Marissa Lanza

December 16, 2025,

CS 470 Final Reflection

Presentation Video: https://youtu.be/9Wf4rZrFZvs

GitHub Profile: https://github.com/LiXiu37YaHua

GitHub Repository for module 8, pdf reflection: https://github.com/LiXiu37YaHua/CS470-final-reflection/tree/main

GitHub added collaborator link: https://github.com/LiXiu37YaHua/CS470-final-reflection/settings/access?guidance_task=

CS 470 Final Reflection

Throughout Project Two, I had the opportunity to bring together everything I learned in CS 470—from containerizing a full stack application to deploying a cloud-native, serverless architecture using AWS microservices. Creating the conference-style presentation required me to reflect deeply on not just how each part worked, but why certain design decisions mattered. This process helped me think more like a professional developer who must communicate solutions to both technical and nontechnical audiences.

Experiences and Strengths

One of the most valuable parts of this course was explaining how containerization and orchestration support cloud migration. Developing a deeper understanding of Docker, images, and Docker Compose showed me how developers support consistency across development, testing, and deployment environments. Containerization proven how real-world challenges—such as dependency management, scaling, and deployment reliability—are solved in professional cloud workflows.

Working with AWS serverless tools was another major learning experience. Breaking the full stack application into services such as Amazon S3, DynamoDB, AWS Lambda, and API Gateway required me to understand how these components interact within a distributed system. I learned how serverless architectures reduce operational overhead, drop the need for infrastructure management, and allow applications to scale automatically in response to demand. Explaining Lambda logic, API routing, and supporting scripts helped solidify how each layer contributes to an efficient cloud-native design.

Security was another major takeaway from this project. Configuring IAM roles and policies forced me to think critically about the principle of least privilege and the risks associated with misconfigured access. I learned how IAM policies directly control what Lambda functions, APIs, and S3 buckets are allowed to do, and why securing these interactions is essential in production cloud environments. This experience strengthened my understanding of cloud-based security and access control.

Creating the narrated presentation further strengthened my professional communication skills. Organizing complex technical concepts into a concise, structured explanation required me to justify design decisions, explain trade-offs, and clearly prove how the application functions as an

integrated serverless solution. This skill is critical for software engineers who must collaborate with stakeholders across technical backgrounds.

Overall, this course helped me grow in multiple areas, including cloud architecture, system design, security awareness, and professional communication. It also provided a strong portfolio artifact that I can reference in future job interviews when discussing cloud development, AWS experience, or application migration strategies. Through CS 470, I developed skills that prepare me for roles such as Junior Software Developer, Full Stack Developer, Cloud Application Developer, or Software Support Engineer.

Planning for Growth

Planning for future growth is a critical aspect of cloud-based application development, and CS 470 strengthened my ability to think strategically about scalability, reliability, and cost efficiency. By synthesizing what I learned about cloud services, microservices, containers, and serverless architectures, I can plan for long-term expansion in a structured and cost-conscious way.

To handle scale, I would rely on serverless services and micro service-based design to allow individual components of the application to scale independently. AWS Lambda automatically scales based on incoming requests, which removes the need for manual capacity planning. For error handling, I would implement centralized logging, monitoring, and alerting using cloud-native tools to quickly detect failures and isolate issues without changing the entire application.

Cost prediction plays a key role in planning for growth. Serverless architectures follow a pay-for-service model, meaning costs are based on actual usage rather than pre-allocated resources. This makes serverless solutions more cost predictable for applications with variable or unpredictable traffic. Containers, while offering greater control and portability, often incur fixed costs because resources are distributed even when idle. For growing or early-stage applications, serverless is more cost-efficient, while containers may be more proper for steady, high-throughput workloads.

There are several pros and cons that influence expansion decisions. Serverless architectures reduce operational overhead, scale automatically, and minimize infrastructure management, but they can introduce challenges such as cold-start latency and vendor lock-in. Containers provide more control and portability but require greater management effort and monitoring. Choosing between these approaches depends on traffic patterns, performance requirements, and long-term budget considerations.

Elasticity and pay-for-service pricing are key factors in future growth planning. Elasticity allows applications to dynamically adjust to demand, ensuring consistent performance during traffic spikes. Pay-for-service pricing ensures that costs align directly with usage, reducing financial risk while supporting sustainable growth. Together, these concepts enable informed decision-making and allow applications to scale efficiently without unnecessary overhead.

Conclusion

CS 470 provided hands-on experience with cloud-native development and helped bridge the gap between technical implementation and strategic system design. By developing, migrating, and presenting a full stack application in the cloud, I gained skills that directly support my professional goals. This course prepared me to design scalable, secure, and cost-effective cloud solutions and to clearly communicate the reasoning behind technical decisions in professional environments.

References (Optional)

Amazon Web Services. (2025). *AWS DynamoDB Console – Table permissions* [Computer screenshot].

Amazon Web Services. (2025). *AWS Identity and Access Management (IAM) policies and roles* [Computer screenshot].

LuvVoice. (n.d.). *Text-to-speech conversion tool* [Web tool].