**Automating Image Upload Alerts with AWS**

*Sem End Project Report*

Cloud & Serverless Computing

Department of Computer Science and Engineering

By Chandra Mouli Garlapati

2200032332

Section-31 Advanced under the

supervision of DR.K .V. RAVITEJA

Koneru Lakshmaiah Education Foundation

(Deemed to be University estd., u/s 3 of UGC Act 1956)

Green Fields, Vaddeswaram, Guntur (Dist.), Andhra Pradesh – 522302

December, 2024

## CONTENTS

1. [Abstract 0](#_bookmark0)4
2. [Introduction 0](#_bookmark1)5
3. [Literature Review](#_bookmark2) 06
4. Project Objectives and Scope
   1. [Problem Statement](#_bookmark3) 08
5. [Technical Implementation 1](#_bookmark5)0
6. [Analysis & Problem-Solving](#_TOC_250001) 13
7. [Discussions & Results](#_TOC_250000) 15
8. [Conclusion](#_bookmark7) 20
9. [Future Work](#_bookmark8) 21
10. [References](#_bookmark9) 22

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| Figure No | Title | Page  Number |
| 1 | Architecture | 14 |
| 2 | IAM Roles | 16 |
| 3 | Lambda function | 17 |
| 4 | S3 Bucket | 19 |
| 5 | SNS | 20 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Architecture Overview**

**The architecture of this project is designed using a serverless approach, leveraging AWS services to automate notifications upon image uploads. The system begins with an Amazon S3 bucket, where image files (such as .jpg or .png) are uploaded. This bucket is configured to trigger an AWS Lambda function whenever a new object is created. The Lambda function extracts relevant metadata from the event and publishes a formatted message to an Amazon SNS (Simple Notification Service) topic. SNS then forwards this notification to all subscribed endpoints, such as email addresses or SMS numbers. To ensure secure access, an IAM role is created and attached to the Lambda function, providing it with permissions to read from the S3 bucket, publish messages to SNS, and write logs to CloudWatch. This architecture ensures real-time, scalable, and cost-efficient alerts without the need for manual monitoring or infrastructure management.**

**🔐 IAM Roles**

IAM (Identity and Access Management) roles are essential for secure interactions between AWS services. A custom role is created for the Lambda function, with the necessary permissions:

**✅ Permissions:**

* **Read from S3**: So the function can access image metadata.
* **Publish to SNS**: To send notifications.
* **Write to CloudWatch Logs**: For logging and debugging.

**📄 Example IAM Role Policy (JSON):**

json

CopyEdit

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": ["s3:GetObject"],

"Resource": "arn:aws:s3:::your-bucket-name/\*"

},

{

"Effect": "Allow",

"Action": ["sns:Publish"],

"Resource": "arn:aws:sns:us-east-1:123456789012:ImageUploadAlerts"

},

{

"Effect": "Allow",

"Action": ["logs:CreateLogGroup", "logs:CreateLogStream", "logs:PutLogEvents"],

"Resource": "\*"

}

]

}

Make sure to attach this role to your Lambda function when creating it.

**🧠 Lambda Function  
A computer screen shot of a computer screen

AI-generated content may be incorrect.**

Lambda is the core compute service that is triggered on S3 events. It processes the upload event and sends notifications via SNS.

**⚙️ Trigger Setup:**

* Triggered by **ObjectCreated** event on S3.
* Handles .jpg, .jpeg, .png files only (using suffix filters).

**📝 Example Lambda Code (Python):**

python

CopyEdit

import json

import boto3

sns = boto3.client('sns')

TOPIC\_ARN = 'arn:aws:sns:us-east-1:123456789012:ImageUploadAlerts'

A computer screen with text on it

AI-generated content may be incorrect.

def lambda\_handler(event, context):

for record in event['Records']:

bucket = record['s3']['bucket']['name']

key = record['s3']['object']['key']

message = f"New image uploaded to S3:\nBucket: {bucket}\nFile: {key}"

response = sns.publish(

TopicArn=TOPIC\_ARN,

Message=message,

Subject='New Image Upload Alert'

)

print("Notification sent:", response)

