

Southern New Hampshire University

CS-470 Full Stack Development II

Project Two Conference Presentation: Cloud Development

Eric Slutz

10/15/2023

https://youtu.be/jaa1_rPTk9Y



This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com

Hello, my name is Eric Slutz.



This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com

I'm a student at Southern New Hampshire University, currently completing my last term toward earning my Computer Science degree. Today I'll be sharing with you a presentation on migrating a full stack application to the cloud and is meant for both a technical and non-technical audience. I'll go over the use of Docker and containers as well as the different AWS services utilized to build a cloud native full stack application.



Containerization

- Migration Models
 - Rehost or Lift and Shift
 - Refactor
- Tools for Containerization
 - Docker
 - Docker Compose
 - MongoDB
 - ExpressJS
 - Angular
 - NodeJS

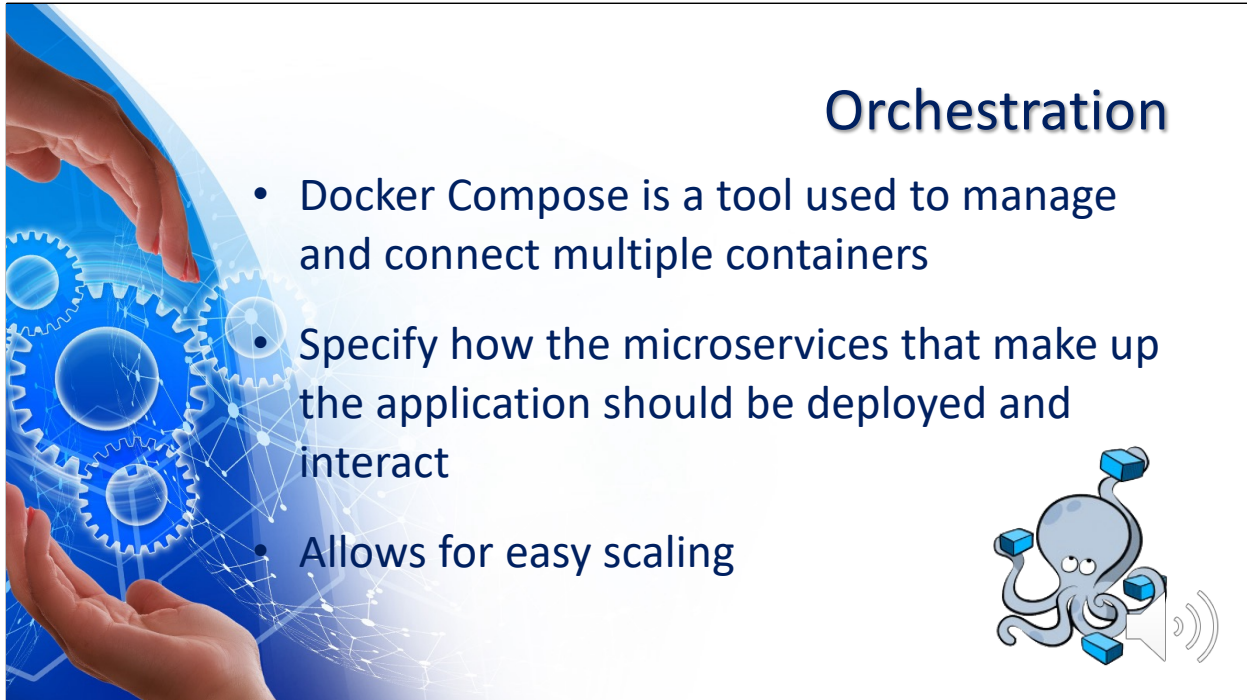


The slide features a collection of logos for various technologies and services. At the top right is the 'Containerization' title. Below it are logos for 'aws' (Amazon), 'docker' (a blue ship icon), 'MongoDB' (a green leaf icon), 'ex' (ExpressJS), 'A' (Angular), and 'node' (NodeJS). A small speaker icon is located at the bottom right of the logo area.

This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com

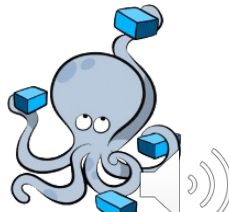
Of the various methods for migrating your application to the cloud, the two models that I will cover are Lift and Shift and Refactor. Lift and Shift is just taking your containers, in this case a MEAN stack application, and moving them from your current hosting environment to one of the cloud providers. Refactoring on the other hand, takes your current application and optimizes it for the cloud by utilizing cloud native services offered by the cloud provider.

