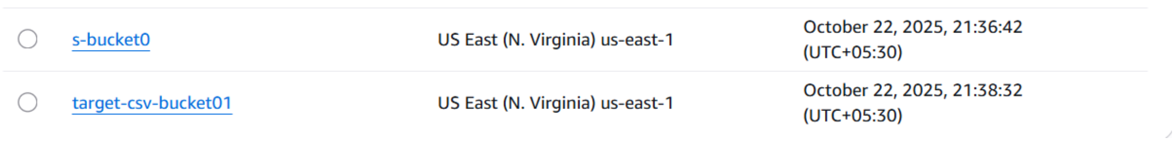
Event-Driven Ingestion with AWS Lambda

To Begin with the Lab

Summary of the Lab

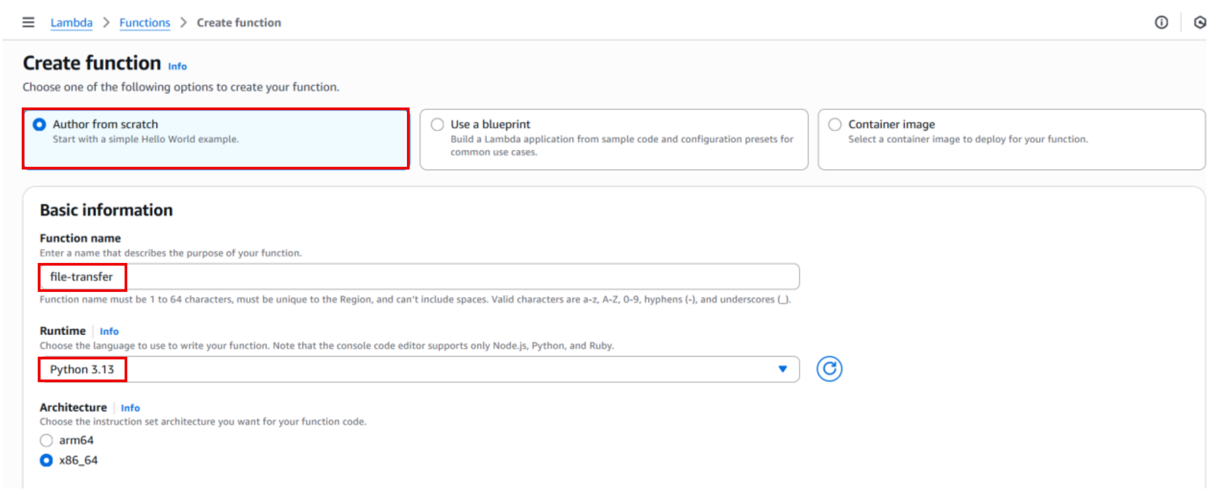
In this lab, you created an **event-driven data ingestion pipeline** using **AWS Lambda and Amazon S3**. Two S3 buckets were created — one as the **source bucket** (for uploads) and the other as the **target bucket** (for storing moved files). A **Lambda function** was built using **Python 3.x** to automatically trigger when a new file is uploaded to the source bucket. The function copies the file from the source to the target bucket. Necessary IAM permissions were granted using the **AmazonS3FullAccess** policy. Finally, the setup was tested and verified by uploading a file and confirming its successful transfer.

- Sign in to the **AWS Management Console**.
- Navigate to **S3**
- Create two S3 Buckets
- Click **Create bucket**
- S-bucket0 → source bucket.
- target-csv-bucket01→ target bucket, this will store the moved files.



○	s-bucket0	US East (N. Virginia) us-east-1	October 22, 2025, 21:36:42 (UTC+05:30)
○	target-csv-bucket01	US East (N. Virginia) us-east-1	October 22, 2025, 21:38:32 (UTC+05:30)

- Now we will create a lambda function.
- Navigate to **AWS Lambda** → click **Create function**.
- Select **Author from scratch**.
- Enter a function name
- Choose **Runtime**: Python 3.x.
- Click **Create function**.



