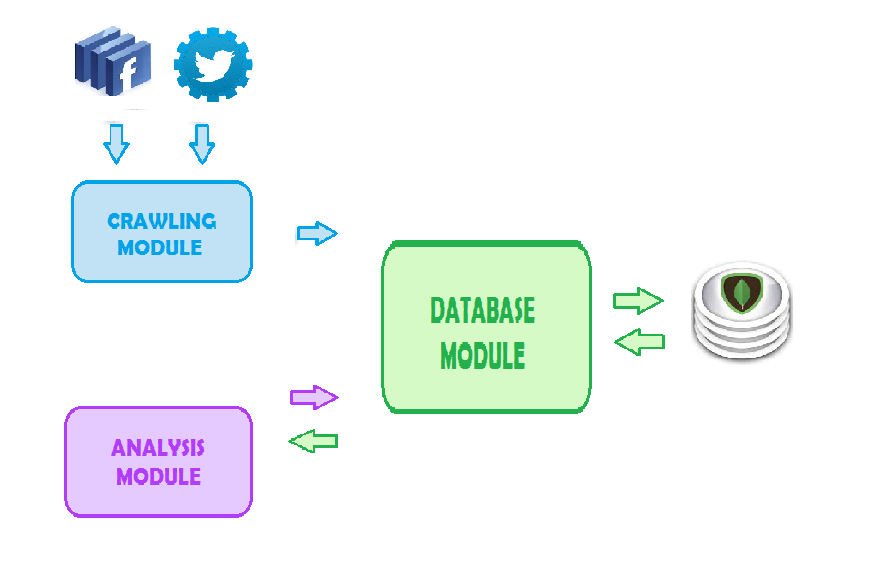
SocialNetworkAnalyzer

## Structure



The package is divided in three logically separate modules which can be divided into sub modules if necessary.

* Crawling module

This module contains scripts that can be divided into sub modules depending on the social networking service you wish to collect data from. Currently contains scripts for:

* Twitter
* Facebook

The *Crawling Module* contains sub module for authentication and authorization of the application and communicates directly with the api of the service. It also communicates with the *Database Module* in order to save all the data retrieved from the internet, into Mongo database.

* Database Module

The *Database Module* is an independent unity that contains one script that is used only for direct communication with the database and wrapper sub packages that contain the modules with logic for querying and formatting the data before inserting into the database.

* Twitter wrapper
* Facebook wrapper (this is a placeholder and not used)
* Analysis Module

Contains sub packages that contain the logic of how should the data be retrieved from the database, special functions that are needed to collect certain information from the collections in the database. Also contains methods that create graphs for further analysis with other visualization and exploration platforms such as Gephi for example.

Structure > organizing files

Resources folder should contain :

-error.log [file for errors]

-twitter\_oauth.txt[ where twitter access token is stored for longterm use]

-facebook\_oath.txt [ not needed at the moment cause access token is really easy and fast to get on very authorization. Might be considered in the future ]

-CC Social media.xlsx[ temporary list with pages from facebook :: optional]

config.py [ file that contains all paths for files used inside the modules(read/write files, error logs, oauth files),system path, and consumer key and secret for facebook and twitter authorization, includes debugging setup methods that allow debug\_printing ]

debugging\_setup.py [ loggers are set up and contains method for printing while debugging]

main.py [ contains menus to choose between functions from both social networks. Some of the functions are tested with hardcoded screen-names/ id-s, names of databases and collections ]

**AnalysisModule>**

**Facebook**

* **pages.py [** methods for manipulation with data for Facebook Pages **]**

**Twitter**

* **grap.py [** methods for creating different types of graphs and exporting graphs in .GMLformat **]**
* **tweets.py [** methods for manipulation with entities from tweets, sorting tweets etc.**]**
* **user.py [** methods for manipulation with data retrieved for a specific user **]**

**CrawlingModule>**

**Facebook**

* **authorization.py [** methods for authorizing and getting access token **]**
* **pages.py [** methods that make api requestsfor data from facebook pages **]**

**Twitter**

* **authorization.py [** methods for authorization and getting a twitter api object **]**
* **list.py [** methods that collect data for members of twitter lists **]**
* **tweets.py [**methods for collecting tweets from the **api ]**
* **user.py [** methods for collecting data for a user **]**

