Project Part IV: Final Report

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

Title: Quitter / Twittical

Questions:

1. **What features were implemented?**

   From our use cases, we implemented the following features (modified for our political focus):

   Tweet List Search - Top presidential candidates are selectable, a user can select one of the candidates from the top view and view the 50 latest tweets for the candidate.

   Tweet Statistic Search - Top presidential candidates are selectable. A user can see stats for a particular candidate or one or more of them combined

   Twitter Stats for a Day - Stats are aggregated on an hourly basis, so we did better than planned!

   Sort-by/Filtering - A user can get stats for particular date time ranges in more detail by clicking the date time range toggles on the top of our graphs.

   Favorite or Retweet Selected Tweet - User can favorite or retweet a tweet from within the UI. System interacts with the Twitter API and pushes the update to Twitter. The system shows a dialog indicating if the action succeeded or failed. User is forwarded to Twitter and request is processed.

   Public REST API - Users can query our public facing REST API so they can use our aggregated data. Link to API endpoints

   Sentiment Analysis - We implemented sentiment analysis on collected tweets to analyze how positive or negative the tweet was

   Graphs showing sentiment over time - Users can see average positive and negative sentiments in tweets of the top presidential candidates.

   Which features were not implemented from Part 2?

   Our project changed focus after part 2 and hence we implement some features from part 2, the following are some of the features that weren't implemented in our final product:

   Request Hashtag - user requests the hashtags (candidate) to be analyzed for the day. System provides a text box for tweets to be analyzed and displays the analyzed data.

Twitter Sign up - user signs up for Quitter using Twitter. If user has a Twitter account, system creates a dialog that makes a Twitter login popup.

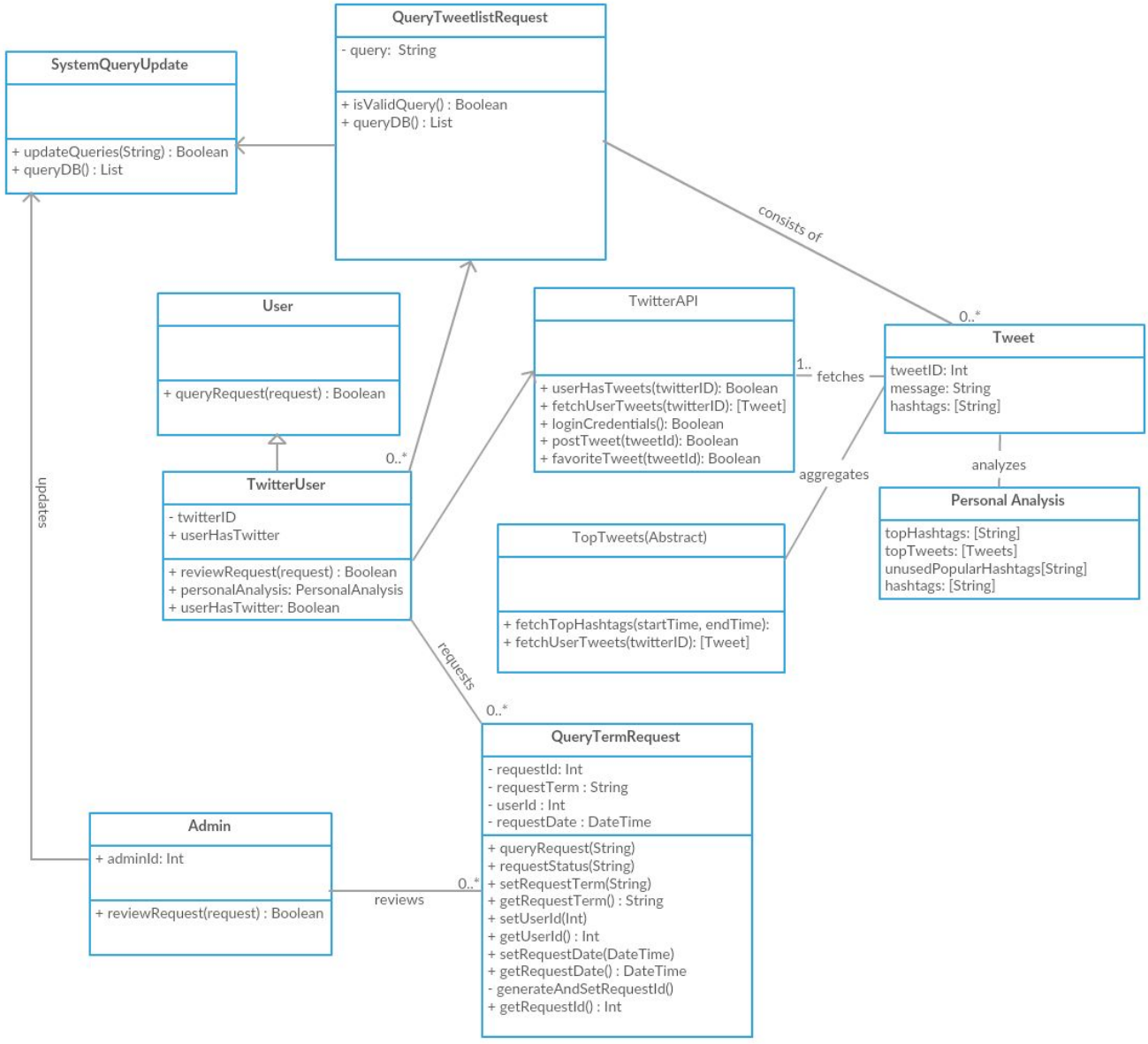
Admin Notified of Query Request - We wanted the admin to be able to review the hashtags that users had requested to be queried for the day. However, when we pivoted to focus primarily on political candidates and the tweets associated with them we decided that we no longer needed this feature because we would just continuously persist data for the candidates as time went on.

