Hunter Richards

10/19/2025

CS 470 Final Reflection

YouTube Video Link: https://youtu.be/KmjNh1lcEGI?si=fPQDWZXcNrRFe1YX

Summary of the Project and Problem It Solved

CS-470: Full Stack Development II allowed me to expand my cloud development skills while completing a hands-on migration project. The final project involved moving a full stack web application into a cloud environment using Amazon Web Services (AWS). The goal was to transform a traditional web stack into a serverless, scalable, and cost-efficient architecture that could operate reliably in production-like conditions. This included implementing Lambda functions for compute, API Gateway for routing, DynamoDB for database storage, and S3 for static web hosting. By completing this project, I learned how to efficiently deploy, manage, and scale a full stack application using modern cloud technologies.

Experiences and Strengths

This course has significantly advanced my understanding of cloud-based development and reinforced my skills as a full stack developer. I developed proficiency in configuring and managing AWS resources, including IAM roles, Lambda, API Gateway, DynamoDB, and S3. I also learned how to containerize applications using Docker and orchestrate them with Docker Compose to support microservices architecture.

My strengths as a developer include adaptability, analytical thinking, and a strong ability to bridge the gap between front-end, back-end, and cloud infrastructure. I excel at designing systems that are both scalable and maintainable while maintaining clean, organized, and well-documented code. These strengths position me well for roles that require cloud integration, full stack design, or DevOps-oriented problem solving.

In a professional setting, I am prepared to assume roles such as Full Stack Developer, Cloud Developer, or DevOps Engineer. I can contribute effectively to projects involving modern web application architecture, API development, and cloud automation pipelines.

Planning for Growth

To prepare this application for future growth, I would continue leveraging microservices and serverless design patterns to improve scalability and management efficiency. Serverless compute through AWS Lambda allows the application to automatically scale based on demand while reducing maintenance overhead. This eliminates the need to manage underlying servers, freeing resources for development and innovation.

Error handling would be managed through AWS CloudWatch and Step Functions to provide monitoring, retries, and alerts for failed processes. For scaling, AWS automatically provisions additional resources as needed, allowing the system to handle increased user load without manual intervention. Cost management would rely on AWS Cost Explorer and usage reports to predict and control spending.

In terms of cost predictability, containers provide a consistent and controlled cost for steady workloads, while serverless functions are more cost-efficient and predictable for applications with variable usage. Serverless architecture supports a pay-for-execution model, which is ideal for projects that experience fluctuating demand.

When evaluating future expansion, the primary considerations include performance, cost, and architectural complexity. Serverless designs offer automatic scaling and low idle costs but may introduce cold-start latency and limited runtime customization. Containers offer more control and consistent performance but require active management of infrastructure and capacity.

Elasticity and pay-for-service are central to any cloud-based scaling plan. Elasticity ensures the system can grow or shrink dynamically with traffic patterns, maintaining efficiency. The pay-for-service model aligns operational costs directly with usage, promoting financial efficiency as the application evolves.

Conclusion

CS-470 has been one of the most valuable experiences in my development as a software engineer. The course reinforced the importance of designing applications for scalability, maintainability, and cost efficiency in the cloud. By mastering serverless deployment, containerization, and API integration, I now have a stronger foundation for building enterprise-grade applications that align with industry standards and best practices. The skills gained from this course will continue to guide my professional growth as I pursue a career in software and cloud development.