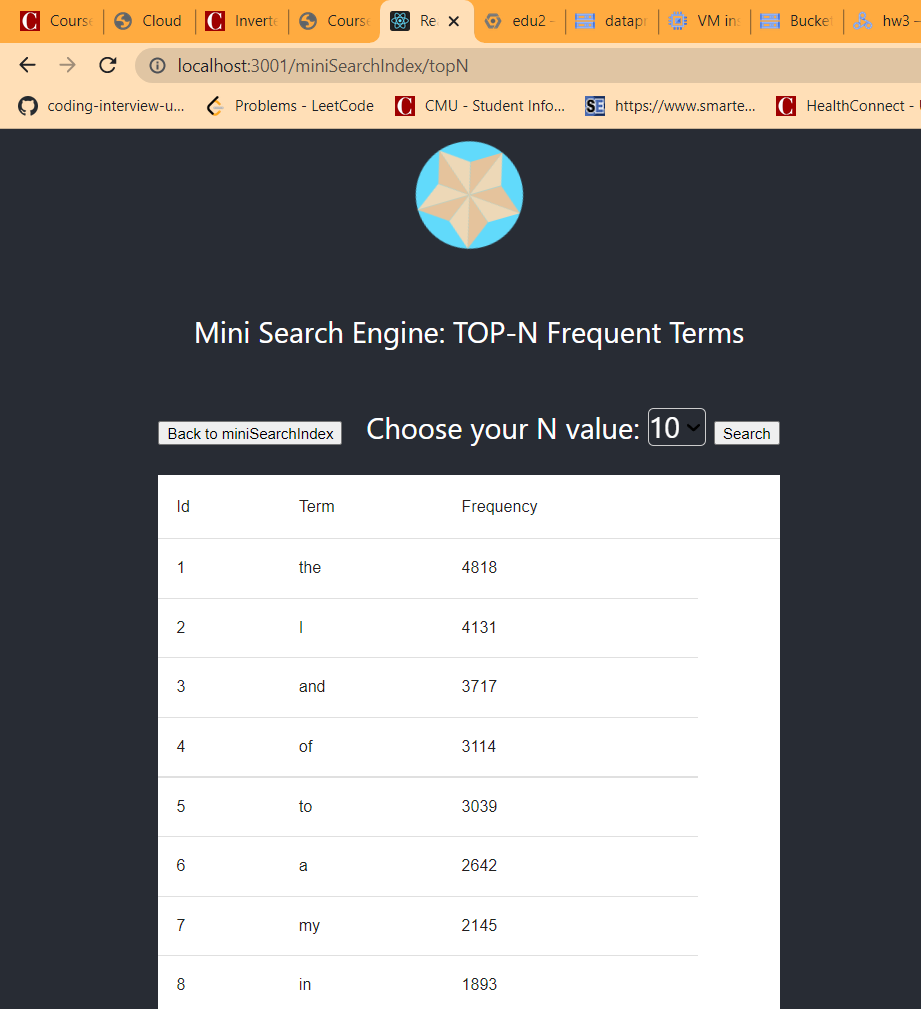


**2.2 Test react app:**

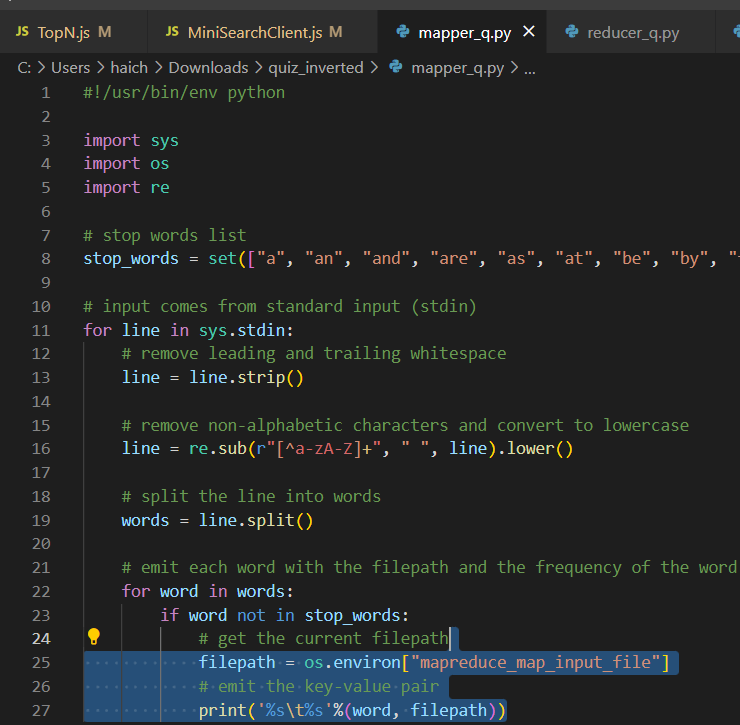
docker run -it --rm -v %cd%:/app -v /app/node\_modules -p 3001:3000 -e CHOKIDAR\_USEPOLLING=true tonyrays/dockerhub:projtimagepush2



Also test out top\_n\_search:



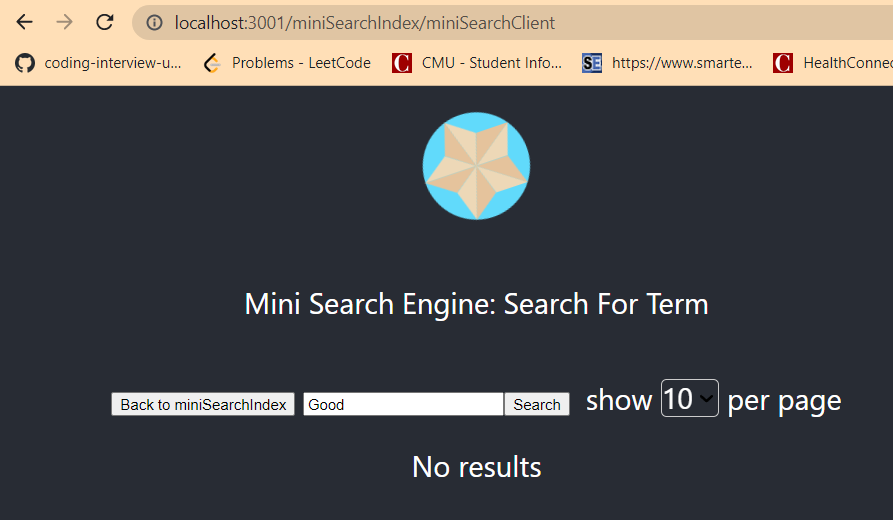
**2.3 Add stop word list**

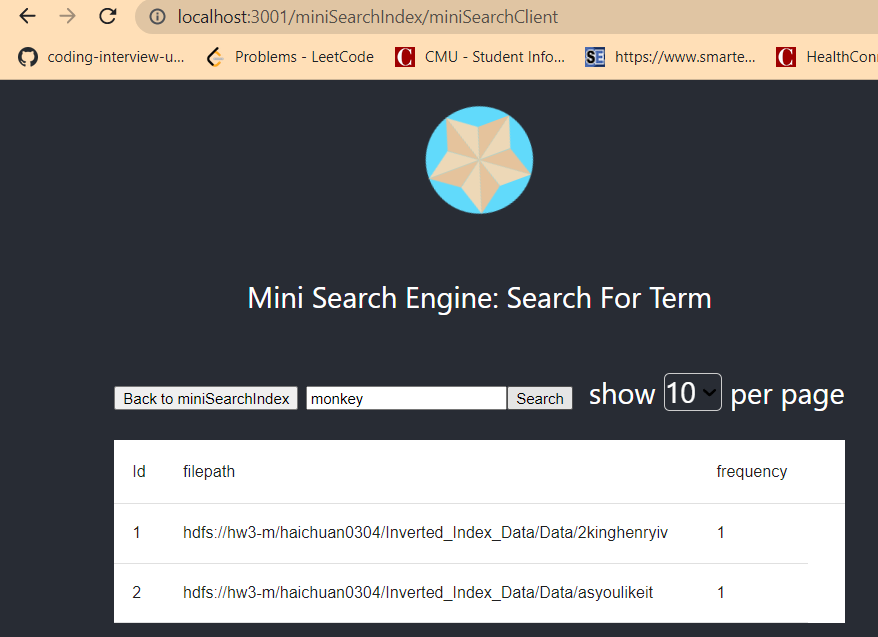


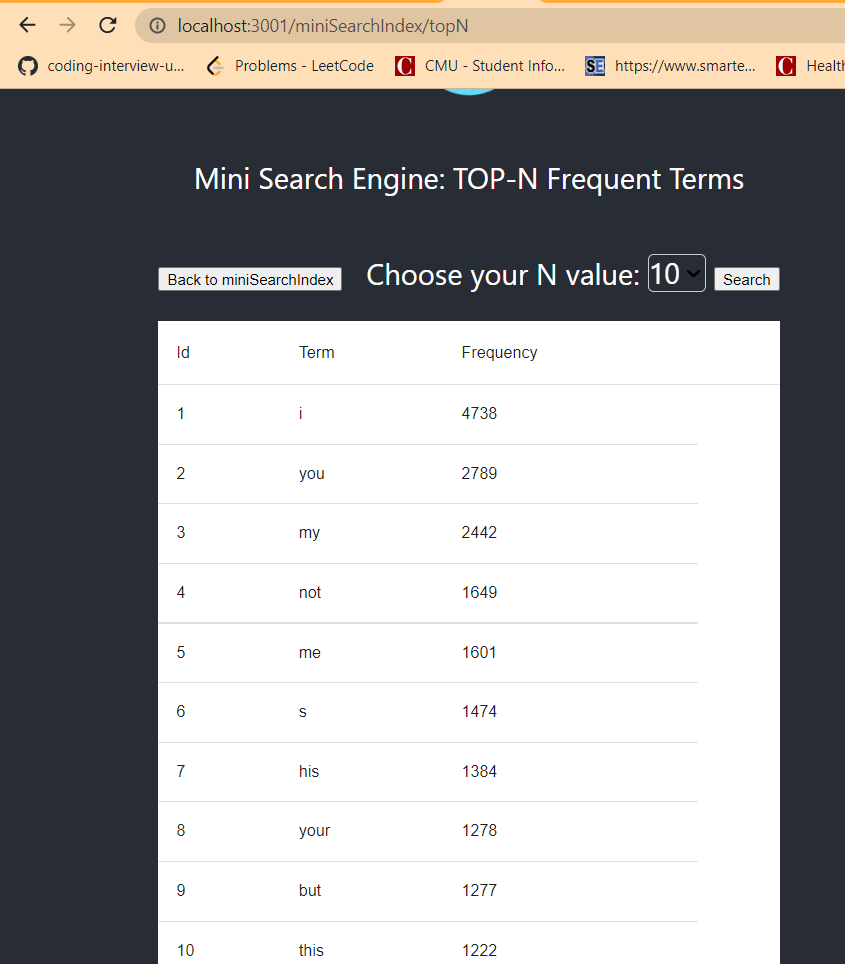
**2.4 Regenerate json file with stop words:**

hadoop jar /usr/lib/hadoop/hadoop-streaming.jar -file mapper\_q.py -mapper 'python mapper\_q.py' -file reducer\_q.py -reducer 'python reducer\_q.py' -input /haichuan0304/Inverted\_Index\_Data/Data/ -output /haichuan0304/output\_inverted\_final7

hadoop fs -getmerge /haichuan0304/output\_inverted\_final7 ./inverted\_index.json





