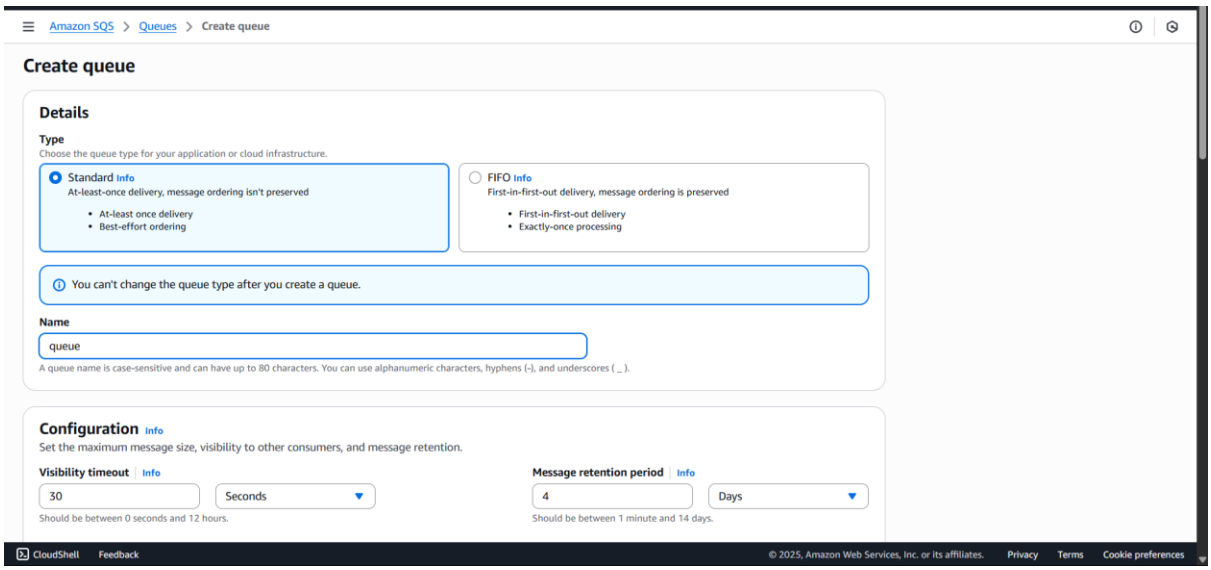
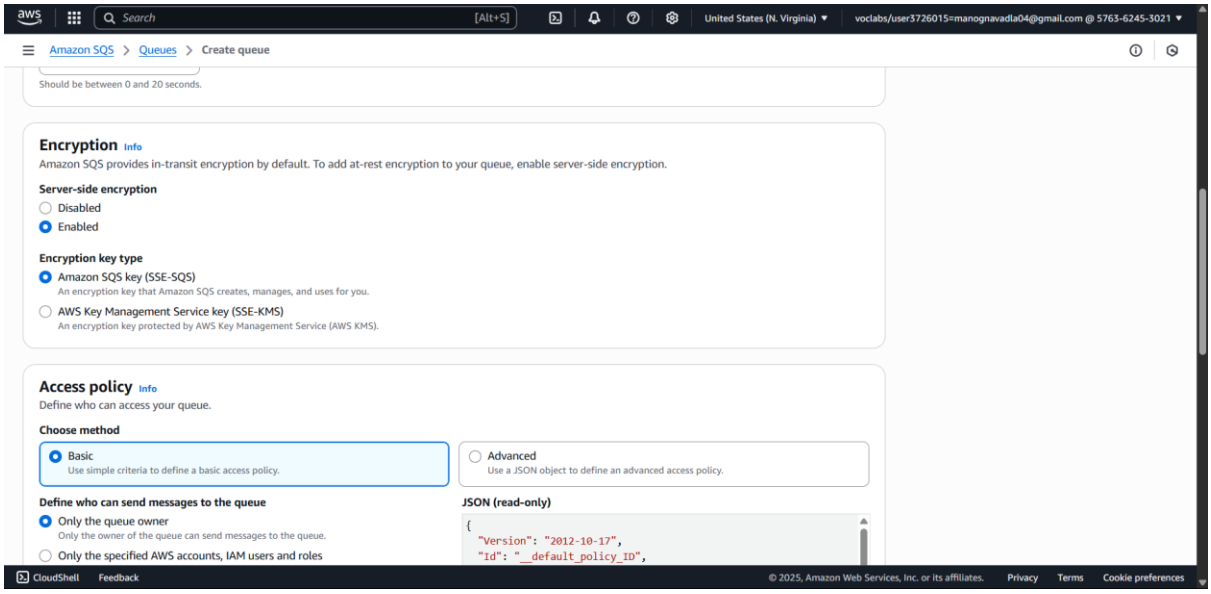
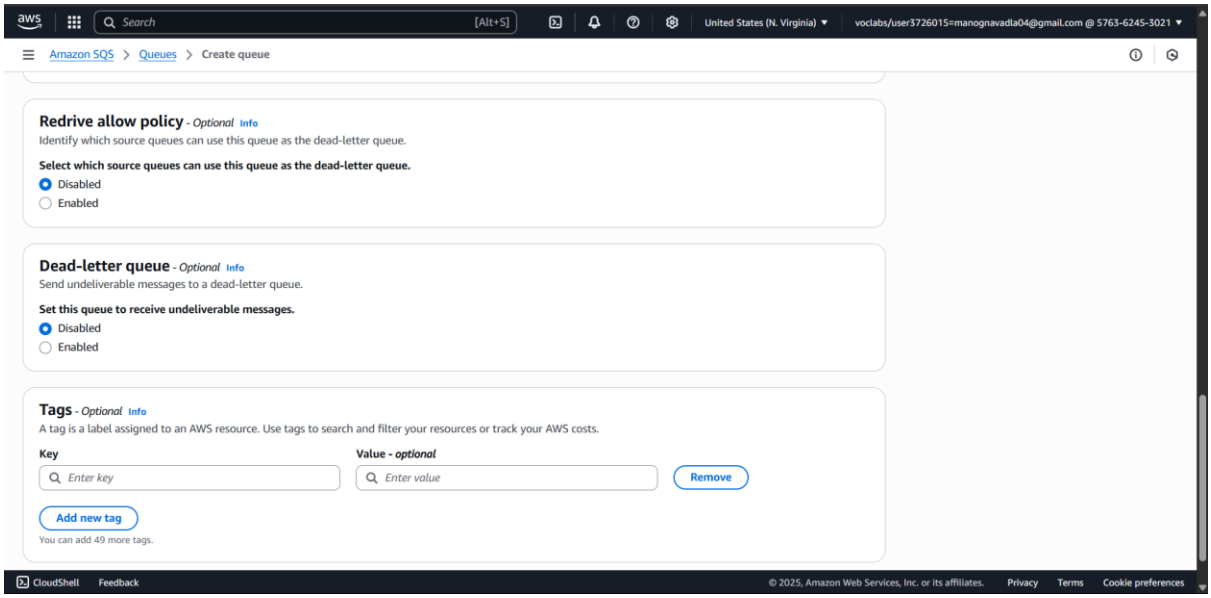
SIMPLE QUEUE SERVICE

