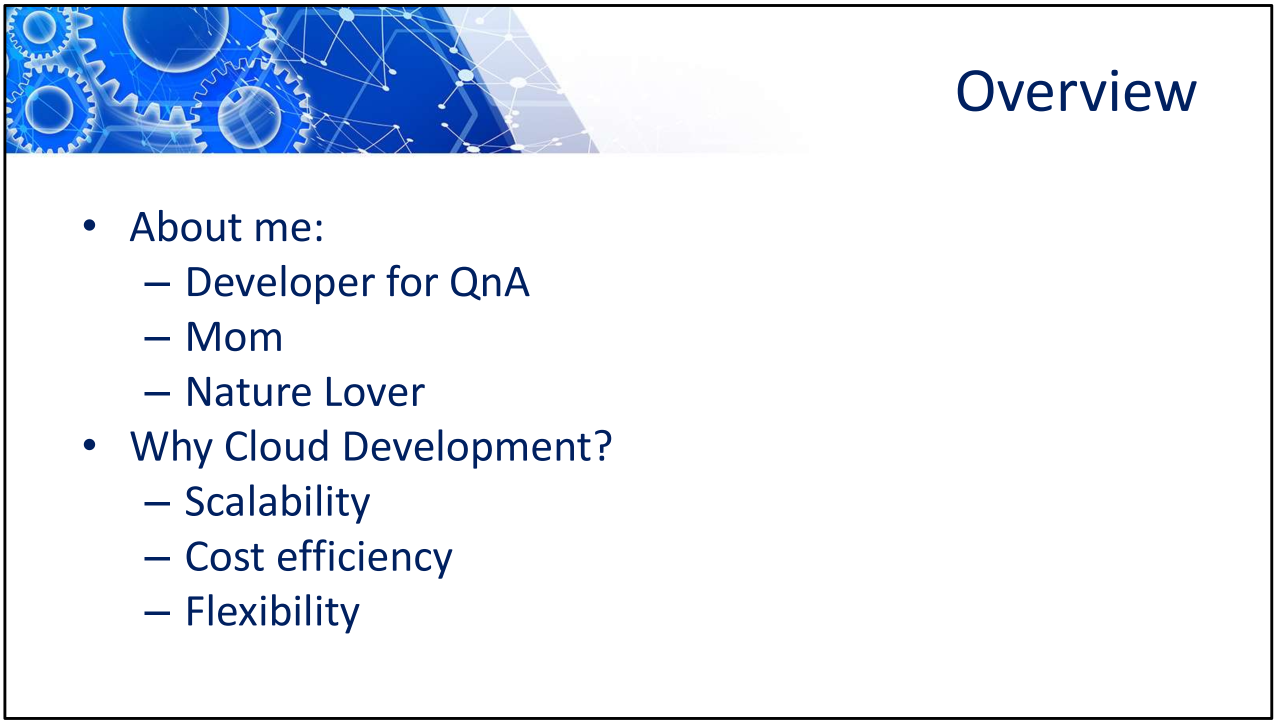
CS 470 Project Two Conference Presentation: Cloud Development

Erica Lerman

August/2025

Hi All! My name is Erica Lerman and I will be talking to you about Cloud Development today.



