CS4624

Multimedia, Hypertext, and Information Access

Library Tweet Support: Team 7

Final Report

May 11, 2021

Daniel Imondo

Client: Bill Ingram
Instructor: Edward A. Fox
Assisted by Prashant Chandrasekar and Makanjuola Ogunleye

Virginia Polytechnic Institute and State University
Blacksburg, VA 24061

Abstract:

This project aims to create an easily browsable interface with a collection of NDJSON formatted tweets from Twitter.

In the Fall of 2020, 4 teams in CS5604 built a system to manage 3 of the University's archive collections, of ETD's, Webpages, and Tweets. This system included a webpage front-end to serve these collections to users, as well as a feature for researchers and curators to manage data using a KnowledgeGraph and Apache Airflow.

The one front-end team developing this website had a very large task and as such, they were unable to fully flesh out all its features. Specifically, the tweets portion of this website was lacking advanced searching functionality, as well as a clear interactive user interface. My project focused on extending the tweets functionality of this website and managed to accomplish this.

My project features a GUI where users can search through the tweets collection and will have results displayed to them one at a time. In my implementation, I used React, CSS, and ElasticSearch. However, the website it is contained in also uses Docker, Flask, Kubernetes, and Python 3.6. The search fields are text, location, and a range search between two dates. When a query is conducted results will be displayed to the user 5 at a time. Each tweet result contains both the information contained within the tweet result (i.e., username, display name, tweet text, date, favorites, replies, and retweets) as well as data on the user who published the tweet (i.e., total favorites, total posts, total followers, and a link to the source tweet). Also, if a tweet contains a hashtag, each of these are linked to a search on Twitter for that hashtag.

This project can be used to browse an archive of tweets. It will be useful in querying tweets for research, such as searching for all tweets made about a subject that were posted from a certain location at a certain time.

Table of Contents

Table of Figures

Introduction:

This project aims to improve on the previously created tweet search page for the Virginia Tech Libraries Archive Website, created last fall. An image of this previous tweet search page is seen in **Figure 1**. As the project was received, there were 25 example tweets available in the repository and this is the data I used in my implementation for testing.

It is important to note that all data displayed is only representative of the tweet state at time of collection. In addition, I had no part in collecting or importing this data at the time of publishing or anytime thereafter. My project solely focuses on querying and formatting the display of this information.