Docker was the primary tool for creating the frontend, API, and database containers. Docker Compose was used to orchestrate the containers; allowing them to communicate with each other. The MEAN stack of MongoDB, ExpressJS, Angular, and NodeJS was used to build the application.

The slide features a blue background with a graphic of hands interacting with gears and a network of nodes. The title 'Orchestration' is in the top right. A bulleted list is in the center, and a cartoon octopus is in the bottom right.


Orchestration

- Docker Compose is a tool used to manage and connect multiple containers
- Specify how the microservices that make up the application should be deployed and interact
- Allows for easy scaling



This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com


Docker Compose was the orchestration tool used to manage the containers for this application. By creating a Docker Compose config file you can manage how the container for each microservice is deployed and how the containers can communicate and interact with each other. Lastly, Docker Compose allows you to easily scale your application up or down as needed. These are just a few examples of the many ways that the use of Docker Compose can add great value to your application and streamline your development and deployment process.



The Serverless Cloud

Serverless

- Application and services run on servers that are managed by the cloud provider
- Removes server maintenance, updating, and patching from your responsibilities
- Automatically scales based on demand



This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com

When talking about the cloud and serverless architecture, what that means is that your applications and services run on servers that are managed by the cloud provider. The cloud provider is then the one who is maintaining, updating, and patching the servers, removing it from your responsibilities. Another benefit of serverless architecture is that it can easily scale based on the demand of your application.



The Serverless Cloud

S3 Storage

- Scalable, growing as needed, with multiple levels of accessibility
- Replicates data throughout a region
- Pay only for storage used

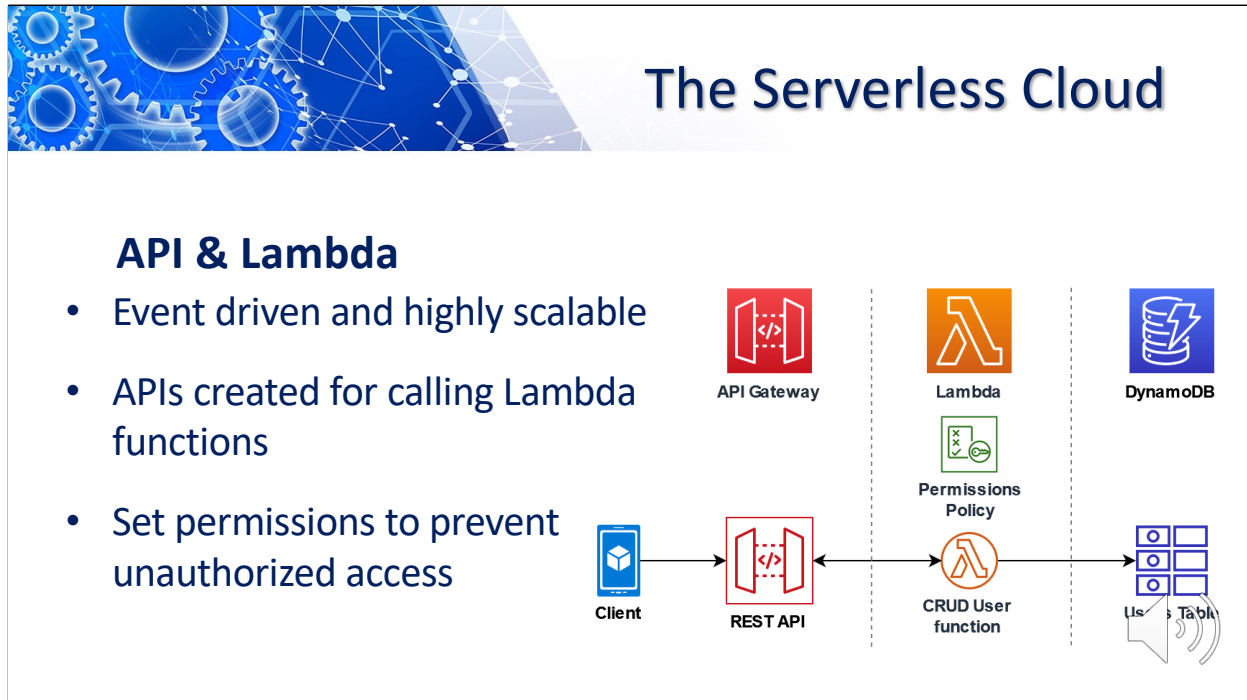


This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com

S3 is a scalable storage service offered by AWS, that can quickly grow to whatever size you need, and offers multiple levels of accessibility. Local disk storage on the other hand, is limited to the physical size of the installed disk. If more storage is needed, you must obtain another disk and work to get it integrated locally. S3 takes care of all of that for you. Also, S3 can offer a high level of redundancy by replicating data throughout a region. Local storage only gives you the option of storing data in at the location of the disk. Lastly, with S3 you pay for the storage you use. With local disk storage you're paying for the upfront hardware costs, with the possibility of purchasing more storage than you need or less than you end up needing. Either way you could end up spending more than initially estimated.

