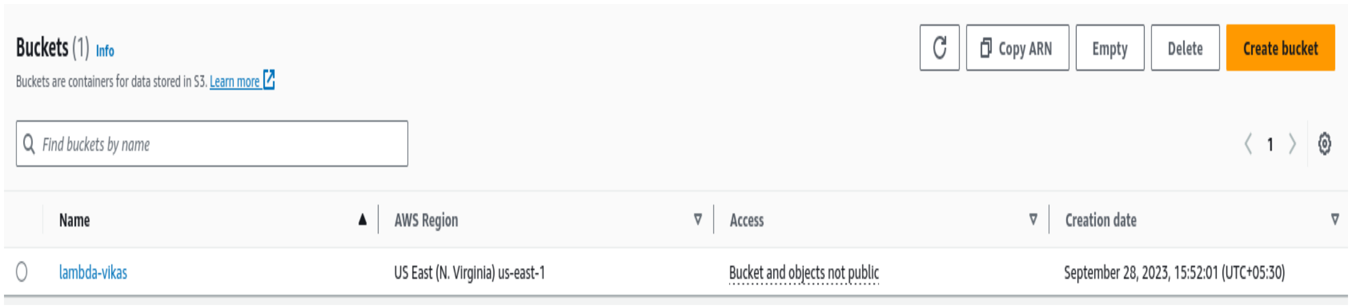
Trigger Email Notification through SNS using Lambda when Object is uploaded to S3

STEP 1: CREATE AN S3 BUCKET

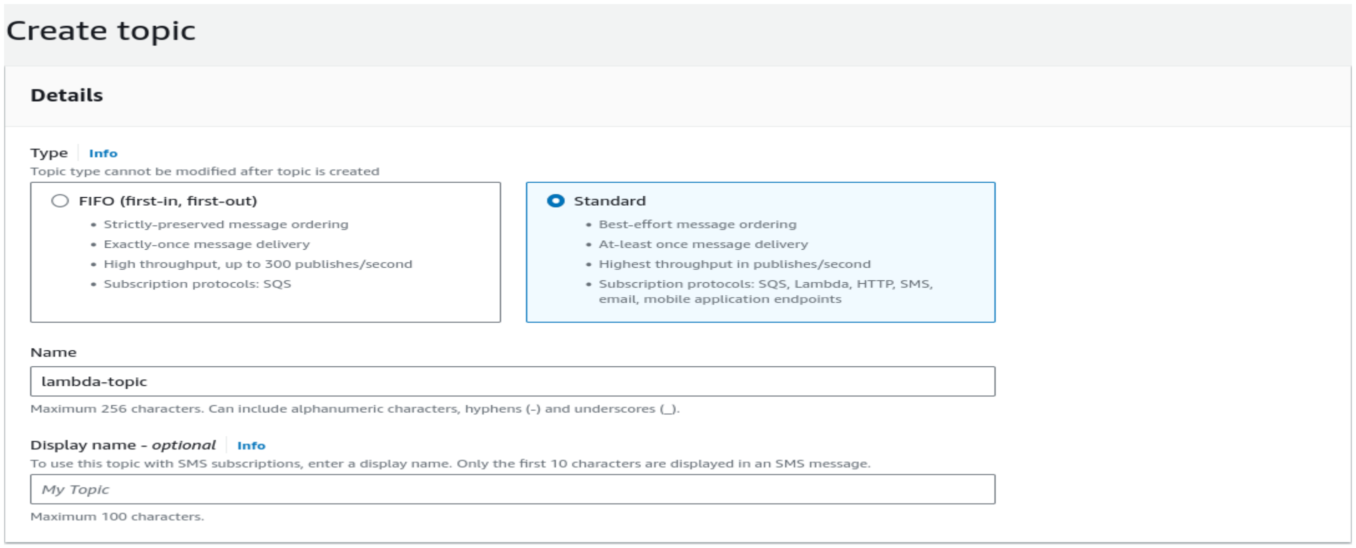
→ Create a bucket with a name and keep other options as default and click on create bucket.



Name	AWS Region	Access	Creation date
lambda-vikas	US East (N. Virginia) us-east-1	Bucket and objects not public	September 28, 2023, 15:52:01 (UTC+05:30)

STEP 2: CREATE AN SNS TOPIC AND SUBSCRIPTION

→ Search for SNS and click on Create topic.
→ Select **Standard** type, give a name to the topic and click on create topic.



