AGGIE-2.0 Documentation and User Guide

This document will serve as a guide for both system administrators and developers who will be installing or extending Aggie-2.0 and for the front end users who would be using Aggie-2.0 for election monitoring.

Installing Aggie-2.0

1. Install node.js on the system. Check out http://nodejs.org/download/ to download an installer/source
2. Install MongoDB which serves as the primary database for the system. http://www.mongodb.org/downloads
3. Download the Aggie-2.0 project source from Github => https://github.com/alexstelea/Aggie2.0
4. Create a data directory for MongoDB with the structure /data/db/
5. Start MongoDB server instance by running **$ mongod --dbpath="path-to/data/db/"**
6. Go to the project folder (the one that contains app.js & package.json) and run npm install
7. Run **$ node app.js**
8. Open a browser and go to http://localhost:9000 to check if the login screen comes up.
9. Use the admin credentials {admin:adminadmin} to login.
10. New Tweet Bots can be started by typing in search terms in the input box on the top right corner and the result can be observed in the live feed.

User Guide (for front end users)

Step -1 => Login

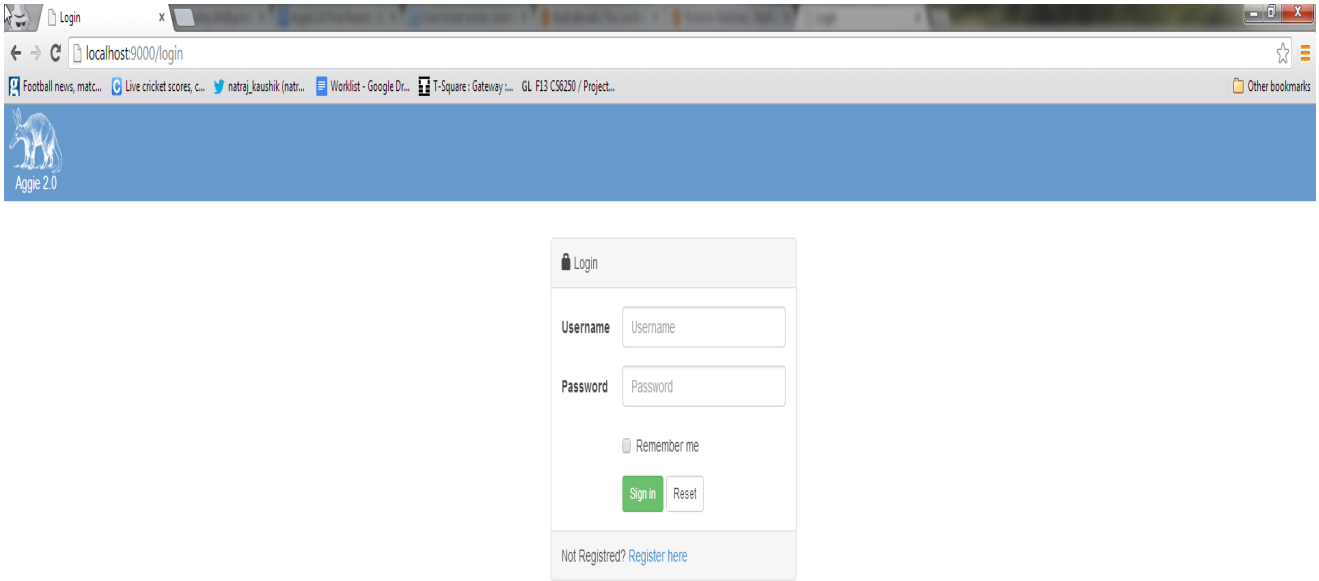

Fig.1 Aggie-2.0 Login Page

This is the page that is expected to be displayed on going to http://server_address/login in a browser after a successful installation of Aggie-2.0. Both admins and regular users can use this page to authenticate themselves and login. At this point, the admin credentials are

Username : admin
Password: adminadmin

The admin can register users and more admins by going to the http://server_address/register page

