Brad Peterson

CS 470

CS 470 Final Reflection

October 26, 2025

https://youtu.be/NPKrrAmsjZQ

http://bradpetersonproject1.s3-website-us-east-1.amazonaws.com

- **Experiences and Strengths:**

This course forced me to jump into the monster that is AWS services. It's something that I think every web developer on earth is familiar with because it's become so ubiquitous in the field. I feel like I can at least wrap my head around a conversation about running a project on a cloud-based environment now. I feel very good about my ability to understand what is meant by "serverless" architecture as well. I'm also aware that there is a whole lot that is left with the AWS environment that is often what folks make entire careers out of doing.

At this point, I can see why a DevOps role is appealing to developers. While it's less actual programming, it is a lot of making the pieces fit together and that has been pretty satisfying. I also feel like the skills gained in this course go beyond just coding to being able to have meaningful conversations about how to run and drive a business forward as a software engineer. I absolutely need more hands-on experience before I feel I would have the knowledge base to be the one driving those, but it is something that I hope to achieve.

- **Planning for Growth:**

A developer's ability to utilize microservices makes them exponentially more impactful it creating a well developed project. It can often feel difficult to get up to speed with these sort of tools but I think it's well worth it because you get a polished result that does a lot of hard work for you, especially in maintaining the demands of your project. Furthermore, serverless architecture enables developers to launch these projects without intense capital to be able to test markets and adjust as they need to. It really just seems to take a lot of difficulty in managing your application and lets you focus that effort on making a good product instead.

Some of the ideas we didn't approach in this course like error handling and actual costs are interesting. I think for error handling, you could lean on containerization to shift that container when it encounters an error to logging it and spin up another one beside it to keep handling demand while that one logs it's error and closes. As far as costs, I know we generally have said pay as you go is much cheaper, which I believe, but what are the pricing tiers offered? Do you pay less per usage if you are a large customer of AWS or is it fixed for everyone. I think to project costs you would need to know these tiers for cost modeling based on how many users you have, most products don't scale linearly so knowing how much you're going to pay in the future is hard to project. Most scarily, would AWS give preferential server usage to larger customers, potentially squeezing smaller products out of server space in the event we really did hit cloud capacity?