**AWS Lambda**

* **AWS Lambda** is a **serverless compute service** offered by Amazon Web Services (AWS).  
  It lets you run your code **without managing servers**. You just upload your code, and Lambda automatically:
* Allocates compute power
* Runs your code when triggered
* Scales automatically based on demand
* Charges you only for the compute time used
* In short: **You don’t manage servers, you only manage your code.**

# Traditional vs Serverless

### Traditional way:

* You deploy code on servers (EC2, on-premises, etc.).
* You must configure servers, patch OS, scale manually, and pay for uptime (even when idle).

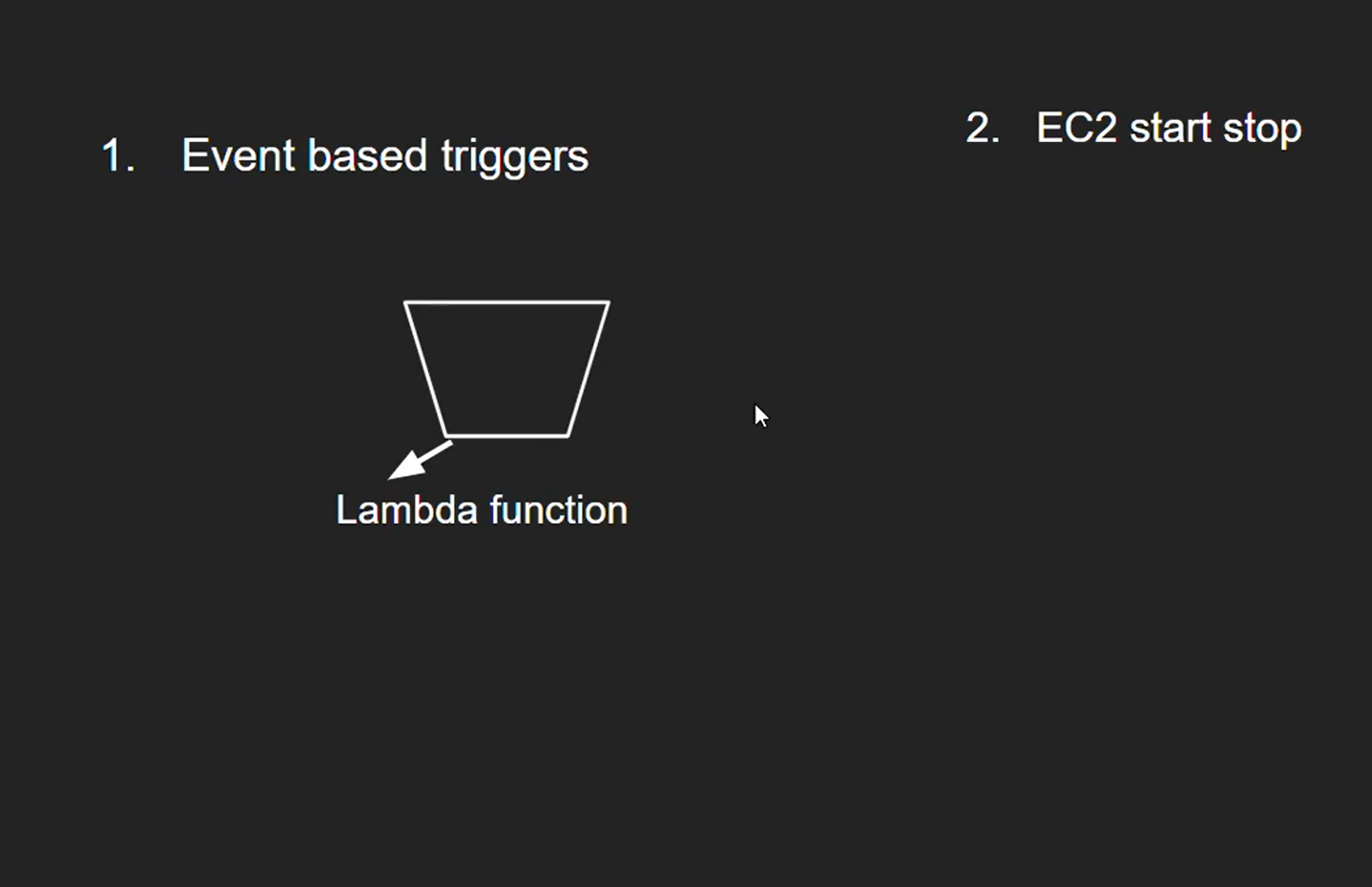
### With AWS Lambda (Serverless):

* No servers to manage (AWS handles them).
* You only write a function (called a **Lambda function**).
* AWS runs your function **only when triggered** (like an event).
* You pay only for execution time (in milliseconds).

