**CSE 587 : DATA INTENSIVE COMPUTING**

**LAB – 2 REPORT**

**SUBMITTED BY:**

**DITHYA SRIDHARAN - 50286923**

**HITESH SANTWANI - 50291123**

# ABSTRACT

In this lab, we will be analyzing Data scrapped from the Internet from sources like New York Times, Twitter and Common Crawl using Word Count Map-Reduce Algorithm and enhance them by adding big data analytics and visualization skills.

We picked the topic “Countries in Europe” to begin our analysis. We have considered 5 subtopics: “France”,” UK”,” Germany”,” Italy”,” Ireland”. We take these as keywords as premise of our analysis.

The Word Count Map Reduce Algorithm was run on data scrapped from Twitter, Common Crawl and NY Times. Using the NY Times API, about 500 articles were collected and about 20k tweets were gathered using the Twitter API and 500 articles from Common Crawl were gathered. This data was scrapped to get text data from the web-pages and tweets. The data was fed into the Map Reduce Engine and the word-counts were collected. The Mapper employs stop words and regular expression filtering. The top words from the word-count is displayed using Tableau word cloud. Building on this, Another Map Reduce Engine was built that analyses word co-occurrence.

# Link to Demo

[Link](https://drive.google.com/open?id=1nukTHkDrYWD6-Zx-FgJg9EER7pu5HbmQ)

# METHODOLOGY AND APPROACH

## 1) Data Collection:

1) NY Times: API Key was created for NY Times. Using the Collect API, articles were collected.

2) Twitter: Created twitter API Key and obtained tweets using tweepy API.

3) Common Crawl: We referred the code from [link](https://github.com/fhamborg/news-please/blob/master/newsplease/examples/commoncrawl.py) and then we modified the code to make it work for out filter criteria and to extract WARC files from 4/1/19 then we wrote the custom script (Transformations) to combine the articles only from English news websites in to the subtopics.

4) These URLs collected from the NY Times were scrapped and the contents were stored to a file. The tweet content is also stored to the file. Similarly, common crawl content is also stored to a file.

## 2) Word count Mapper-Reducer:

**Mapper:**

1. Read input file.

2. Remove unwanted symbols and punctuation

3. Remove stop words.

4. For all words in doc

Emit (word, 1)

**Reducer:**

1. Read input from Mapper.

2. Group the data by key.

3. For each key in data, put key and its count in dictionary.

4. Sorted the dictionary based on count descending.

5. For each key in dictionary

Emit (key, count)

## 3)Word count Visualization:

Used Tableau to display the top 10 and 50 most frequent words obtained by the WordCount Map Reduce Algorithm.

## 4)Word co-occurrence Mapper-Reducer:

**Mapper:**

1. Read input file.

2. Remove unwanted symbols and punctuation

3. Remove stop words.

4. For all words w1 in doc

For all words w2 in Neighbours(w1)

Emit (w1-w2, 1)

**Reducer:**

1. Read input from Mapper.

2. Group the data by key.

3. For each key in data, put key and its count in dictionary.

4. Sorted the dictionary based on count descending.

5. For each key in dictionary

Emit (key, count)

## 5)Word co-occurrence Visualization:

The data obtained from the reducer is displayed using Tableau word clouds.

# DATA FLOW

**Data Collection:**

**Scrape the data from NYTimes, Twitter API and Common Crawl**

**Tableau Visualization for Top 10 and Top 50 of word count pairs**

**Reducer :**

**Emits <word, total-count> pairs.**

**Mapper :**

**Data is cleaned and emits <word,count> pairs. Use of regex and stop words to clean the input.**

# RESULTS:

At first, data from a single day for Twitter, NY Times and Common Crawl was fed into map reduce.

For France,

In the **NY Times** article, commonly occurring words included,

Said(since it is reporting speech), President, Macron, Notre dame, Government, Fire, National, Cathedral, United

In the **Twitter data**, commonly occurring words included,

Notre dame, Fire, Cathedral, Dead, United, World, Statue, Church, Mozambique, Government, Minister

In the **Common Crawl** data, commonly occurring words included,

Said, Macron, Government, World, President, United, EU, May, People, Minister, Security,