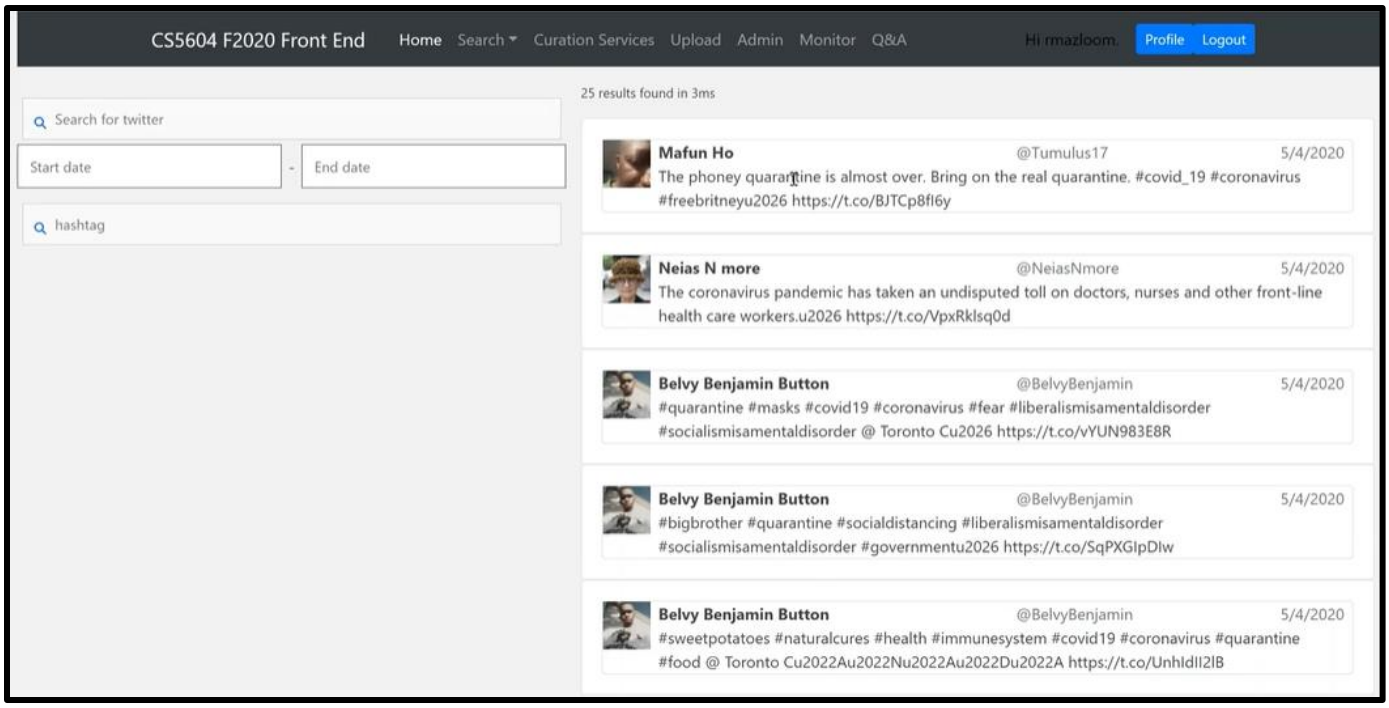

Figure 1: The previous tweets search page

Design/Implementation:

My Design approach for the front-end seeks to style the tweets more accurately so they appear as they do on the Twitter website or app and move away from the more general design this page previously had.

One contributor to this general design was the previous implementation of this project's heavy reliance on the Bootstrap framework for styling tweets, which also leads to issues in properly displaying data.

For example, when a tweet's text had one very long word it would go outside of the bound of the text box. I have remedied this using custom CSS rather than Bootstraps injected CSS.

My custom styling has allowed the project to be portable on desktop devices of varying screen sizes. This was achieved using the CSS **Flexbox** model. Flexbox allows for HTML content to be resized and moved as necessary based on the current size of the screen.

I have added more information to the displayed tweets, such as favorites, retweets, and replies. However, due to the small size of the sample data collected, all tweets have zero for each of the fields because this was the case for all data at the time of collection.

Also, user data has been added including number of followers, total tweets posted, and total favorites this user has gained (all representative of the user's state at time of data collection). I feel the inclusion of this data will allow for researchers and curators to be immediately informed on the level of this user's activity on Twitter. The link to the source tweet was initially contained within the tweet's text in our data. I have moved this link to a button in the user data section, titled "View Source".

Any hashtag contained within the tweet has been hyperlinked to a search on Twitter's website. This will allow a user to see other tweets outside of our database related to the hashtag.

New search features:

A location search has been added, searching both the user accounts listed location as well as the location extracted at the time the tweet was made. This allows for narrowed search results. It also adds new use cases of the page. Locations within our data will be suggested as a user types, informing and guiding the user of available locations and proper query formats.

Requirements:

I: Prerequisites

1. Python version 3.6 *MUST BE 3.6, LATER VERSIONS OF PYTHON WILL NOT RUN THE CODE*
2. MySQL – Running on your Machine.
3. ElasticSearch version 7.3 - open on a terminal on your Machine.
    a. You need to modify the contents of the ElasticSearch configuration file by adding the fields in **Figure 2**. This file is found in ./elasticsearch/config/elasticsearch.yml

```
http.cors.enabled: true
http.cors.allow-credentials: true
http.cors.allow-origin: /https?:\/\/(localhost)?(127.0.0.1)?(0.0.0.0)?(:[0-9]+)?/
http.cors.allow-headers: X-Requested-With, X-Auth-Token, Content-Type, Content-Length, Authorization, Access-Control-Allow-Headers, Accept%
```

Figure 2: Contents of elasticsearch.yml

8. NodeJS, with the following packages installed as shown below.

    b. npm i react-scripts
    c. npm i react-dom react @appbaseio/reactivesearch

II: Clone the Repository.

To access the repository, you must have access to https://git.cs.vt.edu/cs-5604-fall-2020/fe/team-fe-repo/-/tree/knowledgeGraph which is a private repository.