# C:\Users\User\OneDrive\Pictures\Screenshots\Screenshot 2025-08-17 152938.png

# Key Concepts in AWS Lambda

* **Function**
  + The core unit of Lambda.
  + A piece of code you upload (Node.js, Python, Java, Go, .NET, Ruby, custom runtime).
  + Example: A function that resizes an image or processes logs.
* **Event**
  + Something that triggers the Lambda function.
  + Example triggers:
    - S3 bucket (file uploaded)
    - API Gateway (HTTP request)
    - DynamoDB stream (database changes)
    - CloudWatch Events/Alarms
    - Other AWS services



# How AWS Lambda Works (Step by Step)

* You **write a function** (e.g., in Python or Node.js).
* You **upload it to Lambda** (via AWS Console, CLI, or CI/CD pipeline).
* You **configure a trigger** (e.g., S3 upload).
* An **event occurs** (e.g., user uploads an image).
* Lambda **executes your code** in a temporary container (called an execution environment).
* Lambda **scales automatically** (creates multiple environments if many events come).
* Lambda **stops execution** when done (no servers kept running).

# Pricing (Why it’s cheap)

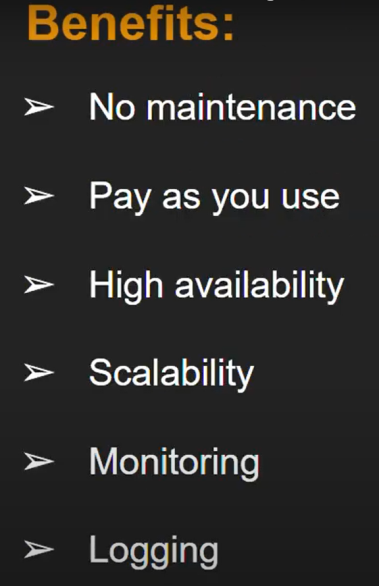
* You pay only for:
* **Number of requests** (first 1 million free per month).
* **Execution duration** (measured in milliseconds).
* **Memory allocated** (128 MB – 10 GB).
* Example:  
  If your function runs for 100ms and uses 128 MB RAM, you’re billed for that short duration only.

# Example Use Cases of AWS Lambda

* **S3 Trigger**
  + Automatically process an image when uploaded.
  + Example: Resize image, create thumbnail, save to another bucket.
* **API Backend**
  + Use Lambda with API Gateway to build a serverless REST API.
* **Automation**
  + CloudWatch scheduled events (run scripts daily).
  + Example: Clean old files, generate reports.

