W205 Data Storage and Retrieval Exercise 2 Application Architecture
Creator: Marcus DeMaster

1. Application Idea
    a. The purpose of this streaming application is to use the Twitter streaming API to mine live tweets, parse the tweets into a list of words, and count the frequency of those words as the tweets come in. The Storm application continually updates a postgres database with a table that stores the words and their counts. A couple of scripts allow the user to query that table and return information from it based on a specified word or range of counts.

2. Description of the Architecture
    a. The application architecture consists of a Storm topology that includes a spout called tweet-spout that reads the live stream of tweets using the python Tweepy library. The topology also has a bolt called parse-tweet-bolt, which parses the tweets for individual words from the tweet-spout and emits the words to the next bolt. This last bolt, count-bolt, counts the number of each word and uses a pscyopg2 library to commit psql statements that update the word counts in the postgres database called Tcount. The tweet-spout, parse-tweet-bolt, and count-bolt have 3,3, and 2 nodes respectively.

3. Directory and File Structure by name
    a. mdemaster_w205_exercise2
        i. add-license.sh
        ii. build
        iii. dist
        iv. doc
        v. examples
        vi. jvm
        vii. screenshots
            1. screenshot-finalresults-no-parameter.png
            2. screenshot-finalresults-with-parameter.png
            3. screenshot-histogram-results.png
            4. screenshot-storm-components.png
            5. screenshot-twitterStream.png
            6. screenshots.txt
        viii. DEVELOP.md
        ix. dev-requirements.txt
        x. examples
        xi. EXtweetwordcount
            1. _build
            2. config.json
            3. fabfile.py
            4. finalresults.py

5. histogram.py
6. logs
7. project.clj
8. README.md
9. _resources
10. src
    a. spouts
        i. __init__.py
        ii. tweets.py
    b. bolts
        i. __init__.py
        ii. parse.py
        iii. wordcount.py
11. tasks.py
12. topologies
    a. tweetwordcount.clj
13. virtualenvs

xii. LICENSE
xiii. MANIFEST.in
xiv. README.rst
xv. requirements.txt
xvi. setup.cfg
xvii. setup.py
xviii. streamparse
xix. streamparse.egg-info
xx. tasks.py
xxi. test
xxii. test-requirements.txt
xxiii. tox.ini
xxiv. CHANGES.md
xxv. DEVELOP.md
xxvi. LICENSE
xxvii. MANIFEST.in
xxviii. NOTES.MD
xxix. PLOT.png
xxx. README.md
xxxi. README.rst
xxxii. Readme.txt
xxxiii. Twittercredentials.py
xxxiv. add-license.sh
xxxv. conda.yaml
xxxvi. create_tcount_db.py
xxxvii. hello-stream-twitter.py
xxxviii. psycopg-sample.py

xxxix. requirements.txt
xl. setup.cfg
xli. setup.py

4. File Dependencies
   a. File: tweetwordcount.clj
      i. Dependencies
         1. tweets.py
         2. parse.py
         3. wordcount.py
   b. File: finalresults.py
      i. Dependencies