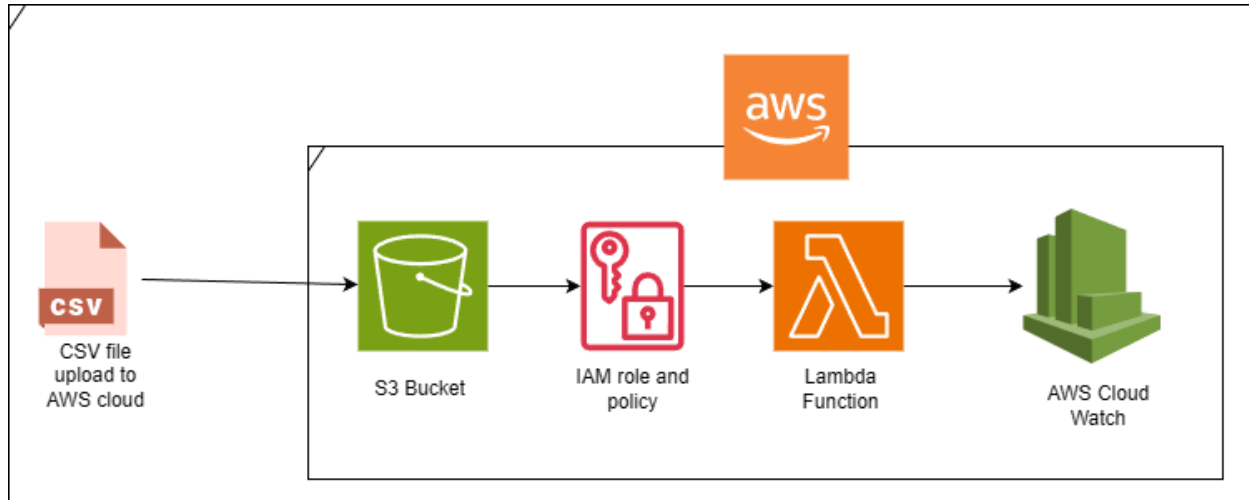


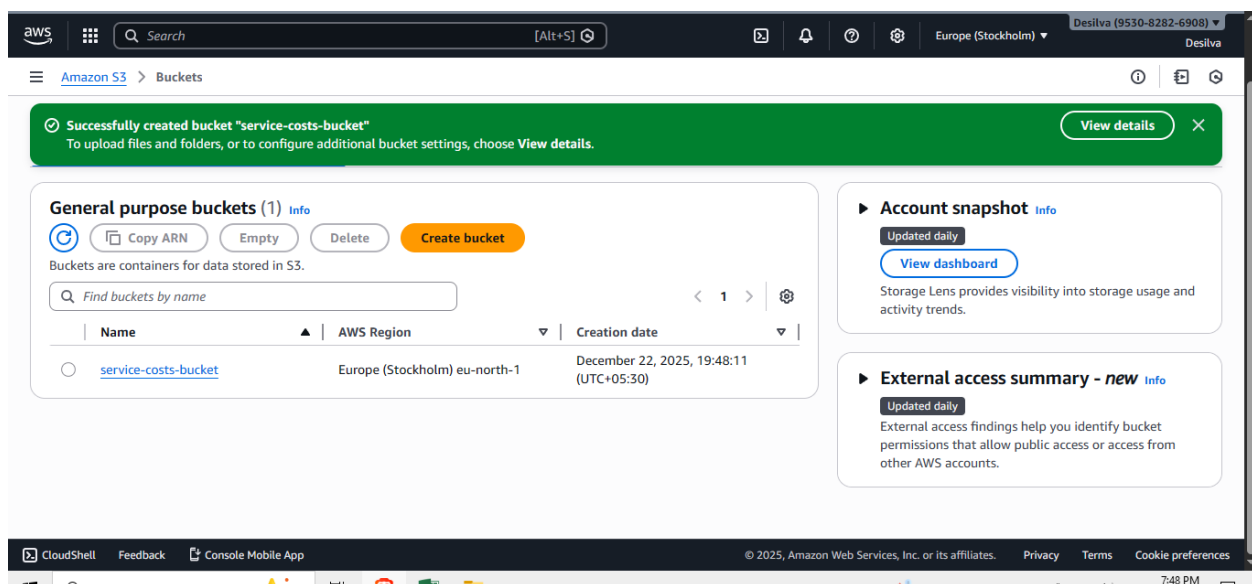


AWS S3 + Lambda + Cloud Watch Project

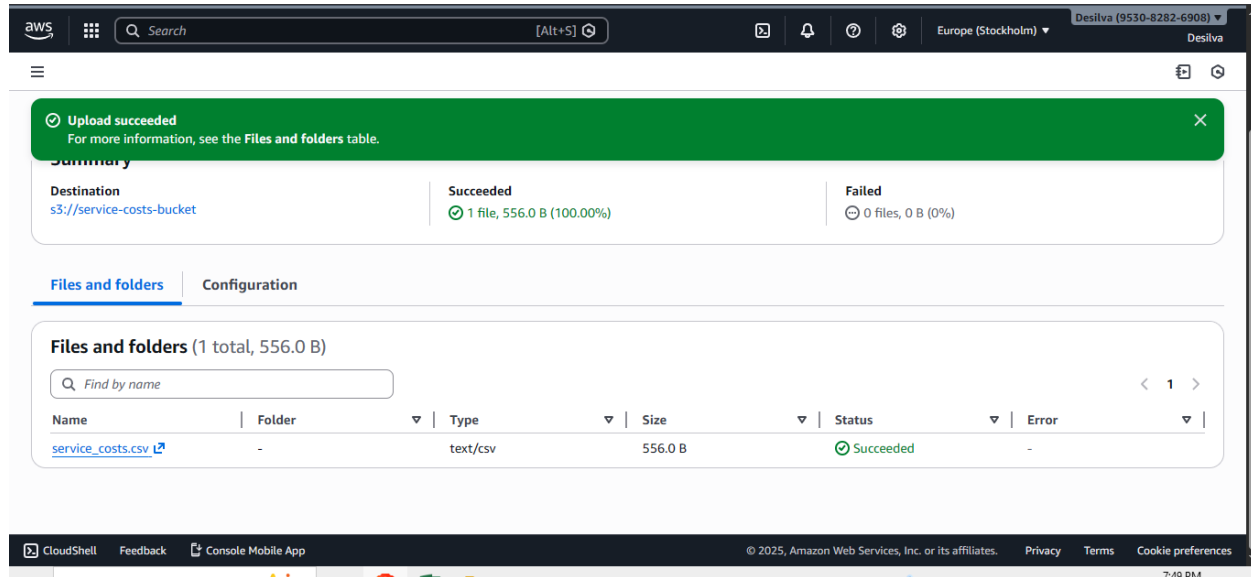


Step by Step how to Create that Project

- ❖ **Steps 1:-** Create a S3 bucket and upload CSV file in to the bucket,
 - In the AWS S3 console, create a new bucket with a unique name, specify the AWS Region, adjust the "Block all public access" setting as required, and proceed to create the bucket.



