Building a Graph-Based Database and API for Real Time Analysis of User Behavior

CS4991 Capstone Report, 2023

Srinija Chelimilla
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
slc8kf@virginia.edu

ABSTRACT
As a software engineering intern under an unidentified company, my team and I built a custom graph-based database and analytics API to examine user behavior. Implementing this capability was a multi-step process that began with the provisioning of Amazon Web Services (AWS) resources for cloud deployment ("Cloud Computing Services", n.d.) and Docker to containerize our application. Half the team worked on the data ingestion side, while the other half worked on developing the graph-based database and building the API. While the company is already utilizing existing tools to address this topic, this project is especially valuable because it processes and retrieves data in real time, enhancing the efficiency of the current solution. The next steps for our project would be to push our application into production and address any production related issues or bugs that may arise. Future developments would likely include adding more analytical capabilities in the API and a user interface to display the API responses.

1. INTRODUCTION
Security has the power to either make or break any company's foundation, as a business is built upon the trust between the company and its customers or clients. The company I interned for has gone to great lengths to ensure customer security and privacy. Every technology or framework used in a project must comply with the company's security policies.

My team and I were tasked with building a custom graph-based database and analytics API to examine user behavior in real time. While graph databases exist, we were told that none of the existing graph databases met the security standards of the company, and thus we had to make our own custom implementation. The existing ingestion process the team was using had a higher latency. Our team had to build a similar ingestion process, but data needed to be accessible in real time.

2. RELATED WORKS
The project outline we were given compares and contrasts two structures of storing data—relational databases and graphical databases. It outlines the advantages and disadvantages of each approach. The team identified that current graphical databases are not in accordance with company standards due to their lack of security. This information, presented by my manager, fueled the team's project of creating a custom graphical database to store user data.

3. PROJECT DESIGN
The goal of our project was to build an API that returns key information about user behavior.

3.1 Review of System Architecture
The project had two key components: the ingestion process to consume relevant data and API development to query and retrieve appropriate responses.

3.2 Requirements
This project was more research and development heavy, so there weren't any strict project requirements. As long as the project met the company's standards and the API returned the desired results, our project would be considered as meeting the requirements.

3.2.1 Client Needs
The client for our project is the Machine Learning team. The Machine Learning team wanted a tool that efficiently queried and analyzed certain user data, took input for specific parameters, and returned data about relevant patterns. The need for this tool is critical to analyzing user behavior in real time.

3.2.2 System Limitations
The limitation of the existing system is that it had a higher latency in the ingestion process. The system our team was working on would provide results in real time.

3.3 Key Components
There are two key components for this project: the ingestion process and API development. I will be further describing the specification of each of these components.

3.3.1 Specifications
The ingestion process involved consuming data from a data source and writing the data into an AWS relational database. The API development part of the project entailed creating a graphical database using NetworkX (NetworkX documentation, n.d.) to grab data from the AWS relational database, and developing the API using Python. The API had several endpoints, each querying and returning different metrics.

3.3.2 Challenges
My team and I ran into multiple challenges throughout our internship. This project had very loose specifications as our manager wanted us to experiment with different technologies and approaches. Our manager suggested the specifications above for our initial approach to the project.

The first challenge our team faced was the approval of AWS gen 3 accounts. AWS Glue is a relatively new tool adopted by the company and so our team had to request permission to access this resource on the gen 3 account. Another challenge we faced was learning how to use Bogiefiles, an internal tool used to provision AWS resources. A large part of our internship was focused on deploying resources through this tool, so it was imperative that we understood how to use it effectively and efficiently.

3.3.3 SOLUTIONS
The solution to the gen3 problem was to switch to gen 2 accounts and use a queueing service to consume data as opposed to using a Glue job. The solution to the Bogiefiles problem was to just keep attempting deployment and figure out how to fix error messages until the pipeline succeeded. The approach to fixing specific error messages was to read documentation and ask for help in the appropriate Slack channels.

4 RESULTS
As per company standards, an individual or team needs to follow a process to implement a new project or feature. Therefore, while my team and I finished the deliverables and presented a working project, there is still a lot to be done before our project can be adopted and used internally. Our project reduced the latency of consuming and analyzing user data

to real time. Because this was such a novel project, its efficiency and accuracy and its effect on the Machine Learning team that intends to use it can only be determined once it has been fully deployed.

5 CONCLUSION

This project is valuable because it assists the machine learning team to analyze user behavior in real time. A key feature of our project is the system’s low latency. Data is transferred in real time. Security is an important aspect customers want from any company. Because the API we have developed helps the company engage in secure practices, I believe this will indirectly be valued by consumers as well.

6 FUTURE WORK

There are several next steps that can be taken to complete the project and expand on it in the future. The first step would be to fully deploy all components to production. Given the limited time we had in our internship, we were only able to push our data ingestion component to production—our API is only deployed to development. I believe this is the main to-do that needs to be completed in addition to other minor administrative tasks. In terms of future expansion, a few objectives we could focus on are creating a user interface for the API, deploying our API across multiple availability zones in AWS, and consuming from multiple data sources.

REFERENCES

Cloud Computing Services—Amazon Web Services (AWS). (n.d.). https://aws.amazon.com/

NetworkX documentation. NetworkX. (n.d.). https://networkx.org/