Principles of Big Data Management

Project Phase - 1

Team Members:
Ankitha Wankhede - 16233344
Rajeswari Devi Namana -18135299
Sravya Reddy Bokka - 16240386

Goal:

- To Collect Tweets using Twitter's Streaming APIs (e.g., 100K Tweets) (https://dev.twitter.com/docs/streaming-apis)
- Extract all the hashtags and URLs in the tweets
- Run the WordCount example in Apache Hadoop and Apache Spark on the extracted hashtags/URLs and collect the output and log files from Hadoop. Add a README file.

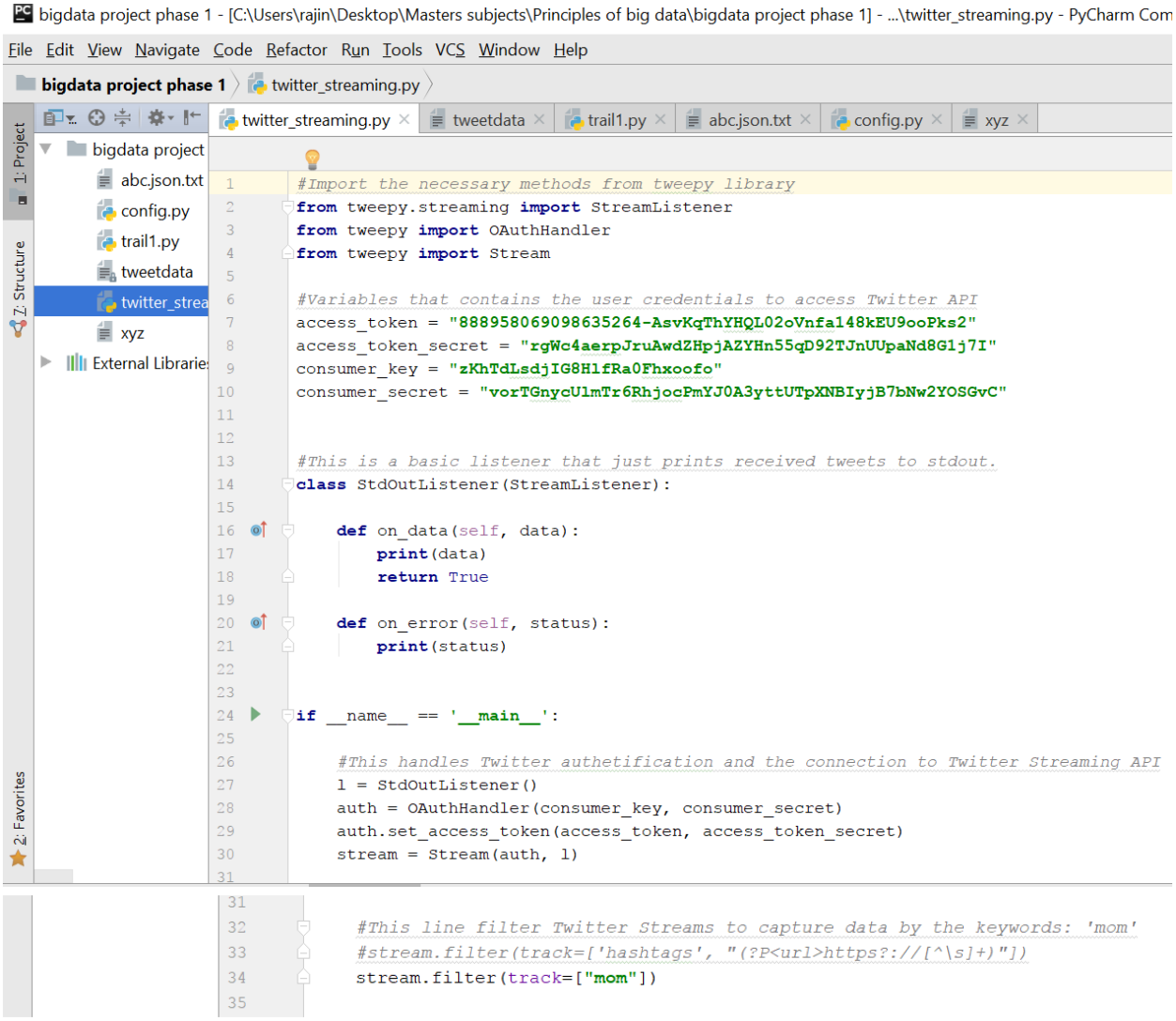
Step 1: Collection of tweets from Twitter Streaming API:

Executed code in python using tweepy library.

