*Journal of Multimedia and Information System VOL. 2, NO. 3, September 2015(pp. 287-294 ): ISSN 2383-7632(Online)*
http://dx.doi.org/10.9717/JMIS.2015.2.3.287

# Twitter Crawling System

Saydiolim Ganiev[1], Aziz Nasridinov[2] , and Jeong-Yong Byun[3,*]

## Abstract

We are living in epoch of information when Internet touches all aspects of our lives. Therefore, it provides a plenty of services each of which benefits people in different ways. Electronic Mail (E-mail), File Transfer Protocol (FTP), Voice/Video Communication, Search Engines are bright examples of Internet services. Between them Social Network Services (SNS) continuously gain its popularity over the past years. Most popular SNSs like Facebook, Weibo and Twitter generate millions of data every minute. Twitter is one of SNS which allows its users post short instant messages. They, 100 million, posted 340 million tweets per day (2012)[1]. Often big amount of data contains lots of noisy data which can be defined as uninteresting and unclassifiable data. However, researchers can take advantage of such huge information in order to analyze and extract meaningful and interesting features. The way to collect SNS data as well as tweets is handled by crawlers. Twitter crawler has recently emerged as a great tool to crawl Twitter data as well as tweets. In this project, we develop Twitter Crawler system which enables us to extract Twitter data. We implemented our system in Java language along with MySQL. We use Twitter4J which is a java library for communicating with Twitter API. The application, first, connects to Twitter API, then retrieves tweets, and stores them into database. We also develop crawling strategies to efficiently extract tweets in terms of time and amount.

**Key Words:** Twitter, Crawler, SNS.

## I. INTRODUCTION

Recently Internet has become one of the most important parts of people's lives. Nowadays more and more people use it because internet provides a lot of services which make human life easier and funnier. Of all services social networks continuously gain its popularity over the past years. Social network service (SNS) is a service in which people can form relationships with, talk to, communicate with and express their thoughts, images and video to each other. SNS also a

llows users to share ideas, activities, events, and interests within their individual network like friends or friends of friends. Twitter is SNS which provides a service of broadcasting short burst messages. It can also be defined as a website which allows discovering interesting people and following them or their burst messages. Twitter obtains 302 million active users as of May 2015. Its 100 million users posted 340 million tweets per day (2012). Twitter also handled 1.6 billion search queries per day. Such big information generated by Twitter users can be target for analyzing. Since now many researches have been done to extract interesting pattern from tweets.

In order to get any data from SNS we are required to use crawlers which are designed to look for particular information users want to find. Generally, web crawlers come into someone's mind when crawlers are mentioned. Web crawlers methodically visit web site, read visible textual information, tags and hyperlinks, and then build indexing for the data people are looking for. The main purpose of web crawlers is to collect data so that users can quickly get relevant web sites when they enter a search term. However, crawlers in context of SNS are quite different. Instead of crawling all pages available, SNS offer using their API to deal with content or data developers are expecting to extract from SNS. Twitter also require developers to use Twitter API by using OAuth which guarantees secure connection and keeping user privacy safe. By changing its policy in 2013 Twitter also set a list of limits to what kind of data and how many of data can be extracted. Thus, Twitter API has become a set of rules designed by Twitter in order to control usage of its tweets by developers.

In this paper, we develop Twitter Crawler system which enables us to extract and store Twitter data. We implement our system in Java along with MySQL. The application first extracts Twitter data after connecting to Twitter API, and then obtained Twitter data is stored into MySQL database. We also develop crawling strategies to efficiently extract tweets in terms of time and amount.