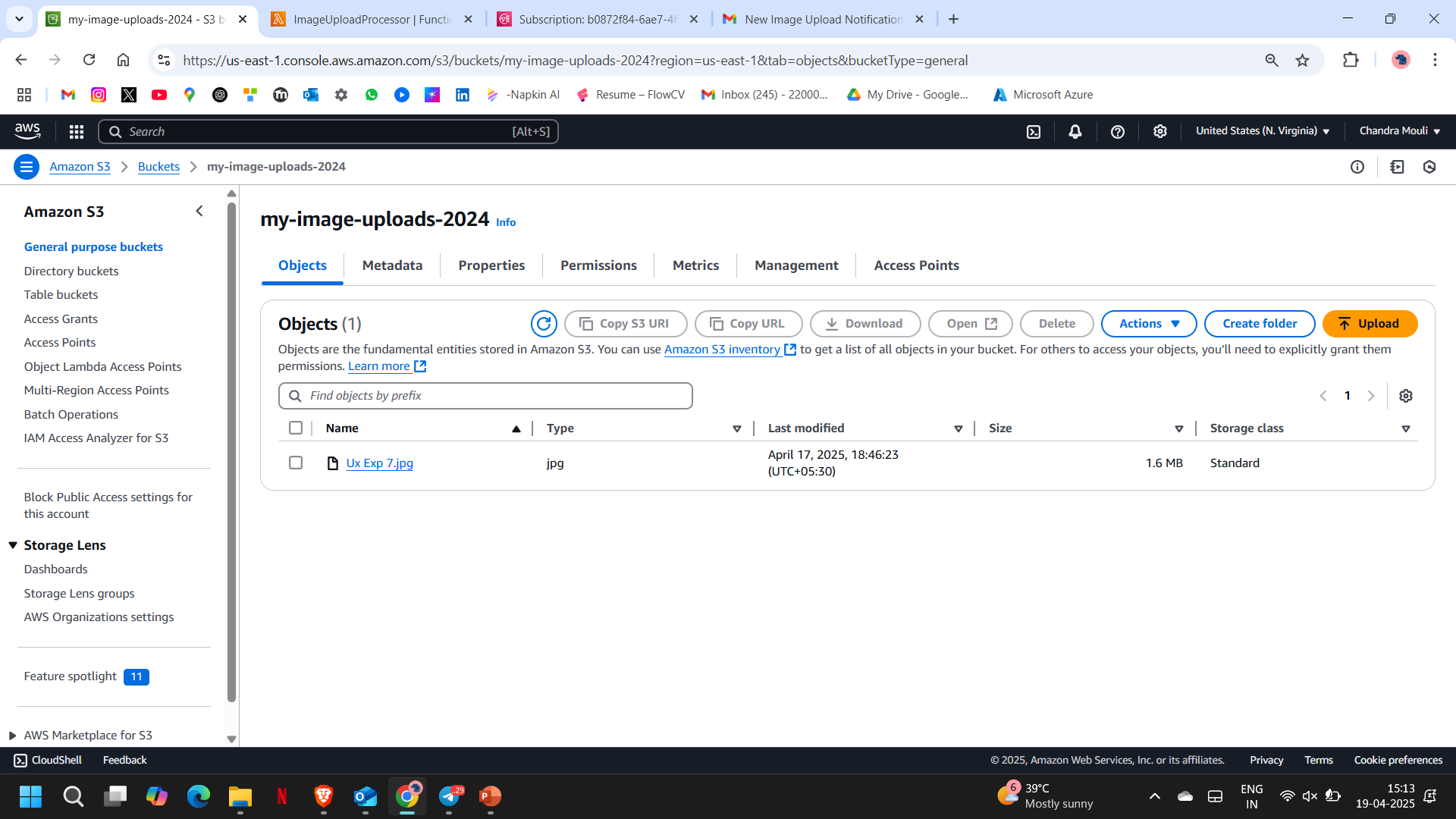
return {

'statusCode': 200,

'body': json.dumps('Success')

}

This function reads the event metadata, formats a message, and sends it to the SNS topic.

**🪣 S3 Bucket  
**

Amazon S3 stores the image files. It’s configured to trigger the Lambda function whenever a new image is uploaded.

**🧩 Key Settings:**

* **Event Notification**: On PUT (ObjectCreated) events.
* **Filters**: To trigger only on image file extensions like .jpg, .jpeg, .png.

**🔧 Example Configuration (AWS Console or Terraform/CloudFormation):**

json

CopyEdit

{

"LambdaFunctionConfigurations": [

{

"Id": "ImageUploadTrigger",

"LambdaFunctionArn": "arn:aws:lambda:us-east-1:123456789012:function:NotifyOnImageUpload",

"Events": ["s3:ObjectCreated:\*"],

"Filter": {

"Key": {

"FilterRules": [

{ "Name": "suffix", "Value": ".jpg" },

{ "Name": "suffix", "Value": ".png" }

]

}

}

}

]

}

Ensure the Lambda has permission to be invoked by S3 using a resource-based policy.

**📢 SNS (Simple Notification Service)**A computer screen with a white background

AI-generated content may be incorrect.

SNS is used to deliver real-time notifications when new images are uploaded.

**📨 Use Case:**

* Sends email or SMS to subscribed users.
* Messages include bucket and file details.