Amazon SQS (Simple Queue Service) is a fully managed message queuing service that enables decoupling and reliable communication between distributed systems and microservices. It allows applications to send, store, and receive messages asynchronously, improving scalability, fault tolerance, and overall system performance. By managing the messaging infrastructure, SQS helps simplify integration and ensures that messages are not lost even if one part of the system fails.

Go to the Amazon SQS Console, click "Create queue", select "Standard" as the queue type, enter a name for your queue (e.g., MyStandardQueue), under Configuration leave the Visibility timeout as default (usually 30 seconds), Delivery delay as default (0 seconds), and Message retention period as default (4 days), leave all other settings as default, and finally click "Create queue”





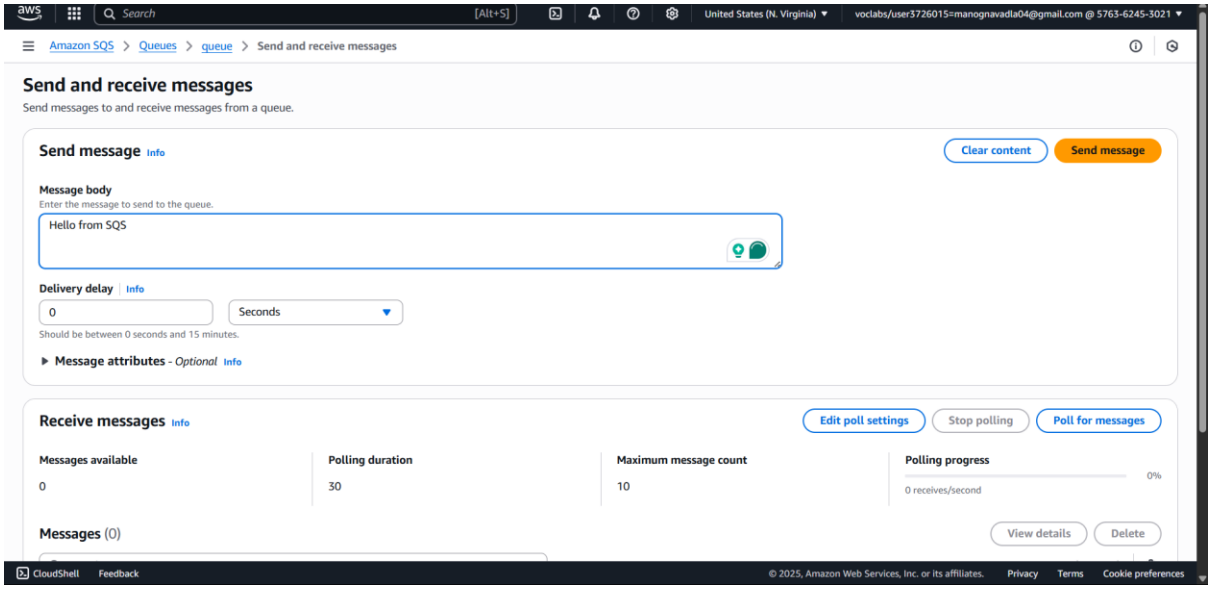


Go to the “Send and receive messages” tab.

