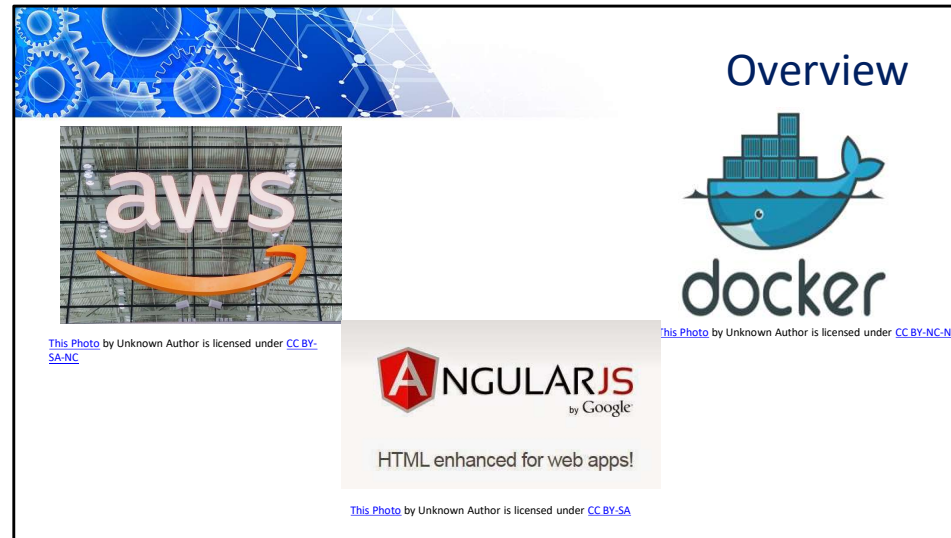


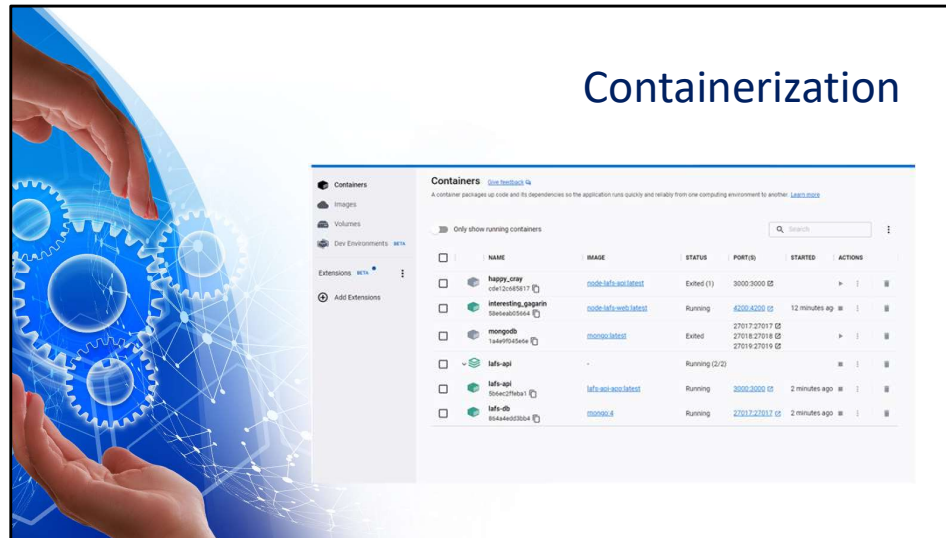


CS 470 Project Two  
Conference Presentation:  
Cloud Development

Jeremia Faust  
February 2023



Hello, My name is Jeremia Faust and today we are going to talk about taking a web application from a standard full stack application to a serverless cloud application. We will start by looking at containerization using the application Docker, then we will migrate the web application to run in the cloud, using Amazon Web Services or AWS.



Containerization is a relatively new technology that contains everything that an application needs to run while being stripped of the operating system. This allows for greater portability to work on different systems without running into compatibility issues. It also improves efficiency by only running the resources that are required to run the application. This is conducive to running applications using cloud services, as the use of resources cost money. Virtual Machines or VMs are abstractly similar to containers, but it contains a virtual copy of an entire system. The problem with this approach, is that they consume a lot of systems resources and have a large overhead. While most serverless companies have their own containerization programs, the most popular is docker. Docker is an open-source program that is more secure than other technologies and very easy to use.