**🧪 Sample Notification Message:**

vbnet

CopyEdit

Subject: New Image Upload Alert

New image uploaded to S3:

Bucket: my-photo-bucket

File: uploads/image1.jpg

**📬 Example Email Subscription (CLI or Console):**

bash

CopyEdit

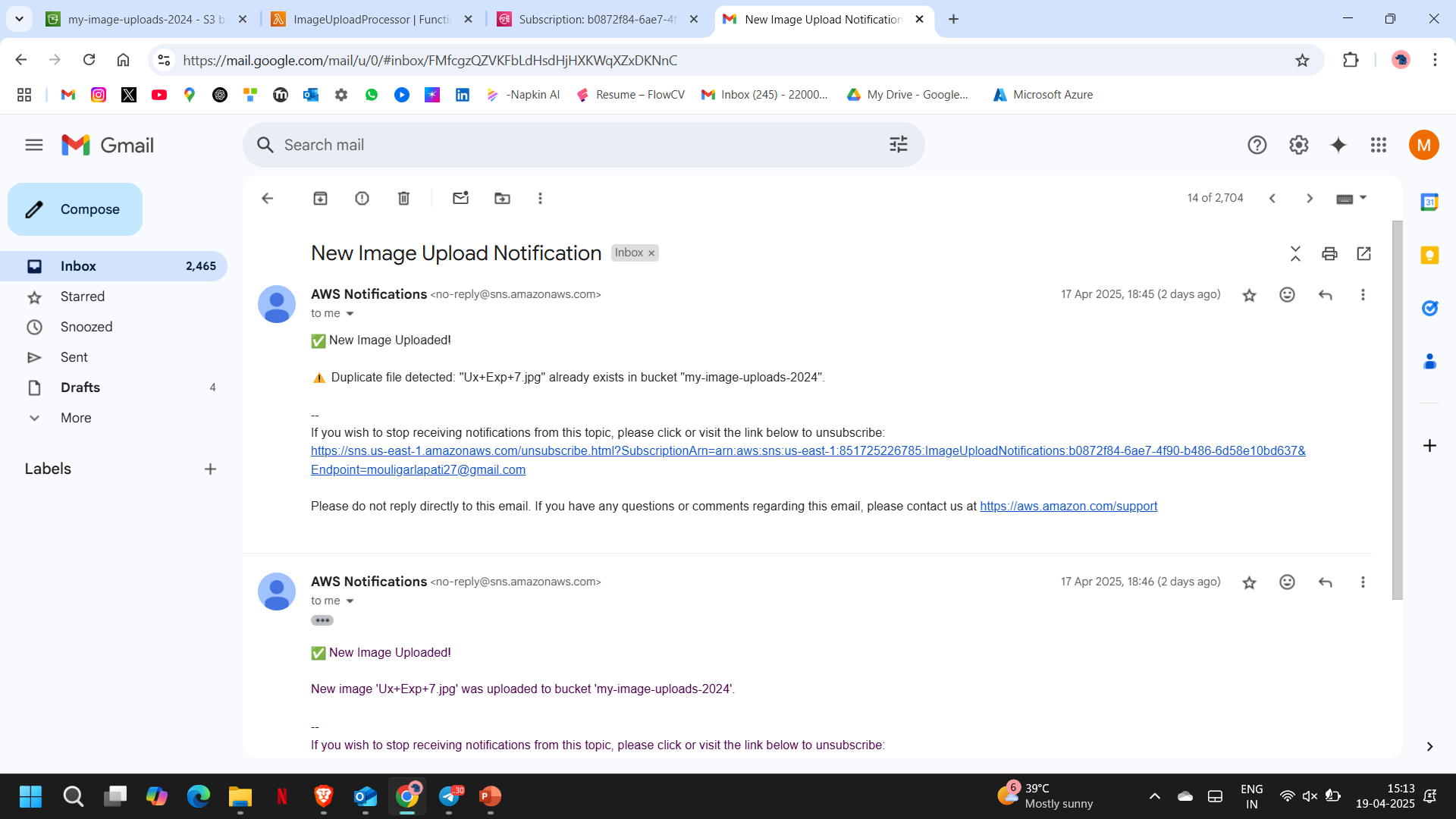
aws sns subscribe \

--topic-arn arn:aws:sns:us-east-1:123456789012:ImageUploadAlerts \

--protocol email \

--notification-endpoint your-email@example.com

Output:-



**✅ Conclusion**

The **Automating Image Upload Alerts with AWS** project demonstrates the power and simplicity of serverless architecture using AWS services. By integrating **S3**, **Lambda**, **SNS**, and **IAM**, we successfully created an automated workflow that provides real-time notifications when image files are uploaded to a designated S3 bucket.

This solution offers several benefits:

* **Scalability**: The architecture is fully serverless and can handle any volume of uploads without manual intervention or scaling.
* **Cost-Efficiency**: AWS services like Lambda and SNS follow a pay-as-you-go model, ensuring you only pay for actual usage.
* **Real-Time Alerts**: Immediate notification of new image uploads enhances responsiveness for systems that depend on fresh content.
* **Security**: Fine-grained IAM roles ensure each component has only the permissions it needs.

This setup can be extended further by:

* Adding image validation or processing in the Lambda function.
* Logging metadata into a database like DynamoDB or RDS.
* Connecting to workflow automation tools like Step Functions or EventBridge.

By leveraging AWS's powerful suite of tools, this project lays the groundwork for more advanced, event-driven image handling systems in the cloud.