Create topic

Details

Type | Info
Topic type cannot be modified after topic is created

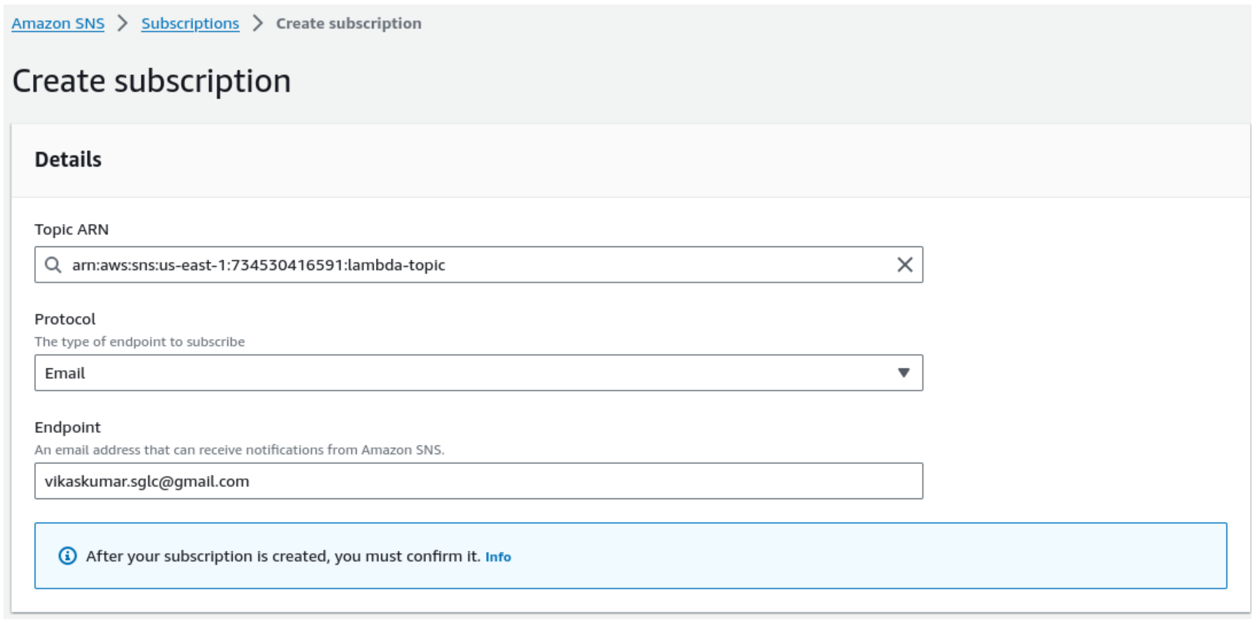
○ FIFO (first-in, first-out)
- Strictly-preserved message ordering
- Exactly-once message delivery
- High throughput, up to 300 publishes/second
- Subscription protocols: SQS

● Standard
- Best-effort message ordering
- At-least once message delivery
- Highest throughput in publishes/second
- Subscription protocols: SQS, Lambda, HTTP, SMS, email, mobile application endpoints

Name
lambda-topic
Maximum 256 characters. Can include alphanumeric characters, hyphens (-) and underscores (_).

Display name - *optional* Info
To use this topic with SMS subscriptions, enter a display name. Only the first 10 characters are displayed in an SMS message.
My Topic
Maximum 100 characters.

→ Create a subscription for the topic by selecting the Topic ARN in the drop down.
→ Select the **Email** as the protocol and **Endpoint** as the email address to get notifications.


