Assignment – 1 - Lambda

Problem Statement:

You work for XYZ Corporation. Your corporation wants to launch a new web-based application and they do not want their servers to be running all the time. It should also be managed by AWS. Implement suitable solutions.

Tasks To Be Performed:

1. Create a sample Python Lambda function.

2. Set the Lambda Trigger as SQS and send a message to test invocations

Create a sample Python Lambda function.

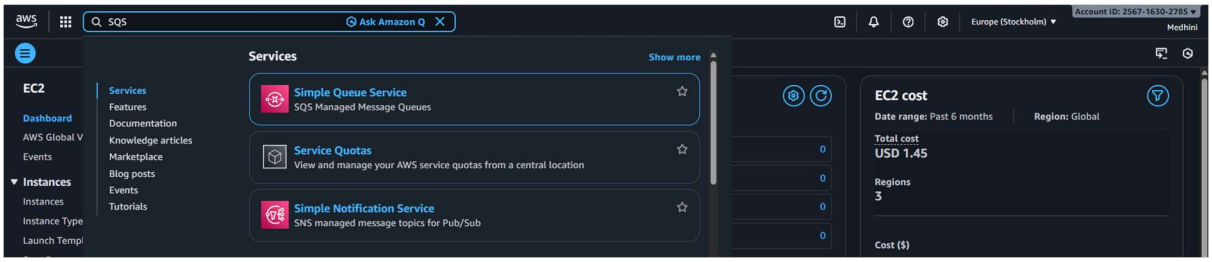
STEP 1: Create an SQS Queue

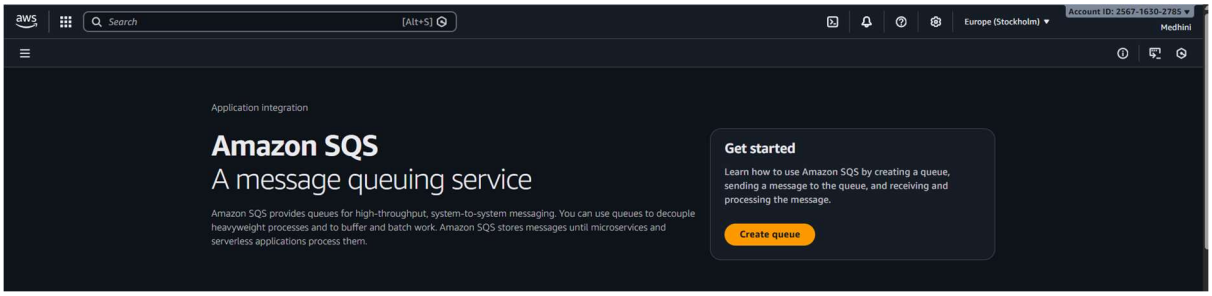
Open SQS

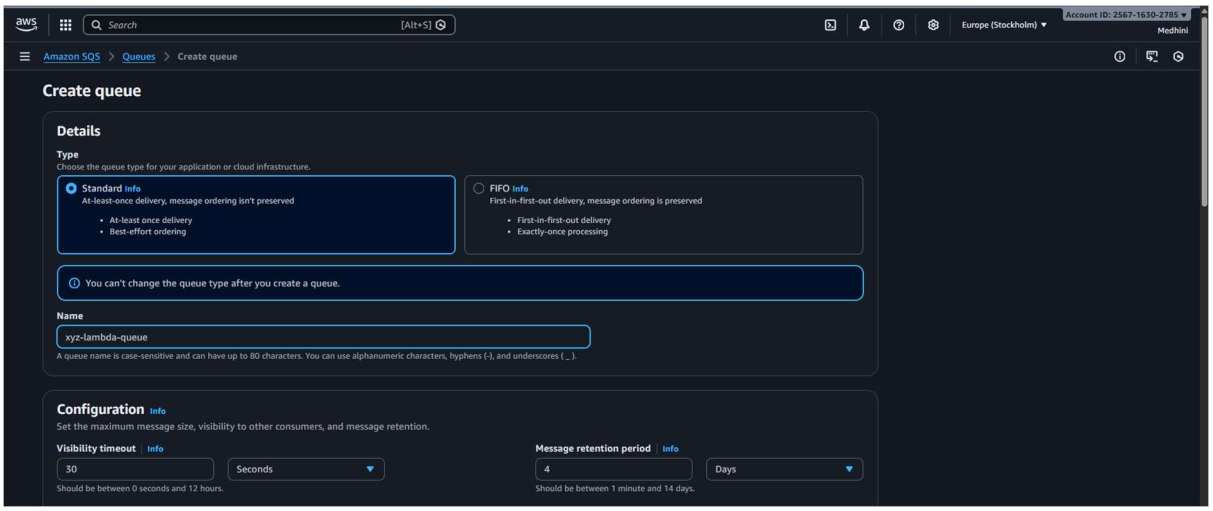
- Login to AWS Console
- Search SQS
- Click Create queue

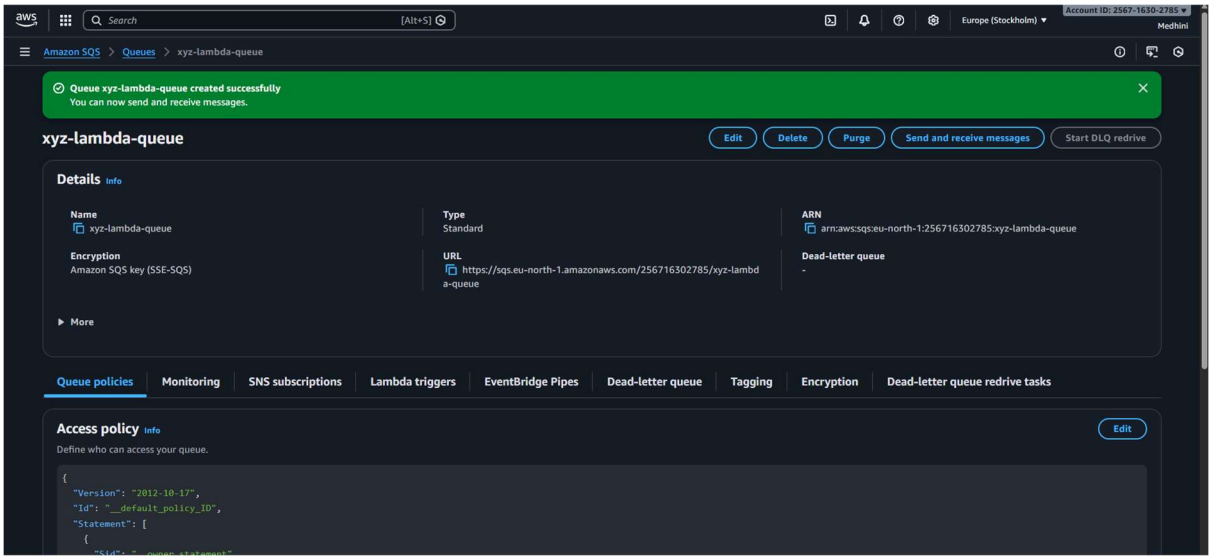
Queue Settings

- Type: Standard
- Queue name: xyz-lambda-queue
- Leave all defaults
- Click Create queue









STEP 2: Create IAM Role for Lambda

Lambda needs permission to read messages from SQS.