Containerization is a great technology, but it has its flaws. If the containers are not set up correctly, they can become a security problem. It also provides easy access where the application can be stolen by a rogue Developer since it is a lightweight program. Extra care will be needed by using some sort of moderating system to protect the application from theft and attacks. Another potential problem is with data storage. If a container is unexpectedly shut down, that data is then lost. It is important to build some sort of fail-safe or redundancies to protect against

data loss. To protect data from unexpected deletion of a container there are a few options that can be taken. The first option is to save the data using backup containers that back up data in real time for areas where data is located or being generated. There also are third party applications such as [neuvector.com](https://neuvector.com) that will further protect data using regulatory compliance laws. With VM many of these issues are not a large problem but that is mainly due to it being an older technology and having more support.

#### Resources:

*How to protect sensitive data in containers with container DLP.* Blog. (n.d.). Retrieved January 7, 2023, from <https://blog.neuvector.com/article/protect-sensitive-data-with-container-dlp>

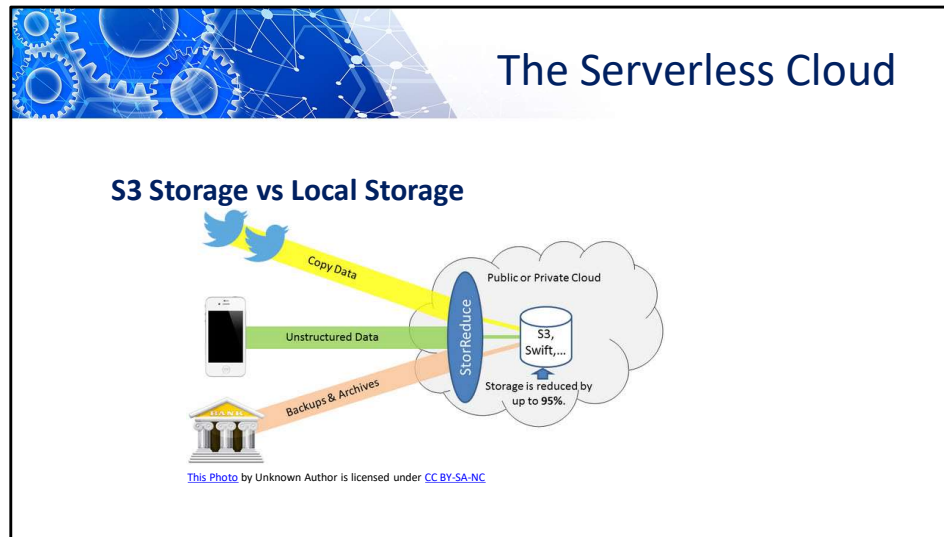
*What is containerization? what are the benefits?* Veritas. (n.d.). Retrieved January 5, 2023, from <https://www.veritas.com/information-center/containerization>



## Orchestration

- Allows for the use of multiple containers that work in concert with each other
- Increased security
- The ability to start each container at the same time.

Docker Compose is a tool within the Docker Desktop that allows for the use of multiple containers that work in concert with each other for one application. So, we have this application that has many services that then have their own database they connect too. Having each service in its own container then each database that is connected directly to a service is in its own container, provides a greater level of security. This is due to the ability to create connections that is limited in scope, meaning that only the login service has access to the account database, but the product service cannot connect to the account database. Finally, the greatest ability docker compose has is that it can start every single service at the same time and once you learn how to configure everything it is very quick to setup. In a Virtual machine, if one of the services goes down the entire application then has to be rebooted. These reboots can take time and that time could cost the company a potential customer or even cost them money. This is because Docker Compose will automatically allocate new resources and restart just that container that failed instead of the entire application.



Local storage is data is stored on onsite computers or servers that the company or person owns. S3 storage is a system designed by amazon to allow for data to be stored in the cloud on data servers that amazon maintains. The problem with local storage is the upfront cost on hardware and manpower to maintain the servers. Also, it is hard to figure out exactly how much data needs to be stored so there is the possibility that too much or too little storage space is bought. One great thing about local storage is that security is in the company hands and can be controlled to prevent access. S3 cloud services there is no need to worry about if you have enough storage space because as you grow there is always more space, and you are only charged for what you use.

