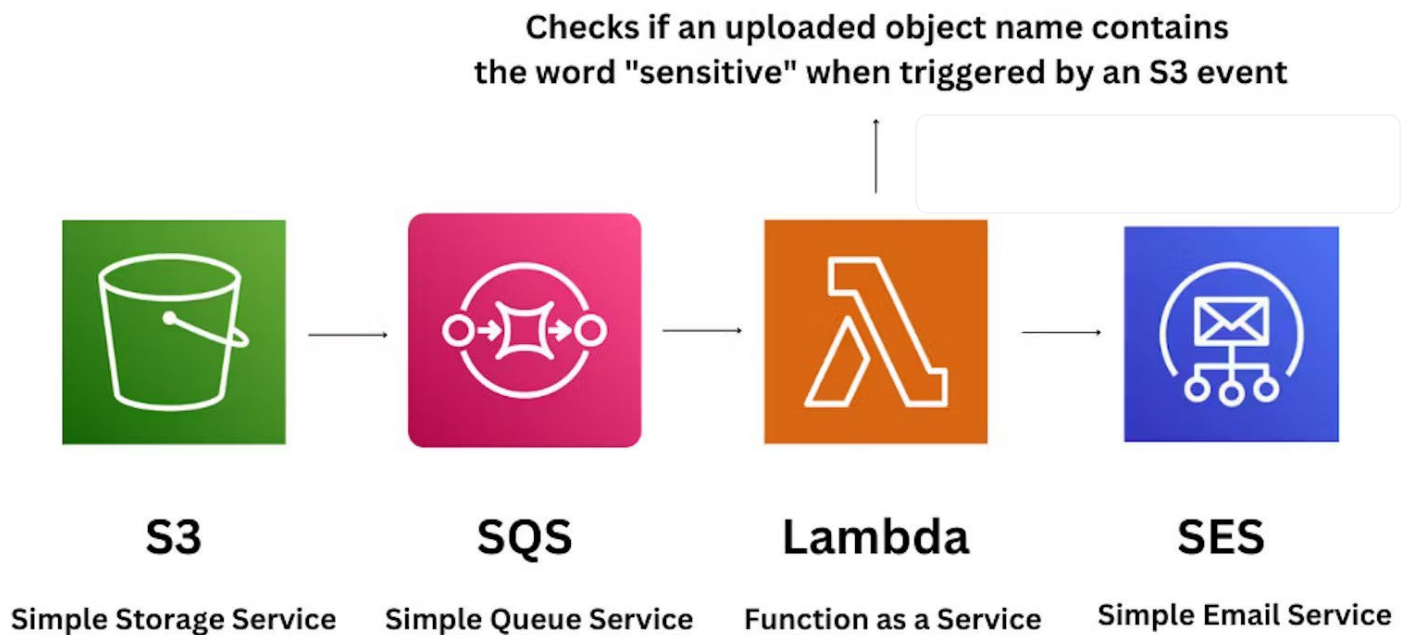


[Type here]



In this project, I will be creating an S3 bucket, Queue, and Lambda function. Whenever an object is uploaded in the S3 bucket, an S3 event will trigger the queue and the lambda function will run which will check if the object name contains the word "sensitive" If so it will send an email notification to the team saying some sensitive file is added in the bucket.

**Prerequisite:**

AWS account and familiarity with AWS services

Let's first understand these AWS services:

**Simple Storage Service(S3):** A highly scalable object storage service that allows you to store and retrieve large amounts of data. It provides durability, availability, and security for your data and can be used for various purposes, such as backup, static website hosting, and content distribution.



[Type here]

[Type here]