**DatabaseModule>**

* **database\_manipulation.py [**methods for retrieving and storing data in mongo database **]**

**Facebook**

**Twitter**

* twitter\_data\_manipulation.py [ has wrapper function that provides specific indexes for twitter data ]

#### ***Hard-coded parts***

-In main.py

– names of database [“twitter”, “facebook”]

-names of users / lists(slugs) [“community-councils”] , WORLD\_WOE\_ID

-dynamically creating mongo collections out of the names of facebook pages, list members

-In CrawlingModule.Twitter > user.py, tweets.py, list.py

-name of database/collection

-In AnalysisModule.Twitter > tweets.py , graph.py and AnalysisModule.Facebook>pages.py

-name of database/collection

#### JSON data returned from APIs example

* Twitter:  ***functions :*** [***https://dev.twitter.com/docs/api/1.1***](https://dev.twitter.com/docs/api/1.1)

SEARCH Api – tweets: <https://dev.twitter.com/docs/api/1.1/get/search/tweets>

* Facebook: ***functions:*** [**https://developers.facebook.com/docs/graph-api/reference/v2.1**](https://developers.facebook.com/docs/graph-api/reference/v2.1)

Open graph- posts : <https://developers.facebook.com/docs/graph-api/reference/v2.1/page/feed/>

## Package components and installation

Here are all the libraries and components used throughout the development process, which may be useful in the future and are required for the package to run.

Required components:

* [Python current version : 2.7.5](#_Python_version_:)
* [Pip](#_Pip)
* [Tweepy library (Crawling/authorization sub module easier to handle authorization)](#_Libraries)
* [Twitter library (Crawling Module/authorization sub module)](#_Libraries)
* [Networkx(Analysis module)](#_Libraries)
* [Setuptools](#_Libraries)
* [Facebook-sdk](#_Libraries)
* [Prettytable](#_Libraries)
* [Pymongo](#_Libraries)
* [Mongodb](#_MongoDB)

For development:

* [Git](#_Git)
* [PyCharm](#_PyCharm)

For Graph Analysis:

* [Gephi](#_Gephi)

## App registration

To be able to use the api of twitter or facebook firstly you should create a twitter/facebook account or use an existing one and register your app here:

-twitter: <https://apps.twitter.com/>

-facebook: <https://developers.facebook.com/apps>

And then write the consumer key and secret in the config.py file

# Installation Appendix

### Python version : 2.7.5

<https://www.python.org/download/>

*Possible problems:* (Windows) “python” not recognised as a command

*Solution:* You need to add the python executable path to your Window's PATH variable.

<http://stackoverflow.com/questions/7054424/python-not-recognised-as-a-command>

### Pip

1)Download :  [get-pip.py](https://bootstrap.pypa.io/get-pip.py) (Save as: )

2)From cmd/terminal go to the folder where your get-pip.py file is and follow the instructions from <http://pip.readthedocs.org/en/latest/installing.html>

### *Libraries*

**pip install name\_of\_library** ( eq.pymongo,tweepy,twitter …)

pip installs setup tools when it’s installed <http://pip.readthedocs.org/en/latest/installing.html> . Follow the link to see how to upgrade the setup tools.

### *MongoDB*

For installation of MongoDB follow the link <http://docs.mongodb.org/manual/installation/>

\*Helpful step by step tutorial with screenshots for windows <http://stackoverflow.com/questions/2404742/how-to-install-mongodb-on-windows>

In order to use mongo via the shell you need to

1) establish a tunnel from one cmd/terminal window by going to mongo’s installation folder usually “c:/Program Files/ MongoDB/bin” and run **mongod**

2)And open another window were you go to the same folder and instead run **mongo** (exe).

