AWS Lambda + S3: Auto-Generate Image Thumbnails (Mini Project)

Step-by-step practical using S3 event triggers and a Python Lambda function

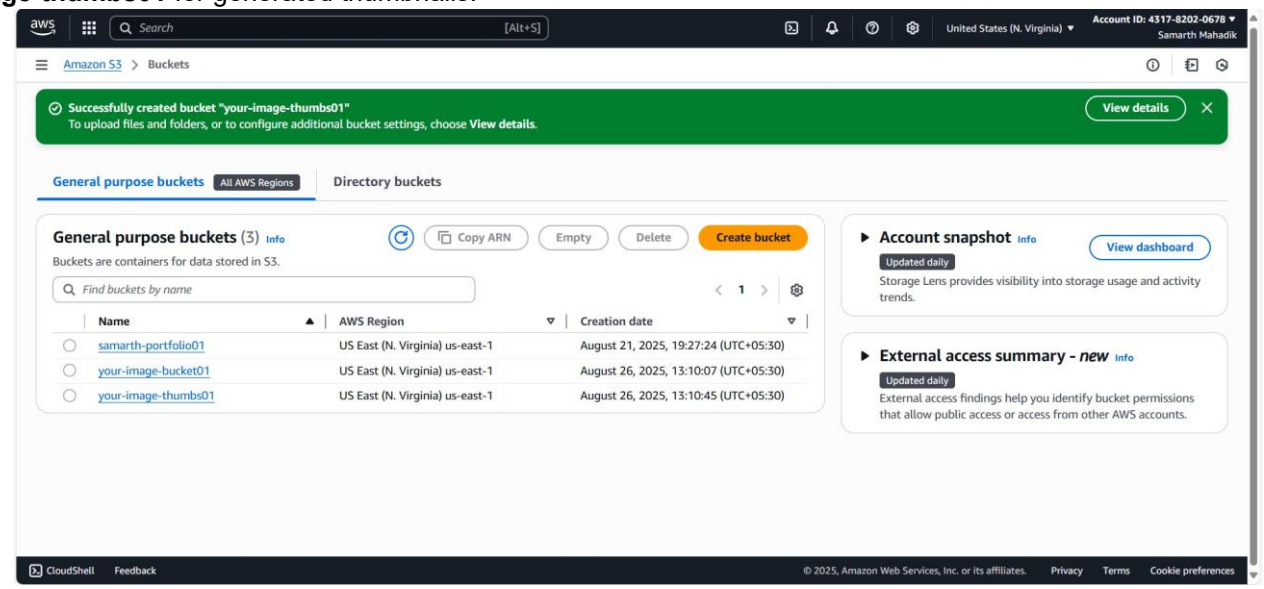
Author: Samarth Mahadik
Region: US East (N. Virginia) — us-east-1
Date: August 27, 2025

This document captures the exact steps I followed to build a serverless image-thumbnail generator on AWS. When a new image is uploaded to a source S3 bucket, an S3 event triggers an AWS Lambda function written in Python. The function creates a 128×128 thumbnail and saves it to a destination S3 bucket.

Services used	**Amazon S3, AWS Lambda, IAM**
Source bucket	your-image-bucket01
Destination bucket	your-image-thumbs01
Runtime	Python 3.8