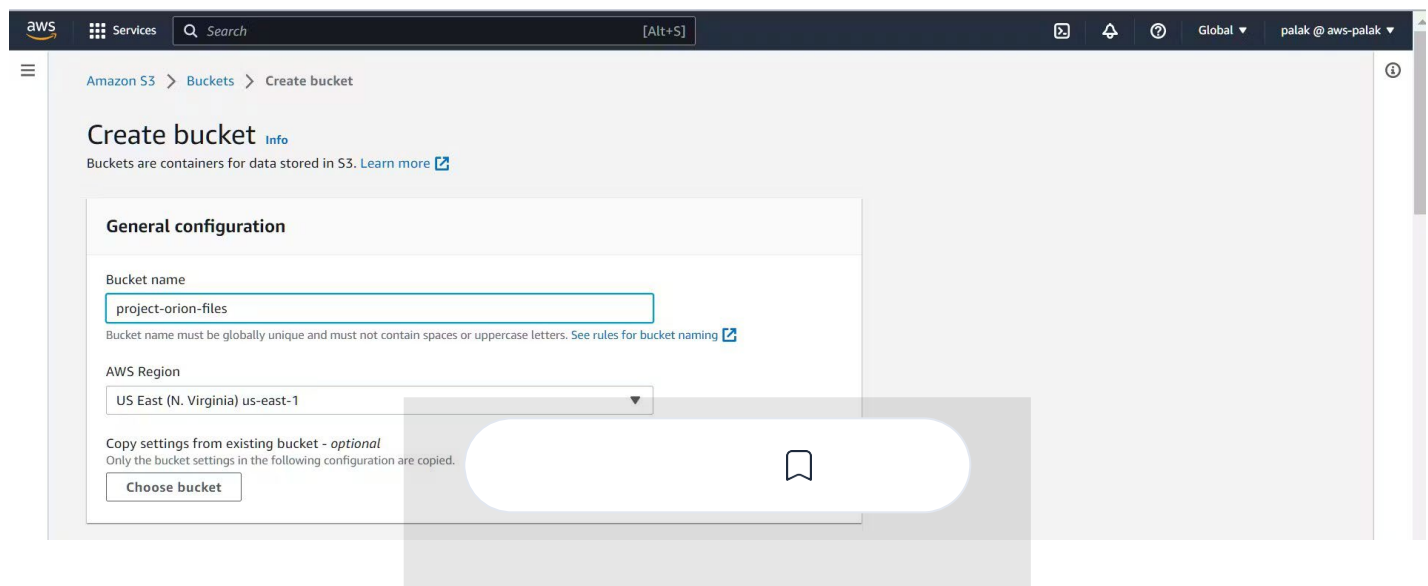
Simple Queue Service(SQS): A fully managed message queuing service that enables you to decouple and scale microservices, distributed systems, and serverless applications. It allows you to send, store, and receive messages between software components.

Function as a Service(Lambda): A computing service that lets you run code without provisioning or managing servers. It allows you to execute your code in response to events, such as changes in data, HTTP requests, or scheduled intervals.

Simple Email Service(SES): A fully managed service provided by AWS that enables you to send and receive email messages. It handles the underlying infrastructure and scaling aspects, allowing you to focus on sending and managing emails without the need to provision or manage servers.

## Step 1: Create an S3 bucket

Go to the AWS console and search for "S3" in services. Click Create bucket and give a unique name to the bucket → choose the AWS region where you want your bucket to reside. Keep other settings default and click Create bucket.