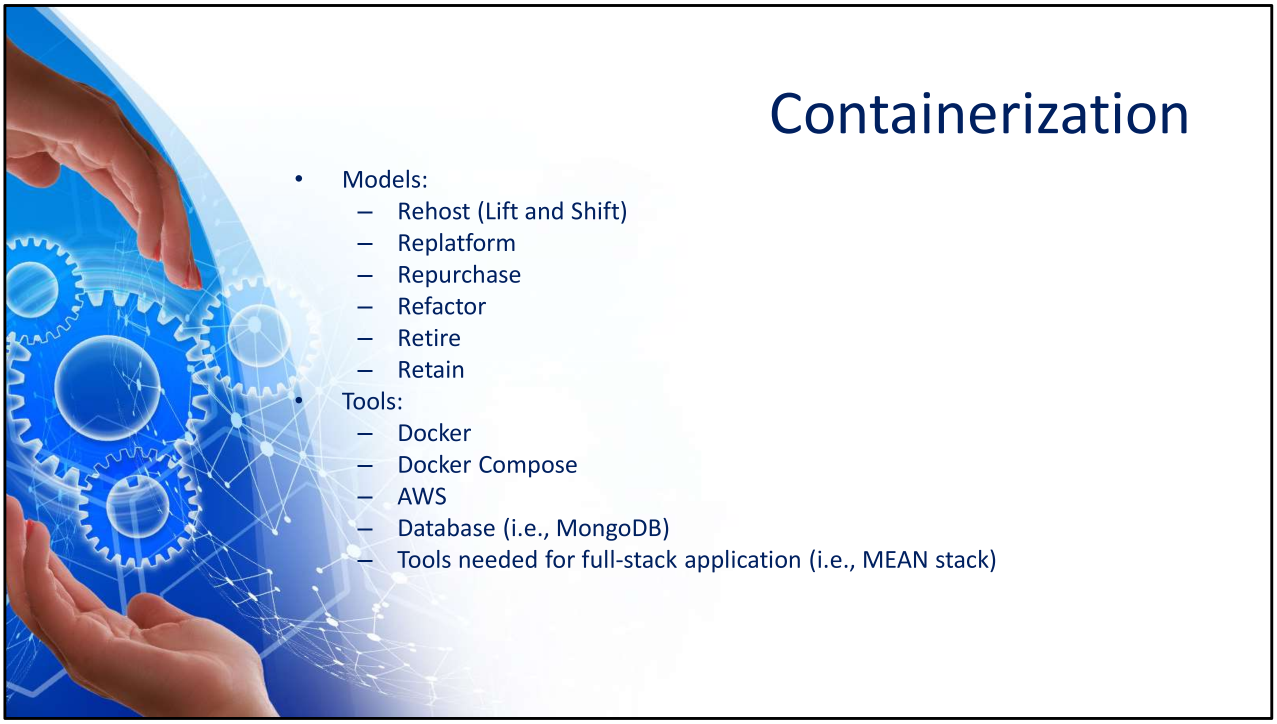
Overview

- About me:
  - Developer for QnA
  - Mom
  - Nature Lover
- Why Cloud Development?
  - Scalability
  - Cost efficiency
  - Flexibility

Before we start, let’s discuss who is presenting to you. So, here’s a little about me. I am a developer for a small start up company called QnA that has been growing at a considerable rate since our application has been deployed to consumers. I’m also a mom of a little boy and a nature lover. I appreciate spending time in the mountains with my family and the thrill of skiing in the winter.

Now let’s discuss the purpose of today’s presentation. Why Cloud Development?

Cloud Development provides companies with scalability. This means being able to grow your application when needed as your company grows (Xperity, 2023). It is cost efficient as many vendors offer pay-as-you-go plans which can greatly reduce those high up-front costs (Xperity, 2023). The cloud is also flexible. It allows for easy collaboration for developers in a central location, provides a way to go from building to deployment faster, and offers more control through various integrated management (Xperity, 2023).



Containerization

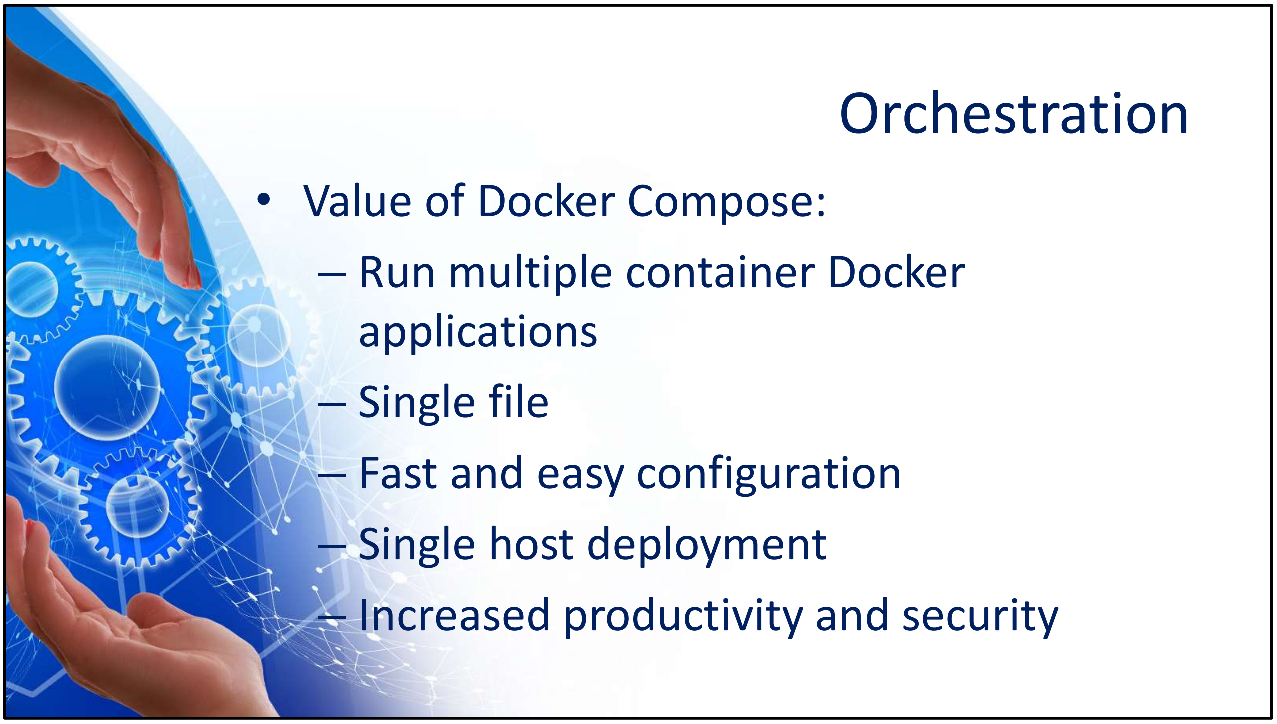
- Models:
  - Rehost (Lift and Shift)
  - Replatform
  - Repurchase
  - Refactor
  - Retire
  - Retain
- Tools:
  - Docker
  - Docker Compose
  - AWS
  - Database (i.e., MongoDB)
  - Tools needed for full-stack application (i.e., MEAN stack)

To start transferring your application to the cloud, you will need to look at containerization.