# Advantages of AWS Lambda

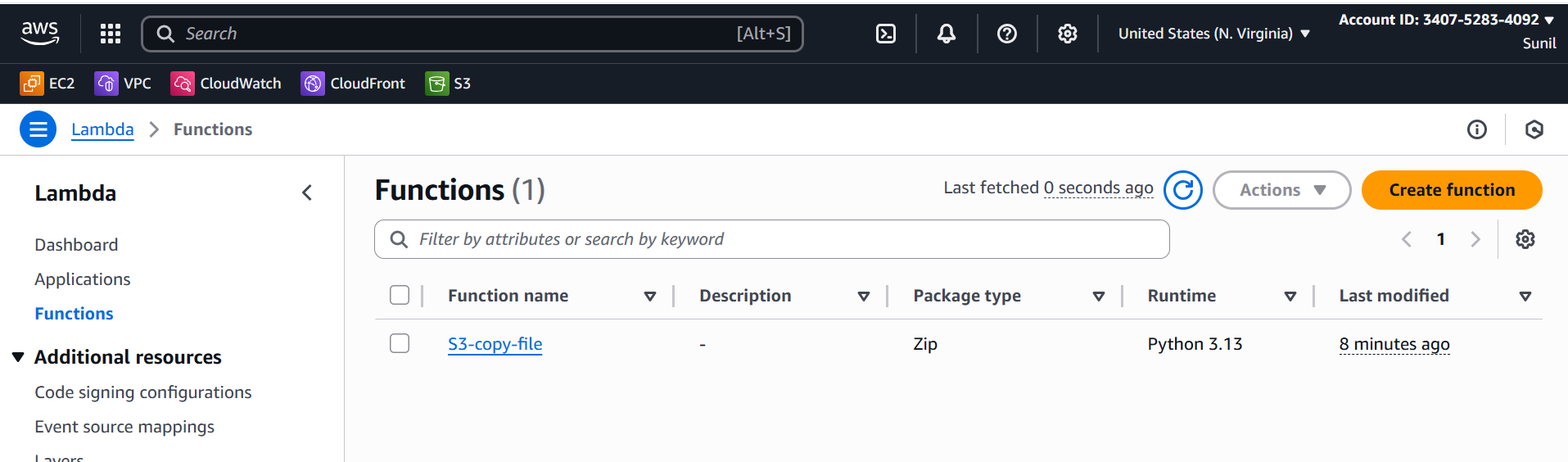
* No server management.
* Auto-scaling.
* Pay per use.
* Integrated with many AWS services.
* Highly available (runs across multiple AZs).

****

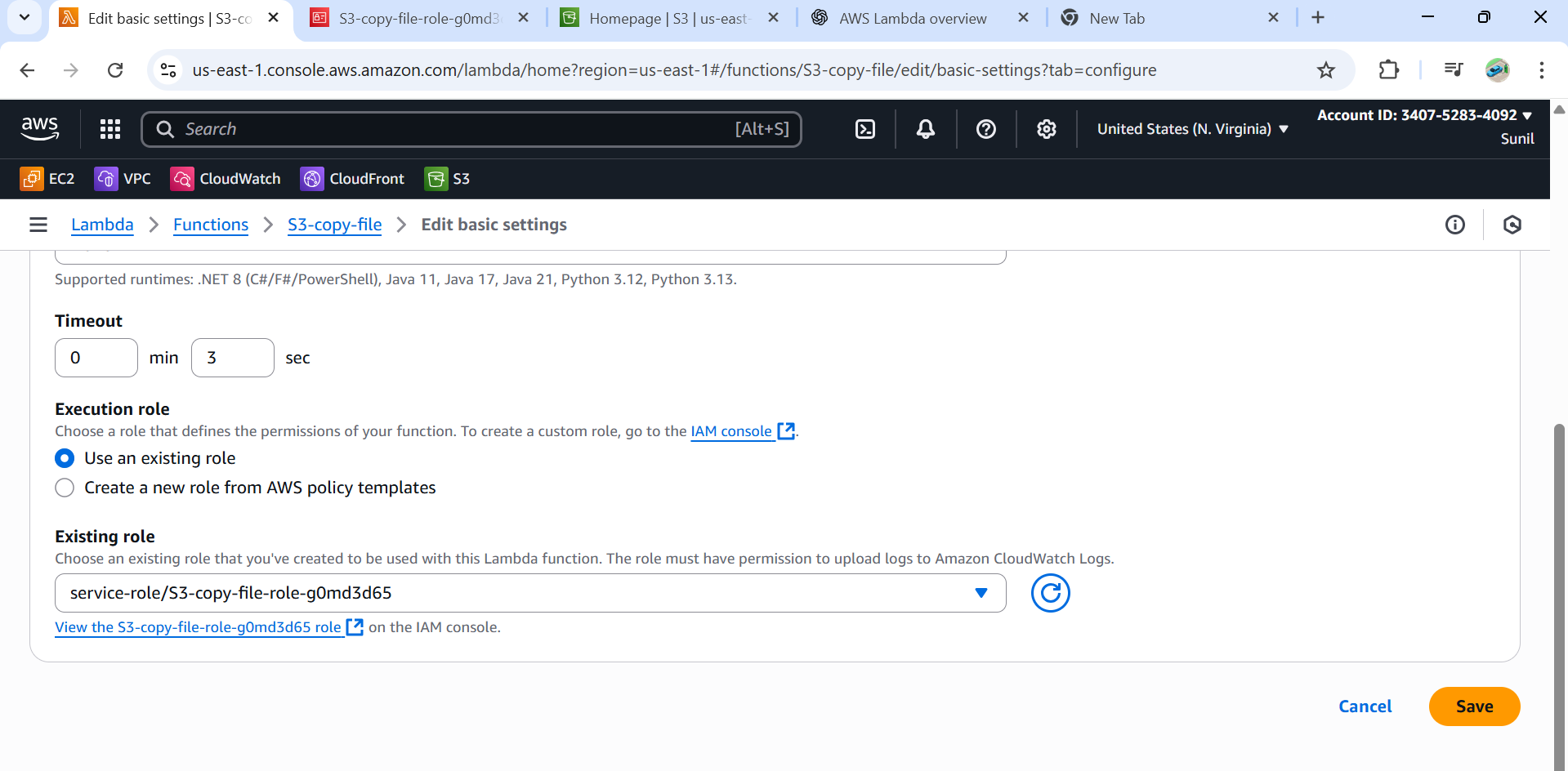
Task1.) Auto Copy Files Between S3 Buckets with Lambda

