Project Part 2 - Group 6

Team: Nika Shafranov, Bryce Strickland, Ahmed Almutawa, Loic Guegan, Rishab Kanwal

Title: Quitter - (Query Twitter)

Project Summary: Our application will allows us to store, view, and analyze twitter data based on a variety of topics and hashtags centering around politics. The display will be some kind of chart that shows the frequency of tweets. We will allow users to create accounts as well as input search terms through a search bar that will return the data mentioned in the previous sentence.

Project Requirements:

Business Requirements

ID	Requirement	Priority
BR-01	Twitter account must be associated with a user's login.	Critical
BR-02	Administrative accounts should be created so that admins can tweak parts of the system using a simple interface	High
BR-03	Daily data aggregations must be done between 2 and 4 pm.	Medium
BR-04	API must be hosted on Heroku and the batch jobs must run on AWS EC2 instances	Critical
BR-05	Weekly data aggregations will be done between 8 and 10 pm	Medium
BR-06	Amazon instances will be monitored daily in order to not exceed budget	Medium
BR-07	Amazon instances must be shut down when they are not being used	Critical

User Requirements

ID	Requirement	Priority
US-01	A user should be able to enter a search term or hashtag and then get twitter statistics	Critical
US-02	A user should be able to click a bar in the chart and then a table should appear containing individual tweets	High
US-03	An administrative user should be able to change or add hashtags and search terms to be queried on	Medium
US-04	Users should be able to export aggregate statistics in some image format after running a query	Medium

US-05	Users should be able to export aggregate statistics and individual tweets as a CSV	Medium
US-06	A user should be able to request hashtags and handles to be queried for the next day	Medium
US-07	An admin should get a notification to approve hashtags to be able to approve them for processing	Medium
US-08	A user should be able to post a tweet containing a statistic chart to Twitter from the application	Low
US-09	A user should be able to favorite and retweet individual tweets displayed in our application	Medium
US-10	A user should be able to see a personalized Tweet Analysis	High

Functional Requirements

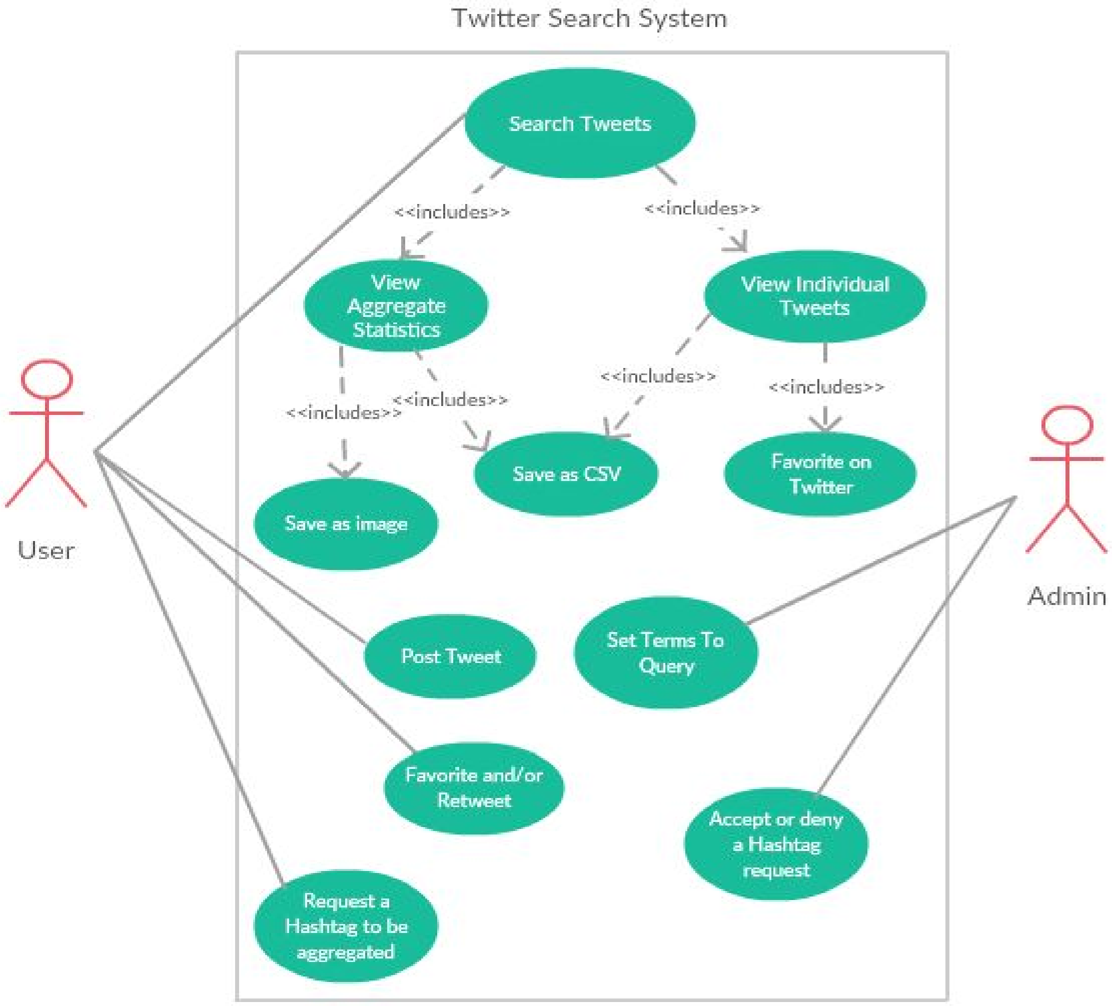
ID	Requirement	Priority
FR-01	When a User clicks the search button a query will aggregate data on the entered search terms, hashtags, or handles and populate a chart	Critical
FR-02	When a user clicks on a bar in a chart a table will become visible that contains relevant tweets sorted by a default attribute	High
FR-03	When a user logs into the landing page a query will return daily and weekly hashtags and display them by popularity	Medium
FR-04	When a user logs a query will return daily and weekly aggregated tweet statistics and the application will display it using charts	Medium
FR-05	When a user changes the “sort by” attribute in the table the table will query the data and display the most relevant tweets to the constraint	Medium
FR-06	When a user clicks “favorite tweet” an association will be made between the user and that tweet	Low
FR-06		

Non-Functional Requirements

ID	Requirement	Priority
NFR-01	Performance: Upon searching on a term results should be returned within 10 seconds and displayed in a table format	High

