Jeremy Woods

2/23/2023

CS 470 Final Reflection

https://www.youtube.com/watch?v=x8h4h-CPDxg&ab_channel=JeremyWoods

Prior to beginning the coursework for CS 470 – Full Stack Development II, helped me grow as a software developer my introducing and furthering skills and confidence related to containerization/virtualization and AWS. Through completing this coursework I feel confident in my ability to navigate AWS as well as run applications in a container. This course pushed my confidence further as I spent a lot of time experimenting, trying to figure out what makes something work or why it may not be working. AWS is a great platform for this type of experimentation to take place as I felt as if I couldn't "break" this system, and any accidents could be easily rolled back. This experience has since carried over to personal projects where prior I would jump to forums at the first sign of a problem, where now I've been inclined to reference documentation and experiment to figure out how and why something works the way it does. My strength as a software developer is my ability to not give up, but also knowing when to step away for a moment. I can't count the number of times I've figured out a coding problem by stepping away and then "connect the dots" on a walk to clear my head. While I wasn't able to utilize this skill as much in this course, but being a lifelong musician, I am a very creative person, so I am able to visualize and create exactly what I see in my head in a very pleasing manner. With that being said, the roles I am looking to pursue is a front end or full stack developer.

AWS microservices and serverless computing handles scaling and error handling as much of it is handled automatically. Lambda functions can be configured to handle errors while AWS will automatically handle scaling as needed. AWS offers a pricing calculator where you can enter data for the expected traffic to get an idea of how much it would cost. You can then adjust the data to see how the pricing will change if there is an increase in traffic and vice-versa for a decrease in traffic. However it should be understood that scaling does occur automatically, so prices can dramatically increase on a whim. As no physical resources and hardware need to be purchased by the developing company, utilizing serverless computing can prove to be more cost effective than running a container-based application as containers demand more resources.

Expansion can be a tricky conversation as typically more of anything usually correlates to most cost. However, needing to expand should correlate with more incoming money. A project should have some consideration for expansion at inception, however, if not, depending on how the project has been created it is possible to be considered on the fly if the project is hosted serverless as no additional servers or other hardware resources would need to be purchased. Elasticity and pay-for-service plays great into expansion on the fly as scaling is all automated as needed. For instance, a sudden influx of traffic would be accommodated for by the cloud provider without a disruption to your service, you'll just have to pay a little extra, however, as previously mentioned, an influx in traffic should equate to higher profit! Expansion becomes a bit more tricky if the application is not hosted serverless, as more physical servers need to be purchased, set up, and maintained, creating the need for more physical office space as well as the possibility of needing additional staff to set up and maintain these servers. Physical servers also run the risk of this influx being temporary, then creating a potential waste of funds as the additional servers are not needed anymore. Wherein serverless, you just pay exactly for how much the system was used. Lower traffic in one month? Pay a little less. Higher traffic in the next? Pay a little more, all done automatically without the need for the developing company to allocate physical resources.