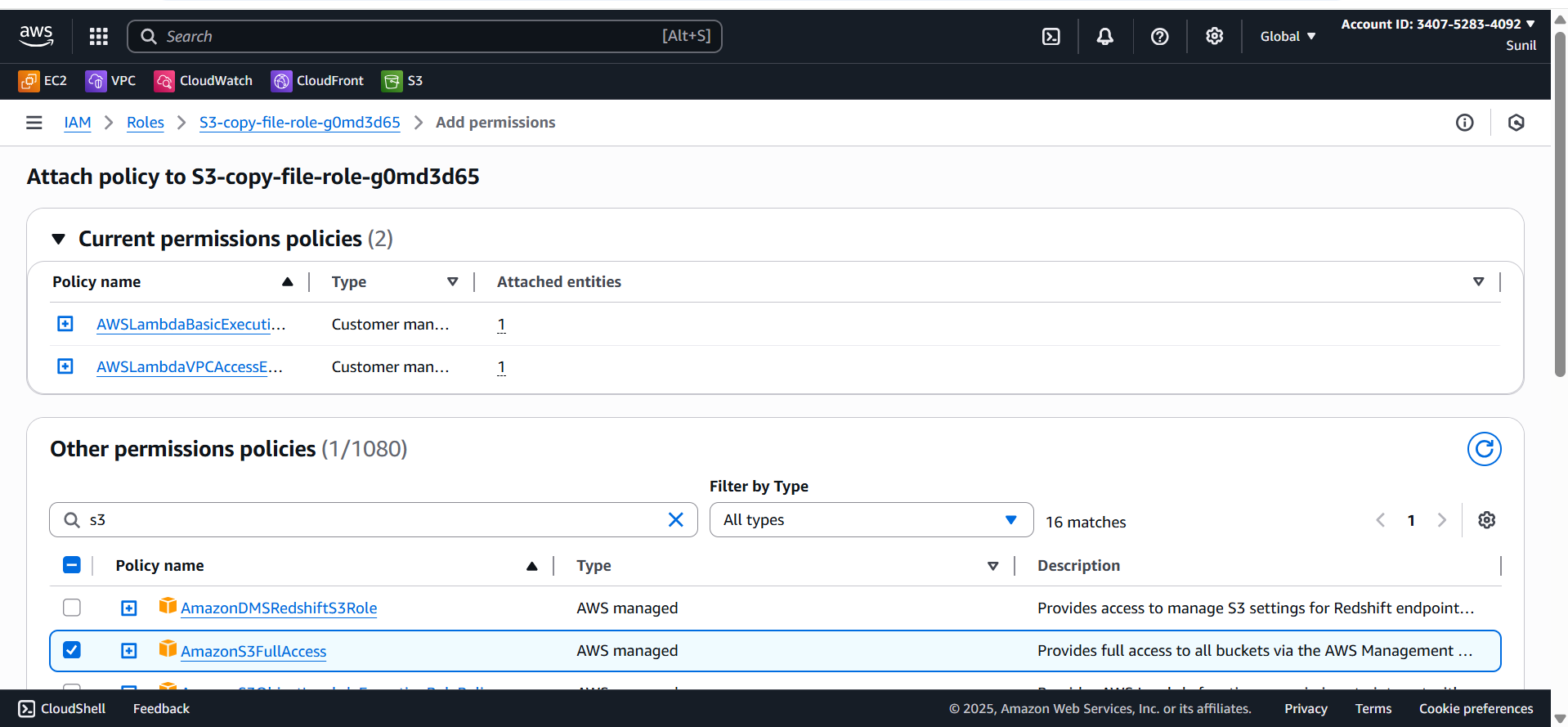
## Step 1: Create Lambda Function

1. Go to **AWS Lambda Console → Create function**.