NFR-02	Usability: On our landing page there will be a brief explanation of how to use our system and what our system displays	Critical
NFR-03	Reliability: The system will be resistant to failure by having a second database instance that will serve as a backup. If the first instance goes down the application will know to point at the second database.	Low
NFR-04	Reliability: If the user attempts to query the database and both databases are down the application will print a message informing the user of the error and telling them to come back later. The message will give a contact email	High
NFR-05	Platform Constraints: If we pursue Android login will be through twitter in both platforms (Web and Android)	Low
NFR-06	Supportability: An API will be maintained for the application so that it can be easily ported to multiple platforms	Medium

Use Cases:
Actors: Admin, User



Use Case Documents:

Use case id: UC-01

Use case Name: Twitter Sign up

Description: User signs up for Quitter using Twitter

Actors: User

Pre conditions: User has a Twitter Account

Post conditions: User has a Quitter login

Frequency of Use: Once per user

Flow of Events:

Actor Action	System Response
1. Click Sign up with Twitter	System creates a dialog that makes a Twitter login popup
2. User logins into Twitter with valid credentials	Twitter provides an OAuth token to login, our

	system saves the user account to our system

Variations: User does not have a Twitter, Twitter authentication server is not responding

Use case id: UC-02
Use case Name: Tweet List Search
Description: User selects the “tweets list” view and searches for a hashtag or handle
Actors: User
Pre conditions: User has logged in
Post conditions: User finds the tweets he is looking for.
Frequency of Use: Throughout the day by users
Flow of Events:

Actor Action	**System Response**
1. User selects “tweets list” and enters a search query	System searches top tweets for hashtag and handle matches and delivers the top 20 tweets
2. User gets a list of top 20 results and selects a tweet to view	The full tweet is shown

Variations: No matching tweets are found

Use case id: UC-03
Use case Name: Tweet Statistic Search
Description: User selects the “statistics” view and searches for a hashtag or handle
Actors: User
Pre conditions: User has logged in
Post conditions: User finds the statistics for a keyword he is looking for.
Frequency of Use: Throughout the day by users
Flow of Events:

Actor Action	**System Response**
1. User enters search query	System searches the database for statistics data based on hashtags and displays a chart depicting the data.
2. User views a chart depicting the statistics related to a particular hashtag or handle	The chart has buttons that allow it to be sorted on certain attributes.
3. User clicks on a chart in the graph	The system queries the tweets that composed that bar in the graph and displays a table of

	tweets

Variations: No matching tweet statistics were found

Use case id: UC-04
Use case Name: Twitter stats for a day
Description: User wants to view a Twitter statistics for a particular day
Actors: User
Pre conditions: User has logged in
Post conditions: User can see the Twitter statistics for the day
Frequency of Use: Throughout the day by users(Default landing page so whenever someone logs in)
Flow of Events:

Actor Action	**System Response**
1. User logs in or clicks the summary page.	System provides pre aggregated daily statistics for the previous day
2. User selects an older day or week to view statistics for that day or stays on current page	System shows pre aggregated statistics for the selected time.
3. User clicks a statistic to see more detail	System provides more detailed statistics and aggregates

Variations: There is no data from the previous day

Use case id: UC-05
Use case Name: Request Hashtag
Description: User requests the hashtags to be analyzed to the day
Actors: User
Pre conditions: User has logged in and wants to view a particular hashtags statistics
Post conditions: User sees twitter statistics for the requested hashtag or handle the next day
Frequency of Use: Occasionally by users
Flow of Events:

Actor Action	**System Response**
1. User clicks on the Request Tweet Statistics button	System provides a text box for tweets to be analyzed
2.User enters the tweets he wants data for.	System sends a request to the admin for the tweets to be analyzed

3. Admin approves the request	System analyzes the data
4. The clicks the Requested tweets buttons the next day and sees analyzed data from them	The system displays the analyzed data.

Variations: There is no data for the tweets requested, the admin does not approve analysis of the requested terms(because of profanity or illegal terms)

Use case id: UC-06
Use case Name: Admin notified of query request
Description: Admin reviews the hashtags that users have requested to be queried for the day
Actors: Admin
Pre conditions: Admin has logged in
Post conditions: Data aggregator has new search terms to query on
Frequency of Use: A few times throughout the day by an administrator
Flow of Events:

Actor Action	**System Response**
1. Admin logs in and goes to the requested hashtags page.	System provides a list of hashtag requests that have not been reviewed yet by an admin.
2. Admin accepts or denies hashtags until the requested list is empty	System updates aggregator program to query for accepted hashtags. A notification is sent to the User who requested the hashtag

Variations: An Admin accepting or denying a request is not properly processed by the system

Use case id: UC-07
Use case Name: Sort-by feature
Description: User chooses from options: sort by date, time, popularity
Actors: User
Pre conditions: User has logged in
Post conditions: User sees the information in the requested order
Frequency of Use: Multiple times a day
Flow of Events:

Actor Action	**System Response**
1. User searches for a hashtag or keyword	System provides a list of results matching the search term (automatically sorted by popularity)

2. User chooses from dropdown menu how the search results are presented	System updates search results in order chosen by the User.

Variations: There is no data to be presented.

Use case id: UC-08
Use case Name: Personal Tweet Analysis
Description: User can view personal stats and tweet analysis
Actors: User
Pre conditions: User has logged in
Post conditions: User sees his personal tweet report
Frequency of Use: Multiple times a day
Flow of Events:

Actor Action	**System Response**
1. User clicks personal analysis button	System cross references a user's tweets with top and trending tweets and computes suggestions on what he should tweet more and less about
2. User views the data and clicks a certain hashtag for more information	The system provides a more detailed description

Variations:The use has no Tweets, the user's tweets do not match top or trending data

Use case id: UC-09
Use case Name: Post Tweet
Description: Users can post tweets containing statistics views from our application
Actors: User
Pre conditions: User has logged in and queried statistics for a keyword
Post conditions: User gets a dialog reporting the status of posting their tweet
Frequency of Use: Multiple times a day
Flow of Events:

Actor Action	**System Response**
1. User queries statistics view using a hashtag tag or handle	The system queries statistics for the keyword a displays a chart of the statistics
2. User posts a tweet from our application containing an image of our statistics graph and a comment	System backend posts a tweet using the twitter API and outputs a dialog indicating success or failure of the tweet posting.