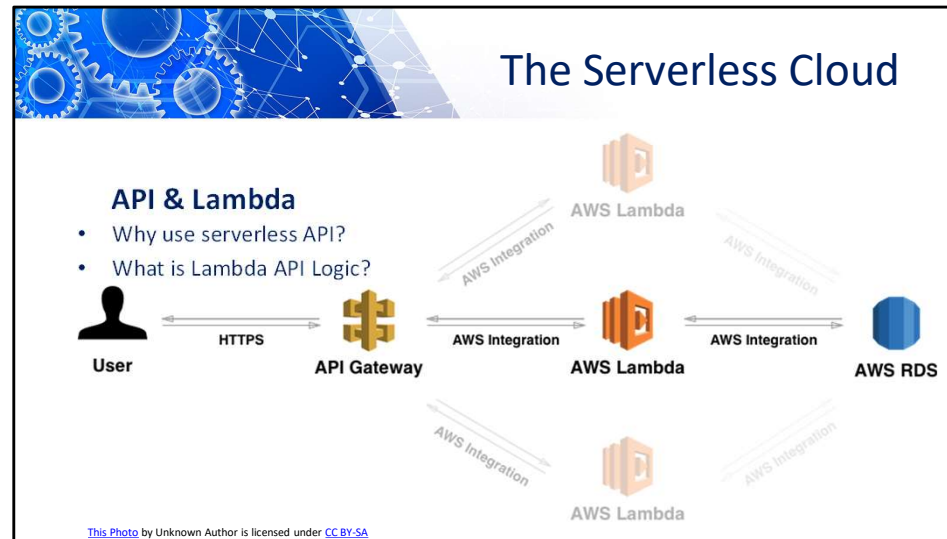
S3 storage can be accessed from any location that has an internet connection while at the same time providing a secure connection. Although local storage has the benefit of providing your own security, AWS security is one of the leading pioneers in API security. When you first start to upload data to AWS by default it is encrypted and copied and stored in different locations to not only provide security but provide backups if something goes wrong at one location. When using local storage this can be tricky and costly. One key feature of AWS is not only that they protect your data, but they also make sure you know how to use their services, that way you have the peace of

mind of your data being secure and you know how to use it.

#### Resources:

McCarthy, T. (2023, January 11). *Amazon S3 now Encrypts Data by default: TechTarget*. Storage. Retrieved January 19, 2023, from <https://www.techtarget.com/searchstorage/news/252529106/Amazon-S3-now-encrypts-data-by-default#:~:text=New%20data%20stored%20in%20Amazon%20S3%20will%20now,unique%20key%20and%20then%20encrypts%20the%20key%20itself>.

Posted by Henry Golas on April 12, & Golas, P. by H. (2022, January 25). *S3 storage: Behind the scenes*. Cloudian. Retrieved January 19, 2023, from <https://cloudian.com/blog/s3-storage-behind-the-scenes/>



Serverless technology one of the best solution for companies as there is no server management so that users can then focus on their applications. Users do not ever have to worry if they have enough resource as AWS has the ability to scale automatically. What this allows for is quick deployments and updates thus saving time and money. Lambda is a serverless event-based service. It is designed to run code that automatically scales up or down to the needs of the code being run. The great thing about lambda is that it only runs when something is triggered. Once the process is done the resources are then released for use. Lambda is perfect to use in applications such as Data analytics or server-less websites since Lambda users are charged by the request. To connect the front-end angular web application to the backend server DynamoDB we first needed to create the scripts for our Restful API. We created 6 Lambda functions with testing scripts that will later be used in API Gateway to connect the two services. The next three slides are GET function of our Restful API.

