**Comprehensive Guide to AWS Lambda with S3 Trigger**

**Introduction to AWS Lambda**

AWS Lambda is a serverless compute service that enables you to run code without provisioning or managing servers. With Lambda, you can focus on writing your application logic while AWS handles the infrastructure, scaling, and maintenance.

**Key Features of AWS Lambda**

* **Serverless:** No need to manage servers.
* **Event-Driven:** Automatically triggers in response to events.
* **Cost-Efficient:** Pay only for the compute time you use.
* **Scalable:** Automatically handles scaling.

**Why Use AWS Lambda?**

**1. Simplified Architecture**

Eliminate the need to manage underlying infrastructure, reducing complexity.

**2. Cost Savings**

Only pay for the execution time of your code and not for idle resources.

**3. Event-Driven Execution**

Trigger Lambda functions based on various AWS services or custom events.

**4. Versatile Use Cases**

Suitable for data processing, API backends, monitoring, and automation.

**Setting Up an AWS Lambda Function to Trigger on S3 Events**

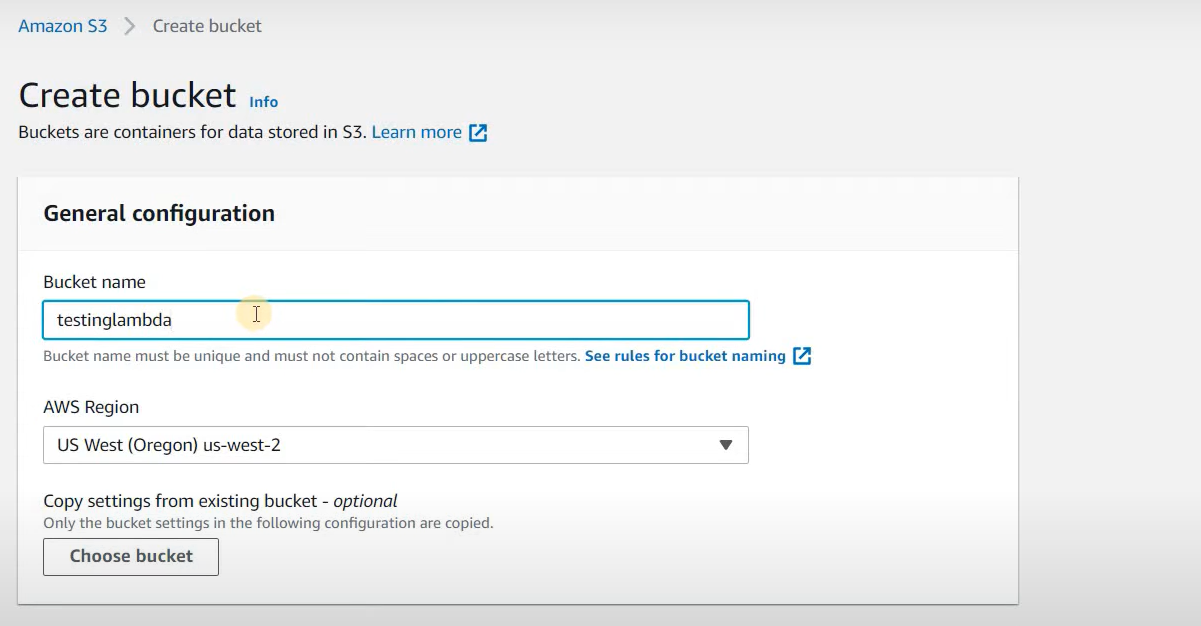
**Use Case Overview**

In this project, we configure AWS Lambda to send an email notification when a file is uploaded to an Amazon S3 bucket. This involves creating a Lambda function, configuring S3 event triggers, and setting up Amazon Simple Email Service (SES).

