Jonathan Wolanyk

August 20, 2023

CS-470 Final Reflection

Link to presentation: https://www.youtube.com/watch?v=iH7xGiErmWQ&ab_channel=JonW

Experiences and Strengths: Explain how this course will help you in reaching your professional goals.

Since the internet is a core component of business in the modern age, this course helped me reach my professional goal of providing website development support to local businesses by allowing me to develop a deeper understanding of full stack development in a serverless environment.

What skills have you learned, developed, or mastered in this course to help you become a more marketable candidate in your career field?

The primary skill I have learned in this course is how to use Docker Compose, DynamoDB, and the AWS platform to create and maintain a fully-functioning website.

Describe your strengths as a software developer.

My strength as a software developer is creating a great user experience. With most of my experience being in the video game development space, creating a superb user experience has always been the most important component that drives development for me.

Identify the types of roles you are prepared to assume in a new job.

If I were to pursue a new role, I would be searching for opportunities in full stack web development, video game development, or UI/UX support.

How would you handle scale and error handling?

To handle scaling, AWS Lambda is an incredible tool that supports any level of traffic along with a pay-for-use billing system. As a result, AWS Lambda should be used for the website I developed in this course since it is unlikely that traffic will be high enough to warrant EC2 support but having the capability to support high traffic times is essential for any website. Regarding error handling, it is important to have errors logged and tracked. Each function should throw an appropriate error with descriptive language of where the error occurred and how. Luckily, AWS has a well-designed platform for error handling with the implementation of its Step Functions.

How would you predict the cost?

AWS is transparent about its costs. For Lambda, the information can be found using this link: https://aws.amazon.com/lambda/pricing/

I would use the above link to get a rough estimate of how much my team would be billed if I were to keep this website running long-term, ensuring a daily log of website traffic. If this task proved too challenging, there are services on the AWS Marketplace platform to log traffic such as "Daily Foot Traffic" by PlaceIQ.

What is more cost predictable, containers or serverless?

From my experience, containers are more cost predictable since billing occurs due to how long the containers are running, not how much traffic (which is variable) is occurring/data is being sent.

Explain several pros and cons that would be deciding factors in plans for expansion.

While considering variables for expansion, one should consider the cost based on the service used in comparison to the nature of the website being run. For example, a static website does not need AWS Lambda to function and may be better-suited to be run by containers, while a marketplace website will have many functions that must be run in real-time with a variable amount of daily traffic, making AWS Lambda a great option. As a website scales on the AWS platform, switching services from Lambda to EC2 should be considered as well.

What roles do elasticity and pay-for-service play in decision making for planned future growth?

As described above, websites should be developed in a manner that balances cost with support. A website that has large fluctuations in traffic, such as a ticket sale website that has strong geographical ties, may be a great candidate for AWS Lambda to have costs lowered while maintaining support during peak hours. Large-scale, multi-national websites, on the other hand, would be better candidates for support measured in large blocks, such as AWS EC2 since the traffic is relatively the same at all times and the contract with Amazon would ultimately be less-expensive. Smaller websites that are may not always be running, such as a static webpage, would be great candidates to be run by containers.