Step 1 — Create two S3 buckets

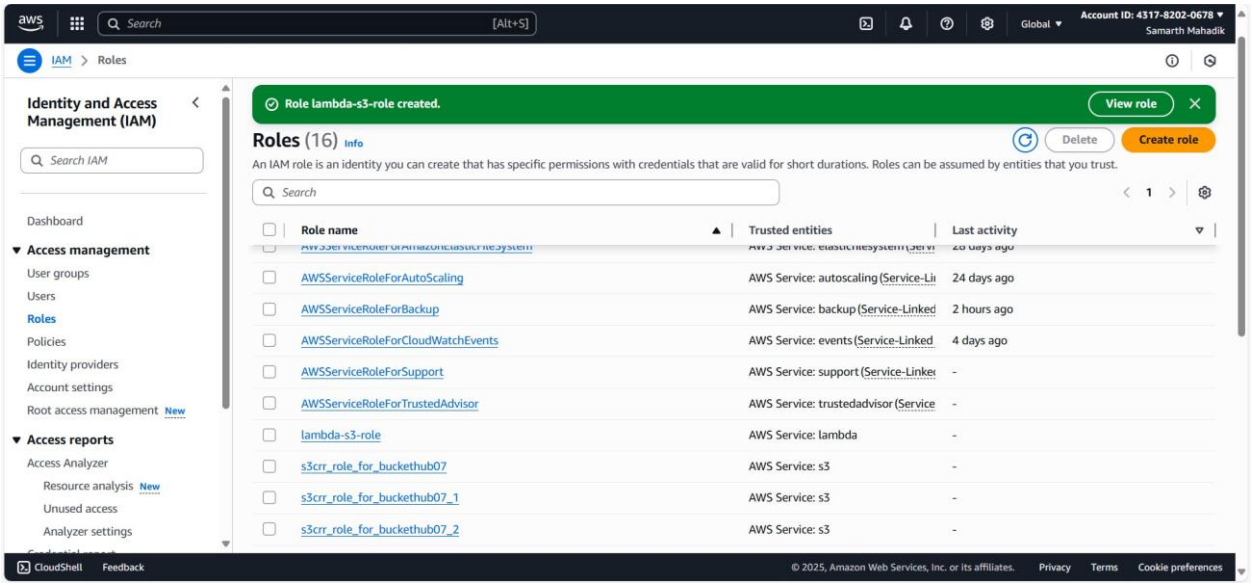
I created two general-purpose S3 buckets in **us-east-1**: 1) **your-image-bucket01** for original uploads, and 2) **your-image-thumbs01** for generated thumbnails.



Screenshot: S3 console showing the two buckets.

Step 2 — Create an IAM role for Lambda

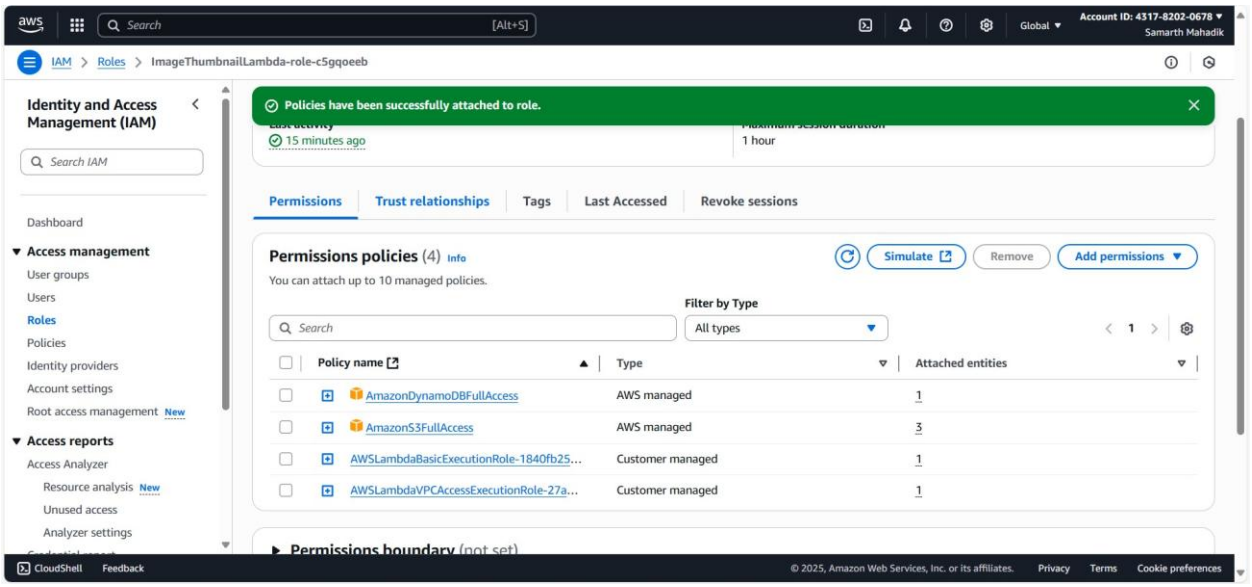
I created an execution role (e.g., **lambda-s3-role**) with a trust relationship for the Lambda service. This role will allow the function to read from the source bucket and write to the thumbnails bucket.



Screenshot: IAM › Roles with success banner for role creation.