Containerization is a way to run software when it is moved from one computing environment to another and includes the application and everything needed to run it (Rubens, 2017).

Models:
Models for containerization include rehosting, replatforming, repurchasing, refactoring, retiring, or retaining.
The rehost or lift and shift model is a direct migration to the cloud (Geeks for Geeks, 2025).
The replatform approach is to move the application to the cloud while applying some optimization with a focus on stability and practicality (Geeks for Geeks, 2025).
Repurchasing is to move to a special product or licensing model available in the cloud (Geeks for Geeks, 2025).
The refactor model is used when a robust business needs to add features, scale, or increase performance and takes full advantage of cloud features (Geeks for Geeks, 2025).
The retire approach can be used for legacy systems and can provide businesses the ability to migrate and maintain resources that matter (Geeks for Geeks, 2025).
Finally, retaining is to leave things as is (Geeks for Geeks, 2025).

Tools:
Tools needed to move your application to the cloud include Docker, Docker Compose, AWS, Database, and the original tools you used for creating the full-stack application.



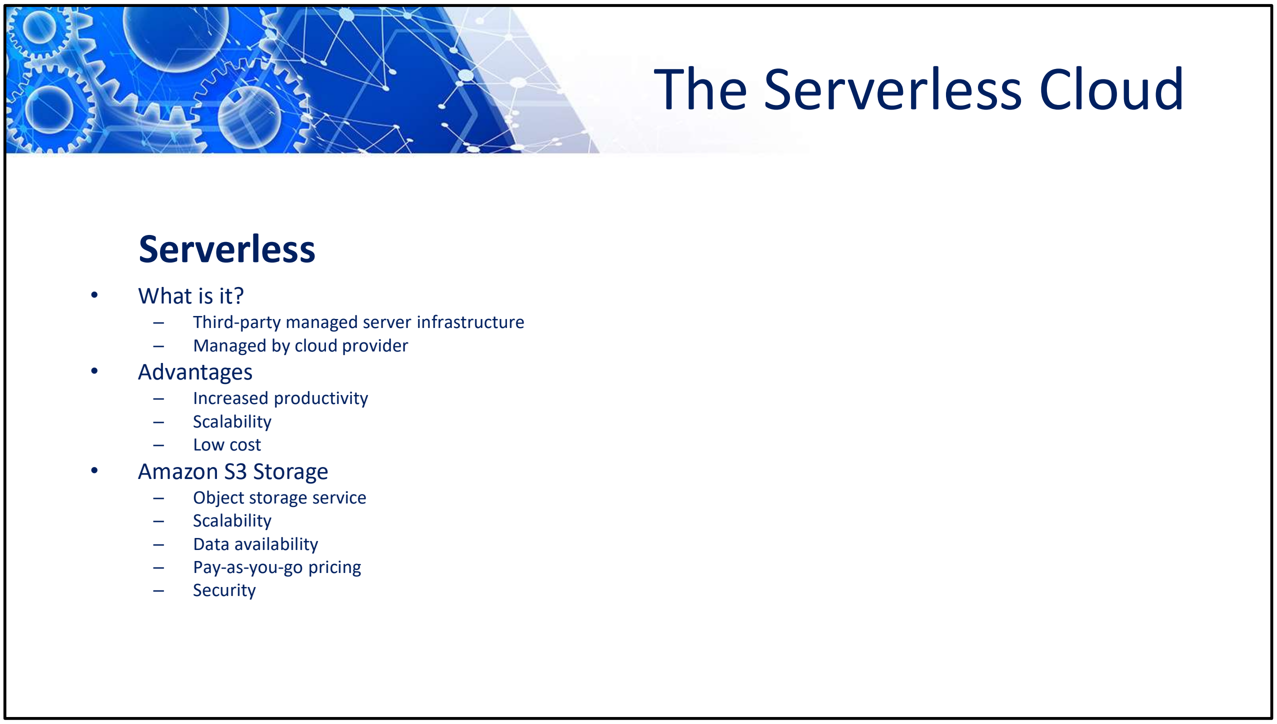
Orchestration

- Value of Docker Compose:
  - Run multiple container Docker applications
  - Single file
  - Fast and easy configuration
  - Single host deployment
  - Increased productivity and security

Docker Compose is a tool that can be used to run multiple container Docker applications (Chakraborty, 2025). Compose allows for all services, networks, and volumes for an application to be defined in a single file, docker-compose.yml (Chakraborty, 2025). Once this is done, the application stack can be run or stopped with a single command (Chakraborty, 2025).

Compose also allows for fast and easy configuration, single host deployment, and increased productivity and security (Walker, 2025).

Keeping the configuration in a single file allows for the reuse of the stack in multiple environments, making it easier to track changes and prevent misconfiguration, and allows for shareability between developers (Docker Inc., 2025; Walker, 2025).



