

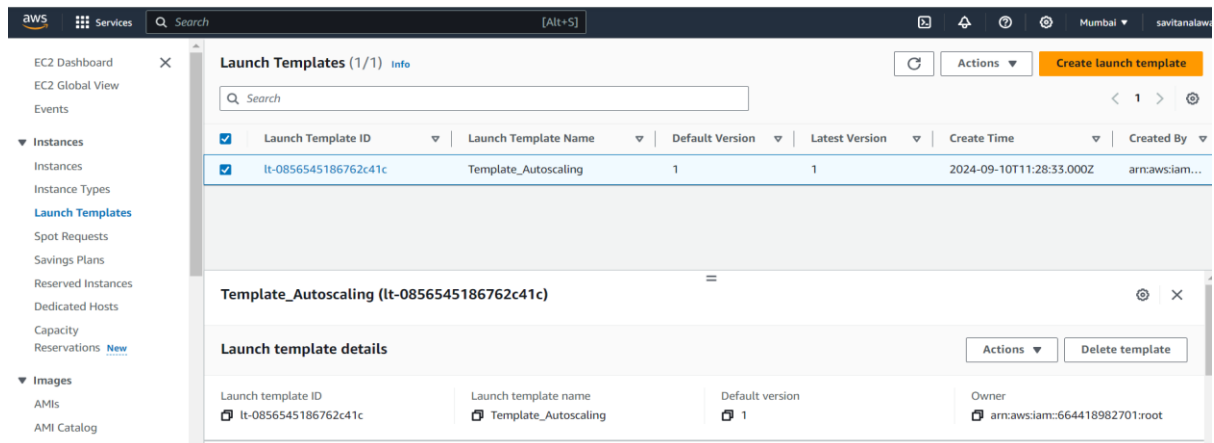
Autoscaling_Practical(10-09-24)

Savita Nalawade

- Autoscaling helps to maintain availability of your application

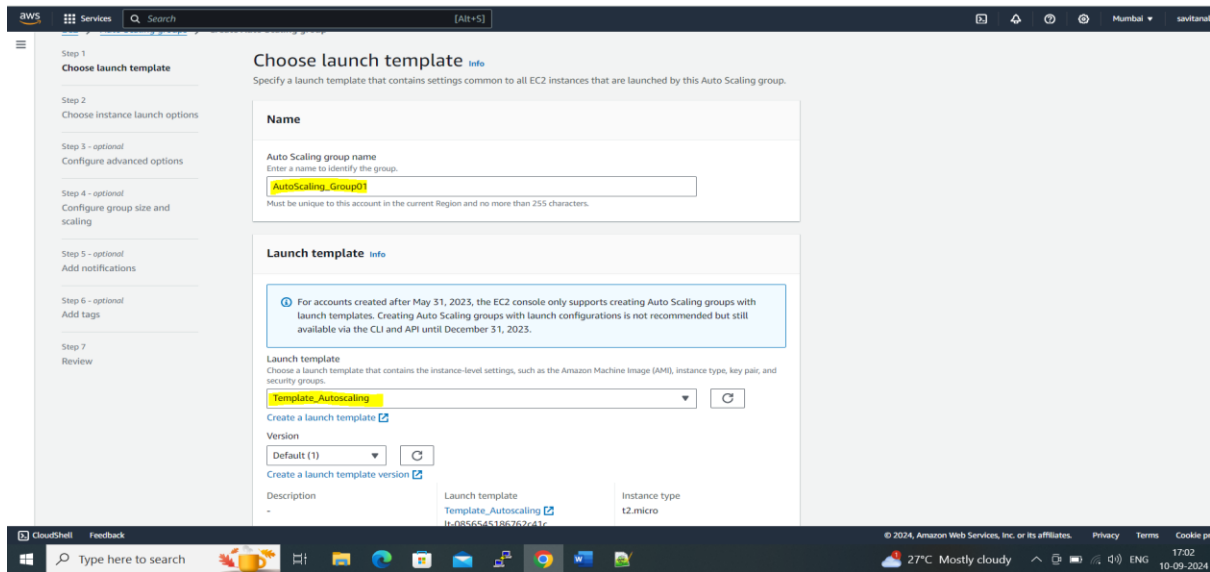
❖ Create template launch configuration

- Select AMI as Amazon Linux
- Select instance type “t2.micro”
- Select key-pair
- Select Security group
- Select role with s3FullAccess
- Advance setting -> add bootstrap script
- Create launch template

