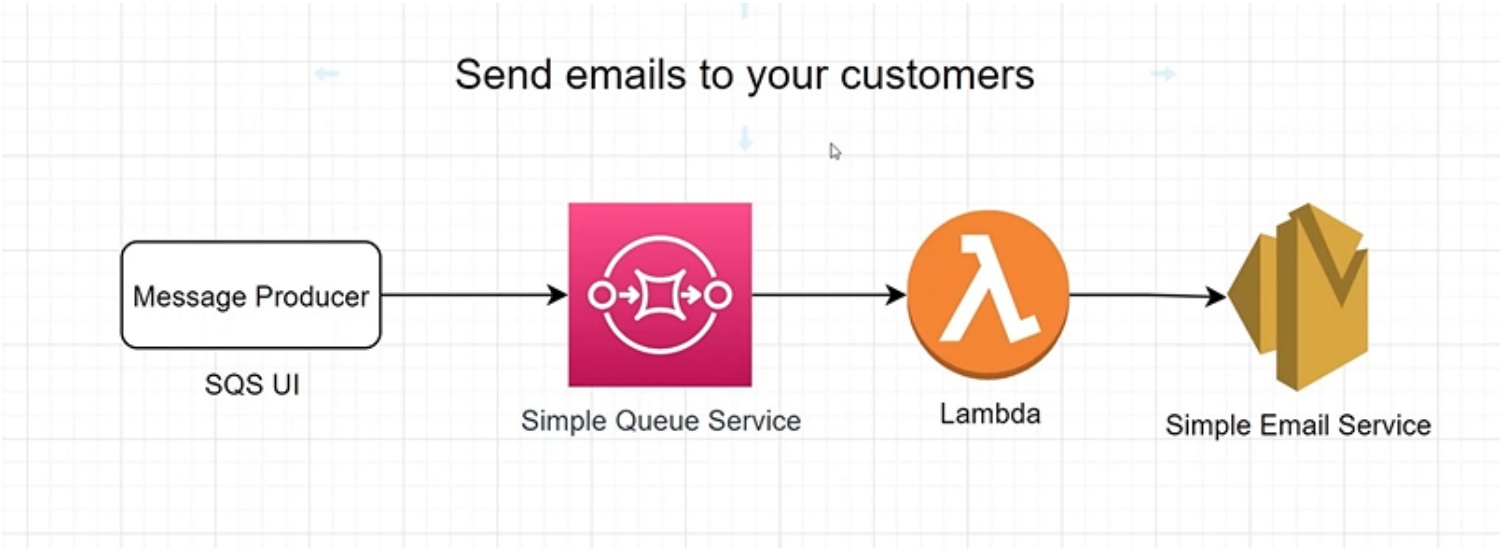
# EMAIL setup with SQS, SES, & Lambda

Hi cloud people, I am **Pawan Gambhir** and I'm an AWS Cloud practitioner. In this blog, I'll be walking you through an **email setup** that I recently architected in the AWS cloud.




Project Architecture.

**Video Guide:**

Setup an emailing system using AWS SQS, SES and Lambda!

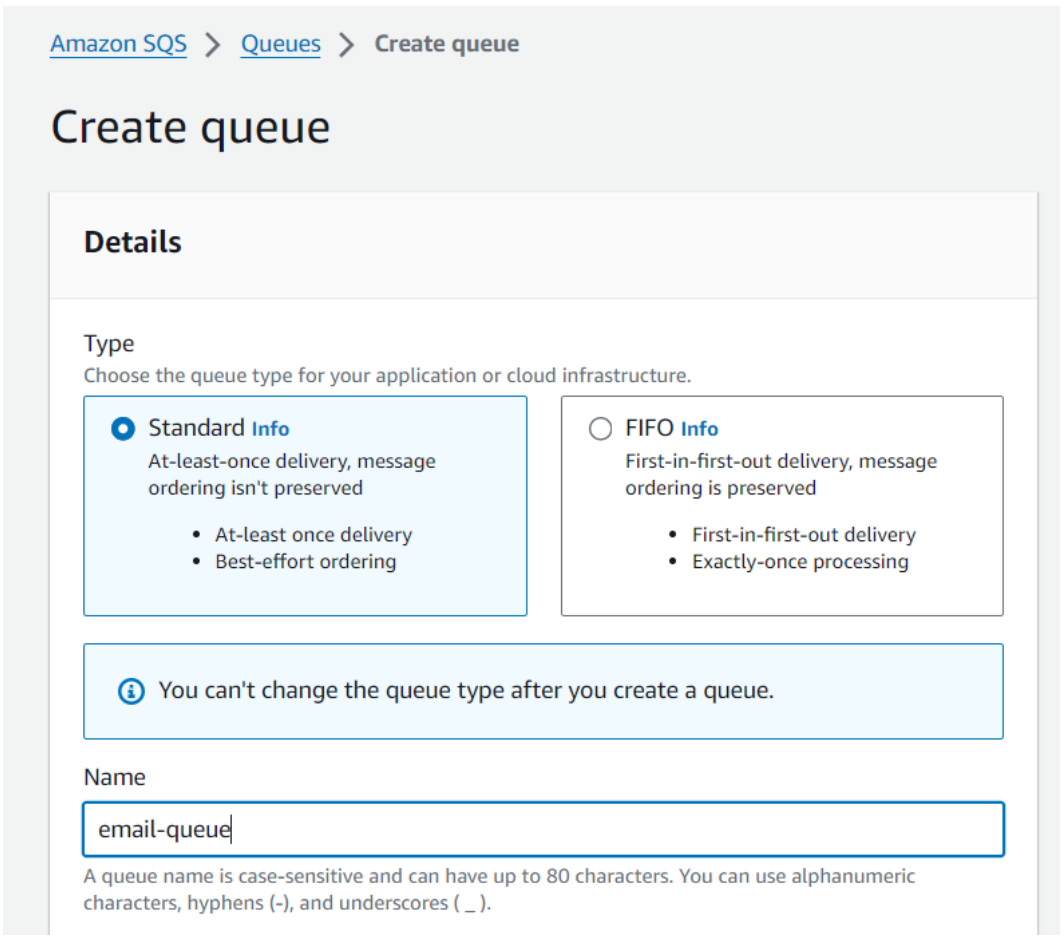
In this video I've architected an AWS emailing setup as a project by using services like SQS. SES, lambda and IAM roles.

https://youtu.be/jMSBTydSBKk


In this project you'll get your hands-on to SQS, SES, Lambda and IAM (roles) services, as the architecture is displayed above. So let's dive into it. 😁