Open IAM

- Go to **IAM - Roles**
- Click **Create role**

Select Trusted Entity

- Select **AWS Service**
- Choose **Lambda**
- Click **Next**

Attach Policies

Attach these policies:

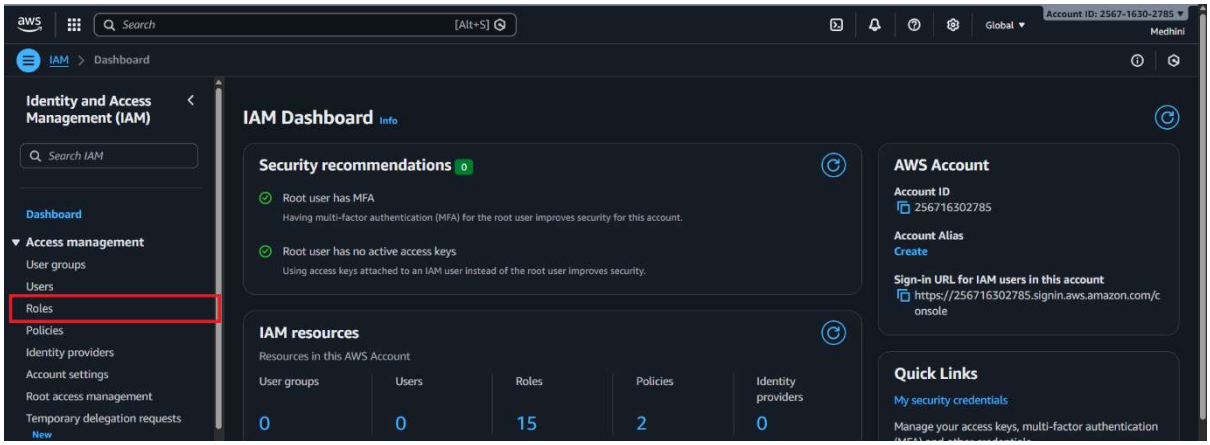
AWSLambdaBasicExecutionRole

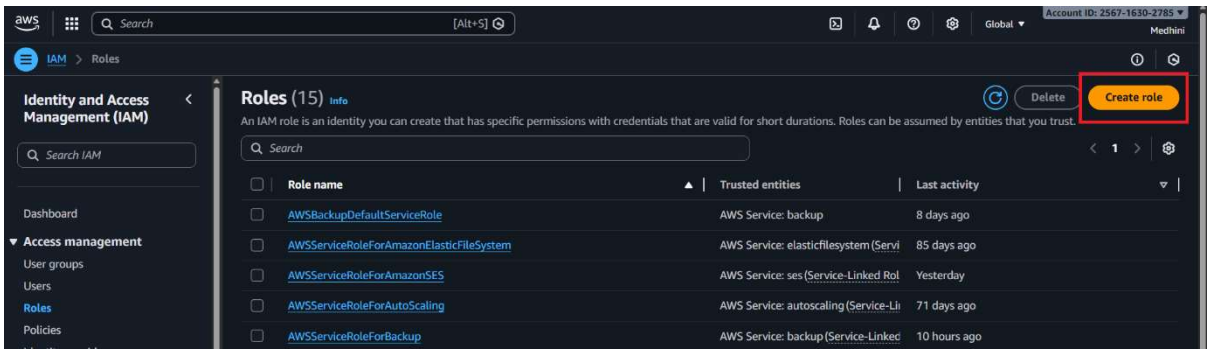
AmazonSQSFullAccess (for learning purpose)

