COST OPTIMIZATION DELETED THE STALE RESOURCES IN AWS

AWS Lambda is a **serverless computing service** that lets you run code without provisioning or managing servers. It **automatically scales** and runs your code **only when needed**, making it cost-effective.

👉 Why use Lambda?

- No need to manage servers.
- Pay **only for execution time** (milliseconds-based pricing).
- Scales automatically.
- Can be triggered by **CloudWatch Events, S3, DynamoDB, API Gateway, etc.**

🔹 How to Use AWS Lambda for Cost Optimization?

AWS Lambda can **automatically delete stale EBS snapshots and unused volumes**, helping reduce storage costs.

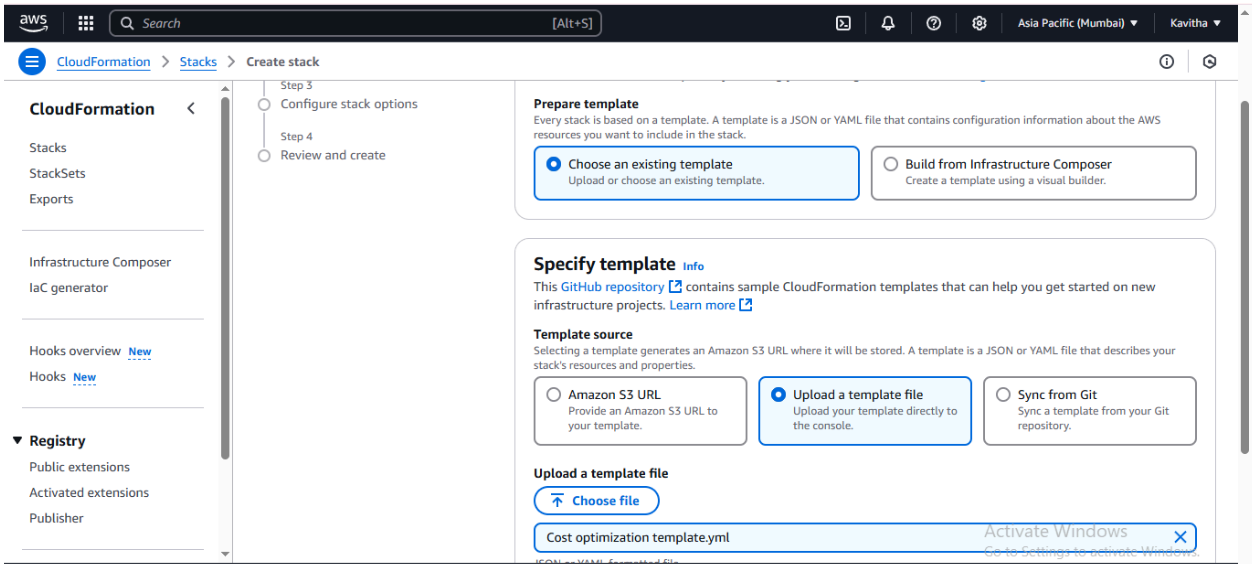
💰 Cost Savings by Cleaning Up Stale Resources:

- **EBS Snapshots**: Old snapshots consume **S3 storage** and cost money.
- **Unused EBS Volumes**: "Available" volumes not attached to an instance still **incur charges**.
- **Unused AMIs**: Old **Amazon Machine Images (AMIs)** create unnecessary snapshots.
- **Orphaned Elastic IPs**: Unattached Elastic IPs are charged.

📌 **Solution**: Use **Lambda + CloudWatch** to periodically **delete** old and unused resources.

Using Cloud Formation Template to create Infrastructure by YAML.

1. Creating a stack for EC2 Instance



Prepare template

Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

● Choose an existing template
Upload or choose an existing template.

○ Build from Infrastructure Composer
Create a template using a visual builder.

Specify template

This GitHub repository contains sample CloudFormation templates that can help you get started on new infrastructure projects. Learn more

Template source

Selecting a template generates an Amazon S3 URL where it will be stored. A template is a JSON or YAML file that describes your stack's resources and properties.

○ Amazon S3 URL
Provide an Amazon S3 URL to your template.

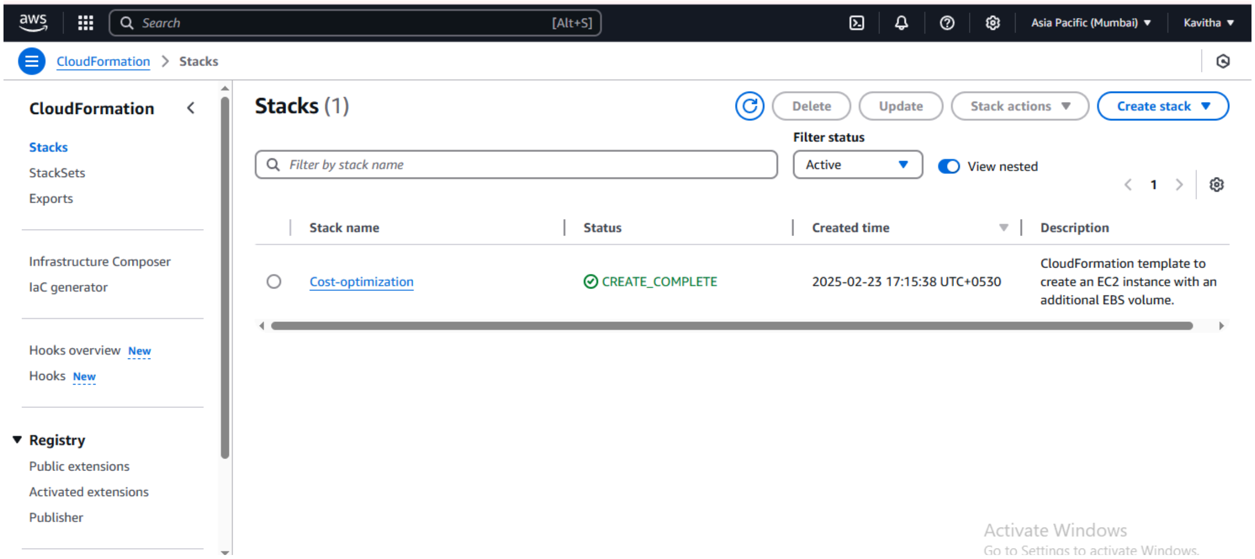
● Upload a template file
Upload your template directly to the console.

○ Sync from Git
Sync a template from your Git repository.