➤ Upload the CSV file into your S3 bucket



❖ **Steps 2 :** - Create IAM policy for Lambda and S3

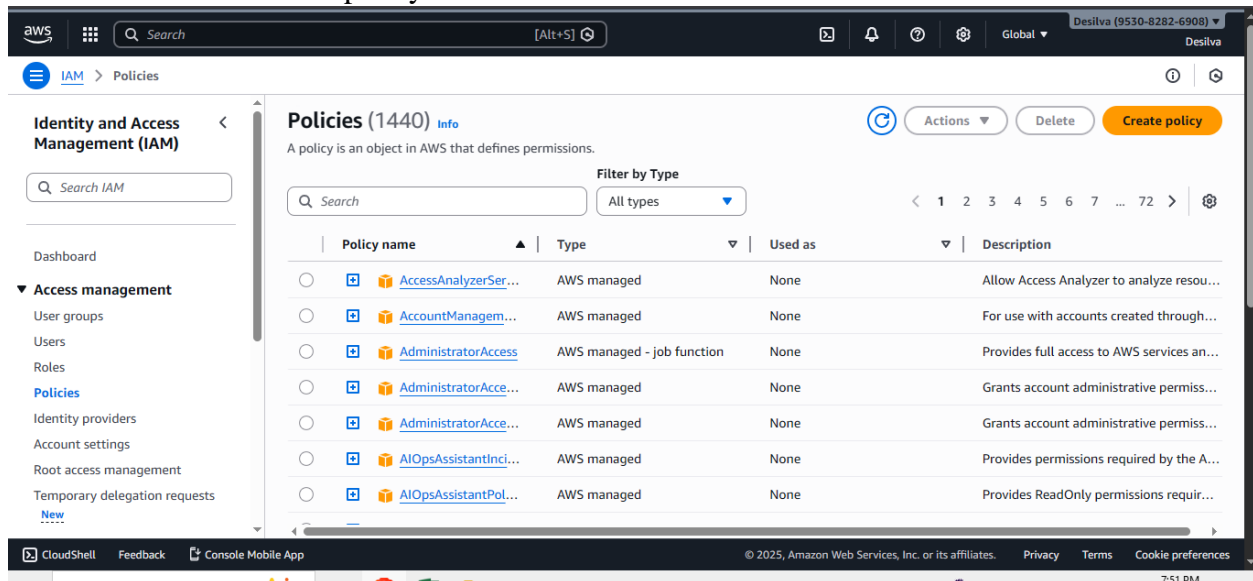
- Go to the IAM Console, click Policies, then Create policy. Click the JSON tab, paste the code below, and name it LambdaS3ReadPolicy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": "arn:aws:s3:::service-costs-bucket/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "*"
    }
  ]
}
```

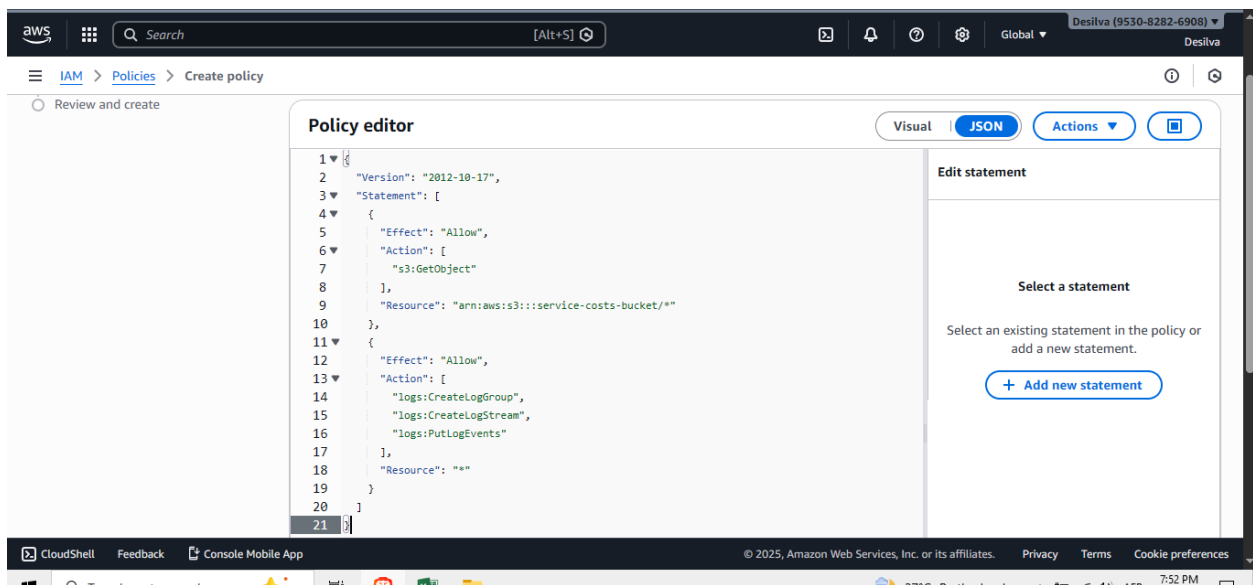
- Make sure, put the S3 bucket name under the Resource.

< "Resource": "arn:aws:s3:::service-costs-bucket/*" >

➤ Create IAM policy



➤ Attached S3 bucket policy json type code here,



- After completing the steps above, create the S3 policy.

➤ Provide S3 policy name,

The screenshot shows the 'Review and create' step in the AWS IAM console. The left sidebar indicates 'Step 2: Review and create'. The main content area is titled 'Review and create' with a sub-header 'Policy details'. Under 'Policy name', there is a text input field containing 'LambdaS3ReadPolicy'. Below it, a description field is empty. The 'Permissions defined in this policy' section is visible at the bottom, with an 'Edit' button. The footer shows the AWS logo, search bar, and navigation links.

Step 1
Specify permissions

Step 2
Review and create

Review and create [Info](#)

Review the permissions, specify details, and tags.

Policy details

Policy name
Enter a meaningful name to identify this policy.

Maximum 128 characters. Use alphanumeric and '+', '@', '-' characters.

Description - optional
Add a short explanation for this policy.

Maximum 1,000 characters. Use alphanumeric and '+', '@', '-' characters.

Permissions defined in this policy [Info](#)

[Edit](#)

CloudShell Feedback Console Mobile App © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Time here to search Hot days ahead AFR 7:53 PM

➤ Create the policy,

