Nathan Book

10/26/2025

CS 470 Final Reflection

https://youtu.be/-FJcskxM2cI

The modern world has transitioned into a state of everything being digital. Businesses now require websites and online ordering to keep up with today's supply and demand. As a business owner, a serverless or cloud-based website is more practical than buying your own hardware and hiring staff to develop a website and maintain it. Learning how to move a static website to a serverless environment is a skill that will serve me well in today's job market. I have learned how to containerize the front-end, back-end, and database of a static website. I have configured the Amazon AWS services provided for developing a cloud-based website including; DynamoDB, S3 Buckets, Lambda Functions, API Gateway, and IAM. My strengths as a software developer are diverse, I like to currently think of myself as a Jack of All Trades, master of none. This is due to the lack of experience I have in the field, but I believe with the right opportunity, I will be able to focus and gain more knowledge in a few specific arenas. I think that in an entry level job, I would be best at UI design, the coding is less confusing and does not have several layers to keep track of.

Microservices can improve management and scale by allowing independent scaling and deployment of services. To handle scale, microservices use tools like containers and automatically scales functions. Error handling involves building resilience into each service and using monitoring or debugging. With the pay for service model, you do not have to have dedicated hardware on site, you are only paying for what functions you execute. Microservices can scale individual services horizontally by adding more containers or instances as demand

increases. The require designing each service to be resilient, using techniques like circuit breakers, and implementing redundancy, load balancing, and failover mechanisms. The containers are more cost predictable when the usage is constant. With fluctuating workloads, containers can be less predictable because you pay for the provisioned resources even when they are not being fully utilized. Cloud providers provide the elasticity automatically, ensuring the functions have the resources to run during high-demand periods and can scale to zero when not in use.