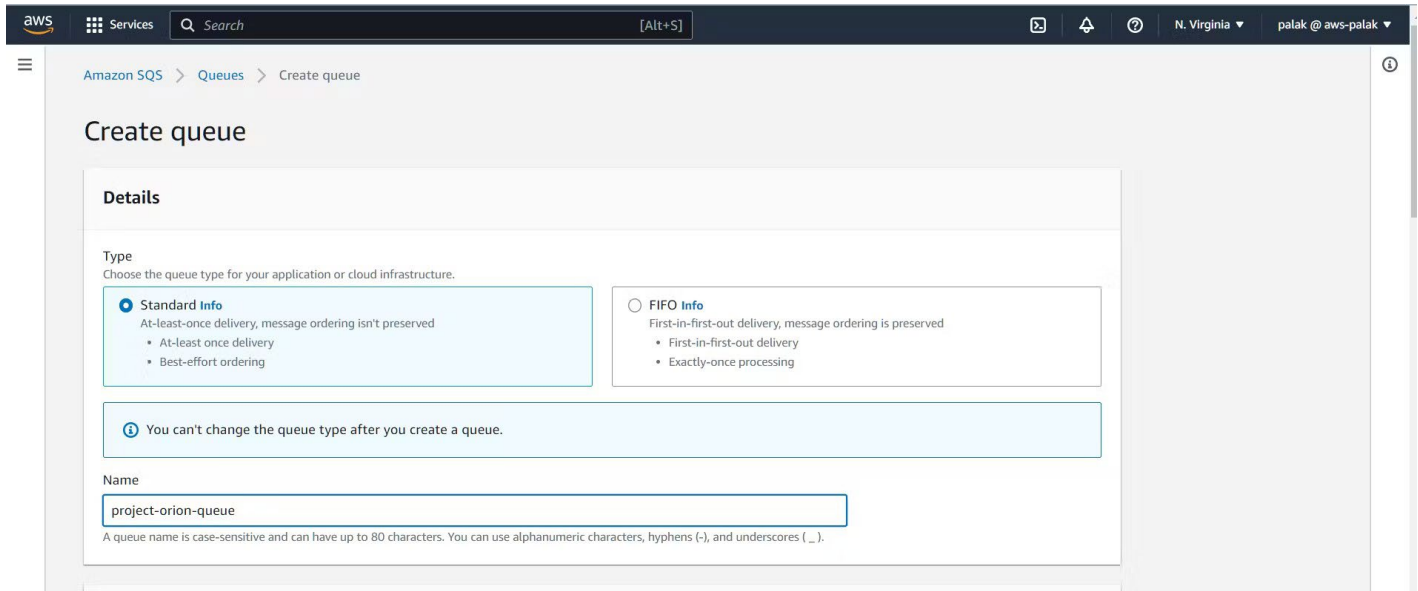


The screenshot shows the AWS console interface for creating a new S3 bucket. The top navigation bar includes the AWS logo, a search bar, and user information. The main content area is titled 'Create bucket' and includes a sub-header 'General configuration'. Under this section, there is a 'Bucket name' field with the value 'project-orion-files', an 'AWS Region' dropdown menu set to 'US East (N. Virginia) us-east-1', and a 'Choose bucket' button. A note indicates that settings from an existing bucket can be copied. The page also features a sidebar with navigation links and a bottom status bar.

[Type here]

## Step 2: Create SQS

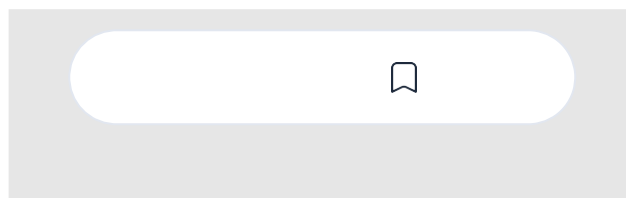
Go to the AWS console and search for "SQS" in services. Click Create queue → Choose the queue type as Standard → Give a name to  settings default and click Create queue.