For working with mongo via the package/scripts just do 1)

### *Git*

<http://git-scm.com/downloads>

For editing the Readme.md file useful tool - https://stackedit.io/#

How to use git: <http://git-scm.com/book/en/Git-Basics-Getting-a-Git-Repository>

1. Use cmd/terminal; go to the new project’s folder.
2. When you’re inside the project run **git init.**
3. To add new changes use: **git add .** (with the dot) then run **git commit –m “must type a message”** and finally run git push url\_link\_to\_project\_on\_github (eg. **git push** [**https://github.com/SnezanaDichevska/twitterAnalyzer**](https://github.com/SnezanaDichevska/twitterAnalyzer))
4. To download a project from git for the first time use: **git clone** [**https://github.com/SnezanaDichevska/twitterAnalyzer**](https://github.com/SnezanaDichevska/twitterAnalyzer)
5. To download changes from project locally use: **git pull** [**https://github.com/SnezanaDichevska/twitterAnalyzer**](https://github.com/SnezanaDichevska/twitterAnalyzer)
6. If there are conflicts with the commits or pull commands use **git reset –hard HEAD**

for going back few versions. Be careful when trying to commit changes because it can erase everything you’ve changed so far.

### *PyCharm*

Community Edition <http://www.jetbrains.com/pycharm/download/>

### *Gephi*

<http://gephi.github.io/users/download/>

Gephi tutorial - <http://www.kateto.net/wordpress/wp-content/uploads/2012/12/COMM645%20-%20Gephi%20Handout.pdf>

If trying to create labels (eg. name of user) you must go to Data Laboritory > Copy Data to other column > save

# ISSUES

* socket.error (104 'connection reset by peer') :: set to sleep for few sec <http://stackoverflow.com/questions/383738/104-connection-reset-by-peer-socket-error-or-when-does-closing-a-socket-resu>
* pycharm – set python interpreter :: <http://www.jetbrains.com/pycharm/quickstart/configuring_interpreter.html>
* pip command not recognized :: <http://stackoverflow.com/questions/23708898/pip-is-not-recognized-as-an-internal-or-external-command-installing-django-w>
* importing modules :: must create \_\_init\_\_.py files in every package/directory
* mongo corrupted files , not running, unexpected shutdown :: go to /path/mongo/bin run mongod –-dbpath path/to/data –reapir

more info : <http://docs.mongodb.org/manual/tutorial/recover-data-following-unexpected-shutdown/>

* xlrd only saves files into xls format

# USEFULL COMMANDS in Mongo

(MONGO QUERY ) find which tweets were written in a reply:

gb.getCollection("community-councils").find( {

"in\_reply\_to\_user\_id" : {

$ne:null

}

}, {

\_id:0, "user.screen\_name":1,  "in\_reply\_to\_screen\_name":1,  "text":1

} ).pretty()

**//**return **user's screen name** ; **the screen name of the user to whom you replied**; and the **text** of the status -> for each status who's **in\_reply\_user\_id** is not null !

 (MONGO QUERY ) find which tweets were written in a reply [for different users]:

gb.getCollection("community-councils").find( {

"in\_reply\_to\_user\_id" : {$ne:null },

$where: "this.user.id !=   this.in\_reply\_to\_user\_id"

}, {

\_id:0, "user.screen\_name":1, "in\_reply\_to\_screen\_name":1,  "text":1

} ).pretty()

**//**return **user's screen name**; **the screen name of the user to whom you replied**; and the **text** of the status -> for each status who's **in\_reply\_user\_id** is not null AND take in two consideration only conversation between different users!

"**this.**user.id !=   **this.**in\_reply\_to\_user\_id"

**MUST INCLUDE this** for comparison of two fields from mongo

**//**useful for selecting Dates - <http://docs.mongodb.org/manual/reference/operator/query/regex/>

<http://stackoverflow.com/questions/3305561/how-to-query-mongodb-with-like>

# Future work

### Structure

Currently the modules are divided firstly by their function and then after the Social Network their meant for. Since they are separated in the package, at the moment there shouldn’t be any confusions about which module belongs to which sub package , but it may be worth considering in the future to create separate packages for Facebook and Twitter and then divide them into Crawling, Analysis and Database module. This way the packages would be smaller and easier to maintain.

### Interface

Creating a fully functional interface that would call the package in the background corresponding to the social network in interest to collect data and then save it/ export it, analyze it etc. would be very useful and would make the package available to the users who do not have a programming background or are not familiar with python or any of the tools included in the package.

### Decisions

For now in the DatabaseModule the appearance of duplicate keys when storing data is handled in a way that the new piece of data is not stored so that when collecting data additional operations for storing repeated data are not being done. But it might be worth considering to create another update function that would occasionally update the information of the stored data, since data is changing with time.