   * Name: s3-Copy-file
   * Runtime: **Python**



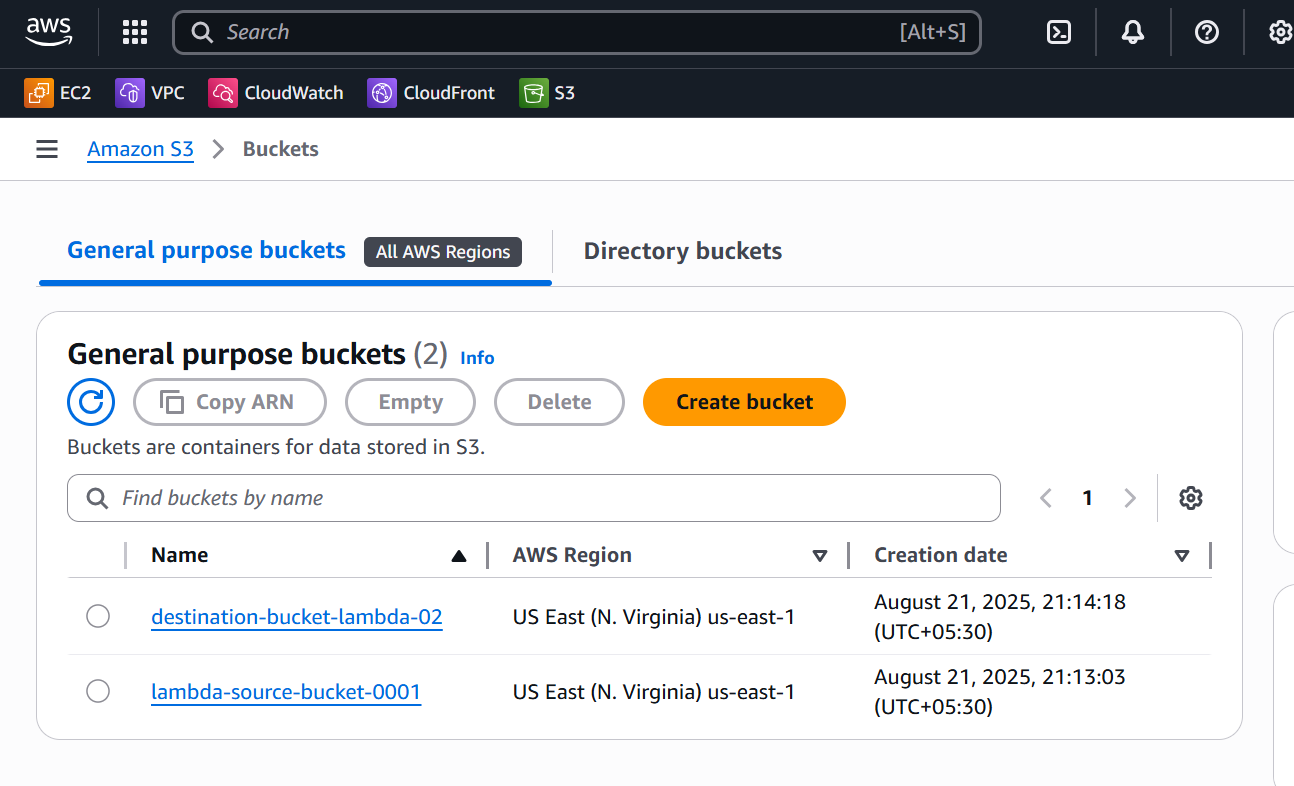
Attach s3 full acesss to the lambda assigned role