Step 2 => Register users

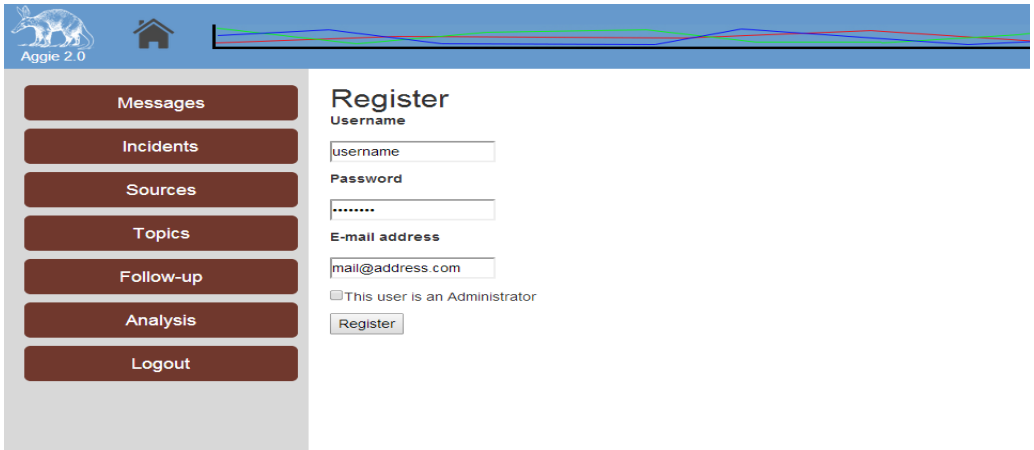

Fig.2 Registration page for admin

Step 3 => Go to messages page

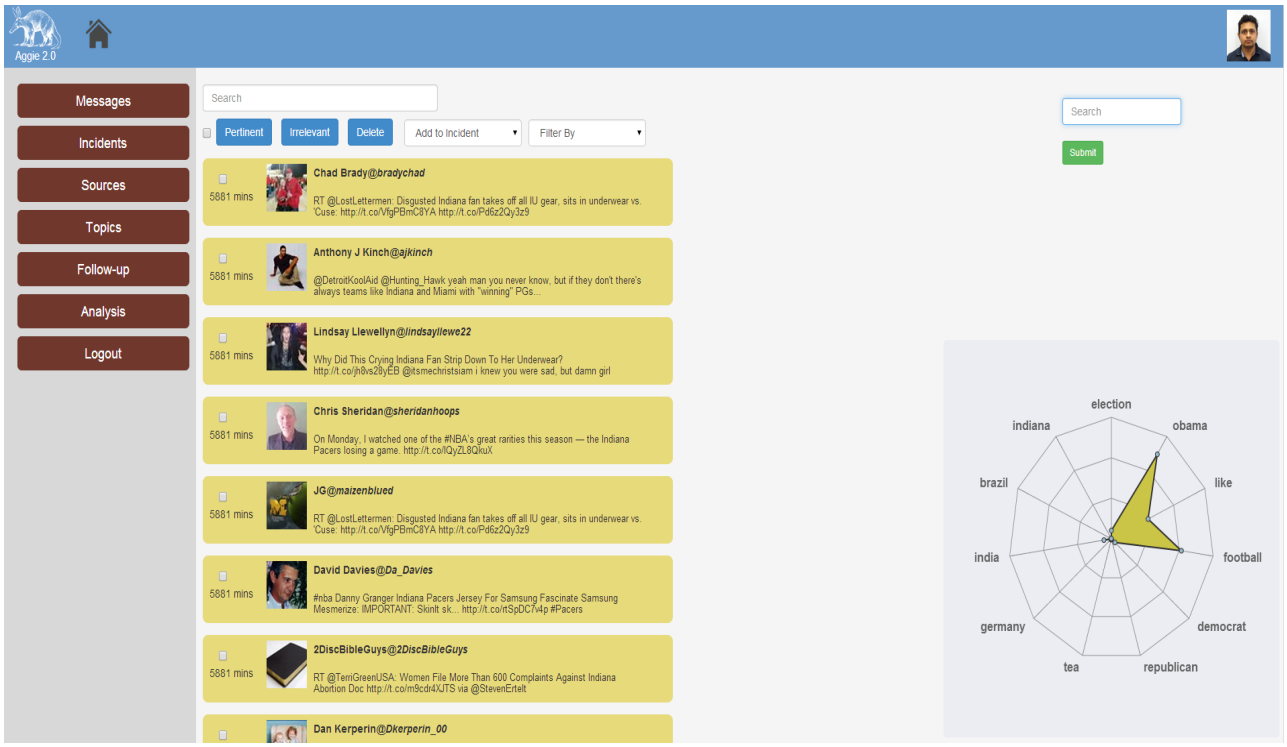

Fig.3 Messages Page

Fig.3 shows the main messages page where live data and trends from the Aggie-2.0 data sources is displayed to the user. The following explanatory figures will show the what each part of the user interface means.

Step 4 => Start new searches

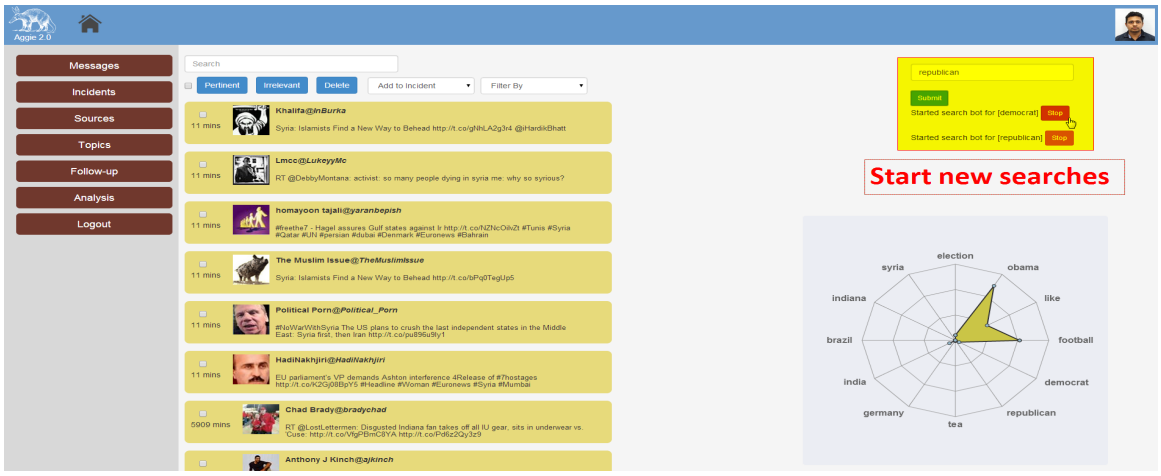

Fig.4 User can start new searches by typing into the search bar on top right

One of the core features that has been implemented is the ability to start a large number of simultaneous keyword searches on live data from Twitter. The user simply enters a search term and clicks on Submit. Feedback that a search has been started on a particular term is provided in the form of text with a “Stop” button to stop the search for that term.

On entering the search terms, the user can then look at the live feed data on the screen as it auto updates which is shown in Fig.5