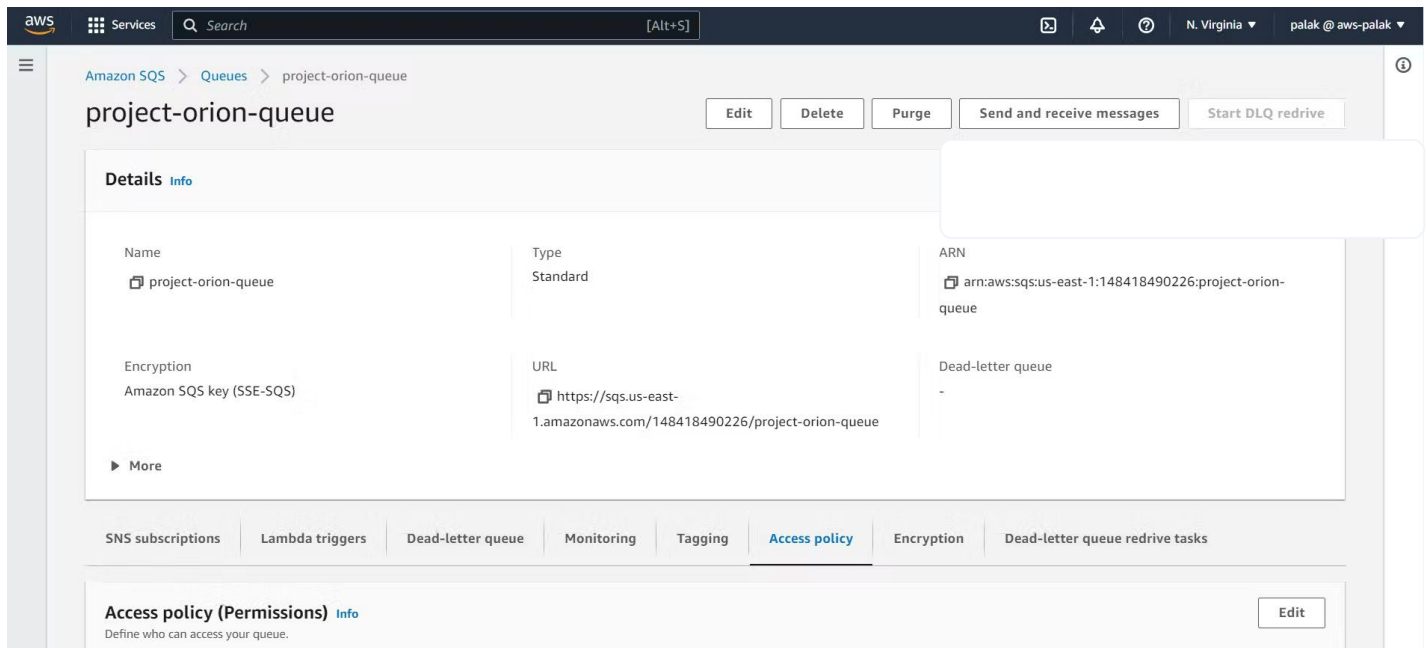


Now attach the access policy to the queue so that it can communicate with the S3 bucket when the S3 event is triggered.

## Step 3: Attach policy to SQS

Go to the Queue that we have created in the previous step and select access policy. Click edit to update the permissions.



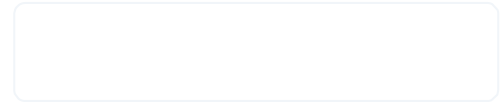


Take the Resource arn from the existing policy shown in your console. Go to the bucket that we have created and in the properties section you will get bucket arn, copy it and paste it into `aws:SourceArn` in the below policy and save.

COPY

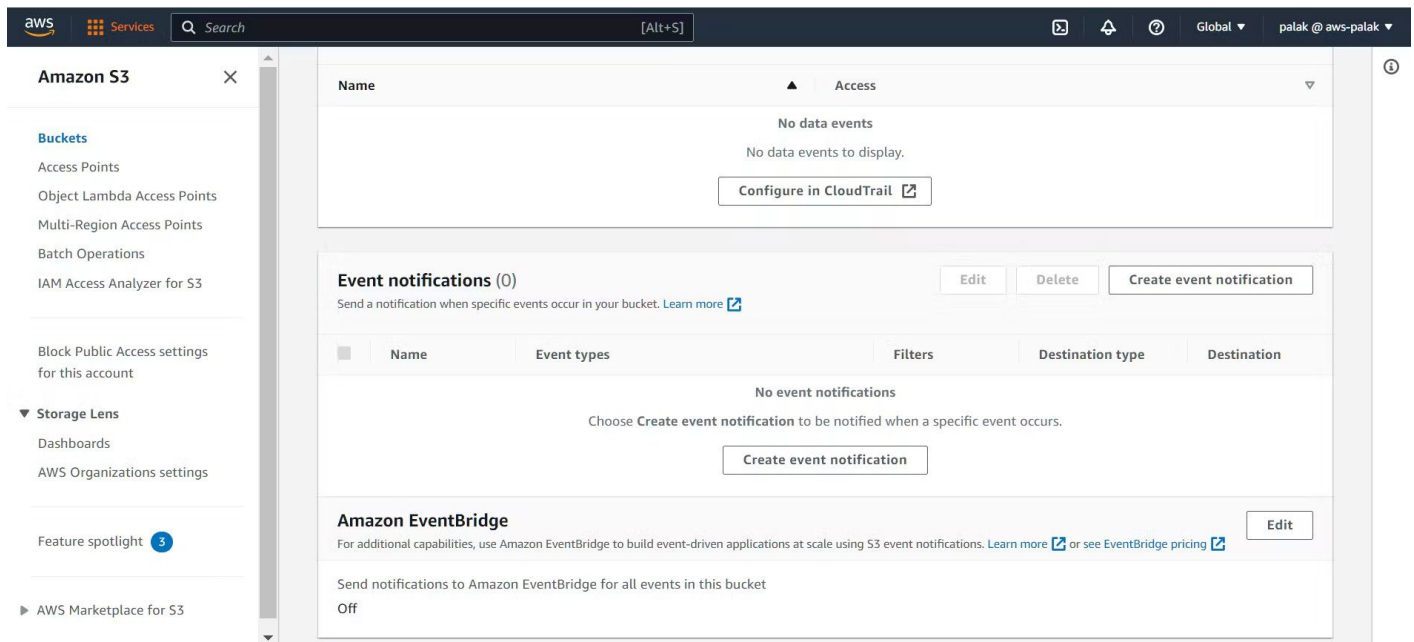
```
{
  "Version": "2012-10-17",
  "Id": "Policy1684249599450",
  "Statement": [
    {
      "Sid": "Stmt1684249581034",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:us-east-1:148418490226:project-orion-queue",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": "arn:aws:s3:::project-orion-files"
        }
      }
    }
  ]
}
```

```
}  
]  
}
```