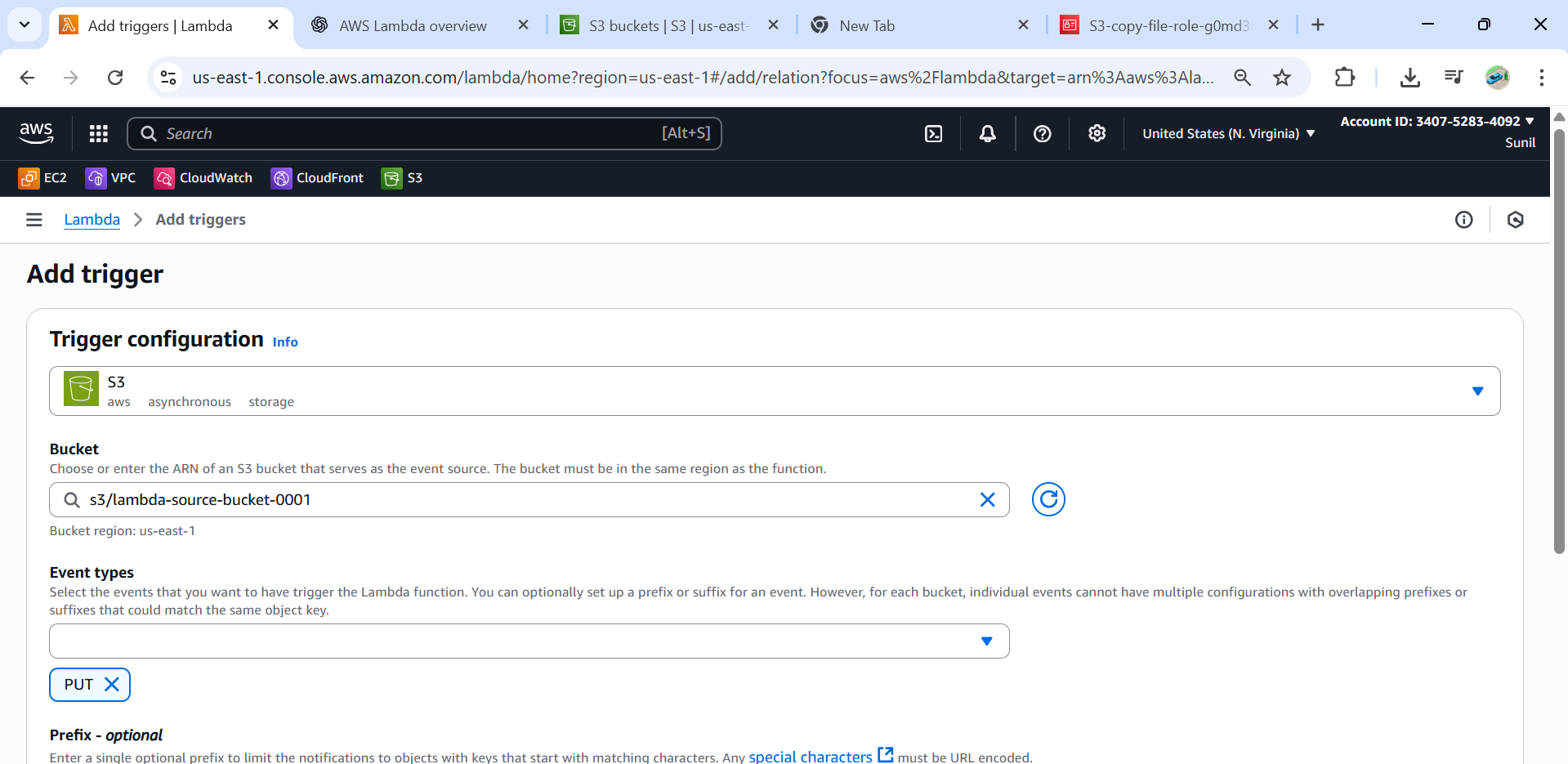
## Step 2: Create Two S3 Buckets

* [lambda-source-bucket-0001](https://us-east-1.console.aws.amazon.com/s3/buckets/lambda-source-bucket-0001?region=us-east-1&bucketType=general) → where you upload files
* [destination-bucket-lambda-02](https://us-east-1.console.aws.amazon.com/s3/buckets/destination-bucket-lambda-02?region=us-east-1&bucketType=general) → where files will be copied

