Capstone Project

Classmates Elasticsearch

Kristopher Rollert, Kai Schniedergers,
Michelle Slaughter, Chuanshi Zhu

UC SANTA CRUZ Baskin Engineering

Abstract

The core of Classmates' web application functionality is reliant on their search engine that queries through Classmates' tens of millions of student records. Classmates has been using Apache Solr for its search engine since 2012 and the engineers feel that it is time to upgrade to something more modern. Solr has been causing performance issues as well as limiting the functionality of Classmates' search engine. With this project, we prototyped Elasticsearch's more modern functionality to reduce query latency, improve result accuray, and set up new features like search completion.

Approach

Classmates had no prior tools for Elasticsearch so for our conversion, we were given csv files with data and built the rest from scratch. We:

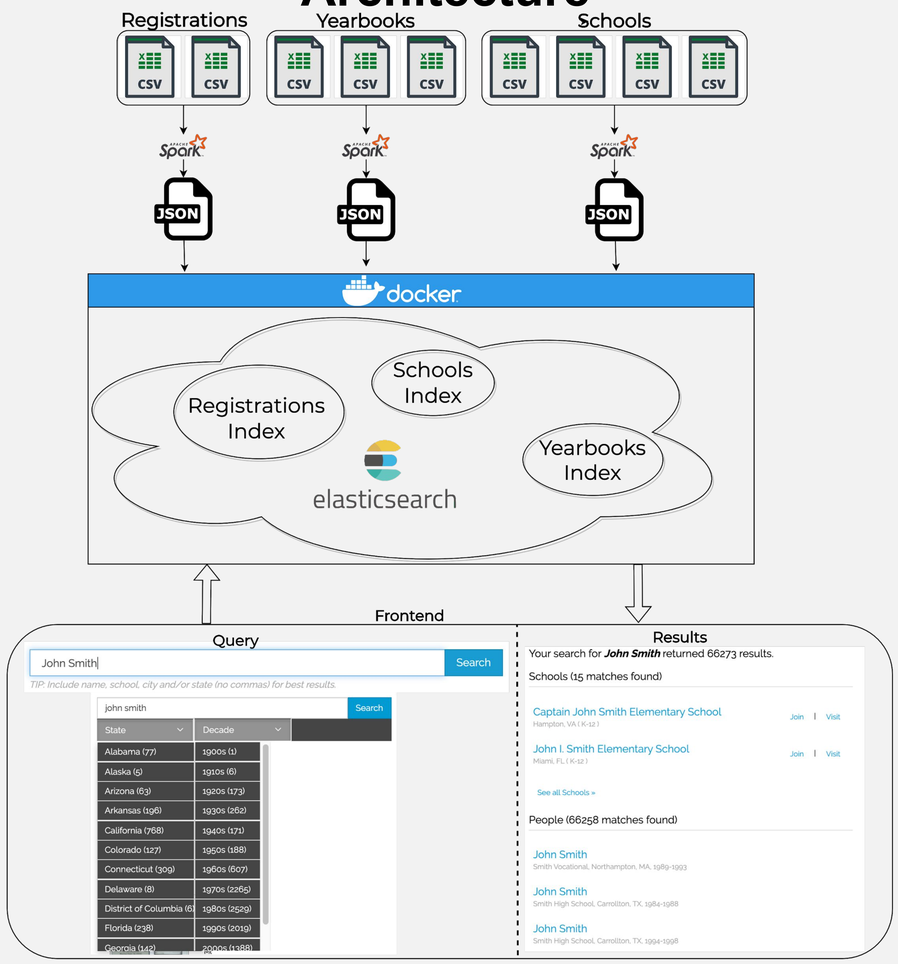
- Converted and merged ~100 GB of csv files into compact json files using Apache Spark
- Used Docker to stand up multiple "nodes" of Elasticsearch on an Amazon EC2 instance
- Created Python tools to send data to Elasticsearch containers so it can be indexed
- Built our one index and three index approaches and analyzed the speed, efficiency, and accuracy to figure out which would be optimal
- Created a query autocomplete feature using a neural network and log files of previous searches

Overview

Classmates has 207 GB of data on old classmates, 371 MB of data on schools and 287 MB of data on old yearbooks, which is searched over 100k times a day.

After years of using Apache Solr, the engineers at Classmates decided it was time to upgrade their system to Elasticsearch (ES). ES provides distributed queries, better functionality around filtering, and many more features that fit better with Classmates' system design. Our team was in charge of switching systems. This required moving over the Solr data, comparing Elasticsearch versus its wrapper software: Appsearch, and configuring the final setup.

Architecture



Acknowledgments

Thank you to our sponsors and mentors for all your help!
Sponsors (Classmates): Yalin Yesiltas & Payal Patel
Mentors (UCSC): Richard Jullig & Akila de Silva

Evaluation

There are two options for Elasticsearch. The one index option is to have schools, registrations,and yearbooks in one big index for searching. The three indices approach is to have an index for each section. In order to choose whether we use one index or three indices, we had eight different sections to evaluate their performance. They are name search, facets, auto-suggest, index size, spell correction, average query response time, development effort and relevance. We wrote our own python scripts to calculate scores for many of the sections.

Results

Result Highlights	One Index	Three Indices
Index Size	171,908 docs (2.7 GBs)	1,678,792 docs (830.5 MBs)
Avg Query Time	9.5 ms	5.0 ms
Relevance	13.78	12.99

In most of the data we compared, the three indices approach performed significantly better. Though the relevance score was lower, we found that the actual results were more relevant.

Conclusion

With the tools we've created, engineers will be able to seamlessly convert their search engine from an outdated version of Apache Solr to Elasticsearch. Our tools will allow engineers at Classmates to stand up Elasticsearch containers with Docker, as well as port and upload data from their existing Solr data. Our tools will help modernize one of the most vital functions of the Classmates.com website: searching.