Assignment 5: MapReduce with Cloud Functions

Problem Statement

Design and implement parallel Map-Reduce system using Google Cloud Functions

- Fetch data from Project Gutenberg which contains large number of text files
- Compute inverted index of text
- Expose a simple web-based search interface

Understanding Problems

- We need to take all files present at a Google bucket location and perform parallel Map-Reduce to assign inverted index to the word
    - Intermediate Problem-
        - After every map task, we need to generate and save index file for reduce task
        - Need to put barrier synchronization for running reduce task in parallel

Design Details

For this project, we will have to create multiple Google Cloud functions. First step will be to enable the Cloud Functions API. Below are the design details:

- Shell file *copy_files_to_bucket.sh* will run the python code *download_files.py* which will
    - Download few files from Project Gutenberg & Decode the text corpus line by line
    - It will remove the punctuation and convert all the words into lowercase characters
    - Copy these files from local to google bucket.

- Shell file *mapreduce-start.sh* will upload main.py and requirement.txt and it will
    - Upload the files from google bucket to the variables & divide the files in two
    - Two *map_function* are triggered in parallel for two sets of files using HTTP
    - Barrier Synchronization step which will wait for all the mapper task to get completed
        - This is to ensure that reduce functions gets triggered once all map functions are completed successfully.
    - *reduce_function* are triggered in parallel which will take files as input based on starts with character. They will
    - The UI function will be triggered which will take a word as input and it will return the index value of the word, i.e., descending order of the number of times words appearing in the document

- Shell file *map_function.sh* will upload main.py and requirement.txt and it will
    - Upload the files from google bucket to the variable
    - It will map every word appearing in the text corpus with number of times it is appearing and the document name
    - These files are divided in two based on starts with character
    - The indexed outputs are uploaded to google bucket

- Shell file *reduce_function.sh* will upload main.py and requirement.txt and it will
    - Combine the common words and unique words across files
    - Common word will be inverted indexed in descending order based on the number of times words appearing in a document
    - These inverted indexed files are uploaded to the google bucket

- Shell file *ui-function.sh* will read the inverted indexed files and search for the keyword which will be given as an input argument

- Shell file *Clean_buckets.sh* will remove the files from the bucket

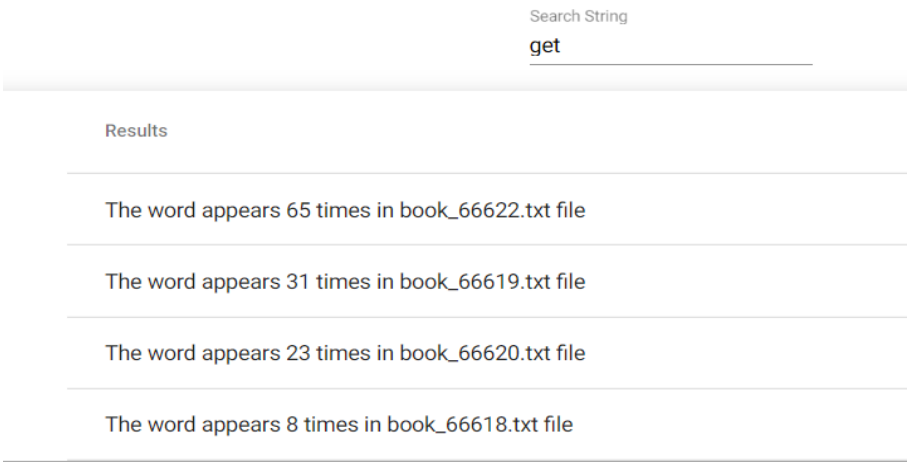
User Interface

I have taken below steps for creating a simple user interface for keyword search:

- Created oAuth 2.0 code(ClientID, Client Secret and other scope variables) and passed these in to access token of Postman to setup connection with Cloud function, and send Http Post request to call the cloud function from Postman, Sent a post request in Raw format (as a Json) to see if it's working

- Next created an AngularJS application for implementing the user interface
    - Used Material UI to fill the search string and display the result in tabular form
    - For calling the API function, created one Post api service which I created and tested on postman
    - Passed Authorization code as header and the search string will be taken and sent as a Json
    - Once the function sends an output, these will be displayed in a tabular form using Material UI

Setup a gitbhub repository of angular application and connected this repository and deployed it on *Heroku* , the link for this is "https://map-reduce-fe.herokuapp.com/"**.** Please find the screenshot of bonus part here