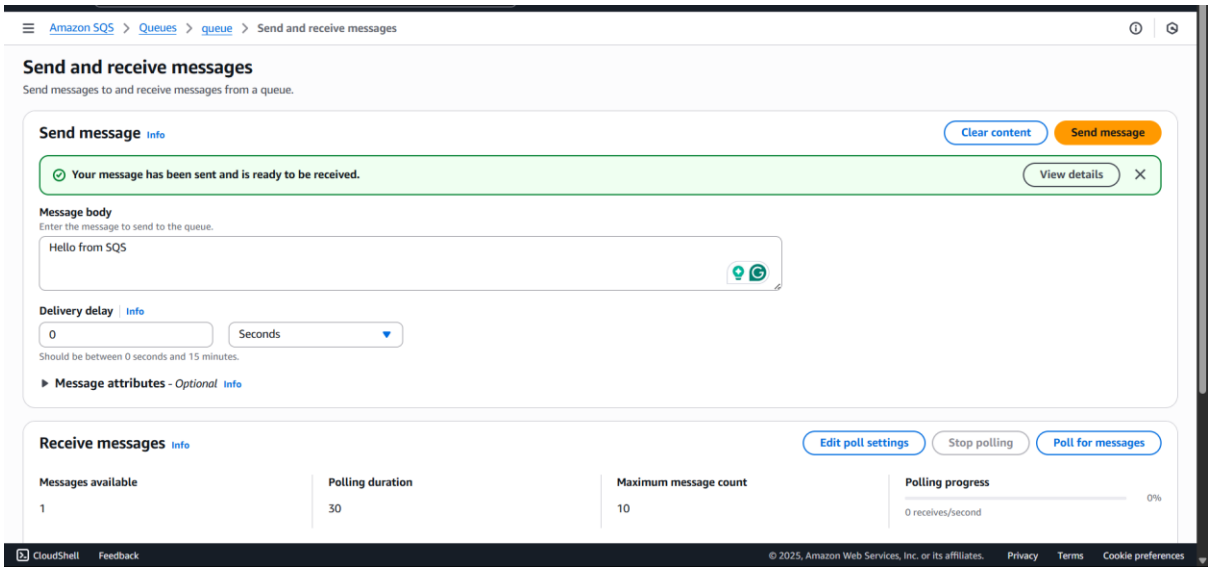
Under “Send a message”:

Clear the content box, type a message (e.g., "Hello from SQS").

Click “Send message”



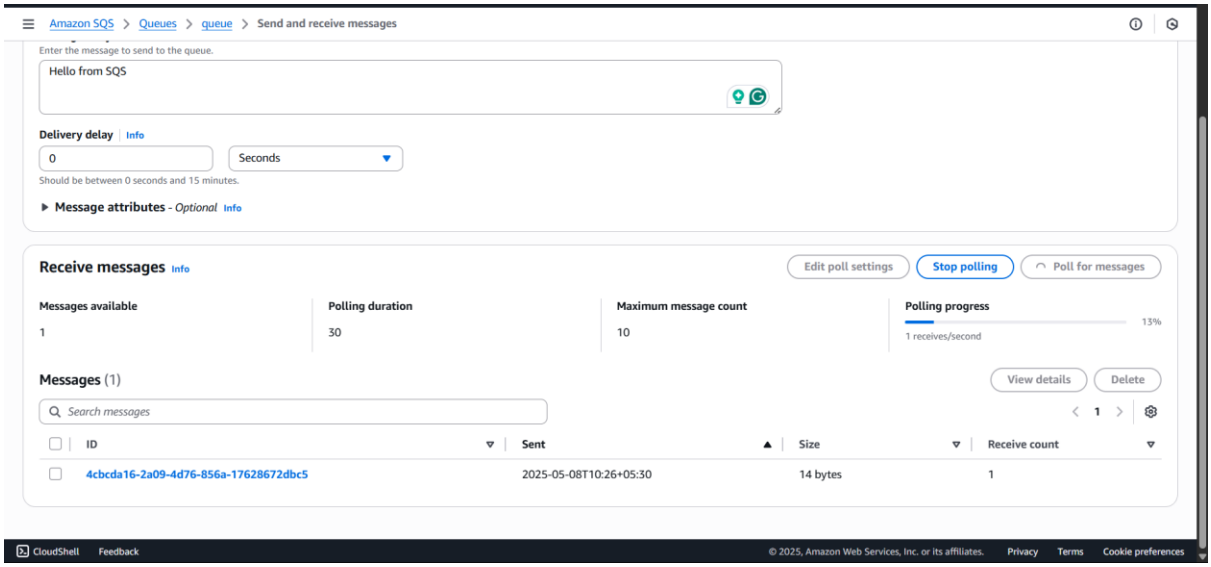
The message has been successfully sent



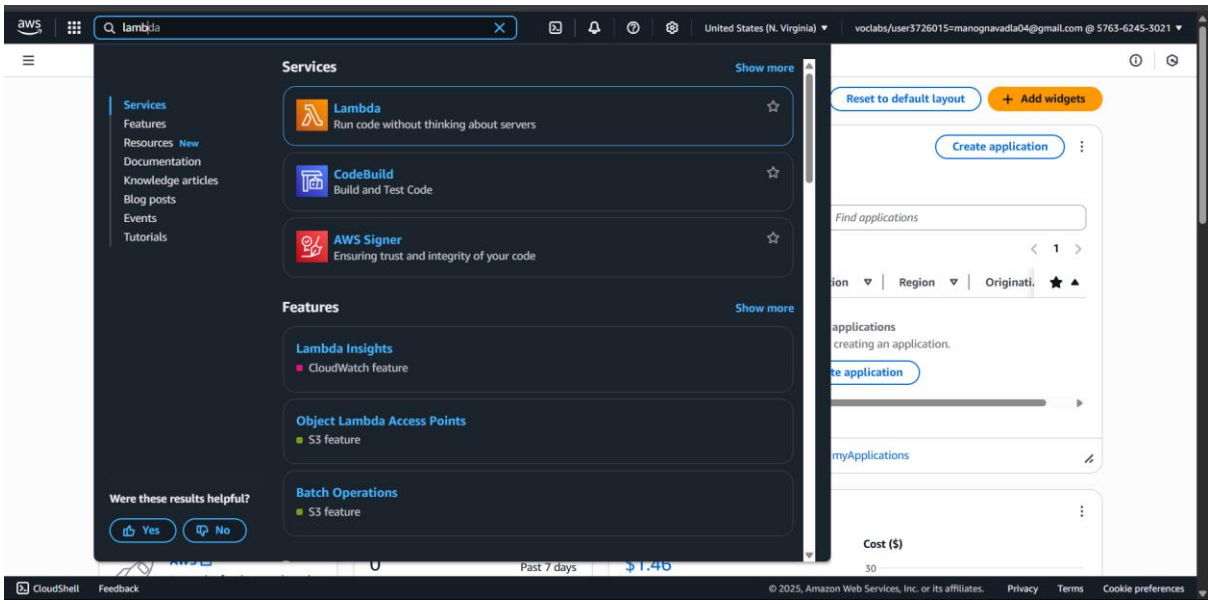
Scroll down to “Receive messages”:

Click “Poll for messages”.

