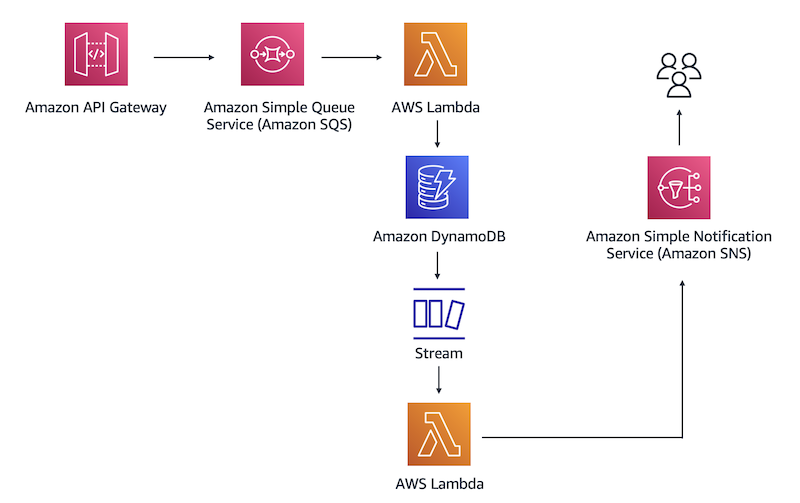
**Architecting Solutions: Building a Proof of Concept for a Serverless Solution**

Build a serverless architecture for a customer who sells cleaning supplies and often sees spikes in demand for their website, and they need an architecture that can easily scale in and out as demand changes. The customer also wants to ensure that the application has decoupled application components.

The following architectural diagram shows the flow for the serverless solution that you will build.



In the above architecture I have used a REST API to place a database entry in the Amazon SQS queue.

Amazon SQS will then invoke the first Lambda function, which inserts the entry into a DynamoDB table.

After that, DynamoDB Streams will capture a record of the new entry in a database and invoke a second Lambda function.

The function will pass the database entry to Amazon SNS. After Amazon SNS processes the new record, it will send a notification through a specified email address.

In this architecture, I have learned how to do the following:

* Create IAM policies and roles to follow best practices of working in the AWS Cloud.
* Create a DynamoDB table to store data.
* Create an Amazon SQS queue to receive, store, and send messages between software components.
* Create Lambda functions and set up triggers to invoke actions in different AWS services.
* Enable DynamoDB Streams to capture modifications in the database table.
* Configure Amazon SNS to receive email or text notifications.
* Create a REST API to insert data into a database.

To complete this exercise, I have used **US East (N. Virginia) us-east-1** and IAM user to follow best practices.

1. **Created a IAM user and define Polices for DynamoDB, SNS, SQS, and DynamoDBStreams to perform their specified actions.**

A screenshot of a computer

AI-generated content may be incorrect.

1. Created IAM roles and attached polices to that Roles

A screenshot of a computer

AI-generated content may be incorrect.

1. **Created a DynamoDB Table**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Created SQS Queue**

SQS will receive data records from API Gateway, stores them, and then sends them to a database.

While creating a SQS, provided ARN for the *APIGateway-SQS* IAM role in “who can send messages to the queue”, and provided ARN for the *Lambda-SQS-DynamoDB* IAM role in “who can receive messages from the queue”

By Configuring this, SQS will receive a record from the REST API and send it to the first Lambda function to store the records in database.

1. **Created a Lambda function and set up triggers**

Created a Lambda function that reads messages from the SQS queue and writes an order record to the DynamoDB table.

Created a Lambda function for the previously created Lambda-SQS-DynamoDB role.

A screenshot of a computer

AI-generated content may be incorrect.

Once the function created, added Trigger for the SQS queue.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Added and deployed the code: The code passes arguments to a function call. As a result, when a trigger invokes the function, Lambda runs the code which was specified.

A screenshot of a computer

AI-generated content may be incorrect.

Created an Event and Test.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Test was successful and Lambda function has sent a test message “Hello from SQS” from SQS templates to DynamoDB table.

A screenshot of a computer

AI-generated content may be incorrect.

Verified that the Lambda function adds the test message to a database

A screenshot of a computer

AI-generated content may be incorrect.

1. **Enabled DynamoDB Strem**

DynamoDB stream will capture information about every modification to the data items in the Table.

