Name: Tim Gallus

Date: 8/14/2023

Assignment name: CS 470 Final Reflection

YouTube presentation link: https://www.youtube.com/watch?v=IWy9CBEnHes

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 had prior experience starting up instances of pre-created containers, but this course gave me the skills and experience to set up containers as well as knowledge of how to use docker-compose to start multiple containers at the same time that could interact with one another. Along with this I gained knowledge of various paths for migrating from a local environment to cloud and experience with doing so.

  I also had some prior experience with AWS, thanks to prior work experience with automation using their device farm and an AWS course that was offered through SNHU, however CS 470 gave me additional experience with how roles and policies work and a strong understanding of how Lambda functions work, how DynamoDB functions and how it compares to MongoDB (and how both of those compare to relational databases). Additionally, I learned and gained experience with using these features in unison in order to set up a full stack application using API calls and allowing Lambda functions to interact with the DynamoDB tables that I had set up.

- Describe your strengths as a software developer.

  I have always been quick to learn new things and have an inquisitive mind that delights in new challenges. In my prior employment this has often led to my being chosen for exploratory new tasks. I have a multi-disciplinary education and bring that perspective with me. While I love learning and hope to continue my education throughout my life, I also have over 15+ years of experience working in software development. My many years working in QA, both as a tester and QA engineer, give me a strong appreciation and good practices for testing and recognizing problems. Meanwhile, the many departments and titles that I've helped to produce on various platforms including both desktop and mobile have given me a very wide range of experience, as has coding in many languages and witnessing the changes that software development has undergone over the years.

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

  While I'm open to just about any role where I continue to code and develop software, I am hoping to return to a software engineering role similar to the one I left to continue my education. I'd like to broaden my horizons and experience what the software industry looks like outside of games, as much as I have enjoyed my time working in games. I'm very much enjoy returning to a role in developing automation similar to my prior position, but the last few courses on full stack development have definitely enticed me towards looking at the role of a full stack engineer. Back-End Engineering would also be of interest as I've had some experience working on automation framework in a similar capacity but would enjoy more of that.

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?

      While error handling logic could be added to Lambda functions, if AWS is the cloud service provider in use, it provides access to AWS Step Functions, which includes error handling support. Conditional logic can be set up with Step Functions based on the error type that occurs.

- How would you predict the cost?

  I cannot stress enough how important data driven development can be when possible, especially when it comes to cost prediction. Both when utilizing cloud based containers or serverless for an application…or even when developing locally…if data is available to analyze from prior applications of a similar nature in order to determine user behavior it can be key to the successful launch of a project and its ongoing estimates of appropriate resourcing to provide the best and most cost effective user experience. In the case of local development this can allow you to purchase the hardware necessary to have a successful launch. In the case of cloud based computing, this can help estimate what your cost needs will be to initially support your product by knowing how many users will be using your application and at what times of the day or week. With containers this will let you know how many containers you will need to have spun up during what time frames in order to support load, and when they can be spun back down to save on costs, thus allowing a good cost estimate. Further data analysis can allow you to predict what user behavior will look like when using the application, which will allow you to estimate how often what lambda functions will be called from which a price estimate can be determined. With local development, it is very difficult to scale after initial hardware purchase and launch however with cloud based development, additional data about user behavior can continue to be gathered and analyzed allowing us to change both our resource management on the cloud to better match a user behavior model that should become ever more accurate and to change our pricing estimates accordingly. Depending on the fluidity of the application in question a good data analytics team should be able to come to predict user behavior on a given application well enough to very accurately predict that behavior and manipulate it in predictable fashions through changes in the application. I have been lucky enough to work closely with a team that was doing just this in the past.

- What is more cost predictable, containers or serverless?

While serverless is oftentimes more cost effective, containers are more cost predictable. With containers you pay a set fee while containers are in use and only while in use with control over how many containers are launched, when, and for what duration. Given that you are also aware of the base cost, this makes their total cost highly predictable, though an argument can also be made that the developer must also be able to predict the amount of resources that will be needed at a given time to know how many containers they will need to be running for their applications to benefit more fully from the scalability of a cloud based environment. With serverless, each call to a lambda function incurs a cost. While this is often more cost effective, in order to be cost predictable it is now on the developer to predict how users will interact with their application in order to determine how often calls will be made to the various lambda functions in use. Data driven assessment of user behavior in order to predict how often various functions will be utilized is more in the realm of a good data analytics team, which may benefit from a relational database more than using DynamoDB or MongoDB.

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

Assuming we are looking at the pros and cons of using serverless cloud based architecture for development with an eye towards eventual expansion versus local, server based architecture, a number of advantages exist and a few negatives:

With cloud based serverless architecture you generally pay for what is in use, even with providers where one must reserve space it is much easier to purchase additional space compared with purchasing additional hardware or conversely to divest one's self of those additional costs when extra resources are no longer needed. With cloud based it is even possible to react to hourly changes to load needs. Also serverless architecture there are no maintenance needs for the environment. This is especially of interest when planning for expansion as there is no need for additional hardware set up. The ability to access your cloud based architecture from virtually everywhere is generally a pro verses local development, however I have seen times when wide scale internet outages or more local denial of service attacks have prevented access to the internet at large, while the network inside the office remained stable and allowed work to continue. This is rare however and with more and more out of office work occurring, cloud based architecture seems like a good fit. Security can be both a pro and a con with cloud based computing. Many cloud based providers, such as AWS offer robust security tools and have a good reputation but it does mean trusting your data to another company.

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

  Pay for service means you are only paying for the services you need while they are in use. Elasticity represents the ability to increase or decrease resources as demand requires. Most cloud based computing models, especially AWS, can provide a lot of elasticity compared to older local development models that can only expand resources by purchasing and setting up additional hardware and lack a good way to recoup those costs when they are no longer necessary. The risk of expanding and failing to purchase an appropriate amount of hardware to support the necessary resource is extremely low with cloud based development…if you need more resources its easy to spin up additional resources and only pay for what your using. If you have more resources in use than needed, you can stop some (or no one will be actively using them) and stop paying for the unnecessary resources. This goes beyond a benefit for planned future growth and can literally be used to adjust resources, and thus costs, to meet active demand at different hours during the day. Many of my past projects have had peak hours where they had more active users and these became predictable over time with appropriate data analyzation. In the case of some of older, fully local development projects I’ve worked on, resources needed to be purchased to meet the highest active demand during a given day (and the highest active demand during the lifespan of the project…typically around launch). Cloud based computing elasticity allows us to run the resources necessary to support high demand hours and then cycle down some resources during lower demand hours, thus saving costs. It also lets us meet high demand periods in a project’s lifespan without purchasing additional hardware that is unnecessary and an un-recoupable cost during lower demand periods in a project’s lifespan.