Lambda > Functions > Create function

Create function Info

Choose one of the following options to create your function.

[✓ 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.

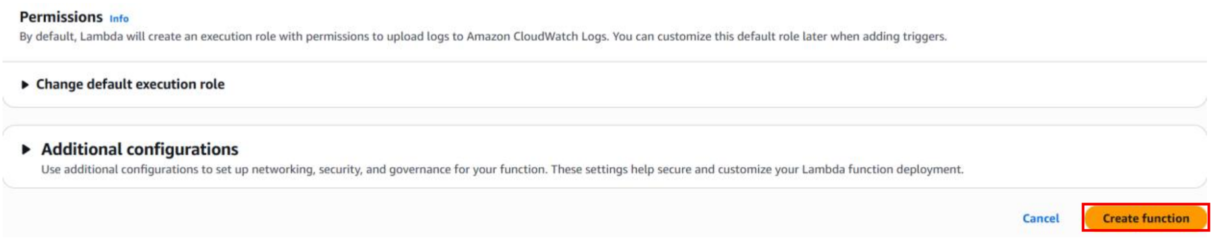
Container image — Select a container image to deploy for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.
file-transfer
Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).

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 3.13

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



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.

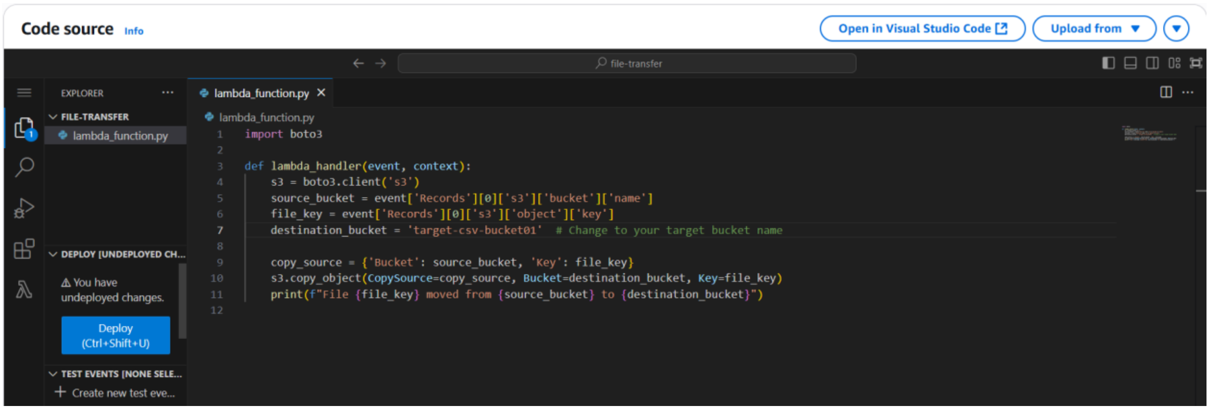
▶ Change default execution role

▶ Additional configurations

Use additional configurations to set up networking, security, and governance for your function. These settings help secure and customize your Lambda function deployment.

- In the **Code source** editor, paste the following Python code

```
import boto3
def lambda_handler(event, context):
    s3 = boto3.client('s3')
    source_bucket = event['Records'][0]['s3']['bucket']['name']
    file_key = event['Records'][0]['s3']['object']['key']
    destination_bucket = 'target-bucket-12345'  # Change to your target bucket name
    copy_source = {'Bucket': source_bucket, 'Key': file_key}
    s3.copy_object(CopySource=copy_source, Bucket=destination_bucket, Key=file_key)
    print(f"File {file_key} moved from {source_bucket} to {destination_bucket}")
```



⚠ You have undeployed changes.