After the Lambda function reads messages from the SQS queue and writes an order record to the DynamoDB table, DynamoDB Streams captures the primary key attributes from the record.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Created a SNS topic and setup Subscriptions.**

SNS will deliver the messages to the endpoints (Subscribers or Clients)

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Created second AWS Lambda function to publish a message to the SNS Topic.**

Created second Lambda function to get records from DynamoDB Streams and send them to Amazon SNS.

A screenshot of a computer

AI-generated content may be incorrect.

Setup DynamoDB as a trigger to invoke a Lambda function.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Deployed the Code

A screenshot of a computer

AI-generated content may be incorrect.

Testing the second Lambda function

A screenshot of a computer

AI-generated content may be incorrect.

After the Lambda function runs successfully, email message was successfully delivered to the specified email address provided while creating SNS Topic.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Created API Gateway**

The API serves as a communication gateway between the application and the AWS services.

Created a REST API and configured it with SQS ARN for the communication between API Gateway and SQS.

Also, configured HTTP header and Mapping Template so that API Gateway can receive the messages from application and sends it to the SQS.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Testing the architecture by using API Gateway.**

Used use API Gateway to send mock data to Amazon SQS as a proof of concept for the serverless solution.

in the API Gateway console – POST Method Execution – Test – in the Request Body, entered the new item entry.

API Gateway successfully processed the request that was entered in the **Request Body** box – received “Successfully completed execution” message.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

New entry was added into the DynamoDB Table.

A screenshot of a computer

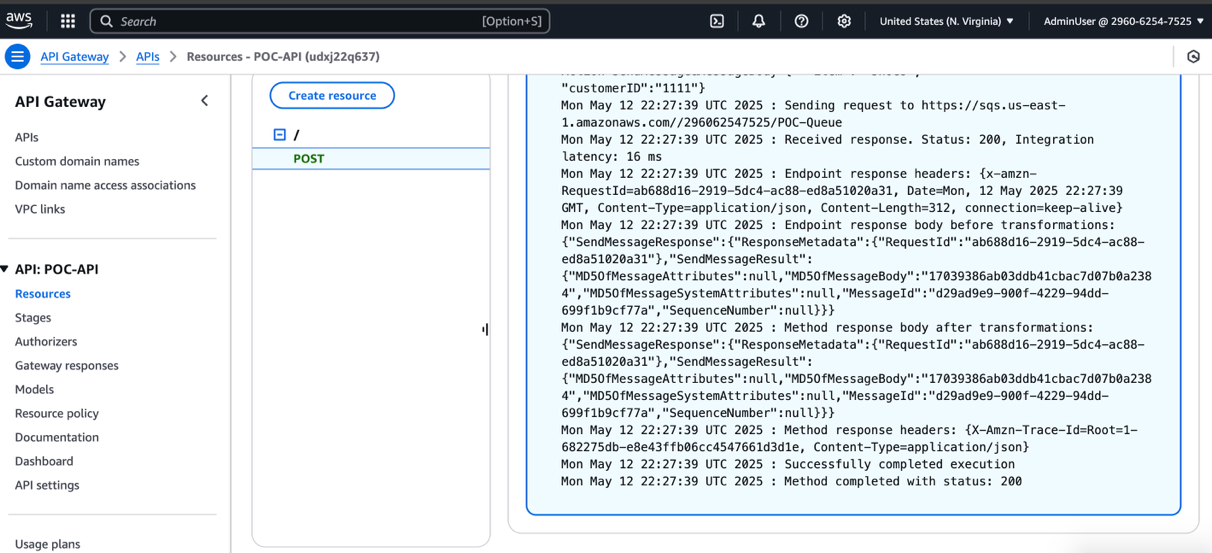
AI-generated content may be incorrect.

**Added second entry -**

A screenshot of a computer screen

AI-generated content may be incorrect.

API Gateway successfully processed the second request that was entered in the **Request Body** box – received “Successfully completed execution” message.



Second entry was added to the DynamoDB table.

A screenshot of a computer

AI-generated content may be incorrect.

Email notification received to the subscribed email address notifying that the new item was added to the inventory/database.

A screenshot of a computer screen

AI-generated content may be incorrect.

**Explanation of how the architecture functions:**

After API Gateway successfully processed the request that was entered in the **Request Body** box, it placed the request in the SQS queue. Because Amazon SQS was set up as a trigger in the first Lambda function, Amazon SQS invokes the function call. The Lambda function code placed the new entry into the DynamoDB table. DynamoDB Streams captured this change to the database and invoked the second AWS Lambda function. This function received the new record from DynamoDB Streams and sent it to Amazon SNS. Amazon SNS, in turn, sent an email notification to the subscribed email address.