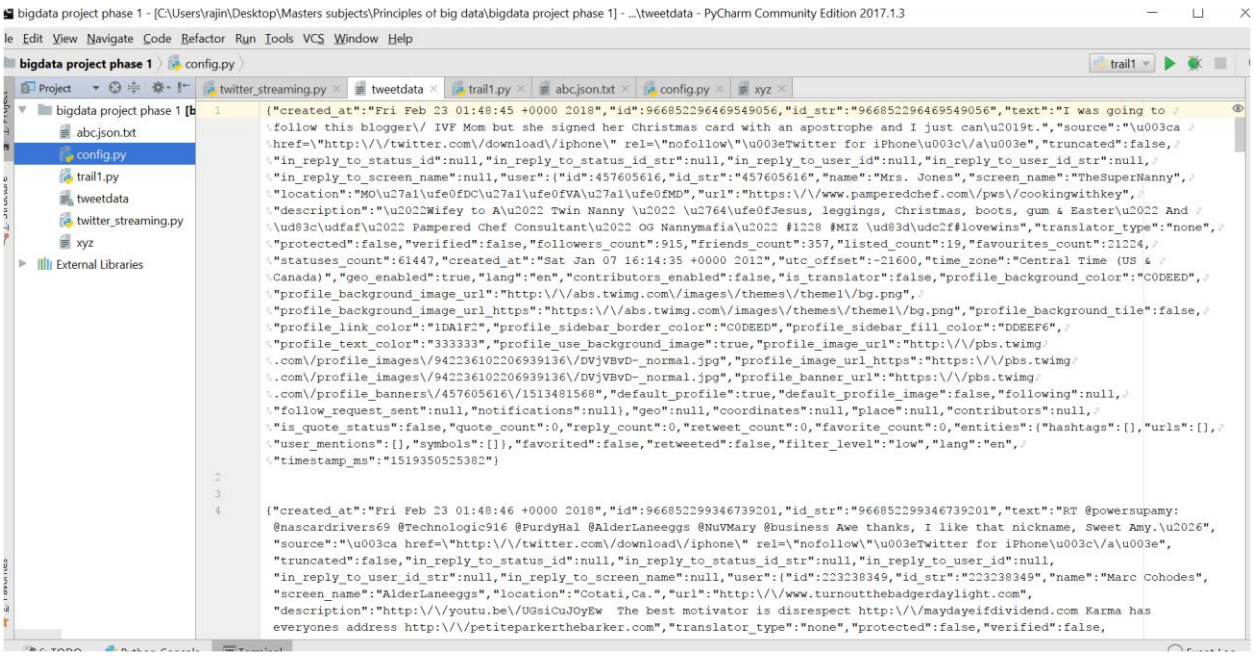
Code Used:

```
#Import the necessary methods from tweepy library
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream

#Variables that contains the user credentials to access Twitter API
access_token = "888958069098635264-AsvKqThYHQL02oVnfa148kEU9ooPks2"
access_token_secret = "rgWc4aerpJruAwdZHpjAZYHn55qD92TJnUUpaNd8G1j7I"
consumer_key = "zKhTdLsdjIG8HlfRa0Fhxoofo"
consumer_secret = "vorTGnycUlmTr6RhjocPmYJ0A3yttUTpXNBIyjB7bNw2YOSGvC"

#This is a basic listener that just prints received tweets to stdout.
class StdOutListener(StreamListener):

    def on_data(self, data):
        print(data)
        return True

    def on_error(self, status):
        print(status)

if __name__ == '__main__':

    #This handles Twitter authentication and the connection to Twitter Streaming API
    l = StdOutListener()
    auth = OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    stream = Stream(auth, l)
    #This line filter Twitter Streams to capture data by the keywords: 'mom'
    #stream.filter(track=['hashtags', "(?P<url>https?://[^\s]+)"])
    stream.filter(track=["mom"])
```

Code Screenshots :



Output Screenshot: Collected Tweets:



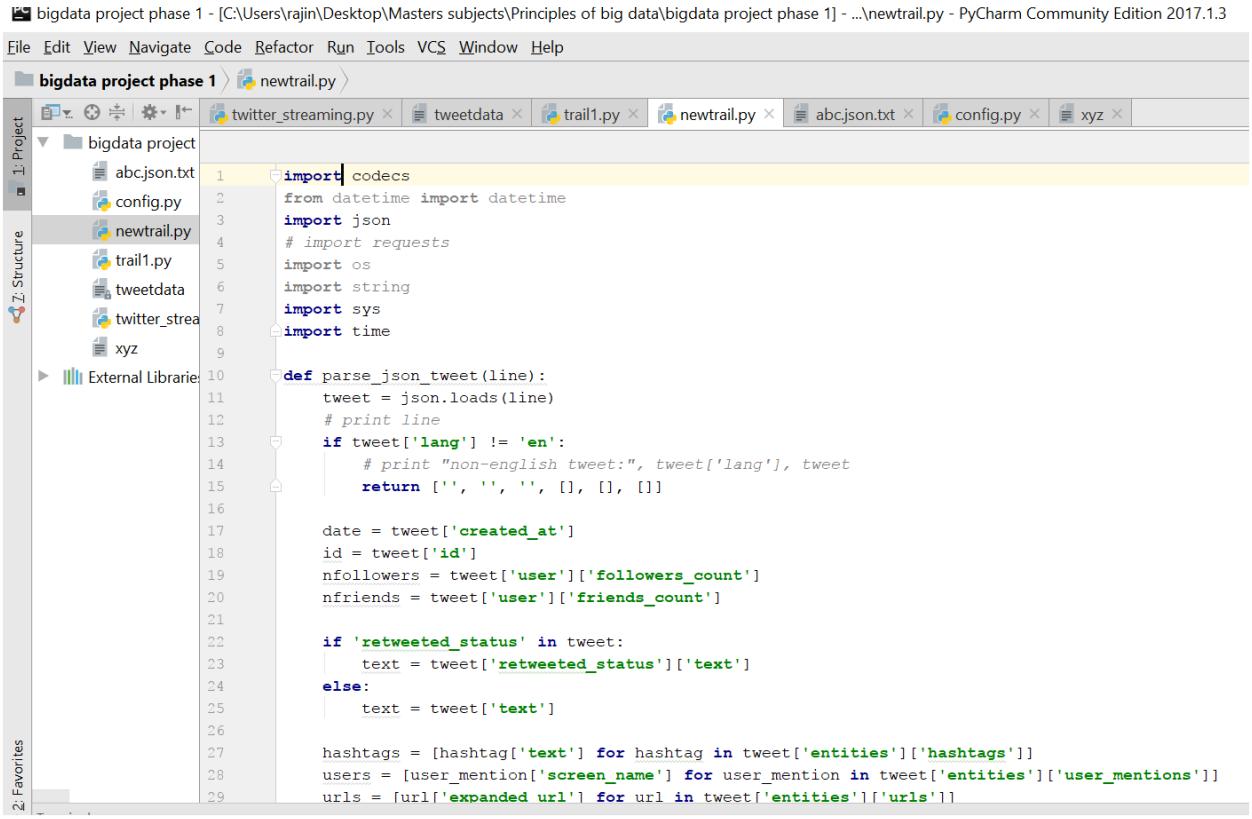
Step 2: Extraction of URLs and hashtags from collected tweets:

Code:

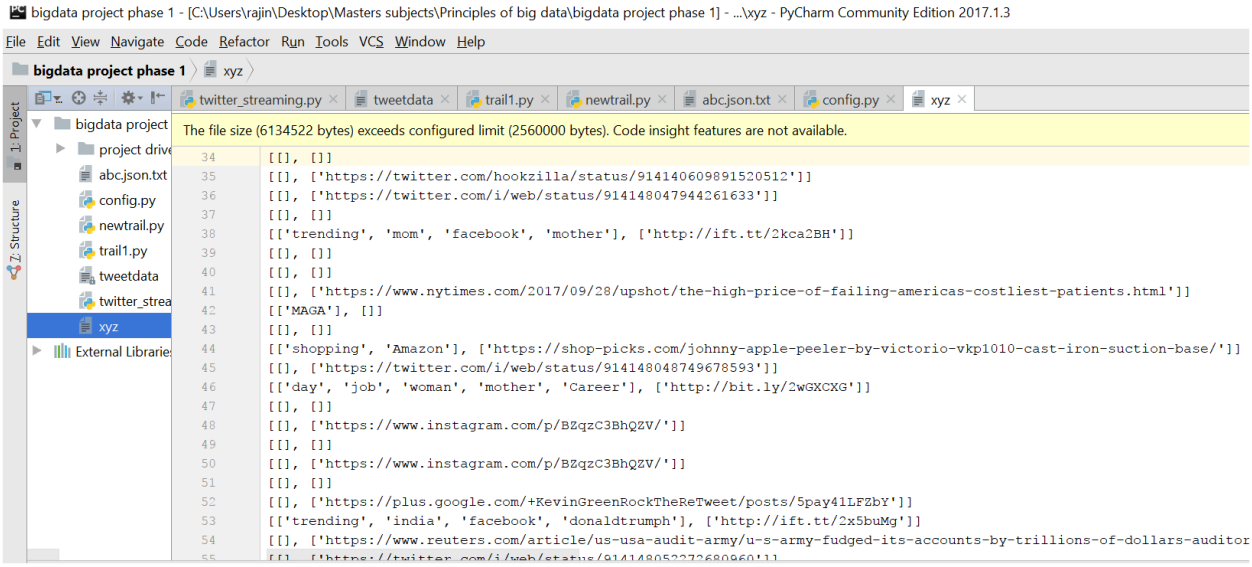
```
import codecs
from datetime import datetime
import json
import os
import string
import sys
import time

def parse_json_tweet(line):
    tweet = json.loads(line)
    # print line
    if tweet['lang'] != 'en':
        # print "non-english tweet:", tweet['lang'], tweet
        return ['', '', '', [], [], []]

    date = tweet['created_at']
    id = tweet['id']
    nfollowers = tweet['user']['followers_count']
    nfriends = tweet['user']['friends_count']

    if 'retweeted_status' in tweet:
        text = tweet['retweeted_status']['text']
    else:
        text = tweet['text']

    hashtags = [hashtag['text'] for hashtag in tweet['entities']['hashtags']]
    users = [user_mention['screen_name'] for user_mention in
tweet['entities']['user_mentions']]
```

```
    urls = [url['expanded_url'] for url in tweet['entities']['urls']]

    media_urls = []
    if 'media' in tweet['entities']:
        media_urls = [media['media_url'] for media in tweet['entities']['media']]

    return [hashtags, urls]

'''start main'''
if __name__ == "__main__":
    file_timeordered_json_tweets = codecs.open(sys.argv[1], 'r', 'utf-8')
    fout = codecs.open(sys.argv[2], 'w', 'utf-8')

    # efficient line-by-line read of big files
    for line in file_timeordered_json_tweets:
        try:
            [tweet_gmttime, tweet_id, text, hashtags, users, urls, media_urls,
nfollowers, nfriends] = parse_json_tweet(
                line)
            #     if not tweet_gmttime: continue
            #     fout.write(line)
            # "created_at":"Mon Feb 17 14:14:44 +0000 2014"
            try:
                c = time.strptime(tweet_gmttime.replace("+0000", ''), '%a %b %d
%H:%M:%S %Y')
            except:
                print("pb with tweet_gmttime", tweet_gmttime, line)
                pass
            tweet_unixtime = int(time.mktime(c))
            #        fout.write(line)
            fout.write(str(
                [tweet_unixtime, tweet_gmttime, tweet_id, text, hashtags, users, urls,
media_urls, nfollowers,
                 nfriends]) + "\n")
        except:
            # print "pb with tweet:", line
            #        print sys.exc_info()[0], line
            pass
    file_timeordered_json_tweets.close()
    fout.close()
```

Code Screenshots :