Upload a template file

Cost optimization template.yml

2. A stack created for launching the resources

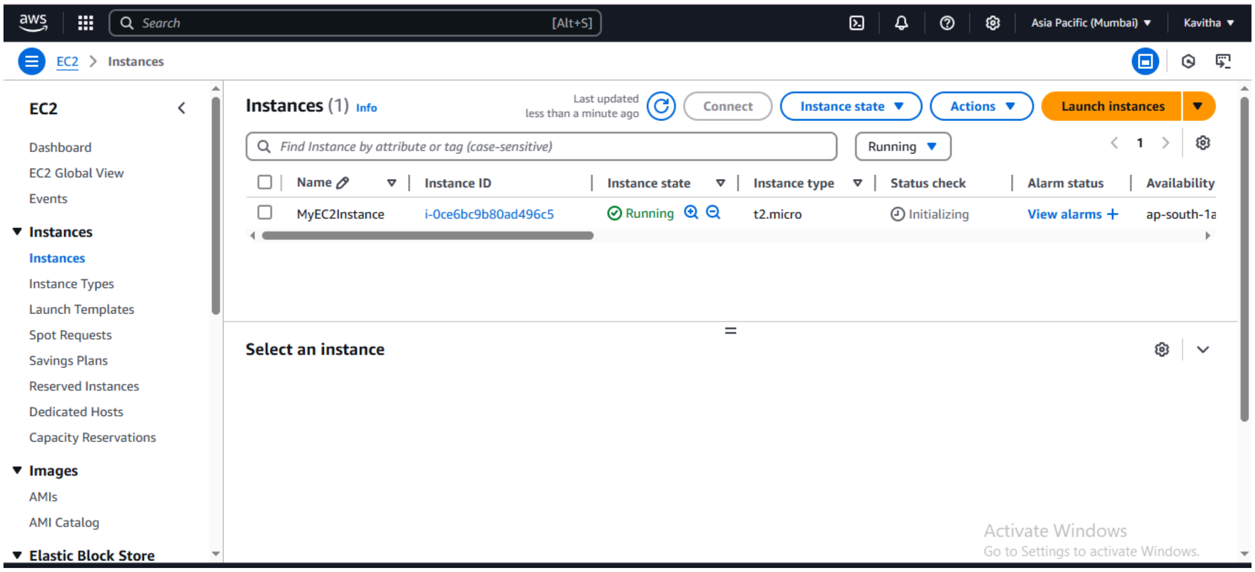


Stacks (1)

Filter status: Active

	Stack name	Status	Created time	Description
○	Cost-optimization	✓ CREATE_COMPLETE	2025-02-23 17:15:38 UTC+0530	CloudFormation template to create an EC2 instance with an additional EBS volume.

3. EC2 Instance got created

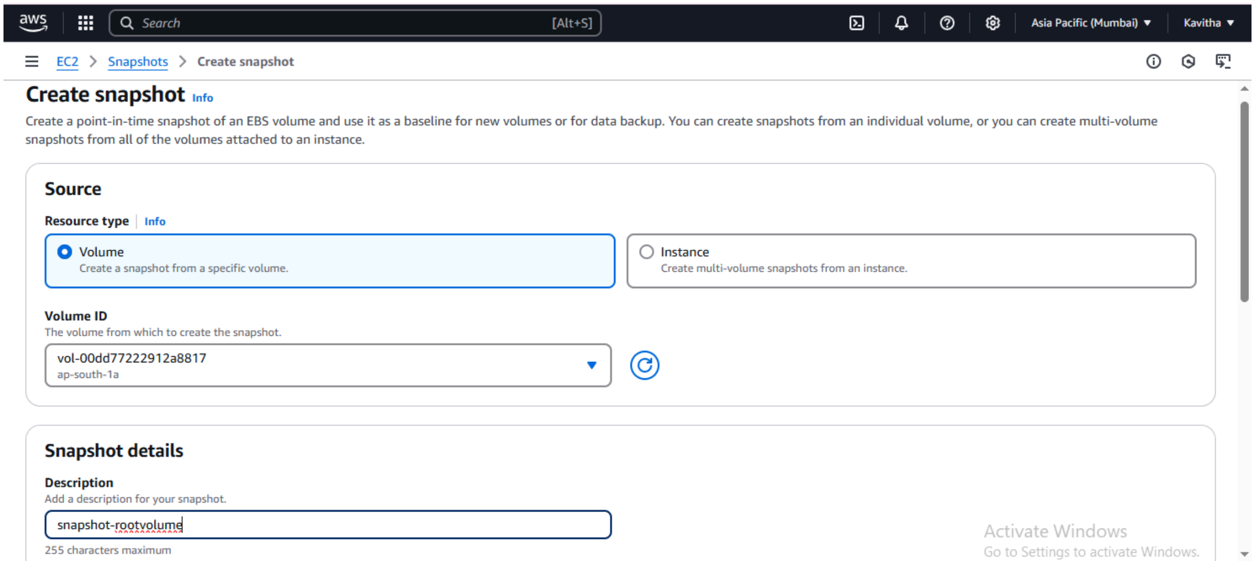


Instances (1) Info

Running

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
MyEC2Instance	i-0ce6bc9b80ad496c5	Running	t2.micro	Initializing	View alarms +	ap-south-1a

4. I am creating a snapshots for root volume


Search [Alt+S]
Asia Pacific (Mumbai)
Kavitha
EC2 > Snapshots > Create snapshot
Activate Windows
Go to Settings to activate Windows.

Create snapshot Info

Create a point-in-time snapshot of an EBS volume and use it as a baseline for new volumes or for data backup. You can create snapshots from an individual volume, or you can create multi-volume snapshots from all of the volumes attached to an instance.

Source

Resource type | Info

● Volume
Create a snapshot from a specific volume.

