Name: Raphael Rose
Date: June 27th 2021
Course: CS 470 Full Stack Engineering II
Assignment: Assignment name: CS 470 Final Reflection

Link: https://www.youtube.com/watch?v=Rt7unqAbFRA

- **Experiences and Strengths:** Explain how this course will help you in reaching your professional goals.
  - What skills have you learned, developed, or mastered in this course to help you become a more marketable candidate in your career field?
  - Describe your strengths as a software developer.
  - Identify the types of roles you are prepared to assume in a new job.
- **Planning for Growth:** Synthesize the knowledge you have gathered about cloud services.
  - Identify various ways that microservices or serverless may be used to produce efficiencies of management and scale in your web application in the future. Consider the following:
    - How would you handle scale and error handling?
    - How would you predict the cost?
    - What is more cost predictable, containers or serverless?
  - Explain several pros and cons that would be deciding factors in plans for expansion.
  - What roles do elasticity and pay-for-service play in decision making for planned future growth?

Experience and Strengths:

In this course I have learned the process of developing and scaling a web application with the use of AWS for providing support and infrastructure. We have learned the process of implementing lambda functions, S3 object storage, AWS Gateway, and IAM policy management.

As my strengths are concerned I feel that I have learned and advanced my abilities through this course in terms of process management and configuration. This class has been a great learning in understanding process documentation and leveraging it for future work. Where I think this will suit me well is that as I learn these new skills they become highly repeatable and easy to memorize. On just this activities I was able to memorize and master the steps of creating lambda functions. Now as I am tasked with building said functions it will become easier to iterate and explore features.

As roles are concerned I would feel comfortable taking on work as a cloud platform engineer or full stack developer. Im also interested in the CD/CI field where you integrate software together to make functioning platforms.

Planning for Growth:

Through this course I have learned about the various ways in which microservices can be engaged. Specifically they are small portable groups of code that reduce the potential for errors and downtime in a serverless build. By independently hosting each service the application becomes the sum of its parts but with more agility. If for instance I have a microservice that is dependent on a lambda function I can edit that lambda without impacting the entire application.

As it relates to scaling an application cloud platforms offer dynamic scaling which I believe can be configured within bounded parameters. So as demand on the platform expand so to does the application. Kubernetes is one such example of this type of scaling in a container environment. Further error handling is one of the better integrated features in the cloud architecture lifecycle. You can easily monitor error and adjust/rollback/or push out new versions easily.

To estimate cost of an application there are a lot of calculators out there especially when migrating existing legacy application. For a from scratch model I believe the cost can be modeled as storage costs and then estimated retrieval costs based on usage. This then can scale dynamically so once you know the cost to run an application for the minimum number of consumers you can scale it easily.

In terms of cost predictability I believe containers would be a more predictable model because they scale more statically. However, serverless scales more dynamically so it would ultimately be more cost efficient. For instance in a container environment cost will jump with each scale in containers so it may have a static base cost like $4 and then each container you add will add a fixed volume as well as an additional cost of $4. But if you only need $6 of services serverless will be better because you can buy essentially in increments that are less than 1 unit.

Elasticity and Pay-for-services offer an abundance of value in a scaling model. In the event that dedicated servers are not needed then these models both provide the user the ability to easily scale to the demand needs of the application. As you plan growth you don’t have to plan expansion which can significantly reduce cycle time since resources can be focused on development over expansion.