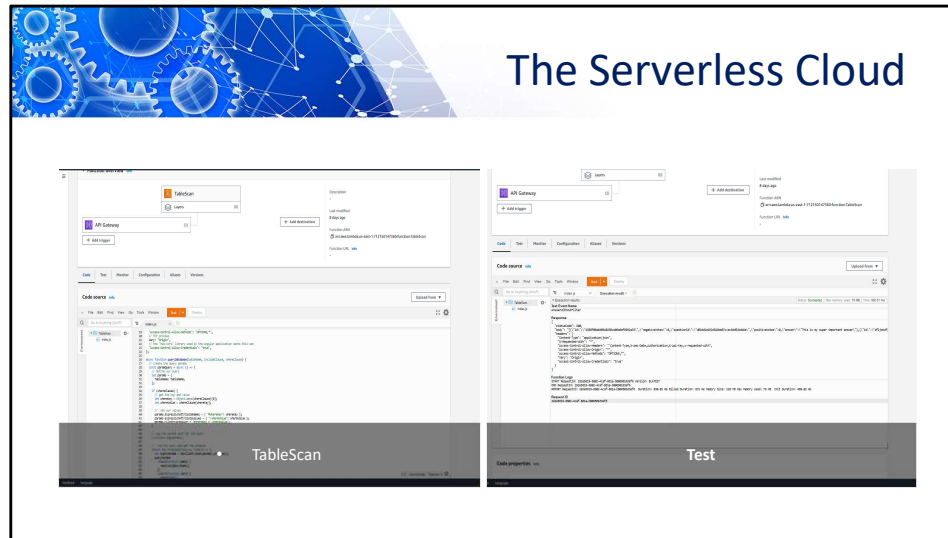
Resources:

Bigelow, S. J. (2021, October 22). *An overview of Amazon EC2 vs. Aws Lambda: TechTarget*. Cloud Computing.



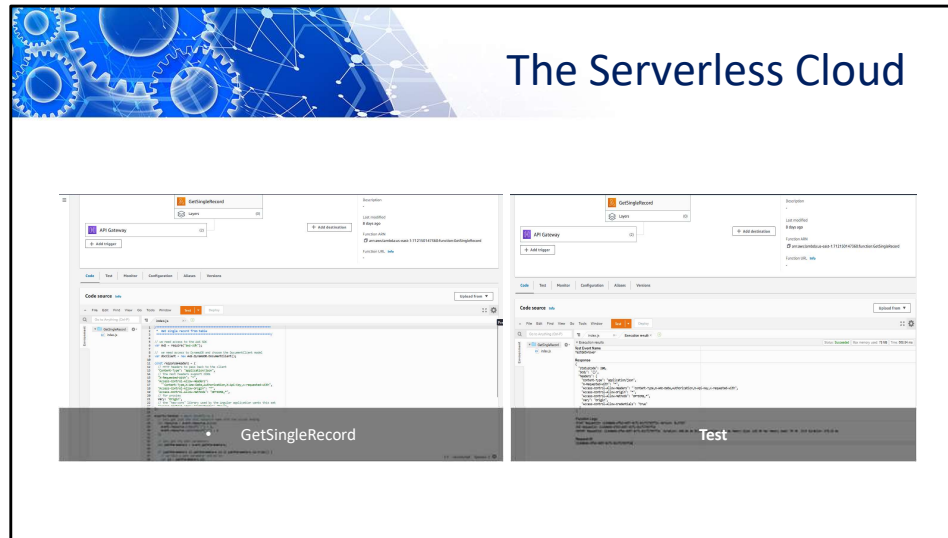
Retrieved January 25, 2023, from <https://www.techtarget.com/searchcloudcomputing/tip/An-overview-of-Amazon-EC2-vs-AWS-Lambda#:~:text=and%20Lambda%20together-,Lambda%20is%20ideal%20for%20short%2Dterm%20tasks.,both%20platforms%20can%20work%20together.>