Variations: The user wants to post a tweet from our application without an image

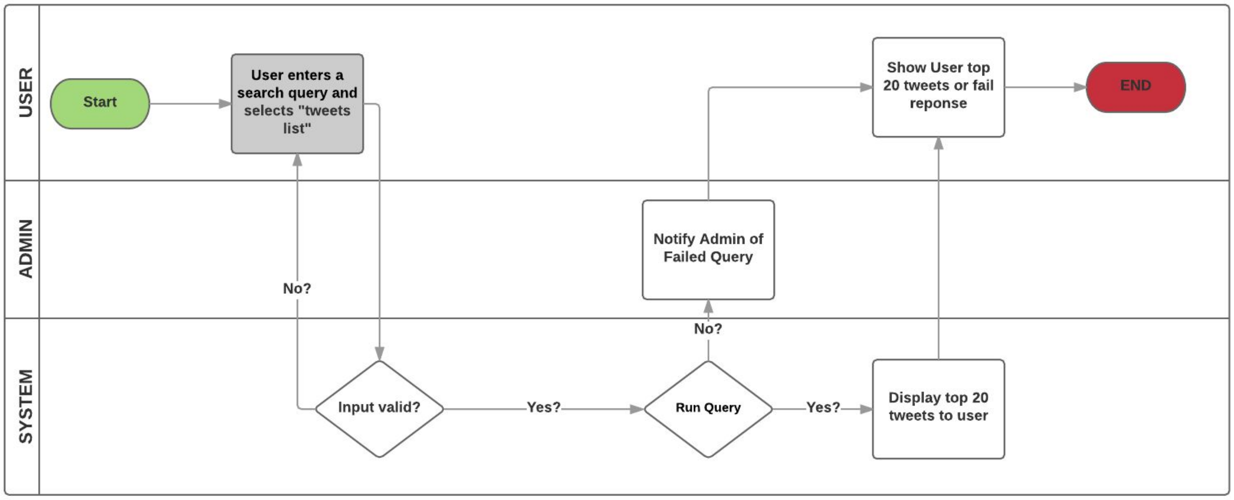
Use case id: UC-10
Use case Name: Favorite or retweet selected tweet
Description: Users can favorite or retweet a tweet from within the UI
Actors: User
Pre conditions: User has logged in
Post conditions: User has a tweet favorited or retweeted from his twitter account
Frequency of Use: Multiple times a day
Flow of Events:

Actor Action	**System Response**
1. User queries tweets using a keyword	The system returns a list of tweets based on the keyword entered
2. User sees a Tweet he likes and decides to favorite and retweet it, he clicks the favorite or retweet button	The system interacts with the Twitter API and pushes the update to Twitter. The system shows a dialog indicating if the action succeeded or failed
3. User clicks the Tweet favorited/retweeted notification	User is forwarded to Twitter and the notification is highlighted

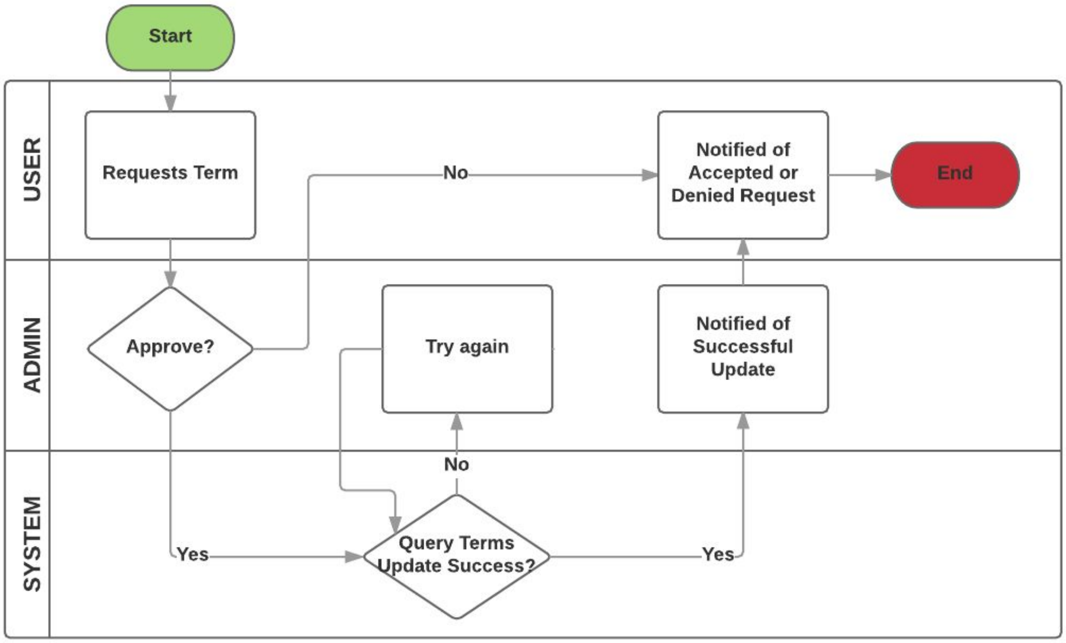
Variations: Twitter API is down or authentication token expired.

Activity Diagrams:

Requirement ID: US-01
Use Case ID: UC-02
Use Case Name: Tweet List Search
Implemented by: Ahmed Almutawa

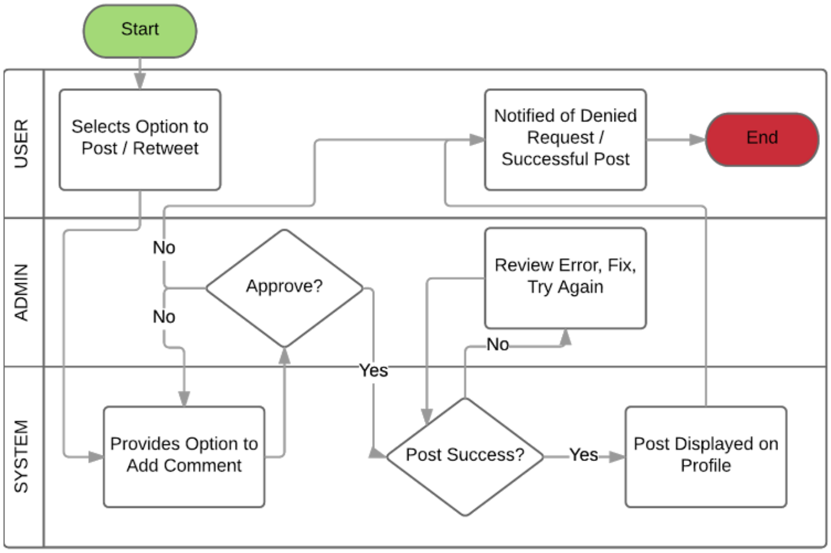

USER
ADMIN
SYSTEM
Start
User enters a search query and selects "tweets list"
No?
Input valid?
Yes?
Run Query
No?
Notify Admin of Failed Query
Yes?
Display top 20 tweets to user
Show User top 20 tweets or fail reponse
END