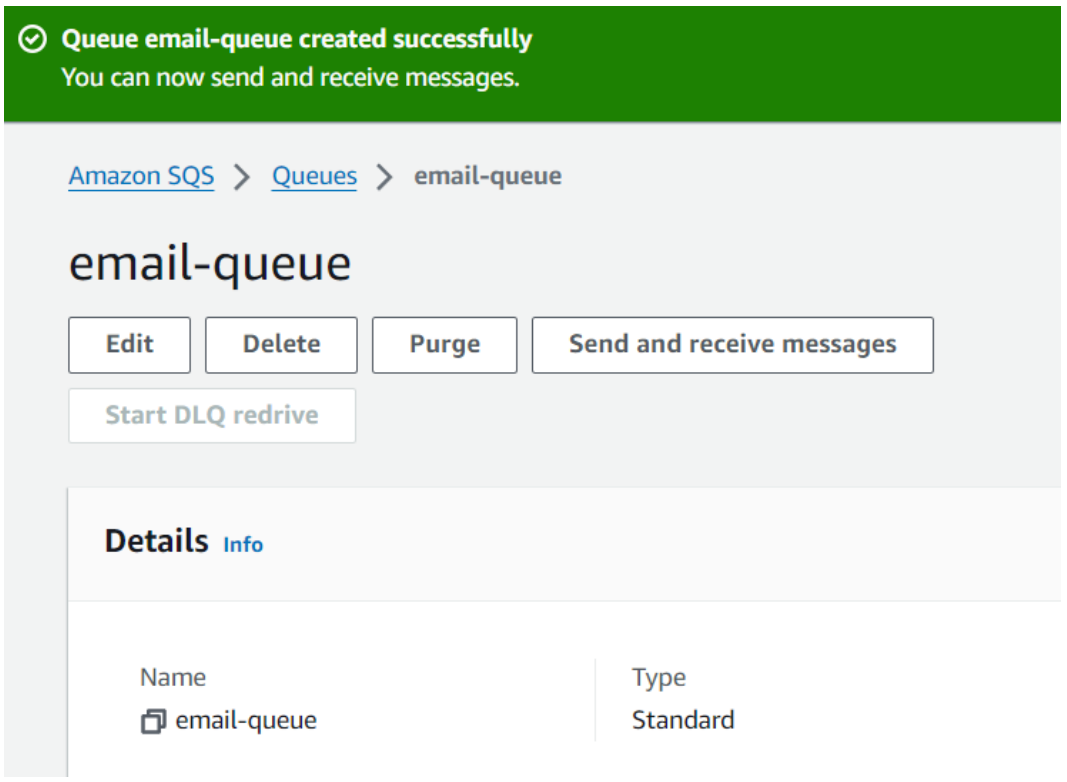
## SQS Configuration

1. Select SQS service from AWS dashboard.




2. Create a Queue and select Standard Type.

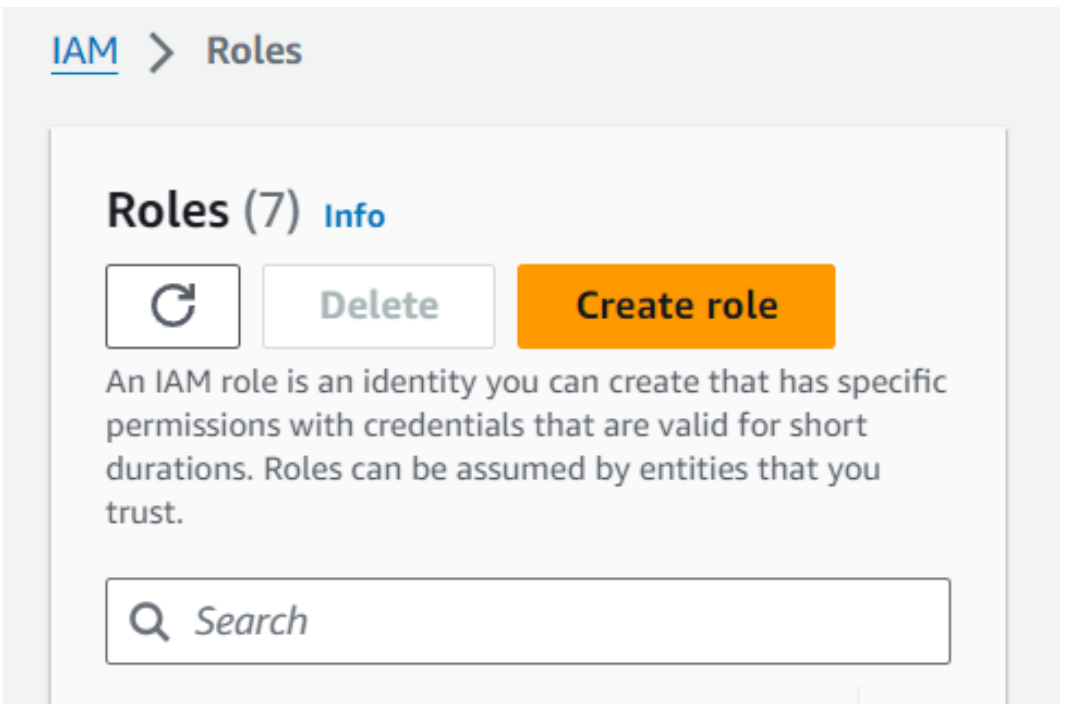
3. Write any name for your queue service. ( you can choose any non-existing name )
4. Now, click on ‘create’ and your queue will be formed.Que



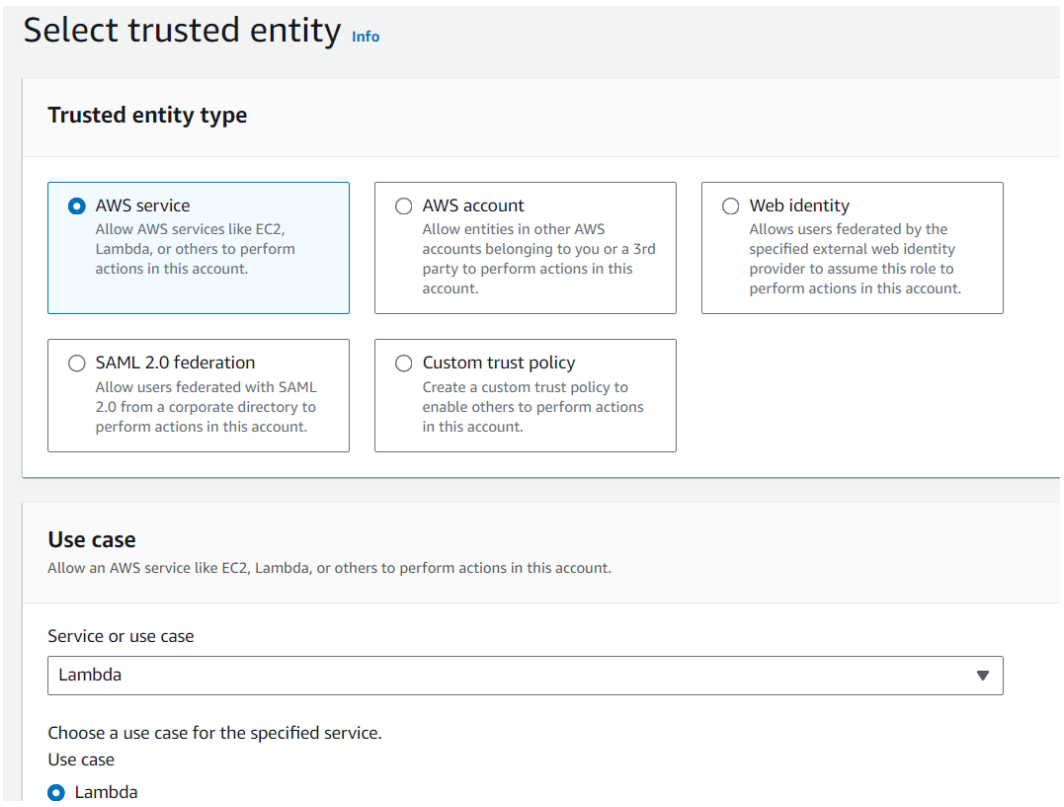

Queue setup completed.

