Derek Paterson

December 16, 2022

Final Reflection

Full Stack Migration Presentation: https://youtu.be/I7D6vLng9YE

Experience and Strengths

I think the main skills I've learned in this course and the previous course are how to work with angular, as well as building cloud skills in general and AWS specifically. Additionally I really enjoyed learning about and using Docker, containerization makes everything so much easier, I feel this is the way forward.

I think my biggest strength as a software developer is my analytical nature, finding bugs and thoroughly testing things is something that comes naturally to me, and thinking outside of the box to find edge cases is part of that.

I am not particularly looking for a software development role or a full stack role for my career, I have elected to focus on information security, but the skills from this course are very applicable, especially the cloud integration process and setting up IAM roles and policies in AWS.

Planning for Growth

One of the key ways that microservices and serverless computing can be used to produces efficiences of management and scale are the fact that all of this is monitored and interrelated. You can set up some scaling to happen automatically but you can also really drill into dynamic demand monitoring and use that to tweak the efficiency of your app. I think scale would be handled well in that manner. Error handling is a whole separate beast, but a lot of that logging and reporting is automated, so maybe building some functions to assist with automating the collation of those reports would be a way to go about this, because clearly as the scale increases so does the amount of individual errors and eventually it isn't cost effective to have humans view reports.

I think cost prediction really depends on what sort of application is being developed, for example if you set up automatic scaling with demand and you have some sort of useful personal project done there and it goes viral, you could quickly be overwhelmed with cost. On the other hand, a product owner would be responsible to try to gauge the amount of monthly users for any web application they're responsible for, and in corporate use cases you could maybe expect relatively static demand based on project scale.

I think container cost is definitely more predictable of the two, you don't have the same potential for runaway automated scaling.

When expanding it really depends on the economic sector you're in, and for use cases that have a static load it might make more sense to keep a lot of functions in house on local servers, and then have a hybrid cloud system for things that could need to be scaled up or are difficult to deal with in a local context. So pros for scaling through cloud services, aside from the ability to scale very quickly, are really just reliability since the cloud data centers are all over. The cons are definitely cost, while you can get some discounts for volume, scaling over cloud can quickly become expensive. This is why I mentioned a hybrid-cloud solution, some things can be easily handled in house and they should absolutely be if it makes economic sense or security sense.

Elasticity and pay-for-service are some of the key features in cloud services, and most really good use cases for cloud services are applications that have inconsistent use across time of day. If your application has a massive demand swing daily, weekly, or monthly, it may make way more sense to scale up via cloud when the demand peaks, rather than paying for maximum capacity at all times. This elastic ability to scale up and down is also intrinsically related to paying as you go, so you're only going to pay for the computing resources you actually use. Again this really all depends on the use case, no solution is going to be an exact fit right out of the box, so deliberation and analysis is important.