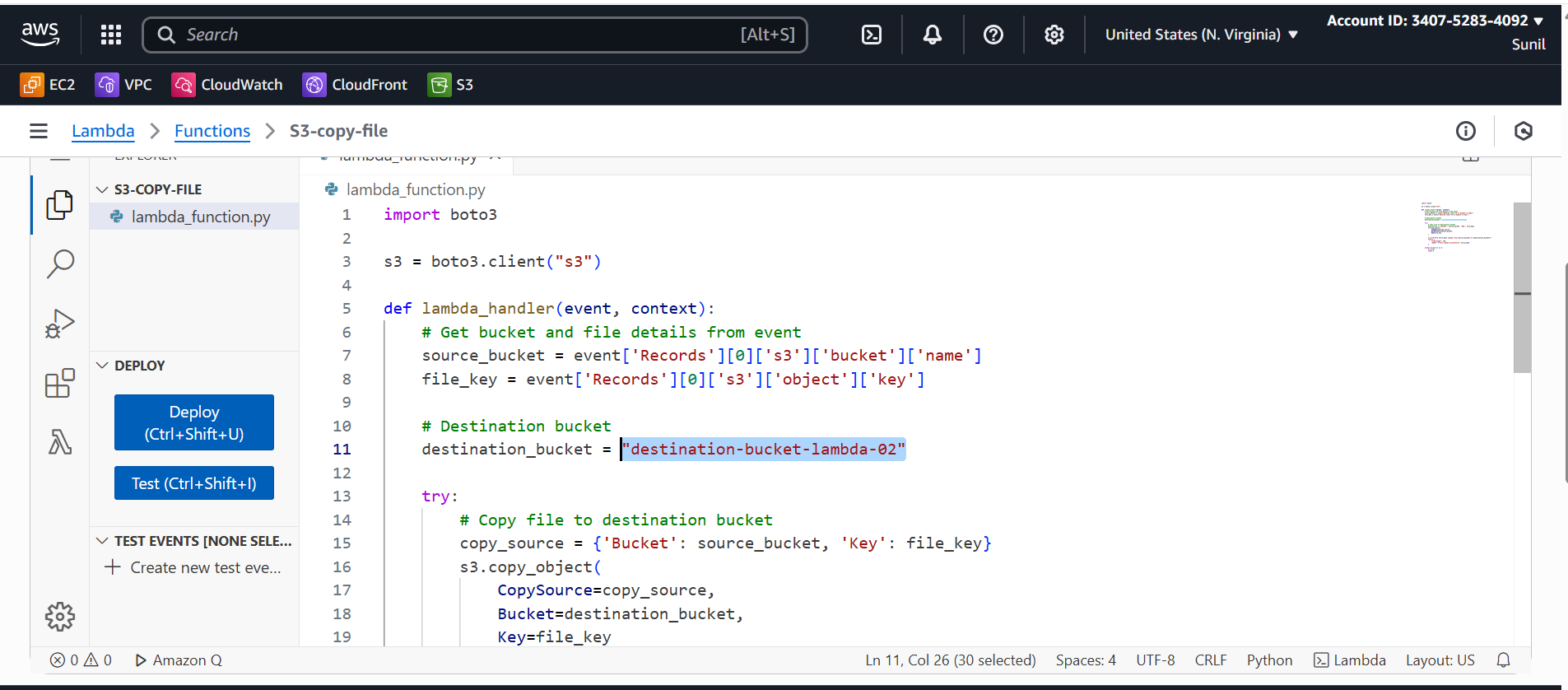


## Step 3: Add S3 Trigger

1. Open your Lambda function → **Add trigger**.
2. Select **S3** → choose [lambda-source-bucket-0001](https://us-east-1.console.aws.amazon.com/s3/buckets/lambda-source-bucket-0001?region=us-east-1&bucketType=general).
3. Event type: **PUT** (on object upload).
4. Save.



Step 4: Add Lambda Code



## Step 5: Test It

1. Upload any file (e.g., test.txt) into your source bucket
2. Lambda will automatically trigger.
3. Check your destination-bucket