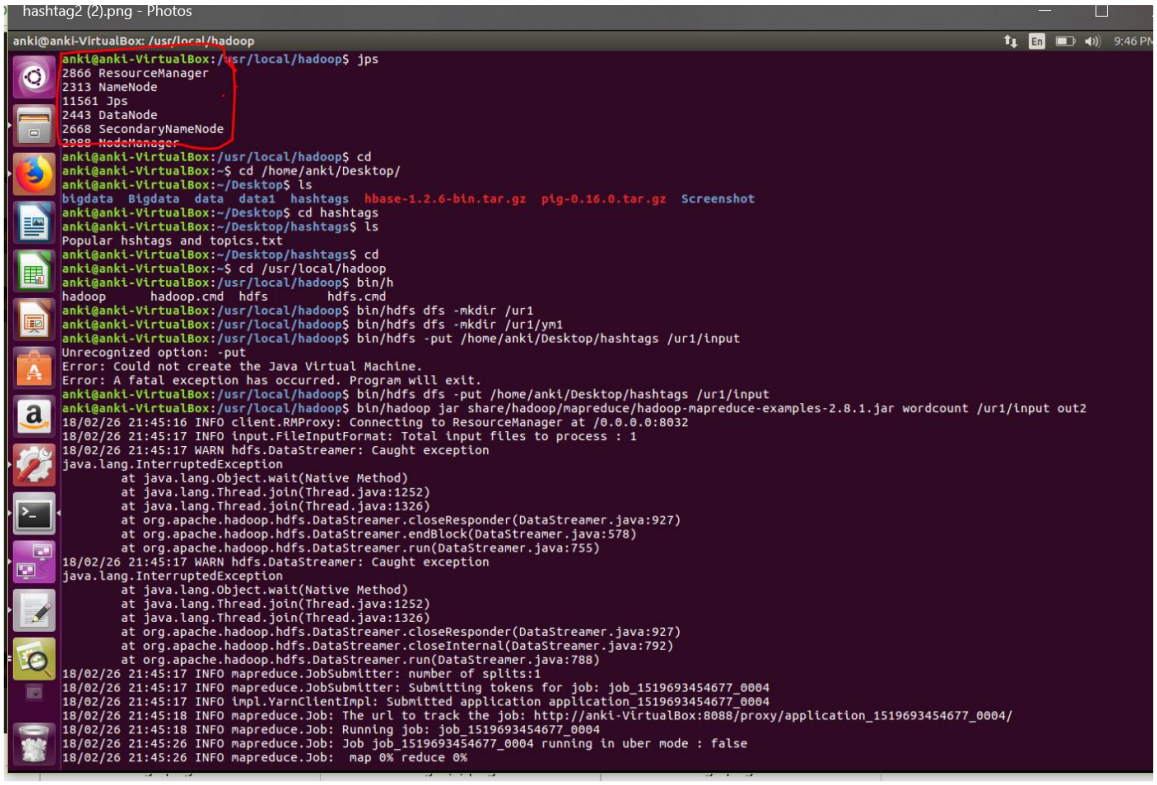


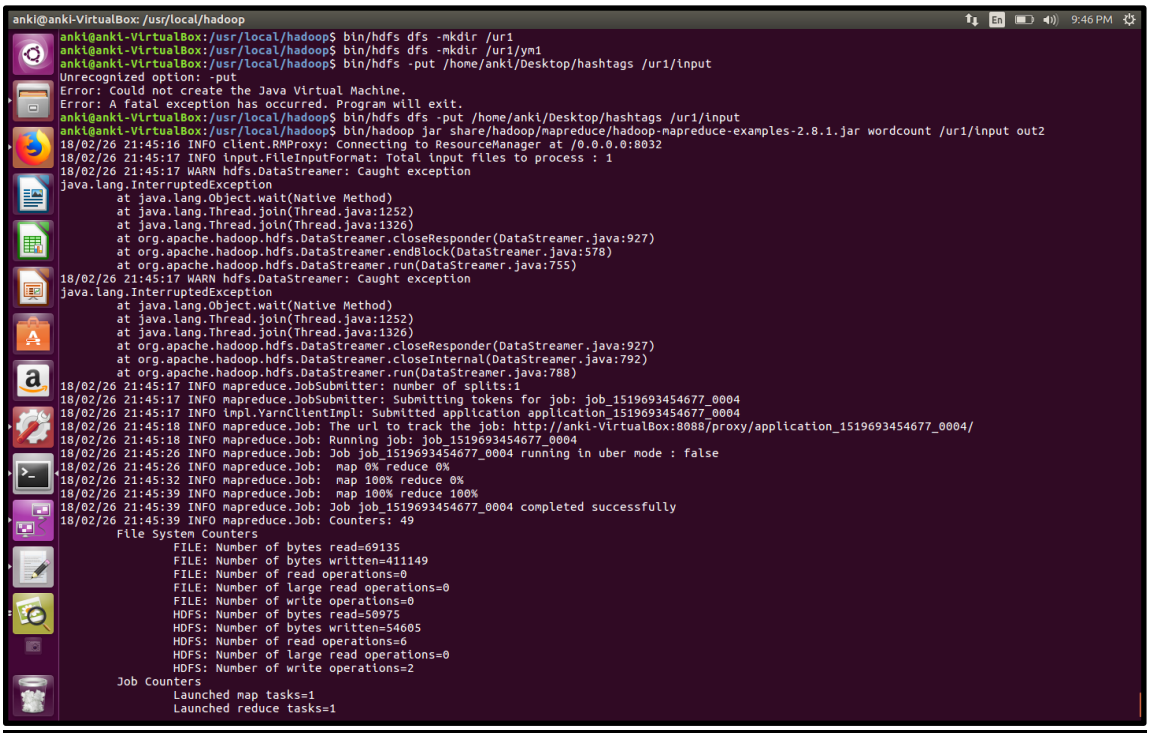
Output Screenshot: Extracted URLs and Hashtags:



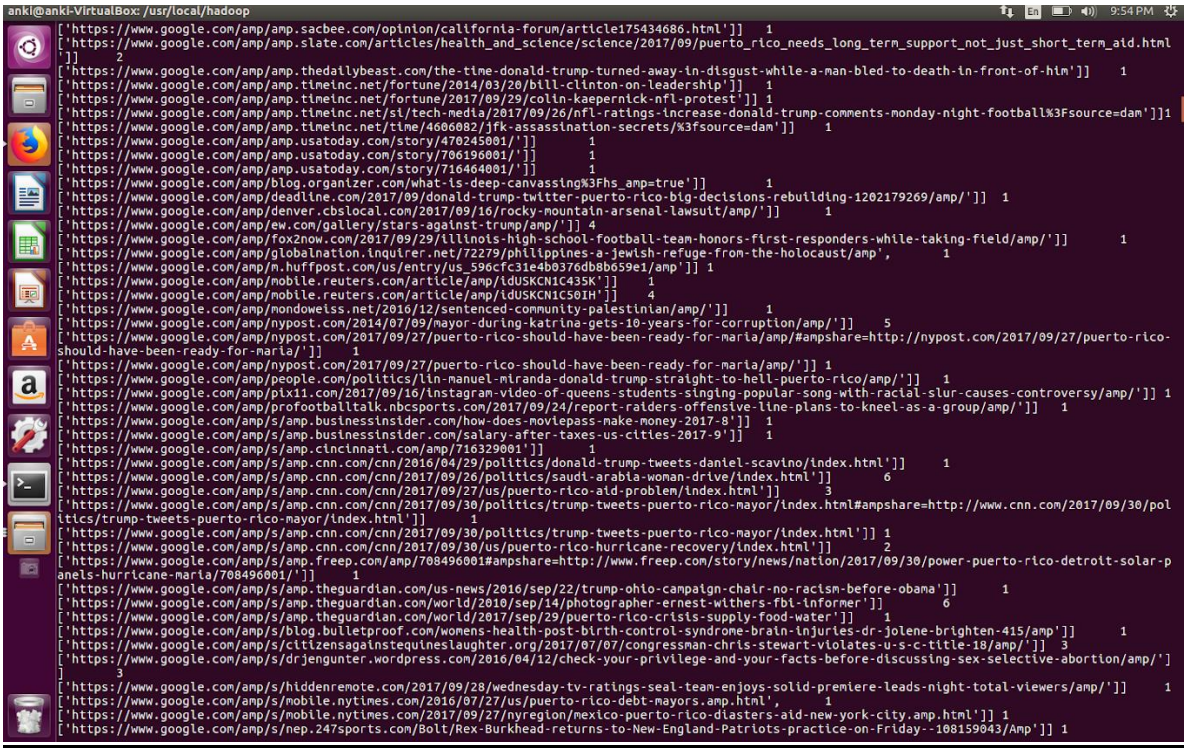
Step 3: Running Word Count Program in Hadoop:

Screenshots of commands executed while running word count program:

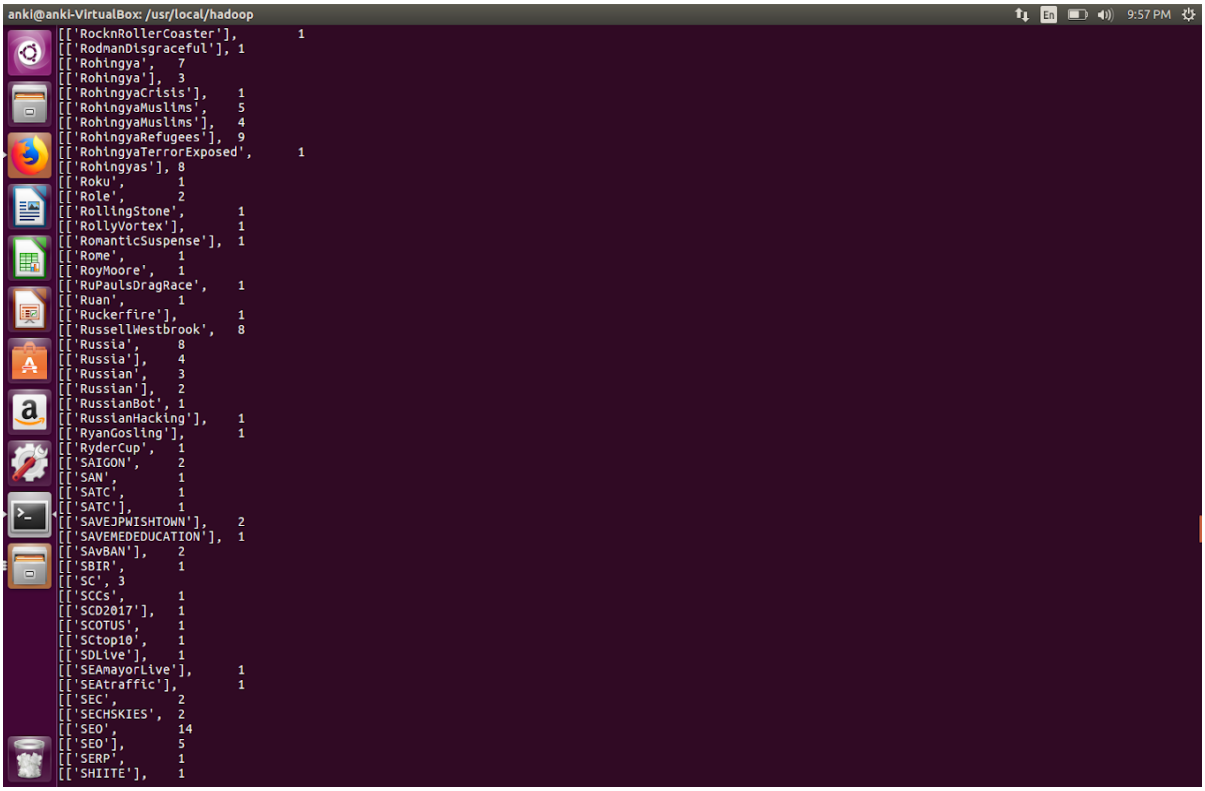




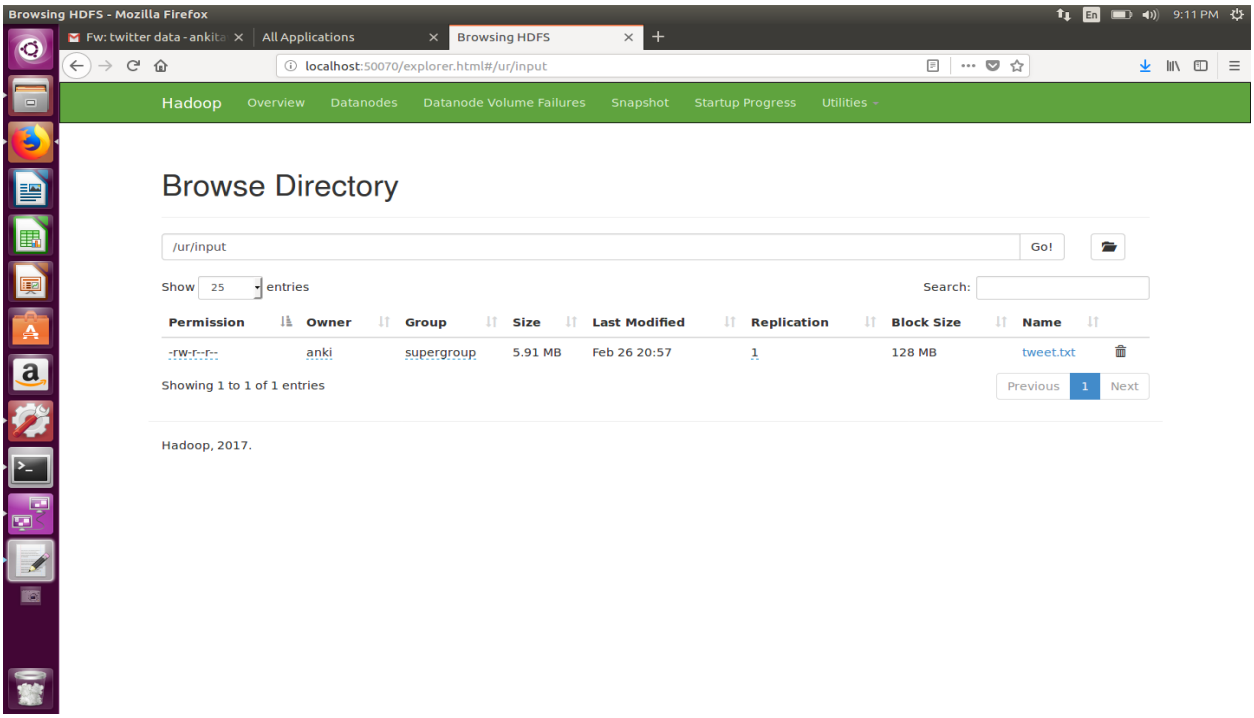
Word Count of URLs:



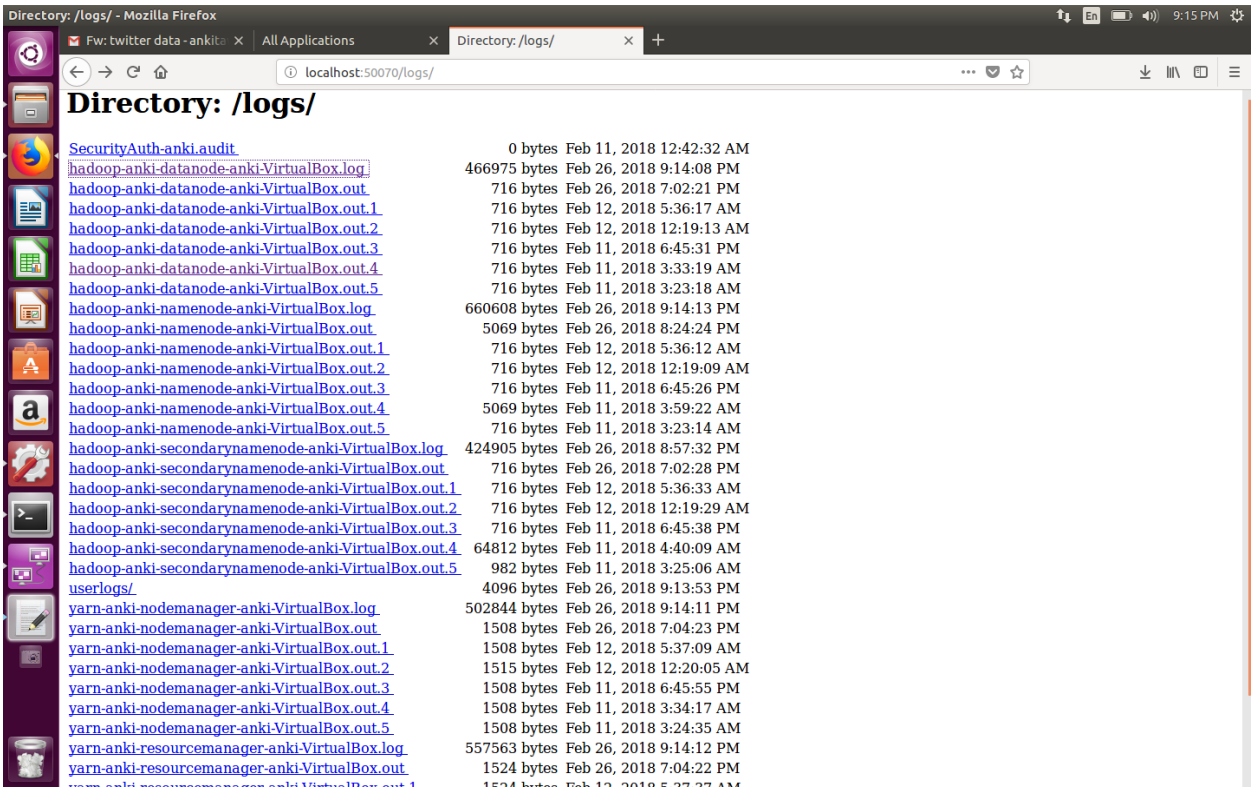
Word Count of HashTags:



HDFS Screenshot :