The screenshot shows the 'Add tags' step in the AWS IAM console. The left sidebar indicates 'Step 2: Review and create'. The main content area is titled 'Add tags - optional' with a sub-header 'Add tags - optional'. Below it, a table lists the permissions defined in the policy. The 'Add new tag' button is visible. The footer shows the AWS logo, search bar, and navigation links.

Add tags - optional [Info](#)

Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

[Add new tag](#)

You can add up to 50 more tags.

[Cancel](#) [Previous](#) [Create policy](#)

Service	Access level	Resource	Request condition
CloudWatch Logs	Limited: Write	All resources	None
S3	Limited: Read	BucketName string like [service-costs-bucket, ObjectPath string like [All	None

CloudShell Feedback Console Mobile App © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Type here to search Hot days ahead AFR 7:54 PM 12/22/2024

- Navigate to the IAM Console, click Roles, then click Create role. Select AWS service as the trusted entity, choose Lambda as the use case, and click Next. Attach the policy named **LambdaS3ReadPolicy**. Proceed to the next step, provide the role name **LambdaS3TriggerRole**, and finally click Create role.

➤ Create role for Lambda

The screenshot shows the AWS IAM console 'Roles' page. The left sidebar contains the 'Identity and Access Management (IAM)' menu with options like Dashboard, Access management, User groups, Users, Roles (selected), Policies, Identity providers, Account settings, Root access management, and Temporary delegation requests. The main content area is titled 'Roles (3)' and includes a search bar, a table of existing roles, and a 'Roles Anywhere' section. The table lists three roles: 'AWSServiceRoleForResourceExplorer', 'AWSServiceRoleForSupport', and 'AWSServiceRoleForTrustedAdvisor'. The 'Roles Anywhere' section has a 'Manage' button and three cards: 'Access AWS from your non AWS workloads', 'X.509 Standard', and 'Temporary credentials'.