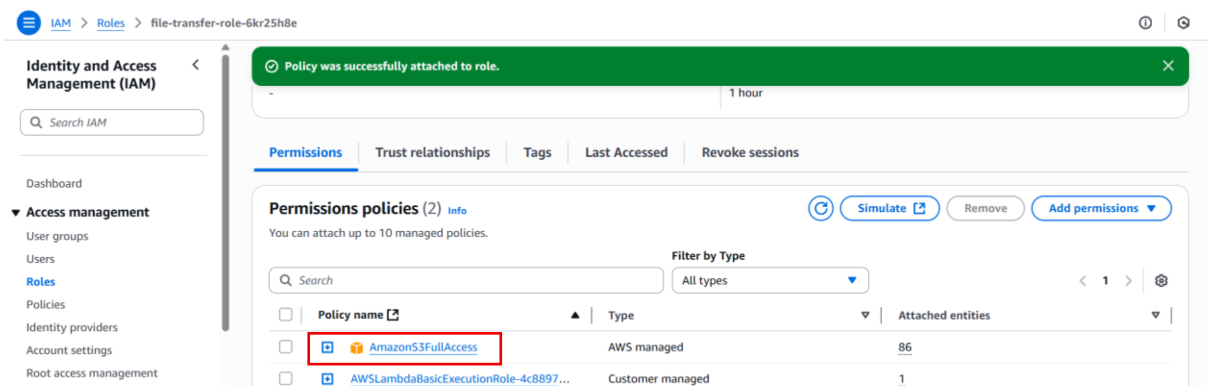
lambda_function.py

```
import boto3

def lambda_handler(event, context):
    s3 = boto3.client('s3')
    source_bucket = event['Records'][0]['s3']['bucket']['name']
    file_key = event['Records'][0]['s3']['object']['key']
    destination_bucket = 'target-csv-bucket01'  # Change to your target bucket name

    copy_source = {'Bucket': source_bucket, 'Key': file_key}
    s3.copy_object(CopySource=copy_source, Bucket=destination_bucket, Key=file_key)
    print(f"File {file_key} moved from {source_bucket} to {destination_bucket}")
```

- Go to the **Configuration** tab → **Permissions** → click the execution role name.
- In the IAM console, click **Add permissions** → **Attach policies**.
- Search and attach the **AmazonS3FullAccess** policy (for this lab only).
- Return to Lambda once the policy is attached.



Policy was successfully attached to role.

-	1 hour

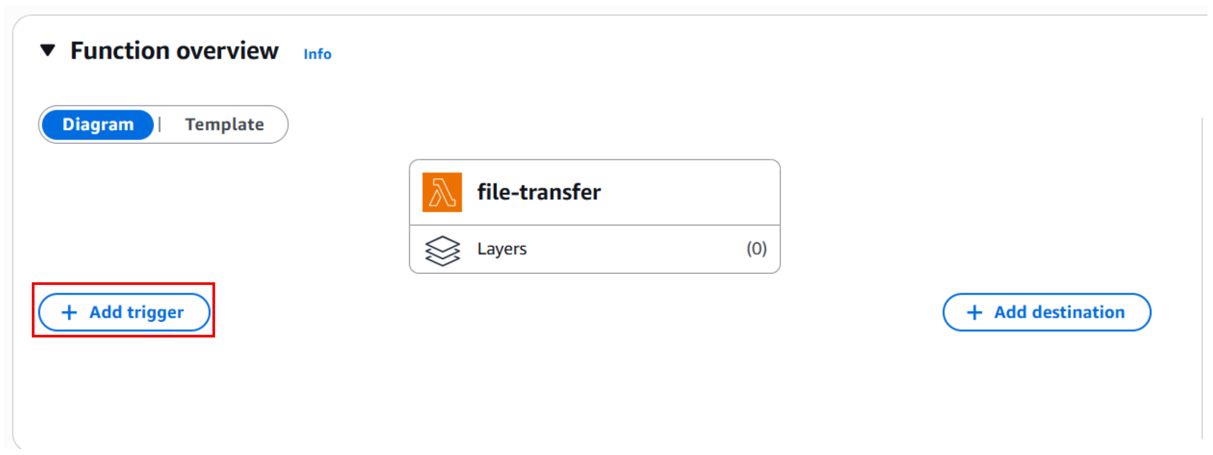
Permissions policies (2) Info

You can attach up to 10 managed policies.

