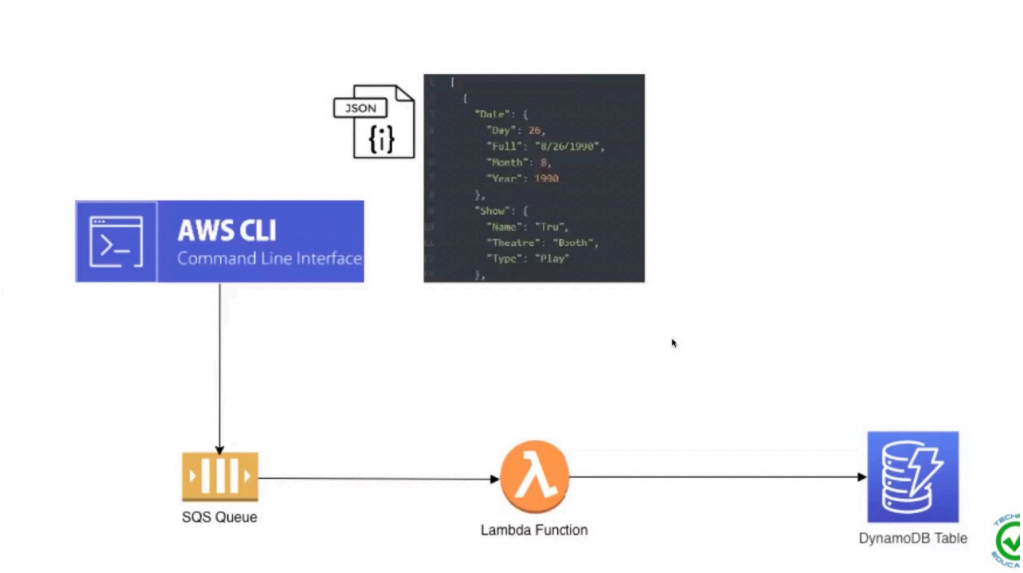
Event Driven App with SQS-Lambda-DynamoDB



SQS is queue publisher based, pushed based working system. SQS lets you send, store, and receive messages between software components at any volume, without losing messages or requiring other services to be available. SQS enables web service applications to quickly and reliably queue messages that one component in the applications generates for another component to consume.

This app basically will work CLI based. CLI is important in order to manage the system. We are gonna create DynamoDB,Lambda Function. We're gonna create table on DynamoDB. We'll have lambda function. We're gonna download node js app by using Lamda Function. This node js app will let to enter data to DynamoDB table. This data will send from SQS Queue. We're gonna use CLI to use json file. Json file consists messages which will send to SQSQ. As soon as the message delivers by AWS CLI, SQSQ will send this message to Lambda function. Lambda function consist node js codes. This code will write to DynamoDB directly.

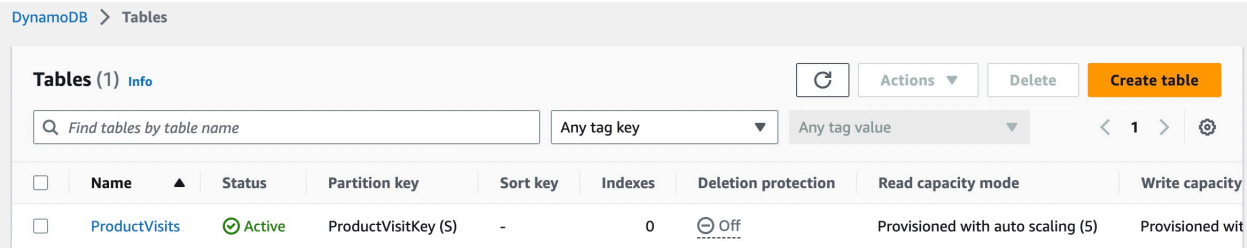
Step 1

Region has to be us-east-1(N.Virginia)