Manuscript received September 20, 2015; Revised October 2, 2015; Accepted October 12, 2015. (ID No. JMIS-2015-31)
Corresponding Author (*): Saydiolim Ganiev, 123, Dongdae-ro, Gyeongju-si, Gyeongsangbuk-do, Korea, Dongguk University Gyeongju Campus, +82-54-770-2114, byunjy@dongguk.ac.kr
[1]Computer Science and Multimedia Engineering, Dongguk University, Gyeonju, saydiolim@dongguk.ac.kr.
[2]Computer Science and Multimedia Engineering, Dongguk University, Gyeonju, aziz@dongguk.ac.kr

More precisely, we make the following contributions:

- As name suggests, the main purpose of our system is crawling Twitter data, especially tweets, and store them in database for the further use. We developed Twitter Crawler which searches for tweets by using a user input keyword. We utilized well-known java library for Twitter API called Twitter4J in order to handle Twitter data. This library enables us to set connection to Twitter API and extracts tweets from Twitter which match search criterion. Retrieved tweets along with other Twitter data are stored in MySQL database so that the user can conduct analysis on them.
- In order to deal with Twitter API restrictions we develop crawling strategies. They help our system process more efficiently in terms of time and tweets extracting capacity.
- We demonstrate the performance of our developed system by providing screenshots.

The rest of the report is organized as follows. We review the related works in Section 2. We describe the proposed method in detail in Section 3. We present implementation of the proposed system in Section 4. Section 5 summaries and concludes the paper.

## II. RELATED STUDY

Many studies have been conducted during past years on SNS data analysis. Those researches provide various results regarding importance, interestingness and efficiency. However, most of them revealed different aspects of SNS as a major part of Internet in people's daily life. In this section, we present studies aiming at exploiting.

Zhixeng Xu et al. [2] proposed a new framework exploiting modified author-topic model, namely twitter-user model to find interesting topics. Twitter users usually publish noisy tweets about their lives, which prevent from discovering proper topics of interests. For each tweet authors applied extended (retrofit) author-topic model by latent variable in order to discover if it is related to author's interest. User's interest includes two parts: original interest and retweeted interest. Experimental results demonstrate that their approach outperforms traditional approaches in terms of discovering user interests on Twitter.

Xinyue Wang et al. [3] proposed an adaptive crawler model to automatically capture popular-event hashtags. While crawling tweets based on predefined search terms, namely keyword, the proposed model identifies popular event-related hashtags applying Simple Keyword Adaptation (SkwA) algorithm or Refined Keyword Adaptation (RKwA) algorithm. SKwA algorithm captures and holds new hashtags within a particular period of time. If after time frame, a number of hashtags are bigger than threshold, the identified hashtags are placed into a list of keywords. This process repeats each time frame allowing new hashtags to be added into a search term list. Results show that the proposed method allows collecting more relevant tweets and reduce amount of irrelevant ones.

Kim Younghoon et al. [4] proposed a probabilistic algorithm to recommend tweets and friends to Twitter users. Authors apply MapReduce to present parallelized algorithm to handle big data that includes tweets and relationships between the users. Expectation Maximization (EM) algorithm also is built to learn parameters of the proposed generative model. Authors also present the ranking algorithms to recommend top-K followees and top-K tweets to users. Experimental results demonstrate TWITOBI outperforms content based recommendation methods in terms of scalability and effectiveness.

Min-Chul Yang et al. [5] proposed a novel statistical and unsupervised model called Trend Sensitive-Latent Dirichlet Allocation (TS-LDA) to recommend interesting tweets to users. First, they exploit LDA to find latent topics and then its modified version to uncover trends in Twitter. In content-based proposed method, each potentially interesting tweet is indicated by its interestingness measure that represents how interesting a tweet might be to users. Thus, if a tweet possesses interestingness score higher than threshold they are identified as an interesting tweet. Experimental results show that the proposed method outperforms several traditional methods in tweet interestingness prediction and tweet classification.

Melike Yigit et al. [6] proposed two topology based recommendation systems that consider not only user relationships but also their actions and mentions. In Extending topology based recommendation algorithm user relationships are classified in four depths. Each depth represents followers of previous followers group. In extending Friends of Friends (FOF) algorithm, the same steps must be taken with only three depths. Using users score measure in the chain of followers are recommended to the target user only if they are not already followers of him. Experimental results show that the proposed algorithms outperform graph-based and Conceptual Fuzzy Set based algorithms in terms of best precision.

