Deonne Ludwig
06/25/2024
CS 470 Final Reflection
https://youtu.be/_BgxtQn9ppk

Experiences and Strengths:
Explain how this course will help you in reaching your professional goals.
Completing a Full Stack II course is a significant achievement that enhances your skills and marketability in the software development field.

What skills have you learned, developed, or mastered in this course to help you become a more marketable candidate in your career field?
I have developed a wide array of skills that are crucial for becoming a proficient and marketable software developer like advanced front-end development (Angular), back-end development (Node.js, Express, RESTful APIs and MongoDB), full-stack integration, DevOps and deployment (GitHub Actions, Docker and AWS).

Describe your strengths as a software developer.
Problem-solving skills, adaptability, attention to detail, collaboration and communication, continuous learning.

Identify the types of roles you are prepared to assume in a new job.
Full Stack Developer, Front-End Developer, Back-End Developer, DevOps Engineer, Technical Lead, Software Engineer

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:
Cloud services provide a range of functionalities that enable scalable, reliable, and cost-efficient web application development and deployment like Functions as a Service (FaaS)/Serverless that allows developers to execute code in response to events without provisioning or managing servers and Microservices Architecture that breaks down applications into small, independent services that can be developed, deployed, and scaled independently.

How would you handle scale and error handling?
Scale: auto-scaling, load balancing and error handling: retries and circuit breakers, monitoring and logging, graceful degradation

How would you predict the cost?
Cost estimation tools and monitoring usage.

What is more cost predictable, containers or serverless?

Containers are more predictable because you typically allocate a fixed amount of resources (CPU, memory) and are billed for their provisioned capacity. Serverless can be less predictable but potentially lower during low usage periods.

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

Containers:

Pros:

Greater control over the environment and dependencies.

Easier to lift and shift existing applications.

Consistent performance due to dedicated resources.

Cons:

Requires managing and maintaining the infrastructure.

Potentially higher costs during low-usage periods due to allocated but unused resources.

Serverless:

Pros:

No server management required.

Automatic scaling with demand.

Pay-per-use model can lead to cost savings.

Cons:

Limited execution time and memory constraints.

Cold start latency can affect performance.

Potentially complex integration with existing systems.

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

In decision-making for planned future growth, these factors help ensure that the infrastructure can support scaling needs while maintaining cost-effectiveness and operational efficiency. Balancing the benefits and drawbacks of different cloud service models (containers vs. serverless) will depend on the specific requirements and growth patterns of the application.