You should see your message appear in the list.



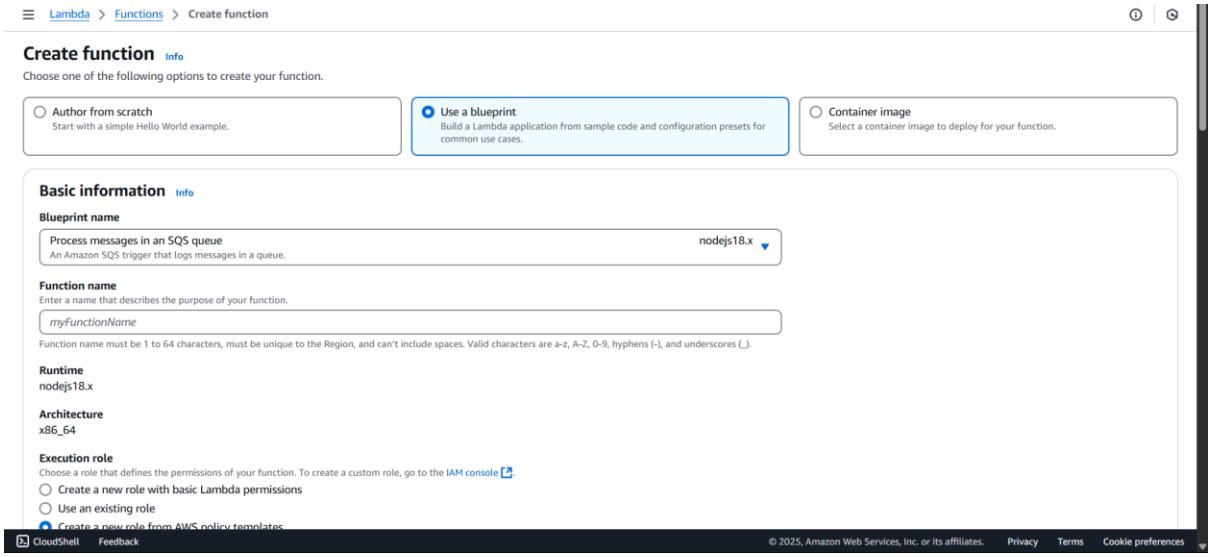
Search for Lambda service and click on create function

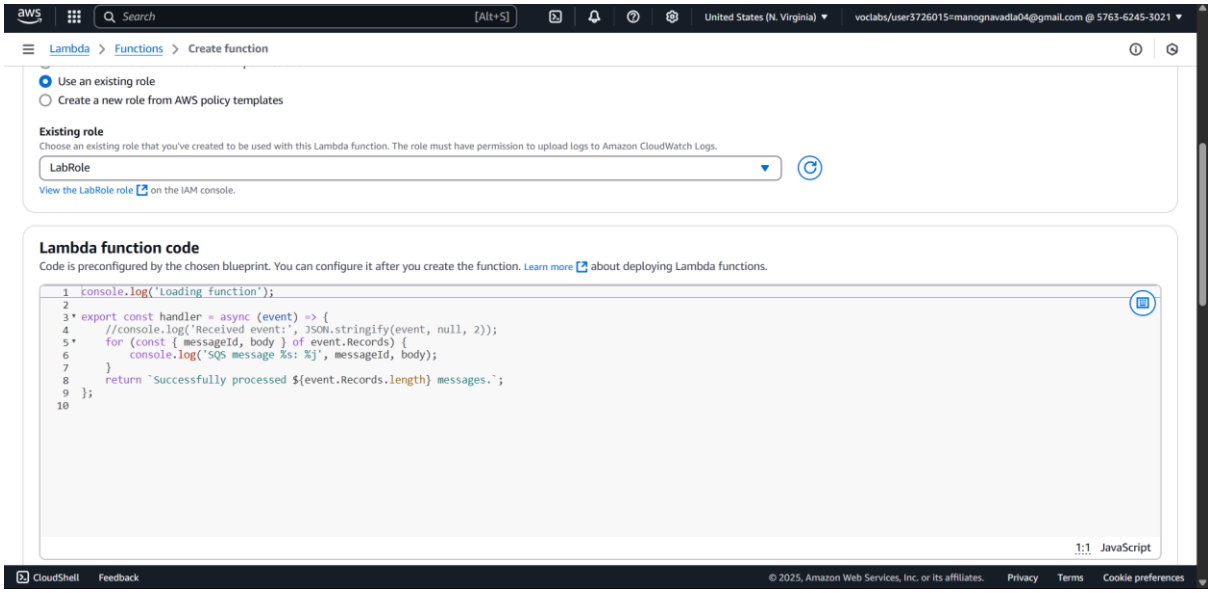


Give the name for the lambda function.