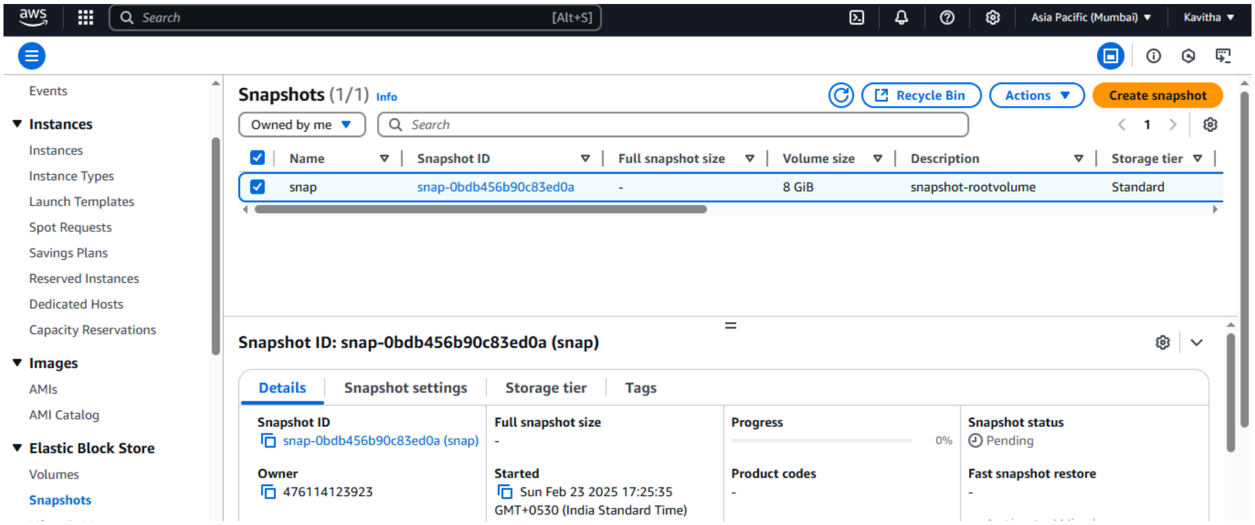
○ Instance
Create multi-volume snapshots from an instance.

Volume ID
The volume from which to create the snapshot.
vol-00dd77222912a8817
ap-south-1a

Snapshot details

Description
Add a description for your snapshot.
snapshot-rootvolume
255 characters maximum

5. Snapshot created for root volume



Snapshots (1/1) Info

☑	Name	Snapshot ID	Full snapshot size	Volume size	Description	Storage tier
☑	snap	snap-0bdb456b90c83ed0a	-	8 GiB	snapshot-rootvolume	Standard

Snapshot ID: snap-0bdb456b90c83ed0a (snap)

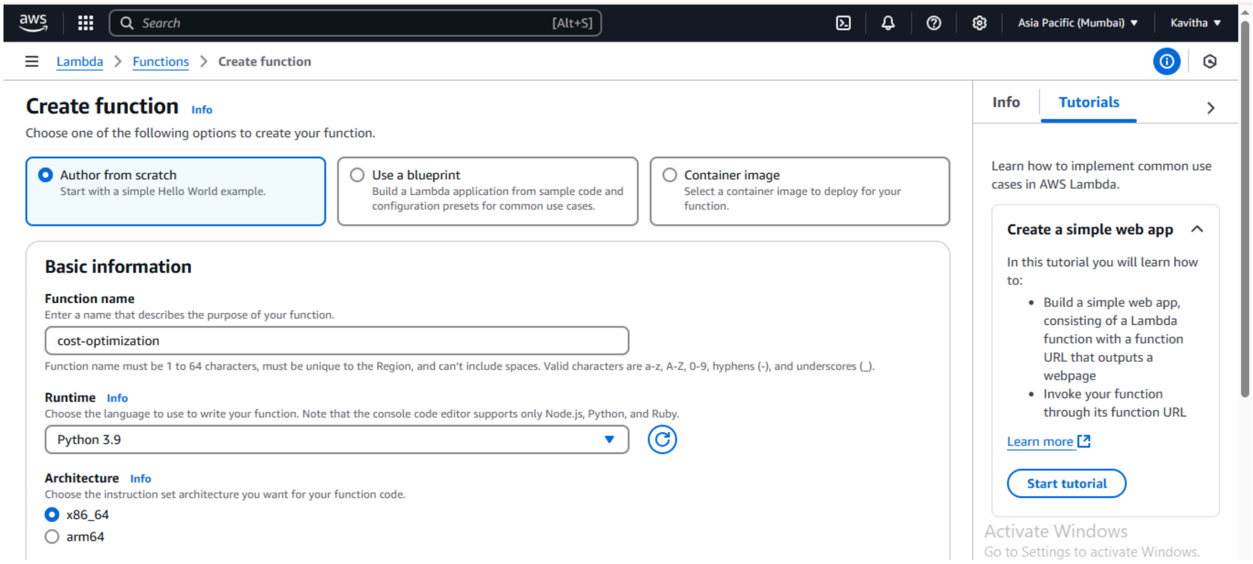
Details | Snapshot settings | Storage tier | Tags

Snapshot ID	Full snapshot size	Progress	Snapshot status
snap-0bdb456b90c83ed0a (snap)	-	0%	Pending
Owner	**Started**	**Product codes**	**Fast snapshot restore**
476114123923	Sun Feb 23 2025 17:25:35 GMT+0530 (India Standard Time)	-	-

🔹 Steps to Automate Deletion of Stale Resources (Snapshots)

1 Create an AWS Lambda Function

1. Go to **AWS Lambda Console** → Click **Create Function**.
2. **Select**: "Author from Scratch".
3. **Function Name**: `Cost-optimization`.
4. **Runtime**: Choose `Python 3.9`.
5. Click **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.
cost-optimization
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 3.9

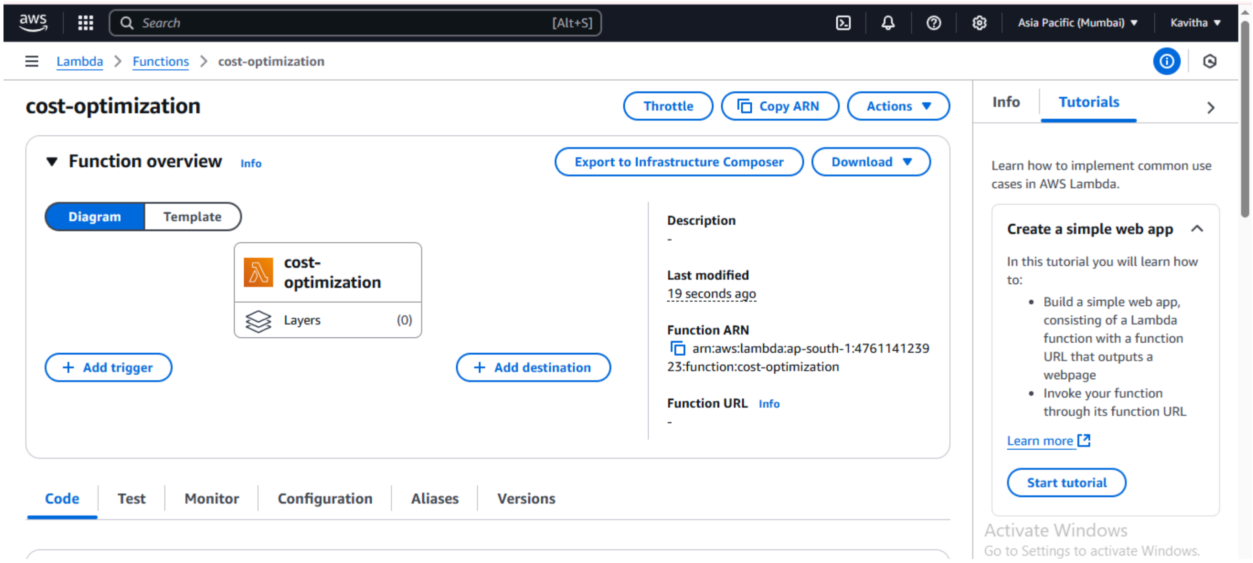
Architecture Info
Choose the instruction set architecture you want for your function code.
● x86_64
○ arm64

Learn how to implement common use cases in AWS Lambda.