Hassan Saif et al. [7] proposed an approach called SentiCircles to analyze tweets for sentiment detection. SentiCircles is lexicon-based approach which builds a dynamic representation of words to discover their semantics.This enables to update their sentiment strength and polarity in a particular lexicon. In order to find word's contextual semantics, they apply distributional hypothesis in which words appear in similar contexts are likely to have similar meanings. The proposed method allows detecting sentiments at both entity level and tweet level. Experiments are conducted with three tweet datasets and using different sentiment words lists. Authors declare their approach outperforms previous entity-level approaches and perform better than SentiStrength at tweet-level.

Luca Cagliero et al. [8] presented a new data analysis system called TFC Analyzer (Twitter Flipping Correlation Analyzer) to capture interesting events such as topic trend analysis, context-aware service profiling and outlier detection. By analyzing tweets TFC Analyzer supports to uncover frequent itemsets in order to discover contrasting and potentially interesting features. Their main focus is on

comparing itemsets at different abstraction levels. Therefore, they obtain a new pattern, the Strong Flipping Generalized Itemset (SFGI) from tweets which consist of generalized itemset X and its descendants showing correlation change regarding itemset X. They apply LCM-based (Linear Time Closed Itemset Miner-based) itemset mining algorithm along with ad-hoc post-pruning phase. Experiments have been conducted on both real and synthetic dataset, and its result shows the proposed method is effective in discovering interesting patterns.

## III. PROPOSED SYSTEM

The proposed system is built to crawl, store, analyze data, and, if needed, to make recommendation or alert on topic we will work on. Our proposed system consists of three major parts: Twitter crawler, database and alert levels modeling

First step is to construct Twitter Crawler. Firstly, our application gets connection to Twitter to extract data. When we retrieve Twitter data from Twitter, a crawling strategy must be considered carefully. The crawling strategy represents the way how to efficiently deal with Twitter rate limits. Once data is extracted, it is stored in database for further analysis. In the second step, we build alert level modeling. In context of our system, alert modeling represents suicide-related tweet detection. Firstly, we list up all significant words or phrases which can be related to suicide. Then we set weights to all those words in order to rank tweets while analyzing them.

### 3.1. Twitter Crawler

In this subsection, we talk about extracting and storing Twitter data. Specifically, we will overview the overall design of our system, present crawling strategy, describe constructed database and explain the proposed method in details.

The design of the system is shown in Figure 1. It describes the overall process of the performance of our system. From the Figure 1, we can see that application in the system plays main role on manipulating flow of data.

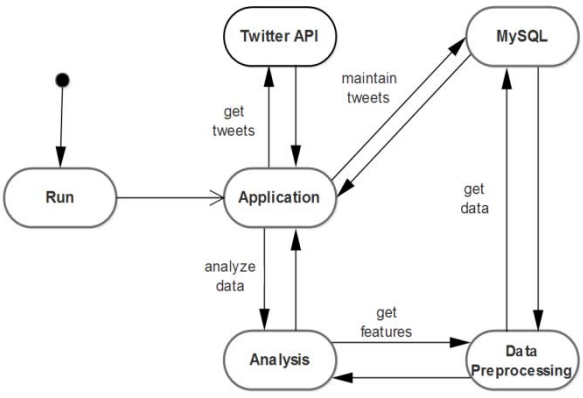


Fig. 1. The overall system design.

Sequence diagram depicted in Figure 2 shows the flow of performance of our system in details. The process starts when an application sends message to Twitter API in order to get tweets. In response, Twitter sends back json files that stores tweets in human-readable way. Tweets are stored in database going through the application since they need to be converted into sql-readable state. In the next step, we analyze tweets. Because tweets include not only interesting patterns but also needless data, they have to be preprocessed. During the preprocessing step, we remove unnecessary words, numbers and symbols from tweets.