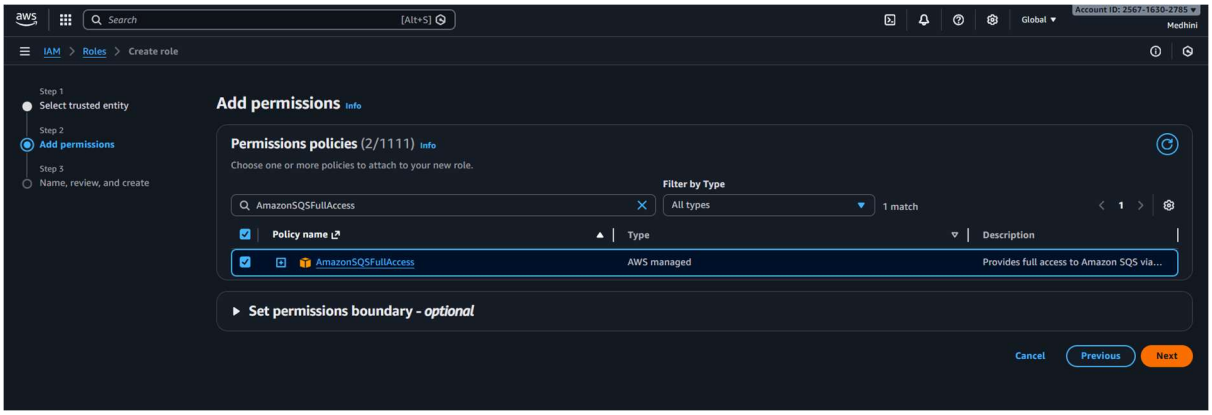
Click **Next**

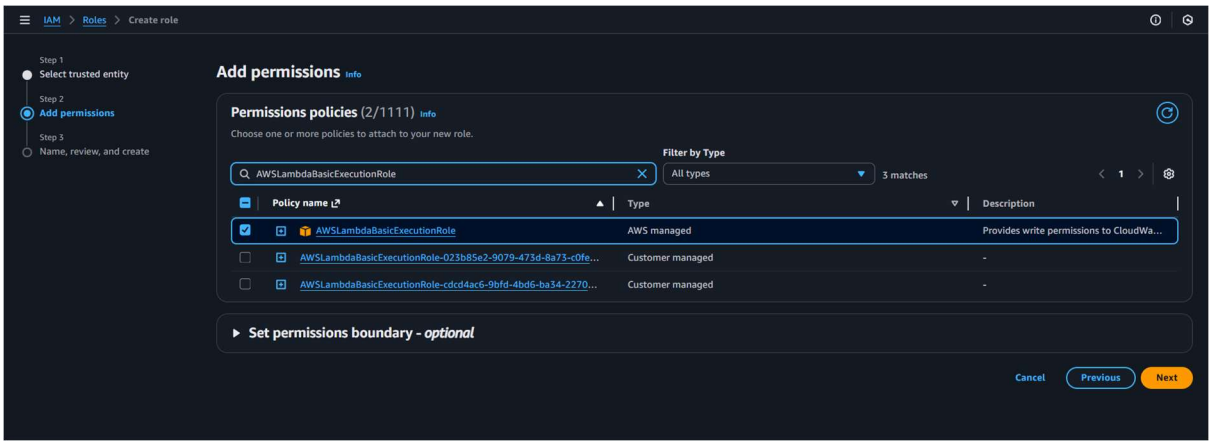
Role Name

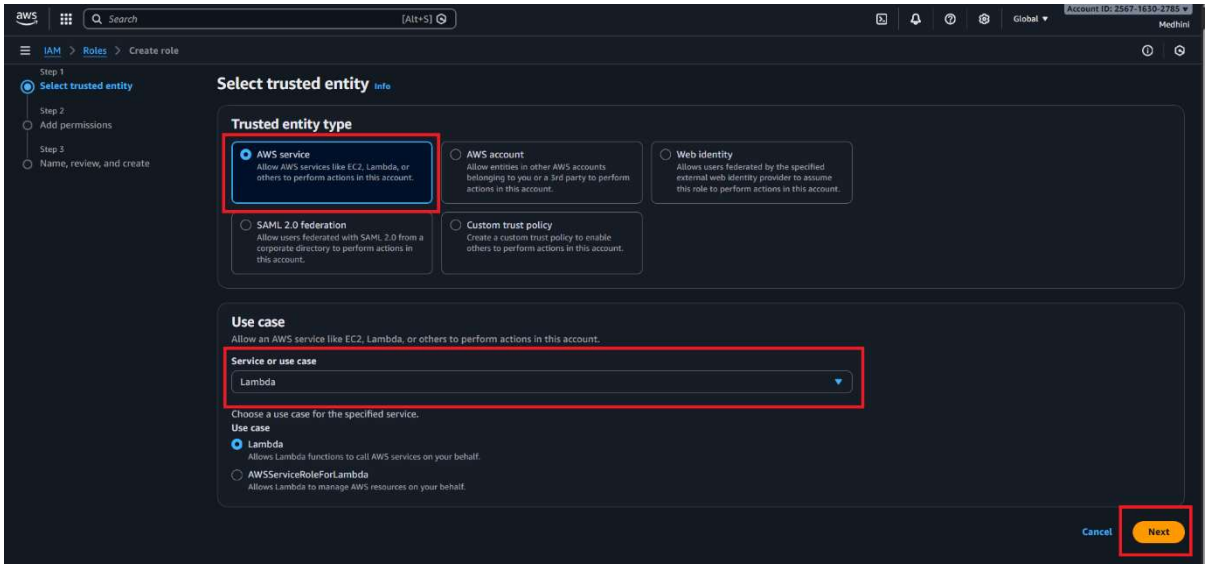
- Role name: lambda-sqs-role
- Click **Create role**

