Douglas Bowers

8/18/2025

CS-470 Final Reflection

https://youtu.be/ZF8oDTtMfEs

Experiences and Strengths

This course has provided me with knowledge and skills that I will carry forward into my career goals. Learning how to build and manage containers using Docker and Docker Compose will be especially useful to me in future projects, since containerization has become a core part of modern application development. Being able to create containers from the ground up means I will be better equipped to design applications that are portable, scalable, and easier to maintain.

I also plan to use cloud computing skills from this course to expand my professional capabilities. Understanding how AWS and related tools fit together will help me approach cloud-based projects with more confidence. These skills are valuable in positions that involve deploying, scaling, and maintaining web applications.

As a developer, one of my strengths is my eagerness to learn. I am motivated to continue building on what I have gained here, exploring new technologies, and applying them to real-world problems. With the foundation from this course, I feel well prepared to take on roles such as Full Stack Developer or Cloud Application Developer, where I can apply both my programming knowledge and my ability to work with cloud infrastructure.

Planning for Growth

Looking ahead, I see microservices and serverless architecture as key tools for building efficient, scalable applications. By using serverless tools, I can reduce the time spent managing

infrastructure and instead focus on writing high-quality code. This will also allow applications I develop to automatically scale up or down depending on demand, ensuring performance and cost-efficiency.

To handle cost planning, I would analyze application usage trends and monitor demand to better predict serverless expenses. Containers will also remain an important option, since they are more predictable in terms of cost and provide consistent availability, though they require more direct management.

Each approach offers advantages and trade-offs. Serverless provides scalability and a pay-for-what-you-use model, which will be useful in projects with fluctuating traffic. However, for more complex applications that require always-on availability, containers may be the better choice. Going forward, I plan to use a mix of both, selecting the right tool for each project's needs.

Elasticity and the pay-for-service model will play a central role in my decision-making. For future growth, being able to dynamically scale applications while paying only for the resources I need will allow me to design systems that are both cost-efficient and adaptable. These strategies will help me plan for long-term growth while ensuring that applications remain reliable and responsive.