## IAM role Configuration

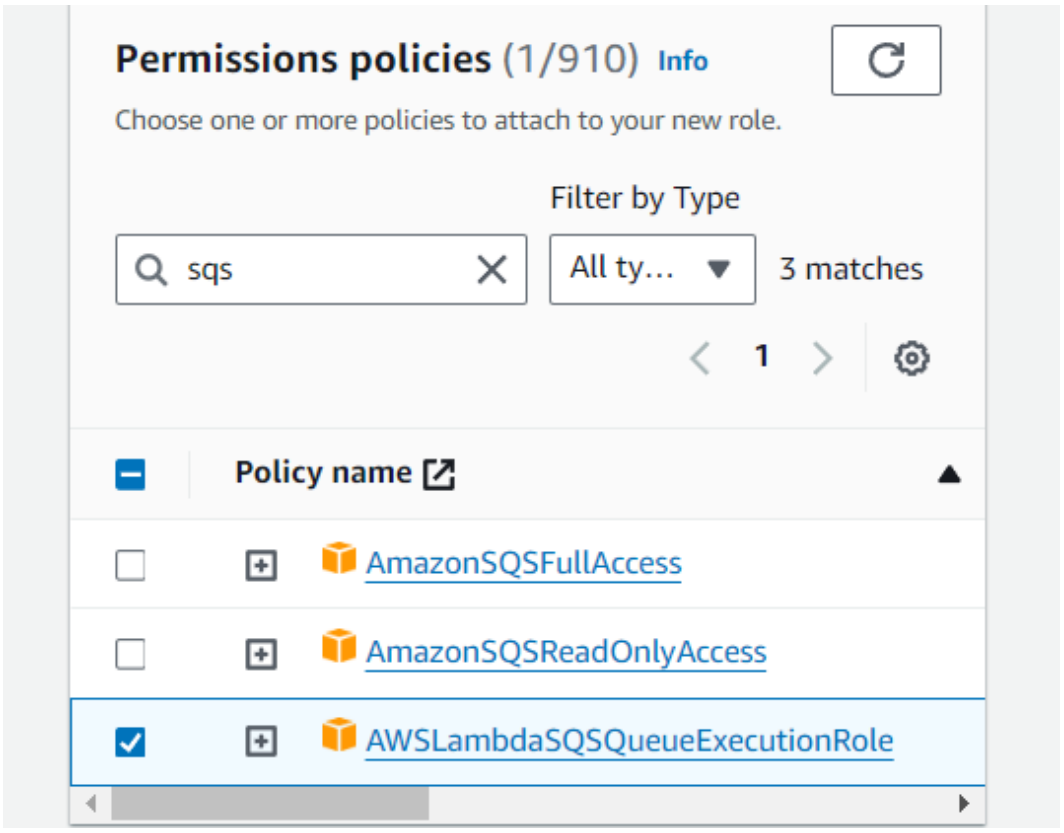
1. Select IAM service from the AWS dashboard.
2. On the left panel, click on roles.
3. Create a new role

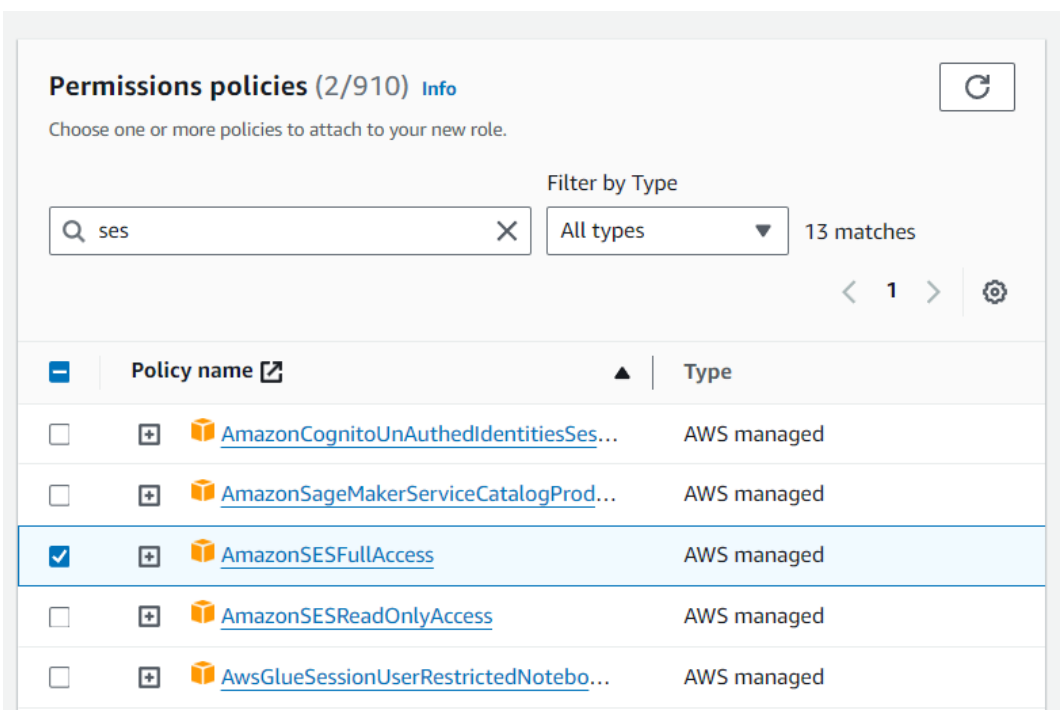



4. Select default settings, and in select “Lambda” as service.

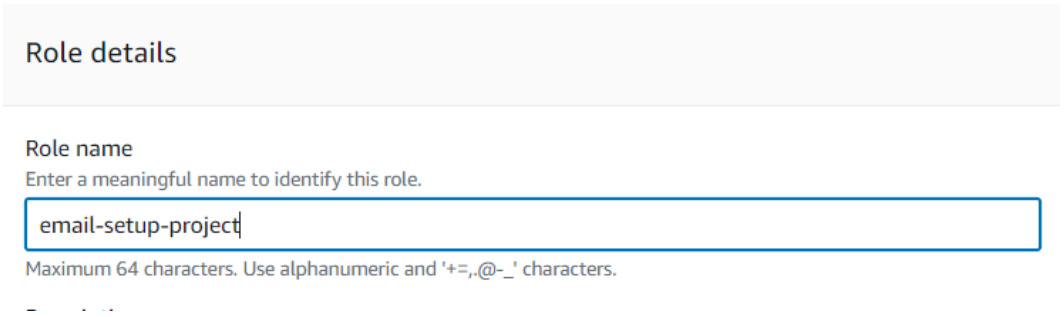



5. Click next, and choose **“AWSLambdaSQSQueueExecutionRole”** & **“AmazonSESFullAccess”** permissions to give access of the SQS, SES services.