Step 2

Create DynamoDB Table

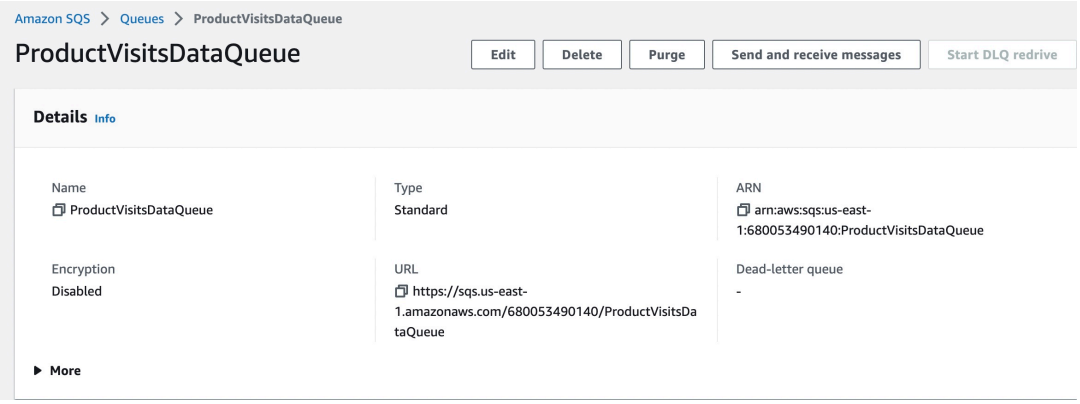
Name: ProductVisits Partition key: ProductVisitKey



Step 3

Create SQS Queue

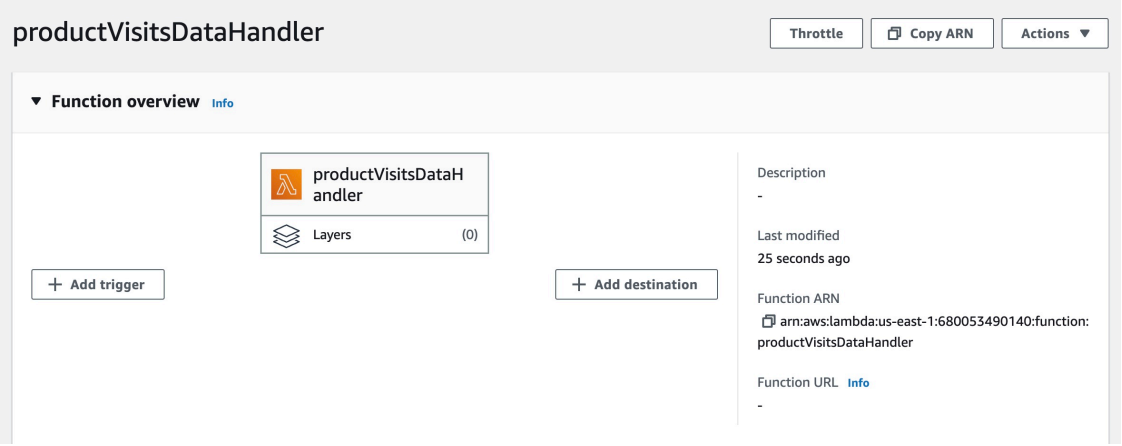
Name: ProductVisitsDataQueue Type: Standard



Step 4

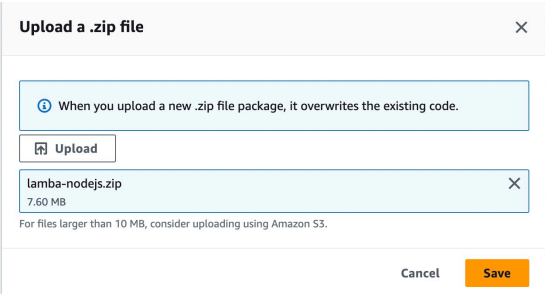
Create Lambda Function

Name: productVisitsDataHandler
Runtime: Node.js 14x
Role: Create new role from AWS policy templates
Role name: lambdaRoleForSQSPermissions
Add policy templates: “Simple micro service permissions’ and “Amazon SQS poller permissions”



Step 5

Upload Lambda Function Code



I uploaded lambda-nodes.zip file

Step 6

Test Messaging

Under VS Code, paste the code given below

AWS CLI Command: aws sqs send-message --queue-url https://sqs.us-east-1.amazonaws.com/938251564890/ProductVisitsDataQueue --message-body file://message-body-1.json

I modified the queue name and file name.

```
Umits-MacBook:sqs-lambda-app muhammedumit$ aws sqs send-message --queue-url https://sqs.us-east-1.amazonaws.com/680053490140/ProductVisitsDataQueue --message-body file://mess
age-body-1.json
{
    "MD5OfMessageBody": "658e1072653211e7bae2758b09c37157",
    "MessageId": "ed4c5a6a-8526-439f-9e69-163c414a49a8"
}
Umits-MacBook:sqs-lambda-app muhammedumit$
main*  ⊗ 0 ⚠ 0
```

When you go back to SQS and open “ProductVisitDataQueue”, you can go to find and receive messages and look at the message body.