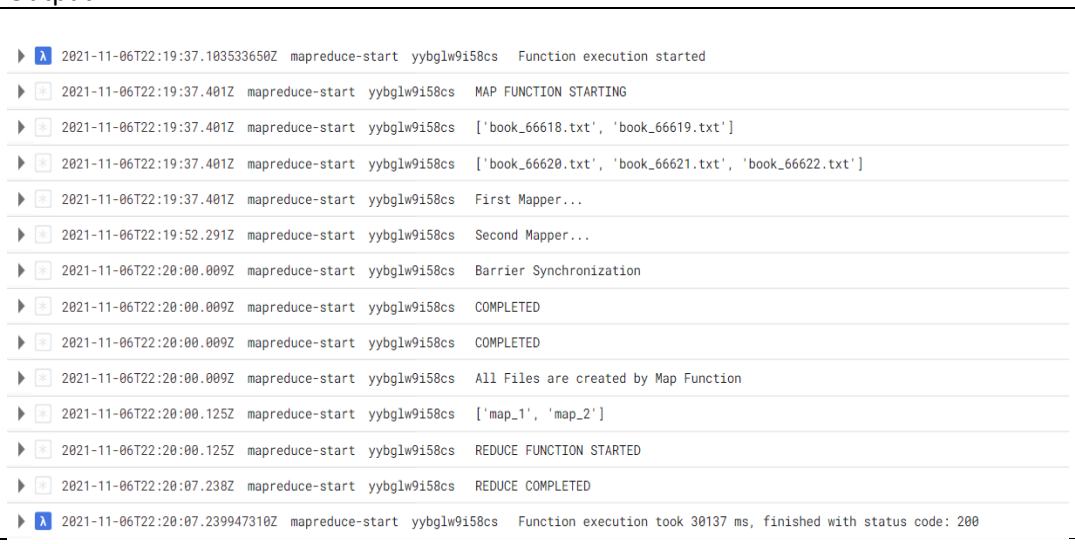
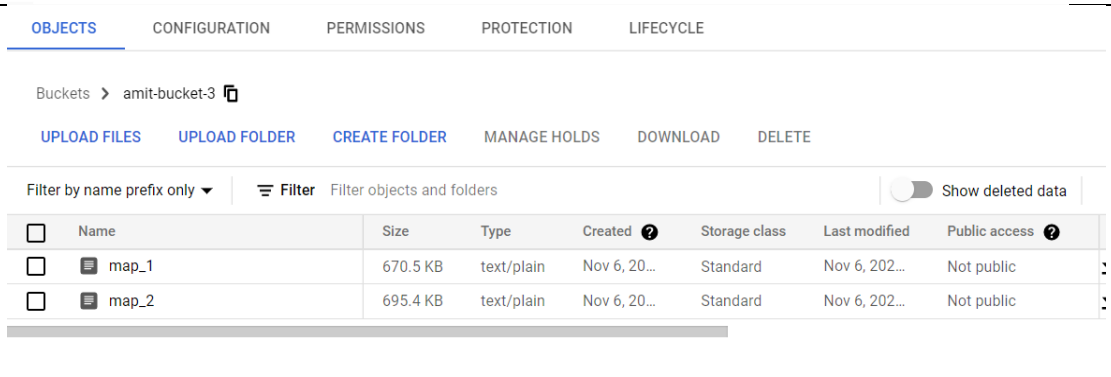


