Michael Tekin

6/26/22

CS 470 Final Reflection

Final presentation - https://www.youtube.com/watch?v=FeCEV7mKy5M

- **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've learned how to containerize a full stack application, migrate a full stack application to a serverless system using AWS microservices, create policies for managing security for a serverless application, deploy a static website using S3, create an API using API Gateway, and create tables in DynamoDB

    - Describe your strengths as a software developer.

My strengths are in quickly getting up to speed on using new technologies

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

I'm prepared to assume a role working with APIs and full stack applications including in cloud based development environments.

- **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?
        - How would you predict the cost?
        - What is more cost predictable, containers or serverless?

Scale could be handled by using serverless solutions that scale automatically (up to predefined constraints). I would use the pay-for-use pricing model that AWS offers for its serverless services together with expected usage to estimate costs. Containers are more cost predictable in the sense that you know how much you are going to be charged when you are charged for the amount of resources that you request. As opposed to serverless where you're less likely to pay for unused resources but you won't know exactly how much you'll be paying until after the resources are used.

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

Containers pros: greater portability, built in version control, greater ability to fine tune systems

Container cons: highly likely that there will be waste due to unused resources from giving yourself a buffer and over-provisioning rather than risk under-provisioning, requires more time and effort to provision and manage servers

Serverless pros: less money spent on idle resources, automatic scaling and server management, less time and resources that need to be devoted to managing those tasks

Serverless cons: prone to provider lock-in, more limited in how something can be designed

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

Considering elasticity and pay-for-service need to be considered when planning how to allocate resources and what resources are available.