Step 3 — Attach permissions policies

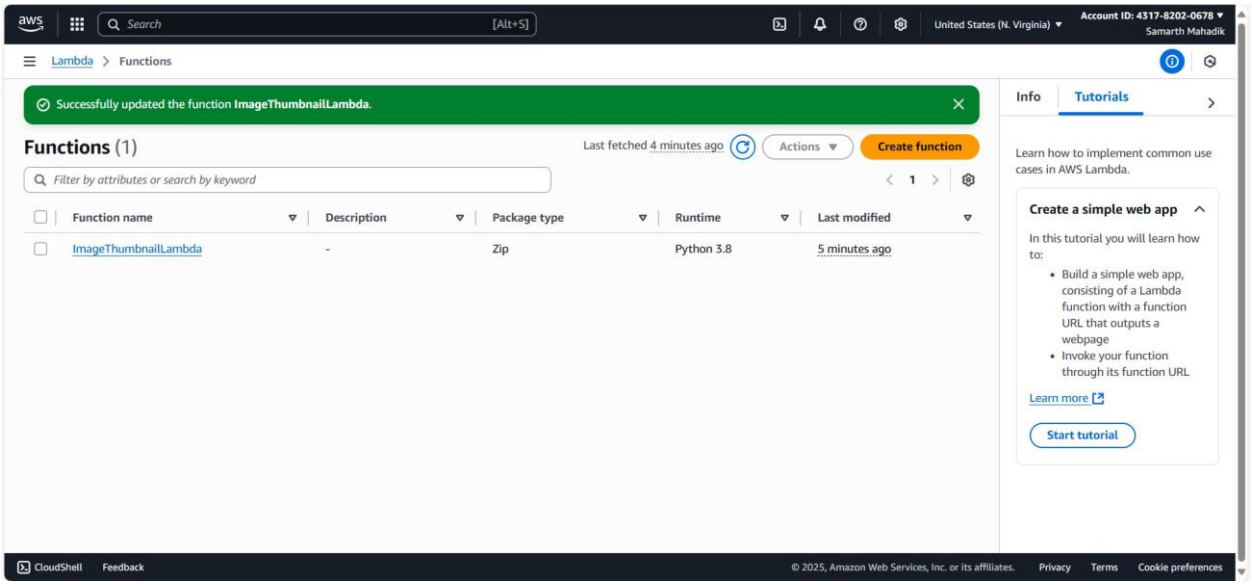
I attached the following policies to the role to enable S3 and basic Lambda execution: • AmazonS3FullAccess • AWSLambdaBasicExecutionRole (customer managed in my account) • AWSLambdaVPCAccessExecutionRole (customer managed in my account) (Shown in screenshot; note: for production, prefer least-privilege policies.)



Screenshot: Policies attached to the Lambda execution role.

Step 4 — Create the Lambda function

I created a function named **ImageThumbnailLambda** (runtime: Python 3.8). The function will be triggered by S3 events.



Screenshot: Lambda › Functions list showing ImageThumbnailLambda.

Step 5 — Add the function code

Below is the code used to generate 128×128 thumbnails. It reads the uploaded object from the source bucket, creates a thumbnail using Pillow (PIL), and uploads it to **your-image-thumbs01** with the prefix **thumb-**.

```
import os import
boto3 from PIL import
Image import tempfile

s3 = boto3.client('s3') THUMBNAIL_SIZE = (128, 128)
DEST_BUCKET = 'your-image-thumbs01'  # thumbnails go here

def lambda_handler(event, context):
for record in event['Records']:
        src_bucket = record['s3']['bucket']['name']
key = record['s3']['object']['key']

        # avoid reprocessing thumbnails
if key.startswith('thumb-'):
continue
```