1. Once you have access, clone this repository onto your local machine.
2. Then create a config folder in the root of the repository, with a settings.yaml file in it.
    a. The route should look like: ./config/settings.yaml
3. And the contents of settings.yaml should look like **Figure 3**, password and database reflect the values corresponding to your MySQL database.
4. Next you need to set the values in ./reactivesearch/set_env.sh on FLASK_DB_STRING's username and password fields to reflect your MySQL database. See **Figure 4**.

```
Default:
  mysql:
    host: localhost
    user: root
    password: password
    database: database

  baseuri: http://localhost:3000
  elasticsearch: http://localhost:9200/
```

Figure 3: Contents of the settings.yaml file

```
export REACT_APP_BASE_URI=http://localhost:3000 && \
export REACT_APP_ELASTICSEARCH_URI=http://localhost:9200 &&
\
export FLASK_DB=mysql && \
export FLASK_DB_STRING=localhost\;\;username\;password\;fe &
& \
export REACT_APP_ES_MAIN_ETD=fe_etd_metadata && \
export REACT_APP_ES_MAIN_TWT=fe-twt && \
export REACT_APP_ES_MAIN_WP=fe-wp
```

Figure 4: Contents of set_env.sh

III: Install Dependencies

1. In the root directory of the repository there is a file requirements.txt you must use.
    a. Run “pip install requirements.txt” to install all Python packages.
    b. Also, it is recommended to run “pip install ‘dynaconf[all]’ ” to ensure this package installed correctly.
2. Navigate to the ./reactivesearch folder.
    a. Run “npm run build” to compile this folder.

IV: Initialize Databases

1. MySQL must have three Tables in the database you use for this project It is recommended to name your database “fe” or you will need to change the value of “fe” in FLASK_DB_STRING on ./reactivesearch/set_env.sh. Here is a listing of each table with its name and fields.
    a. user
        i. user_type
        ii. username

iii. email
iv. register_time
v. password
vi. vt

b. IndexTable
i. idIndextable
ii. Iid
iii. Raccess

c. PermissionTable
i. idPermissiontable
ii. Iid
iii. Uid
iv. Permission

2. Import the ElasticSearch data. For the purposes of testing my project, the only data needed to import is the tweets found in ./data/twt_sample_indexed.json. ElasticSearch uses NDJSON (Newline Delimited JSON), this means each JSON entry must contained on one line, and each new entry is delimited by a newline. See **Figure 13** for a readable version of one tweet entry. NOTE: Most of these scripts are adapted from the CS5604F2020FEreport[0].
    a. Create an index in ElasticSearch for the NDJSON data to be contained within, this can be done using the curl script shown in **Figure 5**. Be sure to have ElasticSearch open on its own terminal while you do this.
    b. Then import the NDJSON data into ElasticSearch, which can be done using the script shown in **Figure 6**.
    c. Lastly you can check that your data has been imported by using the script in **Figure 7**.

```
curl -X PUT \
    http://localhost:9200/fe-twt\
    -H ‘Content-Type: application/json’\
    -d ‘{
    “settings” : {
        “index” : {
            “number_of_shards” : 3,
                “number_of_replicas” : 2
                }
            }
        }’
```

Figure 5: CURL script to create an ElasticSearch index.

```
curl -s -H "Content-Type: application/x-ndjson" \

-XPOST http://localhost:9200/fe-twt/\_bulk \
--data-binary @twt_sample_indexed.json; echo
```

Figure 6: CURL script to import tweet data.

```
curl -s -H "Content-Type: application/x-ndjson" \
-XGET http://localhost:9200/twtr/\_count?pretty -d \
'{
    "query": {
        "match_all": {}
    }
}'
```

Figure 7: CURL script to test if data has been imported.

V: Start the Application.

There are two ways to start this application.

1. Full application. This starts the entire application, with account authentication required before viewing the collections.
    a. Navigate to the root directory and run “Python app.py”
2. “Test Mode”. This starts the ReactiveSearch portion of the page and allows you to view the collections without authenticating or creating an account.
    a. Navigate to ./reactivesearch and run “npm start”
3. In either mode, the page will now be hosted at localhost.

