Find the assignment here:
https://homes.luddy.indiana.edu/nehasupe/projects/sentence_length.html

1. **Detailed design, with pseudo-code and code-snippets if necessary.**

This assignment has 3 components- The UI, cloud functions and cache

I. UI: The UI is designed html, bootstrap and Javascript. The page contains a form which on submitting triggers the sent_word function using the trigger https://us-central1-cc-mapreduce.cloudfunctions.net/convert-sentence. A post request is preformed using JavaScript and the output image is displayed. A table contains some example URLs to try

II. Cache: The assignment says we could use a cloud key value store. However, the purpose of cache is to reduce the response time for a request. And using a cloud based cloud key value store adds to the time as it requires authentication and then transferring the data out of function. The cache is built using inbuilt python data structure. This allows for a quick access when a url is requested a second time. Drawback for this is that the cache will only exist as long as the cloud invocation is alive. The url and its corresponding image data in base64 format is stored in a key value format.

III. Cloud Functions: There are two functions designed- sent_word and create_histogram.

Sent_word: This function after receiving a request checks the url in the cache. If it exist the returns the image data. Else it uses nltk library to tokenize the text to sentences. I decided to use this library as it takes care of different kinds of sentences which not just end with '.' and special cases like 'Dr.', 'Mr.', etc. The I count the number of words in sentence and turn the length of the sentence and number of times the length occurs into a dictionary and send to a request to trigger the function which create the histogram graph. The value returned from histogram function is wrapped around html tags are returned as response.

Create_histogram: This function gets the histogram in key value pair and its converted to graph using matplotlib library and this graph is encoded to base64 and this is returned as a string back to sent_word.

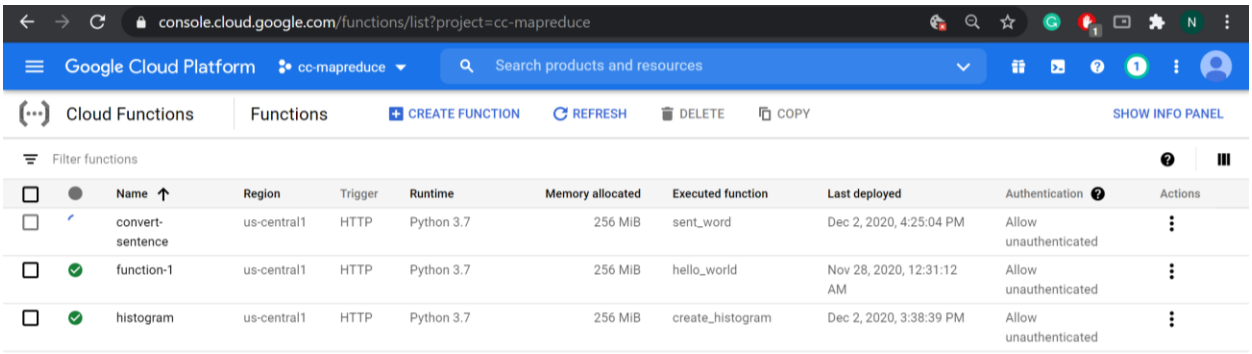
2. **Which cloud APIs you used**

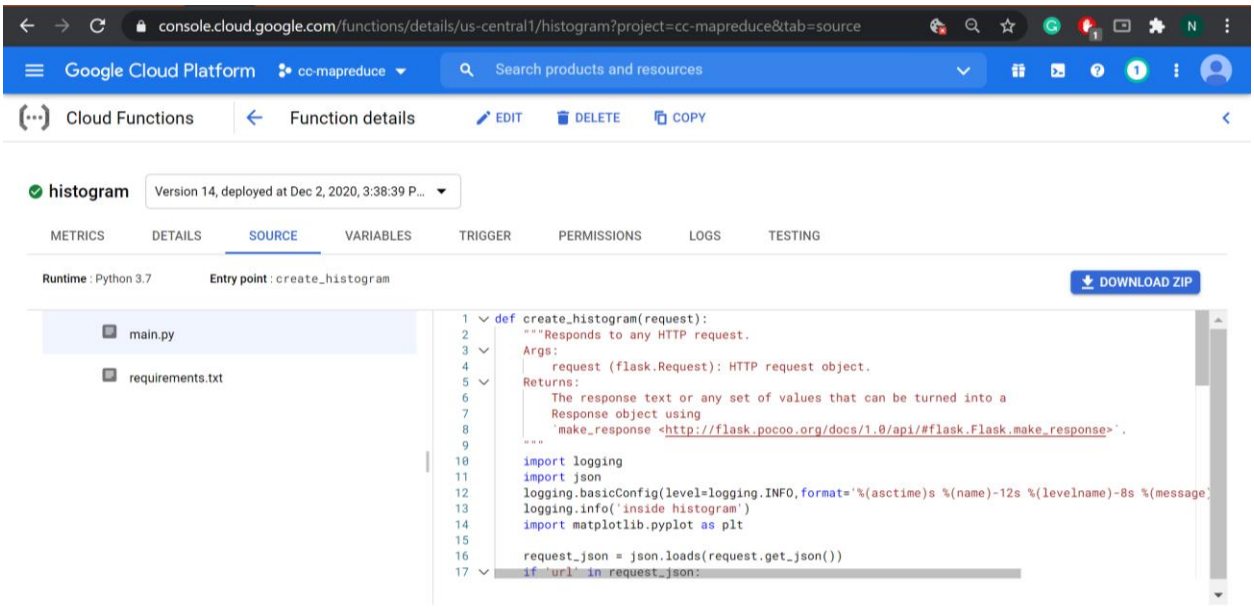
Cloud Functions for functions with http triggers