Create a simple web app

In this tutorial you will learn how to:

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



cost-optimization

Description
-

Last modified
19 seconds ago

Function ARN
arn:aws:lambda:ap-south-1:476114123923:function:cost-optimization

Function URL
-

Learn how to implement common use cases in AWS Lambda.

In this tutorial you will learn how to:

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

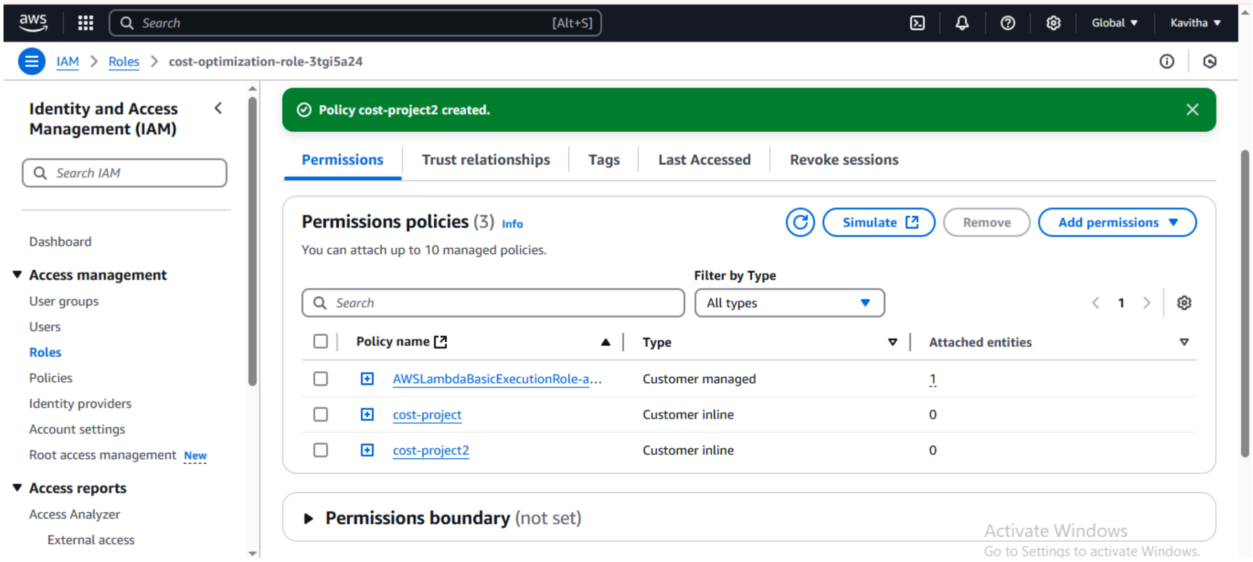
Needed Roles:

For Lambda:

- "ec2:DescribeInstances"
- "ec2:DescribeVolumes"
- "ec2:DescribeSnapshots"
- "ec2:DeleteSnapshot"

For CloudWatch:

- “lambda:InvokeFunction”



Policy cost-project2 created.

Permissions policies (3)

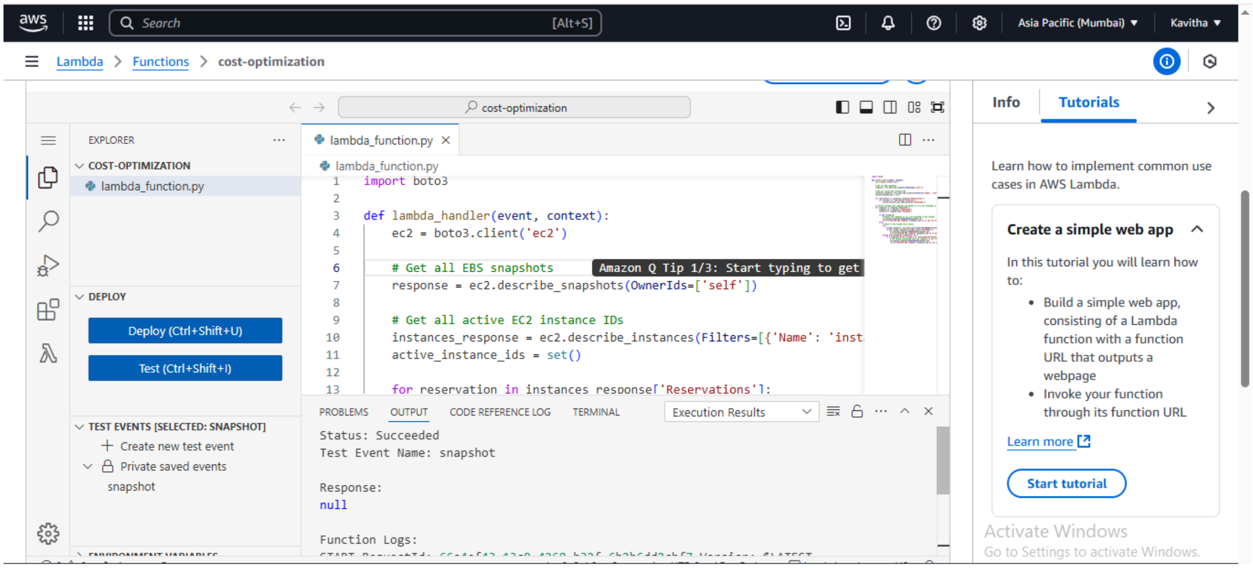
You can attach up to 10 managed policies.

