**How to Build an Order / Message Queue System on AWS using SQS and Lambda functions**

A picture containing text, screenshot, person, cartoon

Description automatically generated

**Order Placement:** When a customer places an order on the KFC shop application, the application backend receives the order details, such as the customer's name, address, and the items they want to order.

**Queueing the Order:** The backend of the application uses AWS SQS to queue the order. SQS allows for reliable and scalable message queuing, ensuring that orders are processed in a decoupled manner.

**Processing the Order:** The backend worker or processing component, which could be running on AWS EC2 instances or AWS Lambda functions, continuously polls the SQS queue to retrieve new orders. Once an order is retrieved, the worker processes it, which involves tasks like preparing the food, updating the order status, and notifying the customer.

if KFC Shop 2 closes, KFC Shop 1 can continue to operate and process orders. The backend worker responsible for KFC Shop 1 will ignore orders intended for KFC Shop 2 and focus on processing its own orders.

1. Create a Message Queue system with Amazon SQS

Navigate to AWS SQS and create a SQS

A screenshot of a computer

Description automatically generated with medium confidence

I just keep everything as default, and I enabled the encryption and created a SQS.

A screenshot of a computer

Description automatically generated with medium confidence

1. Create an IAM role for Lambda function

A screenshot of a computer

Description automatically generated

Attach SQS Full Access and CloudWatchLogs Full Access permission to this role

Name it and create a role for lambda.

A screenshot of a computer

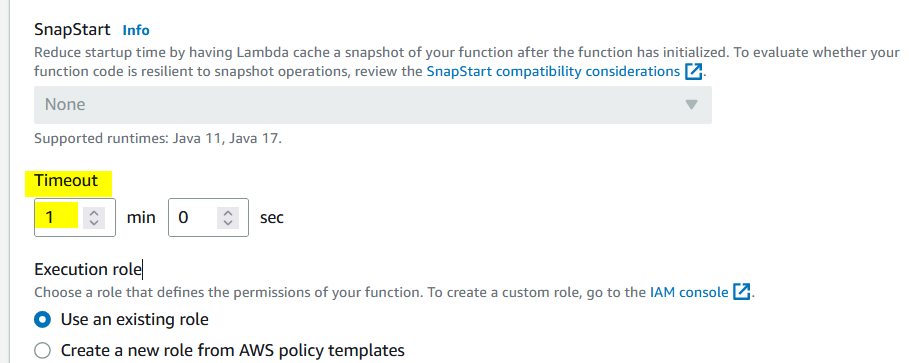
Description automatically generated with medium confidence