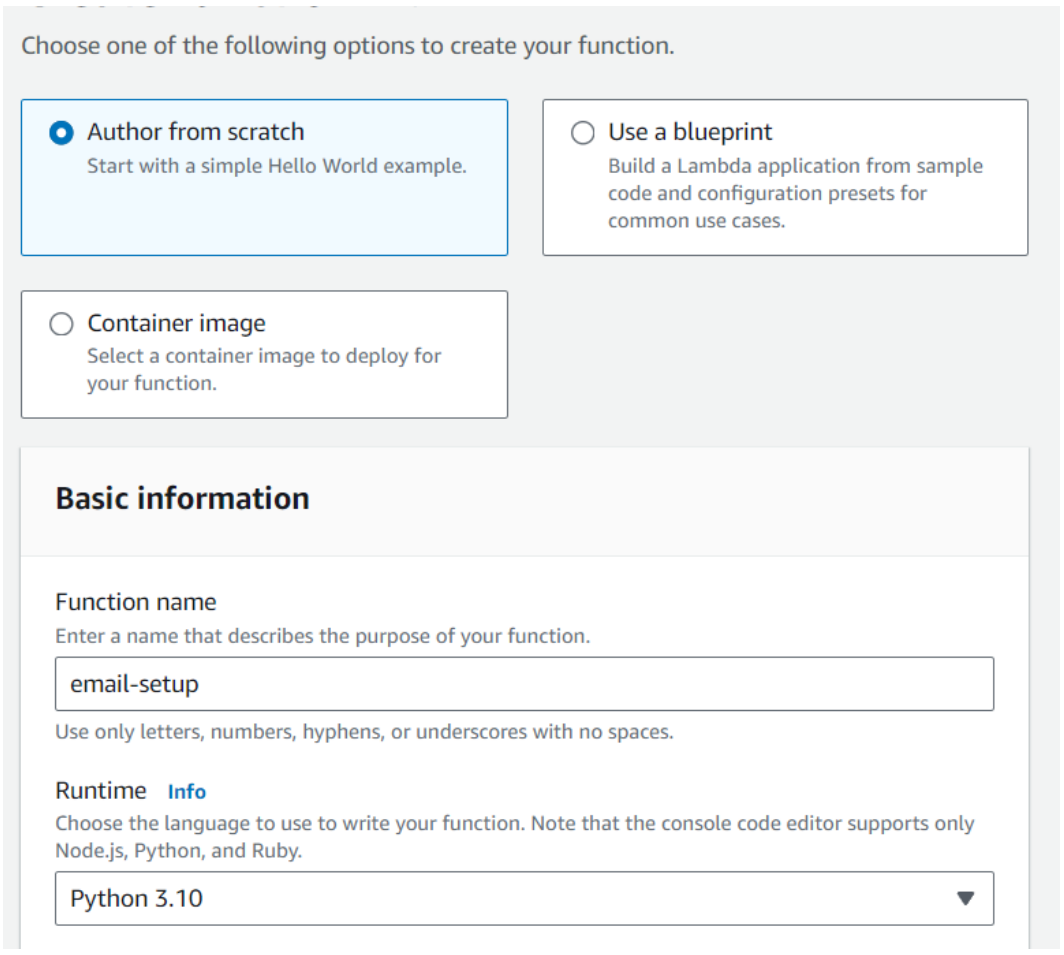




6. Finally, write the name for the role you want to give it and click create.

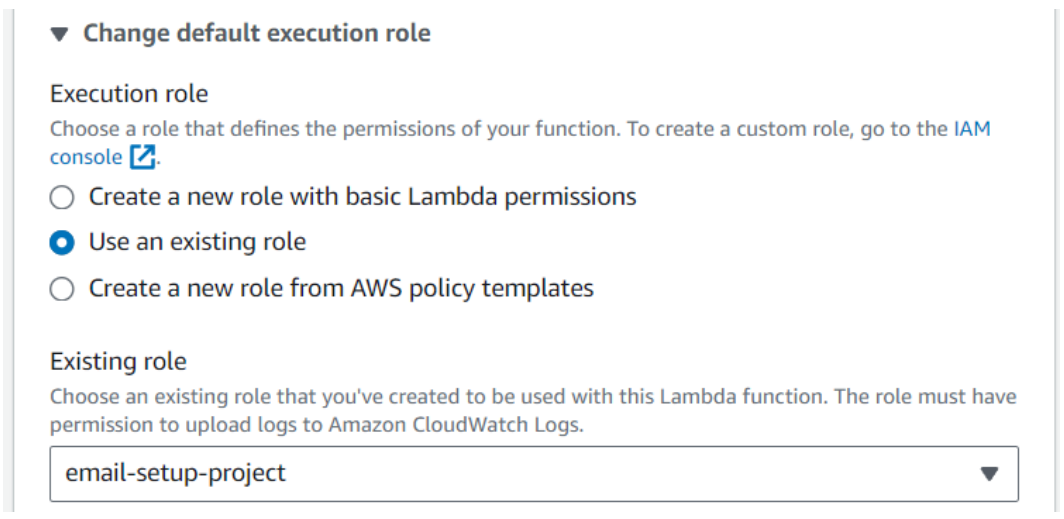



## Lambda Configuration

1. Select Author from scratch as we'll be putting our own code for this project.
2. Give the Lambda function a name of your choice ( try naming services with meaningful names ).
3. We'll be using a python code for this project. So, select **Python 3.10** as runtime to get no errors.



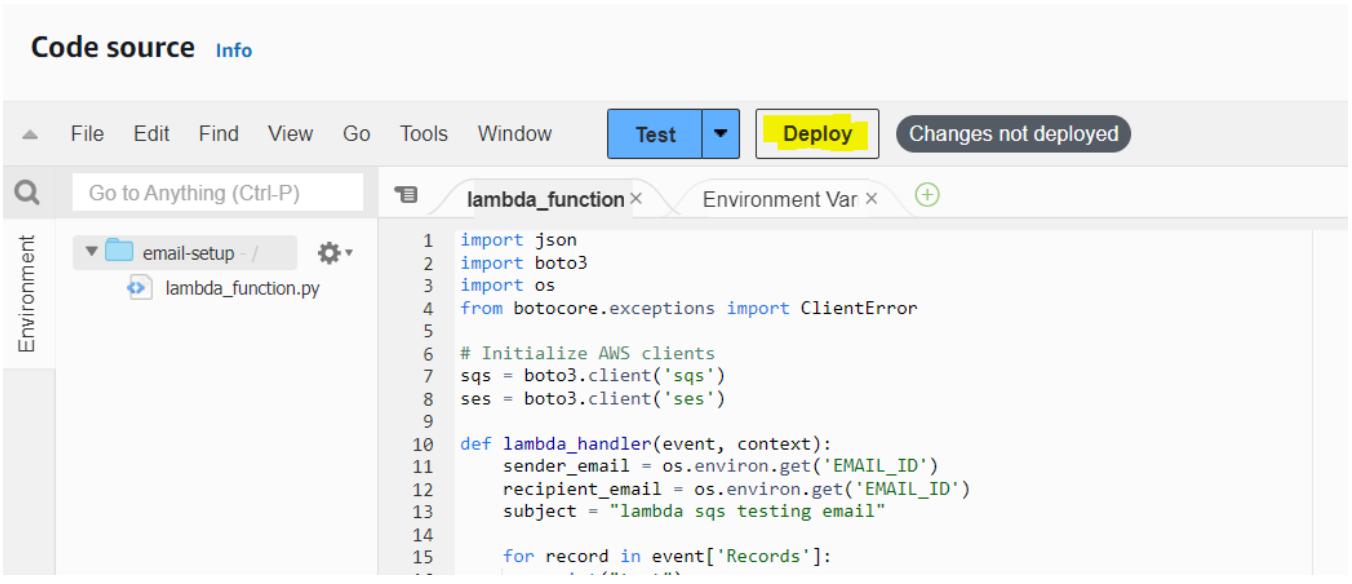

4. Click on "execution role" drop menu and select "Use an existing role" as we have created an IAM role just for this function.
5. In the drop down menu of "Existing role" select the IAM role that you created before.
6. Click on create and your Lambda function will be created.



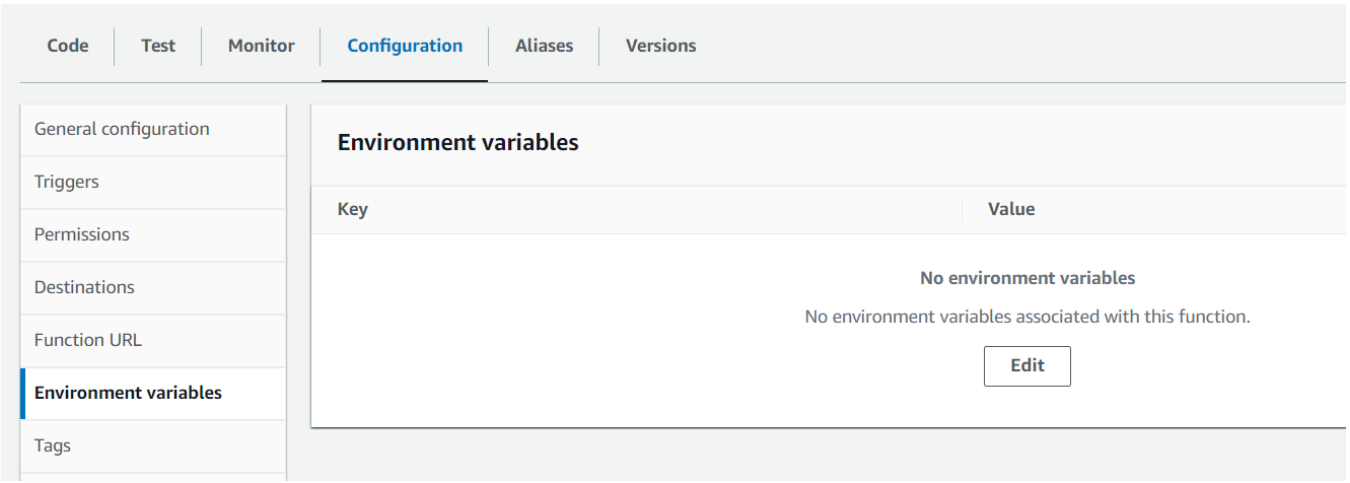

7. Scroll down on the Lambda service page, and paste the Python code in the Code Source.
8. Click on **Deploy** to save your code.

```
import json
import boto3
import os
from botocore.exceptions import ClientError
```