□	Policy name	Type	Attached entities
□	AWSLambdaBasicExecutionRole-a...	Customer managed	1
□	cost-project	Customer inline	0
□	cost-project2	Customer inline	0

2 Add Python Code to Lambda

📌 This function will:

- Find snapshots and delete them.
- Find EBS volumes in "available" state (unused) and delete them.
- Log the cleanup process in CloudWatch Logs.



```
import boto3

def lambda_handler(event, context):
    ec2 = boto3.client('ec2')

    # Get all EBS snapshots
    response = ec2.describe_snapshots(OwnerIds=['self'])

    # Get all active EC2 instance IDs
    instances_response = ec2.describe_instances(Filters=[{'Name': 'inst
    active_instance_ids = set()

    for reservation in instances_response['Reservations']:
```

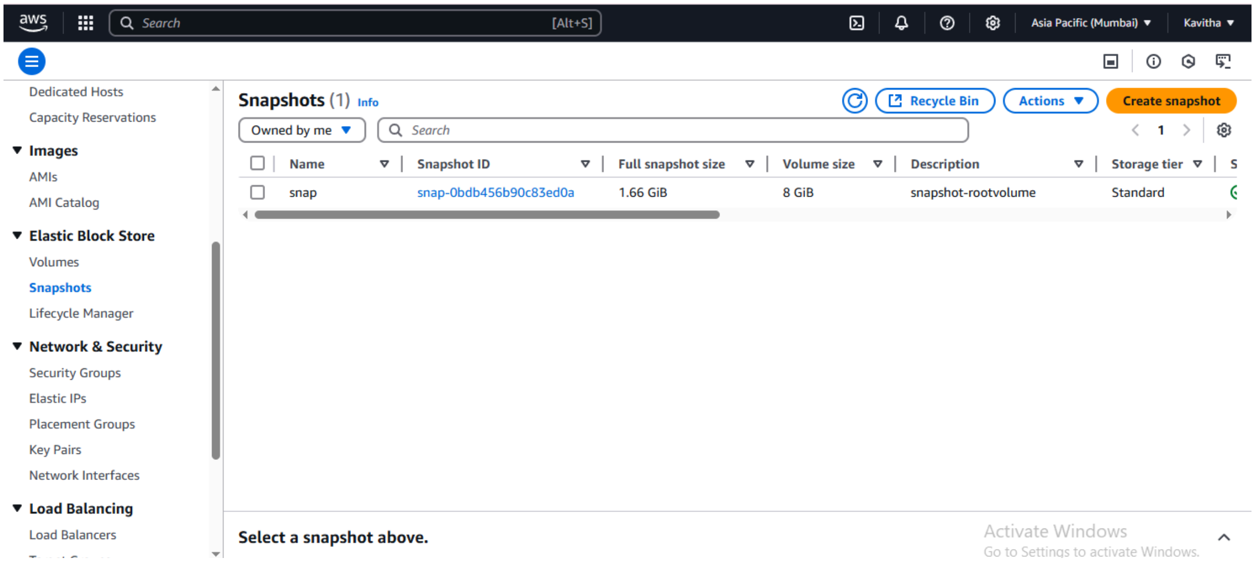
```
Status: Succeeded
Test Event Name: snapshot

Response:
null

Function Logs:
```

Learn how to implement common use cases in AWS Lambda.

In this tutorial you will learn how to:

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

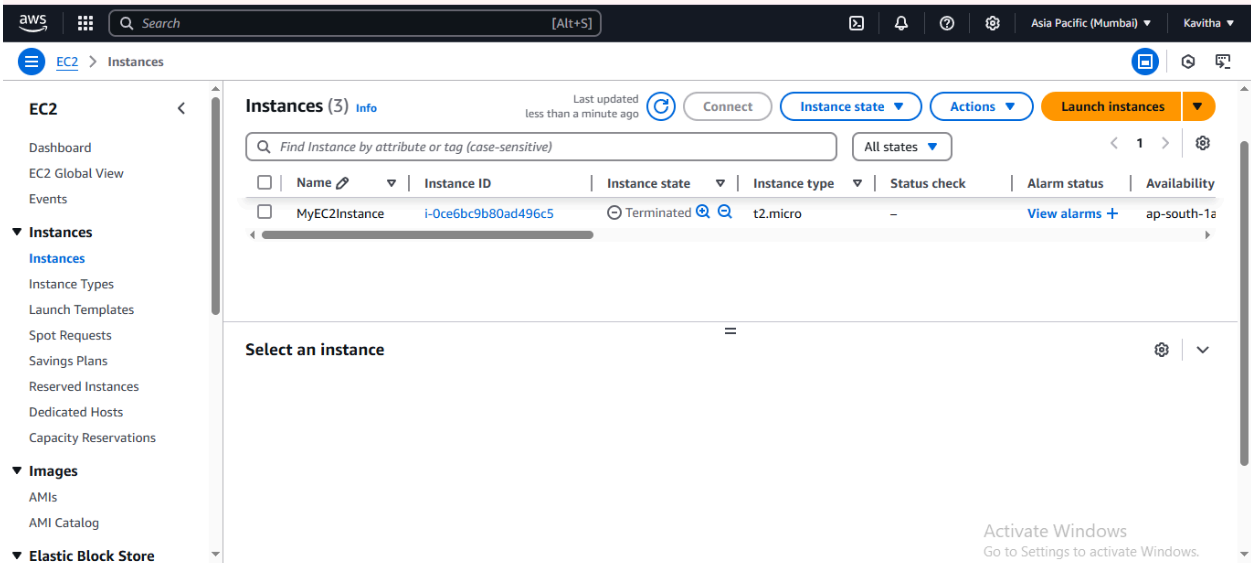
Snapshot was not deleted because its attached with running EC2 Instance



□	Name	Snapshot ID	Full snapshot size	Volume size	Description	Storage tier	S
□	snap	snap-0bdb456b90c83ed0a	1.66 GiB	8 GiB	snapshot-rootvolume	Standard	

Select a snapshot above.