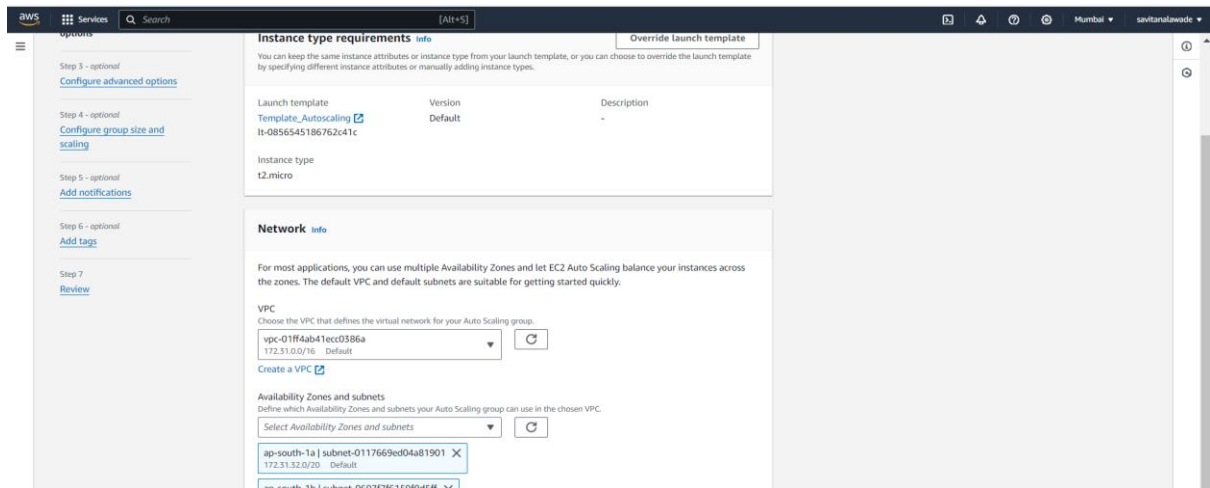


❖ Create Autoscaling Group

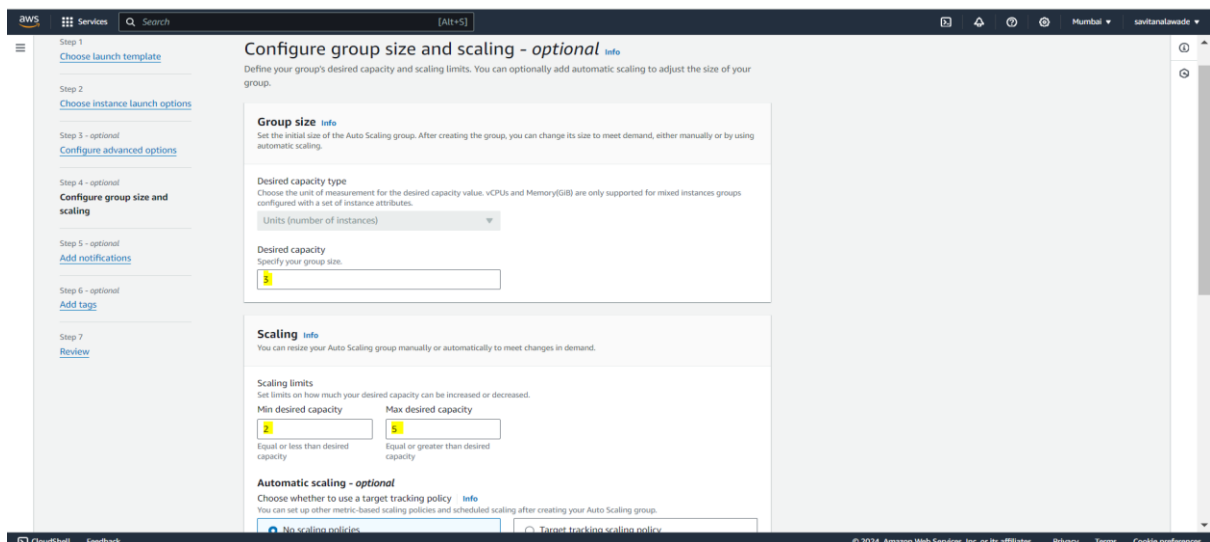
- Provide group name and select template which is created



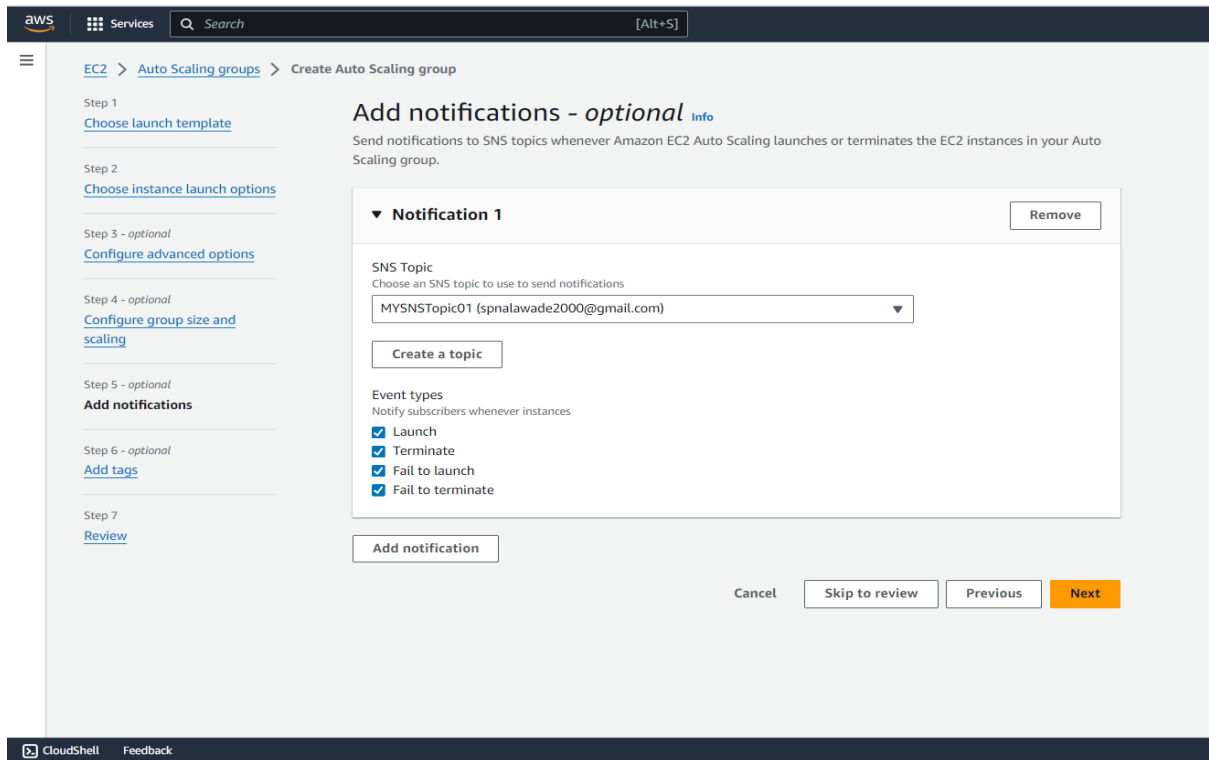
b. Select Availability Zone which you want to create instance