Amazon SNS > Subscriptions > Create subscription

Create subscription

Details

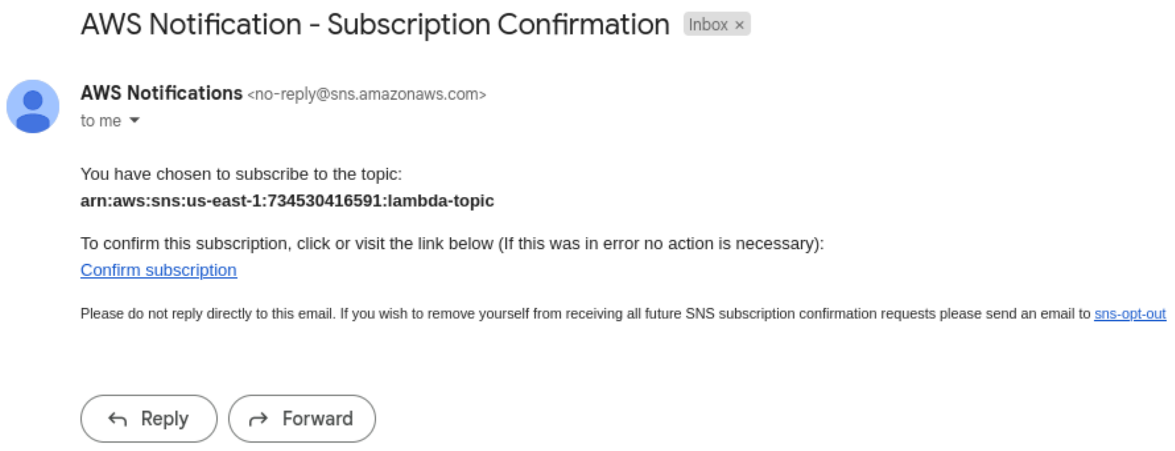
Topic ARN
arn:aws:sns:us-east-1:734530416591:lambda-topic

Protocol
The type of endpoint to subscribe
Email

Endpoint
An email address that can receive notifications from Amazon SNS.
vikaskumar.sglc@gmail.com

After your subscription is created, you must confirm it. Info

→ Confirm the subscription by clicking on confirmatio link in the mail given.


AWS Notification - Subscription Confirmation Inbox ×

AWS Notifications <no-reply@sns.amazonaws.com>
to me

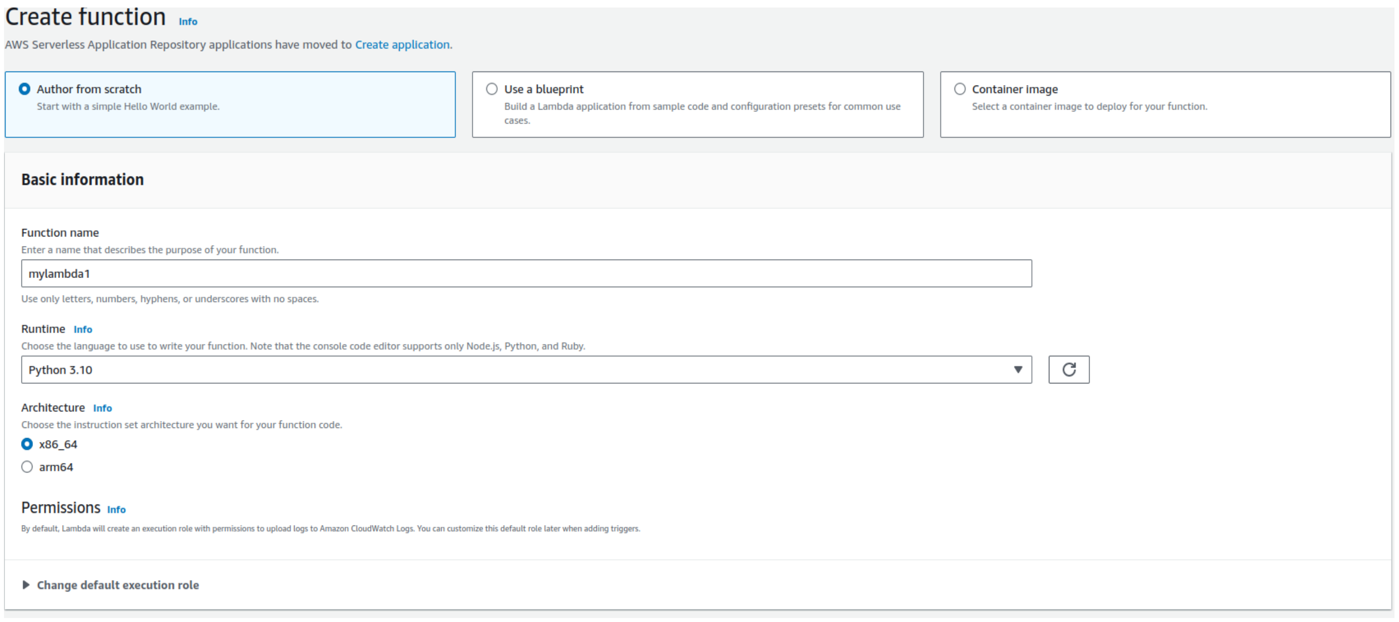
You have chosen to subscribe to the topic:
arn:aws:sns:us-east-1:734530416591:lambda-topic

To confirm this subscription, click or visit the link below (If this was in error no action is necessary):
Confirm subscription

Please do not reply directly to this email. If you wish to remove yourself from receiving all future SNS subscription confirmation requests please send an email to sns-opt-out

STEP 3: CREATE A LAMBDA FUNCTION

→ Search for lambda service and click on create function.
→ Choose runtime as python 3.10 or any version which is latest.
→ Under permission, select “ **Create a new role with basic Lambda permissions** “ and then select create function.