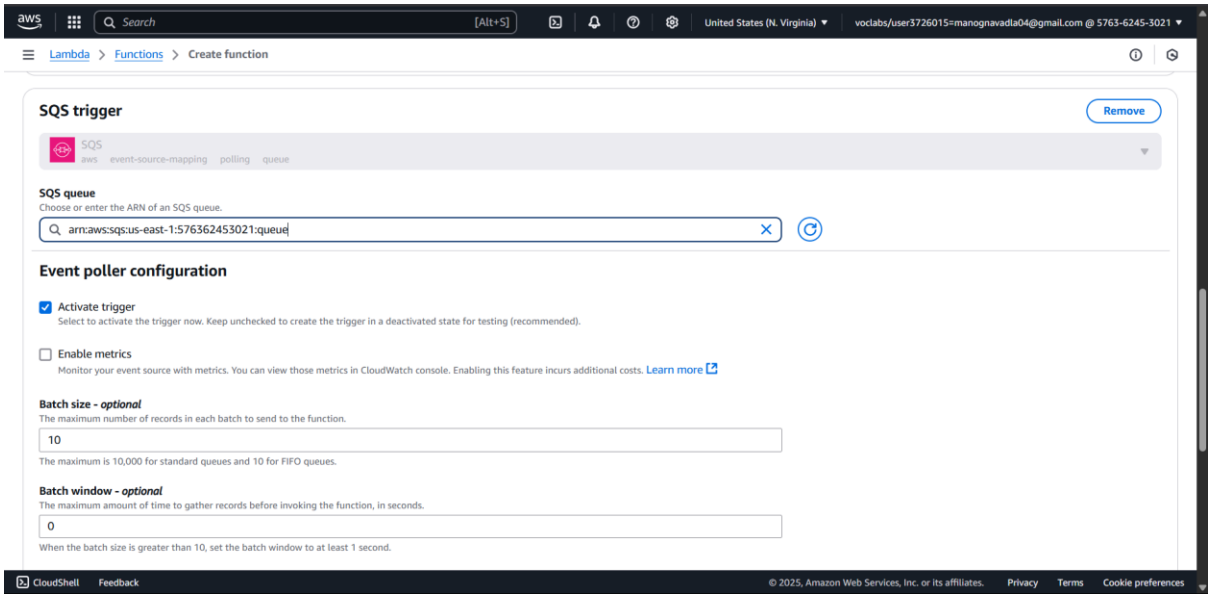
Click create with blueprint Execution role :

Use existing role- labRole

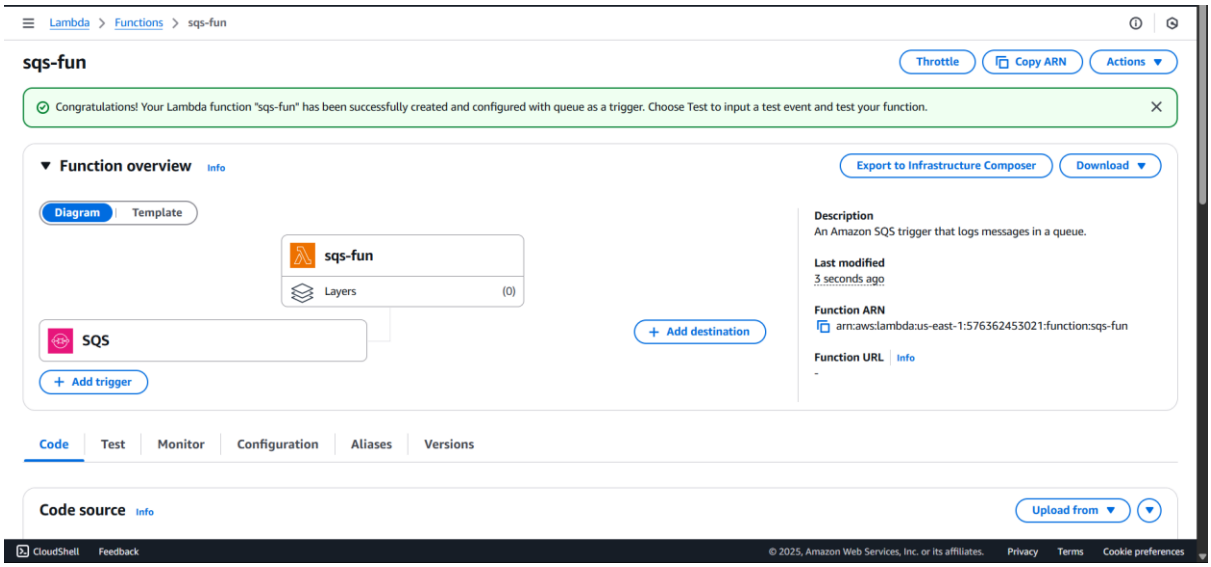

Lambda > Functions > Create function
Create function Info
Choose one of the following options to create your function.
Author from scratch
Start with a simple Hello World example.
Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.
Container image
Select a container image to deploy for your function.
Basic information Info
Blueprint name
Process messages in an SQS queue
An Amazon SQS trigger that logs messages in a queue.
nodejs18.x
Function name
Enter a name that describes the purpose of your function.
myFunctionName
Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).
Runtime
nodejs18.x
Architecture
x86_64
Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
Create a new role with basic Lambda permissions
Use an existing role
Create a new role from AWS policy templates
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


aws
Search
[Alt+S]
United States (N. Virginia)
voclabs/user3726015=manognavadla04@gmail.com @ 5763-6245-3021
Lambda > Functions > Create function
Use an existing role
Create a new role from AWS policy templates
Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
LabRole
View the LabRole role on the IAM console.
Lambda function code
Code is preconfigured by the chosen blueprint. You can configure it after you create the function. Learn more about deploying Lambda functions.
1 console.log('Loading function');
2
3 export const handler = async (event) => {
4 //console.log('Received event:', JSON.stringify(event, null, 2));
5 for (const { messageId, body } of event.Records) {
6 console.log('SQS message %s: %j', messageId, body);
7 }
8 return `Successfully processed ${event.Records.length} messages.`;
9 };
10
1:1 JavaScript
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

