Thomas Bartlett
March 3, 2024
Assignment name: CS 470 Final Reflection
https://youtu.be/Ip3MYmg5O-k

Throughout this course I have had the opportunity to refine and learn skills relevant to full-stack development. I have learned the skills needed to take code and containerize it with tools like Docker. I also had the opportunity to learn about AWS and how to deploy a web application using the different functionalities available.

Throughout my time here at SNHU I have found that some of my strengths as a developer include the ability to pay attention to details, problem solving, prioritizing tasks, and learning new concepts and skills quickly. I feel with my strengths and skills that I have acquired there is a wide range of roles that I could pursue such as an embedded software engineer, full stack developer, or mobile software developer.

When speaking about cloud services, they offer a wide range of benefits for web applications which include scalability, flexibility, and cost-efficiency. Microservices and serverless architectures are two different approaches that developers use to achieve efficiencies in management and scale. When it comes to error handling and scaling, it would be handled differently depending on the approach used. With Microservices scaling is done through scaling individual components as needed. With applications broken down into these smaller components, error handling is made easier, being able to isolate and troubleshoot issues. Looking at serverless architectures, scaling is automatic and is triggered by demand. In most cases, error handling is taken care of by the cloud provider, but some error handling mechanisms will need to be built into functions. Cost predictions would also be different depending on the approach. With microservices predicting costs can be challenging because of the need to monitor and manage multiple services. With a serverless architecture it is a little more straightforward as costs are based on execution time and resources used, and cloud providers typically provide cost calculators to help estimate expenses. For determining which is more predictable for cost, it really matters the size of the project. With smaller applications, containers offer a slightly more cost predictable model, but due to how applications are scaled in the different approaches, serverless architectures may be more cost predictable as the application scales.

When trying to plan for expansion there are several pros and cons to consider. Probably the biggest con that every company brings up and looks hard at is cost. Some questions often asked is what is the upfront cost, where may there be unexpected costs, and are there alternatives that may be more cost efficient? Another con could be as the application expands will the need to add more staff to support it arise. This also kind of goes with cost as well, as new staff means more salaries paid, cost of hiring, and cost of bringing the new staff up to speed on the current

code base through training. Although these are seen as possible cons, the possible upside could be enough to overcome these concerns. As the application expands, it offers up the opportunity to increase revenue offsetting the cost of expansion. Another pro of expansion also provides a model for future expansion. If the expansion is ultimately successful, there is now a base of how to expand the application. This can be refined to adapt to hiccups that may have been encountered, but shows that expansion is possible.

Elasticity and pay-for-service should play major roles in decision making when considering future growth of an application. Elasticity of an application has a major impact on the performance of the application. The less elastic an application is, the less efficient the performance will be. As the application expands, it will become less and less efficient if the application is not elastic. If you are using a pay-for-service model, it does make it easier for future growth, because scaling is automatic and although the planned growth is in place, the cost will not go up unless the usage goes up. Also, if the usage goes up and then starts to decline again, the pay-for-service model helps to have in place. If it is not, you are still paying the higher cost after expansion, even if there is a decline in use.