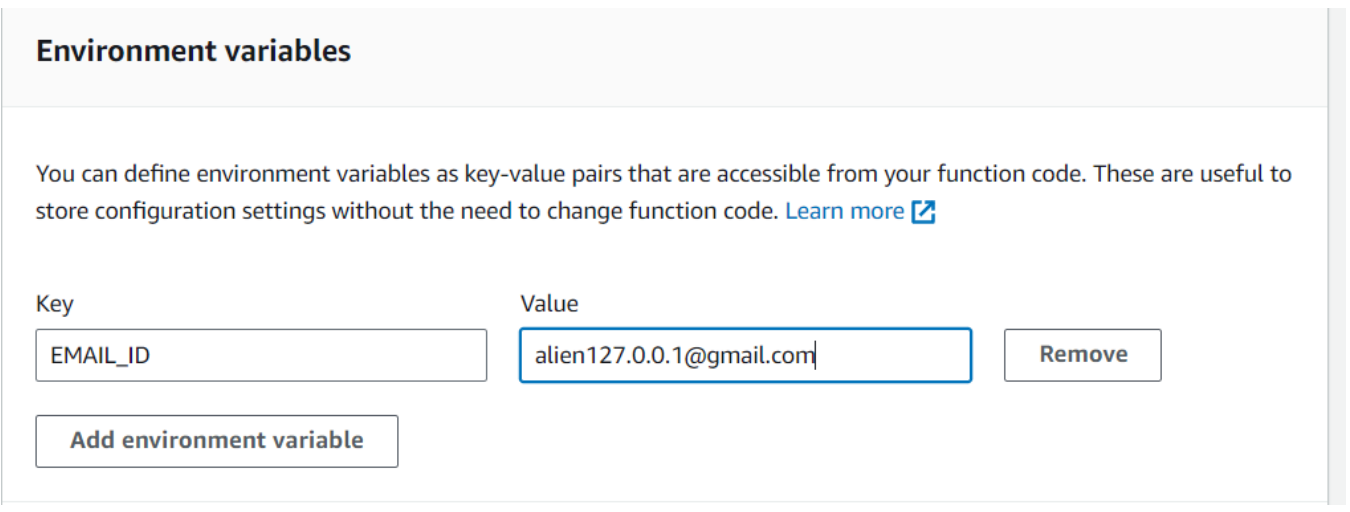
```
# Initialize AWS clients
sqs = boto3.client('sqs')
ses = boto3.client('ses')

def lambda_handler(event, context):
    sender_email = os.environ.get('EMAIL_ID')
    recipient_email = os.environ.get('EMAIL_ID')
    subject = "lambda sqs testing email"

    for record in event['Records']:
        print("test")
        email_body = record["body"]
        print(str(email_body))
        send_email(sender_email, recipient_email, subject, email_body)

def send_email(sender_email, recipient_email, subject, body):
    try:
        response = ses.send_email(
            Source=sender_email,
            Destination={
                'ToAddresses': [recipient_email]
            },
            Message={
                'Subject': {
                    'Data': subject
                },
                'Body': {
                    'Text': {
                        'Data': body
                    }
                }
            }
        )
        print(f"Email sent! Message ID: {response['MessageId']}")
    except ClientError as e:
        print(f"Error sending email: {e.response['Error']['Message']}")
```



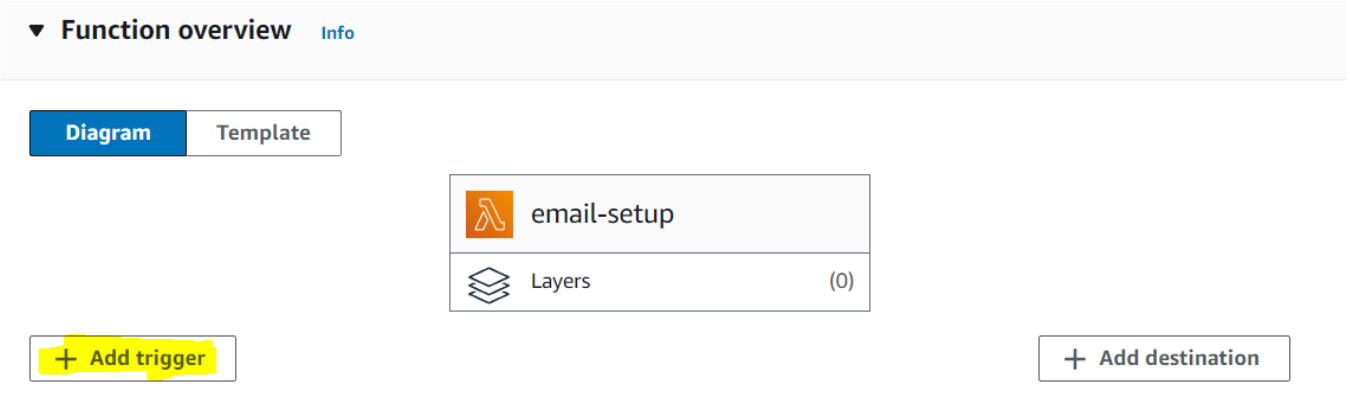

9. Now, we have to set the environment variable for the “**EMAIL_ID**” parameter.
10. On top of Code Source, click on **Configuration** and then select **Environment variables** from the left panel.



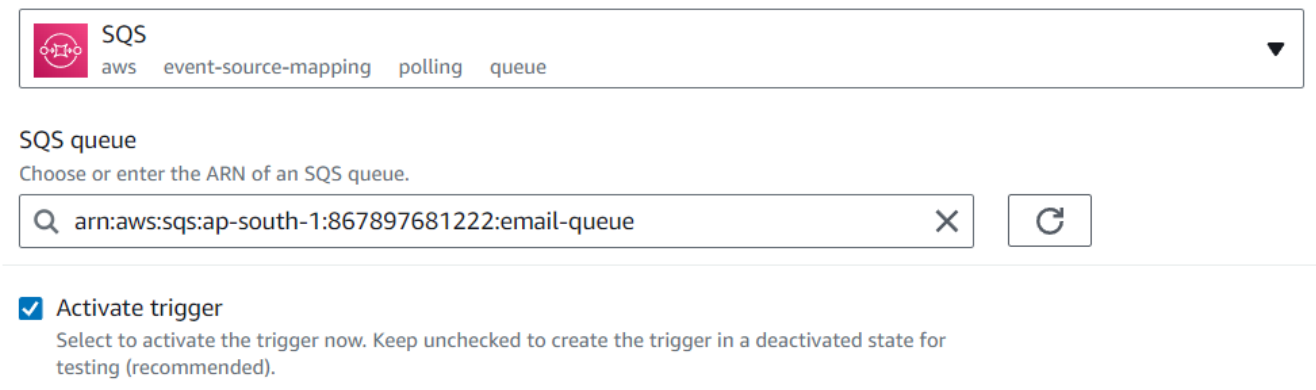

11. Select Edit and write **“EMAIL_ID”** as **Key** and **your email id** as **“VALUE”** and click create.




12. Now we are all done with setting up the Lambda function.
13. Click on **Add Trigge**r to add the SQS service as a trigger to send emails.