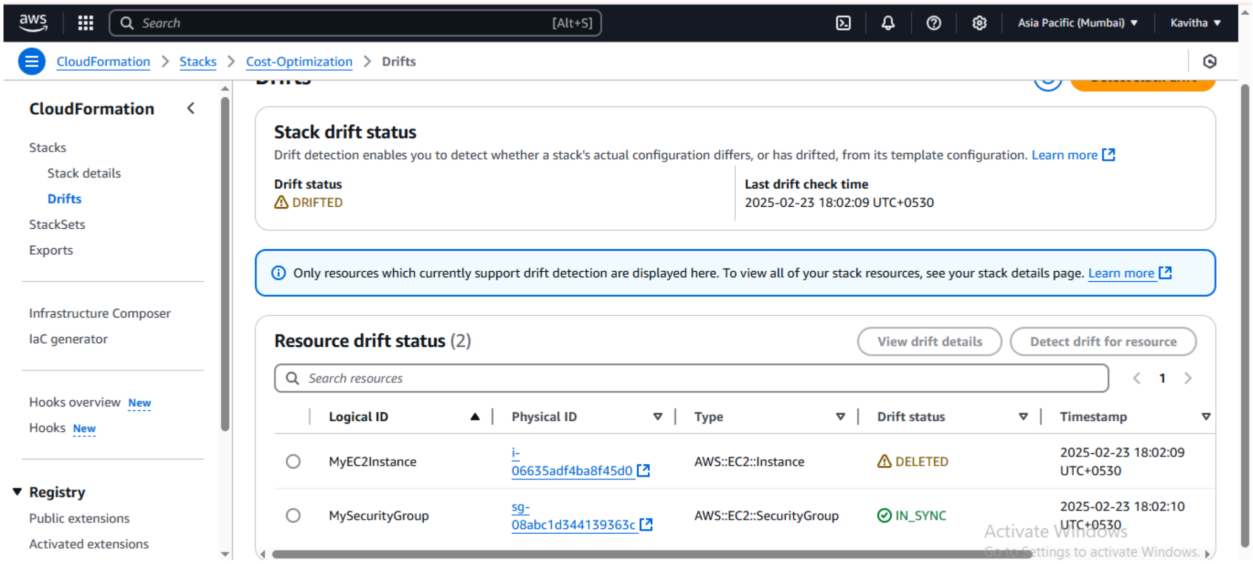
Now, Manually Terminated the EC2 Instance



□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
□	MyEC2Instance	i-0ce6bc9b80ad496c5	Terminated	t2.micro	–	View alarms +	ap-south-1a

Select an instance

And changes shows in **Drift detection** Cloud Formation Template



CloudFormation > Stacks > Cost-Optimization > Drifts

Stack drift status

Drift detection enables you to detect whether a stack's actual configuration differs, or has drifted, from its template configuration. Learn more

Drift status
DRIFTED

Last drift check time
2025-02-23 18:02:09 UTC+0530

Only resources which currently support drift detection are displayed here. To view all of your stack resources, see your stack details page. Learn more

Resource drift status (2)

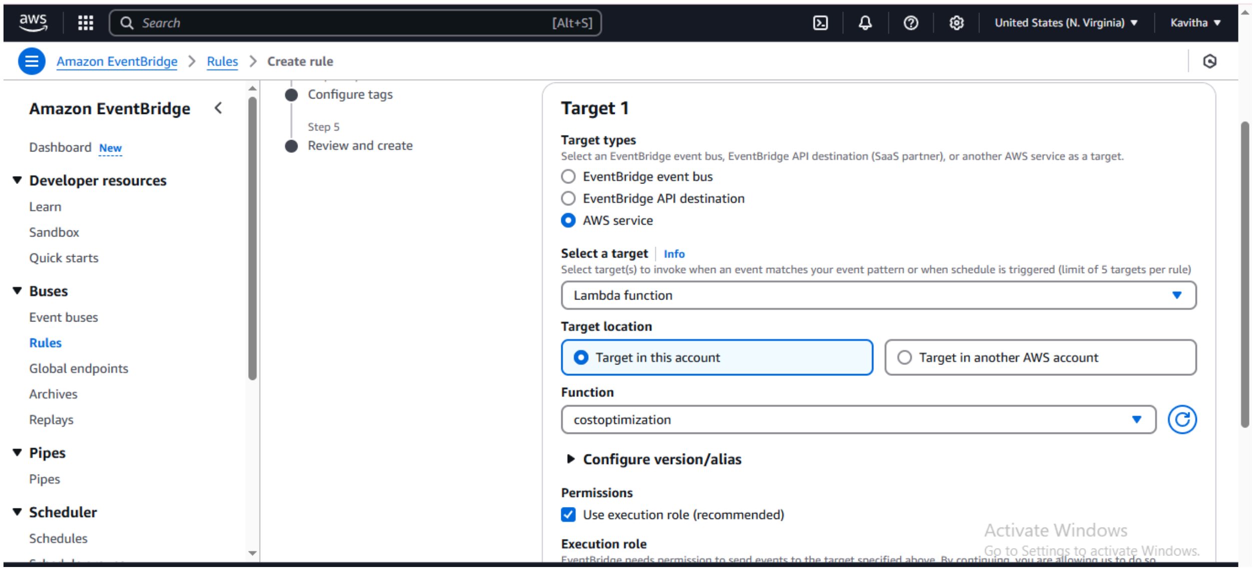
	Logical ID	Physical ID	Type	Drift status	Timestamp
○	MyEC2Instance	i-06635adf4ba8f45d0	AWS::EC2::Instance	DELETED	2025-02-23 18:02:09 UTC+0530
○	MySecurityGroup	sg-08abc1d344139363c	AWS::EC2::SecurityGroup	IN_SYNC	2025-02-23 18:02:10 UTC+0530

3️⃣ Create a CloudWatch Rule (Trigger)

Now, we need to automate this Lambda function to run every day.

Steps:

1. **Go to Amazon EventBridge (CloudWatch).**
2. **Click Rules → Create Rule.**
3. **Name: `lambda trigger`.**
4. **Event Source: Select Schedule.**
5. **Expression Type:**
   - `cron(0/1 0 * * ? *)` **→ Runs at every min UTC.**
6. **Target: Choose AWS Lambda Function.**
7. **Select Function: `Cost-optimization`.**
8. **Click Create Rule.**



Target 1

Target types

Select an EventBridge event bus, EventBridge API destination (SaaS partner), or another AWS service as a target.