Additionally, a unique feature of S3 is Amazon S3 Intelligent-Tiering. According to Amazon, this is "the only cloud storage class that delivers automatic storage cost savings when data access patterns change, without performance impact or operational overhead. The Amazon S3 Intelligent-Tiering storage class is designed to optimize storage costs by automatically moving data to the most cost-effective access tier when access patterns

change."

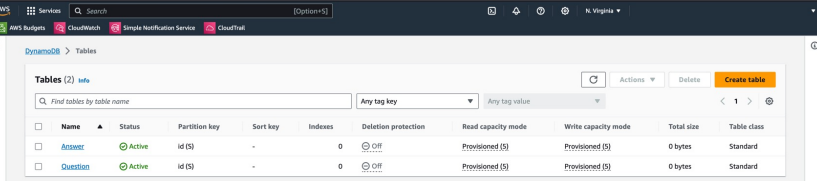


This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com

The Amazon API Gateway and AWS Lambda are both components of serverless architecture and are highly scalable, event driven services. Using Amazon API Gateway, you can create and manage a group of API endpoints. In this case, an endpoint was created for each of the different CRUD operations, using the different HTTP requests such as GET, POST, PUT, and DELETE. This in turn, called our Lambda functions to perform the requested operation on our database. Additionally, endpoints were created to search for and retrieve different types of data from our database. Once again utilizing Lambda functions to perform the operation and return data. Lastly, to allow communication between our frontend and the backend, using the API Gateway, CORS or Cross-Origin Resource Shared was enabled through the OPTIONS method of the API requests. For each of the different Lambdas, access policies were created to ensure only authorized users could gain access.

The Serverless Cloud

Database



Name	Status	Partition key	Sort key	Indexes	Deletion protection	Read capacity mode	Write capacity mode	Total size	Table class
Answer	Active	id (S)	-	0	OFF	Provisioned (S)	Provisioned (S)	0 bytes	Standard
Question	Active	id (S)	-	0	OFF	Provisioned (S)	Provisioned (S)	0 bytes	Standard

- **MongoDB**
 - NoSQL database
 - Uses JSON like data structure
 - Can be run anywhere
 - Can have multiple tables
- **DynamoDB**
 - NoSQL database
 - Uses a key-value pair data structure
 - Can only be used with AWS
 - Uses single table model

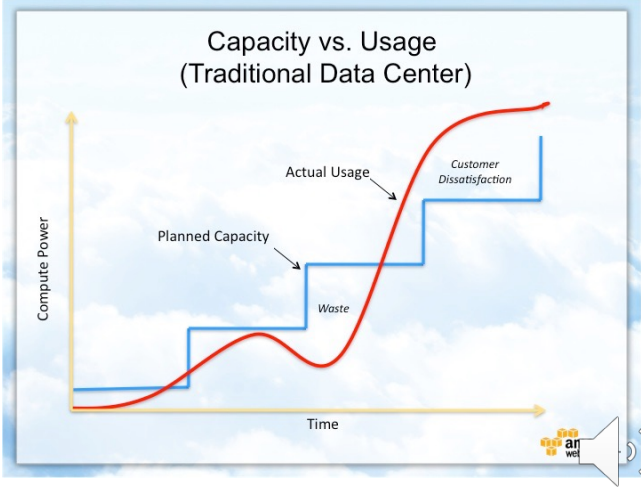
This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com

Both MongoDB and DynamoDB are considered NoSQL databases. MongoDB stores data in documents in a JSON like structure and supports multiple tables. DynamoDB uses key-value pairs for its data. MongoDB can be run anywhere, whereas DynamoDB is a managed service that is offered through AWS. Since DynamoDB is a managed service, you don't need manage updating database versions or setting up security. With MongoDB though, you are responsible for all management. Additionally, both databases are very scalable.