Messages (1)

View details | Delete

Search messages

1

ID	Sent	Size	Receive count
ed4c5a6a-8526-439f-9e69-163c414a49a8	Jul 02, 2023, 14:12:58 EDT	274 bytes	1

Step 7

Configure SQS

Under the lambda function trigger on ProductVisitsDataQueue click on the configure lambda function trigger and select productVisitsDataHandler.

This time you won’t be able to see any messages this time because DynamoDB get the message.

Messages (0)

View details | Delete

Search messages

1

ID	Sent	Size	Receive count

No messages. To view messages in the queue, poll for messages.

Poll for messages

Items returned (1)

Actions | Create item

1

ProductVisitKey	Category	CustomerId	CustomerNa...	PricePerUnit	Pr
38833463-500a-489c...	Aksesuar	be44af0a-7...	Cengiz Aytmatov	10	c9

Step 8

Use AWS CLI to send messages to SQS

I go to AWS CLI and sent all of the messages from 1to 5.

```
Umits-MacBook:sqs-lambda-app muhammedumit$ aws sqs send-message --queue-url https://
sqs.us-east-1.amazonaws.com/680053490140/ProductVisitsDataQueue --message-body file:
//message-body-1.json
{
    "MD5OfMessageBody": "658e1072653211e7bae2758b09c37157",
    "MessageId": "ed4c5a6a-8526-439f-9e69-163c414a49a8"
}
Umits-MacBook:sqs-lambda-app muhammedumit$ aws sqs send-message --queue-url https://
sqs.us-east-1.amazonaws.com/680053490140/ProductVisitsDataQueue --message-body file:
//message-body-2.json
{
    "MD5OfMessageBody": "74cb95bcf2bbf7c98c3fcde127c86b01",
    "MessageId": "eafc7b8b-2f17-4341-a1d8-ec07110b4687"
}
Umits-MacBook:sqs-lambda-app muhammedumit$ aws sqs send-message --queue-url https://
sqs.us-east-1.amazonaws.com/680053490140/ProductVisitsDataQueue --message-body file:
//message-body-3.json
{
    "MD5OfMessageBody": "96ed24b827ce3b884f86fe2c3c02427c",
    "MessageId": "988225d9-b96b-45c5-bd04-3aef8f47d05c"
}
Umits-MacBook:sqs-lambda-app muhammedumit$ aws sqs send-message --queue-url https://
sqs.us-east-1.amazonaws.com/680053490140/ProductVisitsDataQueue --message-body file:
//message-body-4.json
{
    "MD5OfMessageBody": "f62e8b9824296530a9c6c188f2d4727b",
    "MessageId": "bb8d6a25-2d96-4fdd-b3ac-38091016ab43"
}
Umits-MacBook:sqs-lambda-app muhammedumit$ aws sqs send-message --queue-url https://
sqs.us-east-1.amazonaws.com/680053490140/ProductVisitsDataQueue --message-body file:
//message-body-5.json
{
    "MD5OfMessageBody": "0f6c5c586dfdb6519914687dba783507",
    "MessageId": "cb239e95-c049-411c-b1c2-f5f76fc4eeec"
}
Umits-MacBook:sqs-lambda-app muhammedumit$
main*   0 0
```

And now you can see the logs are already taken by DynamoDB.

Items returned (5)

Actions | Create item

1

ProductVisitKey	Category	CustomerId	CustomerNa...	PricePerUnit	ProductId	ProductNa...
dfee3e3c-1682-4dfd-...	Bilgisayar	be44af0a-7...	Pamuk Tekir	29.99	c96b49bb-...	Fare
5af096ab-43b2-4e3d...	Mobil Telefon	b20f30ta-29...	Sibel Haris	1032	c96b49bb-...	iPhone
3fab9242-5f56-4cbf-...	Spor	be44af0a-7...	Nihat Hocaoğlu	160	c96b49bb-...	Dambıl
8256982a-e542-4498...	Giyim	wf9s200a-7...	Ali Kuşçu	189	c96b49bb-...	Ceket
38833463-500a-489c...	Aksesuar	be44af0a-7...	Cengiz Aytmatov	10	c96b49bb-...	Eldiven