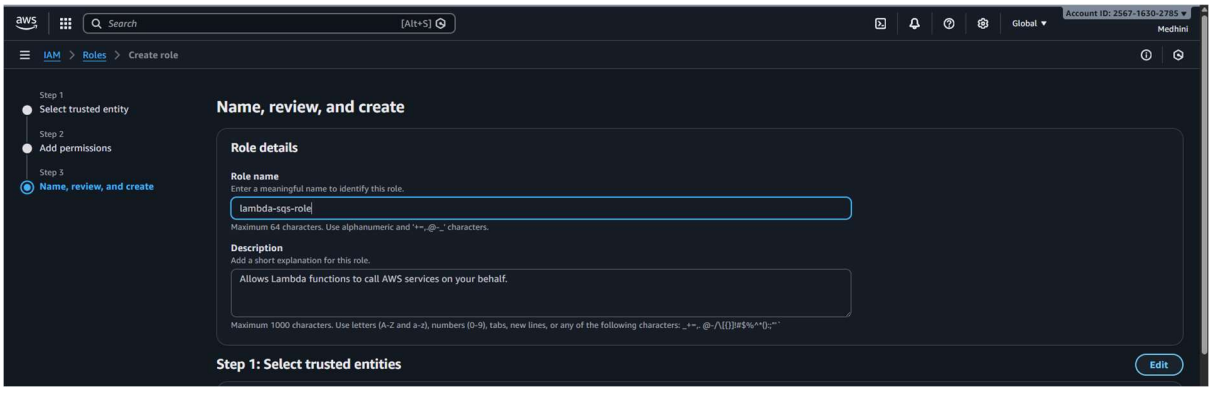


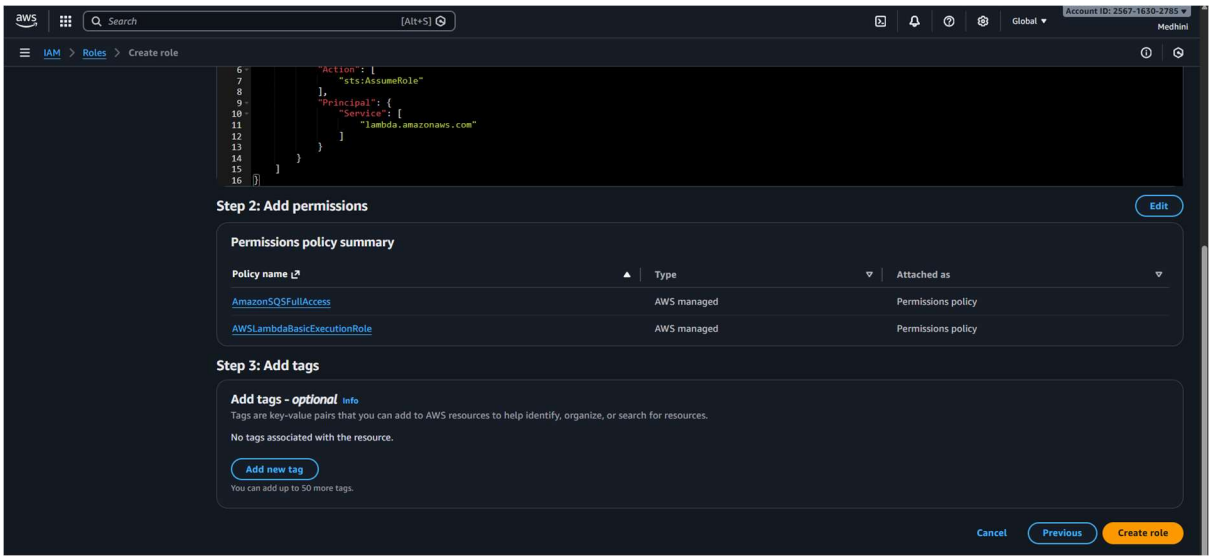


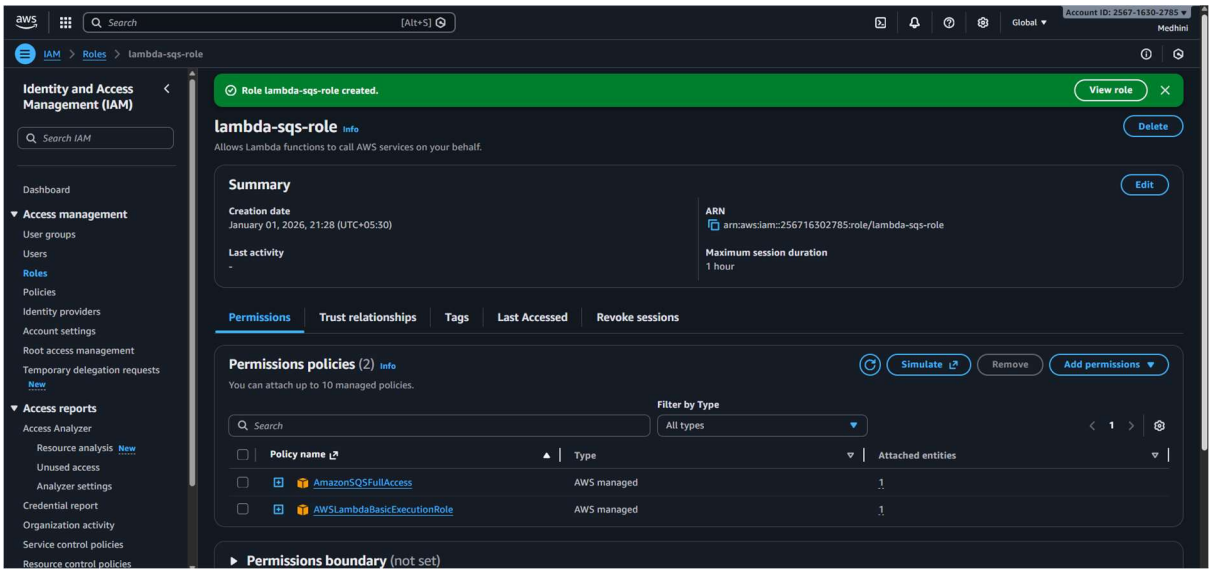



IAM > Roles > Create role
Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create
Add permissions Info
Permissions policies (2/1111) Info
Choose one or more policies to attach to your new role.
Filter by Type
AWSLambdaBasicExecutionRole
All types
3 matches
Policy name
Type
Description
AWSLambdaBasicExecutionRole
AWS managed
Provides write permissions to CloudWa...
AWSLambdaBasicExecutionRole-023b85e2-9079-473d-8a73-c0fe...
Customer managed
AWSLambdaBasicExecutionRole-cdcd4ac6-9bfd-4bd6-ba34-2270...
Customer managed
Set permissions boundary - optional
Cancel
Previous
Next


Search
[Alt+S]
Global
Account ID: 2567-1630-2785
Medhini
IAM > Roles > Create role
Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create
Select trusted entity Info
Trusted entity type
AWS service
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
AWS account
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
Web identity
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
SAML 2.0 federation
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
Custom trust policy
Create a custom trust policy to enable others to perform actions in this account.
Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.
Service or use case
Lambda
Choose a use case for the specified service.
Use case
Lambda
Allows Lambda functions to call AWS services on your behalf.
AWSServiceRoleForLambda
Allows Lambda to manage AWS resources on your behalf.
Cancel
Next


Search
[Alt+S]
Global
Account ID: 2567-1630-2785
Medhini
IAM > Roles > Create role
Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create
Name, review, and create
Role details
Role name
Enter a meaningful name to identify this role.
lambda-sqs-role
Maximum 64 characters. Use alphanumeric and '+=,.@-_' characters.
Description
Add a short explanation for this role.
Allows Lambda functions to call AWS services on your behalf.
Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: _+=,. @-/\[{}]!#$%^*();"'
Step 1: Select trusted entities
Edit





STEP 3: Create Python Lambda Function

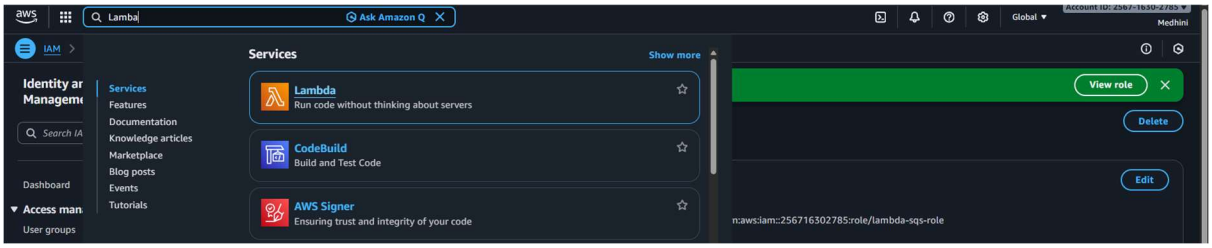
Open Lambda

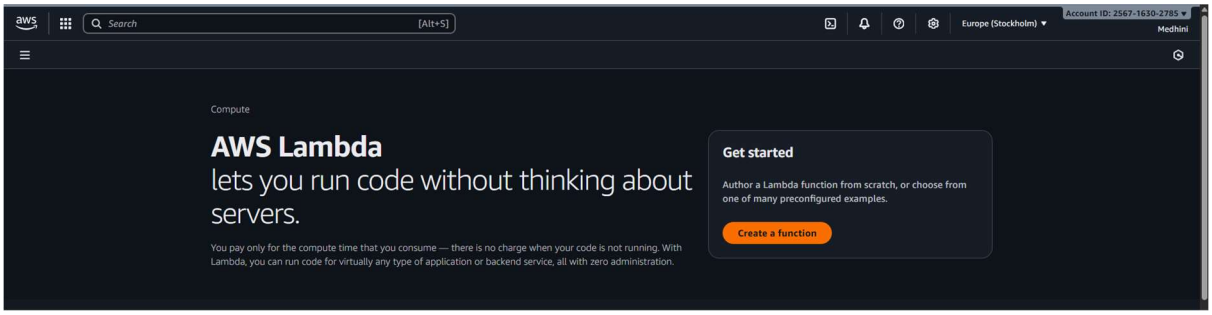
- Search **Lambda**
- Click **Create function**