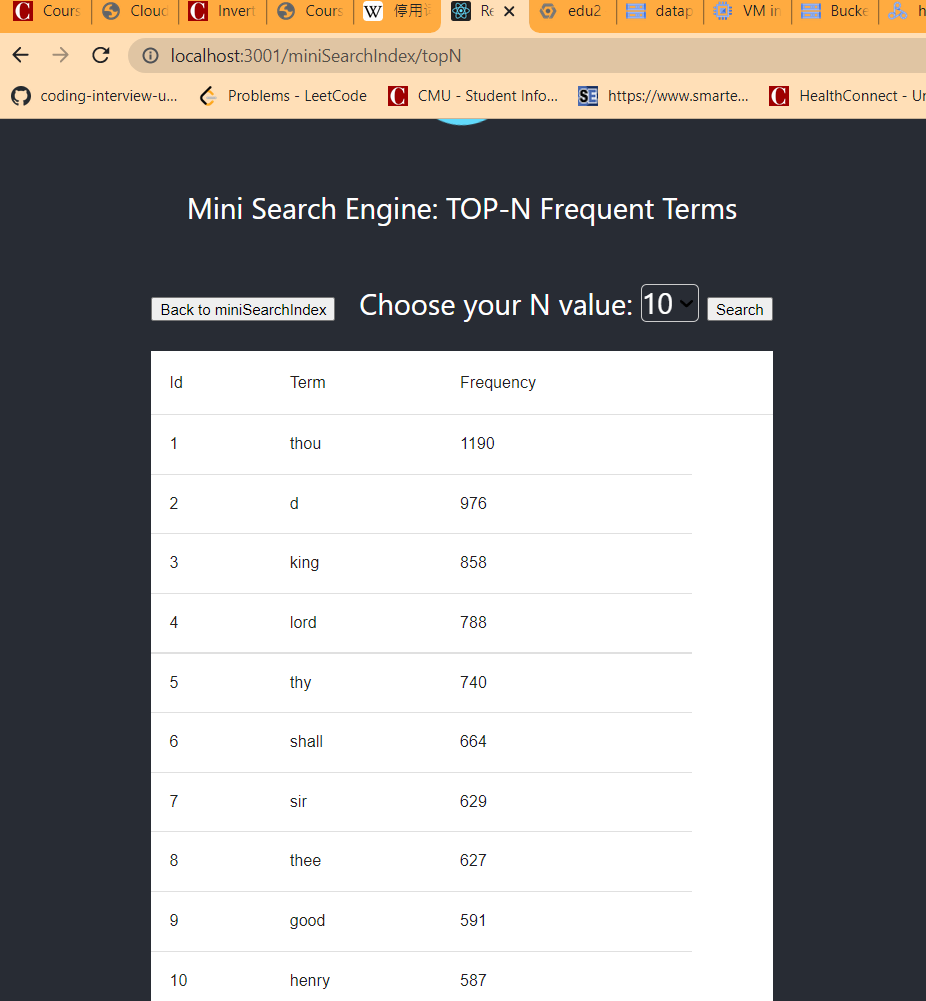


The top-N results are not satisfactory; therefore, it is necessary to include additional stop-words.

**2.5 Regenerate json file with even more stop words:**

Here I used stop words from library nltk.