Create function Info

AWS Serverless Application Repository applications have moved to Create application.

[✓ Author from scratch] Start with a simple Hello World example.

Use a blueprint — Build a Lambda application from sample code and configuration presets for common use cases.

Container image — Select a container image to deploy for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.
mylambda1
Use only letters, numbers, hyphens, or underscores with no spaces.

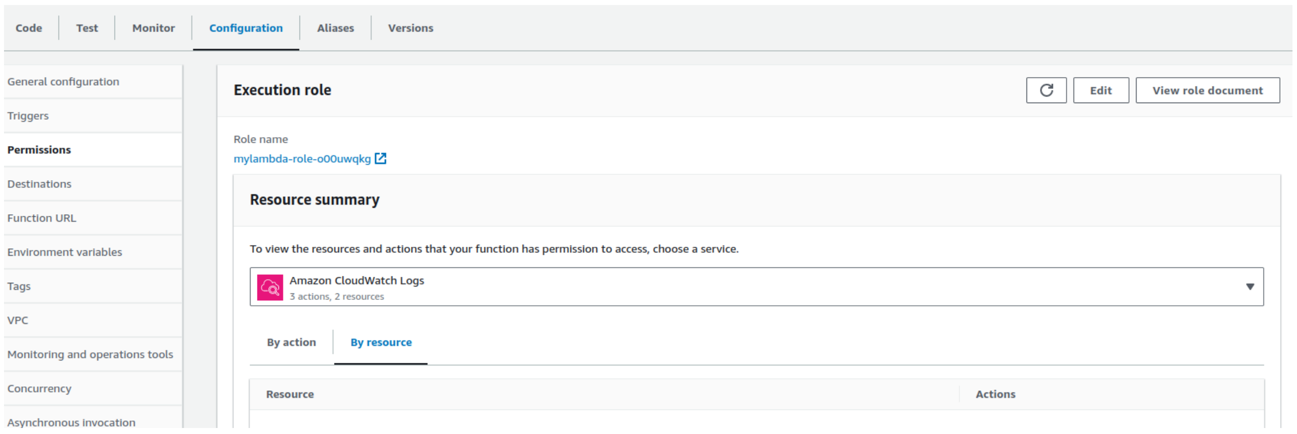
Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.10

Architecture Info
Choose the instruction set architecture you want for your function code.
[✓ x86_64]
arm64

Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▶ Change default execution role

→ Click on the function and go to configuration.
→ Click on the **role name** link and open the IAM role.



Execution role

Role name
mylambda-role-o0Ouwqkg

Resource summary

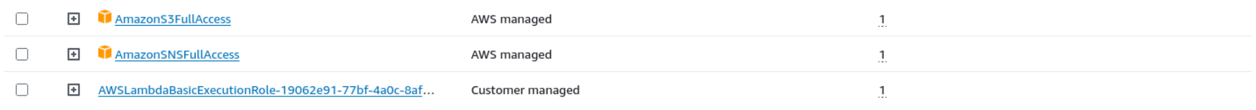
To view the resources and actions that your function has permission to access, choose a service.

Amazon CloudWatch Logs
3 actions, 2 resources

Resource	Actions

→ Add these policies to the lambda role:

- AmazonS3FullAccess
- AmazonSNSFullAccess

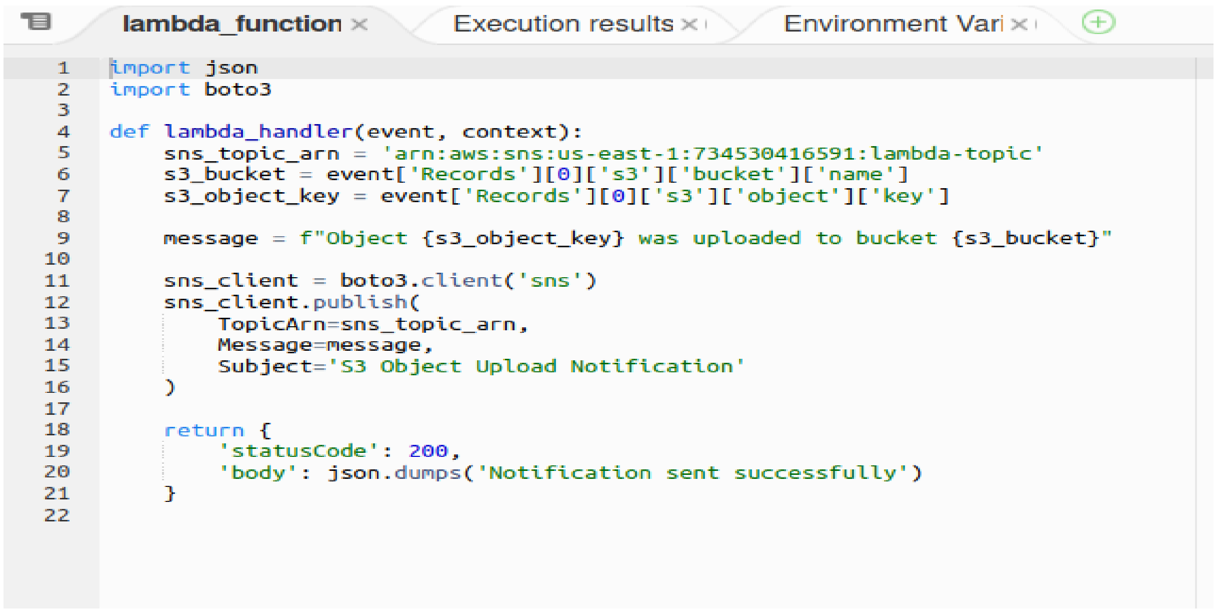


☐	AmazonS3FullAccess	AWS managed	1
☐	AmazonSNSFullAccess	AWS managed	1
☐	AWSLambdaBasicExecutionRole-19062e91-77bf-4a0c-8af...	Customer managed	1

→ Add the following code inside the lambda function:

```
import json
import boto3

def lambda_handler(event, context):
    sns_topic_arn = 'arn:aws:sns:us-east-1:734530416591:lambda-topic'
    s3_bucket = event['Records'][0]['s3']['bucket']['name']
    s3_object_key = event['Records'][0]['s3']['object']['key']

    message = f"Object {s3_object_key} was uploaded to bucket {s3_bucket}"

    sns_client = boto3.client('sns')
    sns_client.publish(
    TopicArn=sns_topic_arn,
    Message=message,
    Subject='S3 Object Upload Notification'
    )

    return {
    'statusCode': 200,
    'body': json.dumps('Notification sent successfully')
    }
```

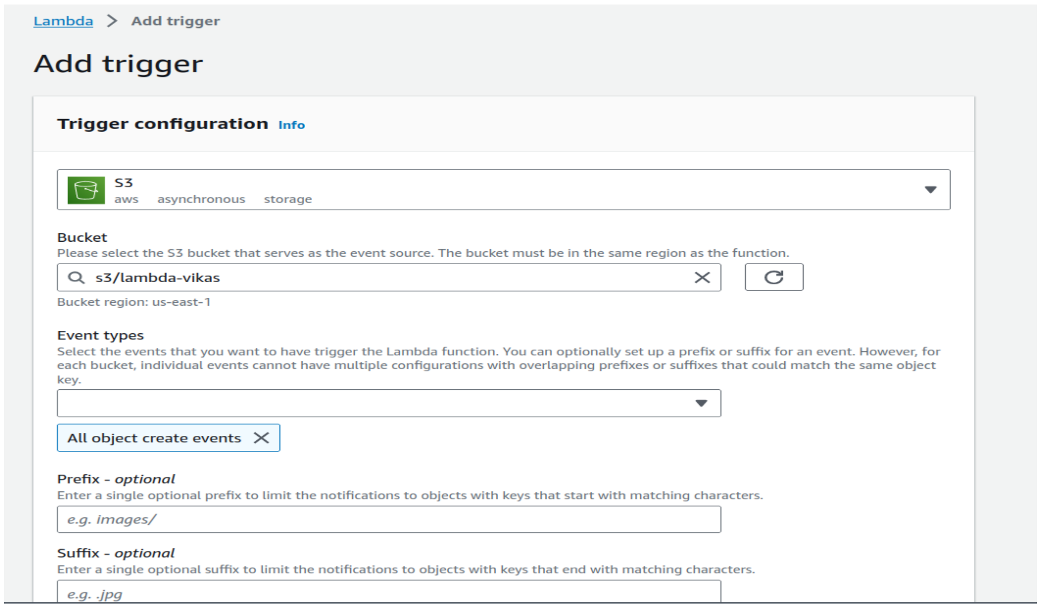
→ Make sure to add the ARN of the topic created in the lambda handler function.



```python
import json
import boto3

def lambda_handler(event, context):
    sns_topic_arn = 'arn:aws:sns:us-east-1:734530416591:lambda-topic'
    s3_bucket = event['Records'][0]['s3']['bucket']['name']
    s3_object_key = event['Records'][0]['s3']['object']['key']

    message = f"Object {s3_object_key} was uploaded to bucket {s3_bucket}"

    sns_client = boto3.client('sns')
    sns_client.publish(
        TopicArn=sns_topic_arn,
        Message=message,
        Subject='S3 Object Upload Notification'
    )

    return {
        'statusCode': 200,
        'body': json.dumps('Notification sent successfully')
    }