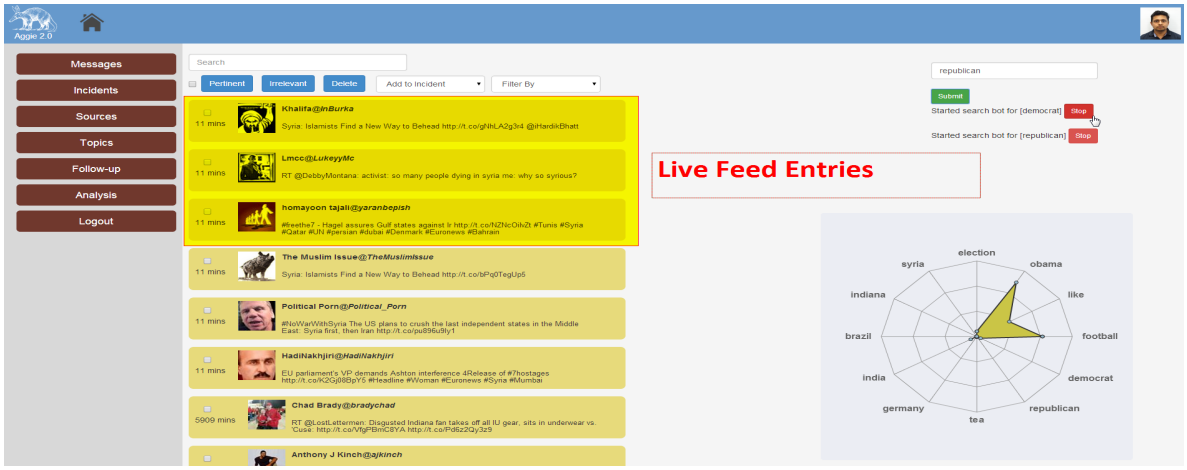

Fig.5 Live Feed container

It must be noted that all the feed entries are also stored in the MongoDB database and can be retrieved independent of the user interface to extract analytics and trends.

We provided a radar chart on the messages page which shows trends for all the search term results. This can be used as a simple indicator of the popularity or anti popularity of a certain term in the social stream.

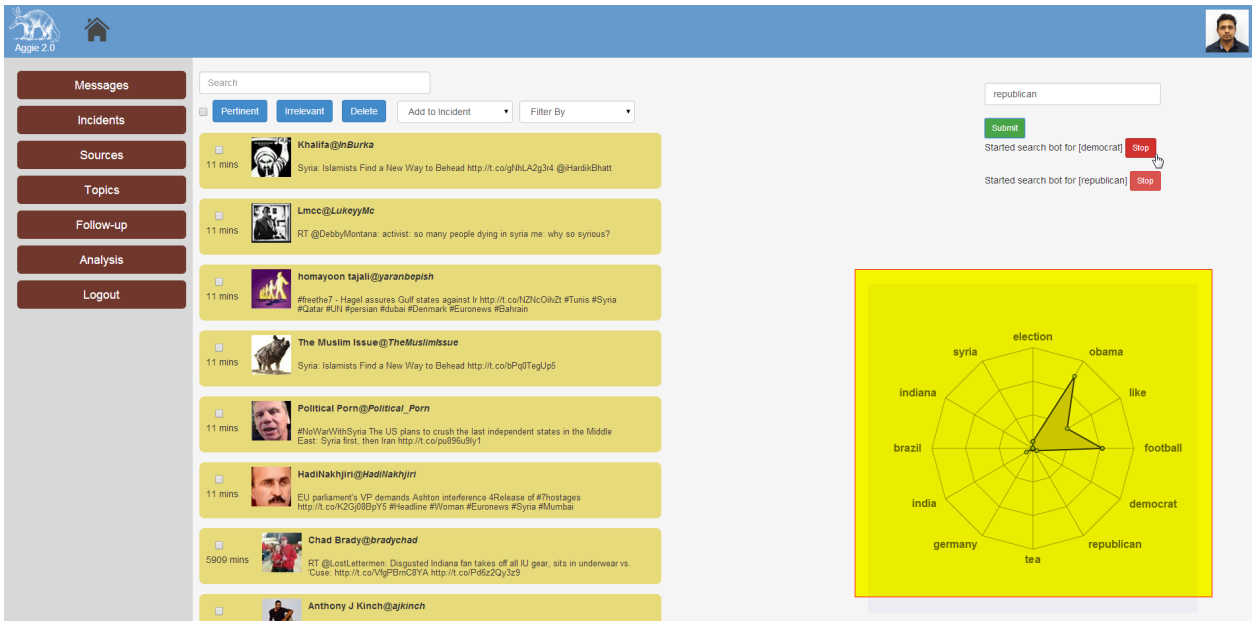

Fig.6 The spokes in the radar correspond to the search terms

The implementation of some features like 1) incident creation and 2) marking of items as relevant/irrelevant to a particular incident is not yet implemented and given that we have all the necessary infrastructure, should not be really hard to do so.