e



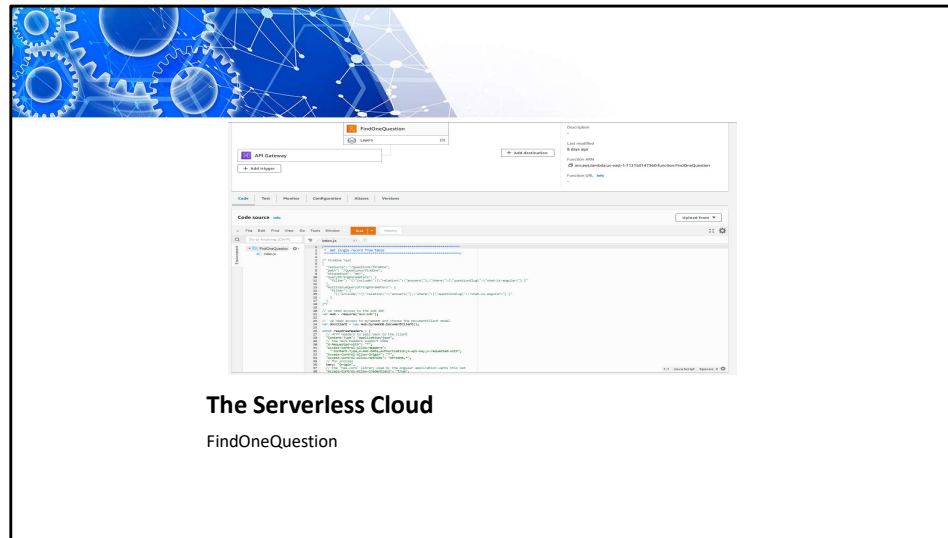
This is the Lambda function tablescan and the corresponding test to make sure it is functional. This function is used to populate the list of questions in the front-end angular portion of the application. It is important to note when testing, these functions will manipulate the data in the tables in DynamoDB.

e



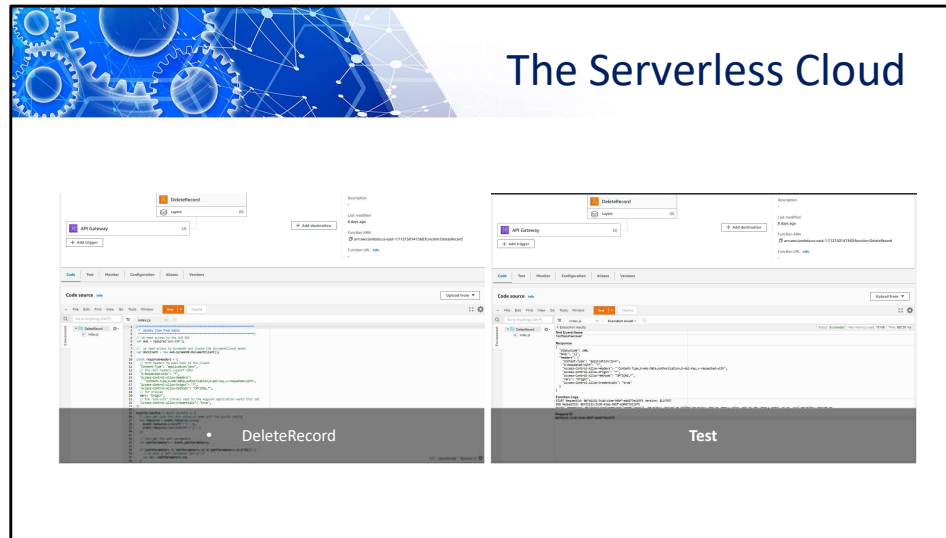
This is the Lambda function GetSingleRecord and the corresponding test to make sure it is functional. This is used once the user clicks one the questions it will open a new page and populate the single questions and answers. If you look at the top of each picture you will see what services, use the lambda functions. In this case it is the service, API Gateway. This is a good way to keep track of what is being used and how.

e



This is the Lambda function FindOneQuestion. In this case there is no test. This is used by the angular application for fetching single record.