Role name	Trusted entities	Last activity
AWSServiceRoleForResourceExplorer	AWS Service: resource-explorer-2 (Service-Linker)	14 minutes ago
AWSServiceRoleForSupport	AWS Service: support (Service-Linker)	-
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service-Linker)	-

➤ Navigate AWS service

The screenshot shows the 'Create role' wizard in the AWS IAM console, specifically Step 1: 'Select trusted entity'. The left sidebar shows the progress: Step 1 (selected), Step 2: 'Add permissions', and Step 3: 'Name, review, and create'. The main content area is titled 'Select trusted entity' and features a 'Trusted entity type' section with five radio button options: 'AWS service' (selected), 'AWS account', 'Web identity', 'SAML 2.0 federation', and 'Custom trust policy'. Each option has a brief description of its function.

Trusted entity type

- ☒ **AWS service**
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- ☐ **AWS account**
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- ☐ **Web identity**
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
- ☐ **SAML 2.0 federation**
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- ☐ **Custom trust policy**
Create a custom trust policy to enable others to perform actions in this account.

Select Lambda and click next

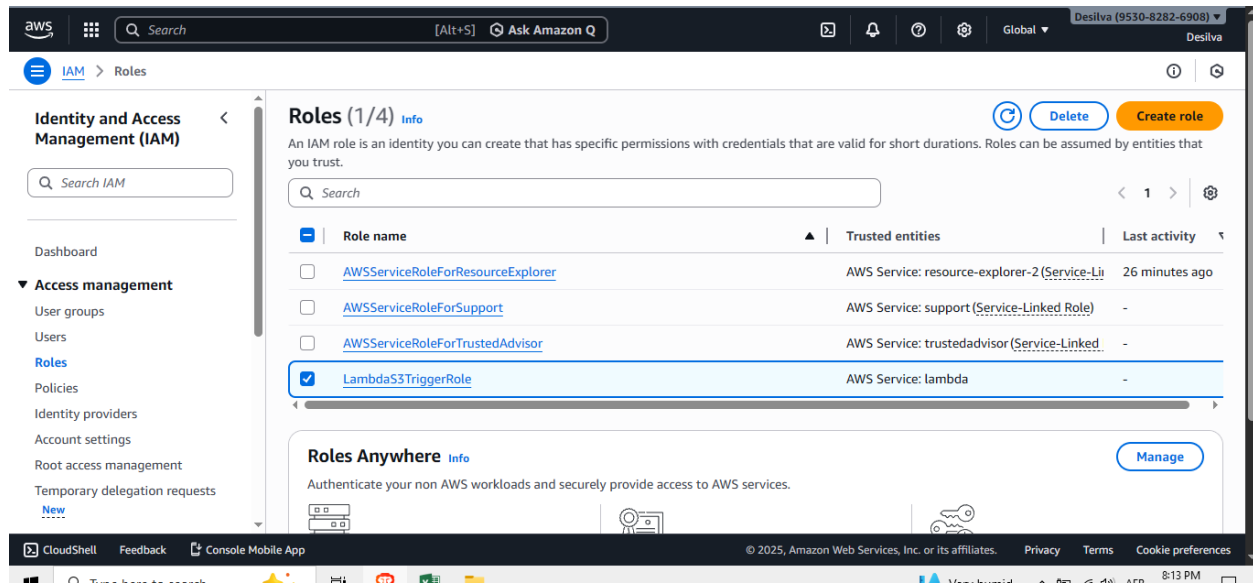
The screenshot shows the 'Use case' selection screen in the AWS IAM console. The 'Service or use case' dropdown is set to 'Lambda'. Under 'Choose a use case for the specified service', the 'Lambda' option is selected, which allows Lambda functions to call AWS services on your behalf. The 'AWSServiceRoleForLambda' option is also visible, which allows Lambda to manage AWS resources on your behalf. 'Cancel' and 'Next' buttons are at the bottom right.

- Attached policy for Lambda (**LambdaS3ReadPolicy**), role name Provide **LambdaS3TriggerRole**