Foreign, Meeting

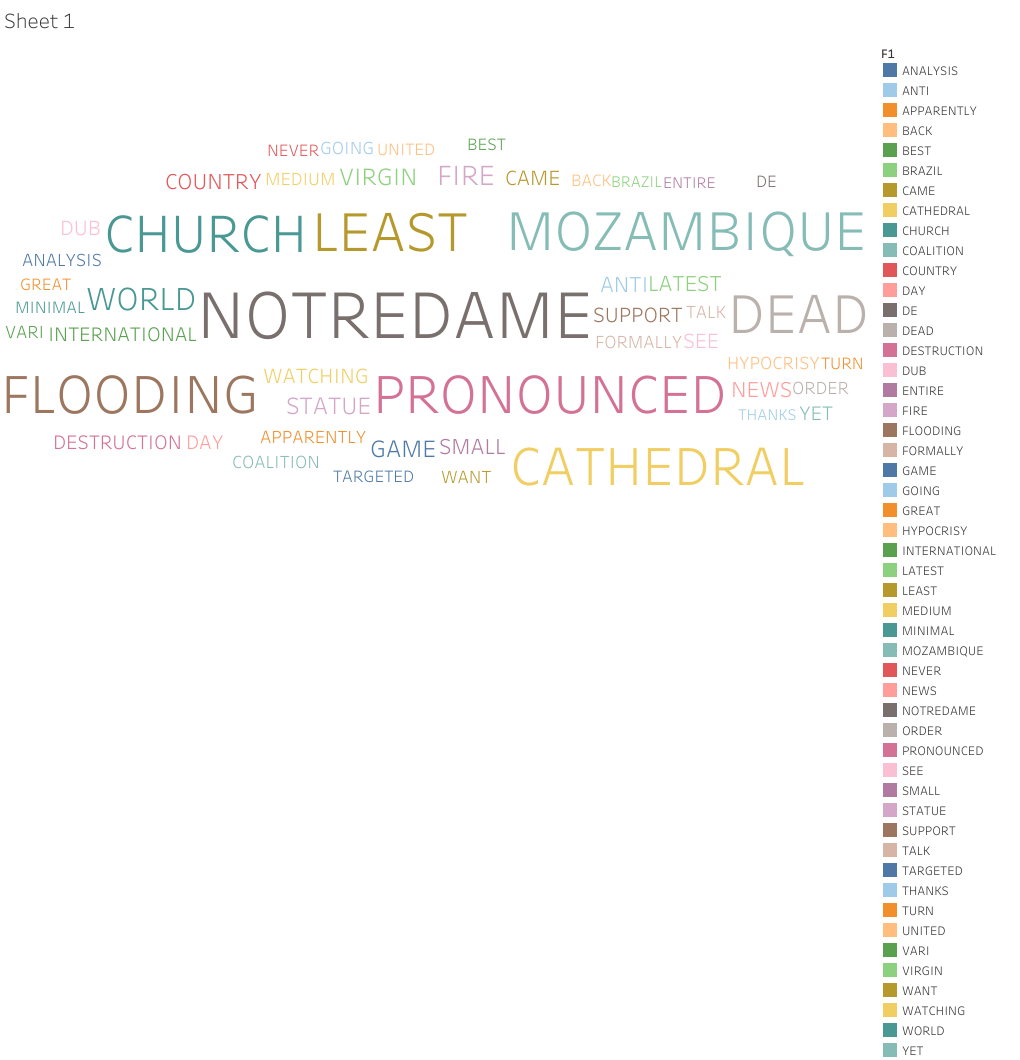
There were commonly occurring words between the three sets of data.

When the MapReduce was performed for the bigger dataset, collected over a week and then over the entire data(20k tweets, 500 NYT articles, 500 common crawl articles),

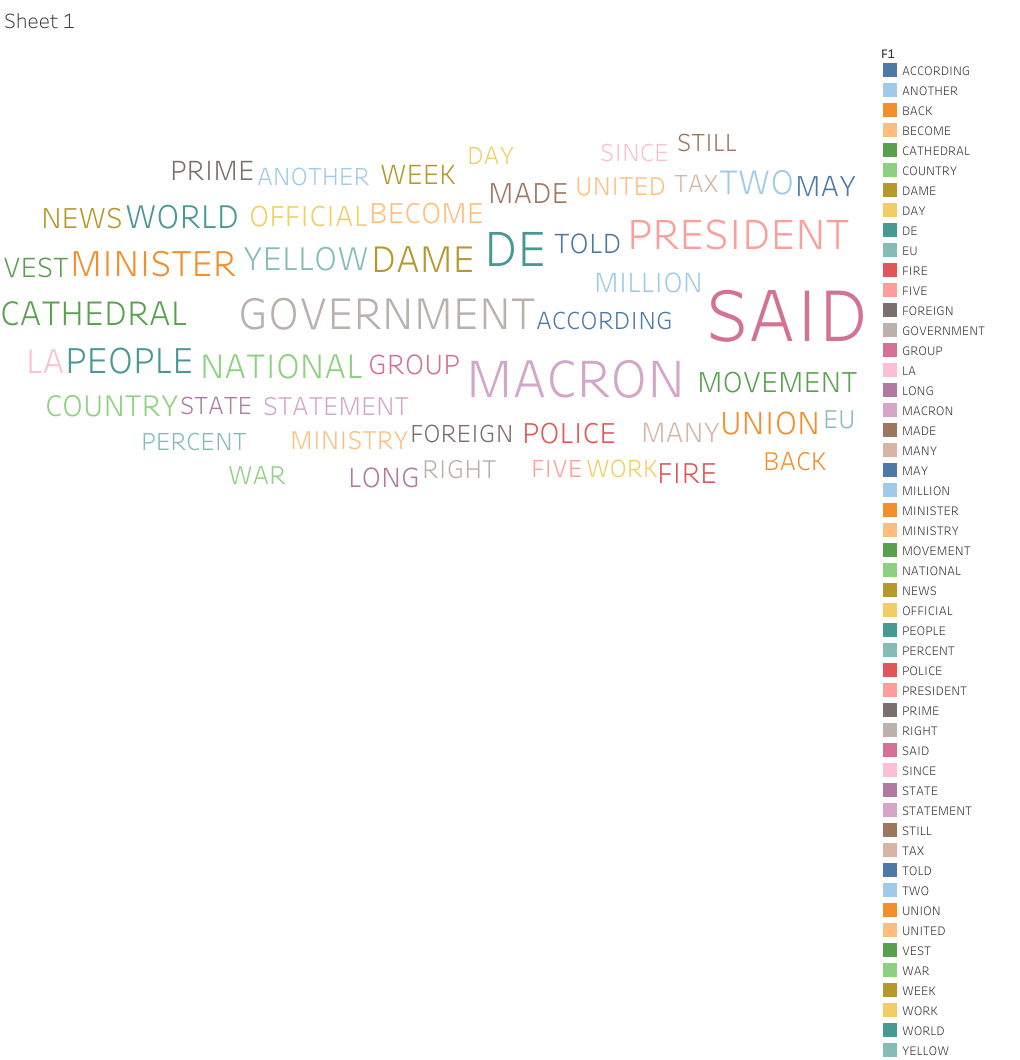
The commonly occurring words among the NY articles, Twitter data converged to give many commonly occurring words.