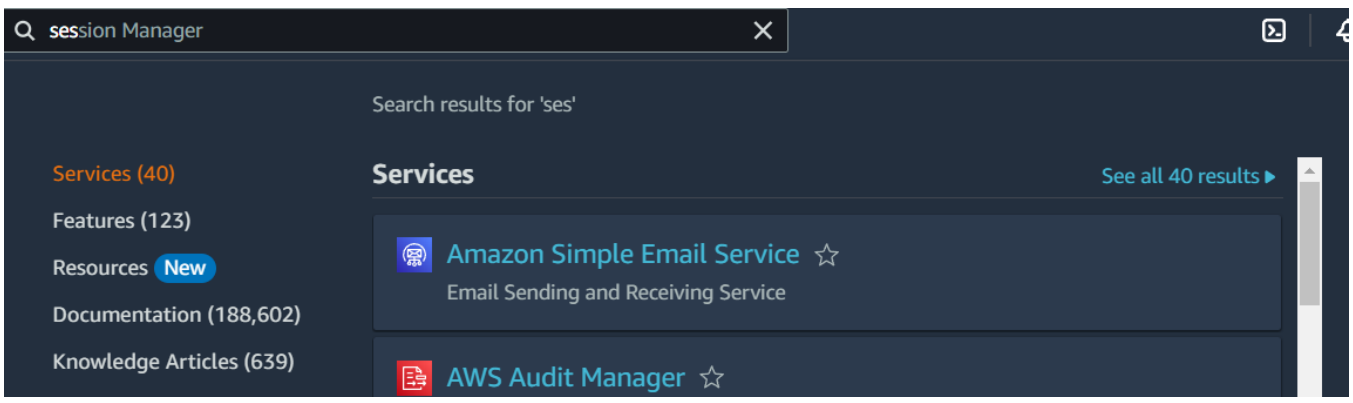



14. In the Trigger menu **search for SQS** and select it. In SQS Queue drop down menu, select the **Queue you created** before.

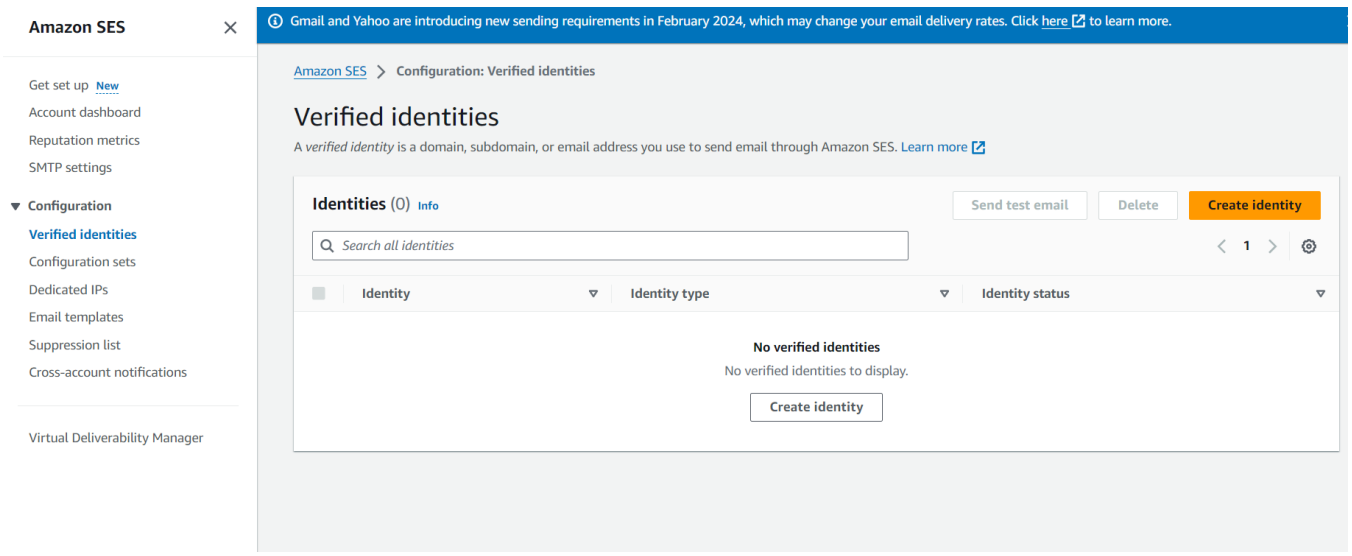



## SES Configuration

1. Select Amazon Simple Email Service from AWS dashboard.



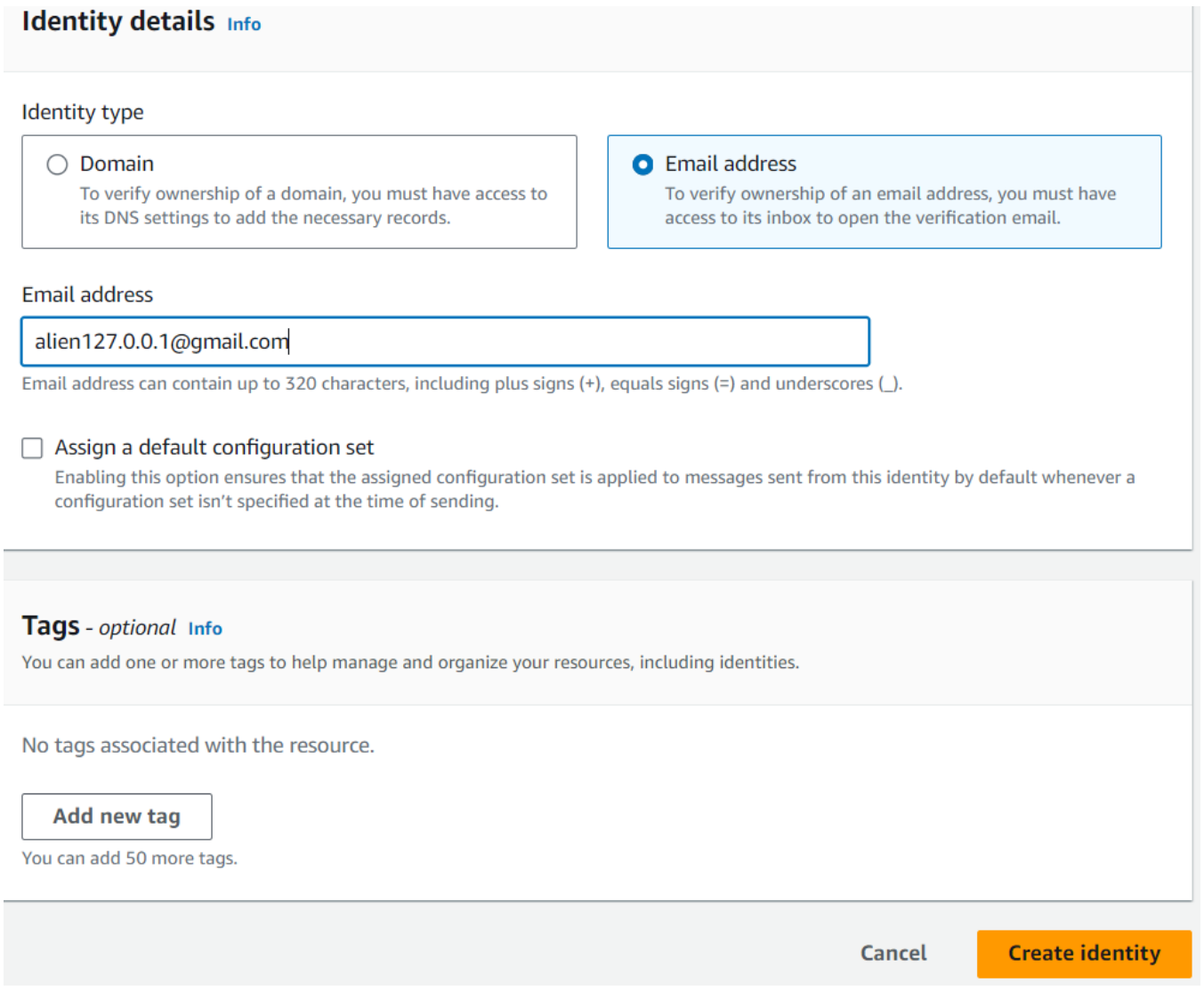

2. Now from the left panel, select **Verified identities** to add verify your email for the service.




3. For Identity type we'll choose Email address as we don't have an active domain right now. But if you are using it for your business or you've a domain then you can select Domain.
4. Write your email address in the Email field.