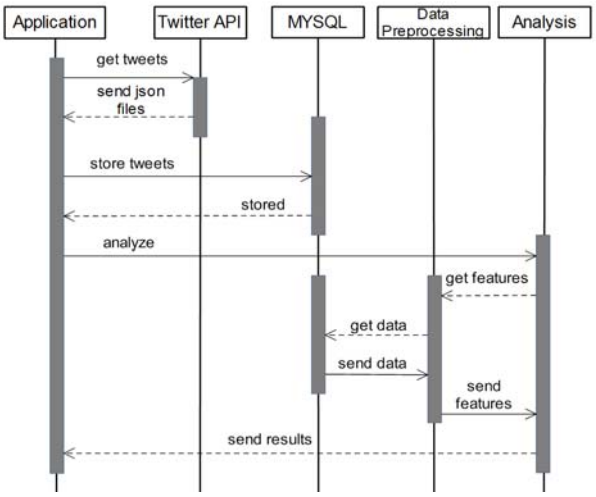


Fig.. 2. The workflow sequence diagram.

### 3.2. Crawling Strategy.

Twitter sets variety of rate limits in order to control third-party applications while they are acting on behalf of users to extract or manage Twitter data. In this section we explain our strategy to deal with Twitter limitations [9].

Firstly, Twitter rate limits include user and application authentications which are a way of allowing third-party applications to access user data without exposing his password. OAuth is a security protocol for Web applications. Twitter allows you to access private data through OAuth as an alternative to standard HTTP Authentication. Secondly, one of the important features of Twitter rate limits is the number of request one can do every 15 minutes. Further, we talk about those two types of authentications and provide ways to handle them.

#### 3.2.1.User-auth.

Twitter supports two different types of authentication. The first type is user authentication which allows an application to act on behalf of a user, as the user. This type of authentication is used when an application need to create or edit status tweets. Alike application-auth, user-auth are not limited to activities the application can perform on behalf of the user. We can see how our application obtains user credentials to perform for the sake

of the user as shown in Figure 3.

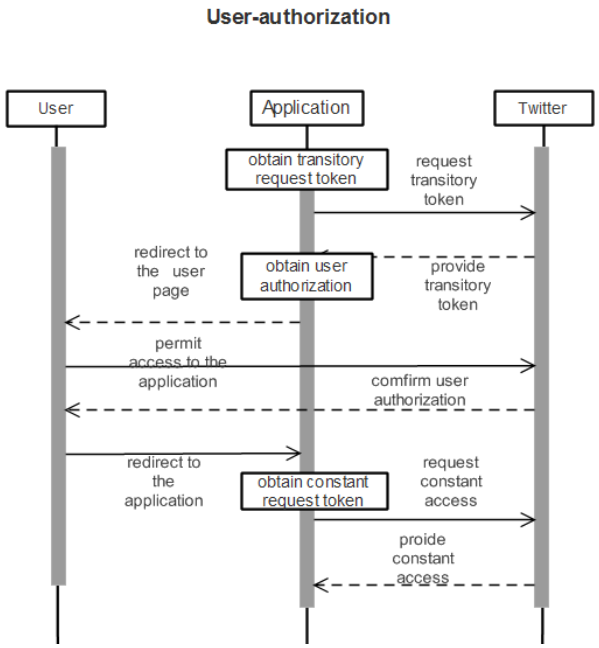


Fig. 3. User authorization.