The Serverless Cloud

Serverless

- What is it?
  - Third-party managed server infrastructure
  - Managed by cloud provider
- Advantages
  - Increased productivity
  - Scalability
  - Low cost
- Amazon S3 Storage
  - Object storage service
  - Scalability
  - Data availability
  - Pay-as-you-go pricing
  - Security

Let’s talk about the serverless cloud.

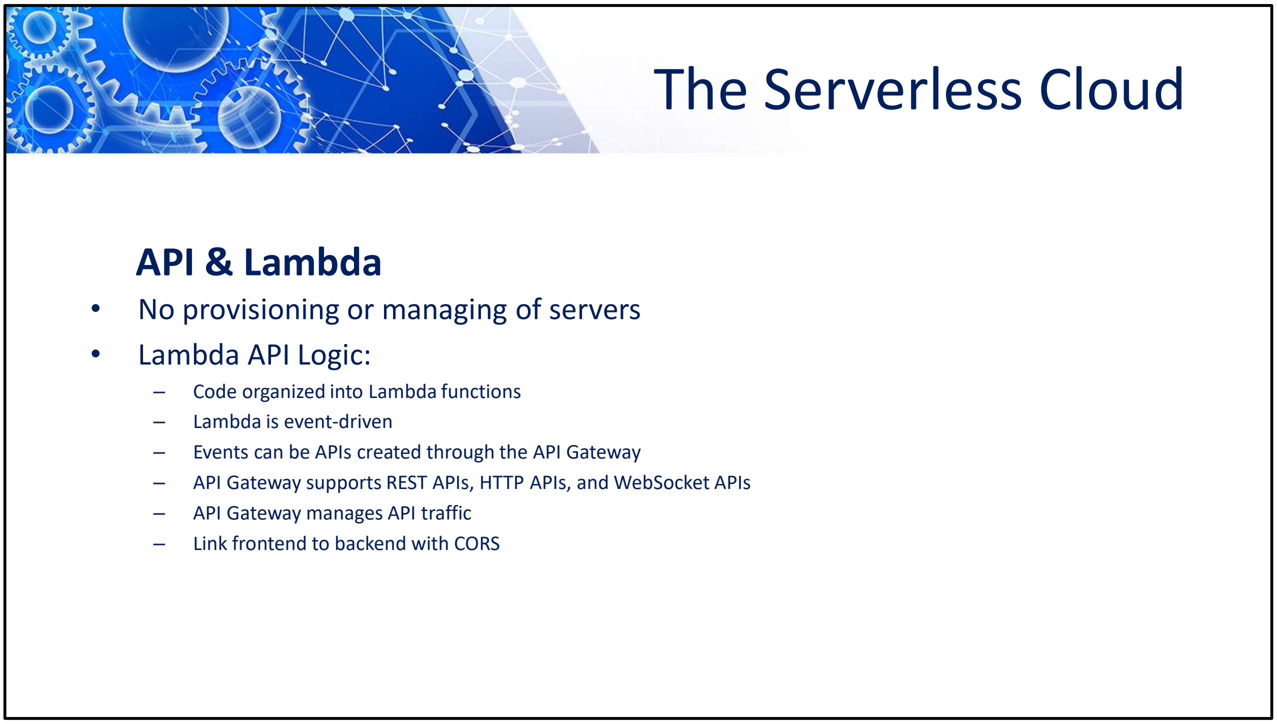
Being serverless means an application can be built and deployed on a third-party managed server infrastructure (Amazon Web Services, 2025e).
The cloud provider manages the operating system management, security, file system management, load balancing, monitoring, and logging (Amazon Web Services, 2025e).

So why is serverless infrastructure important?

Since the cloud team takes care of the server infrastructure for the application, developers can spend more time on developing the application itself (Amazon Web Services, 2025e).
This allows the application to go to market faster (Amazon Web Services, 2025e).
Serverless architecture also comes with automatic scaling based on demand and pay-for-value pricing, meaning you pay only for the resources used when the code runs (Amazon Web Services, 2025e).

Amazon Simple Storage Service is an example of a serverless architecture. It is an object storage service used to store and protect data (Amazon Web Services, 2025c).
Compared to local storage, S3 provides unlimited storage (no need to add more servers down the line), increased availability (no down time for adding more storage), pay-as-you-

go pricing, and AES-256 encryption (Villanueva, 2025).



The Serverless Cloud

API & Lambda

- No provisioning or managing of servers
- Lambda API Logic:
  - Code organized into Lambda functions
  - Lambda is event-driven
  - Events can be APIs created through the API Gateway
  - API Gateway supports REST APIs, HTTP APIs, and WebSocket APIs
  - API Gateway manages API traffic
  - Link frontend to backend with CORS

Using a serverless API like AWS Lambda and API Gateway allows for code to be run without needing to manage or provision servers (Amazon Web Services, 2025d).