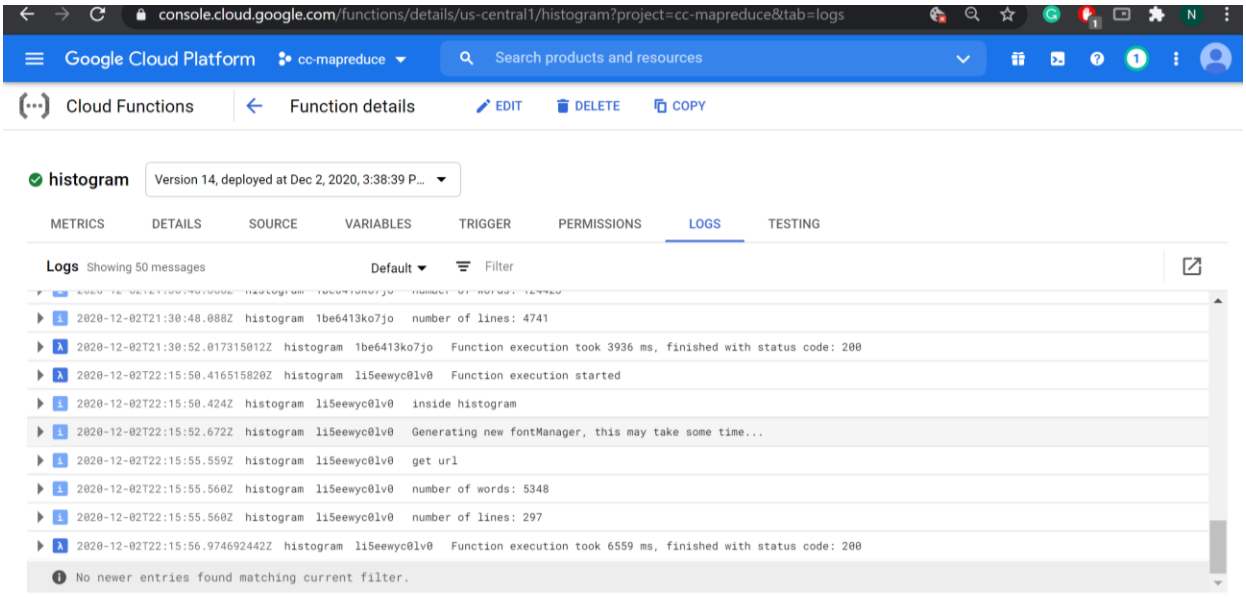
3. **Relevant Gcloud logs**

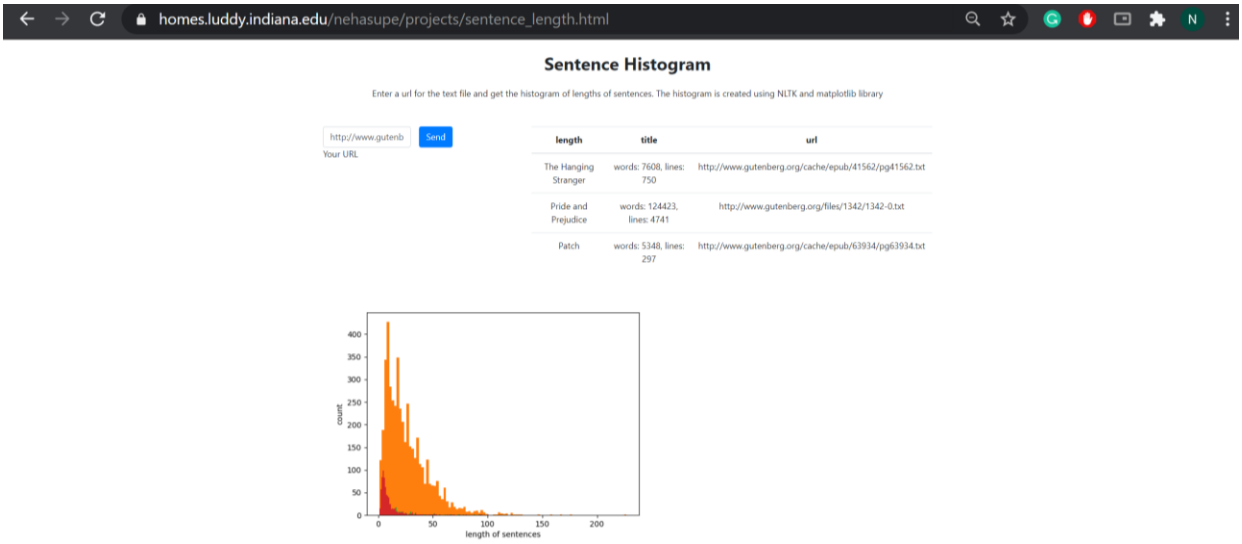
Cloud logs can be found under /logs. downloaded-logs-20201202-175733.csv cntains logs from sent_word function. downloaded-logs-20201202-175304.csv contains logs from create_histogram function.

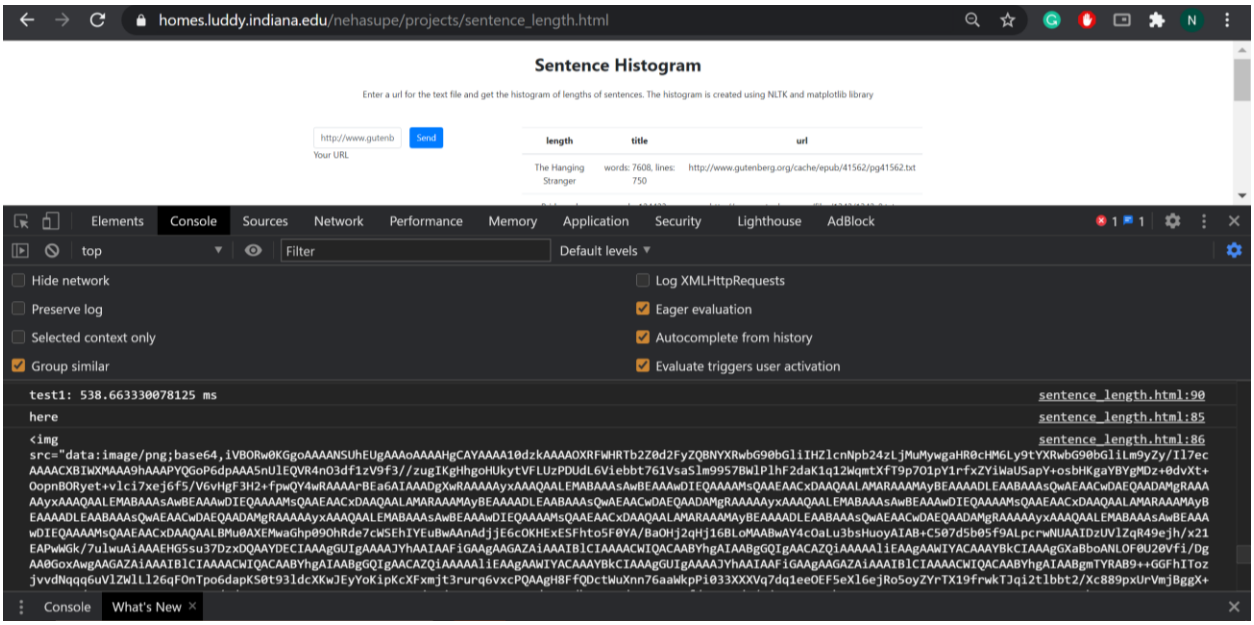
4. **Screenshots!**


console.cloud.google.com/functions/list?project=cc-mapreduce
Google Cloud Platform
cc-mapreduce
Search products and resources
Cloud Functions
Functions
CREATE FUNCTION
REFRESH
DELETE
COPY
SHOW INFO PANEL
Filter functions
Name
Region
Trigger
Runtime
Memory allocated
Executed function
Last deployed
Authentication
Actions
convert-sentence
us-central1
HTTP
Python 3.7
256 MiB
sent_word
Dec 2, 2020, 4:25:04 PM
Allow unauthenticated
function-1
us-central1
HTTP
Python 3.7
256 MiB
hello_world
Nov 28, 2020, 12:31:12 AM
Allow unauthenticated
histogram
us-central1
HTTP
Python 3.7
256 MiB
create_histogram
Dec 2, 2020, 3:38:39 PM
Allow unauthenticated