Requirement ID: US-06
Use Case ID: UC-05 and UC-06
Use Case Name: User Requests Term to be Queried
Implemented By: Loic Guegan

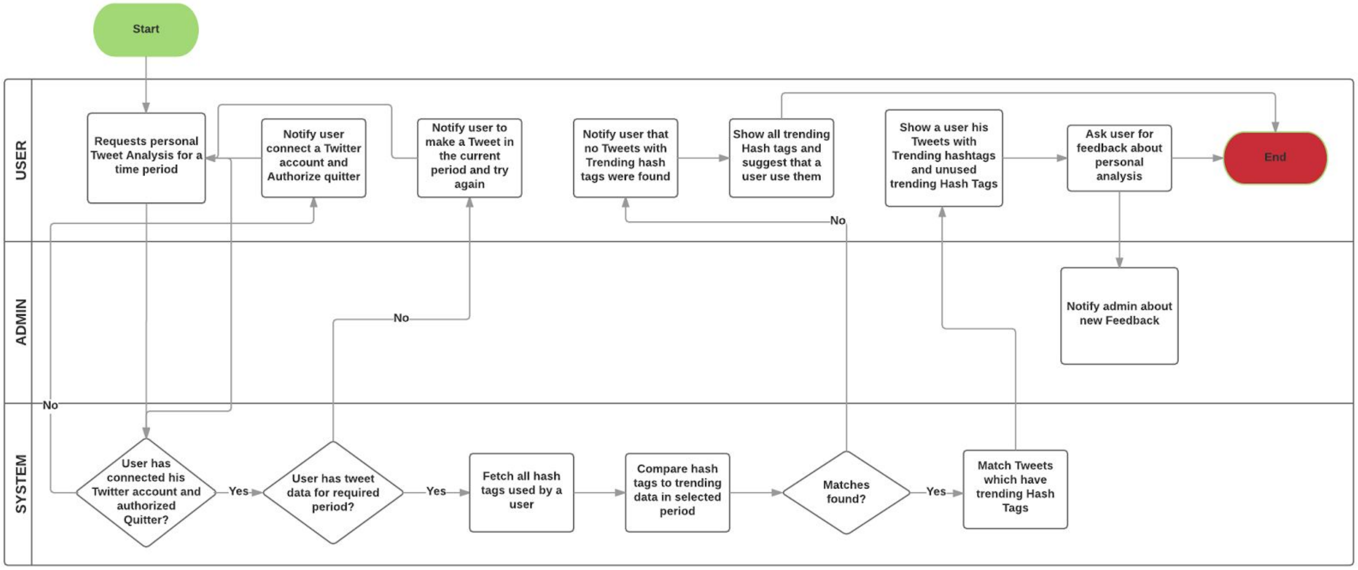

Start
USER
ADMIN
SYSTEM
Requests Term
Approve?
No
Notified of Accepted or Denied Request
End
Try again
Notified of Successful Update
No
Yes
Query Terms Update Success?
Yes

Requirement ID: US-09

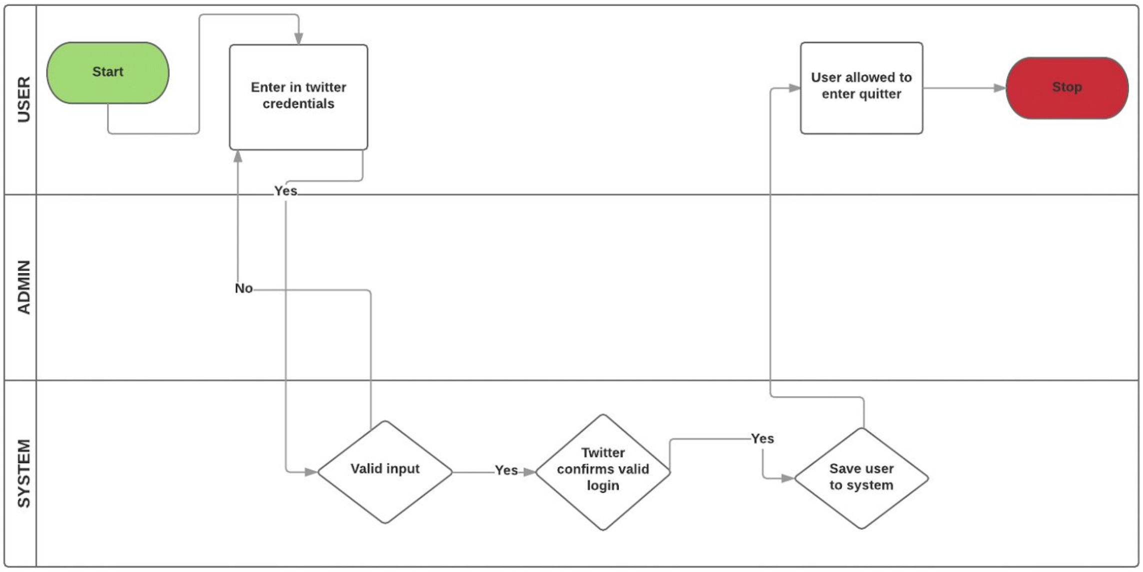
Use Case ID: UC-09 and UC-10
Use Case Name: Post Tweet
Implemented By: Nika Shafranov


Start
USER
Selects Option to Post / Retweet
Notified of Denied Request / Successful Post
End
ADMIN
No
Approve?
No
Review Error, Fix, Try Again
No
Yes
SYSTEM
Provides Option to Add Comment
Post Success?
Yes
Post Displayed on Profile

Requirement ID: US-10
Use Case ID: UC-08
Use Case Name: Personal Tweet Analysis
Implemented By: Rishab Kanwal