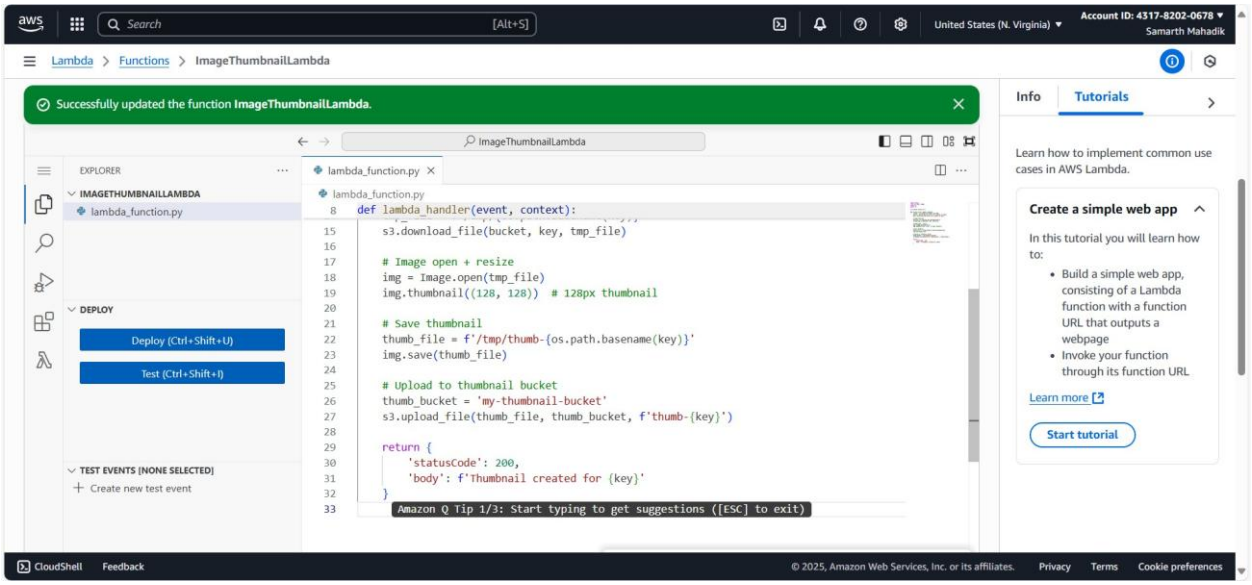
```
        with tempfile.TemporaryDirectory() as tmpdir:
            download_path = os.path.join(tmpdir, os.path.basename(key))
s3.download_file(src_bucket, key, download_path)

            img = Image.open(download_path)
img.thumbnail(THUMBNAIL_SIZE)

            thumb_key = f"thumb-{os.path.basename(key)}"
thumb_path = os.path.join(tmpdir, thumb_key)
img.save(thumb_path)

            s3.upload_file(thumb_path, DEST_BUCKET, thumb_key)

    return {'statusCode': 200, 'body': f'Thumbnail created for {key}'}
```

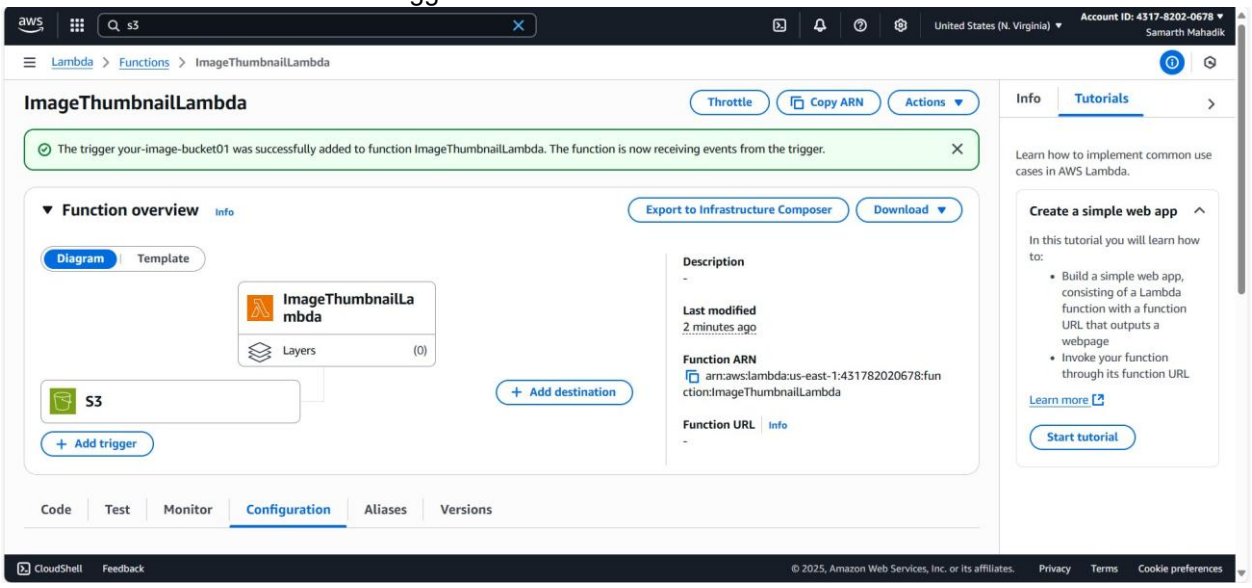


Screenshot: Inline code editor for the Lambda function.