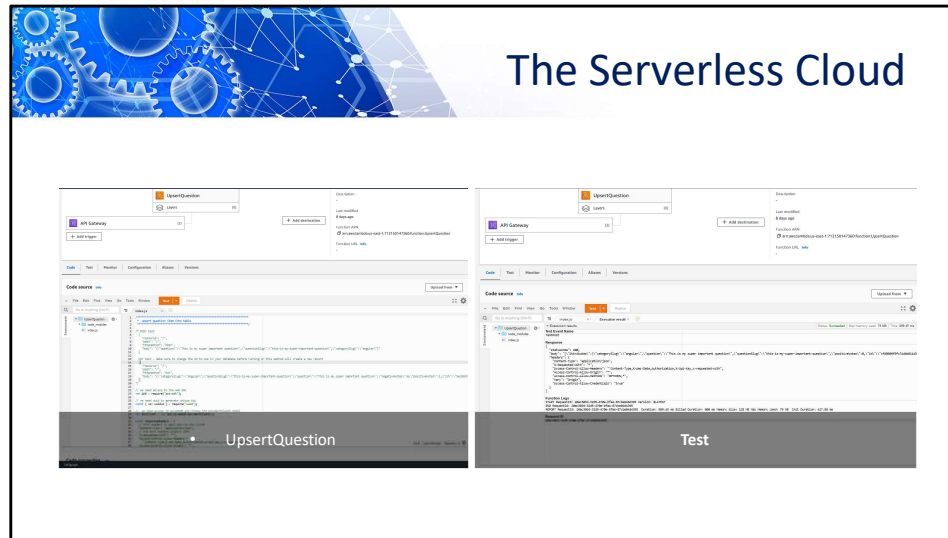
e



This is the Lambda function DeleteRecord and the corresponding test to make sure it is functional. This deletes the question and answer or just the answer when the delete button is pushed. This is the DELETE function of our RESTFUL API.

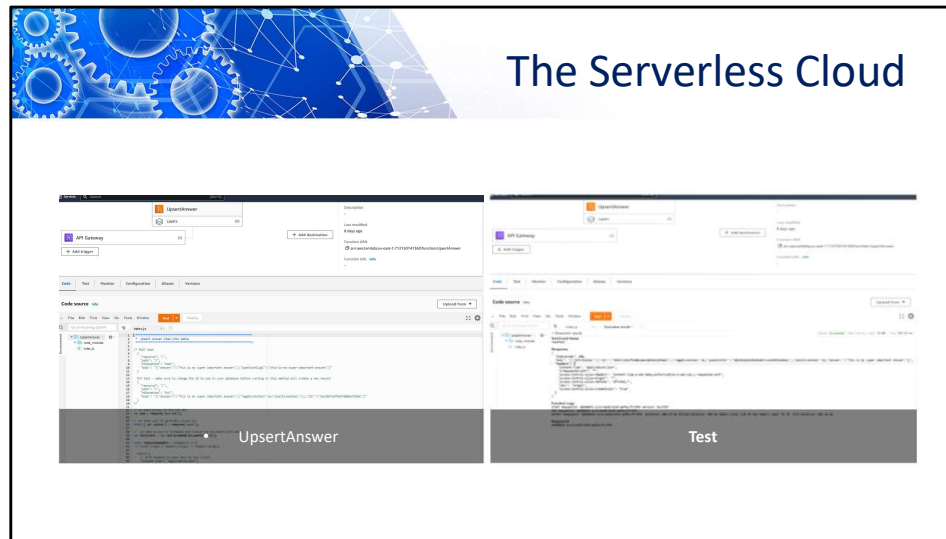
The next two slide show the functions used for both POST and PUT of the RESTFUL API.

e



This is the Lambda function UpsertQuestion and the corresponding test to make sure it is functional. This is used to post new questions or to update an existing question.

e



This is the Lambda function UpsertAnswer and the corresponding test to make sure it is functional. This is used to post new answer or to update an existing answer. It also connects the two by automatically attaching the question id the answer.

# The Serverless Cloud

## Frontend and backend connection



```
src > environments > ts environment.prod.ts > ...  
1 export const environment = {  
2   production: true,  
3   api_url: 'https://11967hgg1.execute-api.us-east-1.amazonaws.com'  
4 },  
5
```

Here we see the connection that connects the frontend to the backend. This is done by adding Backend web address to the frontend angular code.

