April 29, 2016
Brennan Borlaug

W205 Exercise 2 - Architecture

Critical File Locations:

MIDS_205_twitter_wordcount			
/data			
	top_20_counts.csv		
	top_20_words.png		
/screenshots			
	final_results_test.png		
	histogram_test.png		
	sparse.png		
/serving_scripts			
	README		
	finalresults.py		
	histogram.py		
/tweetwordcount			
	/topologies		
		tweetwordcount.clj	
	/src		
		/bolts	
			parse.py
			wordcount.py
		/spouts	
			tweets.py
README			
build_tcount.py			

Storm Topology:

root@ip-172-31-61-152:/data/ex_2/tweetwordcount/topologies

```
(ns tweetwordcount
  (:use     [streamparse.specs])
  (:gen-class))

(defn tweetwordcount [options]
   [
    ;; spout configuration
    {"tweet-spout" (python-spout-spec
          options
          "spouts.tweets.Tweets"
          ["tweet"]
          :p 1
          )
    }
    ;; bolt configuration
    {"parse-tweet-bolt" (python-bolt-spec
          options
          {"tweet-spout" :shuffle}
          "bolts.parse.ParseTweet"
          ["word"]
          :p 2
          )
     "count-bolt" (python-bolt-spec
          options
          {"parse-tweet-bolt" ["word"]}
          "bolts.wordcount.WordCounter"
          ["word" "count"]
          :p 2
          )
    }
  ]
)
~
~
```

Instructions:

1. Insert independent key & token for Twitter app into tweetwordcount/src/spout/**tweets.py**.

2. Run **build_tcount.py** setup script.

3. Run Streamparse:

   $ cd tweetwordcount
   $ sparse run

4. Run serving scripts, **finalresults.py** & **histogram.py** (see 'serving_scripts' folder for more details).

Serving Scripts:

1. **finalresults.py** - Script that takes one word as an argument & replies with the total number of occurences for the word in the stream. If no arguments are passed, script will return all words in stream & their total number of occurences, sorted alphabetically in ascending order.

Example:

$ python finalresults.py hello

$ Total number of occurences of 'hello': 10

$

$ python finalresults.py

$ (Also, 2)

$ (Amazing, 1)

$ (Americans, 3)

$ ...

$ ..

2. **histogram.py** - Script that takes two integers (k1,k2) [No space] & returns all words in the stream that have a greater (or equal) number of occurences than k1 & less (or equal number of) occurances than k2.

Example:

$ python histogram.py 3,8

$ (Announces, 7)

$ (At, 4)

$ (Day, 3)

$ ...