Cody Gregory
CS-470-T3366 Full Stack Development II 22EW3
2/24/2023

CS 470 Final Reflection
Presentation Link: https://youtu.be/UsRCD27W9Jw

Experiences and Strengths:

Perhaps one of my biggest concerns coming into CS 470 that has been with me throughout most of my journey as a CS student has always been on the practical side, “I know how to code, but do I know how to put it together to make a usable application?” Having now completed this course I can now answer that question with a yes.

When I reflect on the skills I have developed from this course I am confident that they have made me a more marketable candidate in the software development field. It is no stretch to say that cloud development is the way of things in our modern world, and the familiarity I have gained with the various AWS services used on our application (S3, API Gateway, Lambda, DynamoDB) will serve as a great foundation for working on cloud applications in the future. While we may have been limited to AWS the core concepts apply to other cloud providers as well such as Google Cloud and Microsoft Azure.

As for my strengths as a software developer I feel that I possess a strong foundational knowledge that when applied allows me to tackle any problem I may come by. This could be through my ability to break a large problem into smaller ones, knowing how to ask the right questions so that when I do seek help the process is efficient, or even just through best practices such as maintaining clean and organized code.

With the aforementioned in mind, I would suggest for myself a role as a junior software developer, a position where I am able to learn the flow of a live development environment and can start building experience so that I may progress my career into more senior positions that include more responsibility where I would be able to use my accumulated experience to oversee others and have a greater impact on the quality of any given development project.

Planning for Growth:

Through microservices and serverless architecture cloud services offer various ways to improve efficiency of management and scale in web applications.

In our project we utilized containerization through Docker. This approach allowed us to break down our application into separate components and provided the capability to scale the application in parts. On the cloud via microservices and serverless architecture the concept remains the same, since each component is run by a different service we are able to ensure that the application can handle any traffic that might be thrown at it without downtime or performance lag. This concept does not only lend itself to scaling however but also error handling, as like containers, individual services offer isolation which allows for an easier time identifying issues.

When it comes to cost prediction, microservices offer real-time estimations of usage through constant monitoring of your resources. Alarms can be set to inform the developer of any caps they might want to set for themselves to ensure remaining in budget, and through the pay-for-use model you can be rest easy knowing you will only pay for what you need, which may not be the case in a local setup where if you over or underestimate your infrastructure you will run into problems. To this end it is undoubtedly more cost predictable to operate on a serverless architecture than via containers. While both offer similar conveniences due to their individual components, serverless features the ability to only pay for usage while containers still require a set capacity.

In general various factors need to be weighed to determine if expansion is worthwhile. The obvious drawbacks are increased cost and complexity. In a local setup this would involve increasing infrastructure and associated resources which means more time and effort will need to be placed on tasks that are indirect to the goal of the development. The upside to this however is the increased capacity for business which depending on the market may prove to outweigh the drawbacks. This is where the concept of elasticity and the pay-for-service model of the cloud shine. When we use the word elasticity what we are referring to is the ability to freely

scale resources to meet demand, for example let's say you are operating an application for a landscaping business, during the spring and summer months you're traffic is at a peak, but during the fall and winter you're traffic drops by two-thirds, as is the nature of the business. Through elasticity and the pay-for-service model of the cloud you don't need to worry about potentially wasting resources on infrastructure that is not needed for half the year, and instead can adapt and pay more for when you need it (the active months) and less when you don't (fall/winter). This plays a big role in planned future growth as it simply removes the planning. The process is streamlined, you don't need to concern yourself with infrastructure and can focus on improving the application or directly on the business at hand.