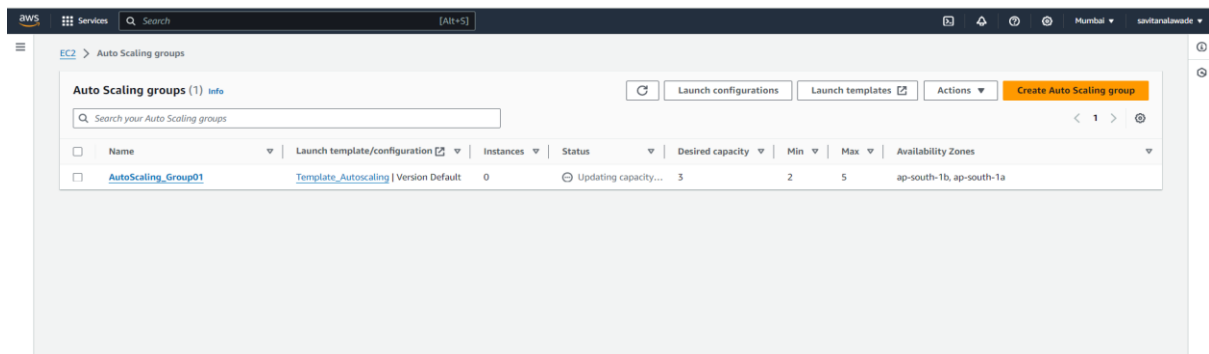
c. Configured group size and scaling



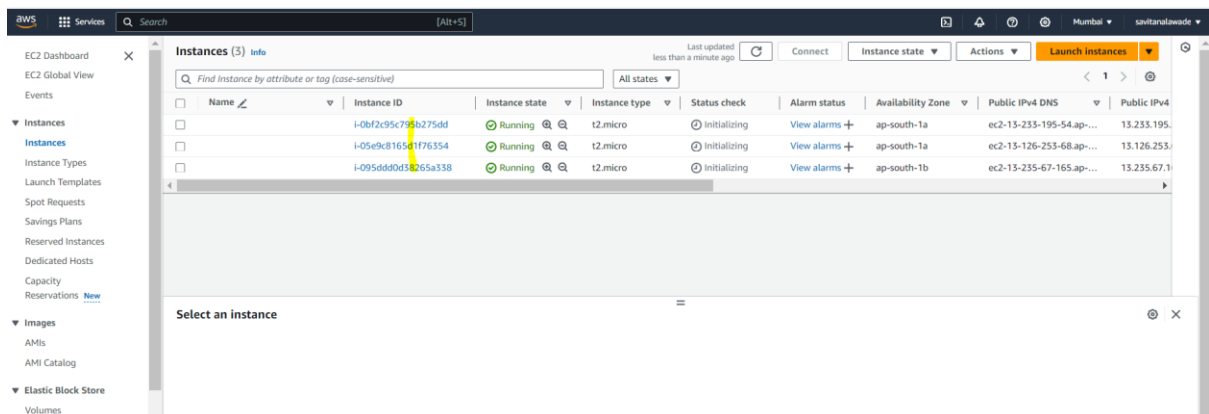
d. Add Notification



e. Auto scaling group have been created

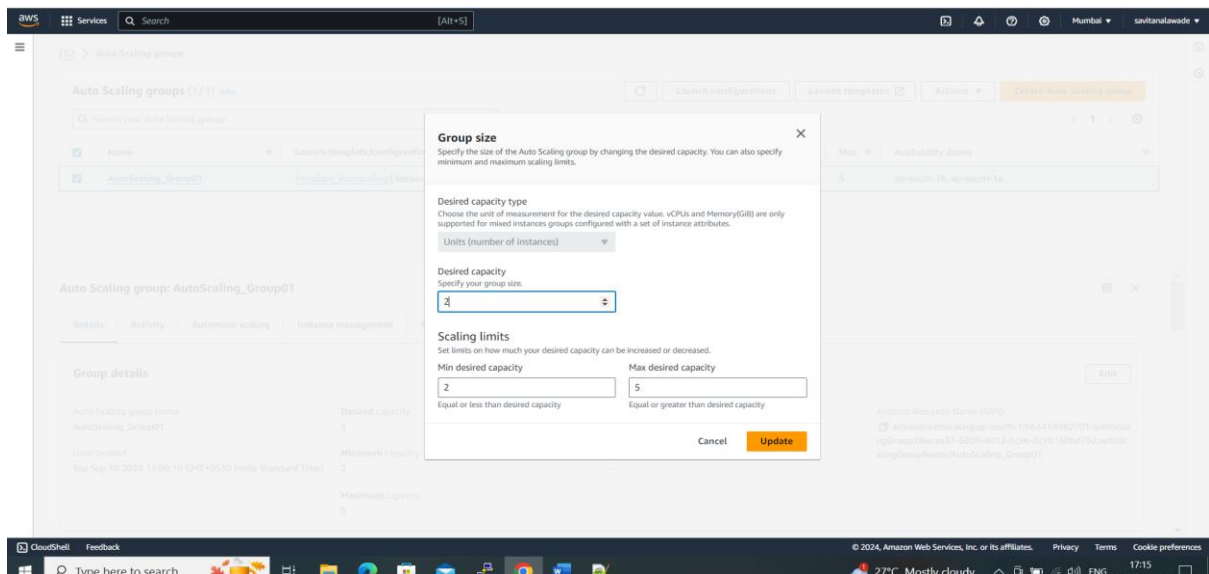


f. AutoScaling group have been created 3 Instance as we mentioned in desire "3" count

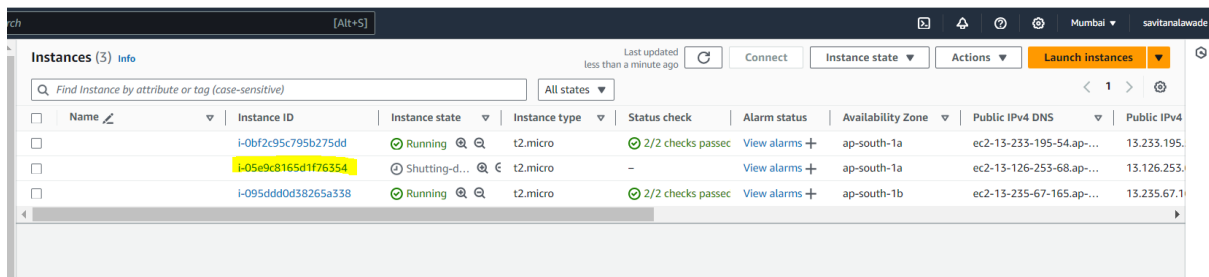


❖ Manual Scaling

If I change desired value manually as “2” so AutoScaling group should remove 1 instance.