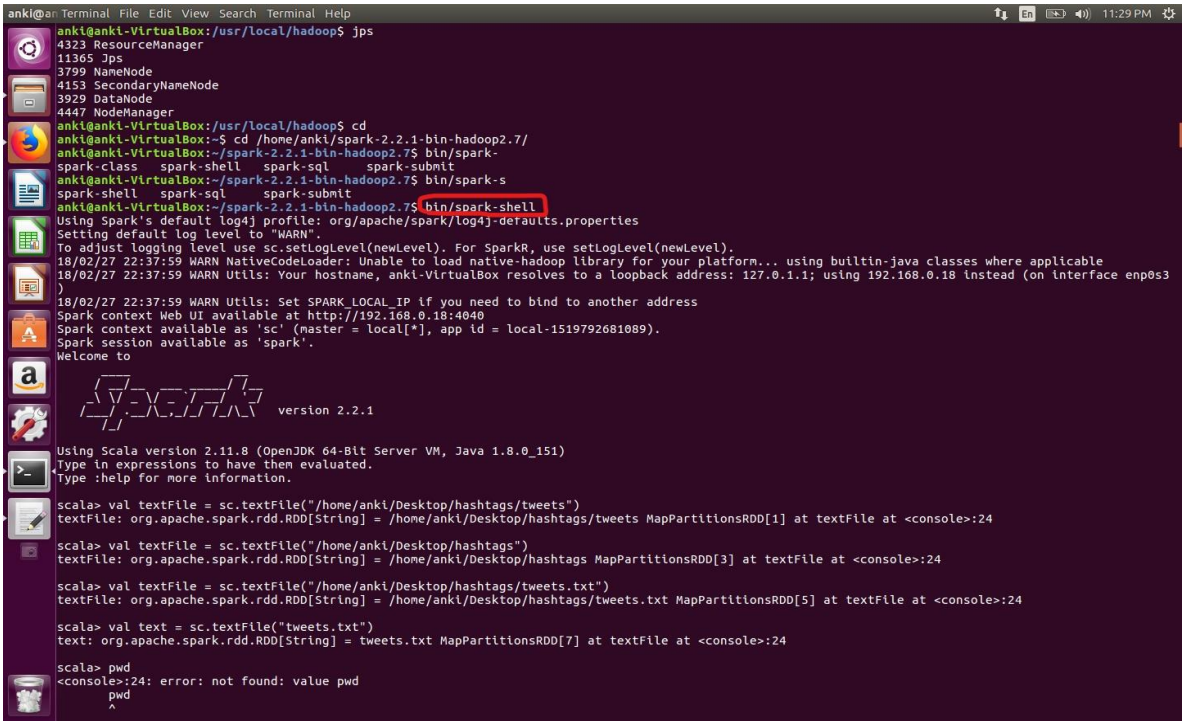
Screenshot of log files created:



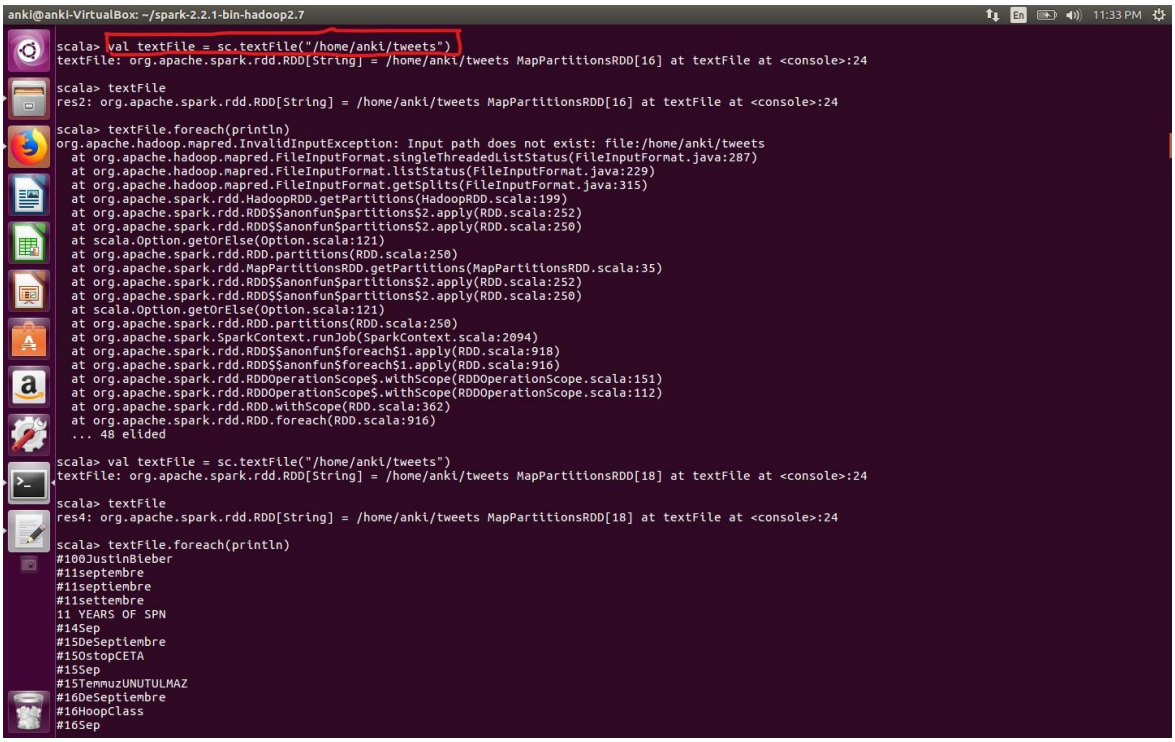
Step 4: Running Word Count Program in Apache Spark:

Screenshots of commands executed while running word count program:

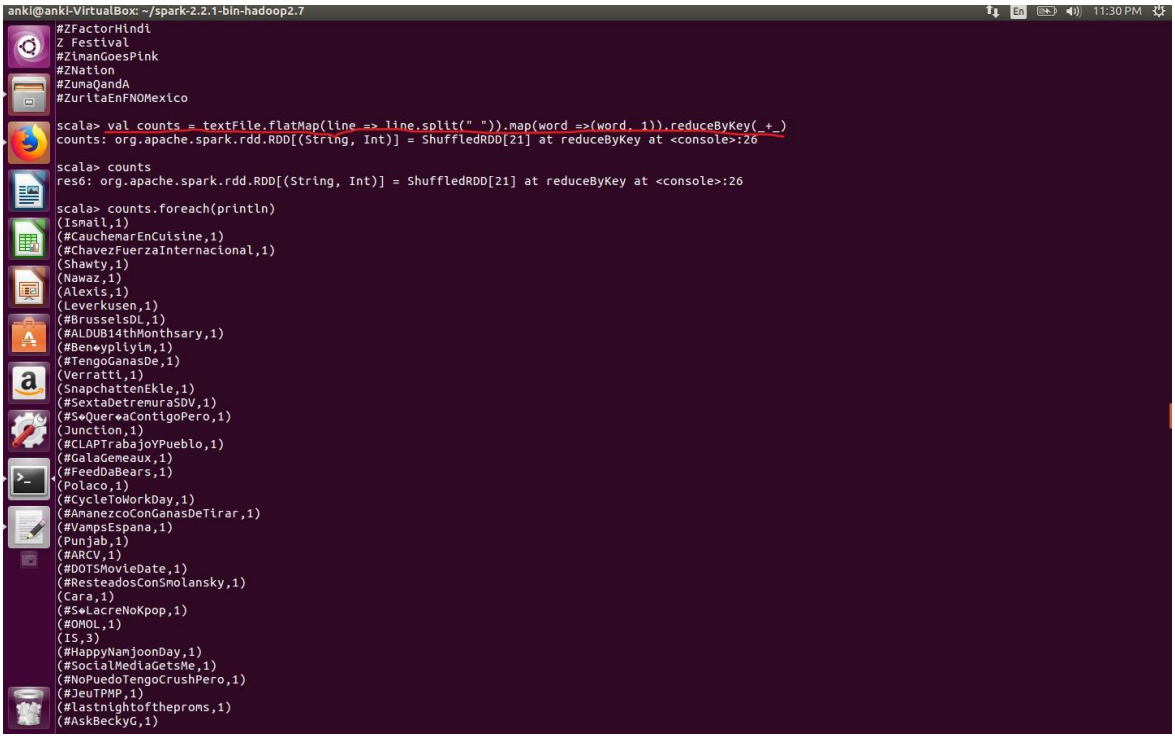
1.Starting Spark: using bin/spark-shell command:



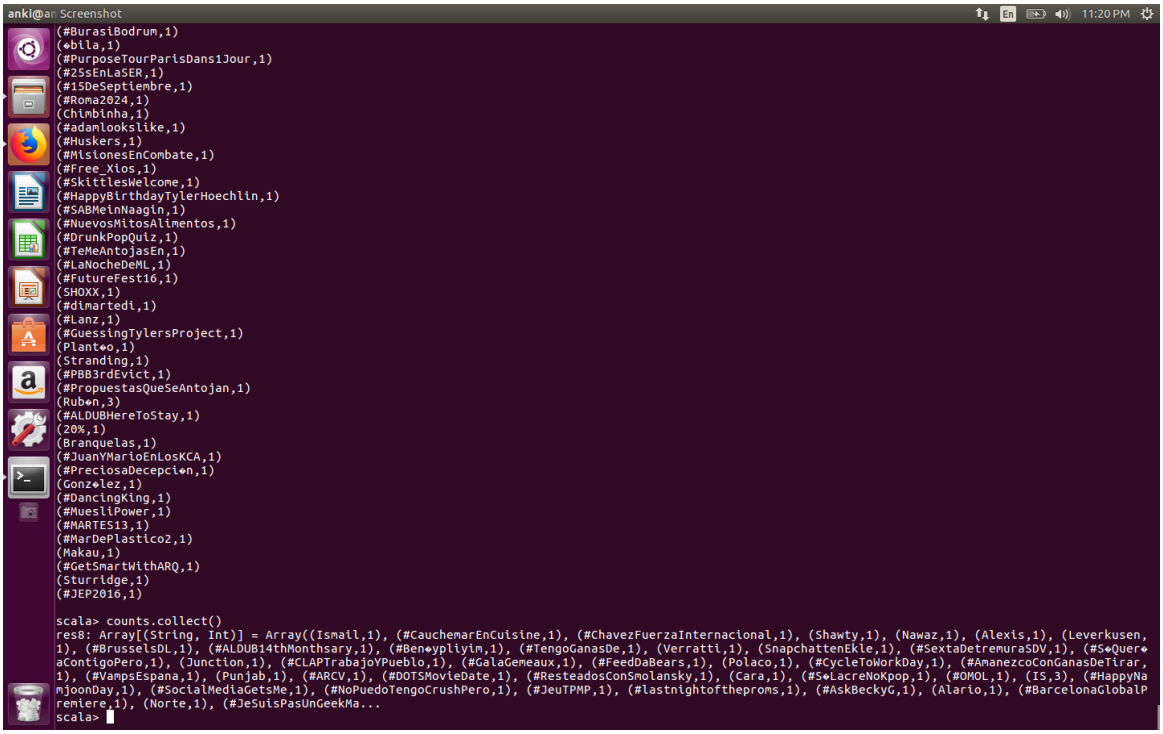
2. Loading the data:



3. Splitting content in our file as strings and applying map and reduce functions to perform word count, using the below highlighted command:



4. Finally collecting output which displays the word count in the form of an array of key value pairs:



References:

http://adilmoujahid.com/posts/2014/07/twitter-analytics/

https://github.com/heerme/twitter-topics/blob/master/extract-json-to-text-stream.py
https://www.youtube.com/watch?v=YZnNb0BTrS4&list=PLJNKKS4iwuamrvNVahopRziurK7XNCc5B&index=3