Project Two Conference Presentation: Cloud Development

CS 470 Full Stack Development II

Sergio Mateos

Southern New Hampshire University

- Link: https://www.youtube.com/watch?v=kvVdD4hFS6s
- Purpose of Presentation
    - Hello everyone, my name is Sergio Mateos and I'm a Computer Science student at Southern New Hampshire University. For the last couple of weeks, I have been working for a startup company that decided to migrate an application that runs on the local host to run completely into the cloud with Amazon Web Services. In this presentation, I will cover Containerization, Orchestration, Serverless Cloud, Cloud-Based Development Principles, and Security for the Could Application. So, let's get started.
- Containerization
    - First, to migrate our application from the server to the cloud, Containerization is needed. Containerization is packing software code with just the operating system libraries and dependencies that are required to run the code to create a single lightweight executable container. In other words, containerization means packing the application into an image that will contain all the code and necessary components to run on any machine. The data would be stored in containers. A container is an executable package that will contain the application with all its necessary components to run and be tested. In the project, I use a containerization software called Docker, in which I create containers for the angular-driven front-end and host a node.js backend rest API.

- Orchestration
    - Orchestration is the automated configuration, coordination, and management of computer systems, applications, and software. Basically, orchestration helps us to simplify operations management. As I mentioned earlier, I used Docker as my containerization tool. Docker helps you to run multiple environments in one host. Docker helps you save money and resources, but not only that it also makes it easy to manage applications. Finally, Docker makes environment portability and testing much easier during the development process. Keeping this in mind I was able to use a single YAMI formatted file to configure all my application services.
- The Serverless Cloud
    - First, let me explain what is serverless. Serverless is a cloud computing application development and executed model that enables developers to build and run application code without a managing server in other words the application your application will run completely in the cloud without having a dedicated server and the cloud server will be managed by the cloud provider. In our case is Amazon Web Services or AWS. A Serverless cloud has a lot of advantages like allowing teams to focus on development and be more productive since the responsibility of managing the servers is being taken away. The cost of the serverless cloud is lower compared to managing their own servers, and the team can build better applications instead of configuring them. Now let's talk about S3 storage. Amazon S3 uses the term "bucket" S3 which is where developers can store their files in the cloud. Once the data is assigned to a bucket developer can set up access controls, modify, etc. One of the advantages of S3 is that provides

automatic backup to ensure that the data is safe. Compared to local storage, the security is higher, and the customer pays-as-you-go, compare local storage, which you must only pay for maintenance, but usually the maintenance service tends to be high cost. And finally, the S3 bucket has high scalability.

- Let me introduce you to Lambda, Lambda is a serverless computing platform provided by Amazon Web Services. It's a computing service that runs code in response to events and manages the computing resource required by that code. The advantages of using a serverless application programming interference or API are running code without provisioning, it only charges when is called, which helps us minimize the cost. Collaborating with other developers is much easier because the API is available for everyone. The manageability of applications is easier since all the applications are stored in one area. In our project, I developed a couple of Lambda computer functions that can retrieve, modify, insert, and delete records of information. The process to integrate the front end with the backed first the code front end code is hosted in the AWS S3 bucket which makes a REST API call to the endpoint utilizing Amazon API Gateway. Once we are in this section, Lambada runs the code from the predeterminate scripts within the relevant compute functions to obtain information from the database and retrieve it back to the user. Amazon provides a NoSQL database called DynamoDB that supports key-value and document data structure. I use DynamoDB to store data for my project.
- During this project, I worked with two different databases MongoDB and DynamoDB, but first, let me define what a database is. A database is an organized

collection of structured information, or data, stored electronically in a computer system. DynamoDB it's exclusive for AWS compared to MongoDB which is more versatile with different platforms. Both are NoSQL databases, in other words, they both don't use traditional tables or structure data concepts from relational databases. Even if they seems similar, there are key differences that make them unique the allowed data types of documents, DynamoDB has a limited key-value store with JSON support equivalent to 400 kb compared to MongoDB which accepts up to 16MB with no JSON limitation. DynamoDB only accepts data in number, string, or binary format, but DynamoDB is simpler to use and understand to work with. In our project queries were performed to test our database. The Question-and-Answer Table serve queries including the addition or removal of questions and answers from the database.

- Cloud-Based Development Principles
    - Amazon Web Services, as I commented earlier, has a model of pay-as-you-use, as you can see in this table the comparison between the capacity and the usage, the table project the computer power needed over time using the blue line, but the actual computer power is indicated with the red line, as we analyze the table help us to view that sometimes the computing powers is been wasted or it's too little. Traditional Data Centers' capacity for users is limited to what is present. Cloud-based serverless approach the capacity is elastic, this means as the number of user increase on the website or application, the capacity would always be as needed.
- Securing Your Cloud Presentation

- In my opinion one of the most important parts of the development of a website or application is security. Some questions that every developer should ask are How can I prevent unauthorized access? Amazon Web Services provides multitudes of tools to easily implement relevant security features. Tools like Identity Access Management or IAM include the capability for setting up rules and policies. This provides specific role permissions to specific individuals to have access to the application or website. To build this role you need to set up a policy, these policies assigned each role what they have permission to do or access. One example in our project was when creating a custom policy to allow Lambda to access the DynamoDB table for the functions put, delete, get, scan, quarry, and update item functions. How can we secure the connection between Lambda and the API Gateway? For Lambda there are unique key sources to identify the use of connect and perform API calls. Lambda also encrypts data at rest and in transit which will include data in DynamoDB. Also, the policies for accessing the DynamoDB database are only assigned to certain Lambda functions. The security of our bucket is important, that's why AWS automatically makes it private so only people with an IAM role can access it.

- Conclusion
  - In conclusion, cloud-based development is a better option since lets the team increase their productivity, minimize the expenses, and in this case, AWS automatically scales and manages itself. Also, developing a full stack application has provided me with a lot knowledge of how to migrate an application to a

serverless environment. I wish all of you the best and thank you for listening to my presentation.