In order to crawling Twitter data, we need to provide our credentials in terms of application tokens to Twitter API OAuth. Thus, Twitter prohibits accessing public Twitter data without authentication at all. To deal with first limit related to user-auth, we suggest the following two strategies:

- Obviously, one of the simplest but efficient ways of crawling as much data as we want is to create more Twitter user accounts, which does not seem difficult. The problem may arise when we need them to operate concurrently. Since while managing crawling a lot of user accounts as well as properly redirecting data to database tables, one can meet serious issues. However, with high constructing of a logical flow of the system we can manage this issue to get more tweets concurrently.
- The second strategy is related both to user-auth and application-auth. Since time plays critical role on both limits. Therefore, it is very important thing when we deal with Twitter API. As mentioned before, Twitter segmented time into 15 minutes intervals. Thus, it is vital to scatter requests within the given time interval so that the distance between each request can be equal. In previous example, request limits for user is 180 and the limit for application is 450. Let's take user rate limit. If someone spends all 180 requests in, say, 10 minutes, it means, first, he has to wait 5 minutes doing no crawling until next 15 minutes windows opens. Secondly, he probably extracts Twitter data he extracted during 10 minutes using only some number of 180 requests. This might mean other requests are used for just nothing. Therefore, it is critical issue to spread all request uniformly in order for the system to perform efficiently.

### 3.2.2. Application-auth.

The second type of authentication is application-only authentication. In this type of authentication, an application does not need to fully act as a user. Therefore, to login in providing user's password and logins is not required. This type of access is very appeal to such developers that look for Twitter public data such as crawling tweets.

- Our last strategy is associated with application-auth. We need one more credentials to gain application-auth. The last credential is called bearer token which is needed for Twitter to allow our application to get limited access to the user activities but with more requests. Once actions from Figure 3 are done, we combine key and secret tokens together and encode it with base64 encoding. New encoded key is sent to Twitter to get a bearer token. Twitter provides a bearer token which is used to get application-auth. Note that Twitter allows developers to create more than one application per user. Considering that we build as many applications as we need. Then we set application-auth to each application. Each application comes with 450 requests per window, for example. 10 applications means 4500 request. Application-auth gives us such opportunity. Also, rate limits of user-auth and application-auth do not overlap. They do not impact on each other to reduce number of requests.
- Since application-auth is also restricted by a number of requests per time window, applying the same strategy proposed previously is efficient. Constantly observing requests given from Twitter API is very critical. Therefore, we develop and add automatic time adjusting function to our system. Thus, this enables us to set a number of request performed by a specific time interval, which allows us to crawl more tweets.

Those strategies proposed are keys for the system to perform more efficiently. Thus, this allows us to deal with Twitter rate's limits and retrieve more tweets as we expected.

### 3.3. Database Design

We describe database to store Twitter data. We create MySQL database to save retrieved tweets. Figure 4 shows the database scheme in which tweets and analysis results is stored. Proposed database consists of 6 tables: AdminInfo, Keyword, UserInfo, Tweets, Analysis and TweetCategory. Each table possesses its unique name that indicates what kind of information it stores. We review each table and its attributes in detail.

AdminInfo table. This table is designed for storing customer information. It includes 3 attributes: admin_id, twitter_ap_name, server_name.

Keyword table. The table is constructed for storing keyword information. Keyword table includes 4 attributes: keyword_id, input_word, type and language.

UserInfo table. As name suggests, UserInfo holds information about Twitter user account. It is critical to have such kind of data since it is essential part of social

data analysis. UserInfo table is comprised of 4 attributes: userinfo_id, screen_name, location and language. Tweets table. This table plays the major role in database. Tweets table stores nearly all interesting information

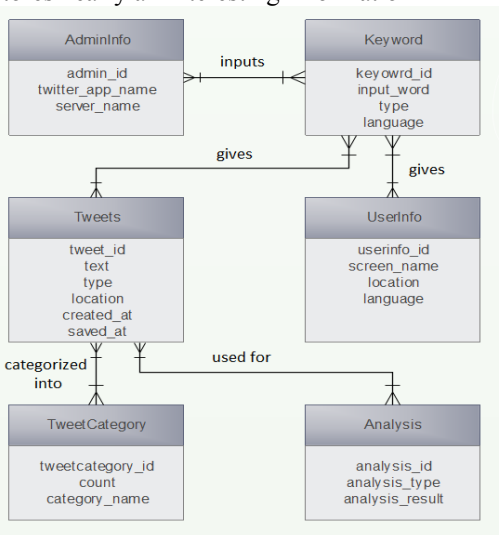


Fig.4.Database E-R diagram

important for data analysis. It consists of 6 attributes: tweets_id, text, type, location, created_at, and saved_at.