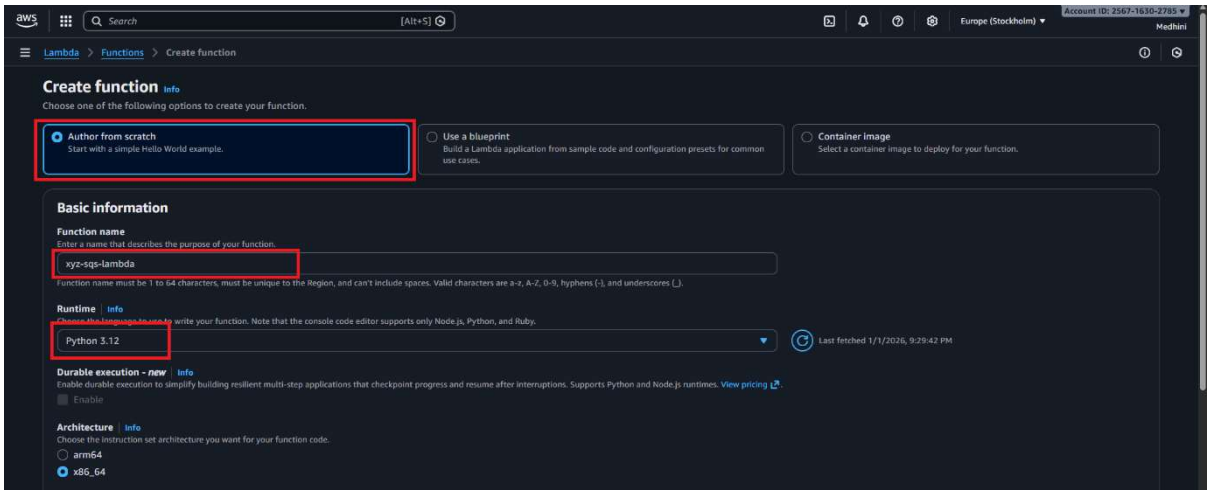
Function Settings

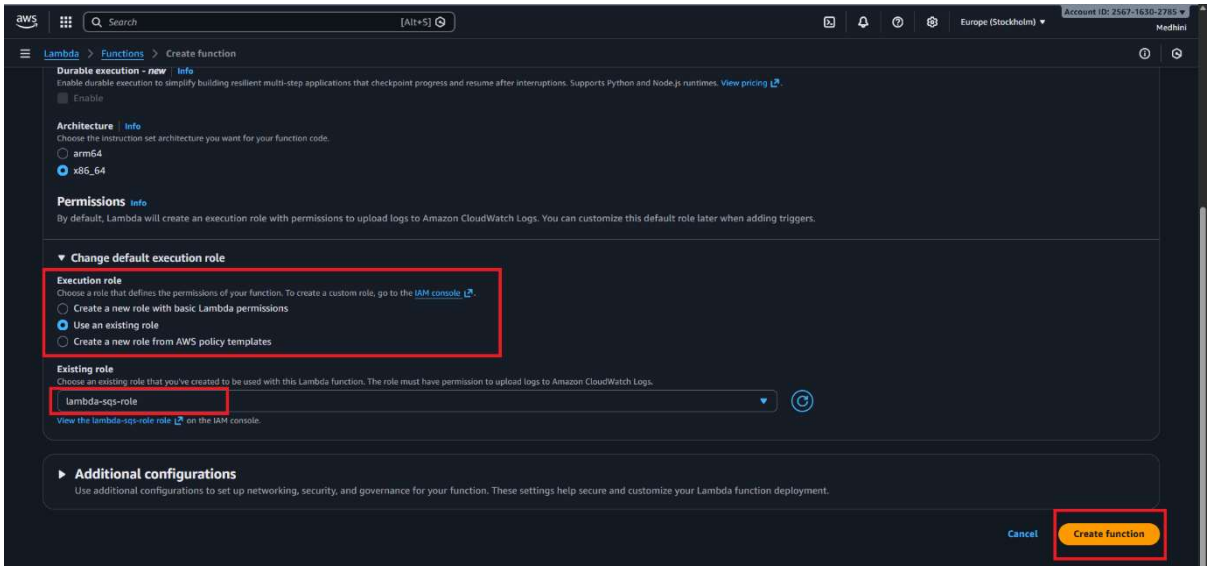
- **Author from scratch**
- **Function name:** xyz-sqs-lambda
- **Runtime:** Python 3.12
- **Execution role:** Use existing role
- Select **lambda-sqs-role**