## Step 4: Create S3 Event

Go to your bucket and select Properties, scroll down to Event notifications and click Create event notification. Give a name to your event then checkmark All object create events.



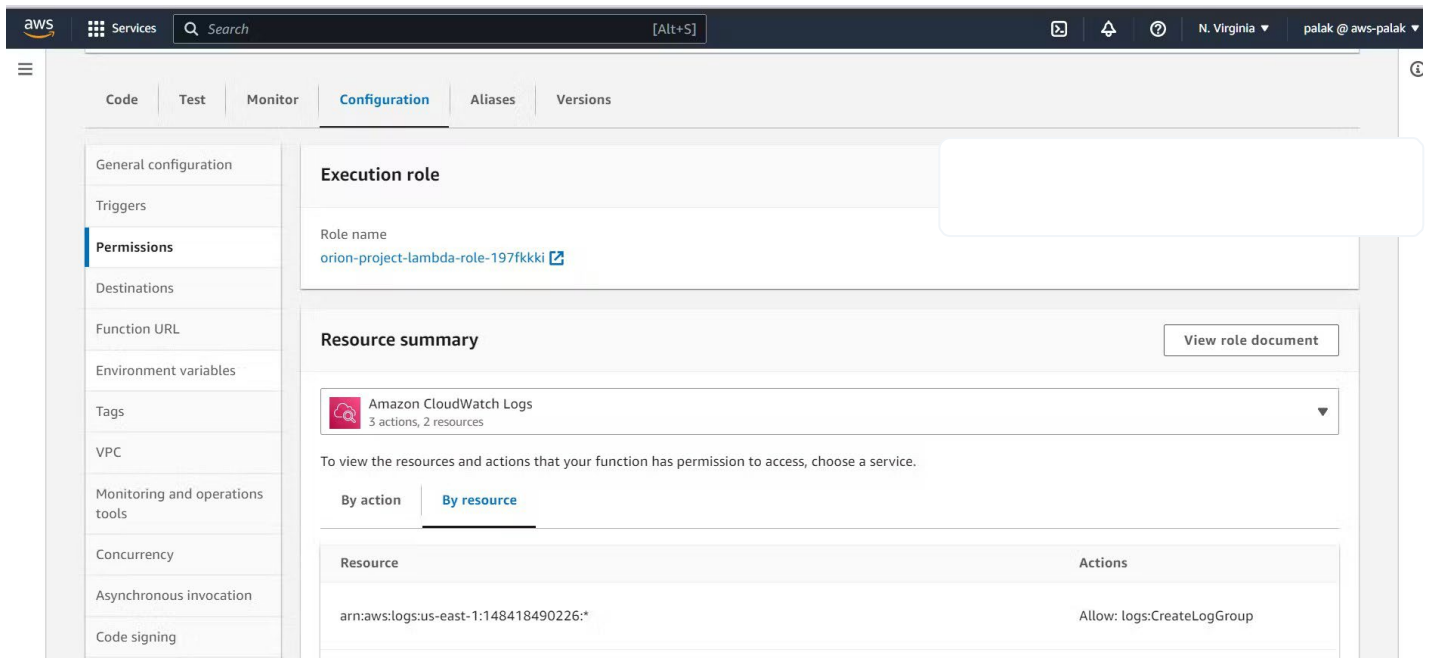
Select destination as SQS queue → Select the queue that we have created in previous steps → click save changes.