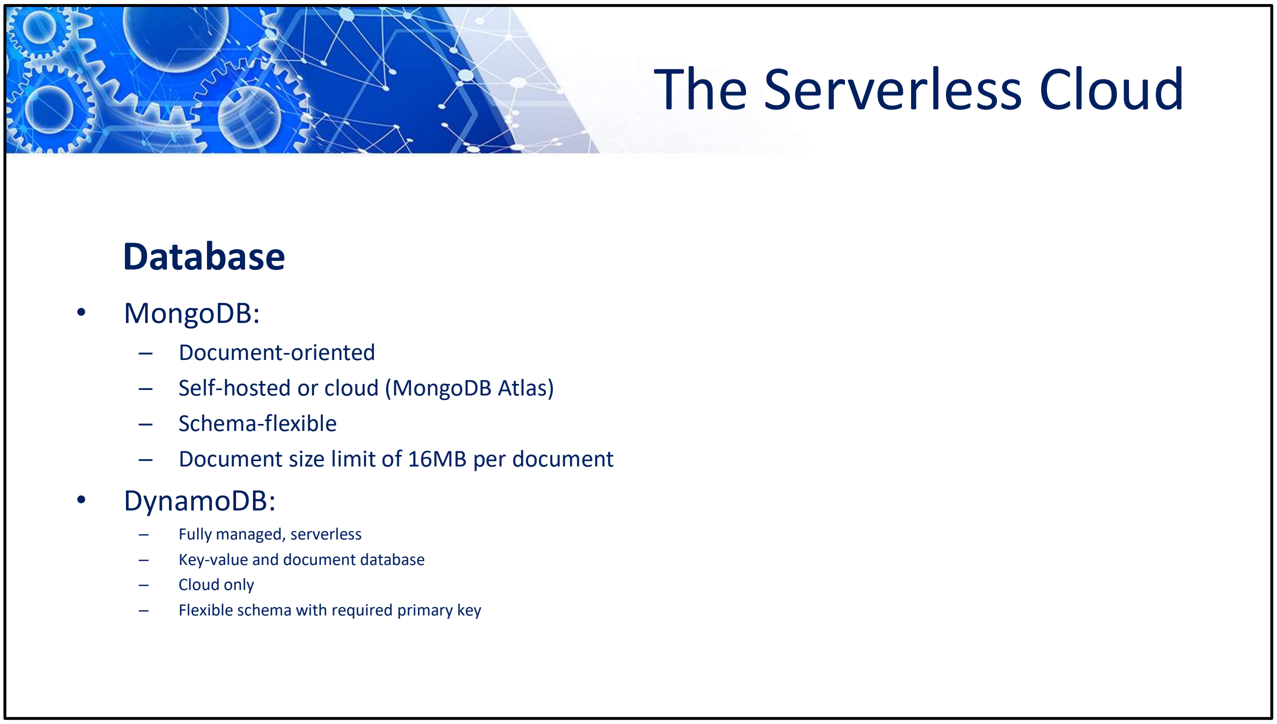
Lambda manages server and OS maintenance, capacity provisioning, automatic scaling, and logging (Amazon Web Services, 2025d).

With AWS Lambda and API Gateway, code is organized into Lambda functions, and the Lambda service runs those functions when needed and scales automatically (Amazon Web Services, 2025c). Events trigger Lambda functions to run (Amazon Web Services, 2025d).

A type of event that can trigger a Lambda function is an API call. Amazon’s API Gateway manages all API traffic (Amazon Web Services, 2025a). The API Gateway supports REST APIs, HTTP APIs, and WebSocket APIs (Amazon Web Services, 2025a).

An API call can be created to PUT, POST, DELETE, or GET data and when an API is called it triggers a Lambda function to run code for that specific API call. This code will access data stored within the database and return or manipulate the database based on the Lambda function code triggered by the specific API call.

To link the frontend and backend of an application, a developer can utilize cross-origin resource sharing. CORs is an HTTP-based mechanism that allows a server to indicate a domain, scheme, or port other than its own that a browser can permit loading resources from (Mozilla, 2025).



The Serverless Cloud

Database

- MongoDB:
  - Document-oriented
  - Self-hosted or cloud (MongoDB Atlas)
  - Schema-flexible
  - Document size limit of 16MB per document
- DynamoDB:
  - Fully managed, serverless
  - Key-value and document database
  - Cloud only
  - Flexible schema with required primary key

MongoDB vs. DynamoDB:

Both MongoDB and Amazon DynamoDB are NoSQL databases that allow for ACID transactions (Adela, 2025). MongoDB is a document-oriented database that can be self-hosted or deployed to the cloud (Adela, 2025). It is fully schema-flexible with a document size limit of 16MB per document (Adela, 2025). Performance depends on the configuration used and pricing can be instance-based, if using Atlas, or free for the Community Edition (Adela, 2025).

