Software Engineering: Developing Frontend and Backend Resources for Geospatial Application

A Technical Report submitted to the Department of Computer Science

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Zachary Lieberman

Spring, 2022

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Briana Morrison, Department of Computer Science

Software Engineering: Developing Frontend and Backend Resources for Geospatial Application

CS4991 Capstone Report, 2023

Zachary Lieberman
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
zl5nrs@virginia.edu

ABSTRACT
A Northern Virginia-based government contractor required additional functionality for their geospatial web application in preparation for a future contract. To meet this need, I utilized both frontend development and backend cloud technologies to enhance the application. For the frontend development, I used the React framework along with Material UI styling to improve the user interface and integrate additional features. In the backend, I created cloud-based microservices to produce various tools for the application that run concurrently for different processes for the front end. My additions to the application include the integration of new Material UI styling and the creation of new tools and widgets that significantly improved the user experience. Additionally, I reintegrated an out-of-date alerting microservice and developed a microservice that translated binary location data into usable map layers. Future functionalities in the project will focus on creating additional widget features and services in accordance with the dynamic needs outlined by the requirements of the future contract.

1 INTRODUCTION

The goal of any product is to supply users with something new and practical that makes a difference in their lives. These products must undergo multiple forms that culminate in something that is transformative and fulfills the needs of the target audience. My division was responsible for internal research and development on future projects that the company foresaw bidding on. This process still reflects the same processes as current contracts using the software development lifecycle with an extended timeline and flexibility.

At the beginning of this project, I was tasked to develop various tools in order to supplement the future goal of a valued geospatial web application. This began with my placement on the base application team, developing and refining tools that could be used in other specialized applications. These applications have a similar interface and widget toolset; however, each is particularly tailored for a different purpose. This meant that any changes made by my team on the base application would be copied to the spinoff applications, thus emphasizing the importance of good software development efforts.

2 BACKGROUND

The frontend development of this project contained all of the information visible to the user: content, styling, and product flow. The JavaScript React framework was used to create content, run automation, and provide feedback to the backend based on

users' actions. React was also used to display component content based on user input while maintaining user data. Styling was utilized through CSS and Material UI to create a symmetric appeal for all tooling throughout the application.

Microservices were the foundation of the backend development allowing multiple smaller modules to be developed concurrently. These modules allow members of the team to develop additional functionality with smaller codebases and minimized errors. If an error arises, only one portion of the application will be affected allowing for easier troubleshooting and streamlined integration for developers. The microservices developed for this project use a proprietary foundational service that uses the Java language.

3 RELATED WORKS

In order to understand the basis for the work done in my experiential learning event, it is important to understand the innerworkings of software development in government contracts. Wrubel & Gross (2015) discuss the objectives, misconceptions, and agile development in government contracting. They first posit that Agile development is crucial in government contracting versus traditional development due to the ease of creating additional requirements and constant customer involvement. They outline how with changing requirements happening frequently in these contracts it is imperative that new requirements evaluated by the clients can be created with continuous involvement. These objectives are set at the start of the contract, however with Agile development the focus is set on "working software" that shifts based on changing requirements. The contracting officers whom work with the programming office to oversee program performance are given "a wide latitude to exercise business judgement" and thus are allowed to make quick changes to the project structure that is facilitated in Agile development. Wrubel and Gross also shine a light on the common misconceptions that are made with Agile development in government contracts. They thoroughly justify that Agile is indeed able to produce enough documentation with investigations and information hubs such as Confluence. They also describe how Program Design Reviews allow insight and oversight into the workflow which quells worries that Agile does not offer enough of a review on program quality and progress.

Dragoni, et al. (2017) provide a thorough overview of microservices evolution over time and their potential future impacts. One of these emerging trends that was also used in my development in this project is the widespread adoption of containerization. Containerization allows developers to group and deploy microservices easily while providing symmetrical experiences for all users. They also discuss event-driven architectures similar to Amazon Web Services which allow the microservices to communicate asynchronously and even efficiently scale resources automatically due to user needs.

4 PROCESS DESIGN

Over the eight months that I was a part of the base application development team, I worked to improve the product towards the goals of the future contract.

4.1 Agile Development

My division utilized agile development with biweekly sprints aimed at tackling specific tickets in a short period of time. At the beginning of a sprint, the team met to discuss new tickets and graded them based on estimated time and complexity. Following a sprint, the team reviewed completed work, discussed investigations

into new work, and surveyed how to better improve on the next sprint. Every day our team had a daily standup meeting, led by the program managers, where we discussed our individual statuses and relevant action points in the project. This allowed for cross-collaboration and congruency in everyday work.