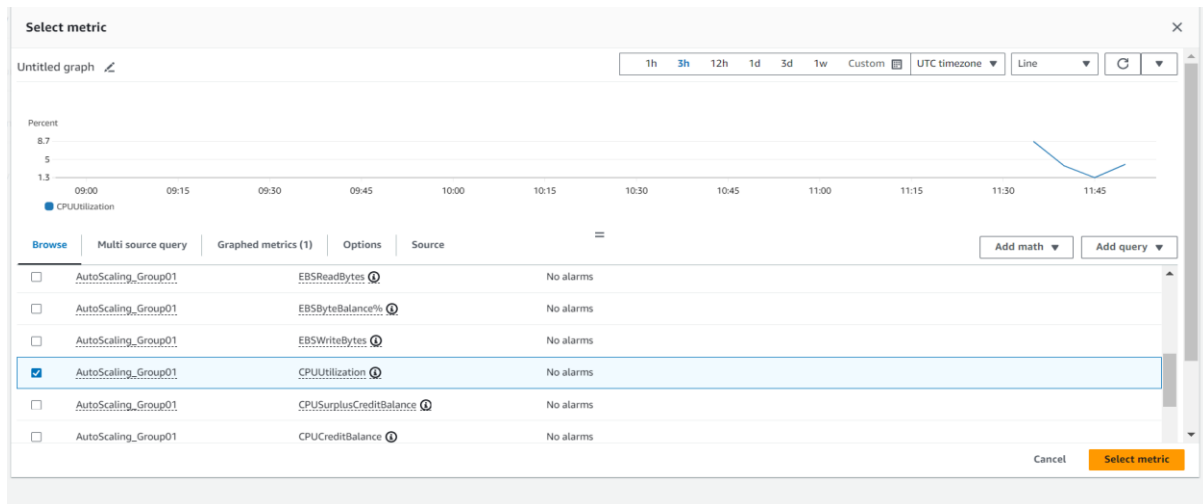
We can see one instance is going to terminate



❖ Automatic Scaling

To create Automatic scaling we required alarm to set Create dynamic scaling policies.

1. Create alarm for Autoscaling group
 - a. Select metric(CPU-Utilization)



b. Select timeperiod"1 min" and threshold value(utilization>=50)

Step 4
Preview and create

Percent

50

25

1e-3

09:30 10:30 11:30

■ CPUUtilization

Namespace
AWS/EC2

Metric name
CPUUtilization

AutoScalingGroupName
AutoScaling_Group01

Statistic
Average

Period
1 minute

Conditions

Threshold type

☒ Static
Use a value as a threshold

☐ Anomaly detection
Use a band as a threshold

Whenever CPUUtilization is...

☒ Greater
> threshold

☐ Greater/Equal
≥ threshold

☐ Lower/Equal
≤ threshold

☐ Lower
< threshold

than...

Define the threshold value.

50

c. Select in which state you want to trigger "In alarm" and add SNS-topic

CloudWatch > **Alarms** > Create alarm

Step 1
Specify metric and conditions

Step 2
Configure actions

Step 3
Add name and description

Step 4
Preview and create

Configure actions

Notification

Alarm state trigger
Define the alarm state that will trigger this action.

☒ In alarm
The metric or expression is outside of the defined threshold.

☐ OK
The metric or expression is within the defined threshold.

☐ Insufficient data
The alarm has just started or not enough data is available.

Remove

Send a notification to the following SNS topic
Define the SNS (Simple Notification Service) topic that will receive the notification.

☒ Select an existing SNS topic

☐ Create new topic

☐ Use topic ARN to notify other accounts

Send a notification to...

Q MYSNSTopic01

MYSNSTopic01

MYSNSTopic01

Email (endpoints)

spnawalade2000@gmail.com - View in SNS Console

Add notification

d. Add name for alarm

The screenshot shows the AWS CloudWatch console's 'Create alarm' wizard, specifically the 'Add name and description' step. The left sidebar shows the progress: Step 1 (Specify metric and conditions), Step 2 (Configure actions), Step 3 (Add name and description), and Step 4 (Preview and create). The main area is titled 'Add name and description' and contains a form with the following elements:

- Name and description** section:
- Alarm name:** A text input field containing 'AutoScaling_Increase_Instance_Alarm'.
- Alarm description - optional:** A text area with a link to 'View formatting guidelines'.
- Buttons:** 'Edit' and 'Preview' tabs, with 'Preview' currently selected.
- Character count:** 'Up to 1024 characters (0/1024)'.
- Information box:** A blue box with an information icon stating: 'Markdown formatting is only applied when viewing your alarm in the console. The description will remain in plain text in the alarm notifications.'
- Navigation buttons:** 'Cancel', 'Previous', and 'Next' buttons at the bottom right.

e. Alarm have been created.

The screenshot shows the AWS CloudWatch console's 'Alarms' page. A green banner at the top states 'Successfully created alarm AutoScaling_Increase_Instance_Alarm.' with a 'View alarm' button. Below the banner, the 'Alarms (1)' section displays a table of alarms. The table has columns for Name, State, Last state update (UTC), Conditions, and Actions. The single alarm listed is 'AutoScaling_Increase_Instance_Alarm' with a state of 'Insufficient data' and a last update of '2024-09-10 12:06:41'. The conditions are 'CPUUtilization > 50 for 1 datapoints within 1 minute' and the actions are 'Actions enabled'.