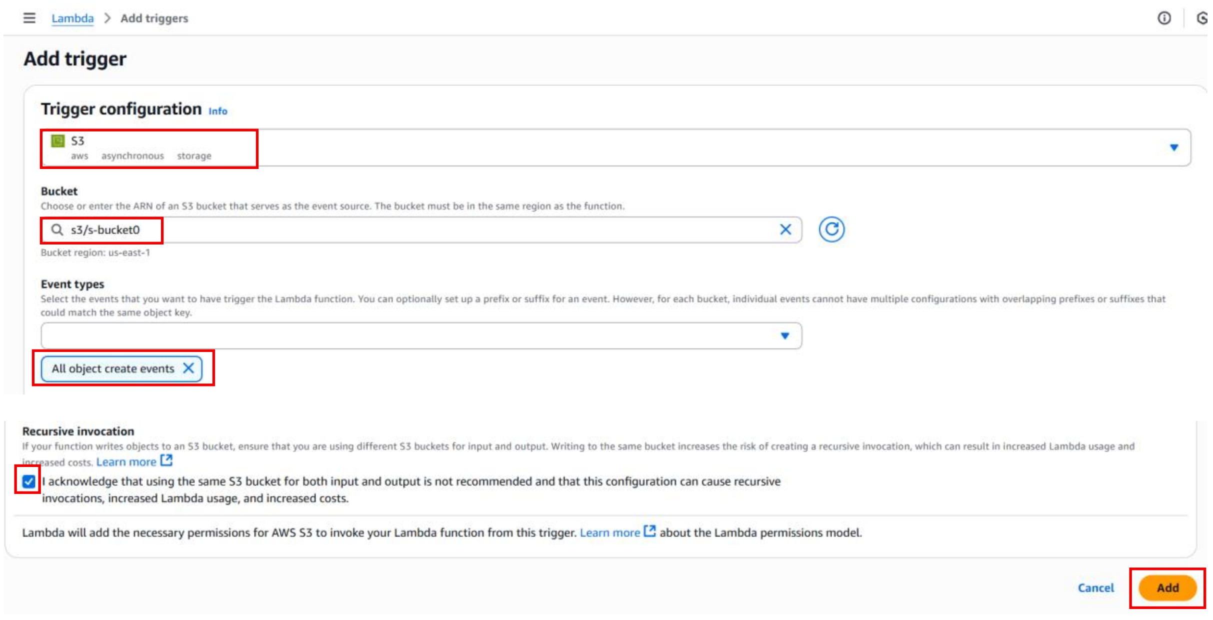
Filter by Type: All types

Policy name	Type	Attached entities
AmazonS3FullAccess	AWS managed	86
AWSLambdaBasicExecutionRole-4c8897...	Customer managed	1

- In the Lambda console, open your function → click **Add trigger**.



- Choose **S3** as the trigger source.
- Select the **source bucket** created earlier.
- Check the acknowledgment box and click **Add**.



Trigger configuration

S3 (aws, asynchronous, storage)

Bucket

Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.