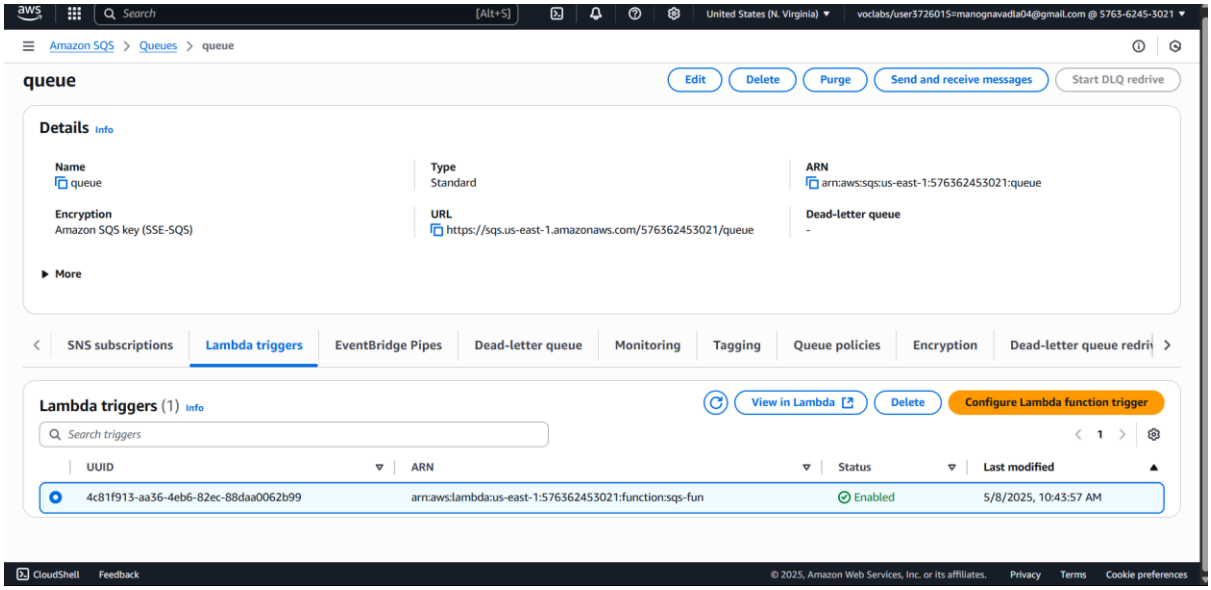
Select Triggers



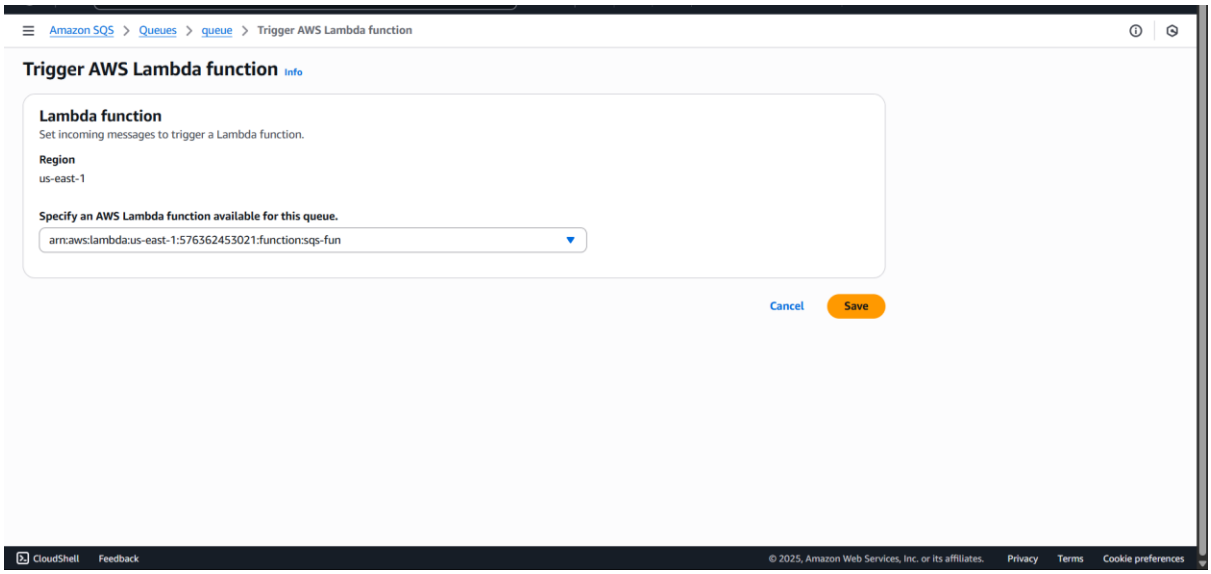
Created a Lambda function



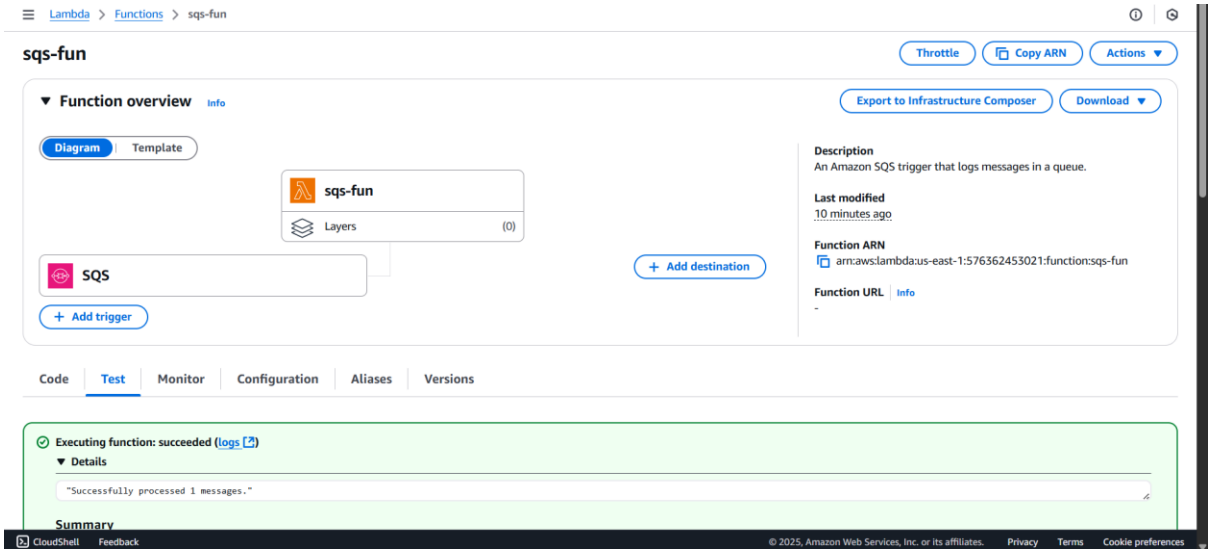
Go to the SQS queue and navigate to Lambda Triggers, you will be able to see the function you created