**Analysis table**. The table is designed as storage for output data which come out after tweets are analyzed. This table is the most important one in terms of data analysis since it stores only meaningful data as opposed to previous tables which store raw data. The table consists of 3 attributes: analysis_id, anaylsis_type, and analysis_result.

**TweetCategory table**. This table can be seen as supplement to Tweets table. However, this table contains a few important features. It includes 3 attributes: tweetcategory_id, count and category_name.

## IV. IMPLEMENTATION

In this section, we explain how we implement our system in detail. We present system functionalities with figures and descriptions. Figure 4 demonstrates the main window of java GUI application. The application of the parts which represent three windows in the application.

1. Crawl Tweets. By clicking this button a user is re-directed to Crawling Window. As the name suggests, in this window, we are able to extract tweets and its related data from Twitter. Search for tweets can be done by adding keyword. The application returns only those tweets in which words match with keyword.

2. Manage Tweets. This button represents Manage Window in which a user can deal with tweets stored in a database. The user will be offered with several functions such as getting tweets from database by keyword and username, delete tweets by keyword and download all tweets as a one csv file.

### 4.1. Crawl Tweets

In Figure 6, we can see Crawling Window. The window enables a user to search for tweets over Twitter. In order to get tweets, the user needs type the word he wants

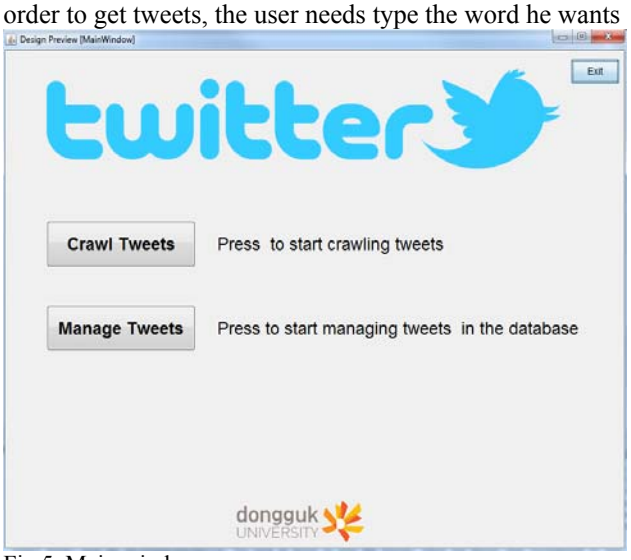


Fig.5. Main window

to find tweets with in the text field. Next, he presses get button in order to start crawling process. The application returns at most 100 tweets with a particular keyword user added. We implement our system in such way that no duplicate tweets are allowed. In addition, in order to efficiently use requests given by Twitter API in 15 minutes window, we set timer which run program in particular period of time automatically. This gives us two benefits: no need to manually managing tweet retrieving process and avoiding wasting requests for duplicate tweets. Moreover, we add text area to display all tweets which are extracting from Twitter. This gives use visualization to observe crawling process.

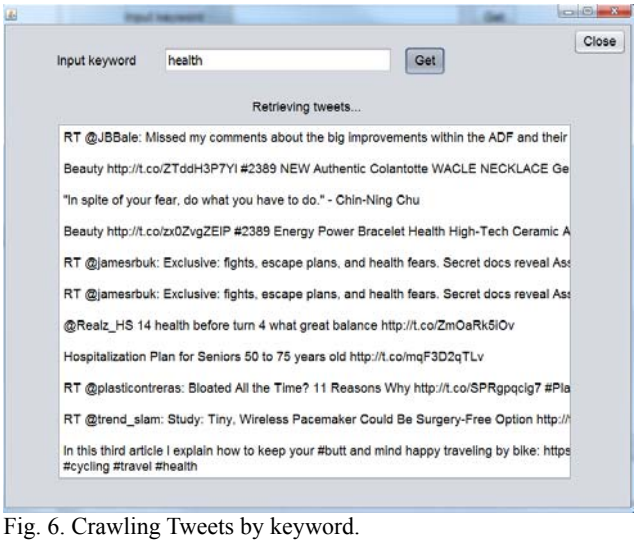


Fig. 6. Crawling Tweets by keyword.