Start
USER
Requests personal Tweet Analysis for a time period
Notify user connect a Twitter account and Authorize quitter
Notify user to make a Tweet in the current period and try again
Notify user that no Tweets with Trending hash tags were found
Show all trending Hash tags and suggest that a user use them
Show a user his Tweets with Trending hashtags and unused trending Hash Tags
Ask user for feedback about personal analysis
End
No
ADMIN
No
Notify admin about new Feedback
No
SYSTEM
User has connected his Twitter account and authorized Quitter?
Yes
User has tweet data for required period?
Yes
Fetch all hash tags used by a user
Compare hash tags to trending data in selected period
Matches found?
Yes
Match Tweets which have trending Hash Tags

Requirement ID: US-01
Use Case: UC-01
Use Case Name: Sign up for Quitter
Implemented By: Bryce Strickland


USER
ADMIN
SYSTEM
Start
Enter in twitter credentials
Yes
No
User allowed to enter quitter
Stop
Valid input
Yes
Twitter confirms valid login
Yes
Save user to system

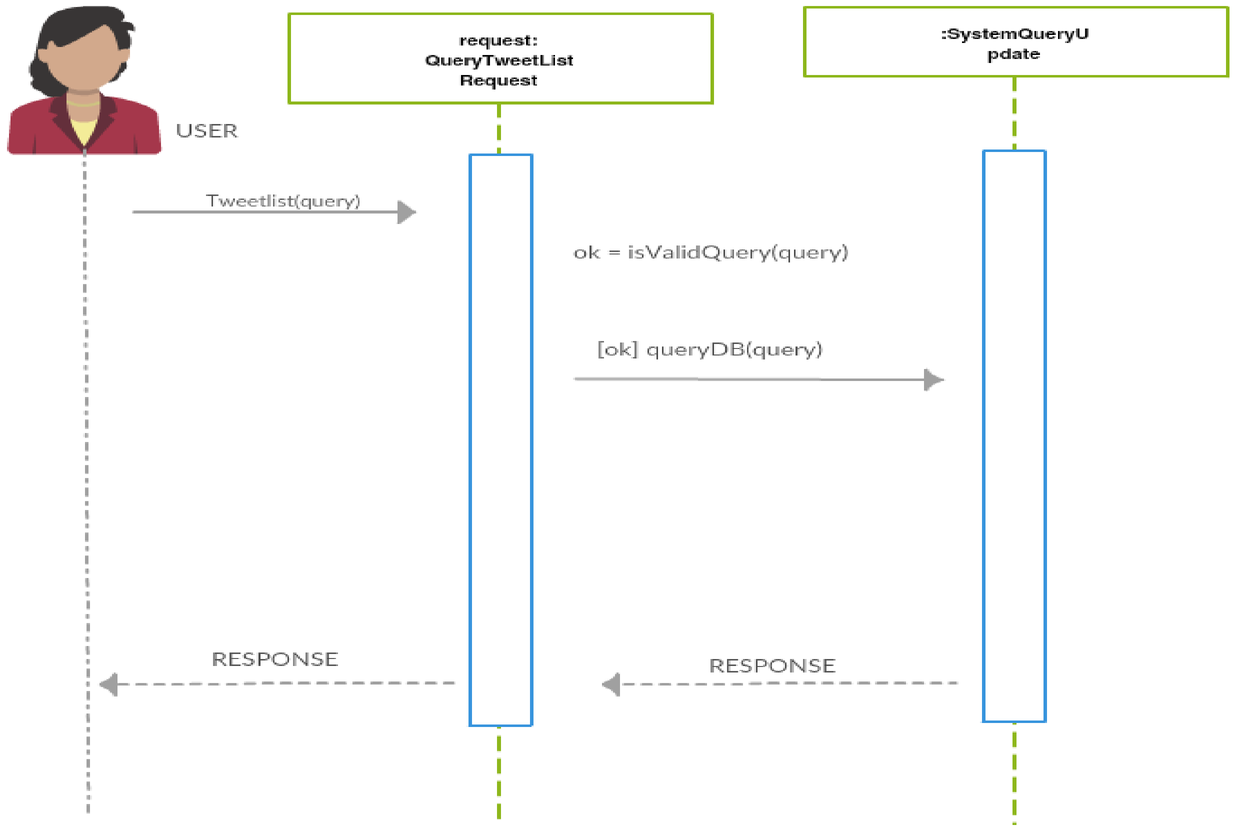
Data Storage: Cassandra data store(Column Family store) with a Django ORM to convert Django Objects.

Classes:

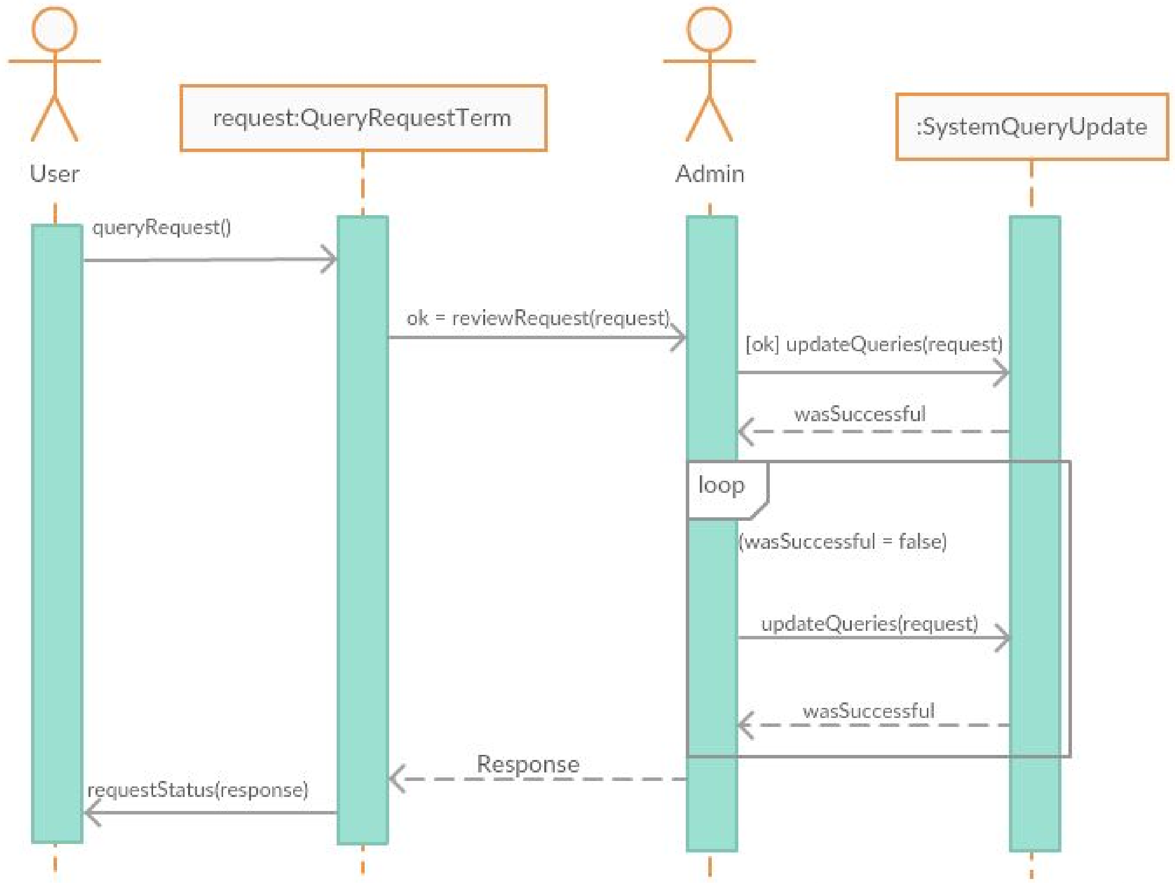
- Tweet class to store information about individual tweets
- Twitter User class to store information on individual users
- User class to store information on users who create accounts on our system
- Statistics class that will allow store statistics based on hashtag and date collected