Similar results were found for the other subtopics of “**Countries of Europe**” – **Italy, Germany, UK, Ireland**. And the similar results were found for the keyword “**Europe**”

# PROTOTYPE WORD CLOUD FOR TWITTER WORD COUNT:



## PROTOTYPE WORD CLOUD FOR NYT WORD COUNT:

****

**PROTOTYPE WORD CLOUD FOR COMMON CRAWL COUNT:**

****

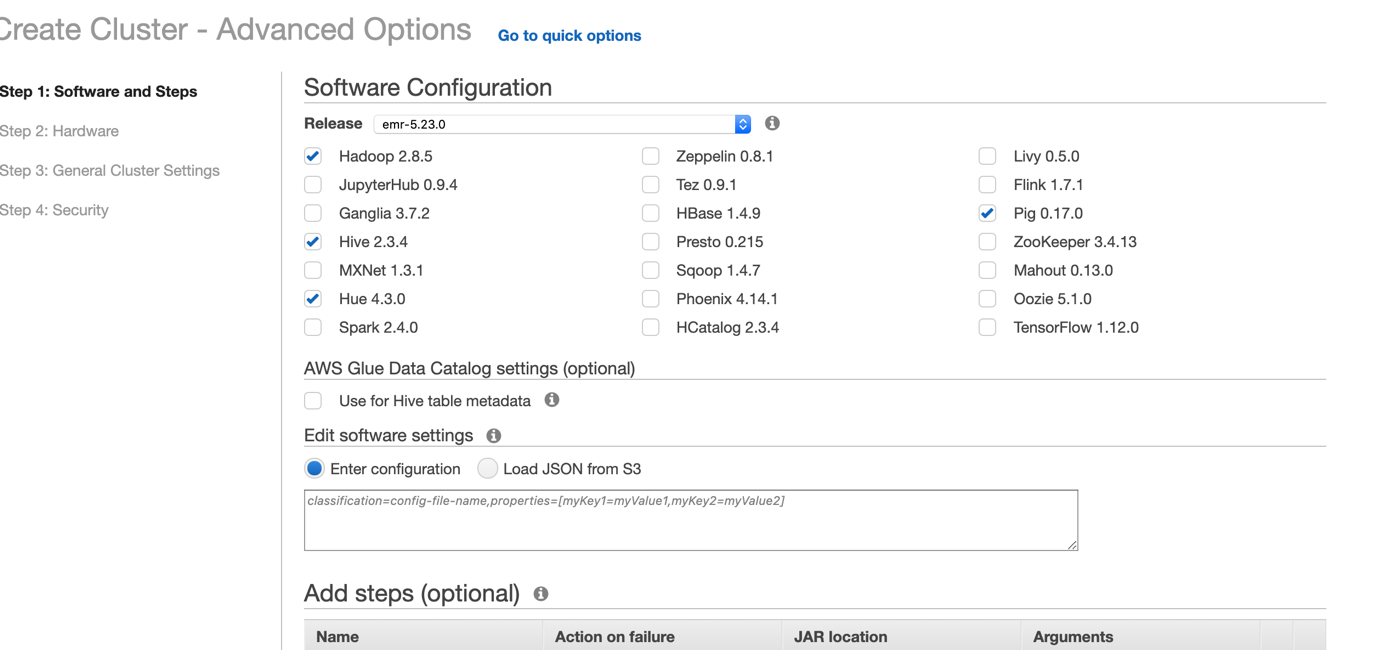
# AWS EMR

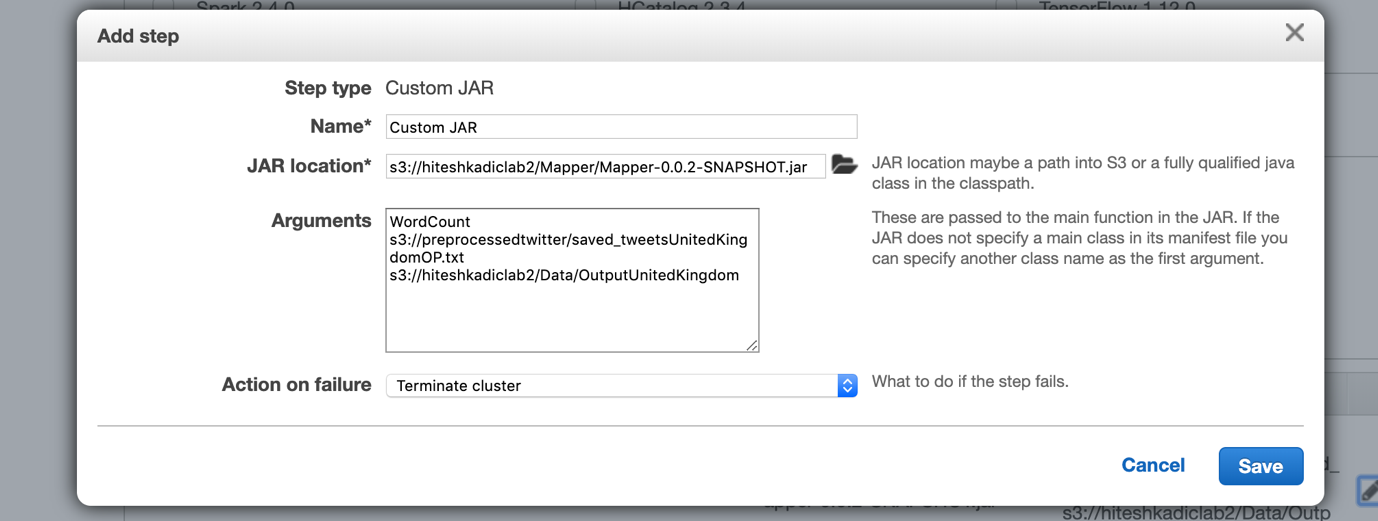
We used AWS EMR for map reduce jobs

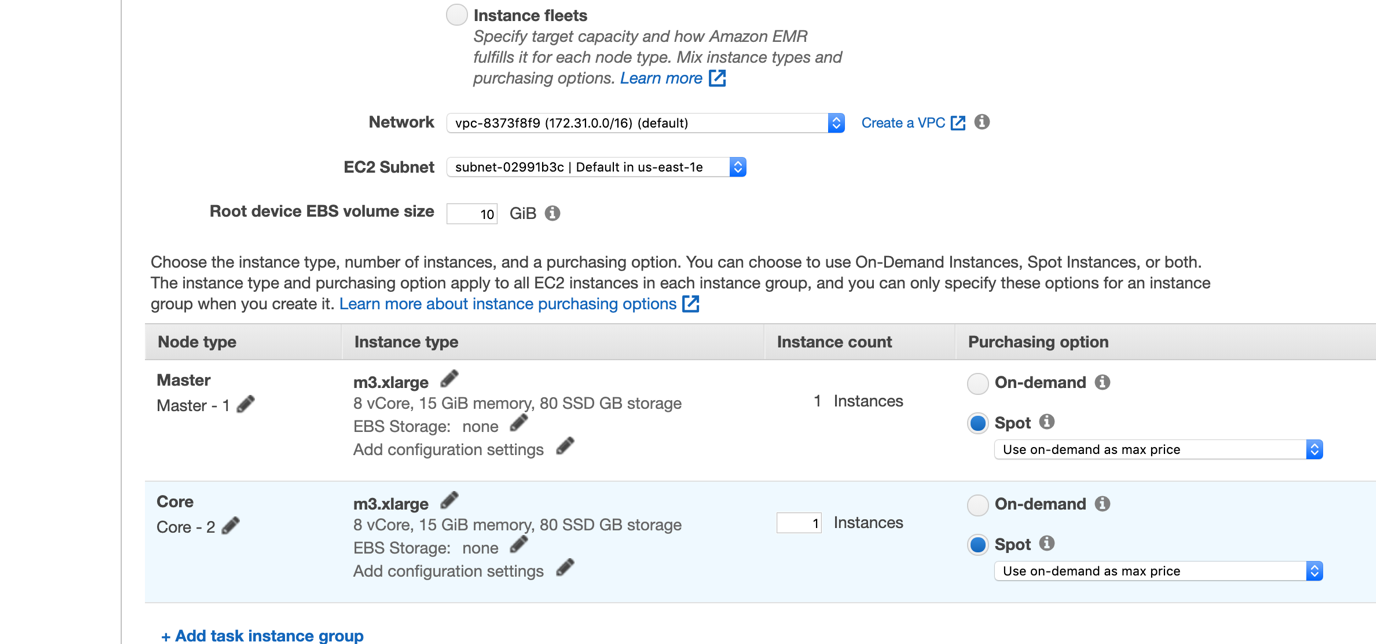
Approach:

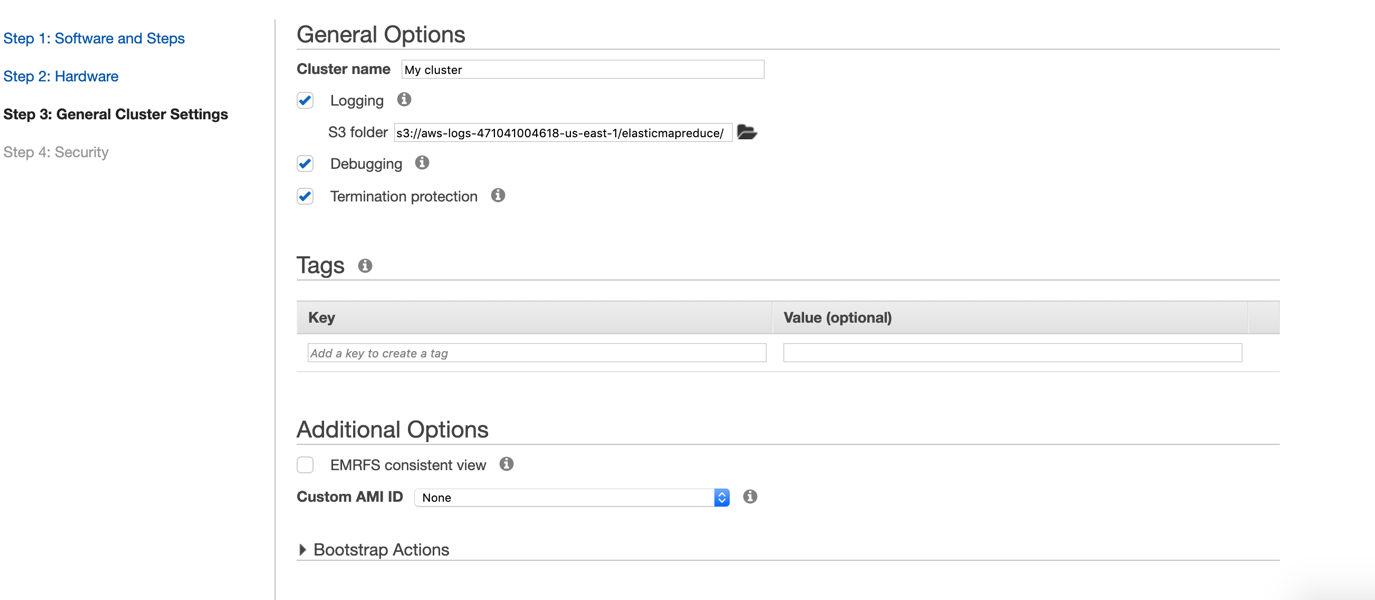
* Upload the cleaned data to S3
* Configure the EMR Cluster
* Configuration:
* Step Custom Jar.
* Arguments: Main class name followed by Input file path and output file path.
* Configure the Hardware by provisioning the MX4.LARGE (1 core and 1 master)
* Select pricing model
* Run the cluster.

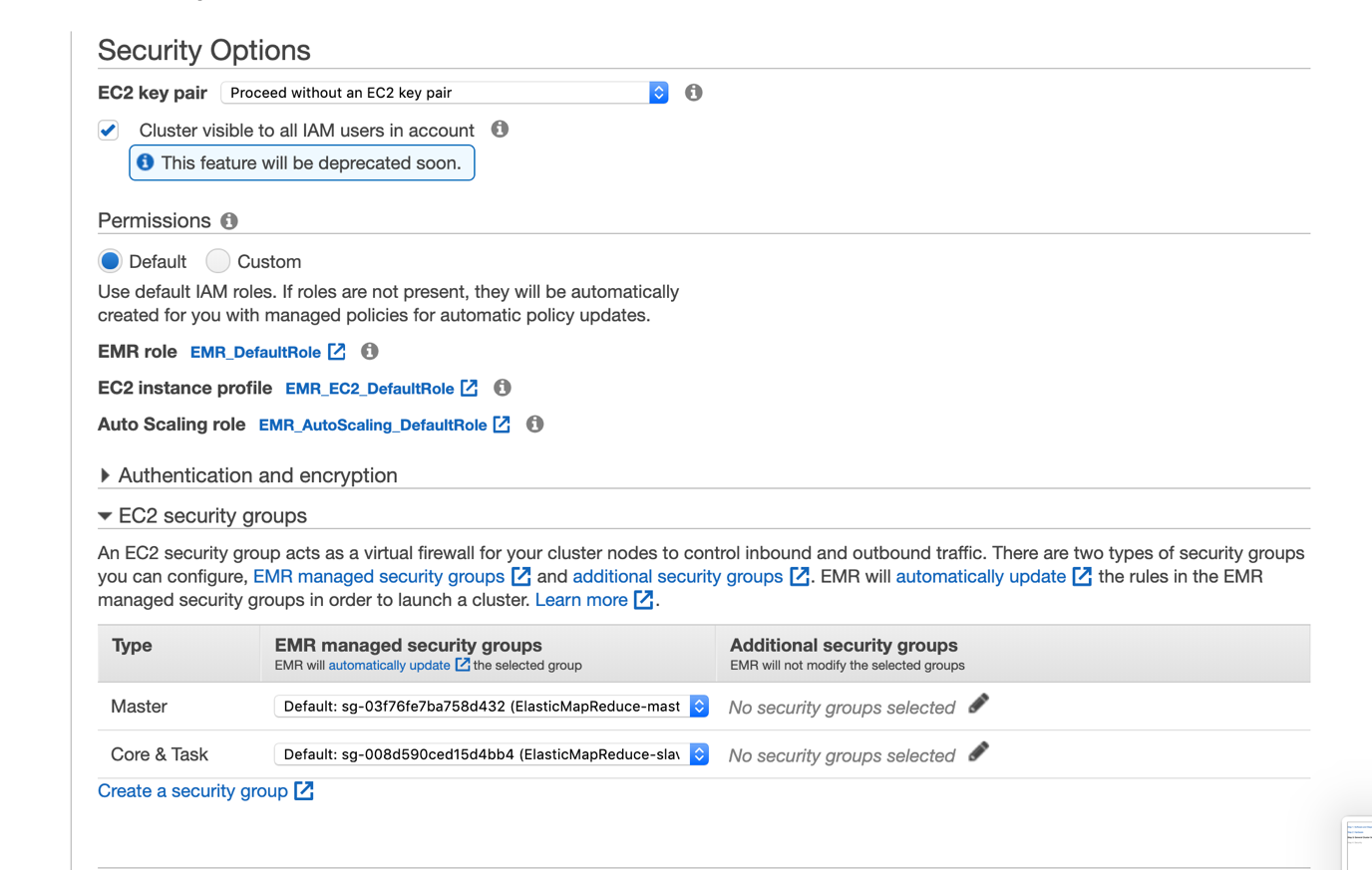
# Below are the screen shots for EMR flow:

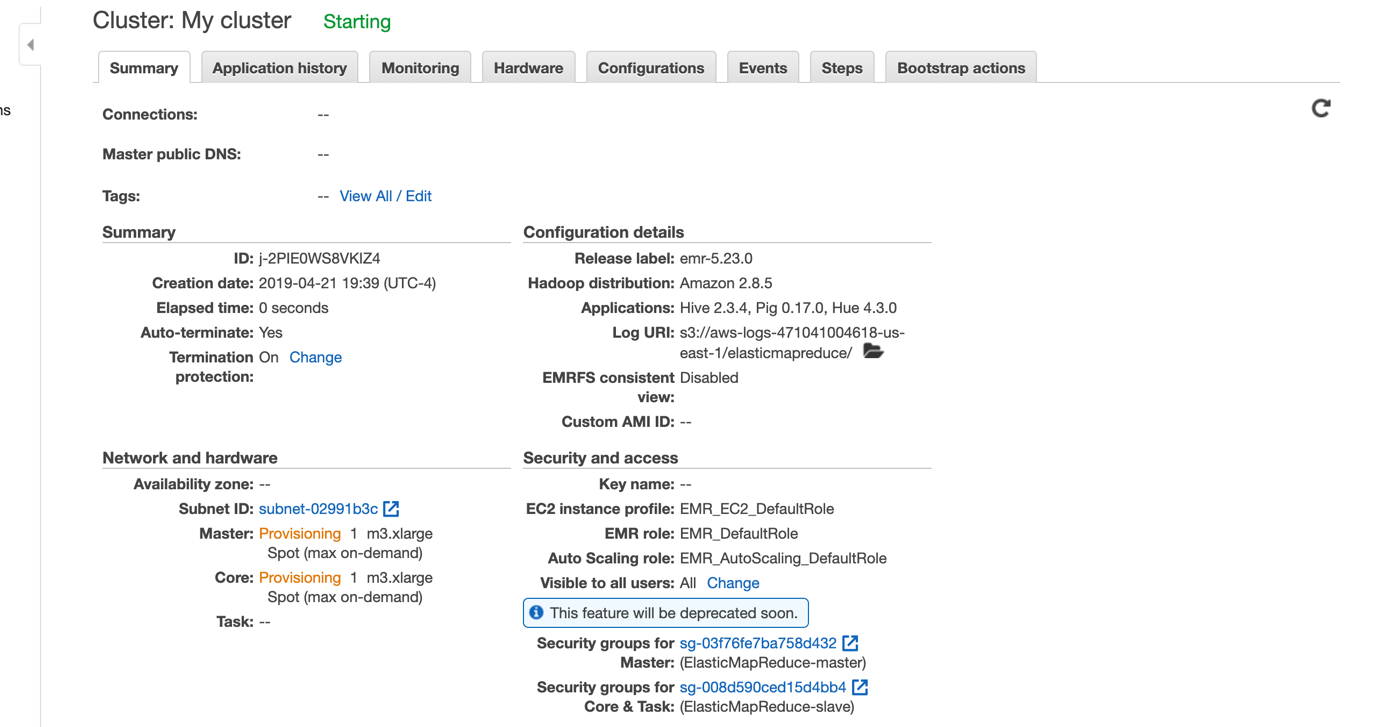


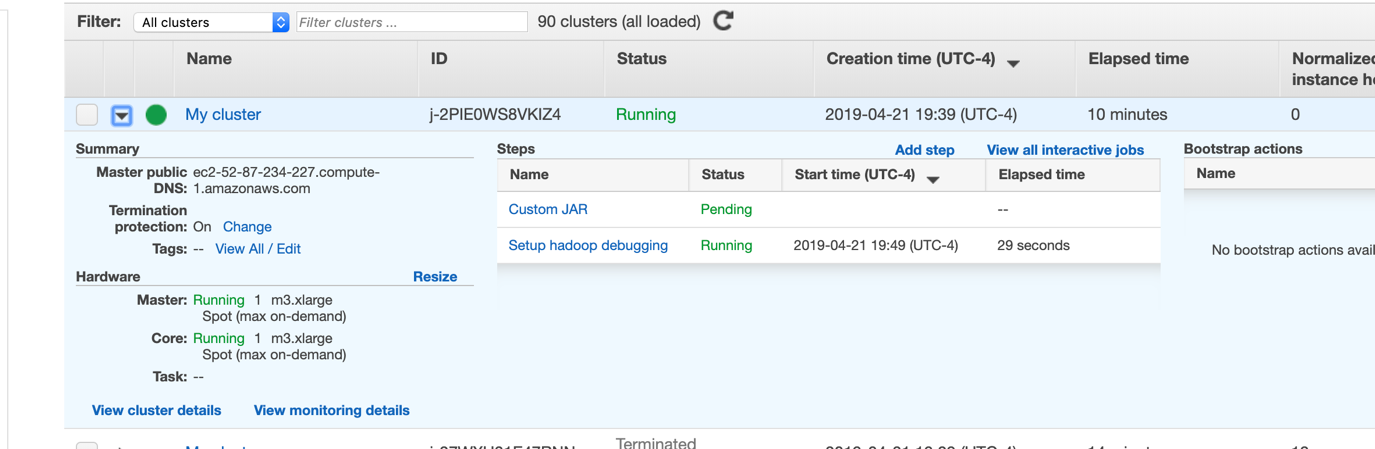


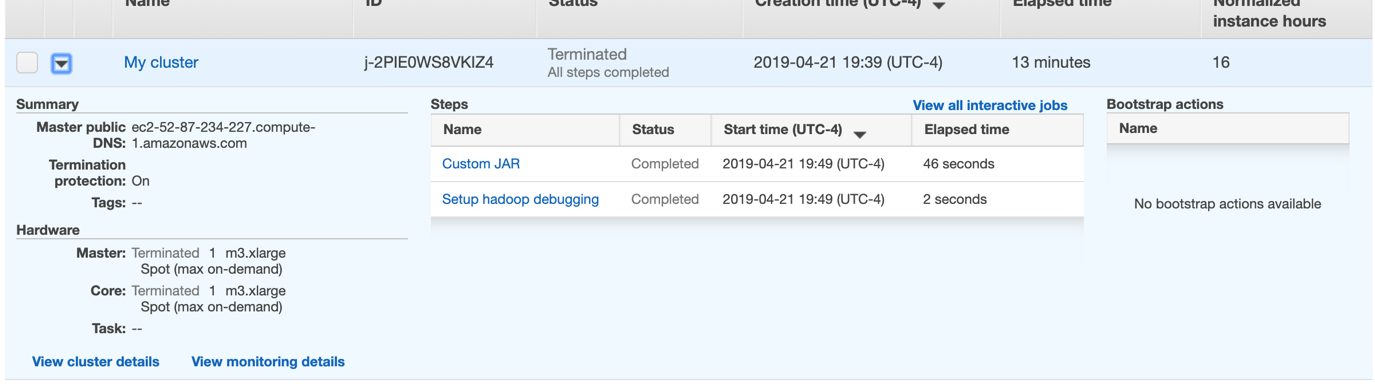


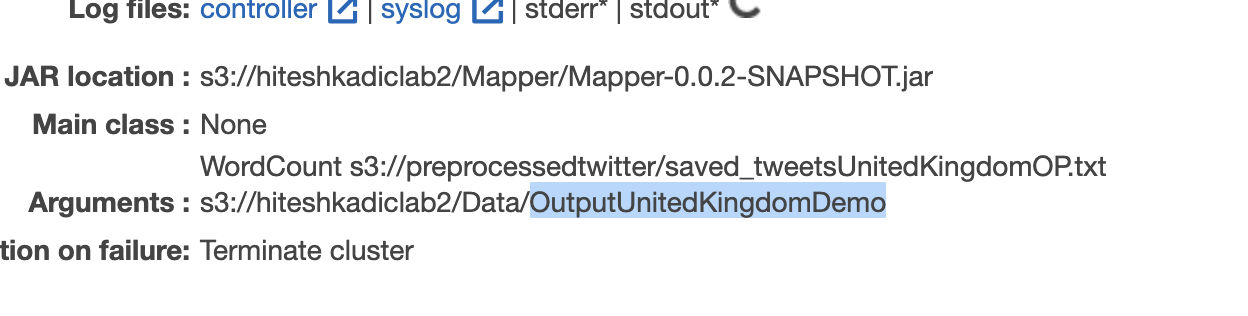