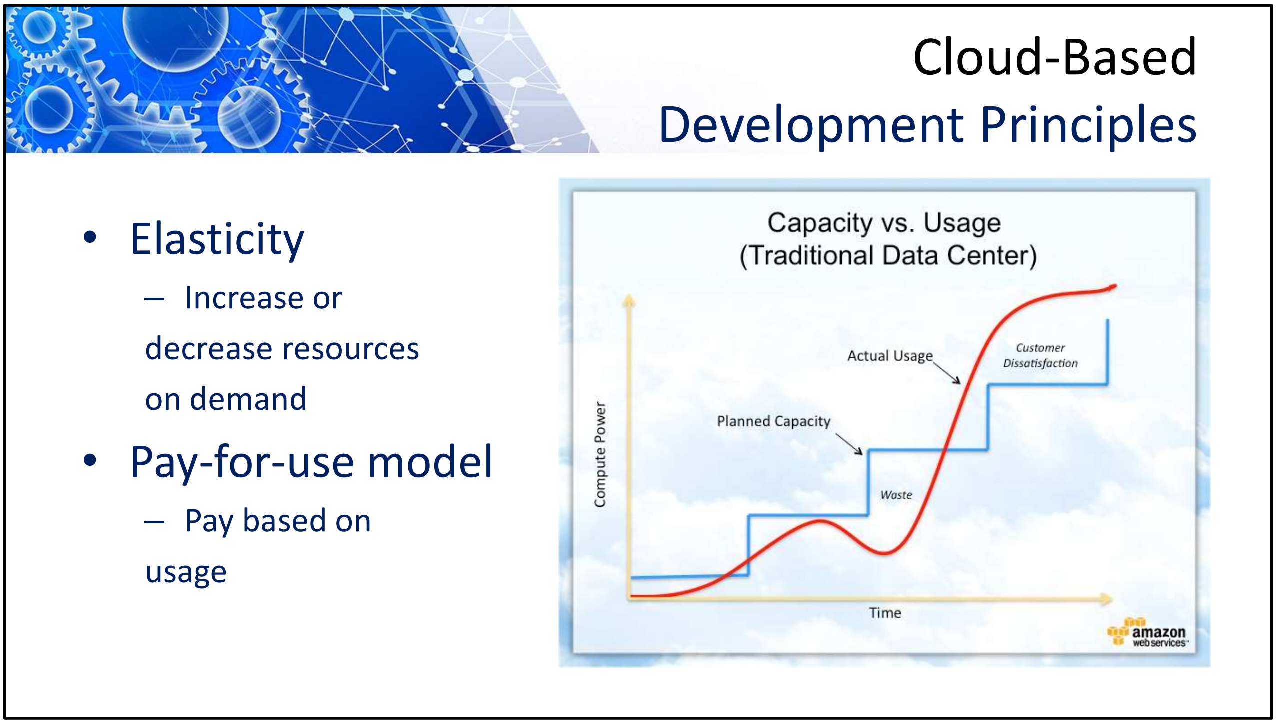
Amazon DynamoDB is a fully managed, serverless, key-value and document database (Adela, 2025). It can only be deployed in the cloud and uses a flexible schema with a required primary key (Adela, 2025). DynamoDB document size limit is 400KB per item and it uses limited query expressions (Adela, 2025). DynamoDB also allows for automatic, unlimited horizontal scaling and pricing is pay-per-request (Adela, 2025). Continuous backups with point-in-time recovery is also supported, as well as millisecond response times (Adela, 2025).

Queries:

The queries used for the application I worked on included queries to find records of data, create records of data, update data, and delete a record.

Scripts produced:

Lambda functions such as (FindOneQuestion) that were triggered by API calls (GET API call in Questions API) created in the Amazon API Gateway.



Cloud-Based Development Principles

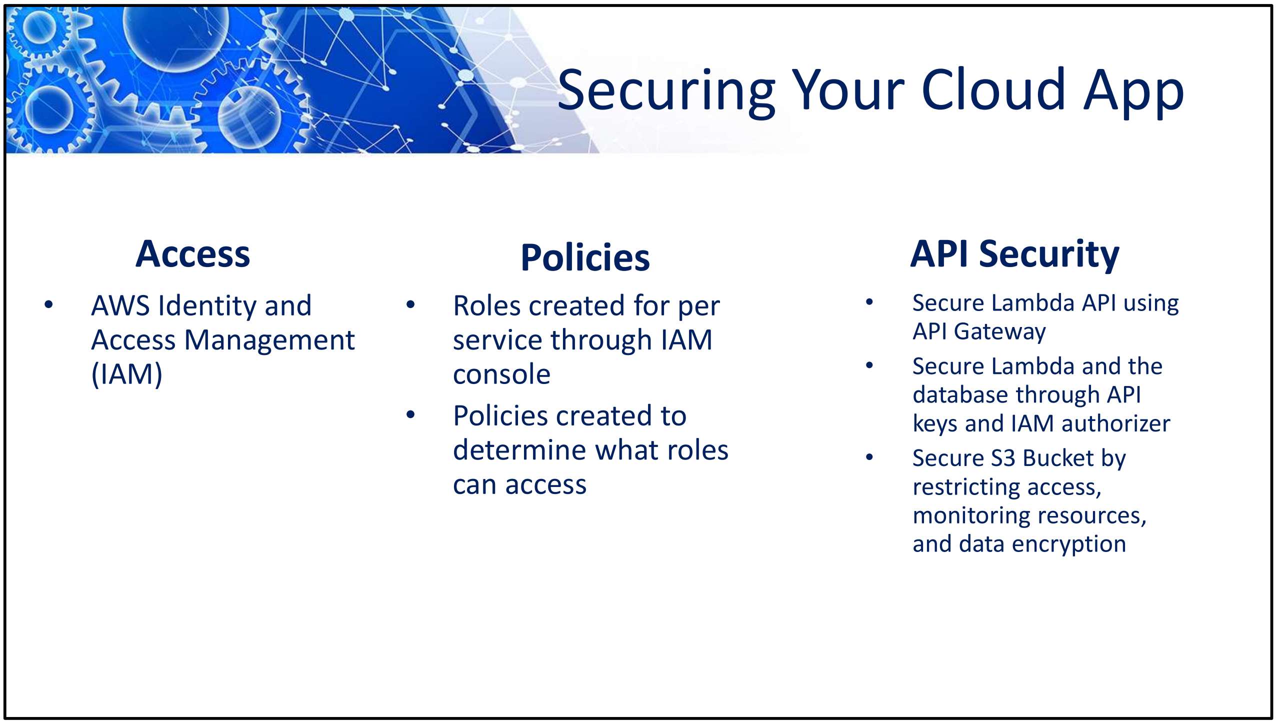
- Elasticity
  - Increase or decrease resources on demand
- Pay-for-use model
  - Pay based on usage