The screenshot shows the 'Add permissions' screen in the AWS IAM console. The 'Permissions policies (1/1110)' section is active, showing a search for 'LambdaS3ReadPolicy' with 1 match. The 'LambdaS3ReadPolicy' is selected. Below this, there is a section for 'Set permissions boundary - optional'. 'Cancel', 'Previous', and 'Next' buttons are at the bottom right.

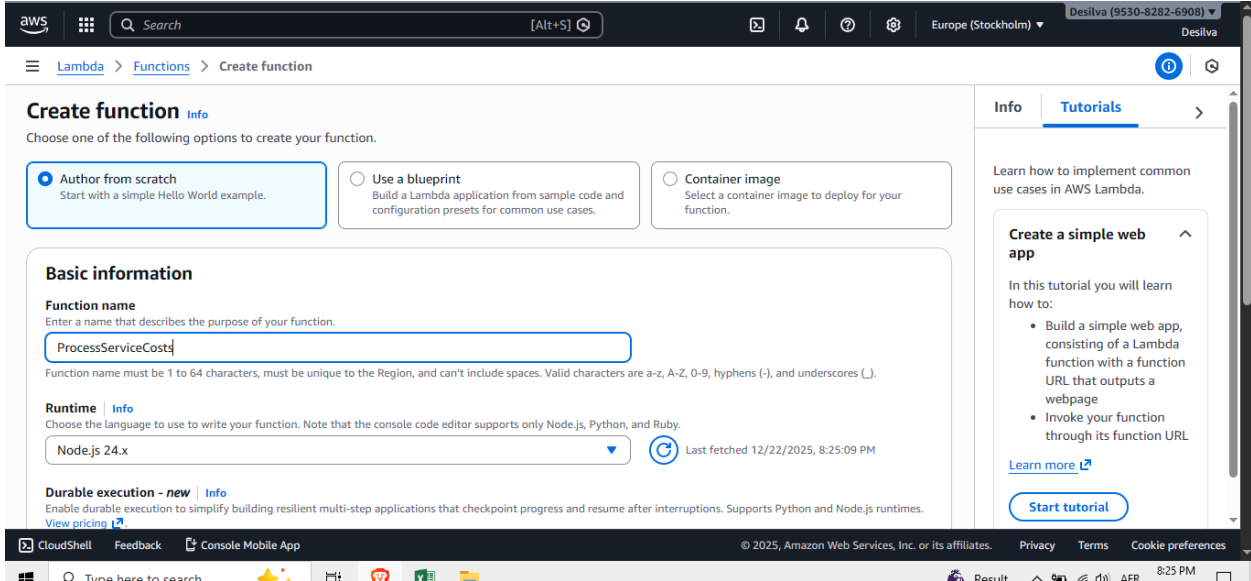
The screenshot shows the 'Name, review, and create' screen in the AWS IAM console. The 'Role name' field is filled with 'LambdaS3TriggerRole'. The 'Description' field is filled with 'Allows Lambda functions to call AWS services on your behalf.' The 'Step 1: Select trusted entities' section is visible at the bottom. 'Edit' and 'Next' buttons are at the bottom right.

➤ Created Lambda Role,



❖ Steps 3 :- Create Lambda Function

- Go to the Lambda Console, create a function from scratch named "ProcessServiceCosts" with a Python 3.9+ runtime, and under Permissions choose "Use existing role" to select the LambdaS3TriggerRole before clicking Create function.



➤ Choose Runtime as a python 3.9

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 3.9



Last fetched 12/22/2025, 8:25:09 PM

Durable execution - [new](#) [Info](#)

Enable durable execution to simplify building resilient multi-step applications that checkpoint progress and resume after interruptions. Supports Python and Node.js runtimes.

how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function

➤ Choose LambdaS3TriggerRole as execution role

Successfully deleted function: ProcessServiceCosts

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

Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

- ☐ Create a new role with basic Lambda permissions
- ☒ Use an existing role
- ☐ Create a new role from AWS policy templates

Existing role

Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

LambdaS3TriggerRole



[View the LambdaS3TriggerRole role](#) on the IAM console.

❖ Step 4 :- In the Code source tab, replace the default code