Note: The Pillow library is a third-party dependency. If the inline editor environment does not include it, zip the code with dependencies (or use a Lambda layer) and upload the deployment package.

Step 6 — Add S3 trigger to the Lambda function

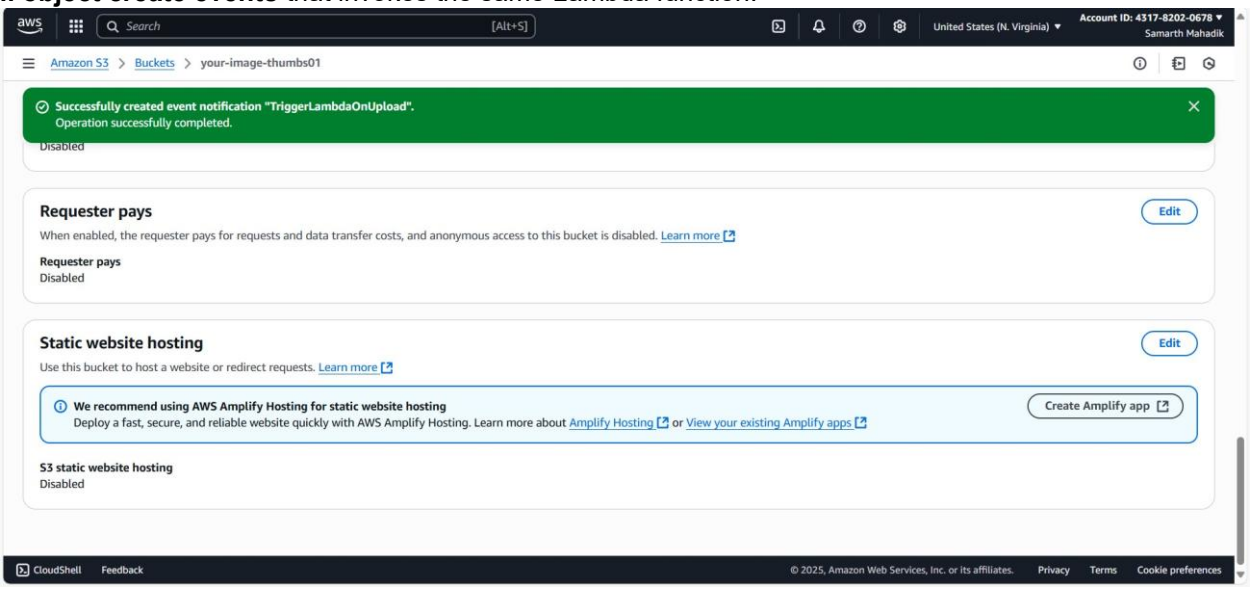
From Lambda, I added an S3 trigger for **your-image-bucket01** on ObjectCreated events. After saving, Lambda shows a confirmation banner that the trigger is connected.



Screenshot: Lambda configuration with S3 trigger attached.

Step 7 — (Optional) Create S3 Event Notification from the bucket

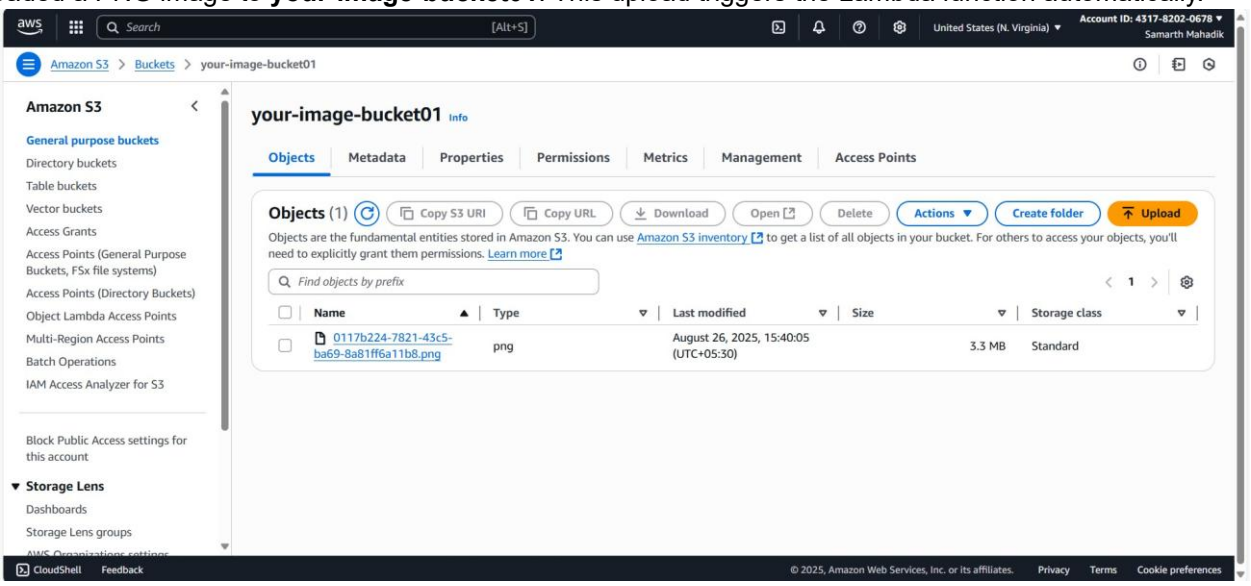
Alternatively (or additionally), on the source bucket I created an event notification (e.g., **TriggerLambdaOnUpload**) for **All object create events** that invokes the same Lambda function.



