CS-470 Full Stack Development II – 2024 C-4

CS-470 Final Reflection

Gregory Isajewicz

August 23, 2024

https://youtu.be/l6-zEz7jRpo

Experiences and Strengths

Throughout this course I have worked with Amazon Web Services and Docker to migrate a full stack application to a cloud-based architecture using AWS, including setting up infrastructure, deploying applications, and managing cloud resources. I have had the opportunity to work hands-on with AWS services including S3, Lambda, DynamoDB, API Gateway, IAM, and CloudWatch, and gained basic proficiency in cloud-based development. I have also developed my skill in DevOps and CI/CD practices, as well as improved and refined my knowledge and understanding of security best practices. I believe these to be a sought-after skillset in the tech industry that will make me a more marketable candidate as I pursue a career in software engineering.

My strengths as a software developer include versatility, technical proficiency, and attention to detail. The skills I have improved upon in this course enable me to work across the full stack, from the front-end to the back-end, and now in cloud environments, demonstrating my versatility as a developer. I have also refined technical skills in multiple programming languages throughout this course, including – but not limited to - C++, JSON, TypeScript, and more. With these strengths and learned skills, I believe I am now prepared to assume a role as an entry level full stack developer, DevOps engineer, Cloud engineer, or software engineer.

Planning for Growth:

Microservices and Serverless:

Cloud services provide a flexible and scalable environment for deploying applications, offering infrastructure, platform, and software as a service. In the context of full stack development, cloud platforms like AWS enable developers to deploy, manage, and scale applications efficiently. These services also offer tools for monitoring, security, and automation, which are essential for maintaining robust and resilient applications. The use of cloud services simplifies the management of resources, allowing developers to focus on building features rather than worrying about underlying infrastructure management.

Error Handling and Scaling:

Both microservices and serverless architectures inherently support horizontal scaling, which means that additional instances of services or functions, as well as cloud resources can be

spun up to handle increased demand as needed. For microservices, each service can be scaled independently based on its specific demand. For serverless, the platform automatically handles scaling, allowing you to respond to spikes in traffic without manual intervention. This provides an overall benefit to the application owner, as their infrastructure upkeep requirements are minimized.

In a microservices architecture, error handling can be managed through well-defined communication protocols. In a serverless environment, error handling is often integrated into the platform, with support for retries, dead-letter queues, and monitoring tools that alert you to issues in real-time. AWS features many security services that can be integrated to assist with error handling beyond the incorporation of such best practices in the code of the application itself.

Predicting Cost:

Predicting the cost of upkeep of a cloud-based application is made simpler and, in some ways, more trivial compared to traditional architecture. The owner of the application does not necessarily need to figure out ahead of time the exact amount of storage or resources needed or invest in building those resources. Instead, the cost scales with the usage. This means that the user will pay for what they use and does not need to be concerned with added costs of purchasing or installing hardware in order to scale up and application. Similarly, if the use is scaled down, the user is not left with extra resources invested in that are not serving a purpose.

Containers vs. Serverless:

Containers tend to be more predictable in terms of resource allocation, as you typically pay for a set amount of computational power (CPU and memory) regardless of what is actually utilized. This can be advantageous for applications with consistent, predictable workloads, but may lead to underutilization and higher costs if resources are not fully used.

Serverless is generally more cost-predictable for applications with variable or sporadic workloads, as you only pay for what you use. However, if your application has high and consistent usage, the costs of serverless can accumulate and potentially exceed those of containerized solutions, which offer a fixed cost for reserved resources.

Pros/Cons, Elasticity and Pay-for-Service:

There are several pros and cons to consider when planning for expansion. One of the pros is scalability and elasticity. Cloud services allow you to easily scale resources up or down based on demand, making it easier to handle growth. Elasticity reduces the risk of both over-provisioning and under-provisioning – both of which can cause problems in business scaling and growth. When considering expansion, elasticity ensures that your application can grow organically without requiring a complete overhaul of – or addition to the infrastructure. This scalability ensures that your application can accommodate an increased number of users and increased traffic without significant change to the infrastructure. On the con side, once you build your application around a specific cloud provider’s services and structures, it can be challenging and costly to migrate to another provider. This sort of vendor lock-in can limit flexibility should you decide down the road to go a different direction.

Another benefit is the access given by cloud services to a wide range of advanced tools and technologies such as AI/ML, big data analytics, and IoT services, enabling rapid innovation and development of new features and services. This could be a significant pro for a small company that may not otherwise have the financial backing or infrastructure to access such technologies readily. However, access to these innovative technologies can have a drawback as well. Technology is a fast-paced industry, and keeping up with the latest-and-greatest in the field may require more intensive continued learning and study of new skills.

Another pro of cloud services is that they often operate on a pay-for-service model, where you only pay for the resources that are used. This can be very cost-effective, especially for small businesses or startups that may not need to, or be able to, commit to large upfront investments. This model plays a key role in decision making for planned future growth, as it works alongside elasticity to allow you to scale up infrastructure incrementally, in line with actual demand, providing a clear understanding of expenses and making it easier to budget for growth.