```
import json
import boto3
import csv
from datetime import datetime

s3 = boto3.client('s3')

def lambda_handler(event, context):
    # Get bucket and file name from event
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = event['Records'][0]['s3']['object']['key']

    # Get CSV file from S3
    response = s3.get_object(Bucket=bucket, Key=key)
    lines = response['Body'].read().decode('utf-8').splitlines()

    # Parse CSV
    reader = csv.DictReader(lines)
    total_cost = 0

    for row in reader:
        cost = float(row['cost_per_hour']) * int(row['total_hours'])
        total_cost += cost
```

```

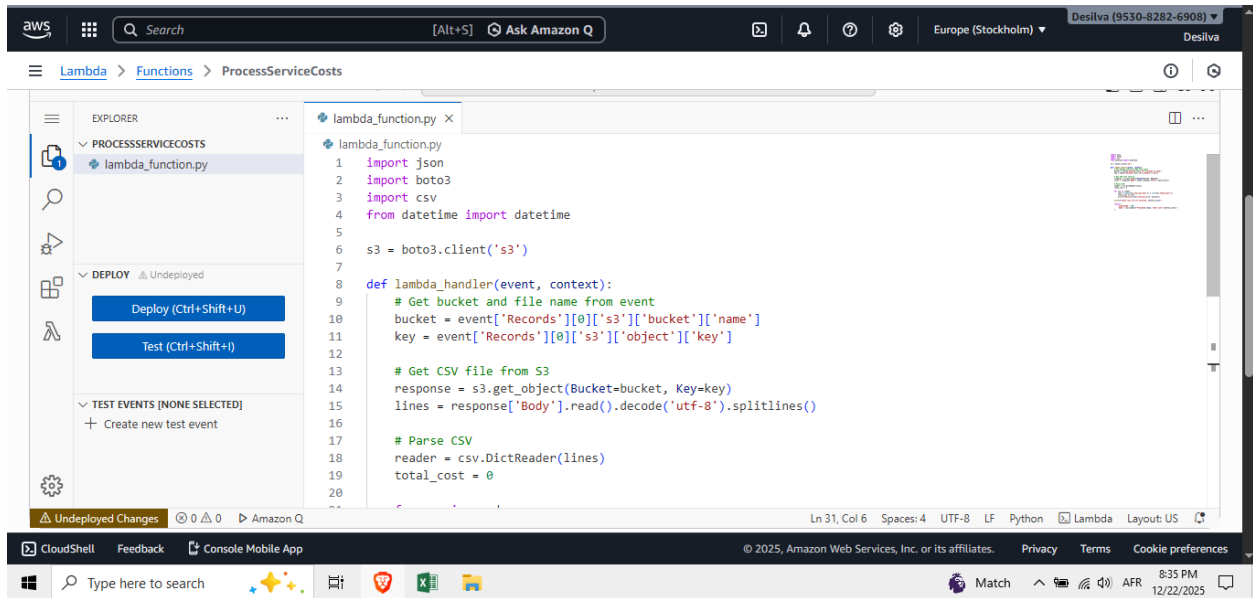
print(f"Service {row['service_id']}: ${cost}")