User Manual:

This user manual only details updated functionality of the tweets section of the full site; for more details on the other sections of the full site see the previous teams report. [0]

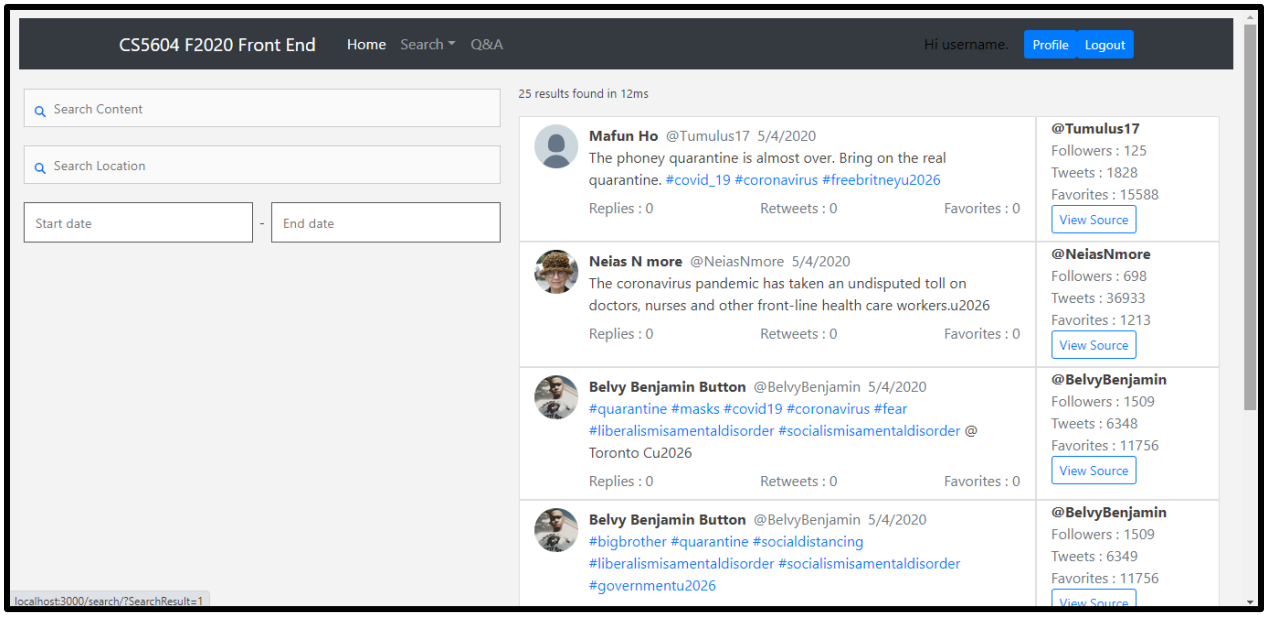

Figure 8: Tweets index

Figure 8 is the main page of the tweets section. There are three different fields that can be used to search, all found on the left side of the page and shown in **Figure 9**.

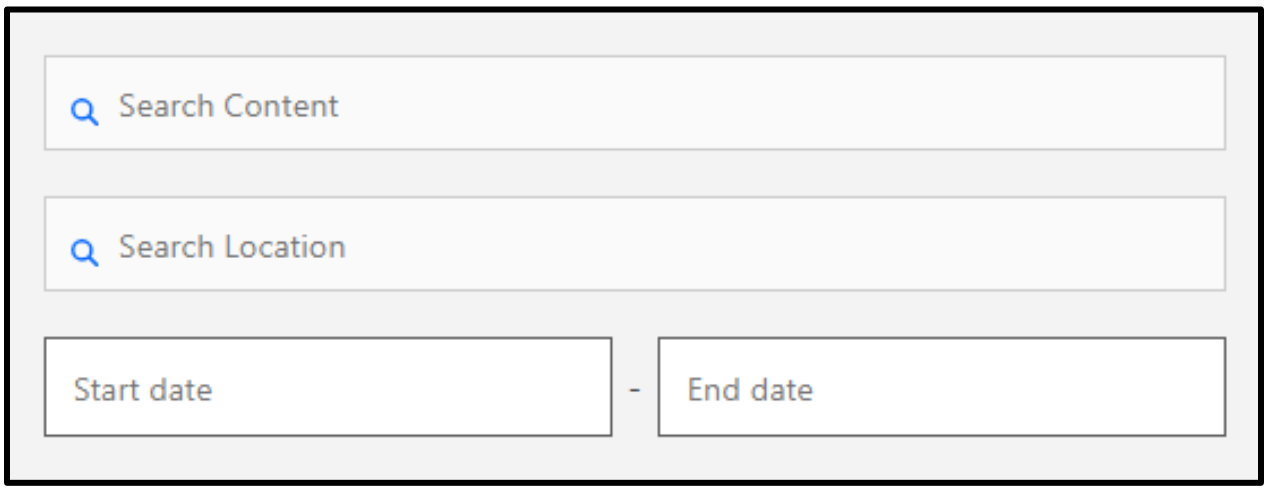

Figure 9: Search Fields.