For use with this application, queries were used to find, create, update, and delete any record from the database. The queries were written as AWS Lambda functions that were called by API endpoints set up in Amazon API Gateway.

Cloud-Based Development Principles

- Elasticity: The ability to scale up or down based on current needs
- Pay-for-use model: Only pay for the resources you are using



This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com

Elasticity refers to the ability of the cloud resources to scale up or down based on the current needs of your application. With the traditional data center, you have to try to predict your future needs far enough out in advance and spend the up-front cost to plan, purchase, and setup any needed additional infrastructure. If your prediction is off and your estimate is too high, you can end up wasting money on purchasing unnecessary equipment and its associated costs. On the other hand, if your prediction is off and your estimate is too low you could be left with a poorly performing application and dissatisfied customers.

By migrating to the cloud, you end up in a pay-for-use model. What this means is, you are only paying for the resources you use. That could mean paying for the storage your data takes up, the minutes your Lambda function is running and processing data, or the number of calls made to your API. In the end, how much you pay is directly tied to how much you use, as opposed to the traditional data center where you could purchase too much and have waste, or not purchase enough and lose money from missed opportunities with customers due to poor application performance or customer dissatisfaction, or from having to pay extra to quickly scale up your applications.


Securing Your Cloud App

Access

- How can you prevent unauthorized access?
 - Use AWS Identity and Access Management (IAM) to create roles and security policies
 - Apply the principle of least privilege when giving permissions

This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com

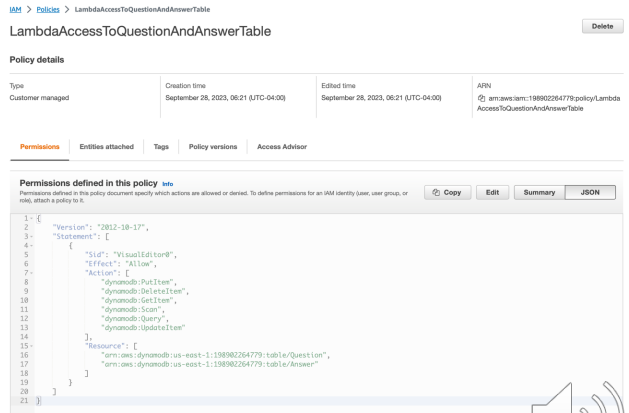
To prevent unauthorized access to your application you can create AWS IAM roles and policies. With the policies, you can apply the principle of least privilege to help ensure that users do not end up with access to something that they shouldn't have.



Securing Your Cloud App

Policies

- Each role is defined at least one policy, but can have multiple
- Users are then assigned roles based on what access they need



This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com

You create a role to assign one or more access policies to. Roles are then assigned to users based off what the user needs access to. If you need to update the access policy for a certain type or group of users, you will then just need to update the policy for the role those users share. This way you only need to update the policy once with the role, instead of having to go to each user and update their individual access policies. Another advantage of using roles having more defined and consistent access across your organization.


For this application, a custom policy called `LambdaAccessToQuestionAndAnswerTable` was created to allow the Lambda functions to read, write, update, and delete data from the DynamoDB table.



Securing Your Cloud App

API Security

- Between S3 and Lambda, the connection is already secure
- Between the API Gateway and Lambda, the connection can be secured using API keys
- Between Lambda and DynamoDB, the connection can be secured using IAM roles and policies



This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com

The S3 bucket needs to be set to public so that the website can be viewed. The connection from S3 to Lambda is automatically secured by AWS. Between the API Gateway and Lambda, the connection can be secure by utilizing API keys. That way, without the key, any unauthorized user would be unable to call the endpoint and gain access to the Lambda function. Between Lambda and DynamoDB, the connection is secured using the custom role and policy we create to only allow certain Lambdas access to modify the database.



Conclusion

- Cloud development is flexible
- Cloud development can lower overhead for an application
- Cloud development is secure

Thank you for your time.



This presentation uses a free template provided by FPPT.com
www.free-power-point-templates.com

In conclusion, cloud development is flexible. It can scale easily to meet your needs and with the pay-to-use model you're only paying for what you use. That model contributes to a lower overhead cost for your application by eliminating the need to spend money upfront on required infrastructure. Additionally, by using serverless architecture the overhead required for managing and maintaining the necessary hardware is also eliminated. This also goes into the last point of cloud development is secure. By having the cloud provider manage all the infrastructure you can be assured that the systems you are using are updated and patched with any security fixes. That, along with creating some well-defined security policies should keep your application secure in the cloud.