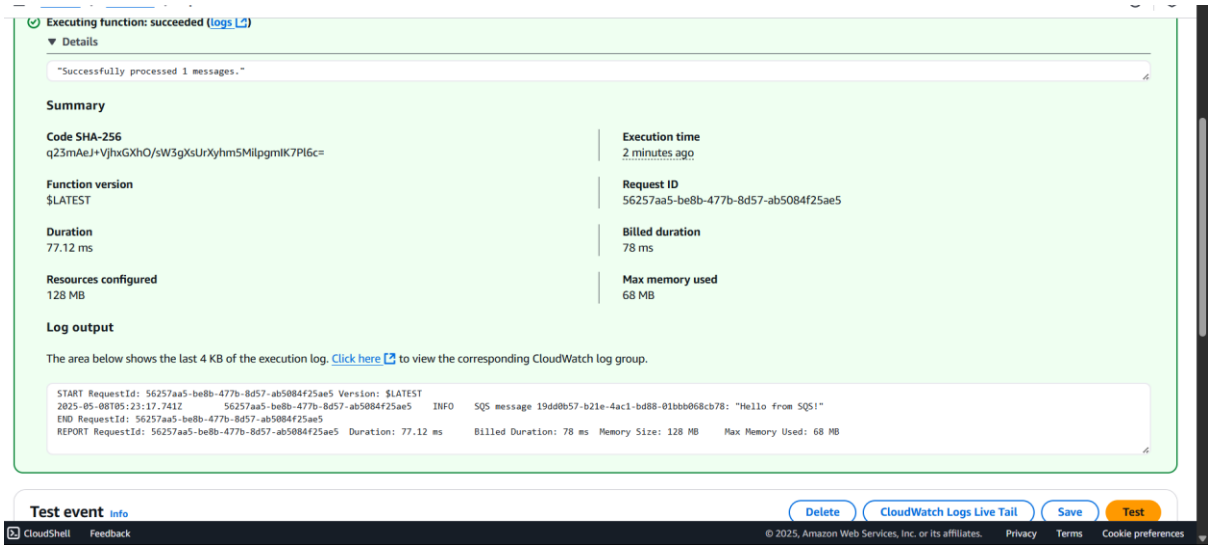


Click on configure lambda trigger

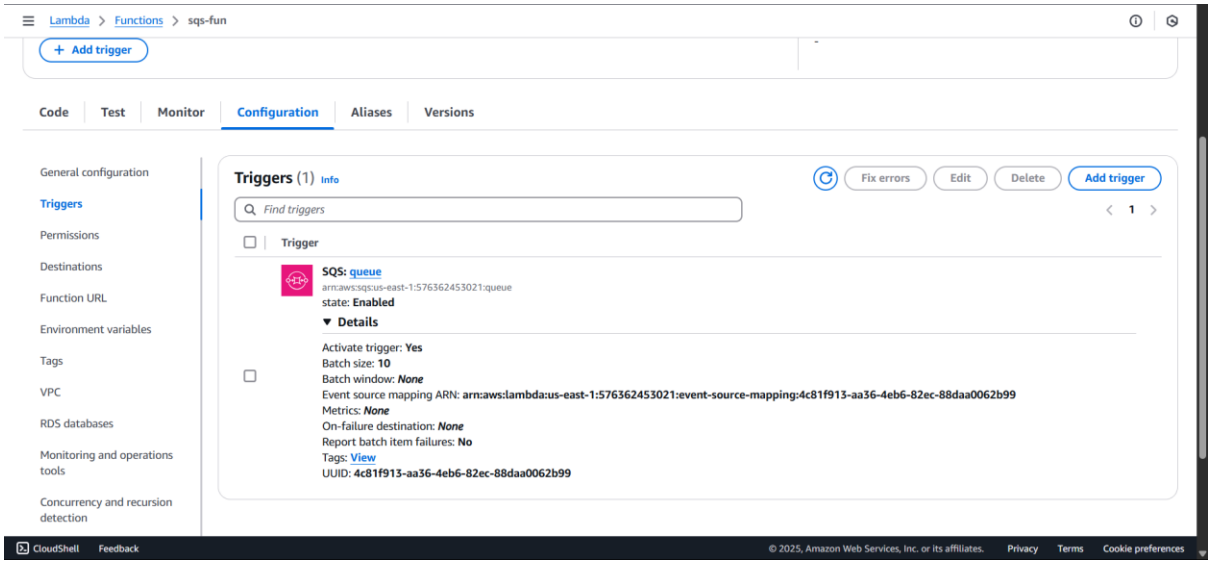


Create an event and test the lambda function

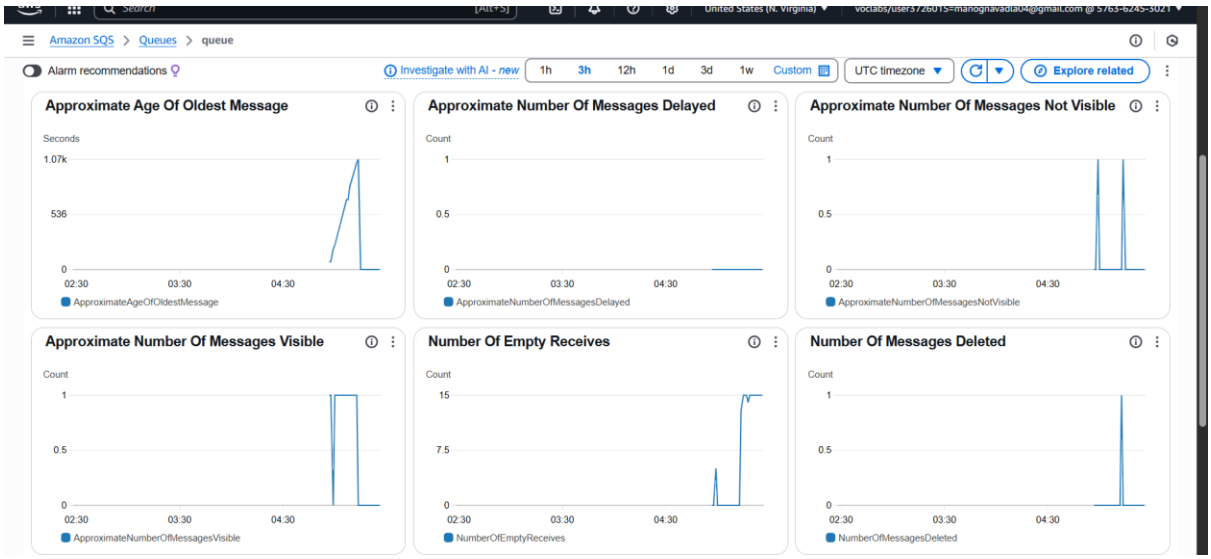