These fields are used to query the database of tweets, and when used in conjunction will have the results filtered by each of their values. The first field “Search Content” will search the entire text content of each tweet. It will display results that contain the text entered in the box. If you wish to only search the hashtags of the tweets, search with “#” prepended to your search text like “#covid”. Additionally, the results do not have to exactly match the search text, so “#covid” would also display “#covid-19” or “#covidVaccine”.

The next search box “Search Location” will query the results based on both the location the users account is registered to, as well as the location the user published the tweet at. Not all users will have this information available.

Lastly the “Start Date” and “End Date” fields will narrow the tweets based on when they were published.

Figure 10: Page navigation

Only 5 tweets are set to display on each page and at the bottom of the Web Page you can progress to the next page of tweet results shown in **Figure 10**. In **Figure 11** you can see the details displayed in each tweet result. Before I dissect a tweet result, it is important to note that all information displayed is only indicative of the tweet and user at the time the tweet was exported from Twitter. First, profile pictures are displayed in the top right. A profile picture will only be displayed if the users profile picture at the time of publishing the tweet remains on their profile. Otherwise, a default profile picture is displayed. In the main tweet data, a user’s key information (i.e., display name, username, and the date the tweet was published) is displayed at the top. Below is the text contained in the tweet, with all hashtags hyperlinked. On click of a hashtag you will be redirected in a new browser window to a search on Twitter of the hashtag. Below the text is the tweet information: replies, retweets, and favorites.

On the right section there is user information displayed including: the username, followers, total tweets made, and total favorites earned. Below this is a link to the source tweet as hosted by Twitter.

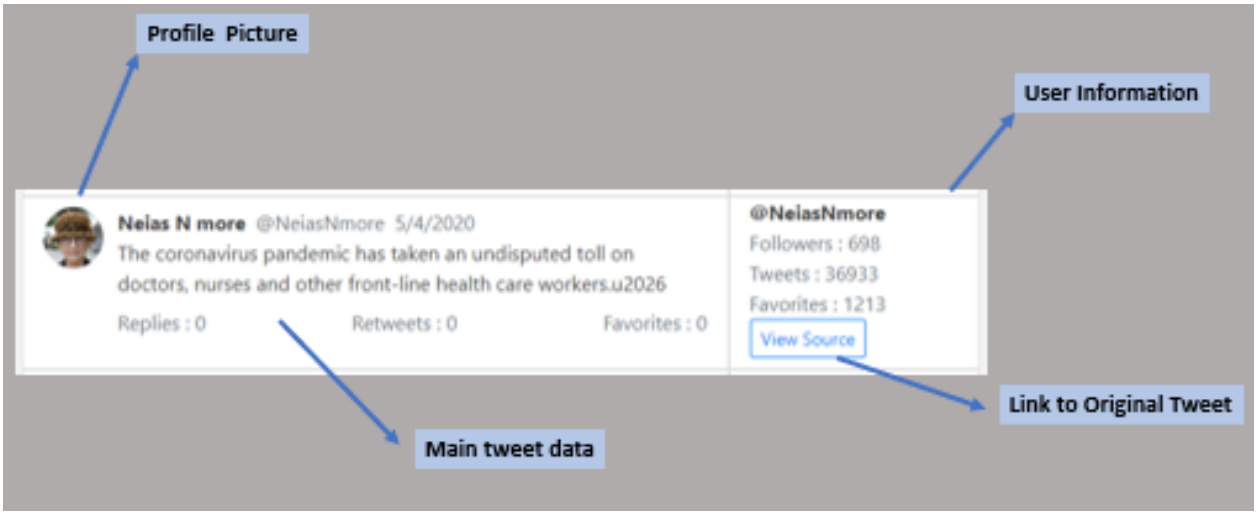

Figure 11: Anatomy of a Tweet Result.

