CS172 - Team Solid Final Project Report

Team members:

- Yuteng Zhang SID: 862017519
- Bohan Zhang SID: 861215636
- Dishon Jordan SID: 862083504

Github repo link: https://github.com/CS-UCR/final-project-solid

Introduction

This is a small search engine for Twitter.
In Part 1, we build a crawler (from scratch) to collect tweets from the Twitter Streaming API

In Part 2, we use an opensource tool called ElasticSearch along with the Kibana console to index all the collected tweets.

In Part 3, we propose and implement an extension of the search engine created in Parts 1 & 2. We chose to build an interactive interface for the user using flask python.

Since C++ is not allowed, We chose to use python to write this assignment.
The submission contains a python file elastic.py to run the search engine, along with the templates folder containing the simple frontend html file and the twitter_stream file containing the json data file and a python file twitter_stream.py for crawling data.

Part 1 : Web Crawler

For this part, we imported tweepy and lxml from the python library.

Run the file twitter_stream.py to crawl the data. The program could be terminated by pressing CTRL + C while running. After the termination, all the data crawled so far will be stored in a json file in the same directory with twitter_stream.py

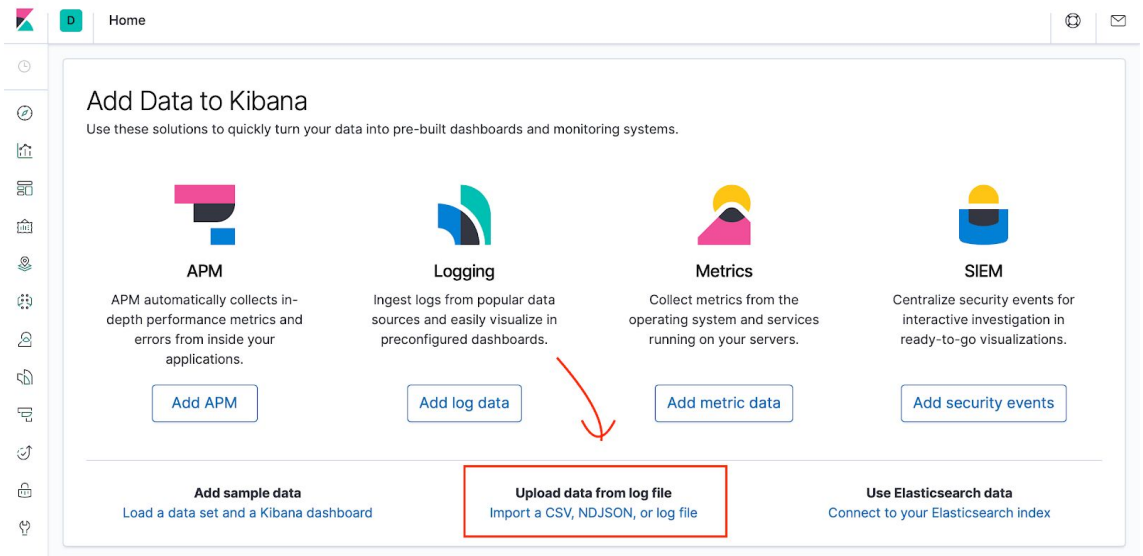
Part 2: Data Indexing

To import data into ElasticSearch, we chose to use the Kibana console.