Name	State	Last state update (UTC)	Conditions	Actions
AutoScaling_Increase_Instance_Alarm	Insufficient data	2024-09-10 12:06:41	CPUUtilization > 50 for 1 datapoints within 1 minute	Actions enabled

❖ **Create Dynamic policy** with alarm in Take Action added count as “1”

It means if utilization is goes above threshold i.e 50% it should add one more instance.

EC2 > Auto Scaling groups > AutoScaling_Group01

Create dynamic scaling policy

Policy type
Simple scaling

Scaling policy name
Increase_Instance_Alarm

CloudWatch alarm
Choose an alarm that can scale capacity whenever:
AutoScaling_Increase_Instance_Alarm [Create a CloudWatch alarm](#)
breaches the alarm threshold: CPUUtilization > 50 for 1 consecutive periods of 60 seconds for the metric dimensions:
AutoScalingGroupName = AutoScaling_Group01

Take the action
Add 1 capacity units

And then wait
300 seconds before allowing another scaling activity

Cancel Create

Created Dynamic policy as “Increase_Instance_Alarm”

Dynamic scaling policy created or edited successfully.

EC2 > Auto Scaling groups

Auto Scaling groups (1/1) info

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones
AutoScaling_Group01	Template_Autoscaling Version Default	4	-	4	2	5	ap-south-1b, ap-south-1a

Auto Scaling group: AutoScaling_Group01

Details | Activity | Automatic scaling | Instance management | Monitoring | Instance refresh

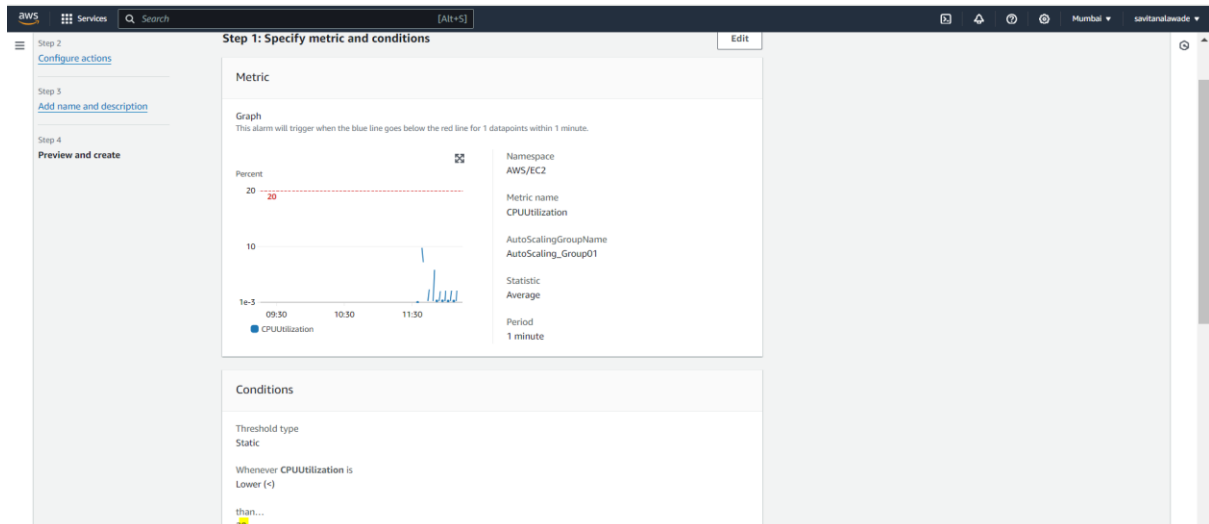
Scaling policies resize your Auto Scaling group to meet changes in demand. With reactive dynamic scaling policies, you can track specific CloudWatch metrics and take action when the CloudWatch alarm threshold is met. Use predictive scaling policies along with dynamic scaling policies in the following situations: when your application demand changes quickly, but with a recurring pattern, or when your EC2 instances require more time to initialize.

Dynamic scaling policies (1) info

Name	Policy type
Increase_Instance_Alarm	Simple scaling

❖ For one more dynamic policy we need to create one more alarm as “Decrease_Instance_Alarm”

1. Now, condition is {if CPU utilization is below 20% it should trigger alarm}



2. Alarm have been created

Successfully created alarm **Decrease_Instance_Alarm**. [View alarm](#)

CloudWatch > Alarms

Alarms (2/2) ☐ Hide Auto Scaling alarms [Clear selection](#) [Create composite alarm](#) [Actions](#) [Create alarm](#)

Alarm state: Any Alarm type: Any Actions status: Any < 1 >

<input checked="" type="checkbox"/>	Name	State	Last state update (UTC)	Conditions	Actions
<input checked="" type="checkbox"/>	Decrease_Instance_Alarm	Insufficient data	2024-09-10 12:18:58	CPUUtilization < 20 for 1 datapoints within 1 minute	Actions enabled
<input checked="" type="checkbox"/>	AutoScaling_Increase_Instance_Alarm	OK	2024-09-10 12:07:53	CPUUtilization > 50 for 1 datapoints within 1 minute	Actions enabled

❖ Create policy as “**Decrease_Instance_Policy**” in Take Action select “Remove” and added count as “1”

It means if utilization its below threshold i.e 20% it should remove one instance.

EC2 > Auto Scaling groups > AutoScaling_Group01

Create dynamic scaling policy

Policy type: Simple scaling

Scaling policy name: Decrease_Instance_Policy

CloudWatch alarm: Choose an alarm that can scale capacity whenever: Decrease_Instance_Alarm [Create a CloudWatch alarm](#)

breaches the alarm threshold: CPUUtilization < 20 for 1 consecutive periods of 60 seconds for the metric dimensions:

AutoScalingGroupName = AutoScaling_Group01

Take the action: Remove 1 capacity units

And then wait: 300 seconds before allowing another scaling activity

[Cancel](#) [Create](#)

Auto Scaling groups (1/1) Info

Search your Auto Scaling groups

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones
AutoScaling_Group01	Template_AutoScaling Version Default	4	-	4	2	5	ap-south-1b, ap-south-1a