1. Create a Lambda function for backend application

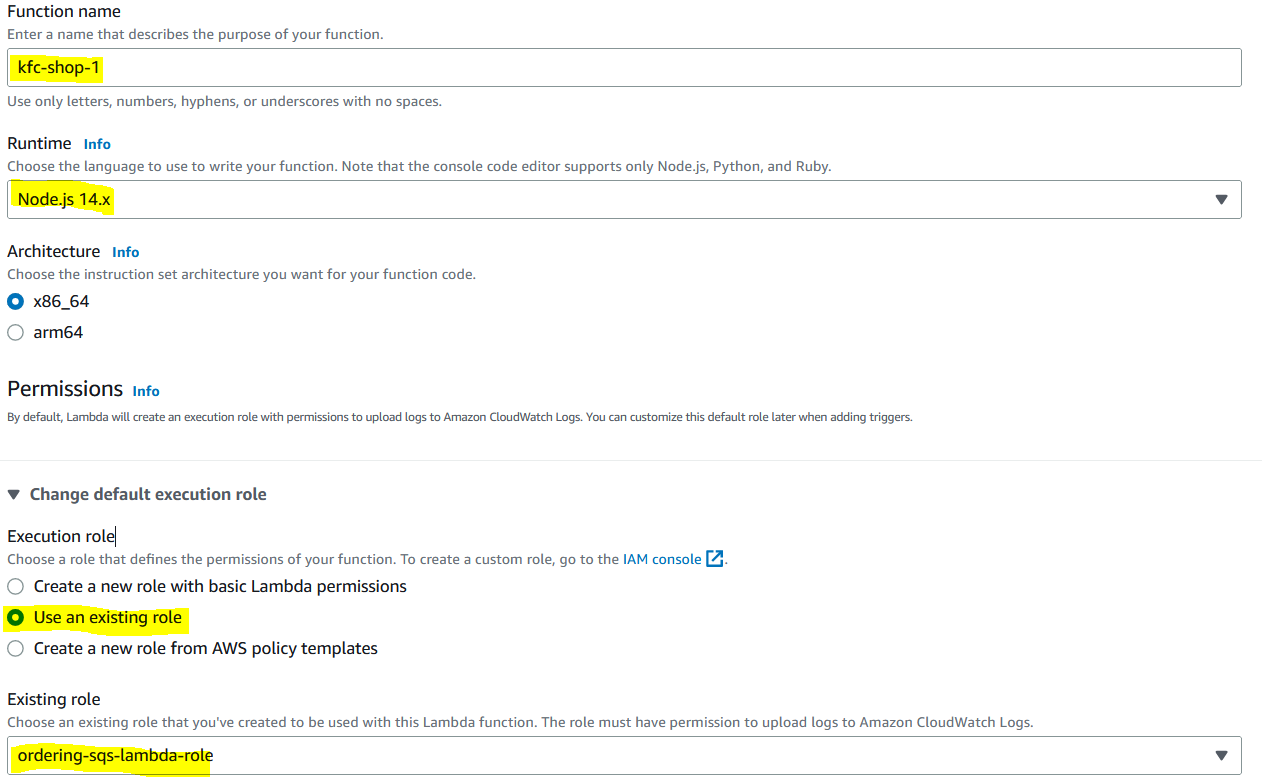
A screenshot of a computer

Description automatically generated with medium confidence

Attach an IAM role (which is created in step-2) and increase the timeout as 1 minute in Lambda timeout.



1. Create a two lambda for two kfc shops



Change the default timeout to 1 minute and concurrency – reserve concurrency – 1 – save changes.

A screenshot of a computer

Description automatically generated with medium confidence

Create one more lambda for kfc-shop-2 and do all the configuration as mentioned kfc-shop-1

Now we have 3 lambdas as below

A screenshot of a computer

Description automatically generated with medium confidence

1. Add SQS as trigger for both kfc-shop-1 & kfc-shop-2 lambda

A screenshot of a computer

Description automatically generated with medium confidence

And add trigger. Now its look like below

A screenshot of a computer

Description automatically generated with medium confidence

1. To update the code in application-backend-lambda

<https://github.com/kohlidevops/ordering-burgers-kfc-using-sqs-lambda/blob/main/application-backend-lambda.js>

1. To update the code in kfc-shop-1 lambda

<https://github.com/kohlidevops/ordering-burgers-kfc-using-sqs-lambda/blob/main/kfc-shop-1-lambda.js>

1. To update the code in kfc-shop-2 lambda

<https://github.com/kohlidevops/ordering-burgers-kfc-using-sqs-lambda/blob/main/kfc-shop-2-lambda.js>

1. To test the application-backend-lambda code

A screenshot of a computer

Description automatically generated with medium confidence

Save and test the code

A screenshot of a computer

Description automatically generated with low confidence

1. Queuing system process the order

A screenshot of a computer

Description automatically generated with medium confidence

1. To check the CloudWatch logs for kfc-shop-1 & 2

A screenshot of a computer

Description automatically generated with medium confidence

kfc-shop-1 logs

A screenshot of a computer

Description automatically generated with medium confidence

kfc-shop-2 logs

A screenshot of a computer

Description automatically generated with medium confidence