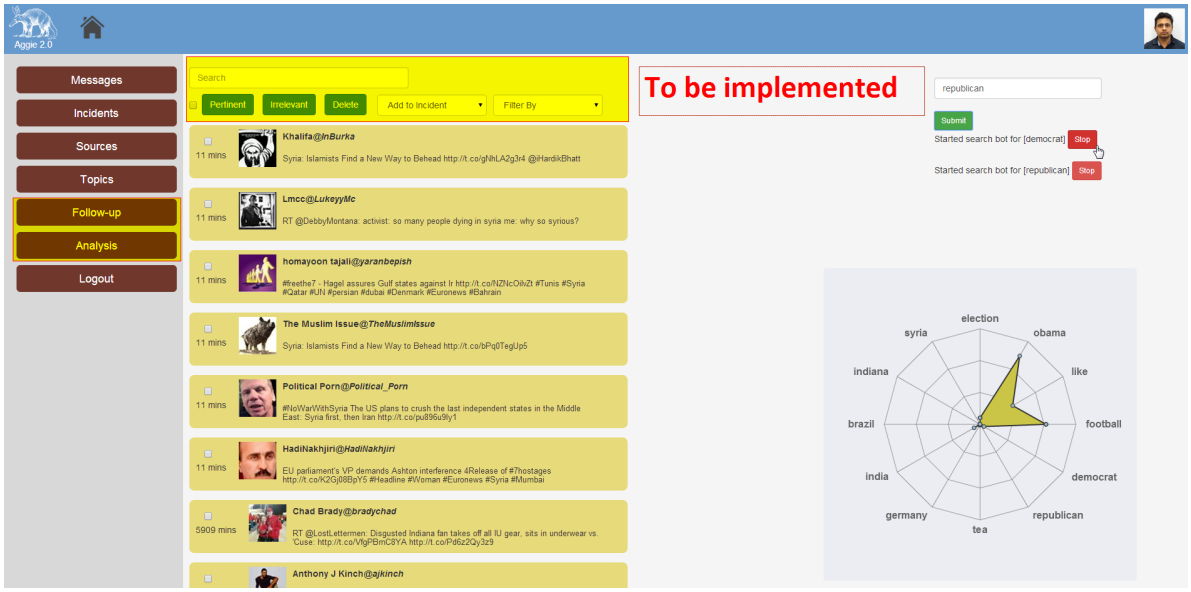

Fig.7 To be implemented features

Logging out of the user interface, is by clicking the logout button on the left navigation bar.