Auto Scaling group: AutoScaling_Group01

Decrease_Instance_Policy

Policy type
Simple scaling

Enabled or disabled
Enabled

Execute policy when
Decrease_Instance_Alarm
breaches the alarm threshold: CPUUtilization < 20 for 1 consecutive periods of 60 seconds for the metric dimensions:
AutoScalingGroupName = AutoScaling_Group01

Take the action
Remove 1 capacity units

And then wait
300 seconds before allowing another scaling activity

Increase_Instance_Alarm

Policy type
Simple scaling

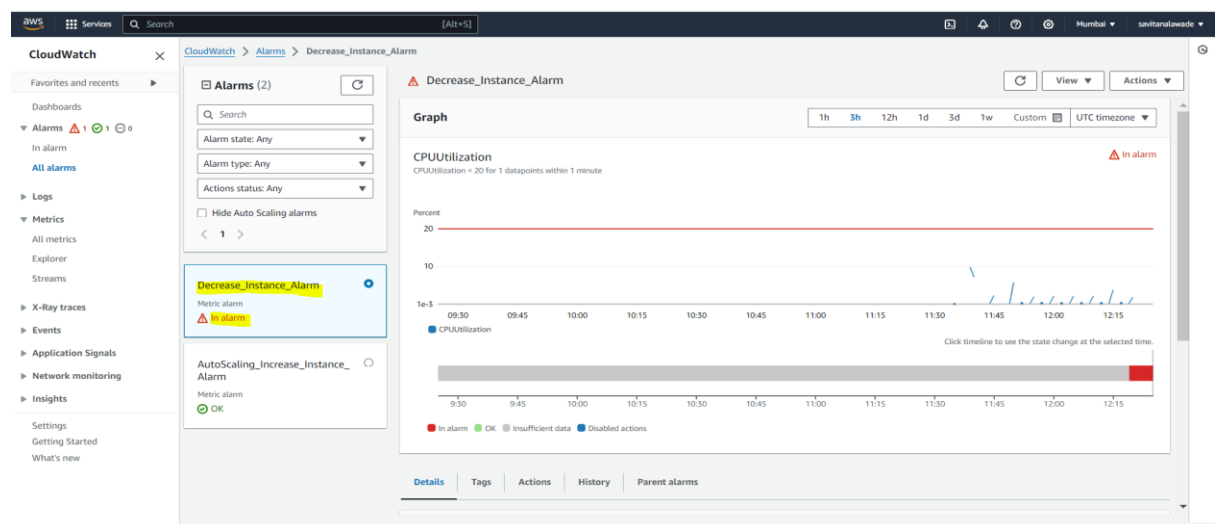
Enabled or disabled
Enabled

Execute policy when
AutoScaling_Increase_Instance_Alarm
breaches the alarm threshold: CPUUtilization > 50 for 1 consecutive periods of 60 seconds for the metric dimensions:
AutoScalingGroupName = AutoScaling_Group01

Take the action
Add 1 capacity units

And then wait
300 seconds before allowing another scaling activity

1. We could see one alarm has been triggered “Decrease_Instance_Alarm”
As checked average cpu utilization of Instance are below 20% only .



5:49 PM (19 minutes ago)

☆ (

View this alarm in the AWS Management Console:

Alarm Details:

- Description:

- State Change

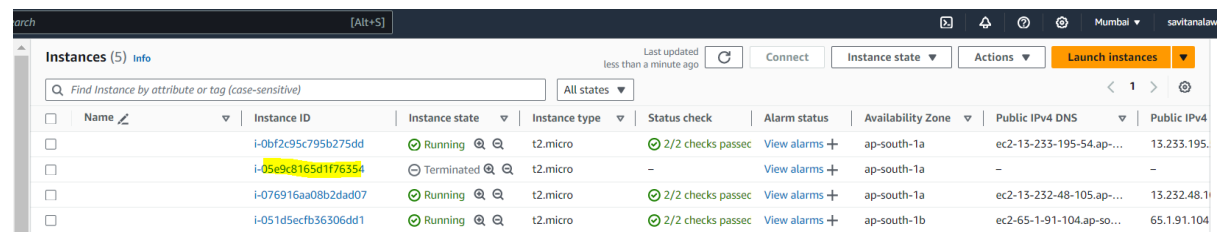
- Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [4.538091111179893 (10/09/24 12:15:00)] was less than the threshold (1 datapoint for OK -> ALARM transition).

- Timestamp: Tuesday 10 September, 2024 12:19:04 UTC

- AWS Account: 664418982701

```
- Alarm Arn:          arn:aws:cloudwatch:ap-south-1:664418982701:alarm:Decrease Instance Alarm
```

2. Now, It will remove one Instance out of 4 running Instance



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4
	i-0bf2c95c795b275dd	Running	t2.micro	2/2 checks passed	View alarms	ap-south-1a	ec2-13-233-195-54.ap-...	13.233.195.
	i-05e9c8165d1f76354	Terminated	t2.micro	-	View alarms	ap-south-1a	-	-
	i-076916aa08b2dad07	Running	t2.micro	2/2 checks passed	View alarms	ap-south-1a	ec2-13-232-48-105.ap-...	13.232.48.1
	i-051d5ecfb36306dd1	Running	t2.micro	2/2 checks passed	View alarms	ap-south-1b	ec2-65-1-91-104.ap-so...	65.1.91.104

❖ To trigger “Increase_Instance_alarm” we need to make high cpu utilization of instances

We'll increase cpu-utilization by installing stress and made cpu changes on both instance:



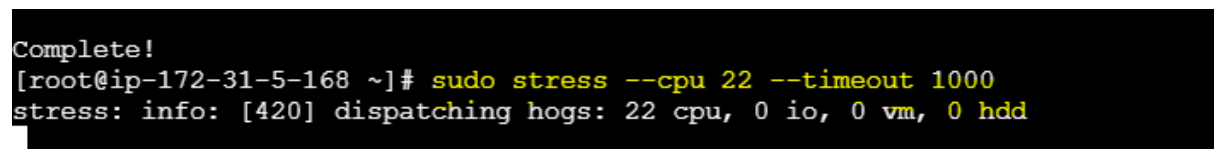
```
Amazon Linux 2
AL2 End of Life is 2025-06-30.

A newer version of Amazon Linux is available!
Amazon Linux 2023, GA and supported until 2028-03-15.
https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-172-31-41-143 ~]$ sudo su -
[root@ip-172-31-41-143 ~]# sudo amazon-linux-extras install epel -y && yum install stress -y
Installing epel-release
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Cleaning repos: amzn2-core amzn2extra-docker amzn2extra-epel amzn2extra-kernel-5.10
17 metadata files removed
6 sqlite files removed
0 metadata files removed
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
amzn2extra-docker
amzn2extra-epel
amzn2extra-kernel-5.10
(1/9): amzn2-core/2/x86_64/group_gz
(2/9): amzn2-core/2/x86_64/updateinfo
(3/9): amzn2extra-epel/2/x86_64/primary_db
(4/9): amzn2extra-kernel-5.10/2/x86_64/updateinfo
(5/9): amzn2extra-docker/2/x86_64/updateinfo
(6/9): amzn2extra-docker/2/x86_64/primary_db
(7/9): amzn2extra-epel/2/x86_64/updateinfo
(8/9): amzn2extra-kernel-5.10/2/x86_64/primary_db
(9/9): amzn2-core/2/x86_64/primary_db
Resolving Dependencies
--> Running transaction check
--> Package epel-release, noarch 0:7-11 will be installed
--> Finished Dependency Resolution

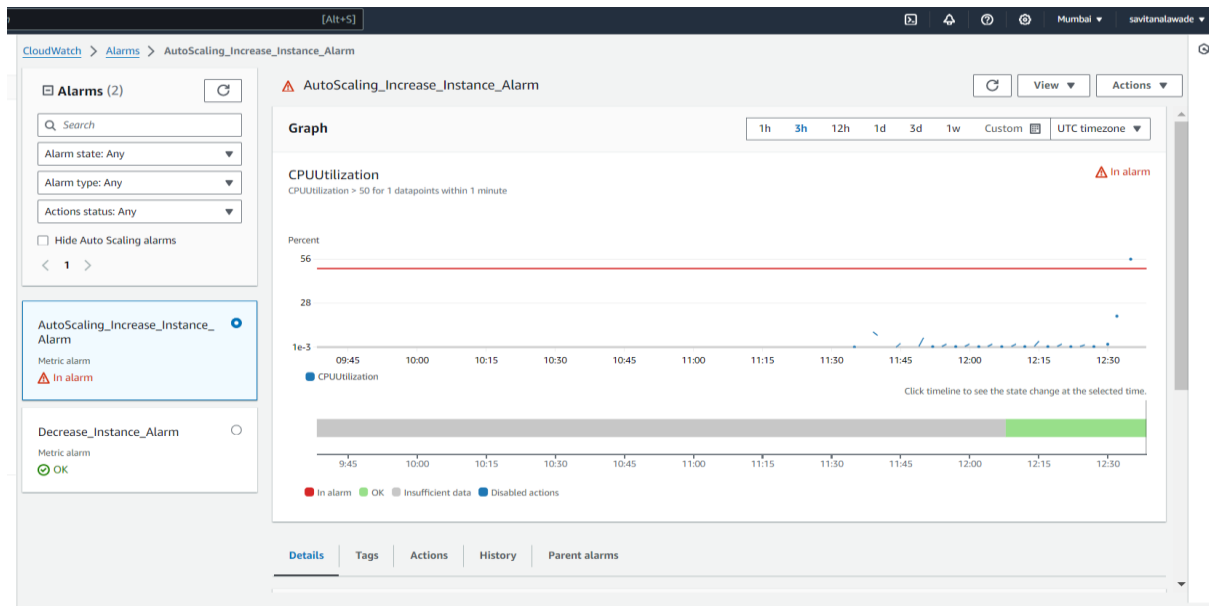
amzn2-core | 3.6 kB 00:00:00
amzn2extra-docker | 2.9 kB 00:00:00
amzn2extra-epel | 3.0 kB 00:00:00
amzn2extra-kernel-5.10 | 3.0 kB 00:00:00
(1/9): amzn2-core/2/x86_64/group_gz | 2.7 kB 00:00:00
(2/9): amzn2-core/2/x86_64/updateinfo | 973 kB 00:00:00
(3/9): amzn2extra-epel/2/x86_64/primary_db | 1.8 kB 00:00:00
(4/9): amzn2extra-kernel-5.10/2/x86_64/updateinfo | 88 kB 00:00:00
(5/9): amzn2extra-docker/2/x86_64/updateinfo | 19 kB 00:00:00
(6/9): amzn2extra-docker/2/x86_64/primary_db | 107 kB 00:00:00
(7/9): amzn2extra-epel/2/x86_64/updateinfo | 76 B 00:00:00
(8/9): amzn2extra-kernel-5.10/2/x86_64/primary_db | 30 MB 00:00:00
(9/9): amzn2-core/2/x86_64/primary_db | 70 MB 00:00:01
```

Increase cpu by stress command “sudo -cpu 22 -timeout 1000”



```
Complete!
[root@ip-172-31-5-168 ~]# sudo stress --cpu 22 --timeout 1000
stress: info: [420] dispatching hogs: 22 cpu, 0 io, 0 vm, 0 hdd
```

Now check cpu-utilization goes high and alarm will triggered



ALARM: "AutoScaling_Increase_Instance_Alarm" in Asia Pacific (Mumbai) inbox x

AWS Notifications <no-reply@sns.amazonaws.com> 6:09PM (0 minutes ago) ☆ ☺ ↶ ⋮

to me ▾

You are receiving this email because your Amazon CloudWatch Alarm "AutoScaling_Increase_Instance_Alarm" in the Asia Pacific (Mumbai) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [55.98300299439454 (10/09/24 12:35:00)] was greater than the threshold (50.0) (minimum 1 datapoint for OK -> ALARM transition)." at "Tuesday 10 September, 2024 12:39:43 UTC".

