Software Development: Building a Suite of Web Applications for Research and Analysis

CS4991 Capstone Report, 2025

Feyona Zhang
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
ket6cc@virginia.edu

ABSTRACT

In today's increasingly connected world, the digital footprint we leave across the internet can pose security risks to governments and organizations that want to keep their search activities private. To aid the research and analysis conducted by analysts at CACI International, Inc, I developed a suite of web applications that replaced the use of third-party tools. Using the Python Flask framework, I implemented the functionalities of two web applications presenting web-scraped data collected in Elasticsearch and AWS EC2 buckets. Throughout development, I communicated with analysts to understand their needs and translate those needs into a software product, collaborated with my team, worked independently, and gained knowledge on cutting-edge software technologies. The result was the development of two web applications, one which presented autonomous system (AS) information and another which indexed and searched a file system based on keywords. These applications enhanced analysts' research processes by increasing speed and offering a diverse set of tools. After the development of these apps, there remains a continued need for maintenance and improvement in the software. Additionally, as research needs evolve and more data becomes available, new applications will need to be developed.

1. INTRODUCTION

While surfing the internet, browsers track a wide range of information pertaining to user activity. This is done through cookies, fingerprinting, scripts, and more. One might think the architecture of the Internet contains identifying information to the website or domain object model (DOM) the user is visiting, however, information can be leaked to third parties through JavaScript and cross-site embeds. Websites can gather user information such as IP addresses, browser, device, and operating system settings, web traffic, and website interaction, such as number of clicks and search terms. These various pieces of information can be used to follow the user across the Internet and form a digital blueprint of user activity.

In addition to tracking, the Internet hosts a large repository of raw data. Often the data is unpurposed because it is in machine-readable format such as comma separated values (CSV) or JSON files. With the proper tools to present and sift through the data, connections and meaning can be drawn from seemingly random information that can have a huge impact for organizations in terms of data analytics. Combining the security concerns and the potential power of big data on the internet, creating web applications can be the bridge between data on the internet with new discoveries at organization.

2. RELATED WORKS

In their study, Bujlow et al. (2017) surveys all the different ways one can be tracked across the Internet. Some security concerns highlighted include third-party web embeds and session data. Their research emphasizes the relevance of companies concerned with keeping their web browsing private. This demonstrates the increasing need for companies to develop their own software tools for secure and trusted use which is my role at CACI.

Another study by Kononenko et al. (2014) presents Elasticsearch as an effective tool in processing "big data". Elasticsearch is a "distributed full-text search engine" that is free, open-source, and scalable. The researchers examine their experience using Elasticsearch, highlighting its strengths and weaknesses. They concluded Elasticsearch as a NoSQL database coupled with machine learning is able to perform preliminary analysis and derive insights from big datasets, making it an effective tool for data analytics of web-scraped unstructured data.

3. PROJECT DESIGN

For my role, I inherited two web applications which had access to a repository of raw data but required software engineering to display it in a meaningful way.

3.1 Requirements

The first application was a website that displayed the network information of organizations, such as autonomous system (AS) information and associated IP prefixes. Most of the data could already be searched through by AS numbers, IP prefixes, organization name, city, and country locations, however, the search parameters did not always return the expected results. My task was to refine the search results, implement a description search of the data, and allow combination searches of the parameters, such as searching name, city, and country at the same time.

The second application was a Python Flask file search that returned documents containing the keyword search terms. The requirement was to return 500 character text block previews of files where the search term was found and allow pagination through each of the snippets per file. Additionally, the app needed to store the last search result upon returning to the home page.

3.2 System Architecture

Both web applications were locally hosted within Docker containers on EC2 instances with configured IP addresses, ports, and allowable traffic. The apps utilized the Python Flask framework and were deployed on the app server using Gunicorn. In the case of both apps, the data was already collected. The first application presented data from a combination of Elasticsearch and JSON files. The second application drew data solely from text files within Elasticsearch. A limitation of the system is that all the technology was already pre-configured.

