In this course I have learned a lot regarding cloud base services, my understanding and knowledge of databases and AWS specifically have helped me learn that this is a flied I sincerely enjoy. I think my understanding of MongoDB, DynamoDB and using docker have really come to light. I feel confident that I can work these platforms and move a local database to a cloud server with minor issues and learning curves. I think determination is one of the most needed things when it comes to software. From my understanding, you cannot be someone who gives up. Through my journey I have been ruthless at being the best and having the best grade. I can’t say that this makes me the most efficient or talented person in development but what this does me for me and a future employer is , you are going to get someone who is not only going to get a project done, but done right and 100%.

Using this course, I would love to get into databasing or UI. There is a difference in many ways but with Database it is fun to navigate and see everything work and function accordingly. With you UI you get to see your creation really come alive and I find that to be fun but also extremely challenging. microservices and serverless architectures can greatly improve the efficiency of managing and scaling web applications. Through decomposing the application into smaller, independently deployable services, microservices allow for more granular control over scaling. Error handling is streamlined as failures in one service do not necessarily impact others, and services can be individually monitored and debugged. Predicting costs in a serverless architecture is facilitated by the pay-as-you-go model, where expenses are directly tied to actual usage, making costs more transparent and scalable. In contrast, containers, while offering more predictable performance, may incur more fixed costs due to continuous resource allocation regardless of usage. Overall, serverless models often provide more cost predictability and efficiency for applications with variable or unpredictable loads, whereas containers might be more cost-effective for stable, consistently high-demand applications.

For microservices there is scalability, fault isolation and technology diversity as great pros but their cons would be complexity, deployment overhead and data management. For serverless the pros would be cost effectiveness, automatic scaling, and simplified management but that comes at the cons of cold start latency, vendor lock-in and resource limitations Overall serverless overhead provides cost effectiveness and simplified scaling with some introduced latency issues and vendor dependencies. Microservices offer the greater control and flexibility but with more complexity and management overhead. Deciding between the two comes down to what the specific of the architecture is. You have to include cost, performance, growth, and development success.

<https://youtu.be/hXTwvrFUk2I>