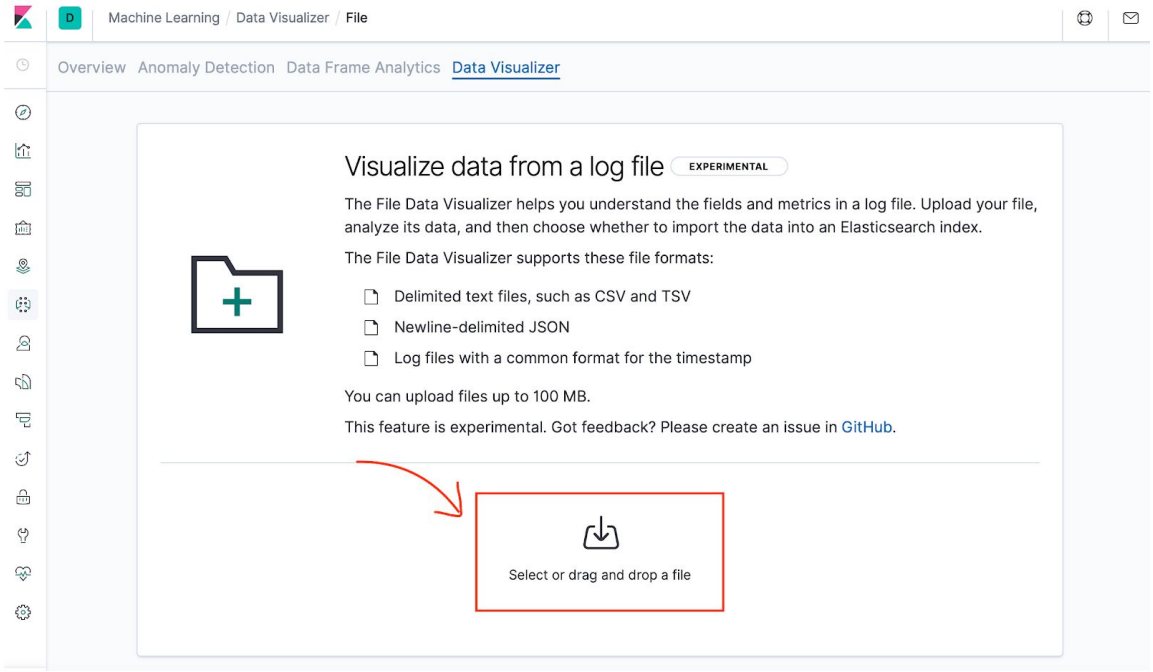
- First download ElasticSearch and unzip
- Then download Kibana and unzip
- Run ElasticSearch and Kibana

Go to your localhost:5601/, if both ElasticSearch and Kibana runs successfully, should be able to see a page like this:

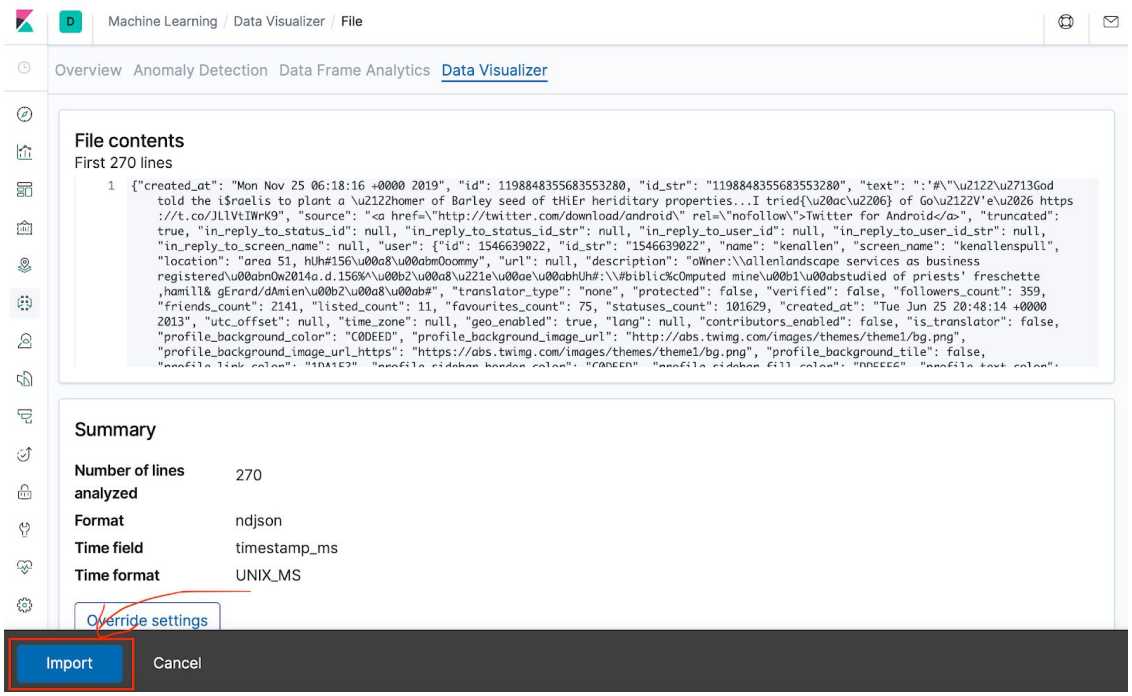
Click on the "upload data from log file" button