Personal Tweet Analysis - we had wanted the user to be able to view personal stats and tweet analysis. However with our decision to narrow our project down to focus on the upcoming presidential election, we found this use case to no longer be necessary because we are just displaying tweets that are relevant to the candidates and the cards that we used enabled the interactions we desired.

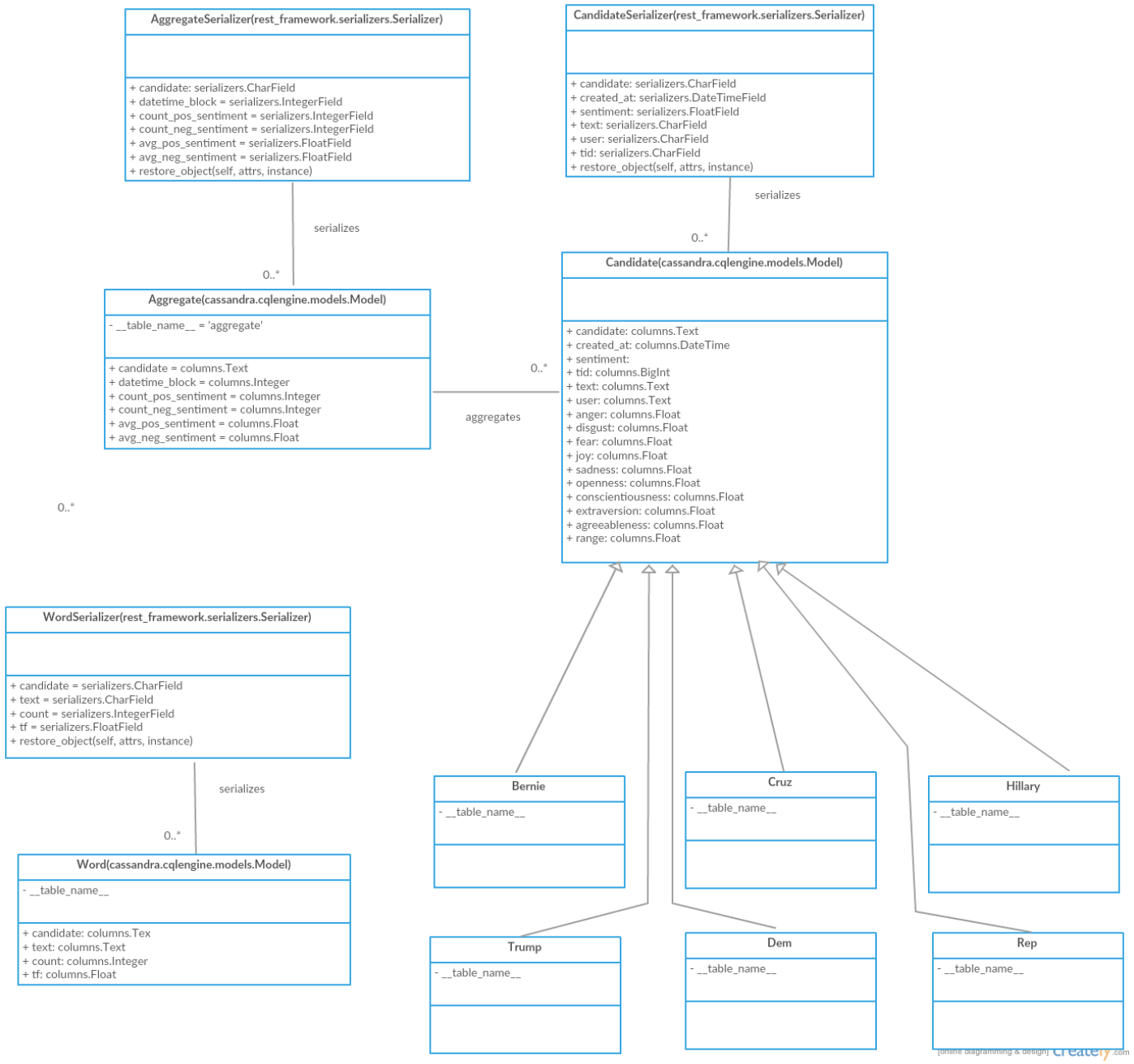
Post Tweet - we wanted the user to be able to post tweets containing statistics views from our application. We ran out of time for this feature, so if we have a little extra time before the due date we may try to find a way to this. Otherwise, we may just enable the ability to export charts using the Highcharts API.

