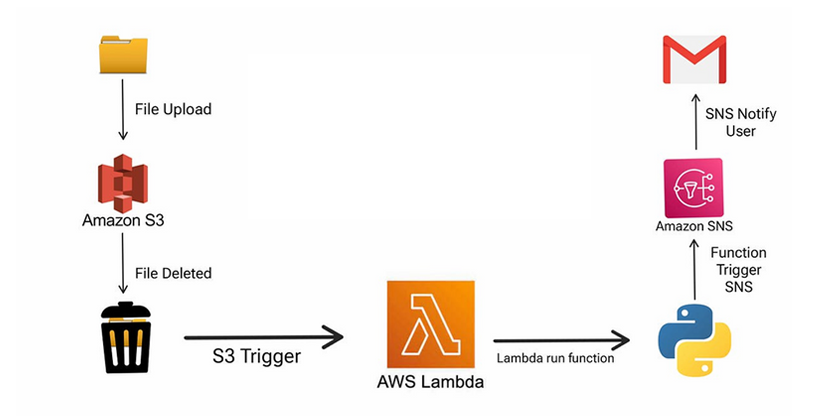
**Lab 8: Working with Event Driven Architectures**

**Objective:**

Learn to build event-driven architectures using AWS Lambda, SES, SNS, and SQS for automated, scalable workflows.

**Architecture diagram:**



Steps:

* Set up S3 to trigger a Lambda function on object addition.
* Use Lambda and SES for email notifications.
* Configure Lambda to send emails based on SQS messages using SNS.

**Part 1: Set up S3 to Trigger a Lambda Function**

**Step 1: Create an S3 Bucket**

1. Go to the **AWS Management Console** > **S3**.
2. Click **Create bucket** and configure:
   * Bucket Name: my-trigger-bucket (unique name).
   * Region: Choose your region.
3. Leave other settings default or configure as needed.

**Step 2: Create a Lambda Function**

1. Go to the **AWS Lambda Console**.
2. Click **Create function**:
   * **Function name**: s3-event-handler.
   * **Runtime**: Choose Python, Node.js, or your preferred language.
   * **Execution Role**:
     + Create a new role with basic Lambda permissions or use an existing one.
3. Write a simple handler to process S3 events. Example (Python):
4. import json
5. def lambda\_handler(event, context):
6. print("Event:", json.dumps(event, indent=2))
7. return {"statusCode": 200, "body": "S3 Event Processed"}

**Step 3: Add an S3 Trigger**

1. In the Lambda function's **Designer** section, click **+Add trigger**.
2. Select **S3** and configure:
   * **Bucket**: my-trigger-bucket.
   * **Event type**: PUT (object creation).
   * Save changes.

**Part 2: Use Lambda and SES for Email Notifications**

**Step 1: Set Up SES (Simple Email Service)**

1. Navigate to the **SES Console** > **Email Addresses**.
2. Verify an email address:
   * Enter an email to send notifications from.
   * Click **Verify** and confirm via the verification email.

**Step 2: Update Lambda Function to Send Emails**

Modify the Lambda function to send emails using SES. Example (Python with Boto3):

import boto3

import json

ses\_client = boto3.client('ses')

def lambda\_handler(event, context):

# Extract S3 object details

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

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

# Send email

response = ses\_client.send\_email(

Source='your\_verified\_email@example.com',

Destination={

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

},

Message={

'Subject': {'Data': 'New S3 Object Added'},

'Body': {'Text': {'Data': f'File {key} added to {bucket}'}}

}

)

print("Email sent! Message ID:", response['MessageId'])

return {"statusCode": 200, "body": "Email sent"}

Deploy the updated code.

**Part 3: Configure Lambda to Send Emails via SQS and SNS**

**Step 1: Create an SQS Queue**

1. Go to the **SQS Console** > **Create Queue**:
   * **Queue Name**: email-notification-queue.
   * **Queue Type**: Standard.
   * Configure permissions to allow Lambda access.

**Step 2: Create an SNS Topic**

1. Navigate to the **SNS Console** > **Topics**.
2. Click **Create topic**:
   * **Type**: Standard.
   * **Name**: email-notification-topic.
3. Subscribe SES-verified email addresses to this topic:
   * Select **Create subscription**.
   * Set **Protocol** to Email.
   * Enter the SES-verified email and confirm the subscription.

**Step 3: Update Lambda to Send to SQS**

Modify the Lambda function to publish SQS messages. Example:

import boto3

import json

sqs\_client = boto3.client('sqs')

queue\_url = 'https://sqs.<region>.amazonaws.com/<account-id>/email-notification-queue'

def lambda\_handler(event, context):

# Extract S3 object details

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

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

# Send message to SQS

response = sqs\_client.send\_message(

QueueUrl=queue\_url,

MessageBody=json.dumps({

'bucket': bucket,

'key': key,

'event': 'ObjectCreated'

})

)

print("Message sent to SQS! Message ID:", response['MessageId'])

return {"statusCode": 200, "body": "Message sent to SQS"}

Deploy the updated code.

**Step 4: Configure SNS to Relay Emails**

Create another Lambda function to process SQS messages and send notifications via SNS:

import boto3

import json

sns\_client = boto3.client('sns')

topic\_arn = 'arn:aws:sns:<region>:<account-id>:email-notification-topic'

def lambda\_handler(event, context):

for record in event['Records']:

message = json.loads(record['body'])

bucket = message['bucket']

key = message['key']

# Publish to SNS

sns\_client.publish(

TopicArn=topic\_arn,

Message=f"New object {key} added to bucket {bucket}",

Subject="S3 Notification"

)

return {"statusCode": 200, "body": "Messages processed"}

Attach this Lambda to the **SQS queue** as a trigger.

**Testing**

1. Upload an object to the S3 bucket.
2. Verify:
   * Lambda triggers and processes the S3 event.
   * A message is sent to the SQS queue.
   * The second Lambda consumes the SQS message and sends an SNS notification.
   * SES delivers the email to subscribed recipients.