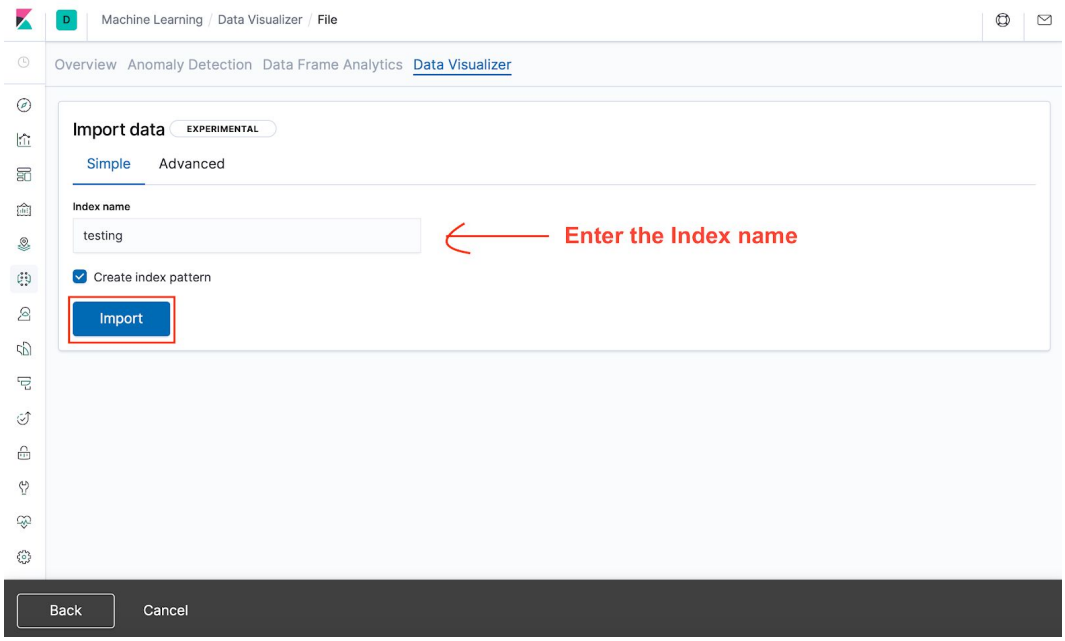
Next, select the data file you wish to import



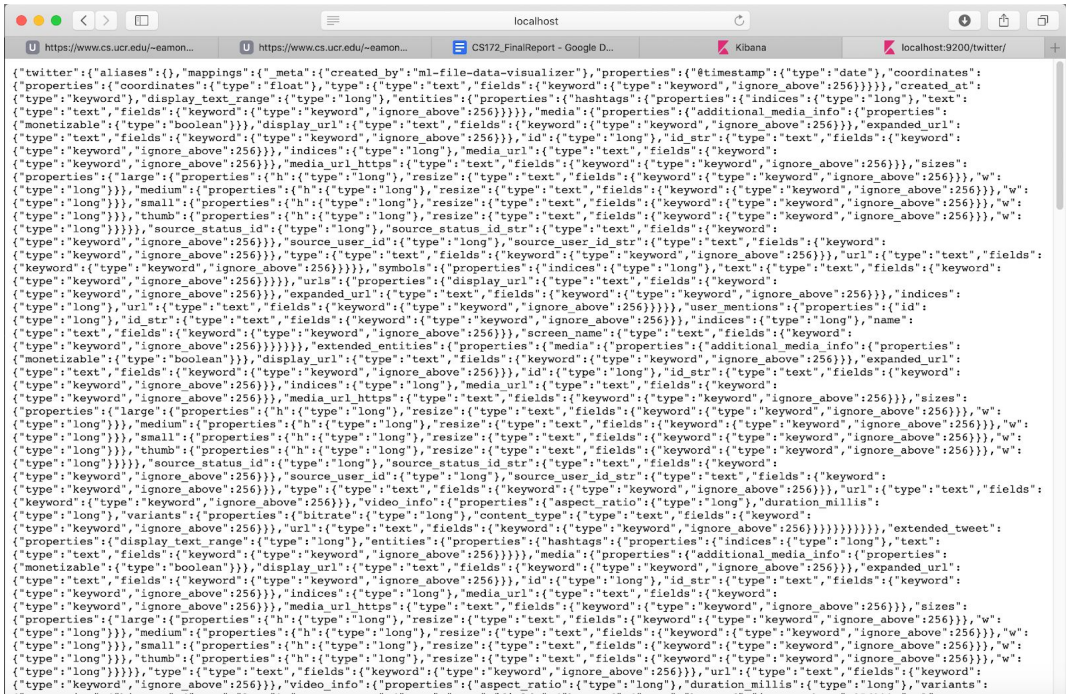
You will see your file contents and a short summary, check if it's correct and hit import



The final step is to enter the index name you wish to give to your data



Wait for it finishes, and after that goto the ElasticSearch port localhost:9200/the_index_name_you_entered/, to see your data



Part 3 Frontend User Interface

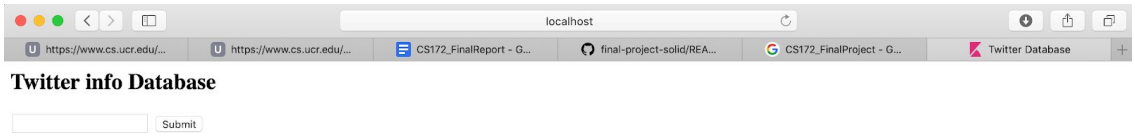
We build our web frontend interface using flask python.