console.cloud.google.com/functions/details/us-central1/histogram?project=cc-mapreduce&tab=source
Google Cloud Platform
cc-mapreduce
Search products and resources
Cloud Functions
Function details
EDIT
DELETE
COPY
histogram
Version 14, deployed at Dec 2, 2020, 3:38:39 P...
METRICS
DETAILS
SOURCE
VARIABLES
TRIGGER
PERMISSIONS
LOGS
TESTING
Runtime : Python 3.7
Entry point : create_histogram
DOWNLOAD ZIP
main.py
requirements.txt
def create_histogram(request):
"""Responds to any HTTP request.
Args:
request (flask.Request): HTTP request object.
Returns:
The response text or any set of values that can be turned into a
Response object using
`make_response <http://flask.pocoo.org/docs/1.0/api/#flask.Flask.make_response>`.
"""
import logging
import json
logging.basicConfig(level=logging.INFO,format='%(asctime)s %(name)-12s %(levelname)-8s %(message)
logging.info('inside histogram')
import matplotlib.pyplot as plt
request_json = json.loads(request.get_json())
if 'url' in request_json:


console.cloud.google.com/functions/details/us-central1/histogram?project=cc-mapreduce&tab=logs
Google Cloud Platform
cc-mapreduce
Search products and resources
Cloud Functions
Function details
EDIT
DELETE
COPY
histogram
Version 14, deployed at Dec 2, 2020, 3:38:39 P...
METRICS
DETAILS
SOURCE
VARIABLES
TRIGGER
PERMISSIONS
LOGS
TESTING
Logs Showing 50 messages
Default
Filter
2020-12-02T21:30:48.088Z histogram 1be6413ko7jo number of lines: 4741
2020-12-02T21:30:52.017315012Z histogram 1be6413ko7jo Function execution took 3936 ms, finished with status code: 200
2020-12-02T22:15:50.416515820Z histogram li5eewyc0lv0 Function execution started
2020-12-02T22:15:50.424Z histogram li5eewyc0lv0 inside histogram
2020-12-02T22:15:52.672Z histogram li5eewyc0lv0 Generating new fontManager, this may take some time...
2020-12-02T22:15:55.559Z histogram li5eewyc0lv0 get url
2020-12-02T22:15:55.560Z histogram li5eewyc0lv0 number of words: 5348
2020-12-02T22:15:55.560Z histogram li5eewyc0lv0 number of lines: 297
2020-12-02T22:15:56.974692442Z histogram li5eewyc0lv0 Function execution took 6559 ms, finished with status code: 200
No newer entries found matching current filter.


homes.luddy.indiana.edu/nehasupe/projects/sentence_length.html
Sentence Histogram
Enter a url for the text file and get the histogram of lengths of sentences. The histogram is created using NLTK and matplotlib library
http://www.gutenb
Send
Your URL
length
title
url
The Hanging Stranger
words: 7608, lines: 750
http://www.gutenberg.org/cache/epub/41562/pg41562.txt
Pride and Prejudice
words: 124423, lines: 4741
http://www.gutenberg.org/files/1342/1342-0.txt
Patch
words: 5348, lines: 297
http://www.gutenberg.org/cache/epub/63934/pg63934.txt
count
length of sentences



5. **Cost of running all your experiments.**

Since I was not making use of any other database like cloud services for caching, I spent $0.9 on running all experiments

6. **Weaknesses and how your design and implementation could be improved**

The function doesn't handle files with too many numbers in it and timesout. (For eg. https://www.gutenberg.org/files/127/127.txt) I have not handled timeout errors. The implementation can be improved using a cloud key value store at the cost of increased time for response. Create more fault tolerant functions and handle errors.

Some performance numbers: end-to-end times for a few different books

Title	URL	lengths	Function execution time	Using cache	Total end to end time	Total end to end time using cache
Pride and prejudice	http://www.gutenberg.org/files/1342/1342-0.txt	words: 124423, lines: 4741	19697 ms	6 ms	22081.631103515625 ms	3102.713134765625 ms
Patch	http://www.gutenberg.org/cache/epub/63934/pg63934.txt	words: 5348, lines: 297	5331 ms	5 ms	7372.10302734375 ms	1909.34912109375 ms

The Hanging Stranger	http://www.gutenberg.org/cache/epub/41562/pg41562.txt	words: 7608, lines: 750	10838 ms	7 ms	13140.330078125 ms	538.663330078125 ms