### 4.2. Manage Tweets

Figures 7-8 shows the functionalities of managing Window. Managing Window consists of 4 parts: extracting tweets from a database by keyword and username,

deleting tweets by keyword and downloading database as a csv file.

Figure 7 demonstrates retrieving tweets stored in a database searching by a keyword. In order to get tweets the user is interested in, he needs to put keyword or username on text field. If there is keyword or username similar to input words, the application returns tweets and display them in the text area.
Usually tweets extracted are more than they can fit one text area. In that situation, the user can scroll down with mouse to see all tweets.

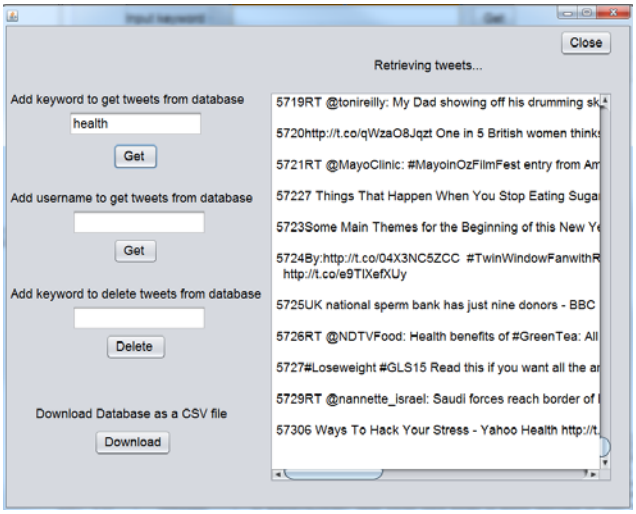

Fig 7. Extracting tweets from database by keyword.

The next functionality for managing tweets is shown in Figure 8. A user can delete unnecessary tweets stored in a database by adding keyword. This function is efficient since tweets can become out of date. Therefore, removing such tweets makes database simpler and saves its capacity. In order to eliminate unwanted tweets the user are asked to input keyword. The application reads keyword, deletes all tweets with this keyword and displays them in the text area. Additionally, pop-up window is shown to inform that the process has done successfully.

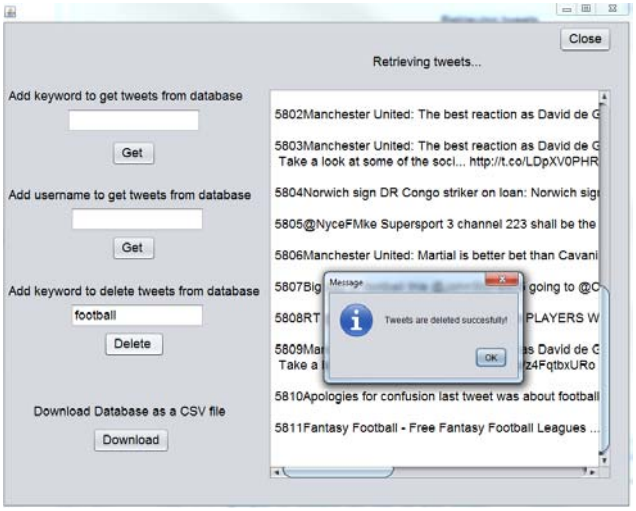

Fig. 8. Deleting tweets from a database by keyword.

