Zachary Meisner

12/18/22

CS 470 Final Reflection

https://github.com/Zach-GH/cs470FullStack2/tree/main/Project2

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

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

I valued the time that I spent on my projects in this class because it helped me understand at a much lower level how it is that my Macbook works and configures with what is supposed to be a Windows only configuration. I am mainly referring to the application that we initially started working on that had to do with hosting web servers and integrating the different changes of an API structure to the application in real time. This initially was an incredibly difficult challenge due to the type of computer I had, but after enough digging into the actual issue, I was able to use my terminal settings to revert to bash which made massive differences in the way that I downloaded deprecated versions of software that we had to use for class that otherwise were not supposed to be able to configure properly.

Apart from this, I also valued the hands-on experience that I got using Amazon, as it gave me a much better understanding of what AWS solutions was offering, and the difference of ease for someone like me that may have hardware issues, in addition to the continuous maintenance, updating, and security of my network and server. I really enjoyed the experience I got with the architecture of the project itself, and the multitudes of different ways each component interacted with one another, and how the totality of the solution is delivered to the customer.

- Describe your strengths as a software developer.

I think that my strengths as a software developer surmise mostly with understanding issues and the applications of problems when it pertains to code. Understanding and segmenting problems to give myself a full view on the architecture of the project, and how that may change the overall interaction between each node, really benefitted me here. This is mainly due to the application of my actual computer when pertaining to these components, understanding what dependency needed to be where, and why it is that I could or could not execute the commands properly to assure that I was using the correct configurations. I do not give up, and I do my best to not stop at anything to assure that I find a solution to a problem, and it is quite apparent here in this instance.

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

There is a plethora of different roles that I am prepared to assume in a new job. I already have and am working my dream job as an Embedded Software Engineer. What I can say about the job I currently have has to do with that I am a working professional, and not only am I expected to know just about everything, but I am expected to be able to learn anything, and overcome everything, as I am the line between good and bad, and at the end of the day, I am the one that makes pertinent decisions and am the sole reason whether something ends up working or not. Obviously, there is much more to this, as in a team setting, there are plenty of other developers and engineers that can be able to do the same that I do. We all offer our own values and experience that we bring to the table, and I think this is the beauty of engineering, as there are always opportunities for everyone to step up and be a team player so we can all play into one another's strengths and weaknesses.

Planning for Growth: Synthesize the knowledge you have gathered about cloud services.

- Identify various ways that microservices or serverless may be used to produce efficiencies of management and scale in your web application in the future. Consider the following:
    - How would you handle scale and error handling?

Handling scale and error handling seems to be best done with AWS. I say this because the solution that is offered of an amorphous state of scaling really seems to be the optimal way to handle the issue. Giving an allowance to the actual needed amount of attention each application and server needs based off demand is the most cost effective, and logical way to handle the ongoing problem, which in the long run reduces cost, and keeps server health high. Error handling on the other hand needs to be done on a case-by-case basis unless it can be automated. If we can take an error and have some type of pipeline or AI check for the possible reason that an error might happen, it could be easy to allow this to potentially be handled by a web service such as AWS. In the case that this AI/pipeline could not figure out the solution, it is pertinent to have knowledgeable and educated engineers on call to fix these problems to assure that customers are able to have the best experience possible, so we can not only keep everybody safe, but keep our software and hardware as healthy as can be straying away from any potential possibility of misconfiguration.

- How would you predict the cost?

Predicting cost is not something that is easy. Again, I do like the idea of being charged only for what you use, but if you were to sit down with a customer, unless they had an expected rate of use, how on earth are you able to give a finite number? This is a problem only potentially with smaller companies, as they may not always be able to deal out the amount of money an uptick in visitations may bring. On the other hand, a big company would have no problem dealing with this cost, so I think that a baseline needs to be set in certain instances to assure that companies can get a hard number instead of a potentially volatile one. I believe that the adaptability that AWS offers is great for both little companies and bigger companies, as deals are always able to be made, and there are an enumerable number of options available to customize it towards the individual.

- What is more cost predictable, containers or serverless?

Containers are always cheaper due to the predictability of use. When using serverless, you are giving control to the vendor which makes it overall more expensive. It is hard to really pinpoint the predictability here because if your containers are not managed properly, then you may be paying for resources that you are not using. I think that this can be a tossup, and particularly matters dependent on the type of customers that needs the solution for their business.

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

Some pros and cons that could be deciding factors in plans for expansion have to do with the architecture of the project that you are looking to maintain, and the use-case of your application and what customers really need, and use, on a day-to-day basis. This matters when it comes to picking your solution because you can really get an identity of your customer, who they are, and what the best possible solution for them would be to give the best experience that they are able to have. In addition to this, we want to think about safety, security, and the times in which our customers are most likely to access your website, to assure a streamlined constant of cost. There is plenty more to take into consideration, but I think that these lie amongst some of the more important factors of picking a proper solution.

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

I think that elasticity and pay-as-you-go services play a massive part when it comes to decision making that is planned for certain growth. When hosting applications, it is incredibly hard to predict what your customers may or may not do to a certain degree. What I am referring to here is anomalies outside of the habitual patterns' customers may become bound to. For example, if your company blows up on Tiktok overnight, you may see a massive uptick in visitations that you otherwise may have not been expecting. This is where elasticity would come in handle as you would be always ready for this as the anomalies of patterns are really where this type of solution handling would be helpful. On the other hand, let's say that from this uptick in attention, you gain a massive amount of new customers, meaning that you must change the type of service you want and are paying for to handle your day-to-day traffic to assure that you are meeting cost and providing the best type of service. This type of changeability when it comes to your solution and business deals is perfect because you can assure that you have engineers handling problems for your business in the best way without having to make it too challenging to assure the closest type of fiscal growth and use.