**Steps to Implement**

**Step 1: Create an S3 Bucket**

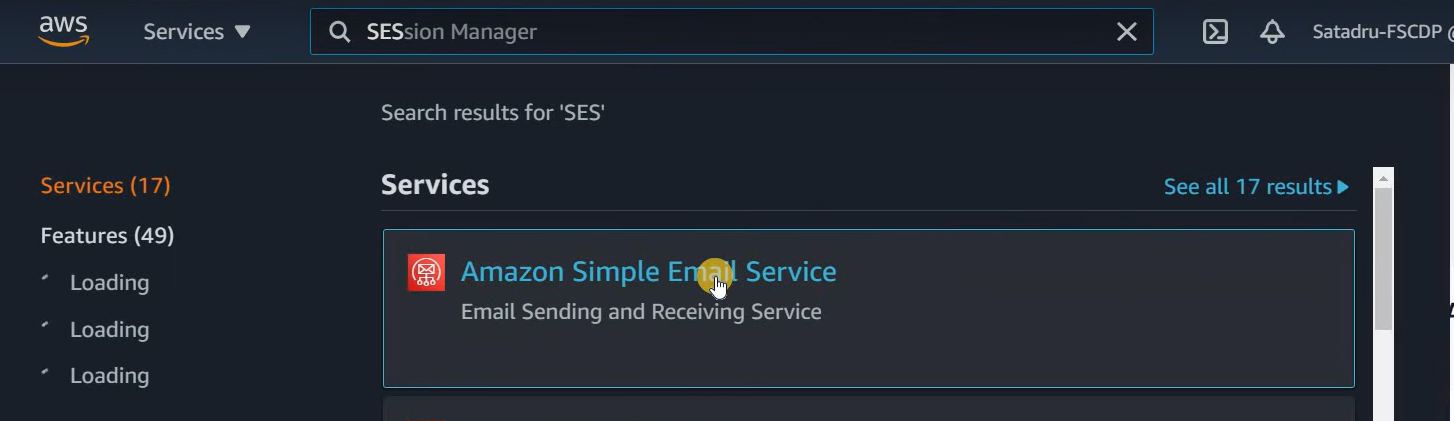
1. Log in to the AWS Management Console.
2. Navigate to the **S3 Dashboard**.

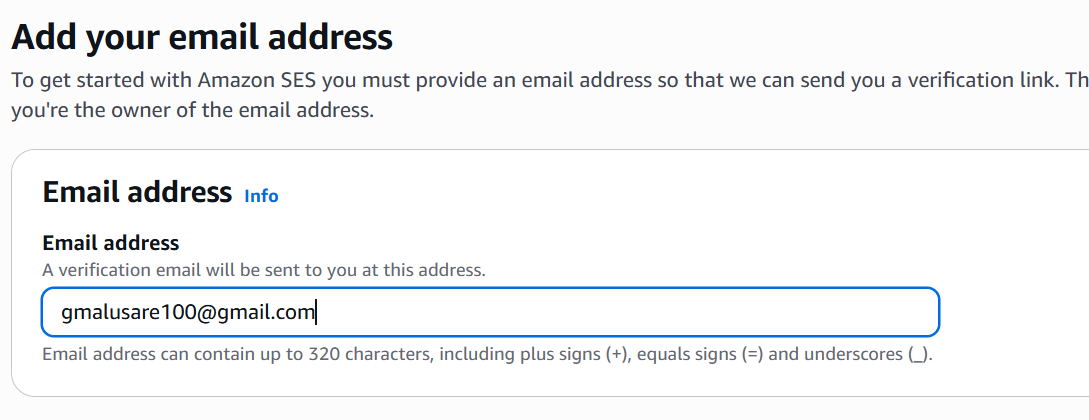


1. Click **Create Bucket** and specify a name and region.
2. Configure permissions as needed, ensuring public access is restricted.