In the graph shown here you can see what a traditional data center does as usage increases. You can see that as usage of the data center increases over time, the capacity of the center can eventually not keep up, which leads to customer dissatisfaction. Additionally, you can see near the middle of the chart that planned capacity is greater than actual usage leading to a waste in resources.

Elasticity in the cloud means the system increases or decreases resources based on demand (Slingerland, 2024). Resources could be CPU, RAM, or network bandwidth (Slingerland, 2024).

Cloud computing also offers a pay-for-use model, meaning you can pay based on resource usage (Yasar, 2024).

With both elasticity and a pay-for-use model, a company can only utilize the resources it needs at the time it needs them, eliminating issues like wastage or customer dissatisfaction.



Securing Your Cloud App

Access

- AWS Identity and Access Management (IAM)

Policies

- Roles created for per service through IAM console
- Policies created to determine what roles can access

API Security

- Secure Lambda API using API Gateway
- Secure Lambda and the database through API keys and IAM authorizer
- Secure S3 Bucket by restricting access, monitoring resources, and data encryption

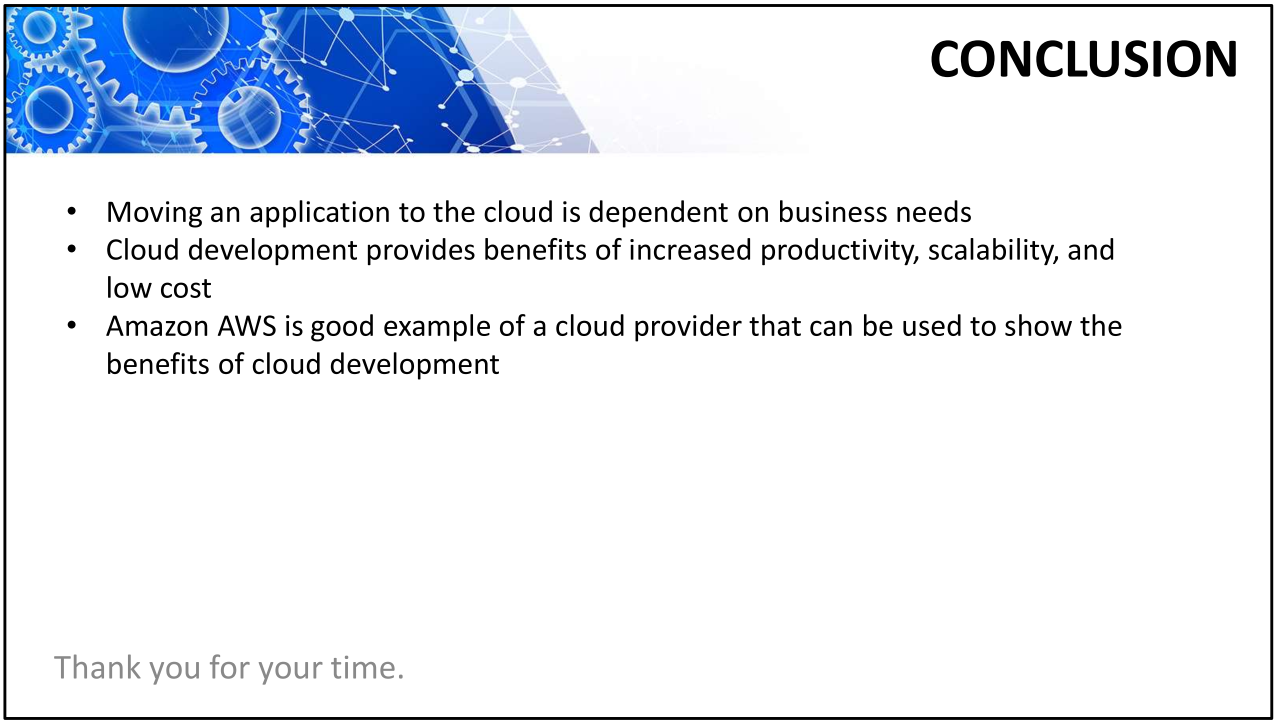
Amazon's Identity and Access Management can be used to manage access to computing, storage, database, and applications services in the cloud (Amazon Web Services, 2014). IAM permissions can be applied to individual API calls and specified for who can access specific services, what actions a user can perform, and what resources are available (Amazon Web Services, 2014). Using the principle of least privilege is best practice when creating roles through IAM.

