Courtney Maxwell

12/19/2024

CS 470 Final Reflection

YouTube link:

Final Reflection:

I have learned many new valuable skills that I will be able to use to further my career in mainframe. I have a greater understanding of programming and the software development lifecycle than I did before. I have not quite mastered the art of creating a complex app but I can create a simple app and connect it to databases, serverless storage, and the Internet while simultaneously creating permissions and roles for the app. My strengths as a software developer revolve around troubleshooting and researching errors in the code. In my current role, most of the code already exists as the systems have been around for decades. However, we occasionally run into issues that are unfamiliar to us or the client, and we have to do research to find the solution. After completing this class, I look forward to working in management roles in which I can apply what I've learned. Management typically requires a general understanding of programming to communicate between the developers and the client. I also look forward to working in more technical roles where I can apply the skills mentioned above to gain a better understanding of mainframe technology.

Microservices and serverless can be used to produce efficiencies of management and scale of the application by decoupling services, using automatic scaling, and faster deployment. Scale and error handling would be managed through load balancing, service discovery, and monitoring and controlling of errors. The cost would be predicted by analyzing the app's usage

patterns, researching pricing models of different cloud providers, determining memory allocation, and estimating the average execution time of each function. Between containers and serverless, using containers is more cost predictable. Serverless systems are pay-as-needed so the cost can fluctuate whereas container costs are based on allocated resources whether usage is high or not. Pros include increased scalability, improved agility, cost optimization, and improved resilience. Cons include complexity, vendor limitations, and performance overheads. Elasticity is the ability for a system to change resources and the number of resources used based on demand for those resources. Pay-for-service requires users to pay only for what they use. Combining elasticity and pay-for-service gives organizations the ability to manage resource usage during peak times, paying more during these high peak times, and minimizing expenses when usage is low.