Developer Manual

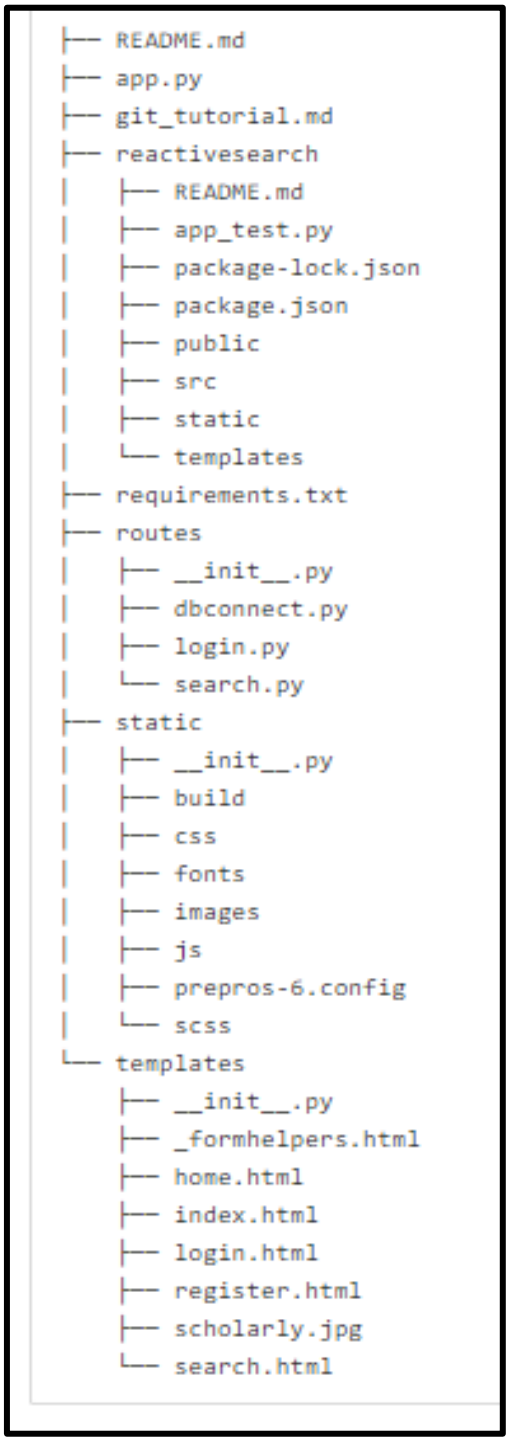

Figure 12: Table of Directories

This Section of the report aims to give the prospective developer information on importing new indexed data to the collection. Furthermore, in **Figure 12** you will see a Table of the Directories and Files contained in this project. This figure should help give developers an idea of how to navigate the project.

Be sure to have read the Requirements section of this report before you begin reading this section.

Importing new tweets:

To import a new set of tweets, first make certain that they follow the formatting of fields in **Figure 13**. Note that when you create the dataset to be imported It will NOT follow the format of **Figure 13** but will be in NDJSON. This figure is only meant to show all the fields of the current tweets in a human readable way.