4.2 Creating Frontend Tools

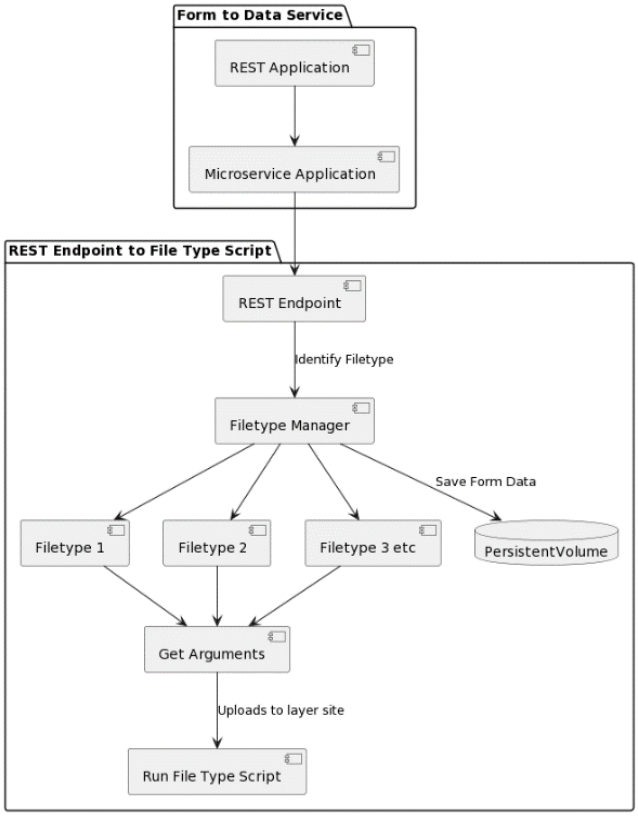
One of the first tickets I was assigned consisted of integrating a full-screen mode into the map component. The first step in the process was to find the best module in terms of functionality, simplicity, and security for the application. After combing through the various options, the module that I chose was react-full-screen which I added using the yarn package manager. A major issue that arose during this ticket was that when the full-screen button was pressed, each widget needed to be configured to properly function while entering and exiting correctly. To address this case, I added additional functions to each widget that controlled the widget's visibility based on the stored viewing state.

Following the full-screen button ticket, the largest frontend ticket that I completed was reintegrating an alert button that was previously started in a previous edition of the application. This functionality was pursued in an investigation but was never added to the application, so I needed to complete the work that was done and add a button with accurate alerts. I started by reading over all of the work that was previously done including the code added, comments in the many commits, and the writeup that covered the goals of the investigation. Then I went into the service and updated the dependencies to the current configurations and linked in the alert objects that were already being displayed when an action warranted them. These alerts included if a service went up or down and object creation or deletion. The final button displayed the number of new alerts and a dropdown with each alert in the current user session.

4.3 Developing a Microservice

My biggest task throughout the experiential learning event was creating a specific microservice that would be used by another team. This microservice would work with a geospatial data server to apply plume layers to the map from binary data. As seen in Figure 1, the process flow consists of a form being sent with the user's metadata and binary data to a REST endpoint, parsed for the particular file type, then running the correct script that converts the binary data to the chosen plume data and uploads the data to the geospatial data server.

Figure 1: Microservice Flow Diagram



I began by creating a form within the application that took in all of the required metadata fields and an upload button for the binary data. The form then would send the data to a rest endpoint through a post request. Then I created a microservice that received the data at the endpoint and parsed the arguments to identify the file type that was requested. The parsing of the arguments utilized a manager class that identified the file type and saved all data in case of errors during execution. I used the Strategy software design pattern, which allows the microservice to choose the correct class to execute at runtime, that correctly parsed and ran the script based on the identified file type. This caused the converted files to be uploaded to the geospatial data server that users could apply to the map in the application.

5 RESULTS

The results of my efforts throughout the experiential learning event were crucial for the base application team and the spinoff application teams. The full-screen functionality was adopted by all other teams that utilized the geospatial map which allowed them to provide additional content for their users. With an easier viewing format, teams were now able to view the map and its widgets in a larger and more in-depth perspective when creating and testing tools which boosted team productivity. The alerting service and button were also adopted by the other teams which allowed them to identify all alerts in the application. The benefit of the alerts was that teams could now troubleshoot errors quicker in their work and provide an additional feature for the users. The work done in the plume data microservice was crucial for the other team's application. This fulfilled a contractual requirement for plume data functionality while allowing the team to work on other features in the process. The microservice was immediately integrated into their codebase and is used on a frequent basis in their application.

6 CONCLUSION

The work that was done in my experiential learning event created additional functionalities for our department's applications and increased work productivity for team members using the platform. The frontend development created two crucial tools that created additional capabilities for the users. The backend development allowed another team to integrate plume layers into the geospatial map which significantly enhances the capabilities for future users. This experience gave me the foundation for future work in software engineering while allowing me to contribute to an interesting and important project.

7 FUTURE WORK

When I left the team there were discussions on the future of the project. The major changes that would be introduced were an overhaul of the organization of the tools within the codebase. This entails modularizing the widgets and features into core blocks which would allow for condensing the large code base, removing repetitive code, and creating a cohesive styling throughout the application.

REFERENCES

Dragoni, N., Lafuente, A., Giallorenzo, S., Mazzara, M., Mustafin, R., Safina, L., & Montesi, F. (2017). Microservices: Yesterday, Today, and Tomorrow. In: Mazzara, M., Meyer, B. (eds) Present and Ulterior Software Engineering. Springer, Cham. Retrieved April 21, 2023, from https://doi.org/10.1007/978-3-319-67425-4_12

Wrubel, E., & Gross, J. (2015) Contracting for Agile Software Development in the

Department of Defense: An Introduction. Software Engineering Institute, Carnegie Mellon University, 2015. Retrieved April 21, 2023, from https://apps.dtic.mil/sti/pdfs/AD1046628.pdf