The screenshot shows the AWS S3 console interface. At the top, there's a navigation bar with the AWS logo, 'Services' link, a search bar, and user information 'palak @ aws-palak'. The main content area is titled 'Destination'. It contains a blue informational box stating: 'Before Amazon S3 can publish messages to a destination, you must grant the Amazon S3 principal the necessary permissions to call the relevant API to publish messages to an SNS topic, an SQS queue, or a Lambda function. [Learn more](#)'. Below this, the 'Destination' section asks to 'Choose a destination to publish the event. [Learn more](#)'. There are three radio button options: 'Lambda function' (Run a Lambda function script based on S3 events.), 'SNS topic' (Fanout messages to systems for parallel processing or directly to people.), and 'SQS queue' (Send notifications to an SQS queue to be read by a server.). The 'SQS queue' option is selected. Under 'Specify SQS queue', there are two radio button options: 'Choose from your SQS queues' (selected) and 'Enter SQS queue ARN'. Below this is a dropdown menu labeled 'SQS queue' with 'project-orion-queue' selected. At the bottom right of the configuration panel are 'Cancel' and 'Save changes' buttons.

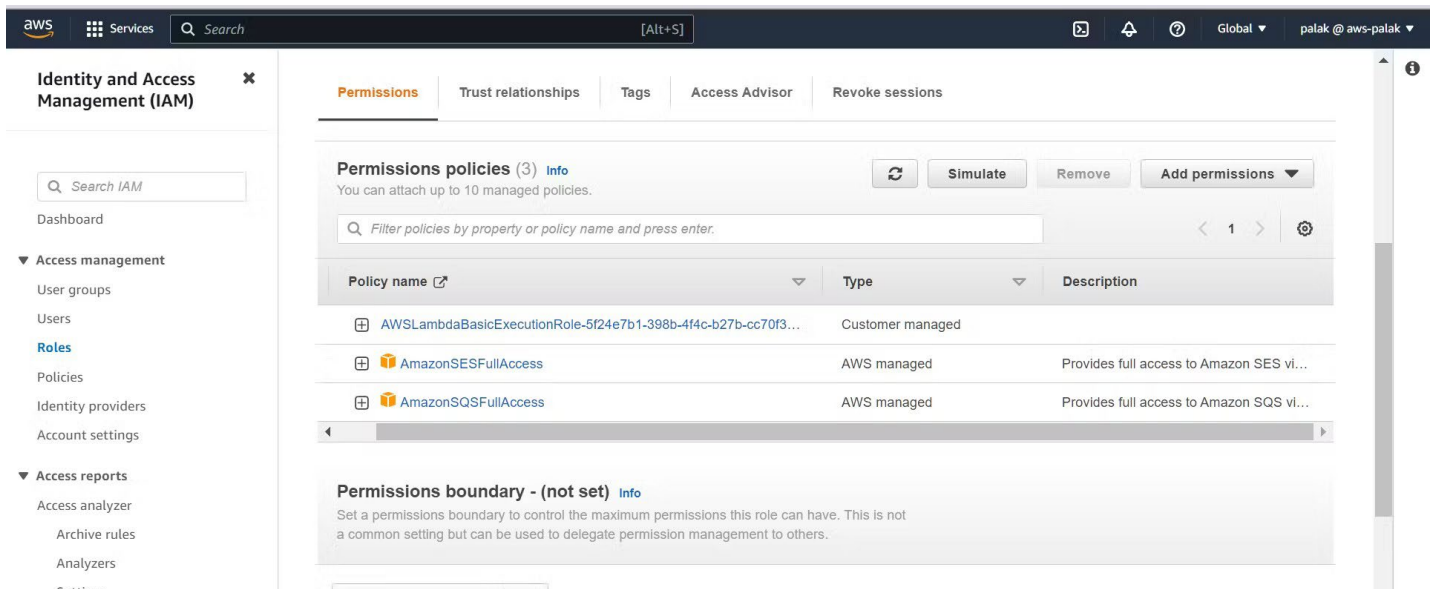
## Step 5: Create Lambda function

Go to AWS services and search for Lambda. Select Lambda and click Create function. Select Author from Scratch → Give a name to function → Select runtime as Python3 → Click Create function.