## V. CONCLUSION

In this paper, we developed Twitter Crawler application. We implemented the application on java exploiting along with MySQL. With Java we built Graphic User Interface (GUI) which includes several functionalities. MySQL was used to construct a database for tweets to be stored. In our example, we constructed twitter database with tweets table. This table contains 4 columns: tweeted, username, text and keyword.

The java application consists of 3 windows: main and 2 inner windows. Main Window serves as an introduction to inner windows. From main windows we can visit 2 other windows: Crawling Window and Managing Window. In Crawling Window, we were asked to input a keyword to search for corresponding tweets which match search requirements. The results were visualized in the text area. Managing windows includes lots of functionalities: Retrieve tweets from a database by keyword and username, delete tweets from a database by keyword and download all tweets as a csv file on a computer. The user was offered to extract tweets stored in database by using keyword and usernames. The application returns queried tweets into text area. Deleting tweets is available by adding keyword. Once keyword is sent to the system, it removes tweets from database and displays them in the text area in order to show what tweets have been deleted. The last function is the whole database as one csv file. The path for saving csv file is set by default and can be changed any time when the user wants.

## REFERENCES

[1] https://en.wikipedia.org/ , 2015
[2] Z. Xu, R. Lu., L. Xiang and Q. Yang, "Discovering User Interest on Twitter with a Modified Author-Topic Model," *IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology,* pp. 422-429, 2011.
[3] X. Wang, L. Tokarchuk, F. Cuadrado and S. Poslad, " Exploiting Hashtags for Adaptive Microblog Crawling," *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining,* pp. 311-315, 2013.
[4] Y. Kim and K. Shim, "TWITOBI: A Recommendation System for Twitter Using Probabilistic Modeling," *11th IEEE International Conference on Data Mining*, pp. 340-349, 2011.
[5] M. Yang and H. Rim, "Identifying interesting Twitter contents using topical analysis," *Expert Systems with Applications,* Vol. 41, pp. 4330-4336, 2014.
[6] M. Yigit, B. Bilgin and A. Karahoca, "Extended topology based recommendation system for unidirectional social networks," *Expert Systems with Applications,* Vol. 42, pp. 3653-3661, 2015.
[7] S. Saif, Y. He, Z. Fernandez and H. Alani, " Contextual semantics for sentiment analysis of Twitter," *Information Processing and Management,* 2015.

[8] L. Cagliero, T. Cerquitelli, P. Garza and Grimaudo, " Twitter data analysis by means of Strong Flipping Generalized Itemsets," *Journal of Systems and Software,* Vol. 94, pp. 16-29, 2014.
[9] https://dev.twitter.com/rest/public/rate-limits, 2015

## Authors

**Saydiolim Ganiev**
Saydiolim Ganiev received his B.S. degree in Computer Science from Tashkent University of Information Technology, Tashkent, Uzbekistan, in 2011. He is currently a M.S. degree student of the Dept. of Computer Engineering in Dongguk University. His research interests include Database Management Systems(DBMS), Intrusion Detection Systems (IDS) and machine learning techniques.

**Aziz Nasridinov**

Aziz Nasridinov received his B.S. degree in Computer Science from Tashkent University of Information Technology, and his M.S. and Ph.D. degrees in Computer Engineering from Dongguk University, South Korea. His research interests include Database Management Systems (DBMS), machine-learning techniques and Web Services.

**Jeong-Yong Byun**
Jeong-Yong Byun received the B.S. and M.S. degrees from Dongguk University, Seoul, Korea in 1980 and 1983, respectively, and the Ph.D. degree from Hongik University, Seoul, in 1994, all in computer science. From 1982 to 1987, he was with Electronic and Telecommunication Research Institute (ETRI), Daejon, Korea where he was involved in the development of UNIX systems. Since 1988, he has been with faculty of computer science and multimedia engineering of Dongguk University at Gyeonju. He had been visiting academy (by Post Doctor program of KOSEF) at University of York in England between 1995 and 1996. His researches have been concentrated on Korean alphabet according to Hunminjeongeum, Database Management Systems (DBMS) and Web Services.

This is blank Page