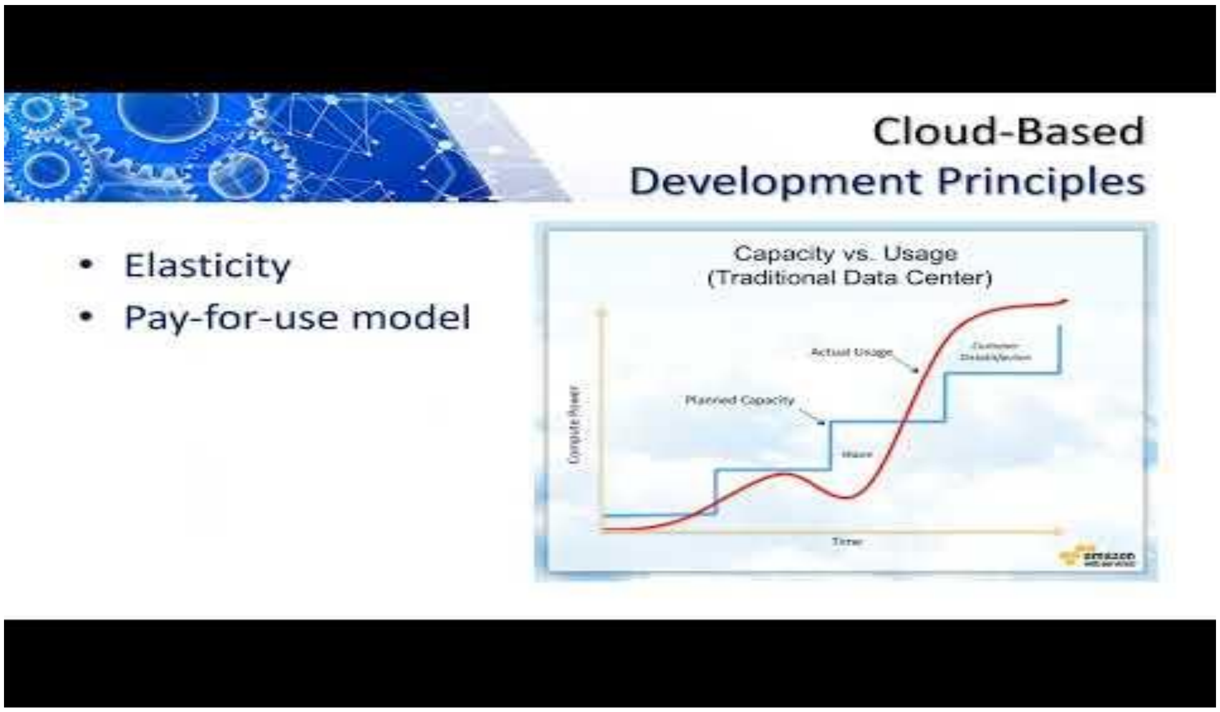
CS 470 Final Reflection

Valeriya Barker

12/22/2024

Assignment Name: CS 470 Final Reflection

YouTube Presentation Link: https://youtu.be/1g2LIsfsDl4



Cloud-Based Development Principles

- Elasticity
- Pay-for-use model

Experiences and Strengths

In CS 470, I learned many valuable skills that will help me in my career as a software developer. This course gave me hands-on experience in building, testing, and managing a full-stack web application in the cloud. I became more confident in my ability to create secure and reliable APIs, work with cloud services, and use tools to deploy applications efficiently. One of the biggest skills I gained is API development and integration. I learned how to design APIs that are scalable and secure, which allows applications to communicate effectively. I also improved my understanding of cloud architecture, which helps ensure that apps run smoothly in the cloud. Another key skill I developed is containerization using Docker and the use of microservices architecture. These tools help break large applications into smaller, easier-to-manage pieces. Additionally, I became familiar with DevOps practices, like setting up CI/CD pipelines, which make it easier to test and deploy updates quickly. These skills prepare me for roles such as Full-

Stack Developer, Cloud Engineer, or DevOps Specialist. I am confident in my ability to work on cloud-based projects and collaborate with different teams to deliver reliable software.
Some of my strengths as a developer include problem-solving skills, attention to detail, and the ability to adapt to new technologies quickly. I'm also a good communicator, which helps me explain technical ideas clearly to others, even if they aren't technical experts. Overall, this course has helped me grow as a developer, and I feel ready to take on professional challenges in cloud-based software development.

Planning for Growth

Planning for a web application's growth in the cloud requires careful thought about scalability, cost management, and system reliability. Thanks to what I learned in CS 470, I now have a clear understanding of how to design an application that can grow smoothly while staying cost-effective and efficient.

Using Microservices and Serverless Computing

Both microservices and serverless technologies are powerful tools for managing growth.
Microservices: These break an application into smaller services that can be scaled independently. This means that if one part of the app gets more traffic, only that part needs extra resources.

Serverless Functions: These allow you to run code without worrying about managing servers. You only pay for the time the function is actively running, making it a cost-efficient option for unpredictable workloads.

Managing Scale and Errors

To prepare an application for high traffic and unexpected errors, I would use tools like:
Auto-Scaling: Platforms like AWS Auto Scaling allow cloud services to add or remove resources automatically based on traffic.
Error Monitoring: Tools like AWS CloudWatch can detect issues in real-time and alert developers, helping fix problems quickly.

Predicting Costs

Accurately predicting cloud costs is important to avoid overspending. Tools like AWS Cost Explorer and Azure Pricing Calculator help estimate costs based on expected usage.
Serverless vs. Containers: Serverless is often cheaper for unpredictable workloads since you only pay when functions run. Containers offer more control and can be better for steady workloads.

Pros and Cons for Future Growth

Microservices Pros: Better scalability, easier debugging, and independent updates.
Microservices Cons: More complex to manage, higher operational overhead.
Serverless Pros: No need to manage servers, automatic scaling, cost-efficient for sporadic traffic.
Serverless Cons: Limited execution time, cold start delays, less control over infrastructure.

Elasticity and Pay-for-Service Models

Elasticity ensures that cloud resources adjust automatically based on demand. During high-traffic periods, more resources can be added, and during low-traffic times, resources are reduced to save costs. Pay-for-service pricing models ensure you only pay for what you use, making financial planning easier.

Conclusion

CS 470 has taught me how to build scalable, reliable, and efficient cloud-based web applications. I’ve learned to design APIs, implement microservices, and use cloud tools for better scalability and cost control. These skills are essential for planning future application growth and addressing challenges like cost prediction, error management, and resource allocation.

I am excited to apply what I’ve learned in real-world projects and continue improving my cloud development skills. This course has prepared me to think ahead, solve problems creatively, and contribute to building technology solutions that can grow and adapt over time.

References:

AWS Documentation. (n.d.). *AWS Cost Explorer User Guide.* Retrieved from [AWS Documentation Link]
Azure Pricing Calculator. (n.d.). *Azure Pricing Overview.* Retrieved from [Azure Documentation Link]
Docker Documentation. (n.d.). *Overview of Docker Containers.* Retrieved from [Docker Documentation Link]