Now go to Configuration → Permissions → click Role name

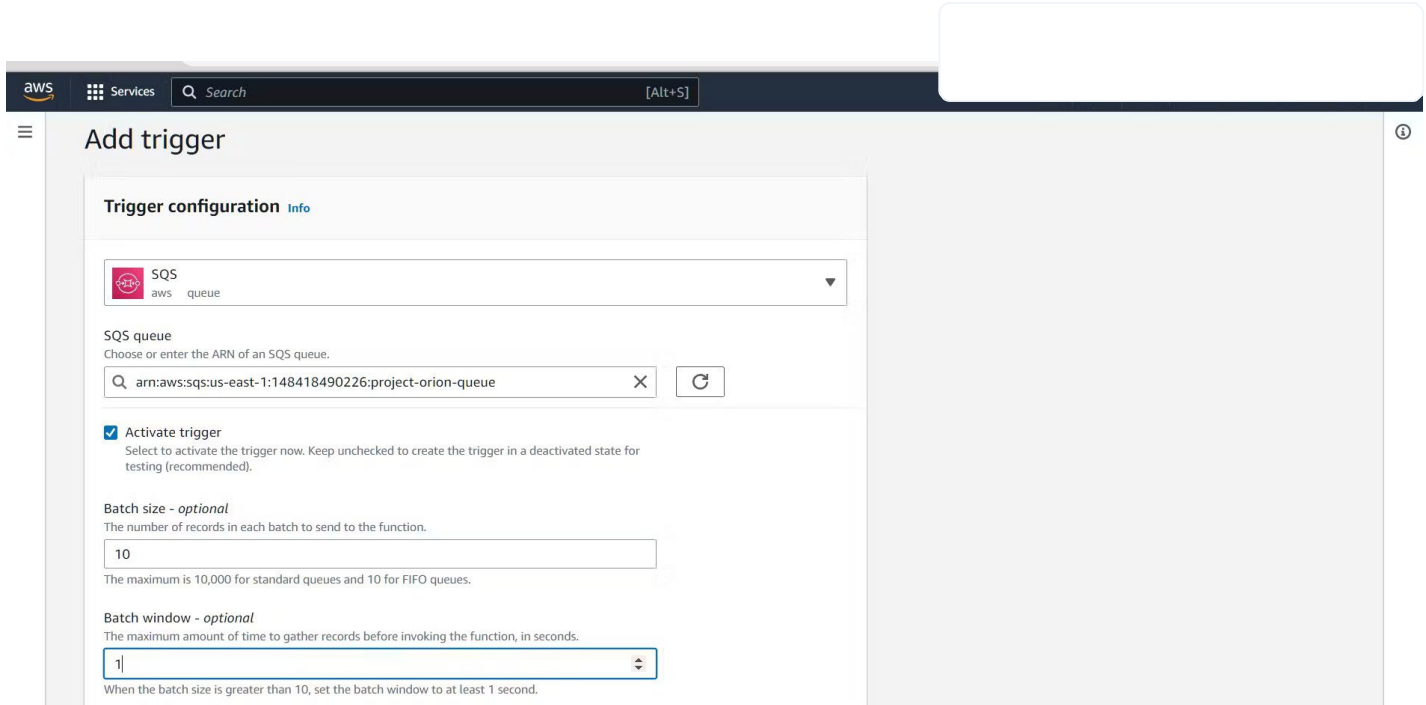


Add permissions to this lambda role so that it can communicate with SQS and SES services. Click Add permissions → Attach policies → Search for AmazonSESFullAccess and AmazonSQSFullAccess and add this policy to the Lambda.



## Step 6: Add Lambda Trigger

We want this function to run when SQS has something to process. Click Add trigger in the Lambda function and select SQS as the trigger source. Select the SQS name that we have created. Keep the Batch window as 1 then click Add.



The screenshot shows the 'Add trigger' configuration page in the AWS Lambda console. The 'Trigger configuration' section is active, showing a dropdown menu with 'SQS' selected. Below this, the 'SQS queue' field is populated with the ARN 'arn:aws:sqs:us-east-1:148418490226:project-orion-queue'. The 'Activate trigger' checkbox is checked. The 'Batch size - optional' field is set to '10', and the 'Batch window - optional' field is set to '1'. The page includes a search bar at the top and a sidebar on the left.