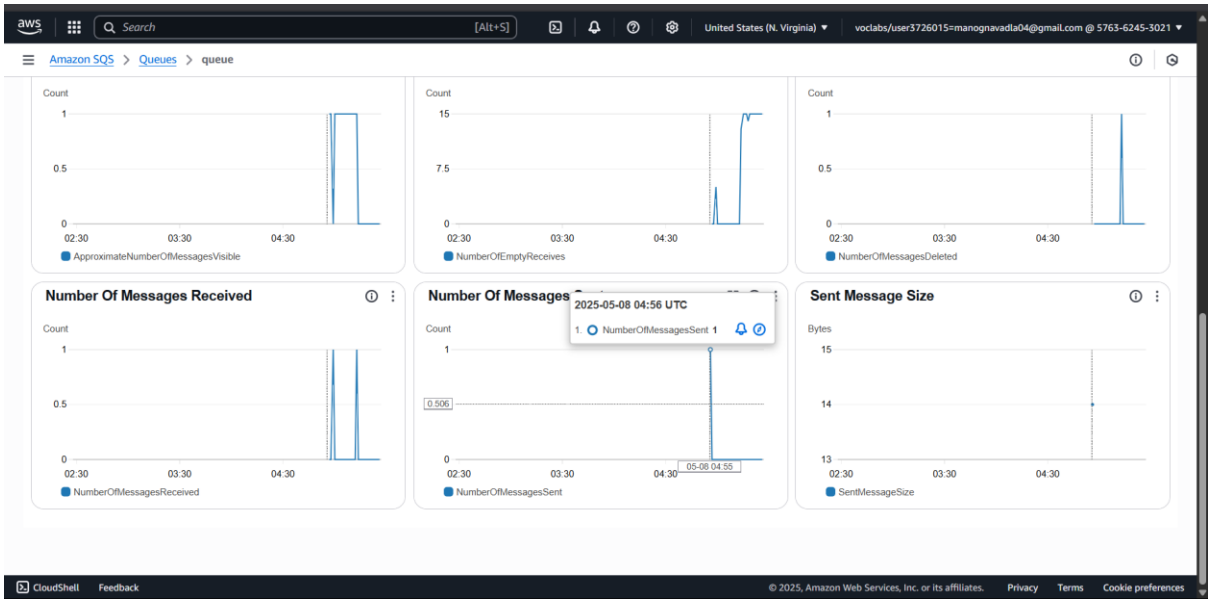


The lambda function is activated



Go to the monitoring tab in queue





Navigate to cloud watch through lambda function, you can see the message that was sent