In NDJSON, each tweet entry will be on its own line and will be prepended with an index. An example of this format is shown in **Figure 14** and can be further examined in ./data/twt_sample_indexed.json. You can then import this data by using the CURL scripts shown in **Figures 5-7** as described in the requirements section of this report.


```
{
    "quote_count": 0,
    "contributors": null,
    "truncated": true,
    "text": "The phoney quarantine is almost over. Bring on the real quarantine. #covid_19 #co
ronavirus #freebritneyu2026 https://t.co/BJTCp8fI6y",
    "is_quote_status": false,
    "in_reply_to_status_id": null,
    "reply_count": 0, "id": 1257271308062208000,
    "favorite_count": 0,
    "entities": {
        "user_mentions": [],
        "symbols": [],
        "hashtags":
            [{"indices": [68, 77], "text": "covid_19"},
            {"indices": [78, 90], "text": "coronavirus"},
            {"indices": [91, 103], "text": "freebritney"}],
        "urls":
            [{"url": "https://t.co/BJTCp8fI6y", "indices": [105, 128],
            "expanded_url": "https://twitter.com/i/web/status/1257271308062208000",
            "display_url": "twitter.com/i/web/status/1u2026"}]},
    "retweeted": false,
    "coordinates": {
        "type": "Point",
        "coordinates": [-80.25, 43.55]},
    "timestamp_ms": "1588591813470",
    "source": "<a href=\"http://instagram.com\" rel=\"nofollow\">Instagram</a>",
    "in_reply_to_screen_name": null, "id_str": "1257271308062208000",
    "retweet_count": 0,
    "in_reply_to_user_id": null,
    "favorited": false,
    "user": {
```

```
        "follow_request_sent": null,
        "profile_use_background_image": true,
        "default_profile_image": false,
        "id": 1096083799534960640,
        "default_profile": true,
        "verified": false,
        "profile_image_url_https": "https://pbs.twimg.com/profile_images/1251785729518288897/csccyZlo_normal.jpg",
        "profile_sidebar_fill_color": "DDEEF6",
        "profile_text_color": "333333",
        "followers_count": 125,
        "profile_sidebar_border_color": "C0DEED",
        "id_str": "1096083799534960640",
        "profile_background_color": "F5F8FA",
        "listed_count": 0,
        "profile_background_image_url_https": "",
        "utc_offset": null,
        "statuses_count": 1828,
        "description": "The subject who is truly loyal to the Chief Magistrate will neither advise nor submit tou00a0arbitrary measures.~~ Junius",
        "friends_count": 428,
        "location": "Dawn-Euphemia, Ontario",
        "profile_link_color": "1DA1F2",
        "profile_image_url": "http://pbs.twimg.com/profile_images/1251785729518288897/csccyZlo_normal.jpg",
        "following": null, "geo_enabled": true,
        "profile_banner_url": "https://pbs.twimg.com/profile_banners/1096083799534960640/1587283898",
        "profile_background_image_url": "",
        "name": "Mafun Ho",
        "lang": null,
        "profile_background_tile": false,
        "favourites_count": 15588,
        "screen_name": "Tumulus17",
        "notifications": null,
        "url": null,
        "created_at": "Thu Feb 14 16:28:36 +0000 2019",
        "contributors_enabled": false,
        "time_zone": null,
        "protected": false,
        "translator_type": "none",
        "is_translator": false},
    "geo": {
        "type": "Point",
        "coordinates": [43.55, -80.25]},
```

```
    "in_reply_to_user_id_str": null,
    "possibly_sensitive": false,
    "lang": "en",
    "extended_tweet": {
        "display_text_range": [0, 160],
        "entities": {
            "user_mentions": [],
            "symbols": [],
            "hashtags": [
                {"indices": [68, 77], "text": "covid_19"},
                {"indices": [78, 90], "text": "coronavirus"},
                {"indices": [91, 103], "text": "freebritney"},
                {"indices": [104, 118], "text": "wrayandnephew"}],
            "urls": [
                {"url": "https://t.co/PF3a1rtMMG", "indices": [137, 160],
                    "expanded_url": "https://www.instagram.com/p/B_w6qFEnTYF/?igshid=1kb2euuf2
aszb",
                    "display_url": "instagram.com/p/B_w6qFEnTYF/u2026"}]},
        "full_text": "The phoney quarantine is almos ...naephew @ Guelph, Ontario https://t.co
/PF3a1rtMMG"},
    "created_at": "Mon May 04 11:30:13 +0000 2020",
    "filter_level": "low",
    "in_reply_to_status_id_str": null,
    "place": {"full_name": "Guelph, Ontario",
        "url": "https://api.twitter.com/1.1/geo/id/2740624a2d391c5c.json",
        "country": "Canada", "place_type": "city",
        "bounding_box": {
            "type": "Polygon",
            "coordinates": [[[-80.326879, 43.473802], [-80.326879, 43.594596], [-
80.153377, 43.594596], [-80.153377, 43.473802]]]},
            "country_code": "CA",
            "attributes": {},
            "id": "2740624a2d391c5c",
            "name": "Guelph"}
}
```


Figure 13: Tweet JSON fields.


```
{ "index": { "_id": "0"}}
{"quote_count": 0, "contributors": null, … … … }
```


Figure 14: Correct format to import.

Lessons Learned:

Timeline/schedule:

The Initial Schedule:

Feb. 10-24: Dive into the existing codebase and have a thorough understanding of the existing code.

Feb. 25 – Mar. 10: Begin Development and have all search UI and displayed data formatted.

Mar. 11 – 27: Implement the new searching features and have them tested.

Mar. 27 – Apr. 7: Implement a remote API for the tweets, allowing for large export of data.

Apr. 8 – 21: Work on final report and presentation.

What the Timeline Actually Looked Like:

Feb. 10: I am given access to the repository, but I am unable to view any code.