2. Show your Part 2 class diagram and your final class diagram.

Part 2 class diagram:



Final class diagram(For Django backend):



What changed? Why? If it did not change much, then discuss how doing the design up front helped in the development.

Most of our changes were because we changed focus from a generalized tweet analysis to one focused on top presidential candidates. These changes included persisting tweets related specific keywords associated with candidates and political parties. This allowed us to get

specific data to analyze using sentiment analysis and create a more focused front end for our users. Despite the amount of changes we went through, we still found that it was hugely helpful to the development process to design the process up front as it allowed us to plan how changes would impact the remaining structure of our project, and the API development was planned with the help of the class diagram we created.

3. Did you make use of any design patterns in the implementation of your final prototype? If so, how? If not, where could you make use of design patterns in your system?

For the front end of our project we used Angular which implements the MVC architectural pattern. Using Angular for the front end and having REST calls to the Django backend was a very favorable decision because decoupling the view from the model allowed us to make schema changes to the Cassandra database without affecting the frontend controllers that were written in JavaScript, so each time the schema changed it wouldn't break all of our views.

Django is also an MVC framework and we were able to leverage the decoupling of all parts of our system to make changes to the front end and the cnn model trainer without risking the breakage of other components

For our Django models we used the abstract factory design pattern since we made a separate model/table for each candidate to improve query performance, but still needed to have a similar schema.

For our api views we used the adapter method to allow the two libraries we were using i.e. django-rest-framework and django-cassandra-engine to work together, since django-rest-framework was designed with sql in mind and did not play well with our cassandra database.

4. What have you learned about the process of analysis and design now that you have stepped through the process to create, design and implement a system?

Going through the process of system design and analysis showed us the value of having a well thought-out approach by setting the guidelines needed to meet our project's objectives up front. System design in general proved itself an important tool without which our project would not have been able to proceed to smoothly; by identifying the components and data necessary for our system to be properly implemented prior to beginning development of our web application, our team was able to work through many of the kinks purely during the developmental stages. Because each of us had little experience with the design process prior to this class, we had a bit of a hard time getting our project started and deciding what we wanted to pursue prior to starting work on the system. We did find that going through this process and drawing up each type of diagram was an effective learning experience and a rewarding process, as we were able to save time on decisions and focus on following our plan for the remainder of the semester.

5. App Demo

Demo

REST API + Endpoint

Latest Trump Tweets From API