Step 5 => Logout

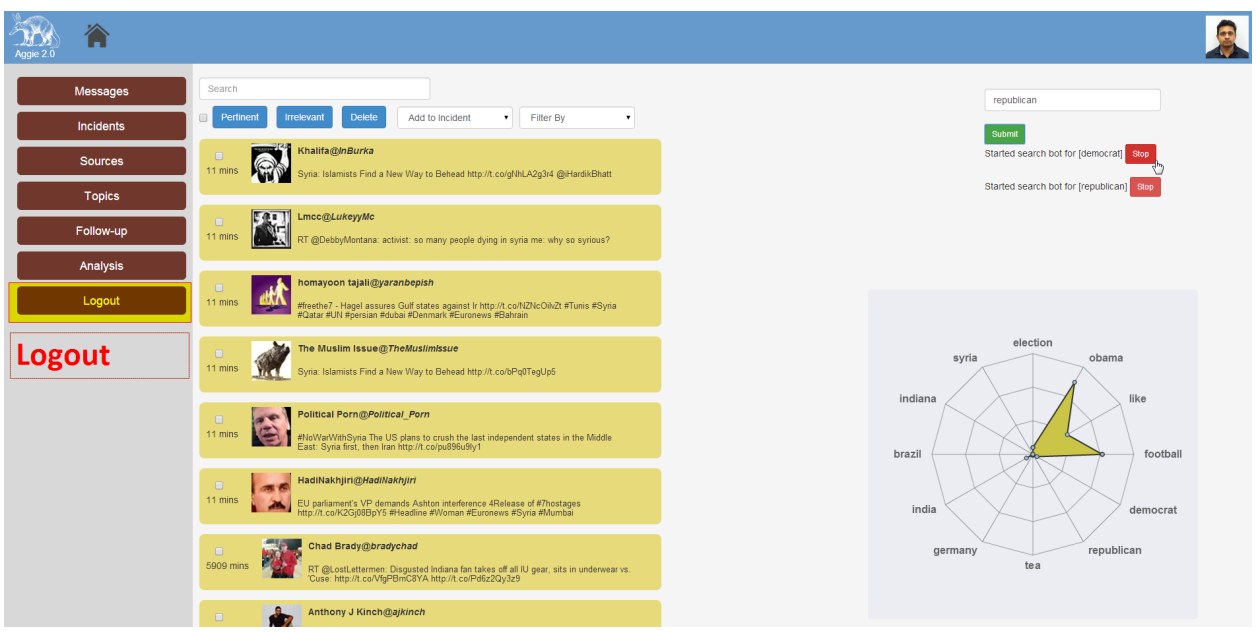

Fig.8 Logout of session

Understanding the Code (for developers)

This section provides an overview of the implementation of Aggie-2.0 platform. The primary technologies used in the implementation are

1. Node.js - which provides the basic framework for all the search bots (Twitter, RSS and Facebook) and for the web front end.
2. Express - the web framework we used to provide a REST API for Aggie-2.0
3. MongoDB/Mongoose - provided the database to store feed from various data sources.
4. Twit - javascript wrappers over Twitter API
5. Feedparser - node.js module to parse RSS feeds
6. Bootstrap 3.0 - the UI framework used on the front end
7. jQuery - UI library used on the web front end which provides functionality for dynamic HTML insertion and making AJAX calls to the server.

We have also used a number of Node.js modules/libraries to assist us in various bookkeeping operations like login/session management and password encryption.

Code Structure

./app.js :
This is the **primary script** that begins the Aggie-2.0 web service by connecting it to the MongoDB database (URL specified in ./config/config.js) and exposing a set of routes that can be queried by a HTTP client.

./package.json :
This file contains all the dependencies of Aggie-2.0.

./controllers/aggie.js :
Contains all the core functionality of Aggie. Most of this functionality is exposed as REST API in ./app.js
./controllers/chart-controller.js :

Contains functionality to extract analytics data from the database
./controllers/bots/twitterbot/twitter-bot-controller.js :
Contains methods to start and stop keyword searches on Twitter
./controllers/bots/rss/rss-controller.js :
Wrapper over the node library - Feedparser to crawl RSS feed for specific search terms

./models/data.js :
Data model for data feeds
./models/user.js : Data model for users - contains built in methods for password encryption

./views/ :
This folder contains all the static content like HTML markup, CSS stylesheets and client side JavaScript.

./config/config.js :
Contains all the configuration settings for the application such as Database URL and server port number.
./config/express.js :
Bootstrapping code for the Express server.

Team

Nataraj Kaushik (nmocherla3@gatech.edu)
Siddharth Choudhary (siddharthchoudhary@gatech.edu)
Ses Goe (ses.goe@gatech.edu)
John Hale (jhale9@gatech.edu)
TJ Strott (tstrott@gatech.edu)