○ EventBridge event bus

○ EventBridge API destination

● AWS service

Select a target Info

Select target(s) to invoke when an event matches your event pattern or when schedule is triggered (limit of 5 targets per rule)

Lambda function ▼

Target location

● Target in this account ○ Target in another AWS account

Function

costoptimization ▼

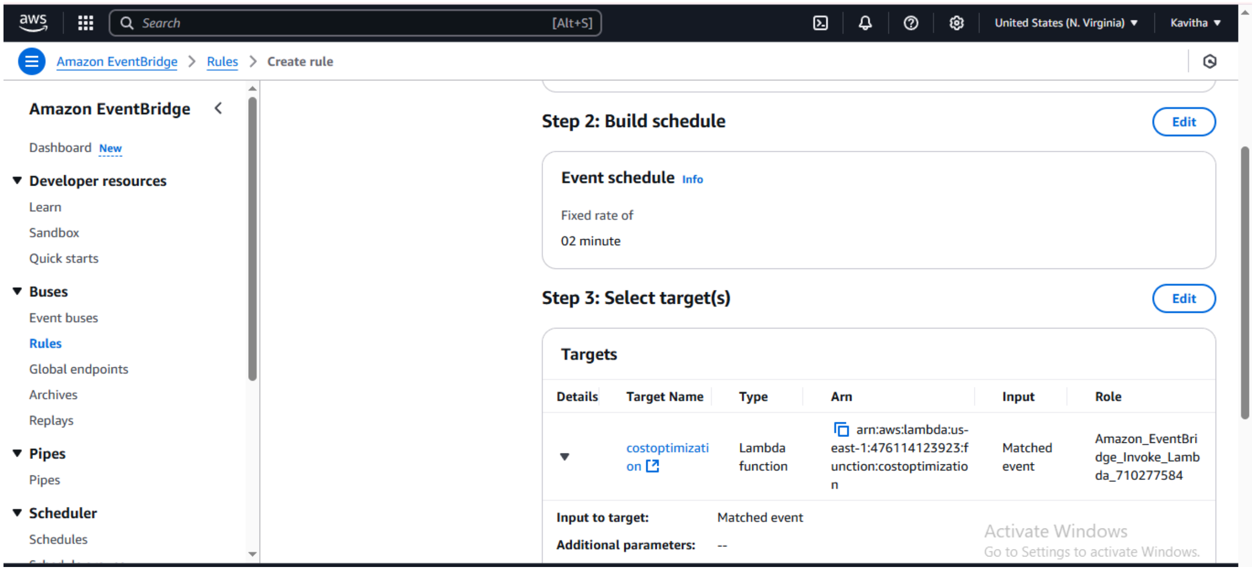
▶ **Configure version/alias**

Permissions

☑ Use execution role (recommended)

Execution role

EventBridge needs permission to send events to the target specified above. By continuing, you are allowing us to do so.



Step 2: Build schedule

Event schedule Info

Fixed rate of

02 minute

Step 3: Select target(s)

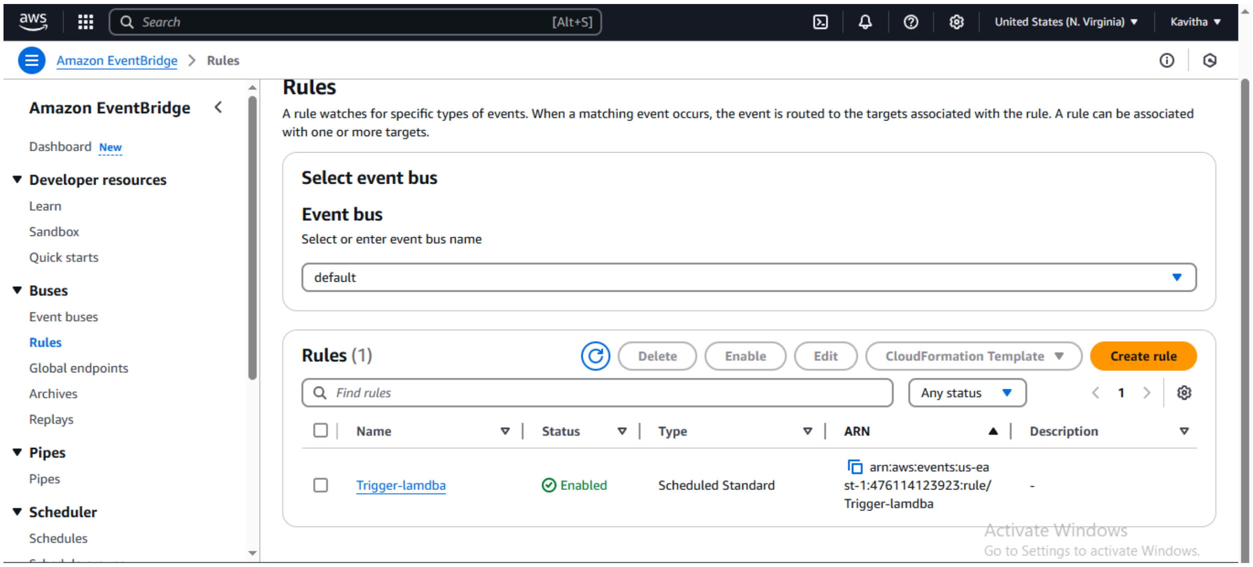
Targets

Details	Target Name	Type	Arn	Input	Role
▼	costoptimization	Lambda function	arn:aws:lambda:us-east-1:476114123923:function:costoptimization	Matched event	Amazon_EventBridge_Invoke_Lambda_710277584

Input to target: Matched event

Additional parameters: --

Rule created successfully



Rules

A rule watches for specific types of events. When a matching event occurs, the event is routed to the targets associated with the rule. A rule can be associated with one or more targets.

Select event bus

Event bus

Select or enter event bus name

default

Rules (1)

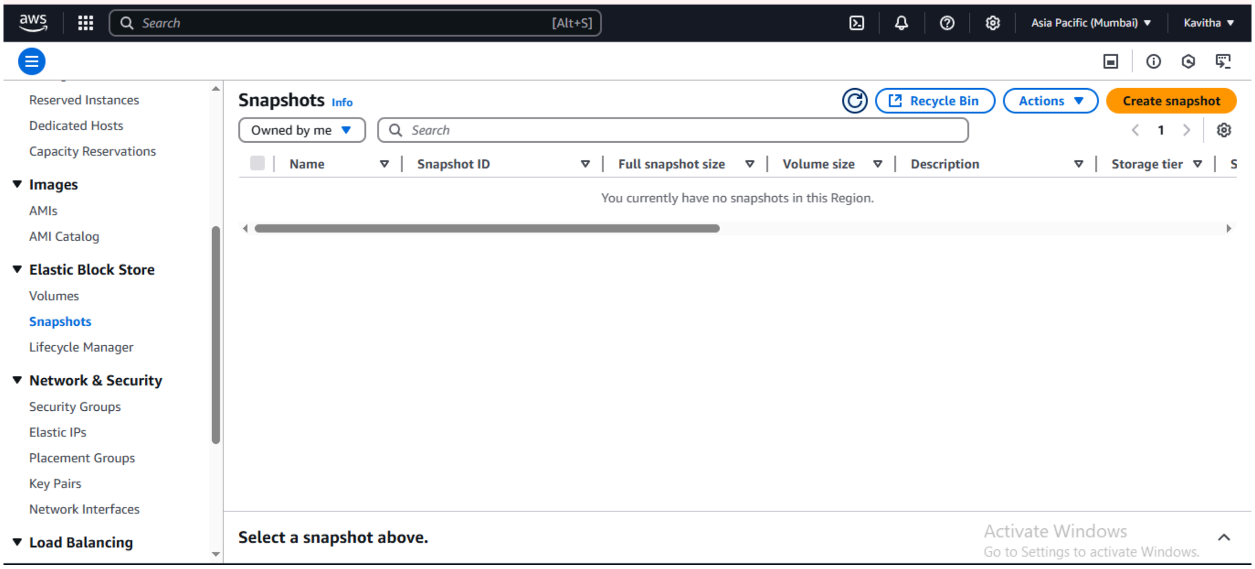
□	Name	Status	Type	ARN	Description
□	Trigger-lamdba	✓ Enabled	Scheduled Standard	arn:aws:events:us-east-1:476114123923:rule/Trigger-lamdba	-