e






MongoDB and DynamoDB are very similar in functionality but there are a few factors that make them stand apart. One key thing to note about the two is availability. While MongoDB is available for most services including AWS, DynamoDB is locked with AWS. This may cause user issues if they decide to migrate away from AWS in the future. The nice thing with DynamoDB is it works seamlessly with the other services for AWS and requires less work to maintain but does not provide the flexibility as MongoDB does. MongoDB is a great software that has great flexibility but with that flexibility comes more complexity that requires more user interaction to make sure everything is running smoothly. An example DynamoDB automatically backups data over of its massive network but MongoDB requires users to back up their data themselves. While we can sit here all day pointing back and forth on which is better, it is better to say each has their own uses and which to use really depends on the needs of the user. DynamoDB and its single table is very easy to work with initially while working with MongoDB can get very complicated with its relational databases and the complexity seems lessened a bit with programs such as MongoDB compass. The structure of single tables is very easy to read in the ASW site and is designed to keep the costs down. As it scales upward, the need for standardization and really good documentation will be paramount to maintain the organizational structure of the database. For our Database in DynamoDB we had two tables,

questions and answers and the were stored by a unique id. Each question and answer is paired up by their id. That way in the front end when a question is clicked, the answers related to the question will show up.

#### Resources:

14, J., & Wickramasinghe, S. (2021, July 14). *MongoDB VS dynamodb: Comparing nosql databases*. BMC Blogs. Retrieved February 2, 2023, from <https://www.bmc.com/blogs/mongodb-vs-dynamodb/>


*Amazon AWS DynamoDB - Single Table Design*. GetInData. (n.d.). Retrieved February 2, 2023, from <https://getindata.com/blog/amazon-aws-dynamodb-single-table-design/>



Items returned (7)							
<input type="checkbox"/>	id	answers	categorySlug	negativeVotes	positiveVotes	question	questionSlug
<input type="checkbox"/>	d026d2c362c549b6b...		angular	0	0	This is my s...	this-is-my-super-important-question
<input type="checkbox"/>	df9ba0f0505450595...	[ ]	angular	0	0	What is An...	what-is-angular2
<input type="checkbox"/>	ed9b8f0c9b9f6a9b9...	[ ]	angular	0	0	What is An...	what-is-angular4
<input type="checkbox"/>	Df5a0e5f0a0a0a0a0...	[ ]	angular	0	0	What is An...	what-is-angular3
<input type="checkbox"/>	f54h0f0f0f0f0f0f0...	[ ]	angular	0	0	What is An...	what-is-angular3
<input type="checkbox"/>	0uf654940cc494a0a...		angular	0	0	one last test	one-last-test
<input type="checkbox"/>	c009099f0c740a087...		angular	0	0	This is my s...	this-is-my-super-important-question

**The Serverless Cloud**  
Database - Questions table

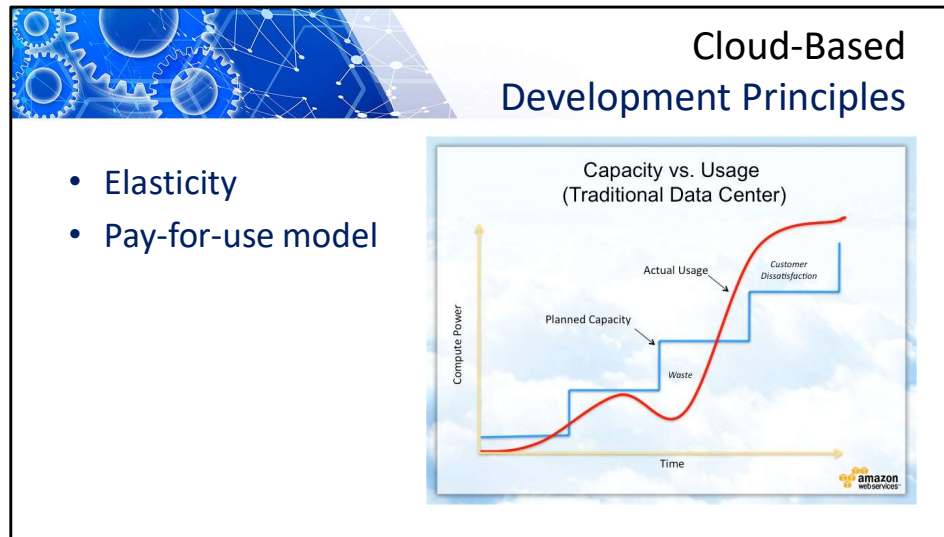
In this slide we see the table that holds our questions. You will notice that each entry has its own unique id.



