Overview of Twitter Application and Architecture

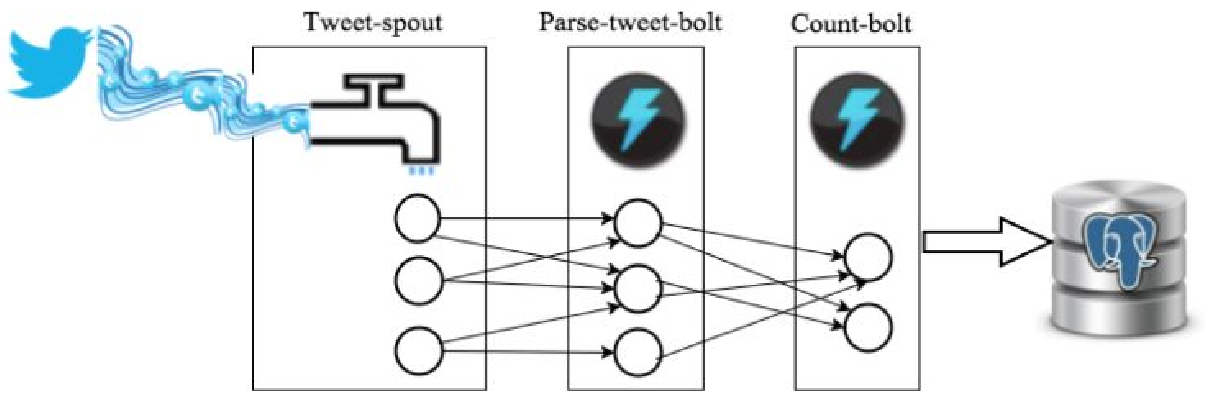
Overview of the application:

This twitter application pulls down tweets, parses them, and then counts the number of times individual words are used. This can be used to understand what words are used most often. The twitter application is connected to pisausa.net a nonprofit organization.

Overview of architecture:

The application works by pulling tweets through the tweets.py script. This script acts as a spout and pulls data into the the streamparse. This script is dependent on the twitter credential information in the script itself and should match the credentials in Twitter credentials.py. The scripts parse.py depend on tweets.py and acts as a parser for the data pulled in by tweets.py. The parser splits tweets into individual words. The parse then feeds into wordcount.py which counts the number of times a word has been used in the tweets pulled. In wordcount the postgres database tcount is updated with new wordcounts for each word. In order for wordcount.py to update the postgres database the setup.py script in the exercise2 folder must be run. This script sets up the tcount database as well as the table needed to track the wordcounts. All of these files are run based based on the topology file tweetwordcount.clj. These files require that the shell_setup script has been run which activates the python 2.7 environment.

A visual of this architecture description is below:



Overview of how to run the application:

To run the application use the readme file. A quick overview is that the shell_setup.sh script and setup.py script in exercise2 must be run first. Then in the tweetwordcount directory you can run sparse run. After that in the exercise2 directory, finalresults.py, top20.py, histogram.py can be used to understand the data that has been stored in the postgres database.

Overview of file structure:

The files for the streamparse of the twitter application are contained in the tweetwordcount folder. In that folder exist the directories topologies (which contains the topology of how the files are connected - tweetwordcount.clj) as well as src/spout (which contains tweets.py which pulls in the twitter data) and src/bolts (which contains parse.py and wordcount.py which parses and counts the words from tweets.py).

The files to setup the environment and postgres database are in the main exercise2 folder. Finally the files to view the results are in the main exercise2 folder.

Directory:

Exercise2				Main director
	readme.txt			Information about the
	shell_setup.sh			Sets up python 2.7 environment
	setup.py			Sets up postgres database and table
	plots.png			Bar graph of top 20 results
	finalresults.py			Shows all words and counts or the count for the parameter given
	histogram.py			Shows words with count between parameters given
	top20.py			Shows top 20 results
	Twittercredentials.py / Twittercredentials.pyc			Example from class directory
	hello-stream-twiter.py			Example from class directory
	psycopg-sample.py			Example from class directory
	tweetwordcount			Streamsparse directory
		README.md		empty
		config.json		Streamparse setup file
		fabfile.py		Streamparse setup file
		project.clj		Streamparse setup file
		tasks.py		Streamparse setup file
		virtualenvs		directory
			tweetwordcount.txt	Explains requirements for streamparse
		topologies		directory
			tweetwordcount.clj	Describes how streamsparse components are related

		src			directory
			spouts		directory
				init.py	empty
				tweets.py	Pulls in twitter data based on credentials listed
			bolts		directory
				init.py	empty
				parse.py	Script for parsing tweets
				wordcount.py	Script to count words from parse.py
		logs			directory
			streamparse_tweetwordcount_tweet-spout...		List of logs from the twitter application
		_resources/resources			directory
			spouts		directory
				init.py	empty
				tweets.py	Pulls in twitter data based on credentials listed
			bolts		directory
				init.py	empty
				parse.py	Script for parsing tweets
				wordcount.py	Script to count words from parse.py
		_build			directory
			classes		directory
				pom.properties	Properties of tweetwordcount
			stale		directory
				leiningen.core.classpath.extract-native-dependencies	Contains dependency data
	screenshots				Directory with screenshots of results
		screen_shot_results_alphabetic.png			First set of results from finalresults.py
		screen_shot_results_between_180_and_450.png			Results from histogram.py 180,450
		screen_shot_storm_components.png			Components in tweetwordcount
		screen_shot_top20.png			Results from top20.py
		screen_shot_topology.png			tweetwordcount.clj
		screen_shot_twitter_stream.png			Twitter stream sample