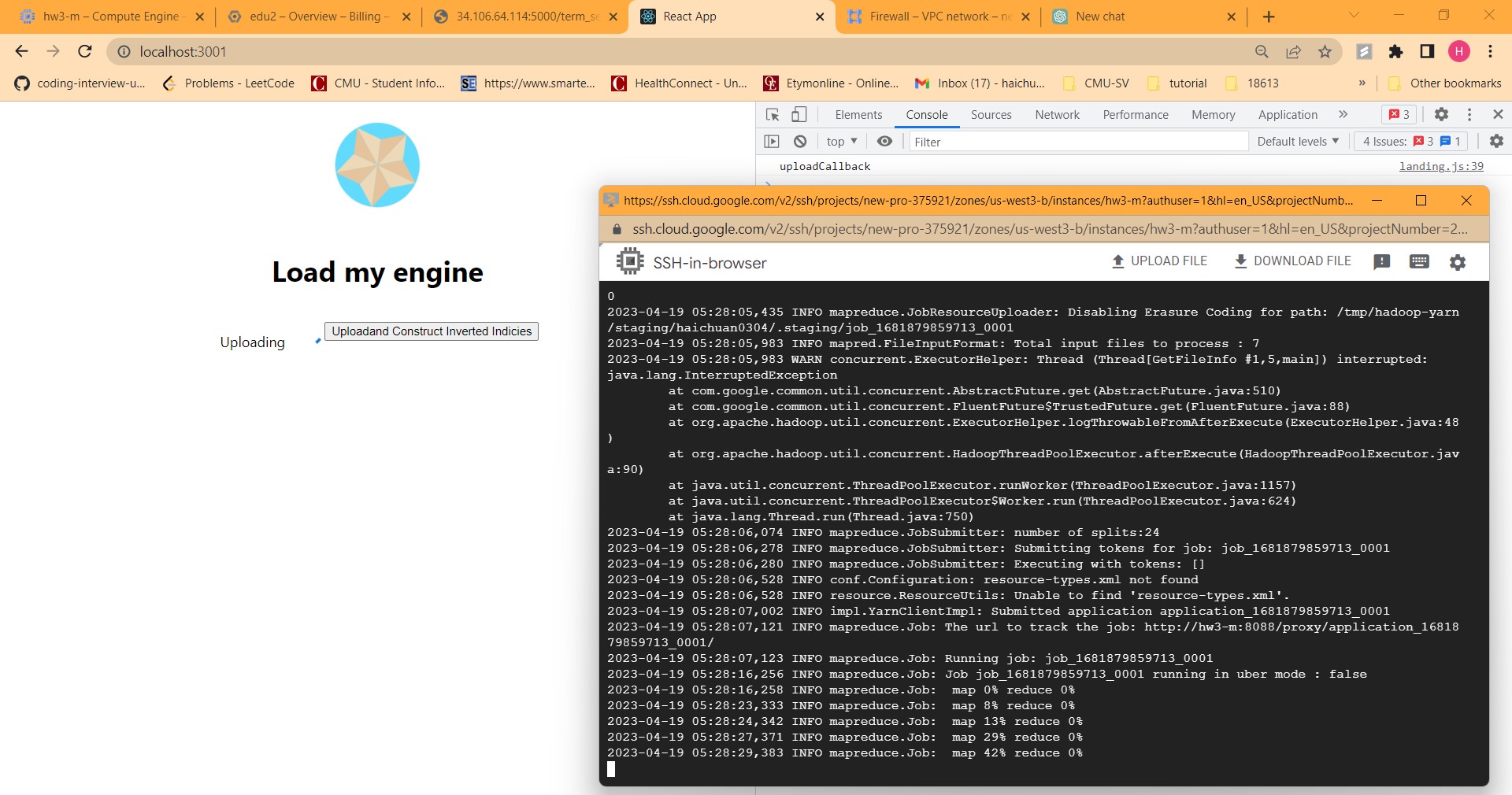


Looks much better now!

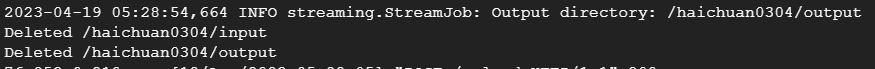
**2.6 Allow user to upload a zip file and automatically generate json file.**

To prevent auto reload, disable use\_reloader.

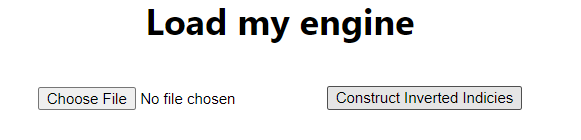
app.run(use\_reloader=False, debug=True, host='0.0.0.0')



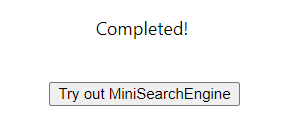
Delete ‘input’ and ‘output’ folder on HDFS to maintain a tidy environment.