**Step 2: Set Up Amazon SES**

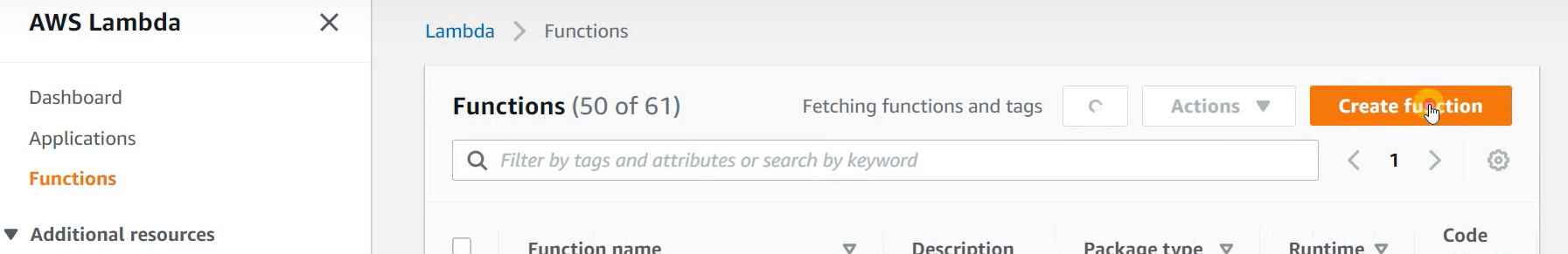
1. Navigate to the **SES Console**.



1. Verify your email address:
   * Go to **Email Addresses** > **Verify a New Email Address**.
   * Follow the instructions in the verification email.
2. (Optional) Move out of the SES sandbox environment to enable production-level access.

**Step 3: Create an AWS Lambda Function**

1. Navigate to the **Lambda Console**.
2. Click **Create Function** and select **Author from scratch**.

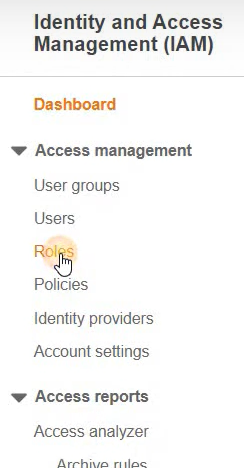


1. Provide a name (lambdasestesting) and choose the runtime ( Python ).

A screenshot of a computer

AI-generated content may be incorrect.

1. Create or assign an IAM role with the necessary permissions:



* + Access to S3.
  + Access to SES.

**Step 4: Write the Lambda Function Code**

Below is an example Python code:

import boto3

import json

def lambda\_handler(event, context):

s3 = boto3.client('s3')

ses = boto3.client('ses')

# Extract bucket name and object key from the event

bucket\_name = event['Records'][0]['s3']['bucket']['name']

object\_key = event['Records'][0]['s3']['object']['key']

# Construct the email

subject = "New File Uploaded to S3"

body = f"A new file has been uploaded to your S3 bucket:\nBucket: {bucket\_name}\nFile: {object\_key}"

# Send email via SES

response = ses.send\_email(

Source='gmalusare100@example.com',

Destination={

'ToAddresses': ['gmalusare100@example.com']

},

Message={

'Subject': {'Data': subject},

'Body': {'Text': {'Data': body}}

}

)

return {

'statusCode': 200,

'body': json.dumps('Email sent successfully!')

}

**Step 5: Configure the S3 Trigger**

1. Navigate to your S3 bucket in the **S3 Console**.A screenshot of a computer

   AI-generated content may be incorrect.
2. Go to **Properties** > **Event Notifications**.
3. Add a new event notification:
   * Specify a name.
   * Select event types (e.g., PUT for new uploads).
   * Choose the Lambda function created in Step 3.

**Step 6: Test the Setup**

1. Upload a file to the S3 bucket.
2. Check the Lambda logs in CloudWatch to verify execution.
3. Confirm receipt of the email notification.

**Best Practices**

**1. Security**

* Use least privilege when assigning IAM roles.
* Verify email addresses in SES before using them.

**2. Monitoring and Logging**

* Enable CloudWatch logging for the Lambda function.
* Monitor SES email sending limits.

**3. Error Handling**

* Implement retry logic in the Lambda function.
* Set up Dead Letter Queues (DLQs) for unprocessed events.

**Conclusion**

AWS Lambda, combined with S3 and SES, offers a powerful and serverless way to automate workflows. This setup demonstrates how to trigger email notifications on file uploads, showcasing the flexibility and efficiency of AWS services.