Brittney Miller 08/19/2024

SNHU

CS 470 Final Reflection

Presentation Link: https://youtu.be/jJWHUfq2L-0

Throughout this term, I have not only learned, developed, and mastered numerous skills, but also understood their practical application in the field. For instance, I have learned how to create containers, navigate lambda functions, and connect them with a serverless database. These skills, along with the ability to assign and create user roles for a site, will not only help me succeed but also demonstrate my understanding of building websites from front to back, and considering customer needs and security.

My strengths as a developer are putting myself in my customers' shoes and making sure I can imagine what they want or what is required in what I am designing. I also don't give up when I run into a solution; I use my resources to figure out the answers. This is something you have to do as a developer since you will constantly run into issues or vulnerabilities that need more attention and don't go as smoothly as you hope.

This course has equipped me for a variety of roles in the software development field. I am now prepared to take on the responsibilities of a full-stack developer, a project team lead overseeing a development team, and a contributor to applications' testing and security aspects. These roles can be integrated to create a comprehensive application, or they can function independently with effective communication during the planning phase.

When you are thinking about how to handle scale and error issues, you want to start by configuring AWS auto-scaling, which can help fix problems on its own. This process is done through CloudWatch, which makes it easier to apply. When you auto-scale a project, you allocate more resources based on traffic, adjusting the read capacity for your database. You can start handling errors by using AWS functions that help automate the process of checking for issues and handling them.

When it comes to the cost of an application and its build process, there are a lot of factors to consider, like what database you want to use. There are pay-as-you-go services where you can choose and pick so you are not wasting money on unused resources, or there are all-in-one resources that you pay for everything regardless of whether it is used or not. AWS has a cost explorer that can help you plan out your budget when considering what you want in your project. This enables you to mitigate what is needed, what is an extra expense, what is pleasing to pay for, and what you don't need or could do without to help save cost.

Predicting cost when it comes to containers or serverless databases can be viewed as such; serverless offers cost savings by using the pay-as-you-go method, which means paying for resources you use. That can make it very unpredictable as you don't know what you want or need until you are in the planning and building process. On the other hand, containers are more predictable because the rate of management is pretty consistent.

If you are making a pros and cons list when considering plans for expansions, you have to think about it in steps. The first thing is how much interest is in the thought of a web application. The next thought would be thinking about the cost associated with the web application. Think about the choice to migrate over to a serverless approach, as this is a significant upfront cost but can result in savings during the maintenance period of the application

lifecycle. You'd then have to think about the level of security you want for your application, as that can also affect the budget.

Elasticity is prepared to handle any anticipated future growth, as resources can increase or be adjusted based on demand. Pay-as-you-go, a model used by AWS, can also adjust to scale from where it was created at the beginning of the build process. Working in this fashion encourages companies to test their capacity regularly when growth is rapid or exceeds quotas while still retaining stability.