3.3 Challenges

The first challenge was choosing the backend data structure to manipulate the data. Since the search results pulled from a combination of data from the JSON files and Elasticsearch results, the data from the two sources needed to be combined. It was a question of whether to do that in the middleware or on the backend. Additionally, in the given code in the middle of pipelining the code to the frontend the JSON data was converted into a Pandas dataframe.

The second challenge was learning the Elasticsearch API for Python Flask. There were three versions of documentation potentially applicable: the Apache Lucene query syntax, the Elasticsearch syntax which wrapped the Lucene query, and the Python Elasticsearch API which used a wrapper method for Elasticsearch queries. It was difficult configuring the Lucene or Elasticsearch syntax to be compatible within the Python API for complex Elasticsearch queries.

Another challenge was understanding the data. The first web applications required a knowledge of computer network architecture such as how AS related to prefixes and how IP prefixes related to IP addresses in order to return the correct results. The second application required recognizing whether the snippets of text returned were correct based on reading the contents of the file.

3.4 Solutions

For the first challenge, the solution was to communicate with my team, specifically the developer who wrote the original code and other members who were familiar with the data setup. From this process, I was able to better understand the abstractions of the code and functions of different sections.

I addressed the second challenge through independent research. The Python Elasticsearch API could perform complex queries through either using Lucene or Elasticsearch, however, the query had to be broken up into the Python method parameters individually, or put into the body argument of the method and could not use other arguments. I discovered this through referring to existing documentation. I also consulted members of my team to learn of relevant documentation such as the Lucene syntax being the source code of Elasticsearch.

For the last challenge of understanding the data, I performed white-box testing with each function in order to verify the results and understand where the data was being pulled from. As I progressed, I sought guidance from my team members and the analysts to ensure I was returning the results they wanted and to ask if there was anything that could be further improved. A team member helped systems test the application and smooth out areas that could be refined.

4. RESULTS

When the web application met the user specifications, I verified the functionality with the requesting client. With their approval, the changes were ready to be pushed to the production server.

To communicate the new changes with the analysts, I updated the sites to display an announcement banner attached to session data so that it would only popup on opening a new tab, and they would be able to read an update of the latest changes.

Since our software is created in-house, we can choose what technologies to use to incorporate into our tools, considering factors such as cost, accessibility, usability, and the integration with our other services. Our custom tools can be readily adapted to different tasks shared across platforms and streamline other processes.

5. CONCLUSION

Developing software at CACI International, Inc. taught me important soft skills and gave me valuable practical experience. Through developing the web applications, I was able to go through the software development lifecycle from gathering requirements to testing and maintenance. My role at the internship was the first time I was able to inherit a medium-sized code base to

interpret and build off and work with industry technologies. My contributions to the project were able to be published live directly after I finished writing the code or optimizations, offering immediate benefits to the analyst's work.

By developing the web applications according to user specifications, I was able to take on software development projects for the team so members could focus on other tasks. In addition, I was able to help around twenty analysts conduct their research by making their data sources more accessible.

The web applications will help the team have their own set of customized off-the-shelf tools that can be assured of their data security. The software tools serve to automate research tasks, speeding up the process and making the data more accessible, allowing analysts to reach meaningful conclusions about data securely.

6. FUTURE WORK

Research needs will change over time as technology evolves and more data is collected. As such, there will be a continual need for more tools that streamline the team's operations. I am currently working on the next software application, a video transcription and translation service that utilizes OpenAI's Whisper machine learning model to make video data accessible as well. Additionally, the team has more apps that require software optimizations or to be built entirely from scratch. As the collection of software tools grows, they can likely be further packages as a useful product applied to more projects.

REFERENCES

Bujlow, T., Carela-Espanol, V., Lee, B.-R., & Barlet-Ros, P. (2017). A survey on web tracking: Mechanisms, implications, and defenses. *Proceedings of the IEEE, 105*(8), 1476–1510. https://doi.org/10.1109/jproc.2016.2637878.

Kononenko, O., Baysal, O., Holmes, R., & Godfrey, M. W. (2014). Mining modern repositories with Elasticsearch. *Proceedings of the 11th Working Conference on Mining Software Repositories*. https://doi.org/10.1145/2597073.2597091.