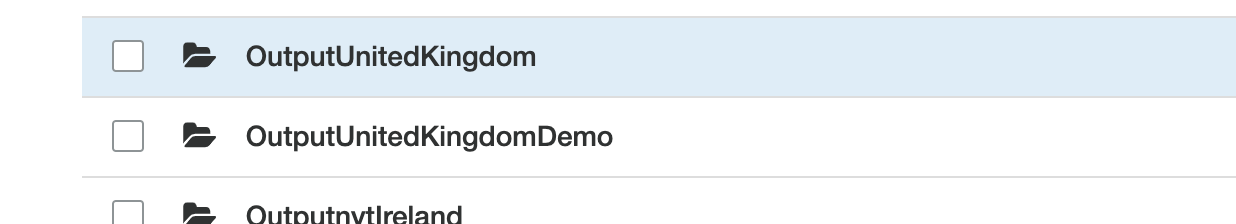


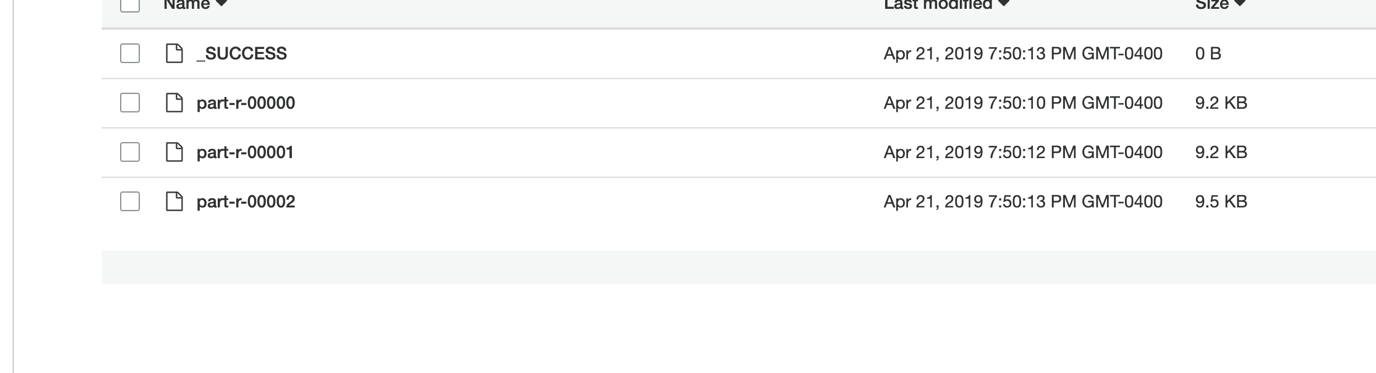


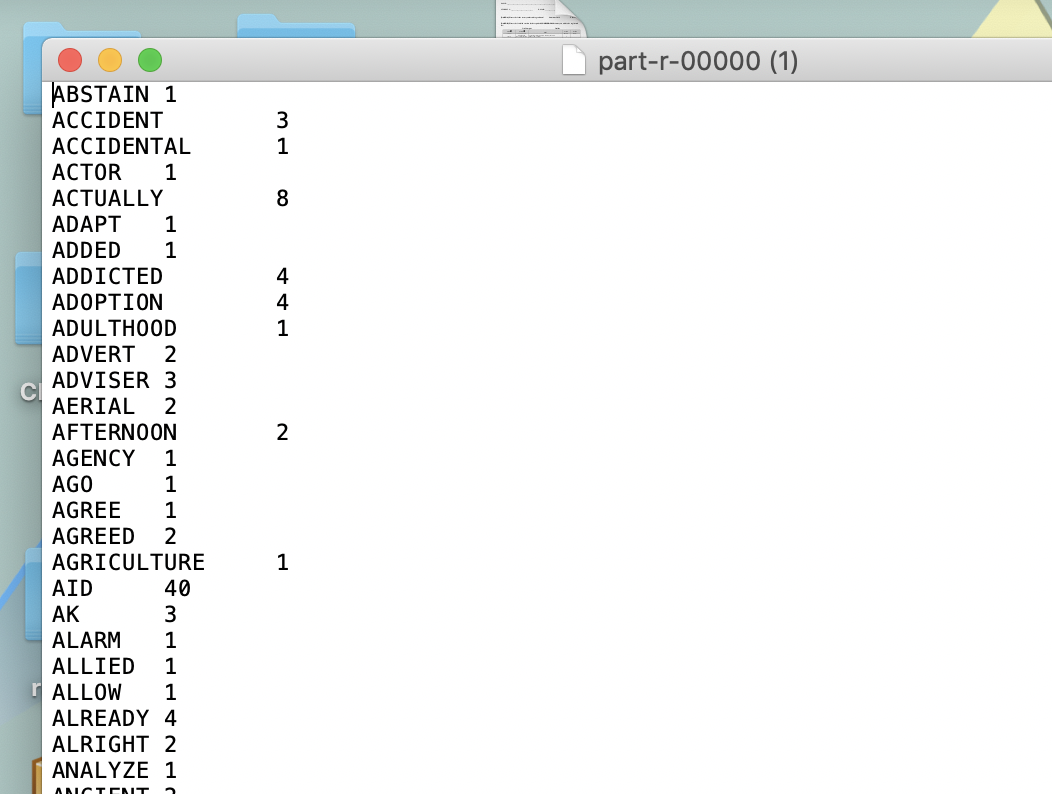












# Conclusion

The output from word count is piped to word co-occurrence EMR job and then all the results are visualized using Tableau.

Resulting Word Count and Word Co-occurrence Data and Tableau workbooks are kept in the submission folder.