Mar. 16: I am finally able to clone the code to my local machine.

Mar. 26: This is the day I was able to get the project running on my machine*

Mar. 26 – Apr. 7: Development of frontend UI and adding location search.

Apr. 9: Interim report

May. 5: Final Report Initial Submission

May. 11: Final Report Edited Submission

In addition to this schedule, I have been meeting with Bill Ingram bi-weekly since early February.

Problems:

I ran into a litany of problems with this project. First was my issue in getting into the repository. Once I had been granted access to the repository, I assumed all was well and so rather than starting by looking at the code, I decided to research React and ElasticSearch instead, to familiarize myself with the tech in use before I look at code without knowing what it does.

This led me into trouble, as I did not realize for a while that I did not actually have access to the code in the repository. Once I figured out this, I contacted **Prashant Chandrasekar** to have my permissions fixed. Getting this issue fixed took two weeks, due to spring break days and slow proceedings.

Then once I was finally able to get the project cloned on my machine, I ran into numerous problems with getting the project running. I was trying to run the old project by using the README found on github. This led to an issue as it is not as comprehensive in its description of configuration.

[a] Firstly it is missing one of the major dependencies, called react-scripts. [b] After installing this I was dumbfounded as to why I was getting a Python error. [c] Once the Python error was resolved, I was able to get the project running on my local machine although I could not log in. [d] Once I could log in, the tweet data was not showing up.

Solutions:

[a] npm install react-scripts

[b] The code uses a deprecated version of Python, and you must install and run the code with Python 3.6, I then figured out I had to reinstall the dependencies with pip on my new Python 3.6 environment.

[c] MySQL must be connected to the project by setting the environment variables in ./root/reactivesearch/set_env.sh, and these variables must be set locally on your machine.

[d] The scripts to import data to reactivesearch import the data under a different database name than the environment variables were set to search for.

These solutions may seem trivial, but the trial was truly in finding what was causing the issues to happen. Remember Python, React, MySQL, and ElasticSearch are all either entirely new to me, or I have little experience with them. Even while meeting with **Makanjuola Ogunleye,** it took us nearly two weeks to get the code to run completely on my machine, also contacting two other members that worked on this project for help.

In my Developer Manual I have made clear process to get this project running, with full disclosure of all dependencies in order to try to remedy these problems.

Future Work

Possibly fix Flexbox model of the page for mobile devices, but this is not a huge priority as no other page on this site is configured for mobile size screens.

Add sorting of results to the search, by fields such as retweets and favorites count.

Import a new, large collection of tweets to give this project value.

Addittional testing is required on a larger dataset, as my testing was limited to the small dataset of 24 tweets.

Need to update the parts of the project that use depreciated Python code to allow for use of the most updated version of Python and not the depreciated version of Python 3.6 it currently requires.

Move this project as well as the website it is contained within to a public website and host the source data on a production version of ElasticSearch for this website.

Acknowledgements:

Bill Ingram: Client

Email: waingram@vt.edu

Makanjuola Ogunleye: Previous Developer

Email: mogunleye@vt.edu

Dr. Edward A. Fox: Professor

Email: fox@vt.edu

References:

[0] Cao, Y., Mazloom, R., & Ogunleye, M. (2020, December 16). *CS5604 (Information Retrieval) Fall 2020 Front-end (FE) Team Project*. Blacksburg, VA: Virginia Tech. http://hdl.handle.net/10919/101526

[1] Damme, T. V., Merkenich, J., Coyier, C., Mejia, L., House, C., Seyedi, M., … Holt, B. *A Complete Guide to Flexbox: CSS-Tricks*. (Accessed 2021, April 7). https://css-tricks.com/snippets/css/a-guide-to-flexbox/

[2] ElasticSearch. (2021). *ReactiveSearch Quickstart*. Appbase.io Docs - Search stack for Elasticsearch. (Accessed 2021, March 16). https://docs.appbase.io/docs/reactivesearch/v3/overview/quickstart/

[3] Cao, Y., Mazloom, R., & Ogunleye, M. (2020). cs-5604-fall-2020. CS 5604 fall 2020 Front End Project. Blacksburg, VA: Virginia Tech. (Accessed 2021, February 10) https://git.cs.vt.edu/cs-5604-fall-2020.

Note about [3]: *This is the initial project that I made changes to; this is a private git repository; for access, please contact one of the authors.*