Michael Devin Moore

08/24/2025

CS 470 Final Reflection

<https://youtu.be/loCUC_zfLWw>

* **Experiences and Strengths:**

Through the Full Stack Development II course I applied all of the concepts I have learned throughout my time at SNHU at combined them once again, as I did in Full Stack Development I, with the addition of creating a cloud-based application. The concepts of creating an application’s frontend, backend, and database were highlighted in this course and I have shown my proficiency in taking a base set of code and expanding upon it to create something reminiscent of what could be seen in a future workplace. In keeping up with modern programming innovation, this course also set up the foundation and developed my proficiency in working in a cloud for all facets of a full stack application. I specifically worked with various Amazon Web Services (AWS) assets to create the application furthering my desirability to employers as with this course as I have shown proficiency in local full stack development, throughout my academic career and in Full Stack Development 1, and cloud development specifically in this final course. My ability to quickly grasp code that is completely new to me, make sense of it, and be able to develop it further are but a few of my biggest strengths as an intelligent and now educated software developer. With my academic career reaching this milestone I find that I am ready to approach any developer, tester, or security manager role with a level of understanding and experience with each I know I could succeed and thrive in any position such as those.

* **Planning for Growth:**

The innovation that is serverless, or cloud-based operation, provides quite a lot of benefits to a developer on many levels. Scaling a product with a serverless environment removes most of the burden that a traditionally hosted application would have with regards to managing the application, the network, security, and it changes the dynamic of how costs are added up as well. Scaling is performed automatically in AWS to meet the demand of the application at hand. The cost that comes with that is based on various factors that can be summed up by resource usage and overall traffic to and from the application. While this is the same consideration of a traditional locally hosted application, serverless provides the benefit of having the hardware requirements being met by the likes of AWS or other options. This drastically reduces startup costs and manpower requirements as most of the application management is covered by the serverless provider. Otherwise, setup of the application locally requires a lot more initial funding and expertise to get going. Setting up a local container-based application may be more cost predictable, but with the plethora of monitoring capabilities on AWS it is possible to determine costs for operation of an application and therefore cost prediction becomes easier.

Error handling and testing become more simplified as all the assets on AWS are separate objects that are correlated to function together, therefore testing the functionality of the application is already conveniently separated into partitions, simplifying finding and correcting errors. The biggest pro that comes with AWS is that the application at rest is secure and backed up by AWS itself and the process of setting up in motion security implementations is very intuitive and simple with the assets I personally worked with. While I am not an expert with AWS, nor what I worked with in this course, I do possess the ability to expand it to whatever the situation calls for. However, if expansion of said application reached a point where other developers would need to get involved, they would need training and proficiency with AWS as I formed to be able to work effectively. However, with how simple it was to get going in AWS, that con is relatively minute.

Since AWS works off of elasticity and pay-for-service, which I mentioned with the automatic scaling and payment based off of resource use and traffic, there are some considerations to make with expansion of an application. Optimization should be the first thing in mind with application hosted in a serverless environment as those tweaks will lower overall operating costs of the application as it becomes more efficient. If resource demand and traffic ever became so substantial that the cost of operation in a serverless environment became hefty in regard to the cost of local hosting and operation, then consideration could be given to transferring to local operations. Once again, the cost of initial setup and pay-for-service is such a boon for starting development teams that it very easily is more cost effective than setting up server operation in a traditional sense. Overall, it truly depends on the scale of the application and if the intent of the application can be met fully through a serverless option. If complete control, monitoring, and operation is desired, the traditional route provides that at that initial setup cost for equipment and manpower.