Jorge Torres

10/23/24

CS 470 Final Reflection

https://youtu.be/697tnr91vMo

Experiences and Strengths

This course has significantly contributed to my professional development as a software developer, specifically in cloud computing and full-stack web application development. I have learned skills such as cloud architecture design, deploying applications to cloud platforms, and effectively utilizing cloud services like AWS to build scalable applications. These experiences have not only increased my technical proficiency but also prepared me for roles that demand cloud expertise, such as Cloud Developer or DevOps Engineer.

One of my key strengths as a software developer is my ability to understand and implement cloud-native solutions. I am adept at designing and building RESTful APIs, managing databases, and ensuring the performance and security of applications in cloud environments. Additionally, I have a strong grasp of version control, automated testing, and continuous integration/continuous delivery (CI/CD), which are crucial for maintaining and improving large-scale applications.

Given my experiences, I am ready to assume roles in cloud development, software engineering, or DevOps, where I can contribute to designing, deploying, and maintaining cloud-based applications. My ability to think strategically and understand the big picture of application growth will allow me to support teams in building scalable, reliable, and efficient systems.

Planning for Growth

As my web application evolves, planning for future growth is essential to ensure it remains scalable, cost-efficient, and robust. Leveraging cloud services such as microservices and serverless architecture will be crucial in managing this growth. Microservices can help break down the application into smaller, independent services, which enhances modularity and allows teams to develop, deploy, and scale different parts of the application independently. Serverless architecture, such as AWS Lambda, could be used to handle specific functions without managing infrastructure, improving efficiency and reducing overhead costs.

To handle scaling, I would utilize auto-scaling features and load balancers to manage traffic spikes dynamically, ensuring that my application can scale up or down based on demand. This elasticity is key to maintaining performance while controlling costs. Error handling would be managed through centralized logging and monitoring services such as Amazon CloudWatch, which would allow for real-time detection and resolution of issues.

Predicting costs would involve closely monitoring usage patterns, as serverless services typically charge based on execution time and resource consumption. While containers offer more control over resource allocation, serverless can be more cost-effective for certain tasks because it eliminates the need to manage infrastructure. However, serverless pricing can become unpredictable with higher usage, so a hybrid approach may be ideal—using containers for consistent workloads and serverless for intermittent tasks.

When planning for expansion, one must balance the flexibility and ease of scaling provided by serverless with the control and stability that containers offer. A pay-for-service model allows the application to scale without upfront infrastructure costs, but elasticity and cost predictability are crucial factors in deciding which services to use. For planned future growth, I would prioritize a solution that provides a mix of elasticity and predictable costs, adjusting based on the application’s needs over time.