UI Mockups:
https://docs.google.com/presentation/d/1ZDi0aF1nq1fXgI2qEGrNecyQtFyQhRZZYQZppVZZ9ME/edit?usp=sharing

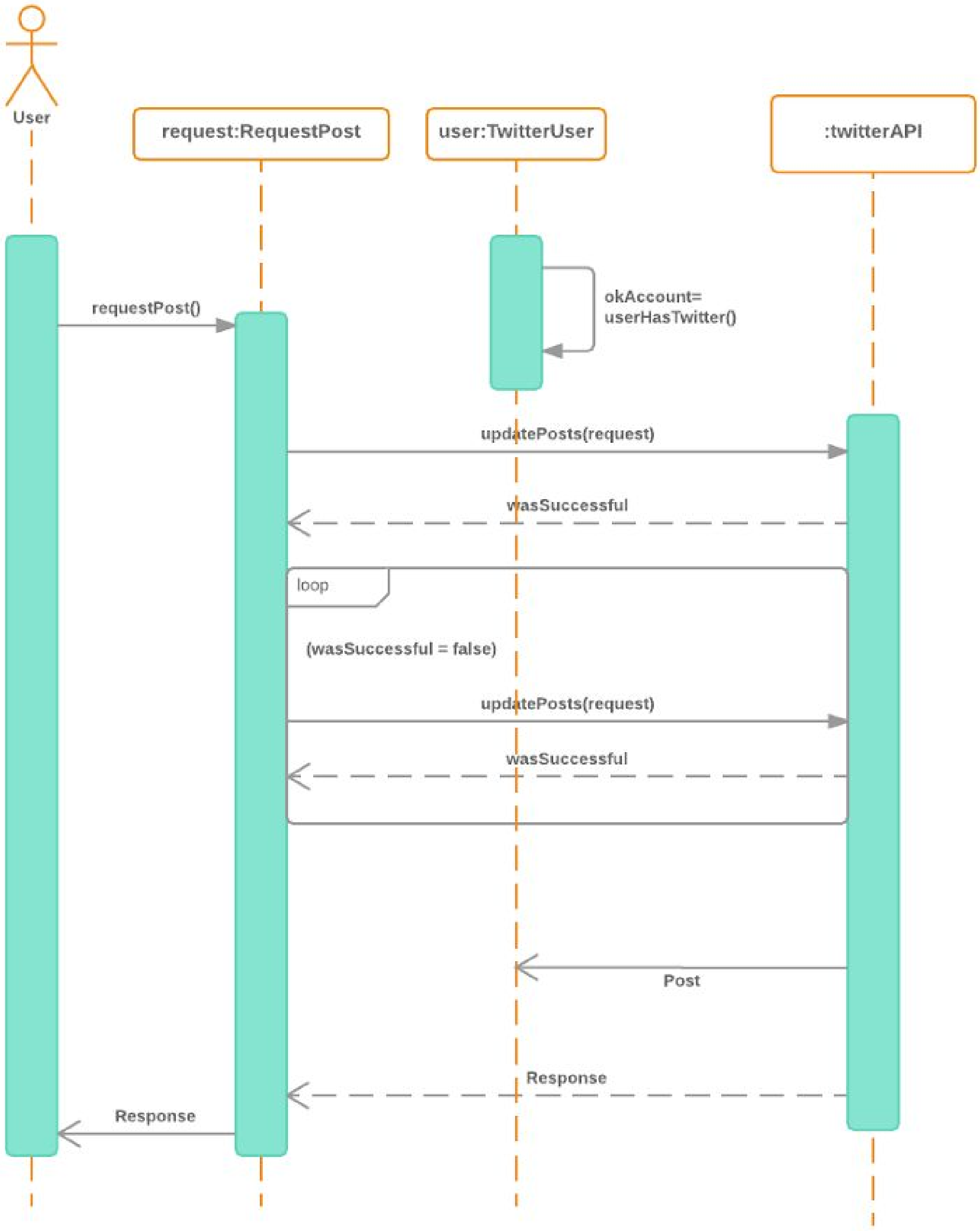
User Interactions:
Requirement ID: US-01
Use Case ID: UC-02
Use Case Name: Tweet List Search
Implemented by: Ahmed Almutawa