   **NOTE: THIS EMAIL ID SHOULD BE SAME AS THE ENV VARIABLE THAT YOU SET BEFORE.**

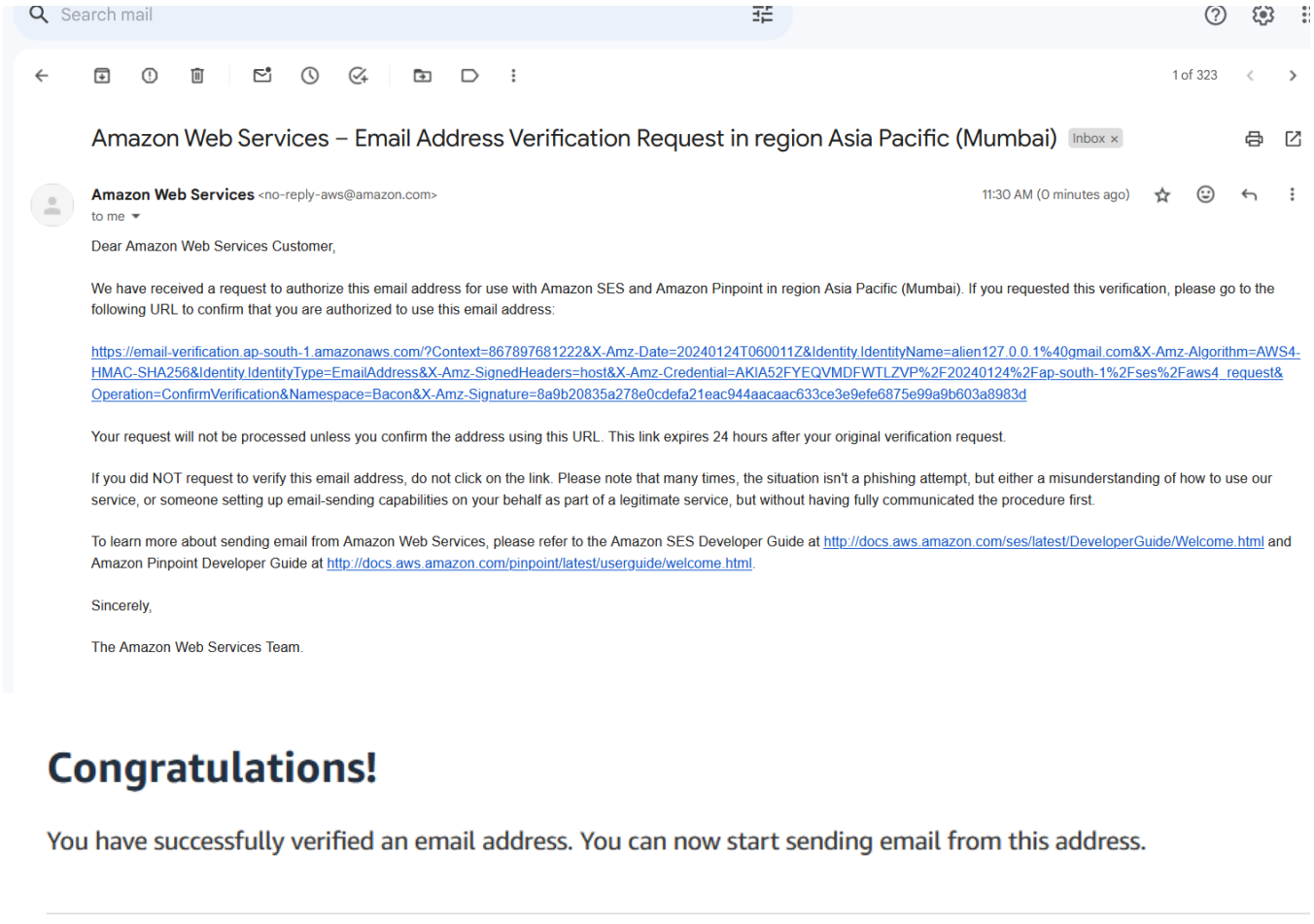
5. Click on Create Identity, and you will be be prompted to a verification page.




6. You'll receive a verification email on the Email ID you've put.

   **NOTE: CHECK YOUR SPAM FOLDER IF YOU DON'T SEE IT IN THE INBOX.**

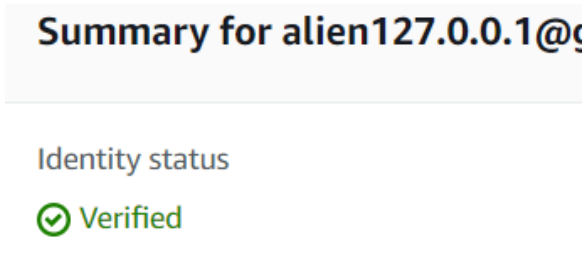
7. Click on the link for the verification and close the tab after you see the Congratulations.





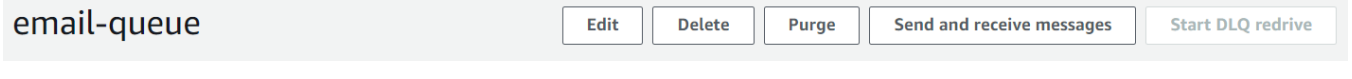


8. Go back to the verification page, refresh it and you'll see Identity status as verified now.

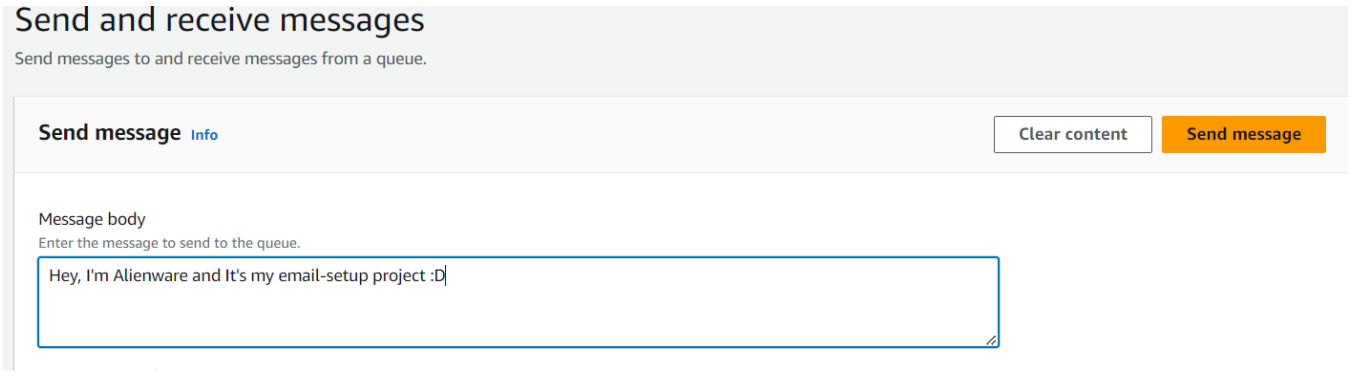



## We are done with all the configuration. Now let's test it out!

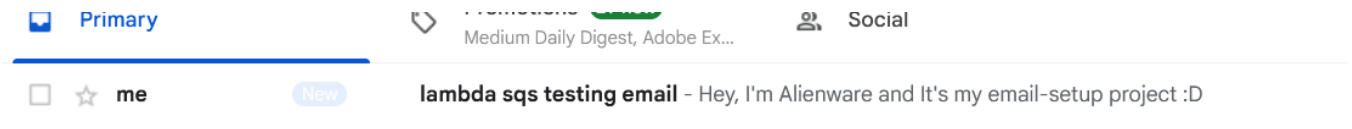
1. Navigate to SQS service, select the Queue that you created before.