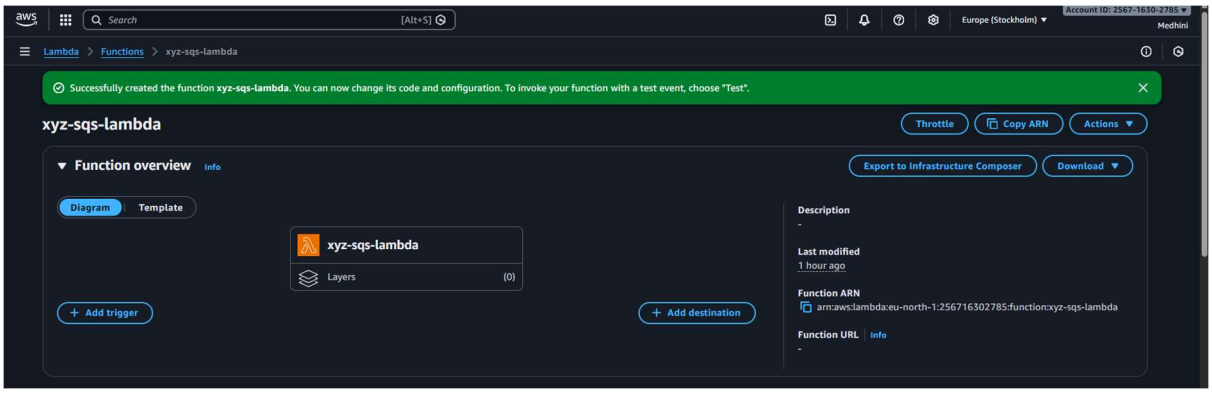
Click **Create function**









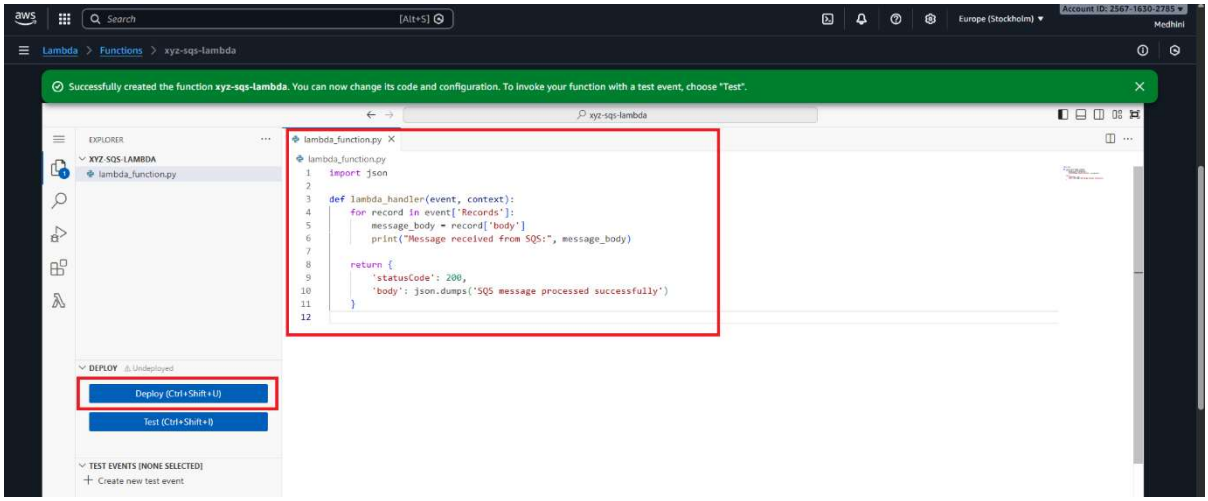


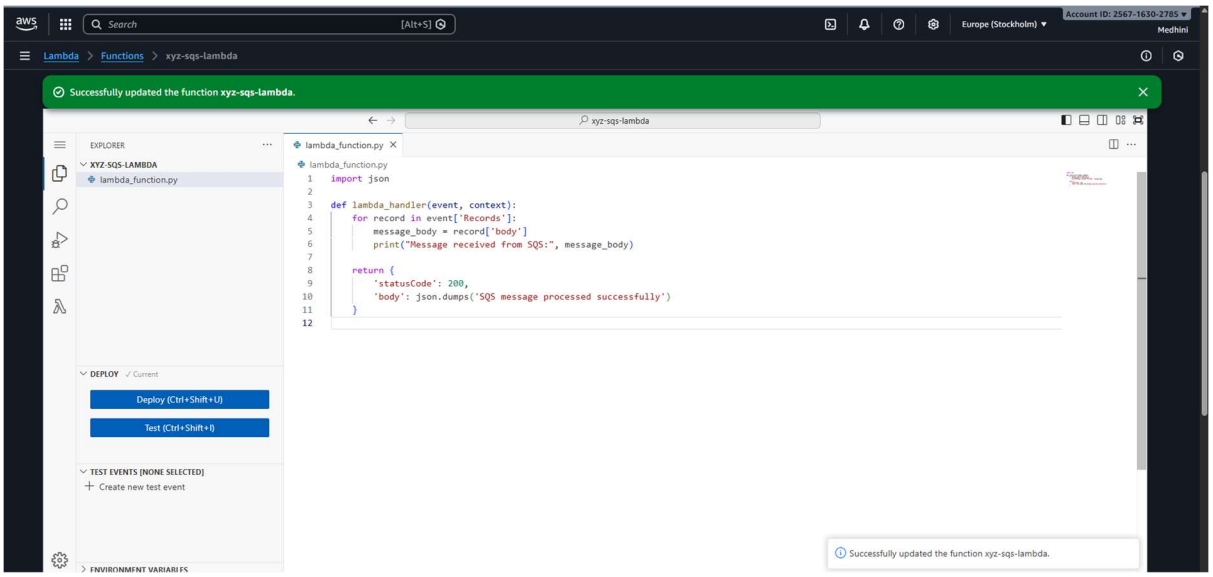
STEP 4: Add Python Code to Lambda

Replace default code with this

```
import json
def lambda_handler(event, context):
  for record in event['Records']:
    message_body = record['body']
    print("Message received from SQS:", message_body)
  return {
    'statusCode': 200,
    'body': json.dumps('SQS message processed successfully')
  }
```

Click **Deploy**





STEP 5: Add SQS as Lambda Trigger

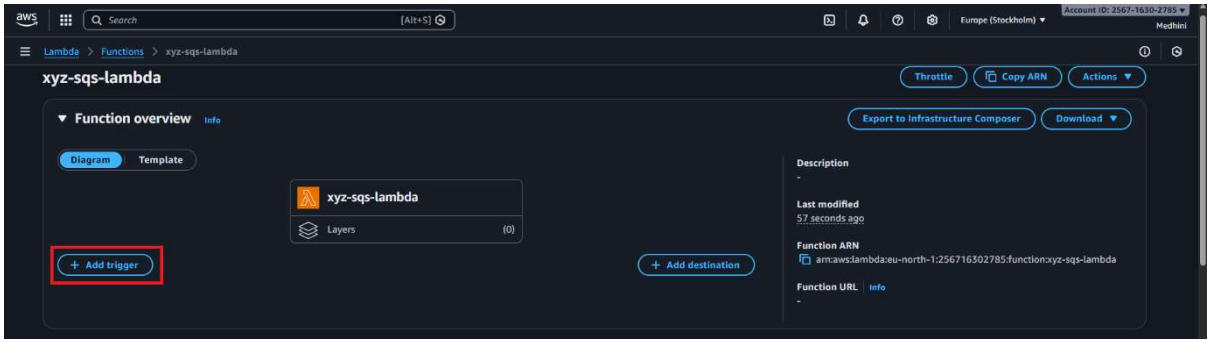
Add Trigger

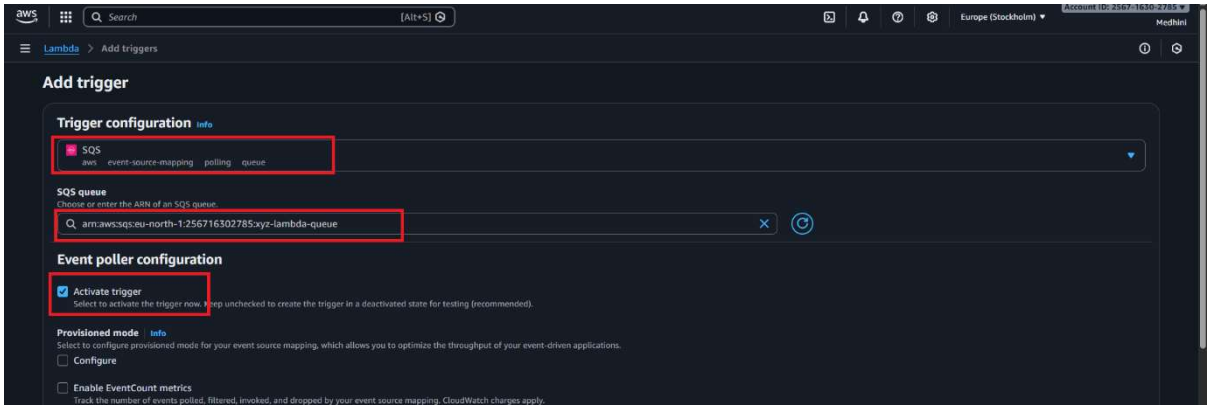
- In Lambda → Click **Add trigger**
- Select **SQS**

Configure Trigger

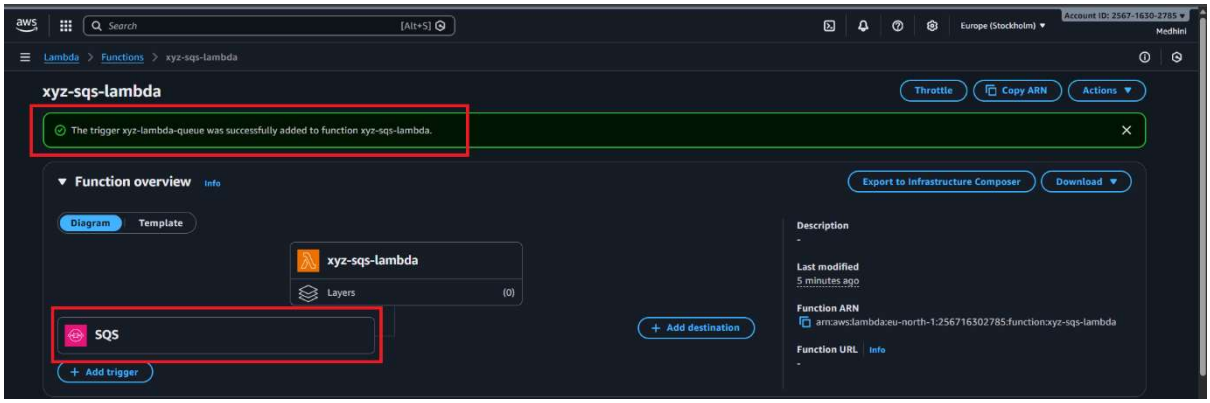
- Choose queue: xyz-lambda-queue
- Enable trigger
- Click **Add**