s3/s-bucket0

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

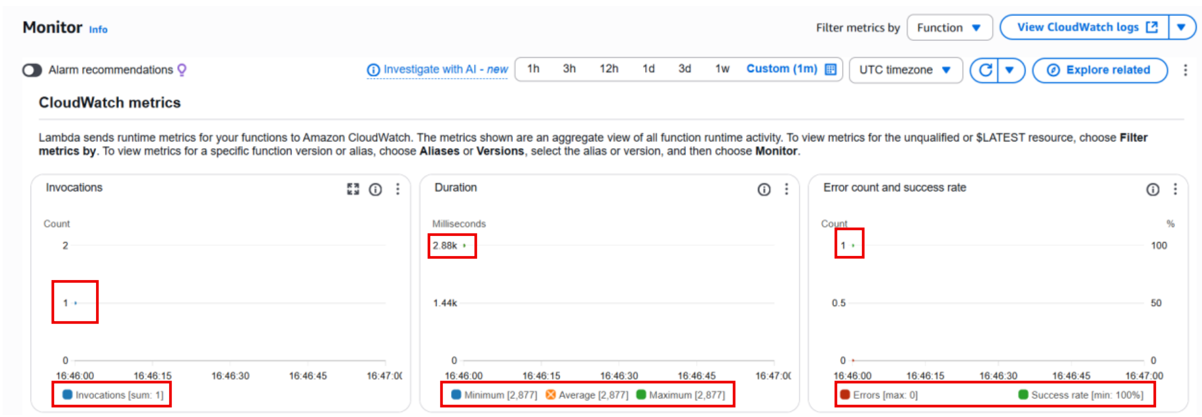
Recursive invocation

If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. Learn more

[✓] I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.

Lambda will add the necessary permissions for AWS S3 to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

- In the **Lambda** → **Monitor** tab, check **Invocations**, **Duration**, and **Errors**.
- You can see the little dot.



Monitor Info

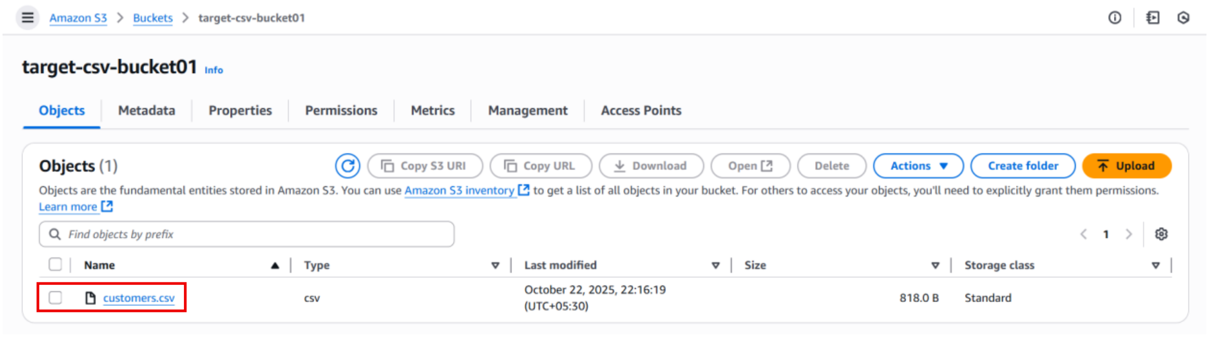
Filter metrics by Function ▼ | View CloudWatch logs

Alarm recommendations | Investigate with AI - new | 1h 3h 12h 1d 3d 1w Custom (1m) | UTC timezone ▼ | Explore related

CloudWatch metrics

Lambda sends runtime metrics for your functions to Amazon CloudWatch. The metrics shown are an aggregate view of all function runtime activity. To view metrics for the unqualified or $LATEST resource, choose **Filter metrics by**. To view metrics for a specific function version or alias, choose **Aliases** or **Versions**, select the alias or version, and then choose **Monitor**.

- The file has successfully move from this folder to a new folder.



Amazon S3 > Buckets > target-csv-bucket01

target-csv-bucket01 Info

Objects | Metadata | Properties | Permissions | Metrics | Management | Access Points

Objects (1)

Copy S3 URI | Copy URL | Download | Open | Delete | Actions ▼ | Create folder | Upload

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

Find objects by prefix

Name	Type	Last modified	Size	Storage class
customers.csv	csv	October 22, 2025, 22:16:19 (UTC+05:30)	818.0 B	Standard