Test the Lambda Function

1. In the Lambda console, click **Test**.
2. Use { } as the test event payload.
3. Click **Invoke** to check if old snapshots are deleted.

OUTPUT:

Snapshot got deleted by Invoking Lambda Function



Snapshots Info

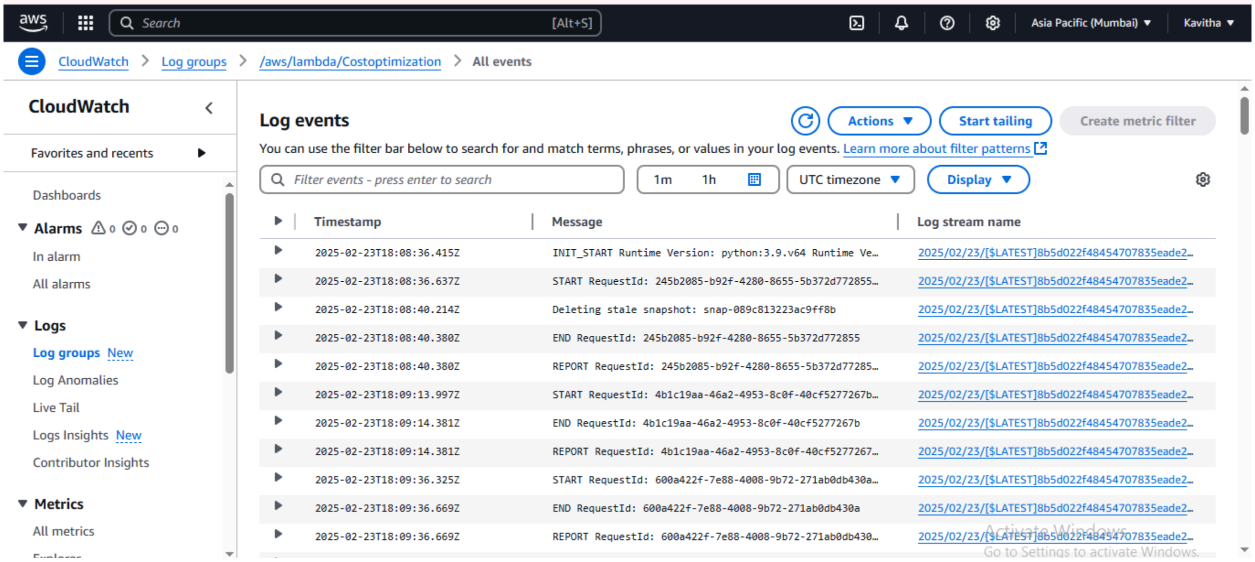
□	Name	Snapshot ID	Full snapshot size	Volume size	Description	Storage tier	S

You currently have no snapshots in this Region.

Select a snapshot above.

4️⃣ Check Logs in CloudWatch

1. Go to **CloudWatch Console**.
2. Click **Logs** → **Log Groups**.
3. Look for `/aws/lambda/costoptimization`.
4. Click on the latest log stream to see deletion activity.



You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

Timestamp	Message	Log stream name
2025-02-23T18:08:36.415Z	INIT_START Runtime Version: python:3.9.v64 Runtime Ve…	2025/02/23/[$LATEST]8b5d022f48454707835eade2…
2025-02-23T18:08:36.637Z	START RequestId: 245b2085-b92f-4280-8655-5b372d772855…	2025/02/23/[$LATEST]8b5d022f48454707835eade2…
2025-02-23T18:08:40.214Z	Deleting stale snapshot: snap-089c813223ac9ff8b	2025/02/23/[$LATEST]8b5d022f48454707835eade2…
2025-02-23T18:08:40.380Z	END RequestId: 245b2085-b92f-4280-8655-5b372d772855	2025/02/23/[$LATEST]8b5d022f48454707835eade2…
2025-02-23T18:08:40.380Z	REPORT RequestId: 245b2085-b92f-4280-8655-5b372d77285…	2025/02/23/[$LATEST]8b5d022f48454707835eade2…
2025-02-23T18:09:13.997Z	START RequestId: 4b1c19aa-46a2-4953-8c0f-40cf5277267b…	2025/02/23/[$LATEST]8b5d022f48454707835eade2…
2025-02-23T18:09:14.381Z	END RequestId: 4b1c19aa-46a2-4953-8c0f-40cf5277267b	2025/02/23/[$LATEST]8b5d022f48454707835eade2…
2025-02-23T18:09:14.381Z	REPORT RequestId: 4b1c19aa-46a2-4953-8c0f-40cf5277267…	2025/02/23/[$LATEST]8b5d022f48454707835eade2…
2025-02-23T18:09:36.325Z	START RequestId: 600a422f-7e88-4008-9b72-271ab0db430a…	2025/02/23/[$LATEST]8b5d022f48454707835eade2…
2025-02-23T18:09:36.669Z	END RequestId: 600a422f-7e88-4008-9b72-271ab0db430a	2025/02/23/[$LATEST]8b5d022f48454707835eade2…
2025-02-23T18:09:36.669Z	REPORT RequestId: 600a422f-7e88-4008-9b72-271ab0db430…	2025/02/23/[$LATEST]8b5d022f48454707835eade2…

The task was successfully completed by leveraging several AWS services to automate infrastructure management, monitor resources, and ensure data persistence and security. The key components, including EC2 instances, root volumes, snapshots, CloudFormation, CloudWatch and Lambda were efficiently configured to meet the requirements.