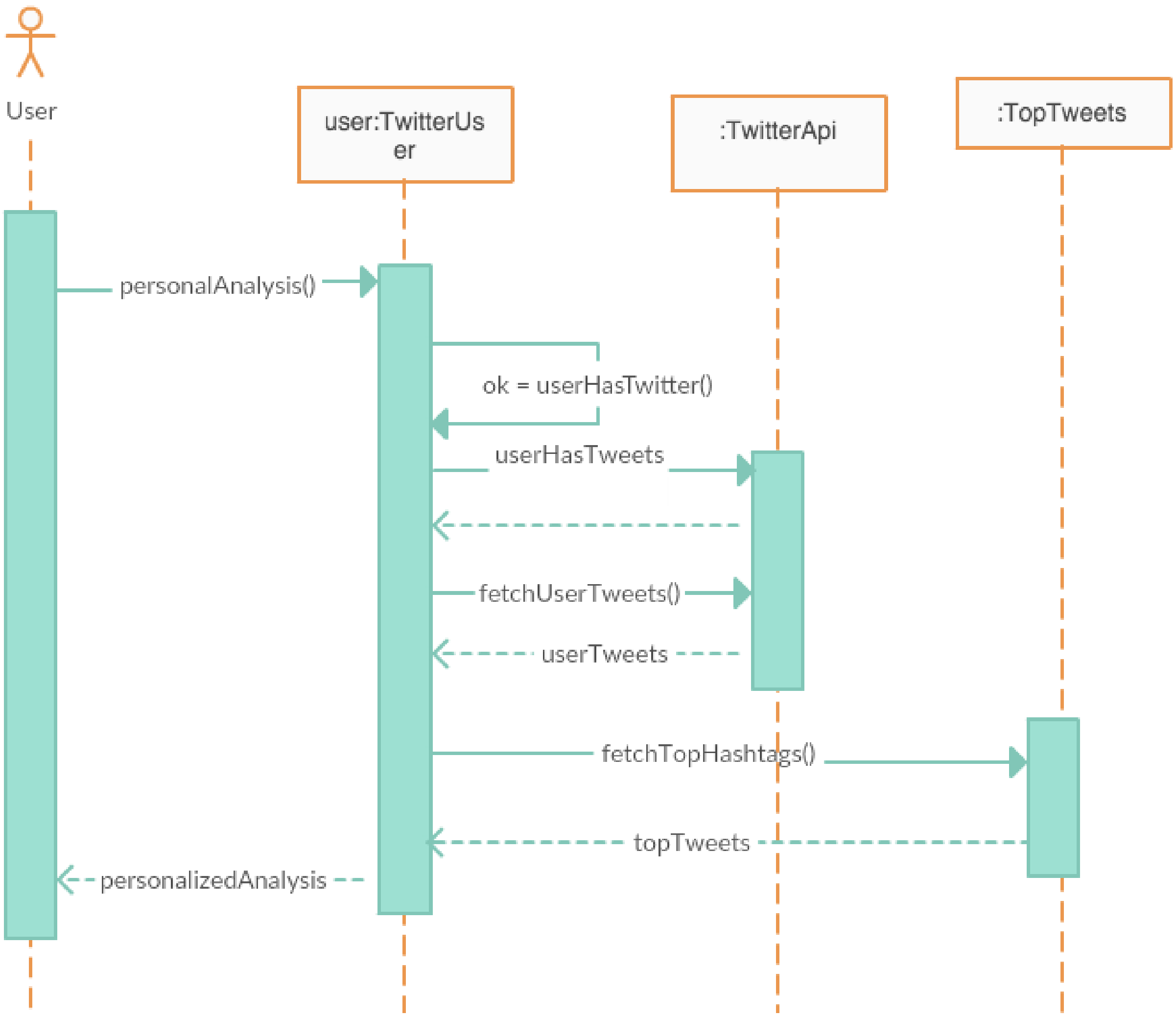
Requirement ID: US-06
Use Case ID: UC-05 and UC-06
Use Case Name: User Requests Term to be Queried (Success Updating terms)
Implemented By: Loic Guegan



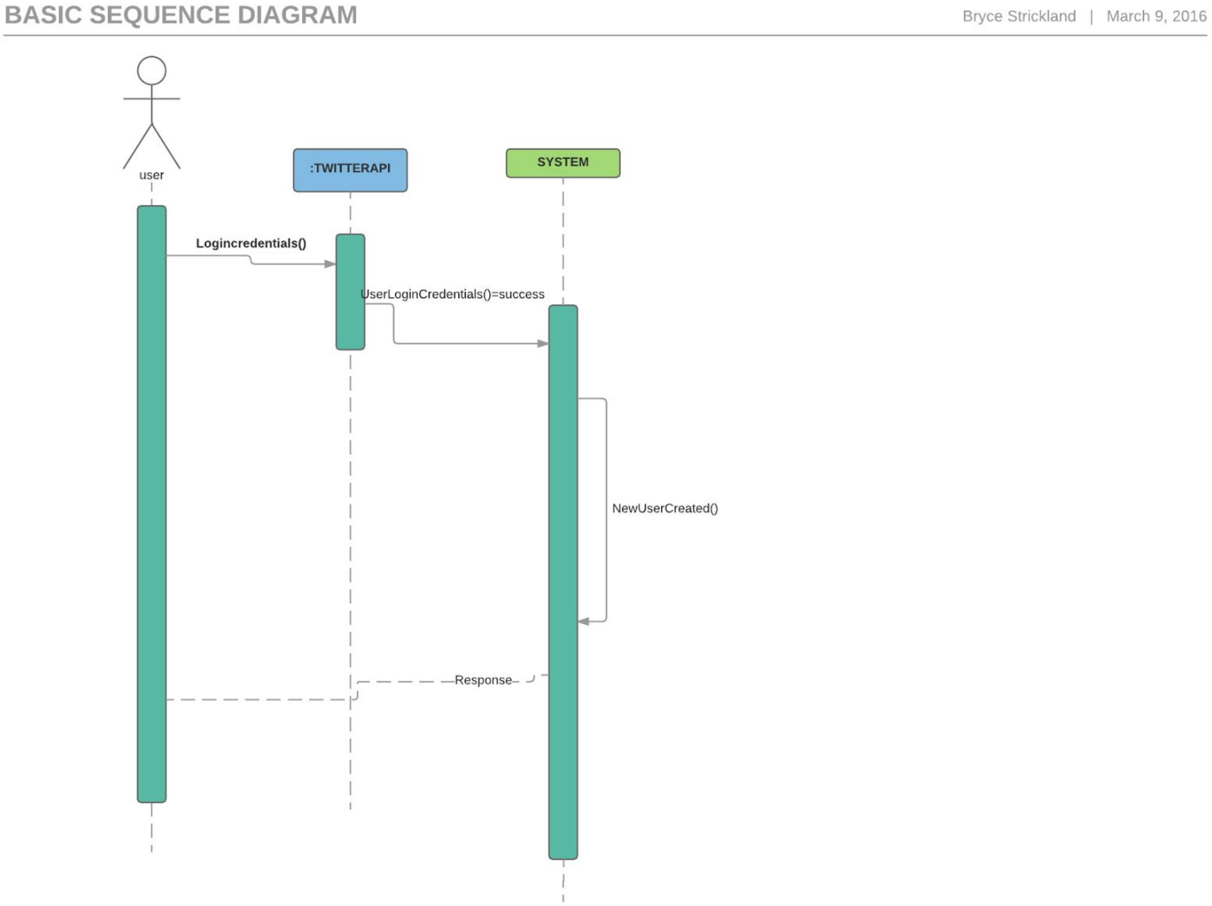
Requirement ID: US-09
Use Case ID: UC-09 and UC-10
Use Case Name: Post Tweet
Implemented By: Nika Shafranov