print(f"Total cost for all services: ${total_cost}")

return {
    'statusCode': 200,
    'body': json.dumps(f'Processed {key}. Total cost: ${total_cost}')
}

```

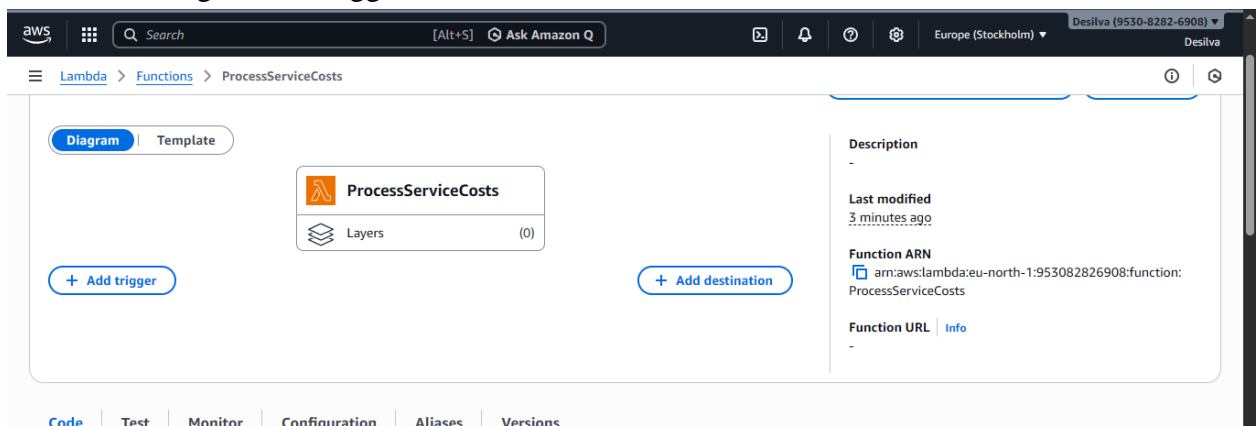
➤ Replace the default code instead existing sample code



❖ Steps 5 :- S3 Bucket trigger using Lambda

- On the Lambda function's Configuration page, add an S3 trigger by selecting your "service-costs-bucket" for all object create events, then click Add.

➤ Navigate Add trigger



➤ Trigger the configuration

Add trigger

Trigger configuration [Info](#)

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.
Q s3/service-costs-bucket X ©
Bucket region: eu-north-1

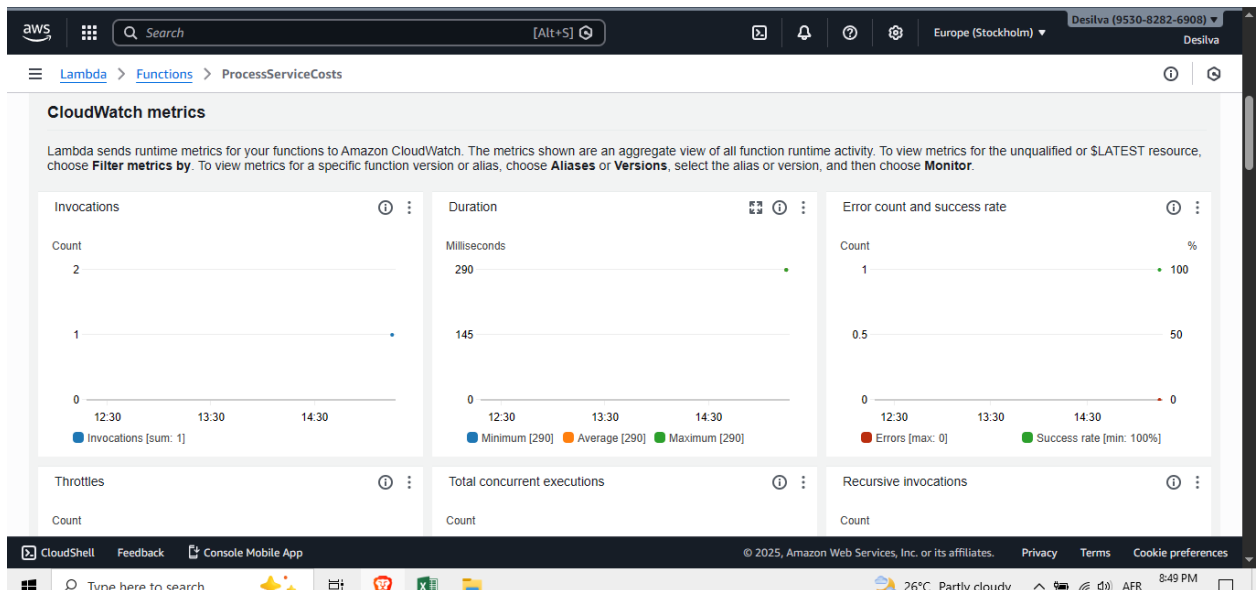