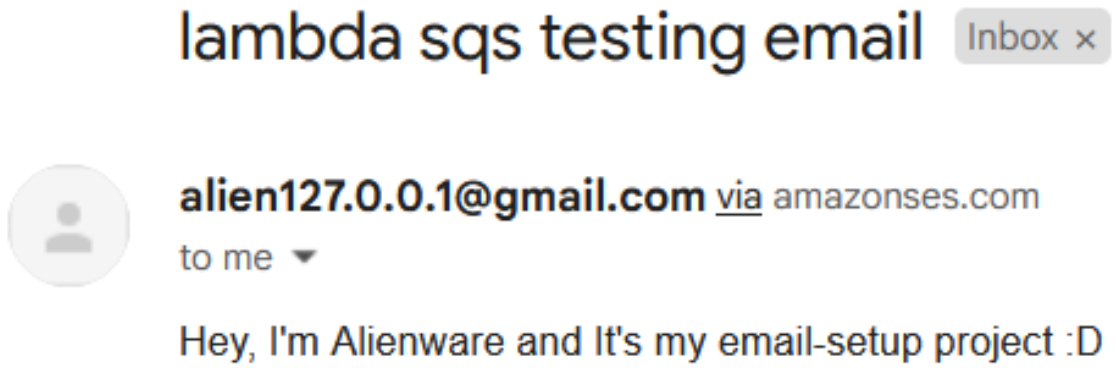



2. Now in the Queue you'll see a **Send and Receive messages** button, click on that.
3. Write the message you want to send in the message body and click on **Send message.**



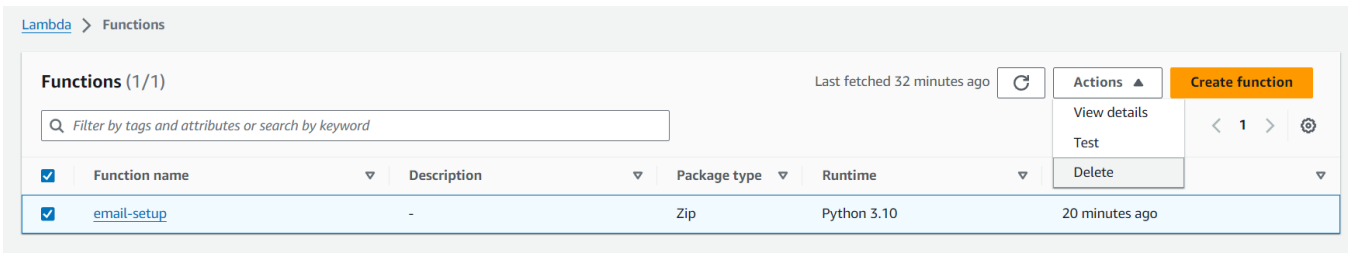

Open your Gmail account and there you'll see an email sent to you via **AMAZONSES** with the message you wrote in the message body.

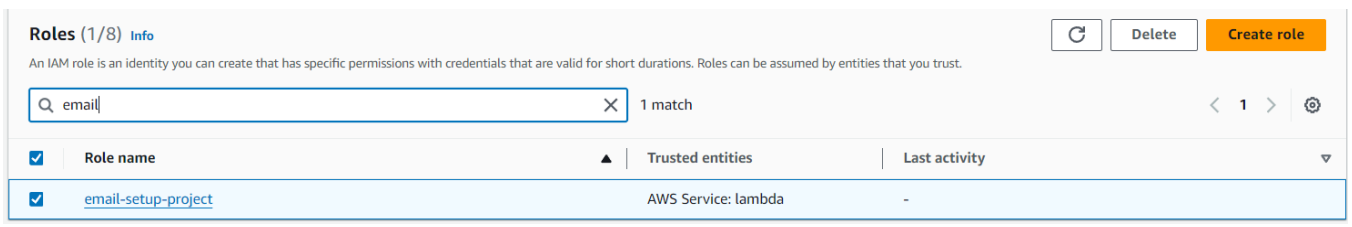


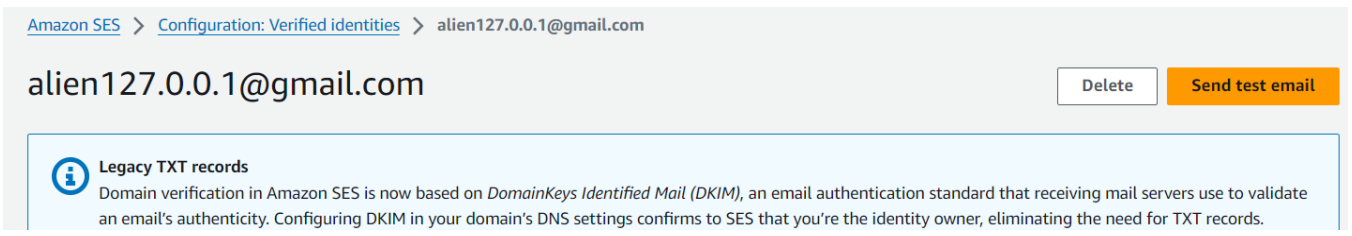


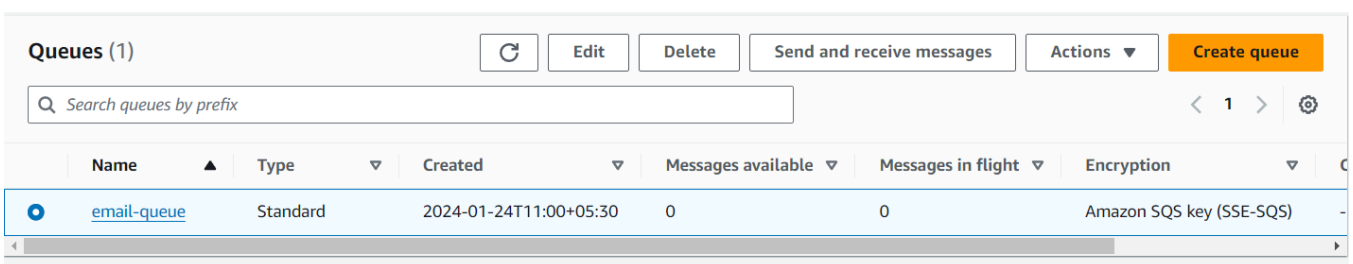

## Cleaning Up

Go to all the services that you have used for this project ( SQS, SES, Lambda, IAM roles ). Select the new entries that you've made and delete them accordingly to don't get any charges.




## THANK YOU CLOUD PEOPLE!!! ☁️💻

**If you find this blog helpful, please do like it and subscribe to my page for more such content <3**

Connect with me on Linkedin : https://www.linkedin.com/in/pawanngambhir/