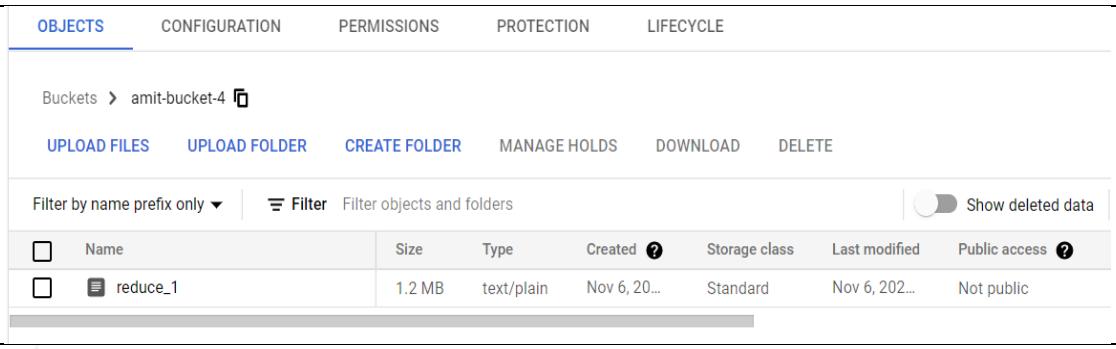
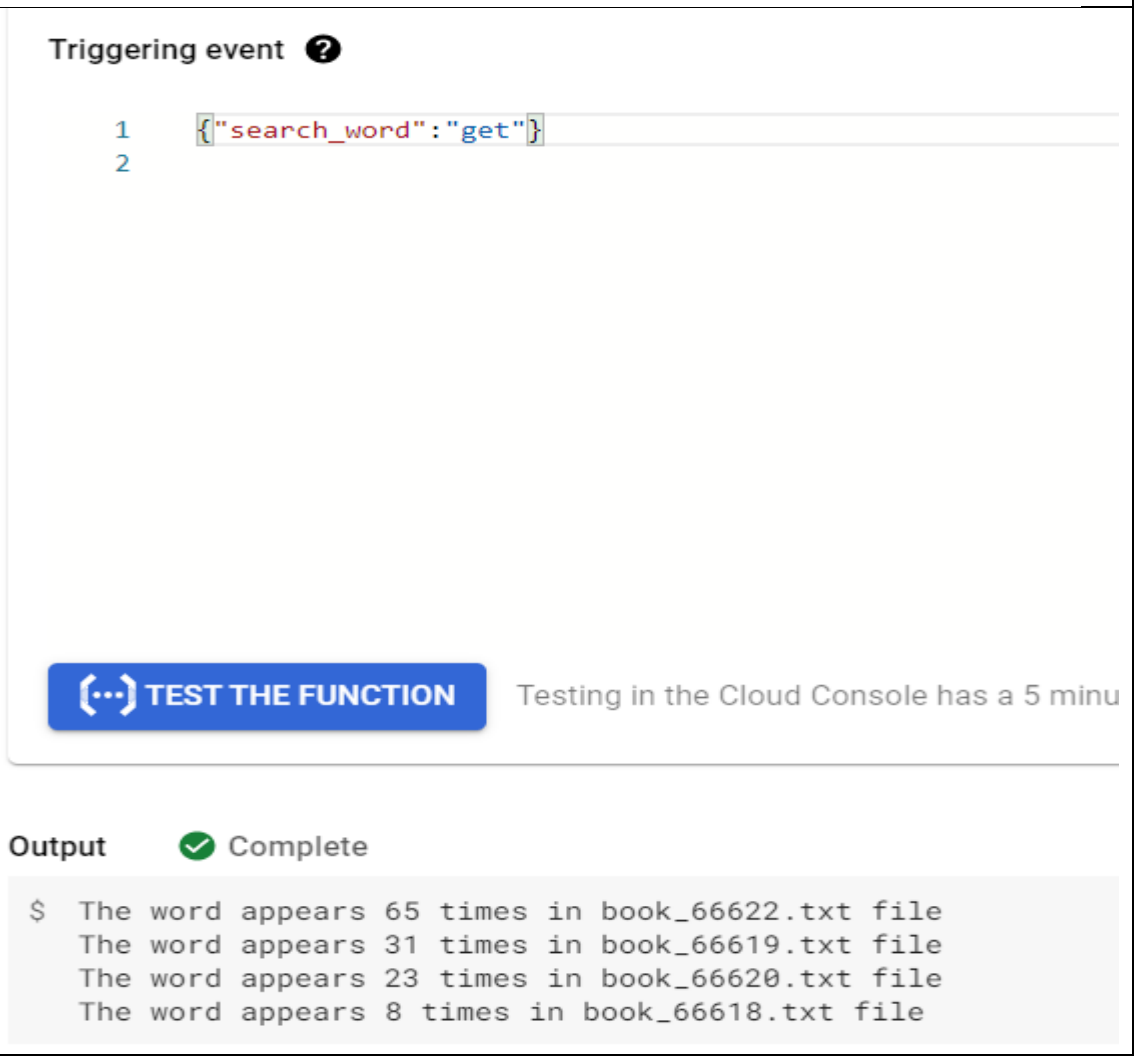
Test Cases

S No	Test Case	Script
1	Should be able to download and copy files to Google Bucket	copy_files_to_bucket.sh
2	It should trigger Map Function in parallel	Run mapreduce-start
3	It should Barrier Synchronize the process	Run mapreduce-start
4	It should trigger Reduce Function	Run mapreduce-start
5	The inverted indexed output file should be created at Google Bucket	Check amit-bucket-4 bucket

Examples

Let's evaluate the working of this code by running the functions setting up the connection and sending following requests and see how the server responds:

Input	Output
Run mapreduce-start function	
Check if Mapper Intermediate files are getting written on Google Bucket	

Check if Reducer output files are getting written on Google Bucket	
Result of search string 'get'	

Solution Evaluation and Review

- The files are getting downloaded from Project Gutenberg website and cleaned as expected
- Files are getting copied from local to mentioned google bucket
- Main function is triggering Map and Reduce functions in parallel successfully
- Map functions are indexing as expected
- The code is barrier synchronizing the trigger for reduce process

- Reduce task is combining the words as expected
    - It is sorting the name of documents in descending according to number of times it is appearing
- Inverted indexed file is getting saved successfully
- UI is working as expected

References

- Python function. (https://cloud.google.com/functions/docs/first-python/)