Event types
Select the events that you want to trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.
All object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

Prefix - optional
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters. Any [special characters](#) must be URL encoded.

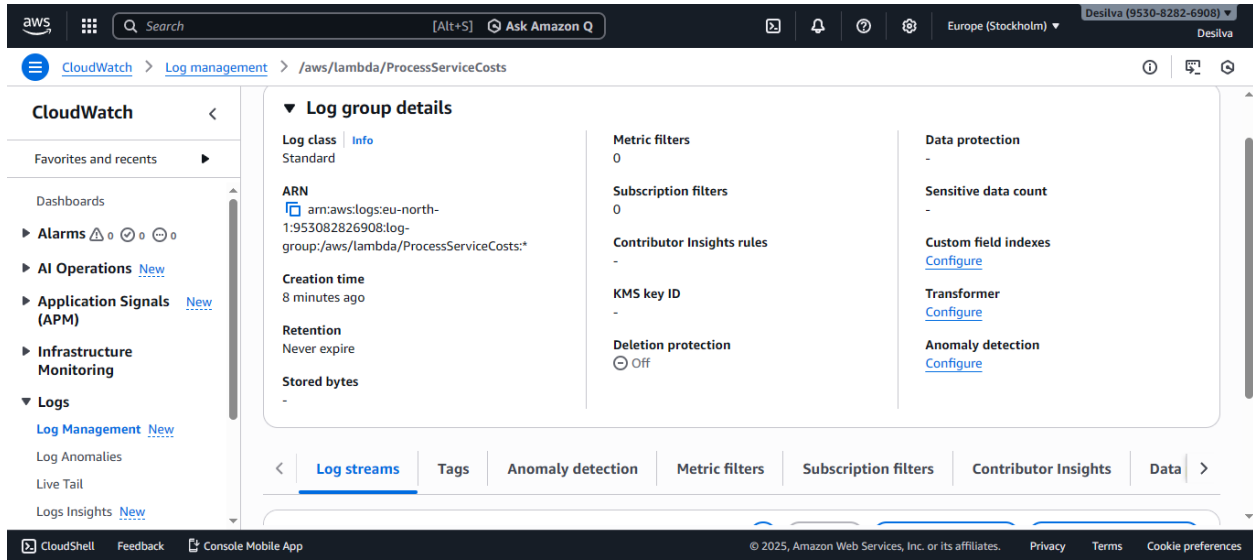
❖ Steps 6 : - Test the Set up

- To test the function, upload the service_costs.csv file again to the S3 bucket and then view its output by navigating to the Lambda's Monitor section and checking the CloudWatch logs.

➤ Monitor CloudWatch using Metrics



➤ Navigate to the Log management under the cloud watch



➤ View the CSV file as log in AWS cloud watch