We imported library called Flask and requests.

The interface does the following tasks:

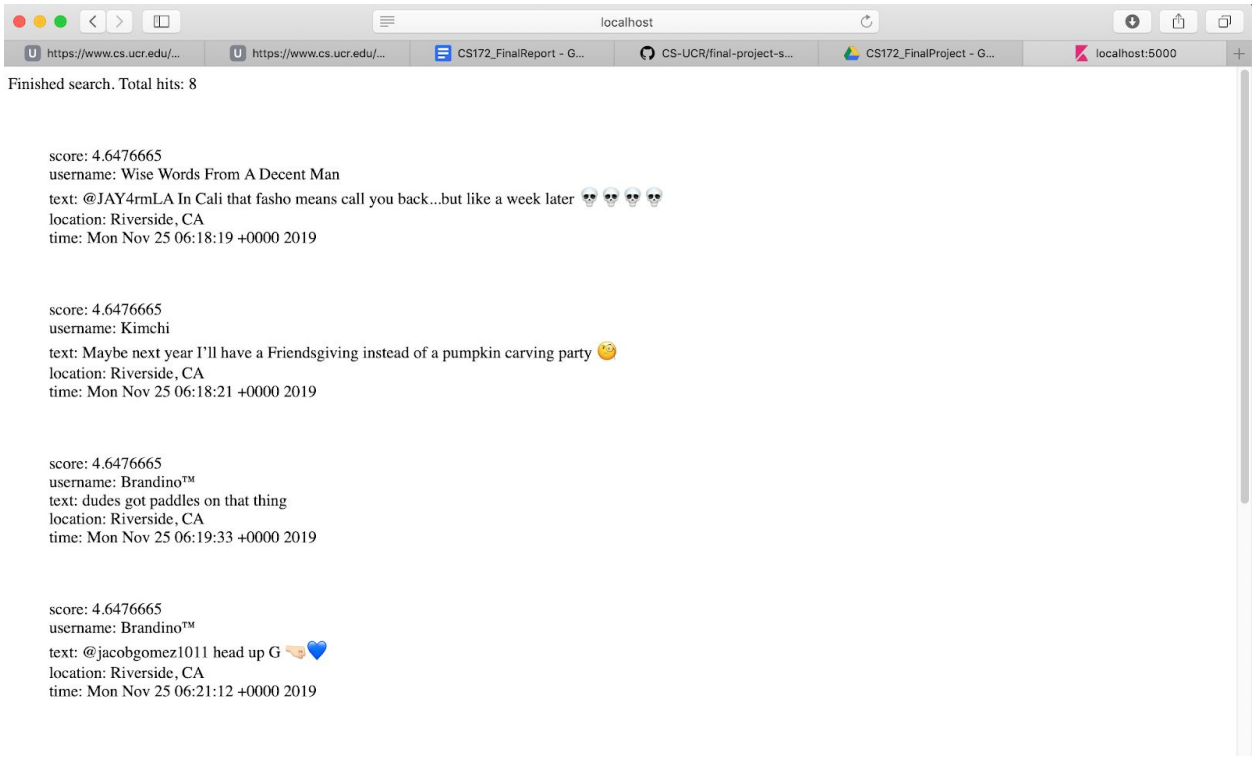
- Take in a query as input
- Pass in the query as a GET request to the elasticsearch
- The elasticsearch is then able to return the related result.
- The result is sent to the browser and displays.

The app runs on localhost:5000/ by default. Below is the web app



Bellow is a sample output result when searching keyword “Riverside”:

As a result, if the number of results appears to be large, ElasticSearch returns the top-10 results according to the score. For this example, there are 8 hits total, which is less than 10, so it will only display 8 results.



Collaboration Details :

Part 1: Dishon Jordan

Part 2 & 3: Yuteng Zhang, Bohan Zhang(we utilized pair programming on Yuteng's computer, so all code was uploaded in by Yuteng)

Research on ElasticSearch and Kibana: Yuteng Zhang, Bohan Zhang

Research on Flask-Python: Yuteng Zhang, Bohan Zhang

Github README: Yuteng Zhang

Project Report: Bohan Zhang, Yuteng Zhang

For detailed information and steps to run the program, see README.md on github