Southern New Hampshire University

8-1 Assignment: Final Reflection

Andreas Galatis

Professor John Watson

CS 470 – Full Stack Development II

February 26, 2023

Final Reflection

Link to presentation on Cloud Development: https://youtu.be/uUwI53JS9so

Experience and Strength

In the course of 8 weeks, we migrated a full stack application to a cloud-native web application using AWS microservices. We began by focusing on the concept of containerization and learned how to orchestrate multiple containers with the use of Docker Compose. We then learned the tools needed to use AWS' serverless architecture. We obtained some valuable skills in how to utilize Amazon's Simple Storage Service (S3) which is a cloud-based storage service. We also learned how to implement a Lambda compute model in conjunction with Amazon Gateway API to make REST API calls to endpoints. We utilized DynamoDB for the database portion of the application.

This course, along with its predecessor, Full Stack Dev I, were instrumental in my skill development in front and backend development with JavaScript frameworks such as AngularJS, Express.js, and Node.js. Building and consuming RESTful web APIs, and data modeling with MongoDB and DynamoDB.

Planning for Growth

What works well for complex and evolving applications are serverless microservices. Their modularity makes them easier to manage and scale. Functions can be integrated with databases and API management tools for different use cases. The functions and resources that are used in one microservice can serve as the foundation for other microservices. Error handling in a serverless architecture can be done with Step Functions. AWS, Step Function feature, for example, is a great benefit when building a serverless application to deal with errors and retries properly.

When planning for expansion by using serverless it is important to consider the pros and cons. Some of the benefits will be a reduced time to market and quicker software release. There will also be a lower operational and developmental cost, especially with AWS' pay-as-go model which means that costs will always remain at scale as you only pay for what you use. Some of the drawbacks of serverless is that it may not be efficient for long-running application, as in certain cases, using long tasks can be much more expensive that running workload on a dedicated server. Lastly, it's important to consider that your application will be completely dependent on the third-party vendor that is hosting it.