Screenshot: S3 bucket event notification created successfully.

Step 8 — Upload a test image to the source bucket

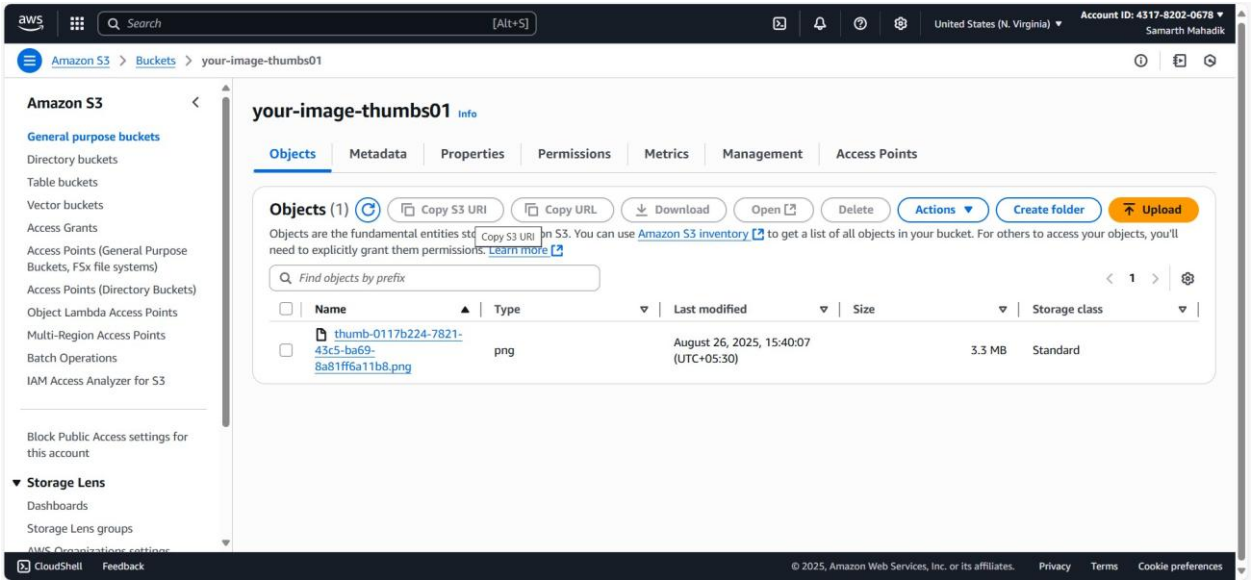
I uploaded a PNG image to **your-image-bucket01**. This upload triggers the Lambda function automatically.



Screenshot: your-image-bucket01 with the uploaded image object.

Step 9 — Verify thumbnail output

In **your-image-thumbs01**, a new object with the prefix **thumb-** appears, confirming the Lambda processed the upload and created the thumbnail.



Screenshot: your-image-thumbs01 showing the generated thumbnail object.

Final Notes & Cleanup

• **Cost:** This demo stays within the free tier for small tests (S3 storage + Lambda invocations). Delete test images if not needed. • **Security:** For production, replace broad policies (e.g., AmazonS3FullAccess) with least-privilege permissions limited to the specific buckets and actions (GetObject, PutObject, ListBucket). • **Cleanup:** Remove the S3 event notification/trigger, delete the Lambda function, detach policies, and delete the buckets to avoid ongoing charges.