Rename the button:

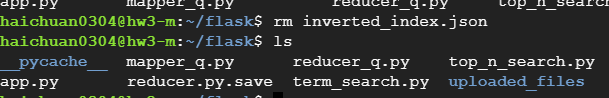


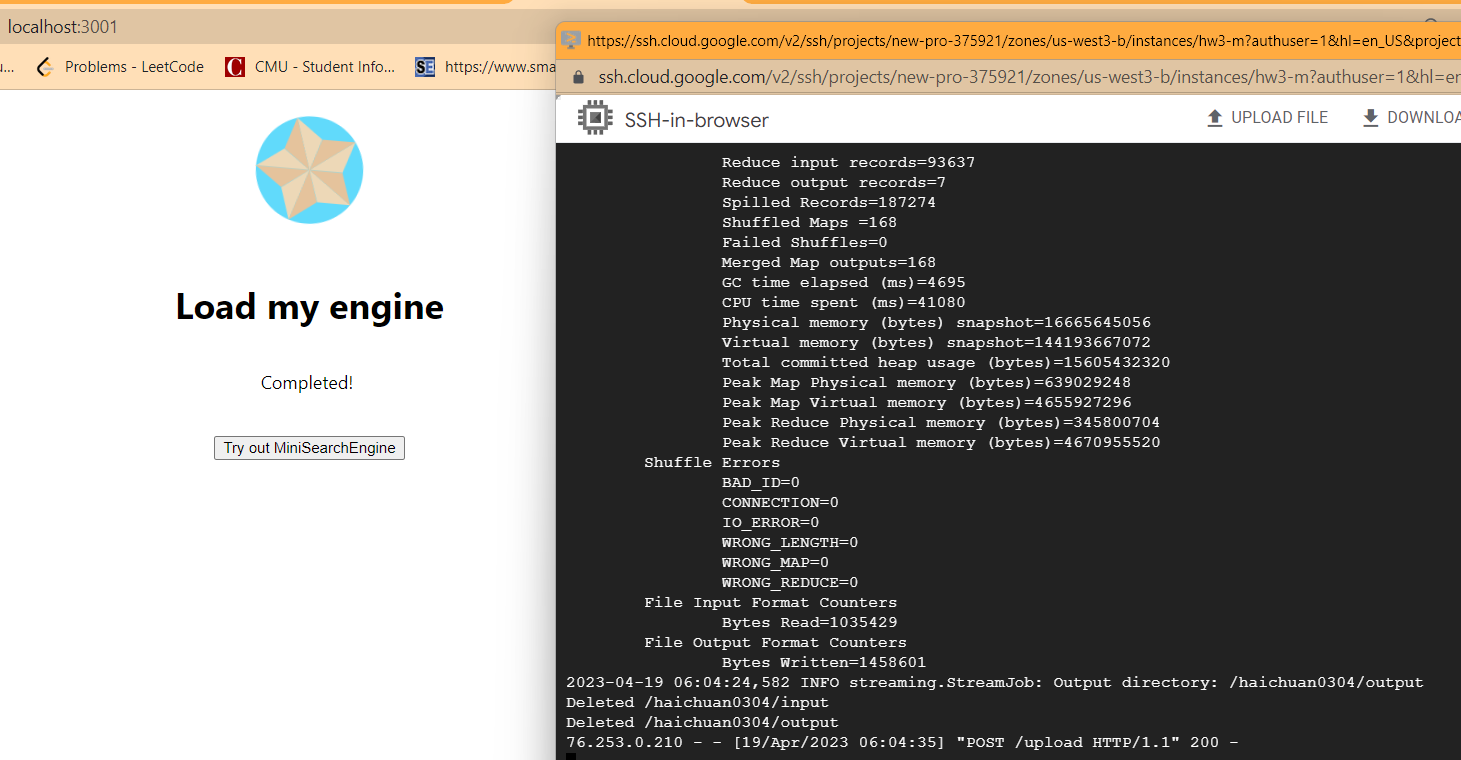




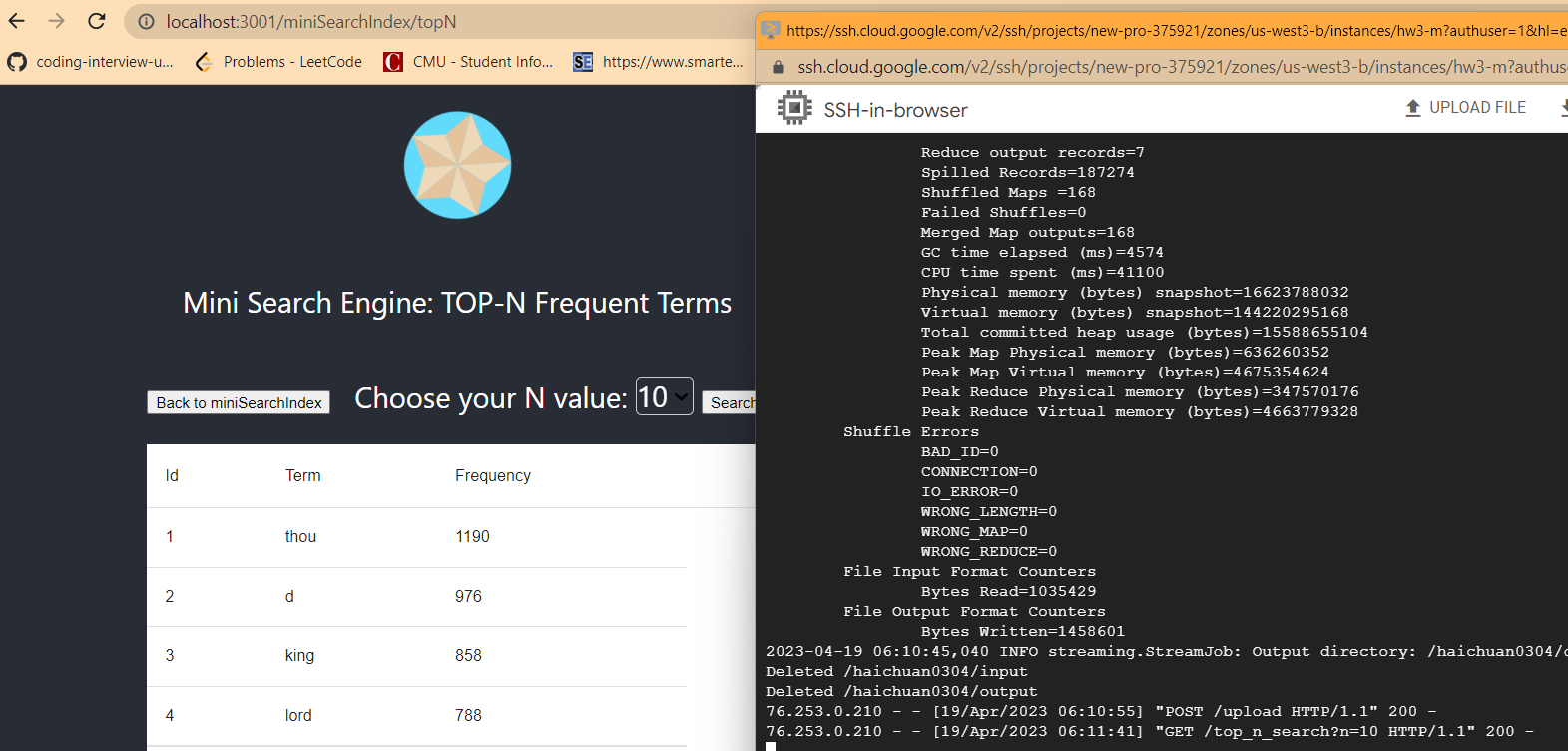
2.7 Double check:

Delete inverted\_index.json





Works well:



# Update:

I recently noticed that I forgot to include the execution time in the output, so I have updated my functions to return this information. Additionally, I have created a second video demonstration to showcase the newly execution time feature. Sorry for the inconvience.

e.g.

