Andrew Wepplo

April 24, 2022

https://www.youtube.com/watch?v=jqPskVHC2do

CS-470 Final Reflection

There are several things that I learned which helped me to understand more including, static websites, transferring them to the cloud, AWS, and all the included functionality of it. I had no experience with AWS or any other similar cloud-based services. I find the many ways that a database server, website server, or even APIs can have access to the outside world very interesting. While using AWS, we were able to give access through APIs to a database by using HTTP requests and responses, with all the required roles and permissions. I would say that one of my strengths is the ability to understand the similarities and limitations of different software languages needed within the software engineering field. Also, my attention to detail has always been a something that I feel has me stand out among others.

Some of the benefits of microservices and/or serverless applications is the ability to use APIs as a solution to client and server applications. For example, a website and its pages were transferred to a cloud-based solution and then API's are able to handle the transfer of data links using HTTP request and responses. With it being serverless, the amount of transferred data can be scaled to fit your needs and not have to be paid for up front. To correctly scale in a serverless environment, I would look at the requirements that the specific application is estimated to require, and then choose an initial cloud-based environment like AWS that would support the requirements. A first pro would be that if a website has unexpected scalability needed, (which it most likely would) then serverless would be the best choice. On the other hand, a con would be fact that if the serverless system was ever on a scheduled maintenance period or a time where the system is unavailable, there would be problems connecting to and using the system during this time. Elasticity in a serverless environment means adaptability to adjusting resources as needed that were not expected during early stages of development. Pay-for-service is paying for the resources used once you require them, instead of upfront, and if you are combining both for future growth, these options are both great for future growth and should be considered to be prepared for events in the future and being able to address them effectively.