Makayla Anderson-Tucker

28 April 2024

CS 470 Final Reflection

Link to presentation video: https://youtu.be/FIL4KPN4CD4?feature=shared

Experiences and Strengths

Before this course, I wanted to work in software development. Now, I know more specifically I'd like to work as full stack software developer working on applications for the web. I knew a little bit about cloud technologies, but this course helped me get comfortable with using serverless solutions to containerize applications and use them to migrate any app to different platforms. If I'm looking to build a MVP, I learned that the AWS ecosystem provides great options for quickly getting it running and hosted on the cloud. They make it easy to create and use REST APIs as well as manage the authorization and authentication.

My strengths as a developer lie in knowing how to host apps locally versus on the cloud and what that means in terms of development. Once the user's needs are determined, it's a strength to be able to analyze and decide the best possible tools to achieve those goals.

Planning for Growth

Elasticity means that system management is minimal for the developer while running instances of the app across a variety of servers easily, quickly to scale up or down based on the current the business needs. One of the ways to make sure an app is elastic is by using serverless solutions. Serverless has a multitude of benefits and uses when developing web applications. Taking a serverless approach would significantly decrease overhead costs of running and maintaining servers on multiple sites. Containers are cost predictable as many tools to containerize an application are already free and open source. However, if you did need to use it on an enterprise level, the cost is typically fixed with containers. Serverless is not the same, though there are some similarities. Some sites allow free hosting on their servers, but only if the traffic and usage falls within certain metrics.

One of the popular payment models in the serverless industry is pay-for-service, which means you only pay for what you use. While it can be cost efficient if you know about how many servers will be used and requests will be made, that’s not always the case. Sometimes small websites blow up overnight and the developers wake up to find out there are astronomical fees for using the extra servers and resources used to accommodate the users. It may be good to think about whether or not you need to limit the number of users that can access your site at one time. Not all cloud services offer the same management features, so I’d definitely be sure to pick one that manages most of the scaling, error handling, and security for me.