Project Two
CS470 Final Reflection

Jesse Peterson

https://youtu.be/efwOsQpnNpc

CS470 FullStack II

Professor Johnson

December 9th 2024

Experiences and Strengths

Working on this project with AWS I have learned many new concepts, some being: Containerization, Cloud Security, and Serverless and microservice architecture. First, we learned how to employ Docker and docker compose to transform an angular application into a container. We were then able to take our image and implement it into an AWS S3 bucket. Second, in our project we managed the security settings of our S3 Bucket, Lambda Functions and learned about IAM policies and roles. Finally, we learned how to employ the serverless model through the usage of AWS Lambda and the API gateway.

After taking this course, I can confidently say that my strengths as a software developer are problem-solving, adaptability, and my technical experience. I have obtained the ability to look at a major project such as this and analyze the requirements and design a scalable solution. I have become very adaptable to managing both the front and back-end, now I am even comfortable managing a cloud-based application. Lastly, my background knowledge of programming languages, Database systems, and Linux have helped me tremendously with understanding issues in a timely manner.

The roles that I am prepared to conquer with my newfound knowledge could either be a Full-Stack developer, SRE (Site-Reliability Engineer), Cloud Engineer, or Backend Developer. Furthermore, I have a keen interest in Full-Stack and SRE work.

Planning for Growth

Microservice or Serverless may be used for efficiencies in things such as handling scaling and error handling. Firstly, in this framework, auto-scaling can be handled via the AWS elastic load balancer. This ensures that the application will scale on traffic spikes. Secondly, through the implementation of retry policies and logging through cloud watch. Error handling can be automatically handled and logged for further investigation.

Another important part of AWS growth is managing or predicting the cost. Luckily, AWS provided a tool called the AWS price calculator, which can help you to plan your project wisely.

Comparing the cost predictability between containers and serverless applications. First, I can say that containers are far more predictable because of their consistent workloads. On the other hand, serverless is better for applications with irregular traffic, since users are billed only on utilization.

Here, I will compare the pros and cons that can be deciding factors when it comes to planning expansion with serverless or container solutions.

1) Pros
    a) Serverless
        i) No upfront infrastructure cost
        ii) Simplifies operations, allowing developers to focus on house
    b) Containers
        i) Better control over environment and dependencies
        ii) Suitable for stateful applications
2) Cons
    a) Serverless:
        i) Limited execution time and resource containers per function
    b) Containers:
        i) More operational overhead for scaling and managing clusters
        ii) Required expertise in orchestration tools like Kubernetes

Finally, Elasticity is a feature of the cloud format that ensures resources are automatically adapting to workload demands. This prevents over or under provisioning. Another concept, Pay-for-service, enables cost control by paying only for what is used. This is performed via lambda functions, which is a good solution for unpredictable workloads. These two concepts play major parts in future growth decisions. For instance, Serverless is the proper solution for rapid scaling and lower operational effort. On the other hand, containers are a good solution when you need more control over deployments or have consistent traffic patterns.