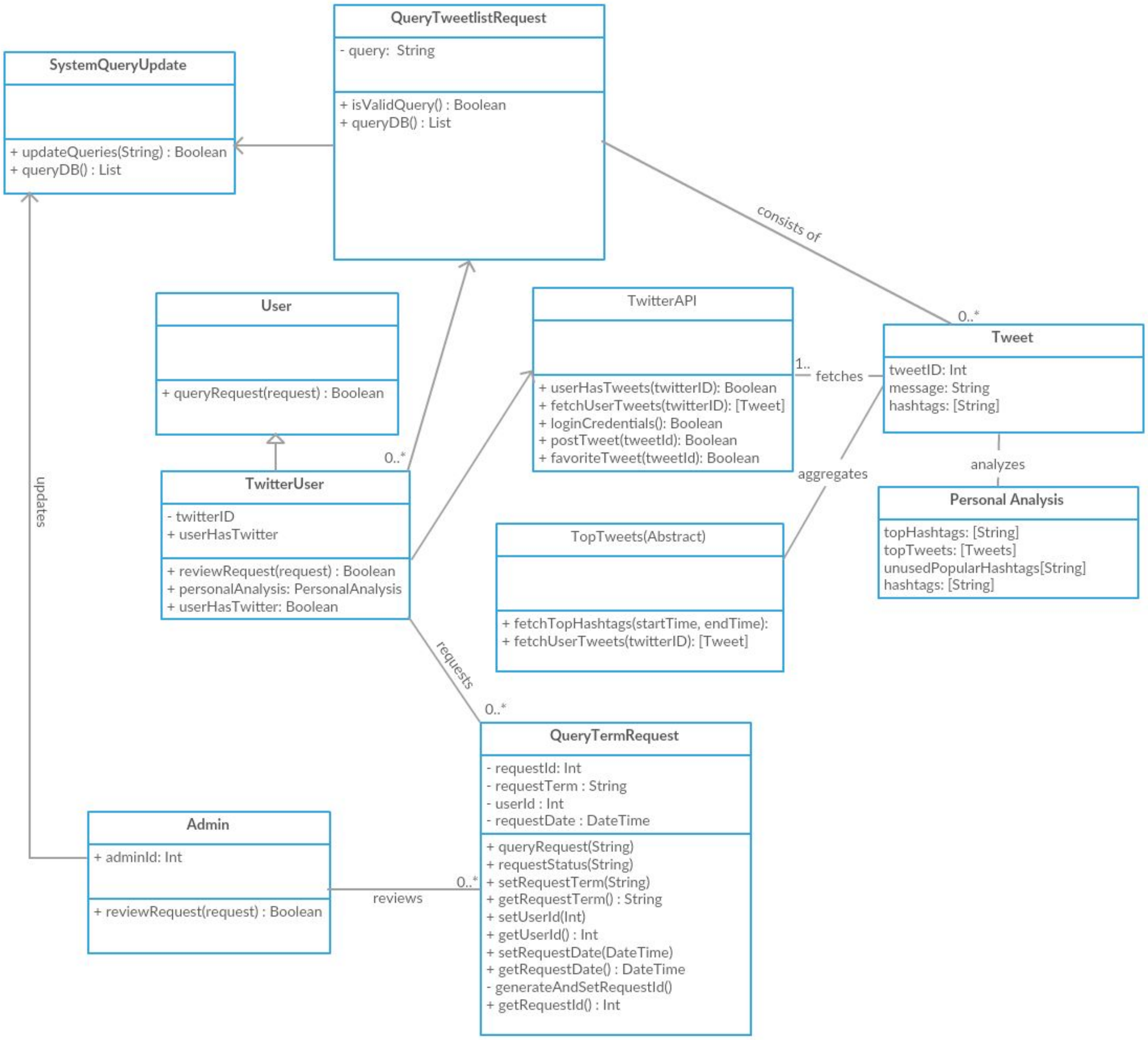
Requirement ID: US-10
Use Case ID: UC-08
Use Case Name: Personal Tweet Analysis
Implemented By: Rishab Kanwal



Requirement ID: US-01
Use Case: UC-01
Use Case Name: Sign up for Quitter
Implemented By: Bryce Strickland



Class Diagram:


SystemQueryUpdate
+ updateQueries(String) : Boolean
+ queryDB() : List
QueryTweetlistRequest
- query: String
+ isValidQuery() : Boolean
+ queryDB() : List
consists of
0..*
User
+ queryRequest(request) : Boolean
TwitterAPI
+ userHasTweets(twitterID): Boolean
+ fetchUserTweets(twitterID): [Tweet]
+ loginCredentials(): Boolean
+ postTweet(tweetId): Boolean
+ favoriteTweet(tweetId): Boolean
1..
fetches
Tweet
tweetID: Int
message: String
hashtags: [String]
analyzes
aggregates
updates
0..*
TwitterUser
- twitterID
+ userHasTwitter
+ reviewRequest(request) : Boolean
+ personalAnalysis: PersonalAnalysis
+ userHasTwitter: Boolean
Personal Analysis
topHashtags: [String]
topTweets: [Tweets]
unusedPopularHashtags[String]
hashtags: [String]
TopTweets(Abstract)
+ fetchTopHashtags(startTime, endTime):
+ fetchUserTweets(twitterID): [Tweet]
requests
0..*
QueryTermRequest
- requestId: Int
- requestTerm : String
- userId : Int
- requestDate : DateTime
+ queryRequest(String)
+ requestStatus(String)
+ setRequestTerm(String)
+ getRequestTerm() : String
+ setUserId(Int)
+ getUserId() : Int
+ setRequestDate(DateTime)
+ getRequestDate() : DateTime
- generateAndSetRequestId()
+ getRequestId() : Int
Admin
+ adminId: Int
+ reviewRequest(request) : Boolean
reviews
0..*