## Step 7: Create SES and verify the Email

Go to AWS services and search for SES. Select SES and click Verified Identities → Create identity → Email address → Enter email address → Click Create Identity.

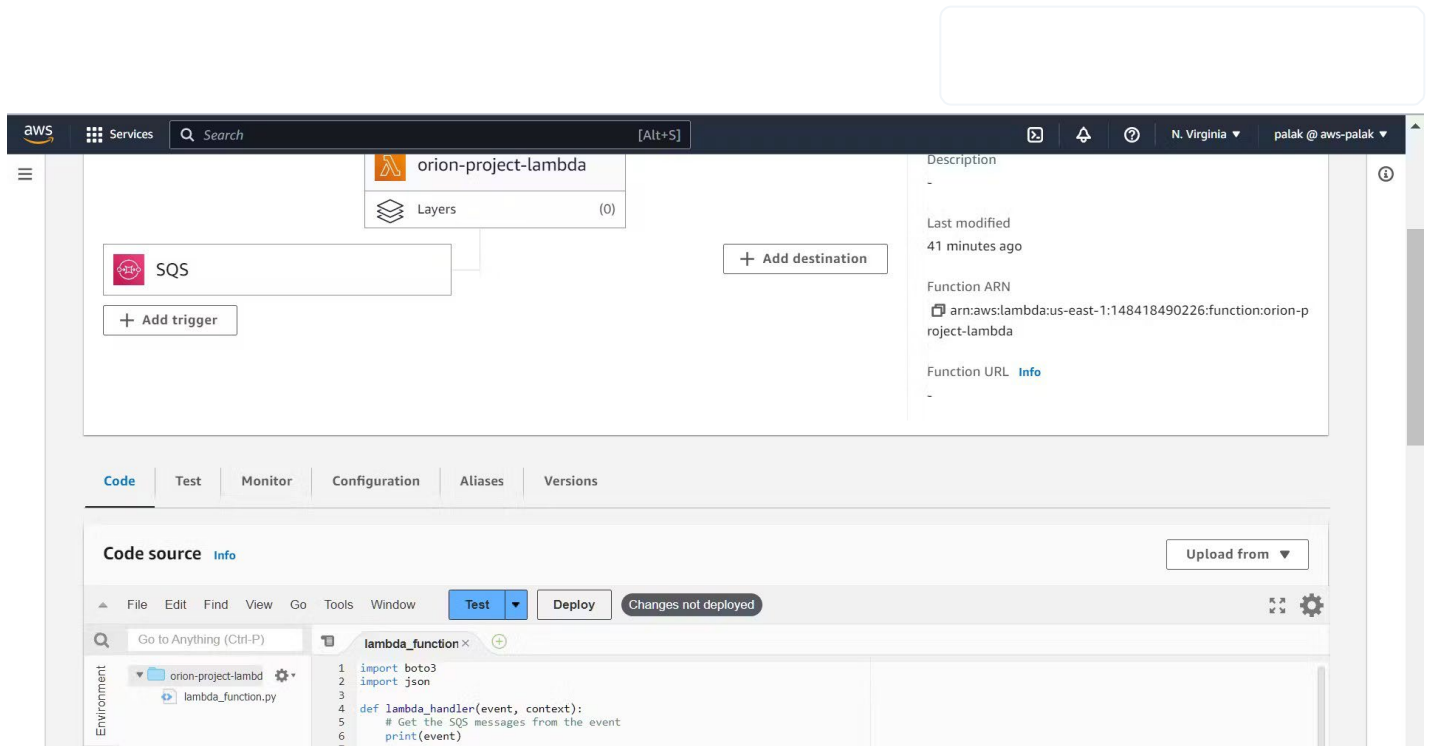
Check your mailbox and verify the email address. Create an identity for both the sender's email address and the receiver's email address. Once the identity is verified move to the next step.

## Step 8: Deploy the Lambda function

Go to Lambda and add the Python code which will filter out the object uploaded in the S3 bucket and send the email to the receiver if the object name contains the



word "sensitive"



After adding Python code in the Lambda function, click Deploy. Now to test this, upload some objects in the S3 bucket. Go to monitor and click view CloudWatchLogs → Select the latest Log Stream. Check if the email is sent to the receiver and the queue is empty.

As soon as you upload a file in the S3 bucket and the name of the file contains the word "sensitive" an email will be sent.