<input type="checkbox"/>	id	answer	negativeVotes	positiveVotes	questionid
<input type="checkbox"/>	4388f08a6d0540268...	This is my s...	0	0	d02d62c262d549be87ccc6dd52bd94e
<input type="checkbox"/>	dffjbdf.DSD54SD5Gd...	Because it I...	0	0	dffjbdf.DSD54SD5G
<input type="checkbox"/>	d.Pfpgbwdffkgnf2fad...	Because it I...	0	0	F54hd56h4f654hd6F54gh6s
<input type="checkbox"/>	djhghjshDFjdbfwhD5J...	Because it I...	0	0	djhghjshghfksdjhghks44gh65sd
<input type="checkbox"/>	a45724c204394b478...	this is workl...	0	0	0af665c940ec494da01c3bc675fb768d
<input type="checkbox"/>	kjfdhgkajfdghkajfdhgj...	Because it I...	0	0	DFGADfGfADGAGAS4FDG5AFD4G6

**The Serverless Cloud**  
Database – Answers Table

In this slide we see the table for our answers for the questions. You will notice like the questions each answer also has a unique id. In the questionid column you will notice that the corresponding question id is here. This is how the two tables are connected. This allows for the API Gateway to pull the data and their connections using the lambda functions.




As I have mentioned in previous slides, AWS services that are used in this application are very scalable. This allows for flexibility for developers, at the same time it saves on cost. Maintaining these types of services inhouse if not planned right can be costly. One thing that can be overlooked is when maintaining inhouse servers is you are paying for service during both busy and quiet times. Using serverless, you are only paying for what you use. Also, so you do not get a surprise bill you can not only set up alerts and throttles, but you can monitor every aspect of the services that you are using including the costs.




## Securing Your Cloud App

Access	Policies
<ul style="list-style-type: none"><li>• Using S3 console</li><li>• IAM role</li></ul>	<ul style="list-style-type: none"><li>• The relationship between roles and policies.</li><li>• What policy was used?</li></ul>

When a user first creates a S3 bucket it only allows the admin access. this is intentional so admin can use the service right away without worrying if their data is safe. By going to the bucket permissions, the admin can open access to the public. They can also set permissions by creating roles and allow access while also limiting what each role has access to. Within our application we used the IAM role LabRole which gave us permission to work within the different services that AWS has. We needed to assign the roles our Lambda functions which allowed for them to be used in our back-end. The entire point of role and policies are to limit access to unauthorized people. Best practice is to give only access to what is needed no more. To make sure the IAM roles and policies are set up correctly is to use the dashboards that is provide to see what permission were given out and to make sure they are working as intended. As always test each role to make sure the security is working and not giving too many privileges.



## Securing Your Cloud App



This Photo by Unknown Author is licensed under [CC BY-SA-NC](#)

### API Security and API Gateway

- Answers
  - GET
  - OPTIONS
  - POST
  - PUT
  - DELETE
  - GET
  - OPTIONS
  - PUT
- Questions
  - GET
  - OPTIONS
  - POST
  - PUT
- FindOne
  - GET
  - OPTIONS
  - POST
  - PUT
- Find
  - GET
  - OPTIONS
  - POST
  - PUT

Amazon API Gateway is a service designed to allow the user complete control on how API's work for their application. It is designed to accept a request and return a response. It acts like the bridge that connect the front end with the backend. Gateway is also a perfect tool to help maintain security. It does this by routing traffic through the correct channels. This can be as simple as validating inputs to preventing DDoS attacks. Essentially it is like a guard dog for your application, only the correct calls with go through. Gateway is priced by calls and data transfers. This allows for that application to scale up while at the same time being efficient in deploying updates to the application. As with other services the connection between services are protected by IAM roles. It is important to remove old IAM users, roles and permissions. It is also important to review current permissions and make any changes. What was secure yesterday does not mean it is secure today.

## Conclusion

**Thank you for your time.**



- Ease of use
- Scalability and low costs
- Security

I leave you with these thoughts. Working in the cloud is very easy to do. It is very scalable while at the same time reduces cost compared to other technologies. Moving to the cloud does not mean that security is sacrificed but in truth it is most likely more secure. Thank you for your time and have a good day.