```

STEP 4: Configure S3 Bucket Event Trigger

→ Go to configuration and click on Triggers section.
→ Click on **Add Trigger** and select the bucket.
→ Under **Event Types** , select **All object create events.**
→ Tick the box at the bottom and click on **Add**.



Add trigger

Trigger configuration

S3
aws asynchronous storage

Bucket
Please select the S3 bucket that serves as the event source. The bucket must be in the same region as the function.
s3/lambda-vikas
Bucket region: us-east-1

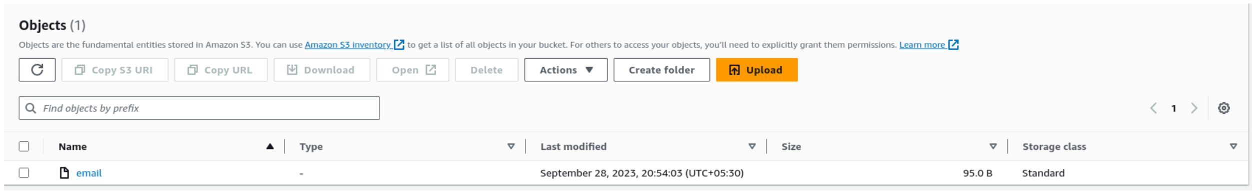
Event types
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.
All object create events

Prefix - *optional*
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters.
e.g. images/

Suffix - *optional*
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters.
e.g. .jpg

STEP 5: Trigger Object Creation Event

→ Go to S3 bucket and add an object to it.



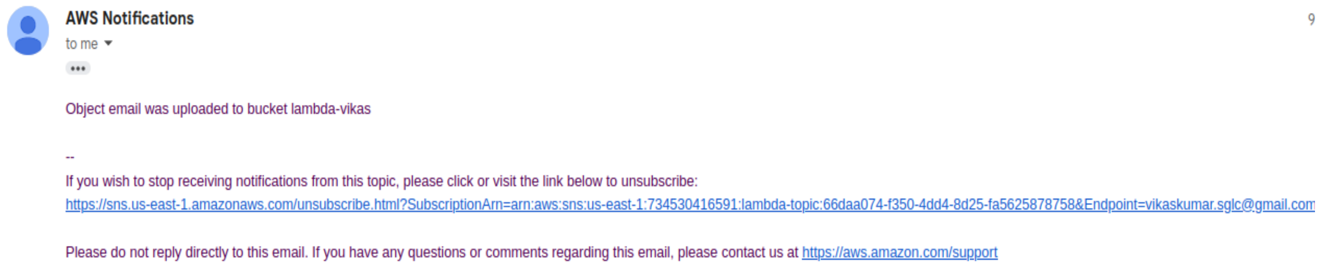
Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
email	-	September 28, 2023, 20:54:03 (UTC+05:30)	95.0 B	Standard

→ After adding the object,check for the the notification in email for adding object.

→ A notification will be triggered for uploading an object to S3.



AWS Notifications
to me

9

Object email was uploaded to bucket lambda-vikas

--
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:734530416591:lambda-topic:66daa074-f350-4dd4-8d25-fa5625878758&Endpoint=vikaskumar.sglc@gmail.com

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support