Now SQS is connected to Lambda





Scroll down and click on **“Add”**



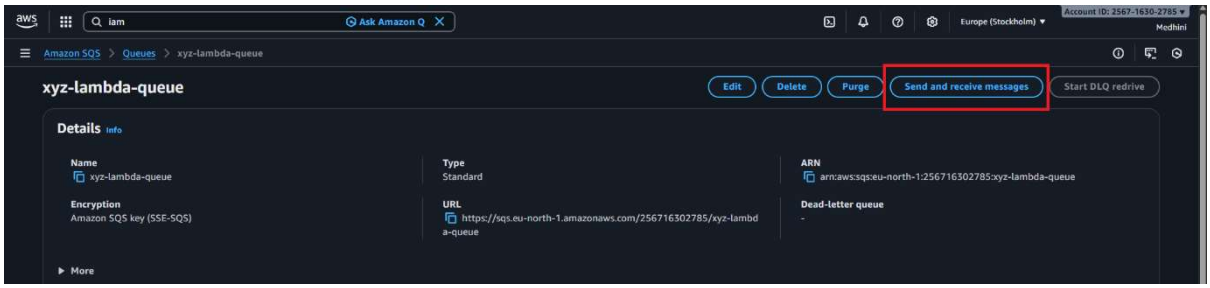
STEP 6: Send Message to SQS (Test)

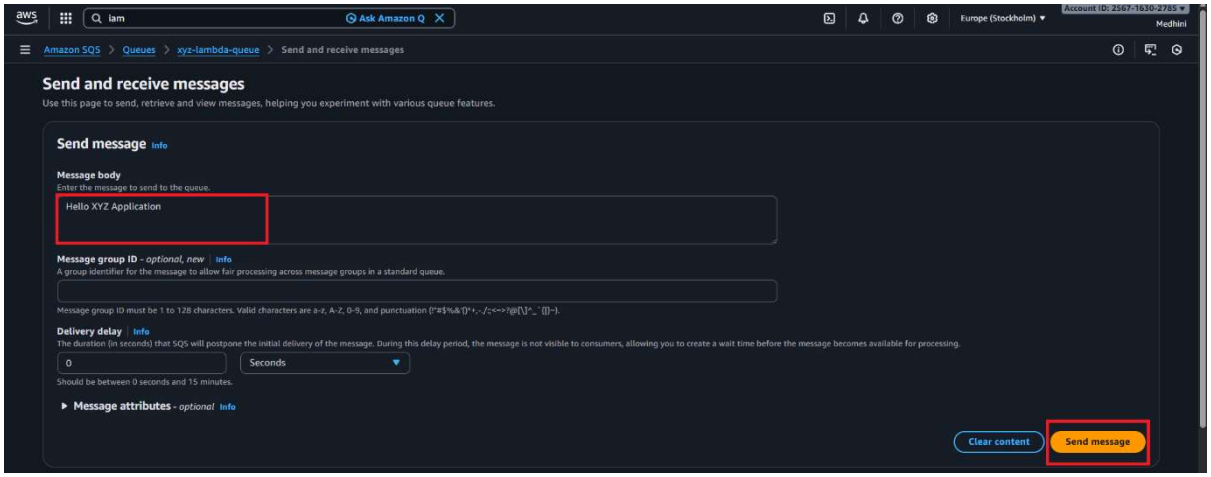
Open SQS

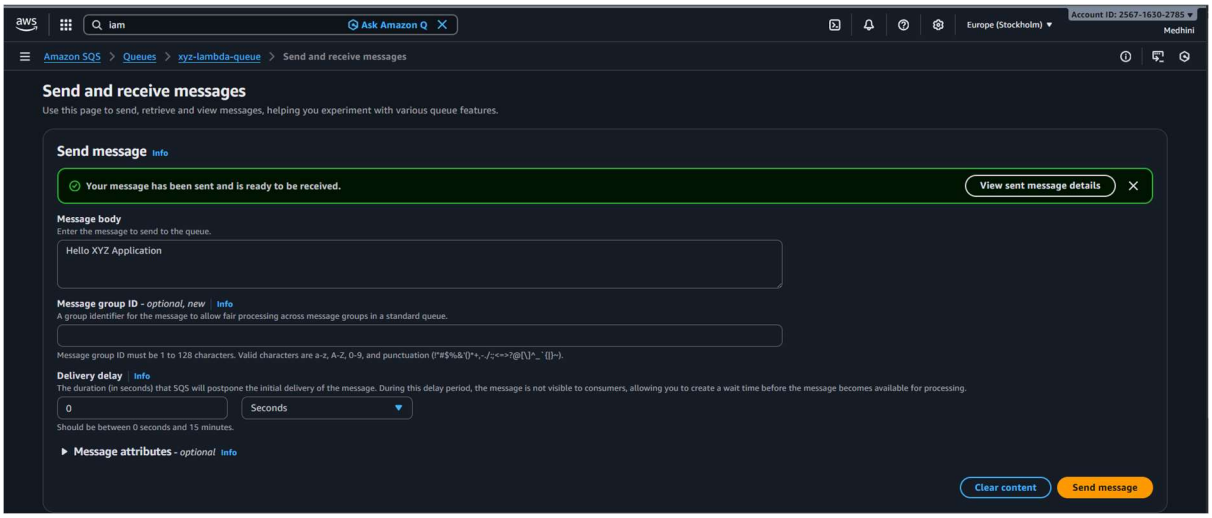
- Click xyz-lambda-queue
- Click **Send and receive messages**

Send Message

Message body example:



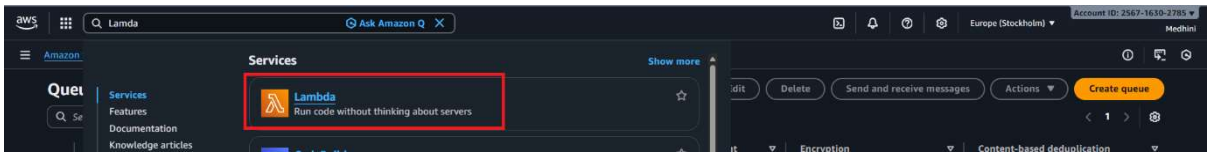




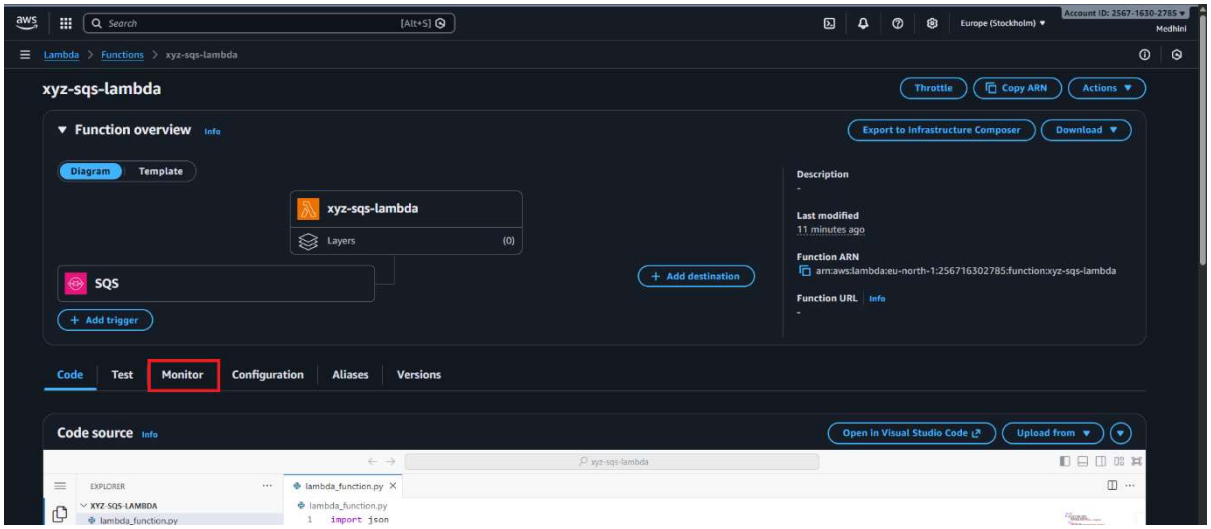
STEP 7: Verify Lambda Execution

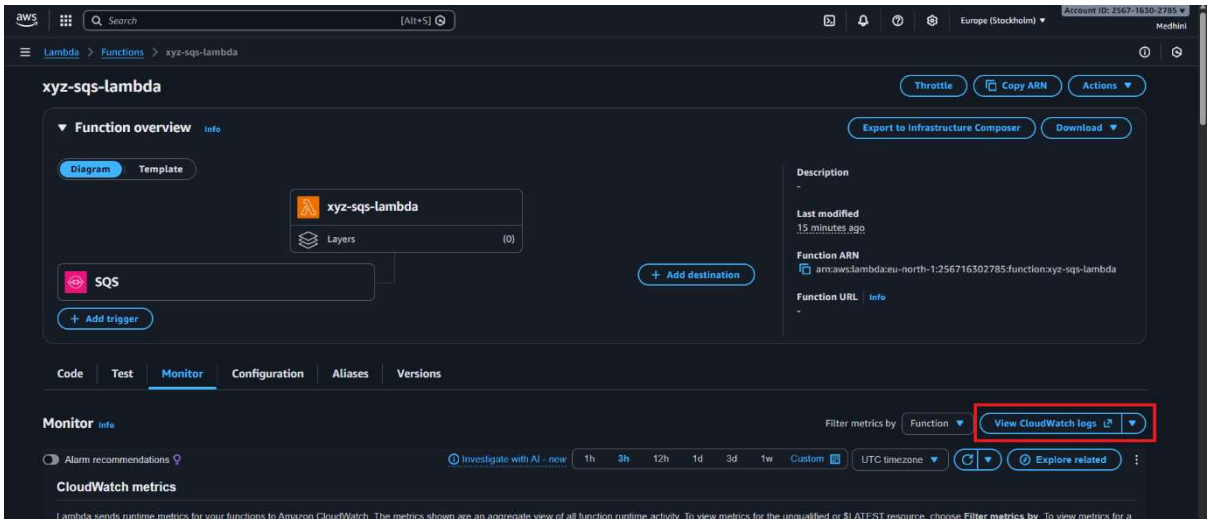
Go to Lambda

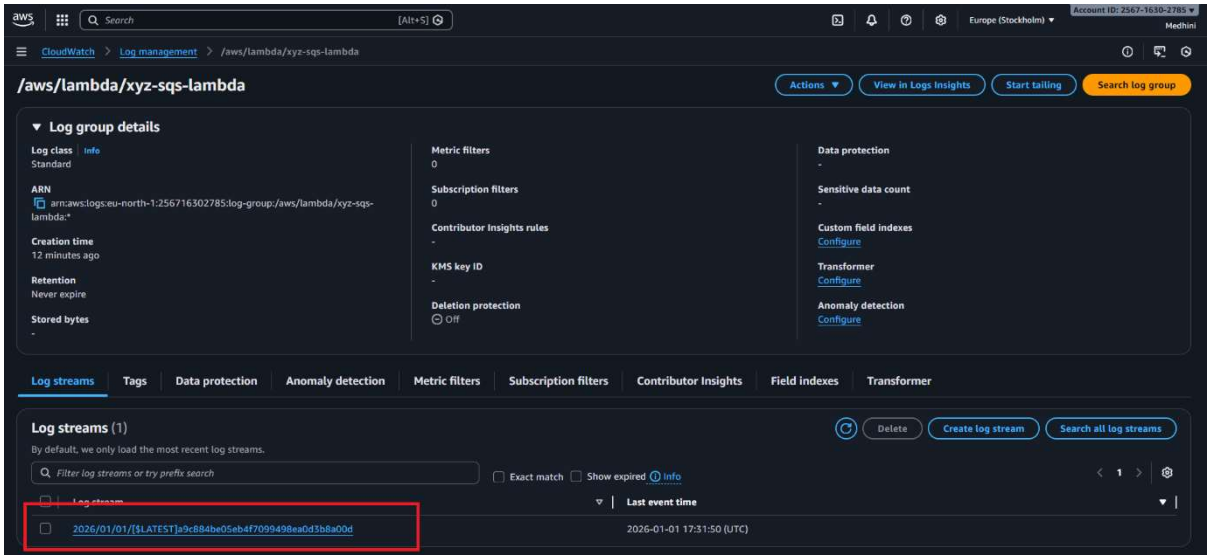
- Open xyz-sqs-lambda
- Click **Monitor**
- Click **View logs in CloudWatch**



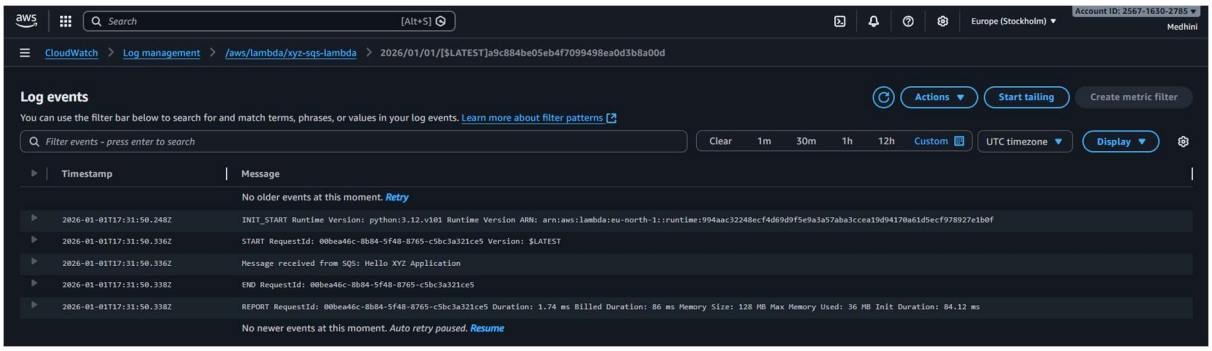
Assignment – 1 - Lambda







Assignment – 1 - Lambda



Lambda triggered automatically