Roles created through IAM need policies associated with them. Policies are documents that contain a set of rules (Learn AWS, 2022). Policies are attached to IAM roles to give a role access to AWS resources (Learn AWS, 2022).

For this project, a custom read only policy was used when creating the S3 bucket and the LabRole was used when creating Lambda functions. Additionally, permission needed to be given to the API Gateway to invoke the Lambda functions when creating API calls.

To ensure security throughout the entire application in the cloud, we will look at API Security as a whole. The Lambda API can be secured using the API Gateway so that function code is only triggered by API calls.
API keys and IAM authorizer can be used in securing Lambda and the database (Isenberg,

2025).
Your S3 bucket can be secured through the best practices of restricting access to your S3 resources to only people that need it, monitoring S3 resources through logs, and using encrypting in multiple ways (Amazon Web Services, 2025b). AWS S3 supports encrypting during transmission, server-side encryption, and client-side encryption (Amazon Web Services, 2025b).



CONCLUSION

- Moving an application to the cloud is dependent on business needs
- Cloud development provides benefits of increased productivity, scalability, and low cost
- Amazon AWS is good example of a cloud provider that can be used to show the benefits of cloud development

Thank you for your time.

References

Adela. 2025. *DynamoDB vs. MongoDB: A complete comparison in 2025*. Bytebase. https://www.bytebase.com/blog/dynamodb-vs-mongodb/

Amazon Web Services. 2025. *Amazon api gateway features*. Amazon Web Services, Inc. https://aws.amazon.com/api-gateway/features/#topic-0

Amazon Web Services. 2025. *How can I secure the files in my Amazon S3 bucket?.* Amazon Web Services, Inc. https://repost.aws/knowledge-center/secure-s3-resources

Amazon Web Services. (2014, March 14). *Introduction to AWS IAM – Identity and access management on AWS* [Video]. YouTube. https://www.youtube.com/watch?v=Ul6FW4UANGc

Amazon Web Services. 2025. *What is Amazon S3?.* Amazon Web Services. https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html

Amazon Web Services. 2025. *What is AWS Lambda?.* Amazon Web Services, Inc. https://docs.aws.amazon.com/lambda/latest/dg/welcome.html

Amazon Web Services. 2025. *What is serverless computing?.* Amazon Web Services. https://aws.amazon.com/what-is/serverless-computing/

Chakraborty, S. 2025. *Why do we need docker compose?.* Medium. https://medium.com/@sd24chakraborty/why-do-we-need-docker-compose-df528aff5c74

Docker Inc. 2025. *Why use compose?.* Docker Inc. https://docs.docker.com/compose/intro/features-uses/

Geeks for Geeks. 2025. *6 strategies for migrating applications to AWS cloud*. Geeks for Geeks. https://www.geeksforgeeks.org/cloud-computing/6-strategies-for-migrating-applications-to-aws-cloud/

Isenberg, R. 2025. 14 AWS Lambda security best practices to secure your serverless applications. Ran the Builder. https://www.ranthebuilder.cloud/post/14-aws-lambda-security-best-practices-for-building-secure-serverless-applications#viewer-bta5n155328

Learn AWS. 2022. *What are the differences between IAM roles and IAM policies?.* Learn AWS. https://www.learnaws.org/2022/03/03/iam-roles-policies/

Mozilla. 2025. *Cross-origin resource sharing (CORS).* Mozilla. https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/CORS

Rubens, P. 2017. *What are containers and why do you need them?.* CIO. https://www.cio.com/article/247005/what-are-containers-and-why-do-you-need-them.html

Slingerland, C. 2024. *What is cloud elasticity? (And how does it affect cloud spend?).* CloudZero. https://www.cloudzero.com/blog/cloud-elasticity/

Villanueva, J. C. 2025. *Amazon s3 vs local storage – where should you store files uploaded to your file transfer server?.* JSCAPE by Redwood. https://www.jscape.com/blog/amazon-s3-vs-local-storage-where-should-you-store-files-uploaded-to-your-file-transfer-server#:~:text=To%20choose%20between%20Amazon%20S3%20and%20local%20storage,compliance%20options%2C%20and%20robust%20security%20with%20AES-256%20encryption.

Walker, J. 2025. *Docker compose – What is it, example & tutorial*. Spacelift. https://spacelift.io/blog/docker-compose#docker-compose-benefits

Xperity. 2023. *13 reasons why application development is better in the cloud*. Xperity. https://xperity.io/13-reasons-why-application-development-is-better-in-the-cloud/

Yasar, K. 2024. *What is pay-as-you-go cloud computing (PAYG cloud computing)?.* TechTarget. https://www.techtarget.com/searchstorage/definition/pay-as-you-go-cloud-computing-PAYG-cloud-computing