View this alarm in the AWS Management Console:
https://ap-south-1.console.aws.amazon.com/cloudwatch/deeplink.js?region=ap-south-1#alarmsV2:alarm/AutoScaling_Increase_Instance_Alarm

Alarm Details:

- Name: AutoScaling_Increase_Instance_Alarm
- Description:
- State Change: OK -> ALARM
- Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [55.98300299439454 (10/09/24 12:35:00)] was greater than the threshold (50.0) (minimum 1 datapoint for OK -> ALARM transition).
- Timestamp: Tuesday 10 September, 2024 12:39:43 UTC
- AWS Account: 664418982701
- Alarm Arn: arn:aws:cloudwatch:ap-south-1:664418982701:alarm:AutoScaling_Increase_Instance_Alarm

Threshold:

- The alarm is in the ALARM state when the metric is GreaterThanThreshold 50.0 for at least 1 of the last 1 period(s) of 60 seconds.

Monitored Metric:

It will generate one instance

The screenshot shows the AWS EC2 console. On the left, there's a sidebar with navigation options like 'Instances', 'Images', 'Elastic Block Store', and 'Network & Security'. The main panel shows a list of instances. The instance 'i-051d5ecfb36306dd1' is selected, and its details are shown in the right pane. The details include the instance ID, name, state (Running), type (t2.micro), and various addresses (Public IPv4, Private IPv4, Public IPv4 DNS, Elastic IP addresses).

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4
	i-0bf2c95c795b275dd	Terminated	t2.micro	-	View alarms +	ap-south-1a	-	-
	i-05e9c8165d1f76354	Terminated	t2.micro	-	View alarms +	ap-south-1a	-	-
	i-076916aa08b2dad07	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a	ec2-13-232-48-105.ap-...	15.232.48.1
	i-0f4637ed2baf85fa	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a	ec2-52-66-213-37.ap-...	52.66.213.3
	i-051d5ecfb36306dd1	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	ec2-65-1-91-104.ap-...	65.1.91.104
	i-095ddd0d38265a338	Terminated	t2.micro	-	View alarms +	ap-south-1b	-	-

i-051d5ecfb36306dd1

Instance summary

Instance ID: i-051d5ecfb36306dd1

IPv4 address: -

IPv6 address: -

Hostname type: -

IP name: ip-172-31-5-168.ap-south-1.compute.internal

Answer private resource DNS name: -

Public IPv4 address: 65.1.91.104 | open address

Instance state: Running

Private IP DNS name (IPv4 only): ip-172-31-5-168.ap-south-1.compute.internal

Instance type: t2.micro

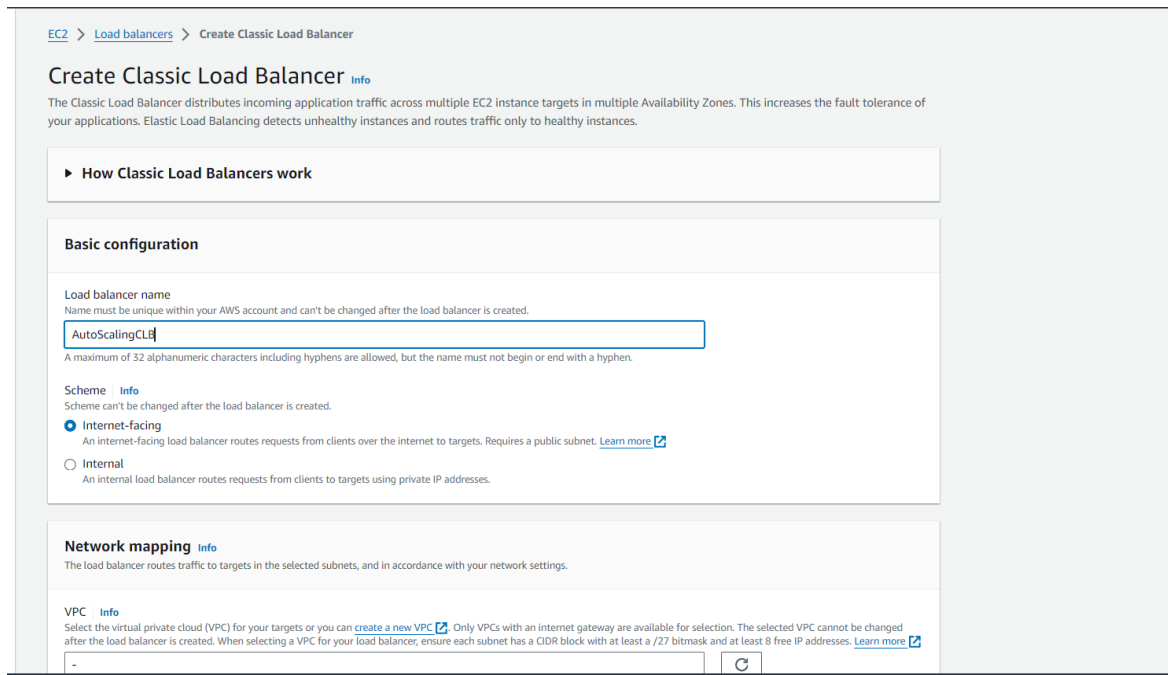
Private IPv4 addresses: 172.31.5.168

Public IPv4 DNS: ec2-65-1-91-104.ap-south-1.compute.amazonaws.com | open address

Elastic IP addresses: -

❖ Create Classic Load Balancer

1. Create load balancer as “AutoScalingCLB”



EC2 > Load balancers > Create Classic Load Balancer

Create Classic Load Balancer [Info](#)

The Classic Load Balancer distributes incoming application traffic across multiple EC2 instance targets in multiple Availability Zones. This increases the fault tolerance of your applications. Elastic Load Balancing detects unhealthy instances and routes traffic only to healthy instances.

► **How Classic Load Balancers work**

Basic configuration

Load balancer name
Name must be unique within your AWS account and can't be changed after the load balancer is created.

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Scheme [Info](#)
Scheme can't be changed after the load balancer is created.

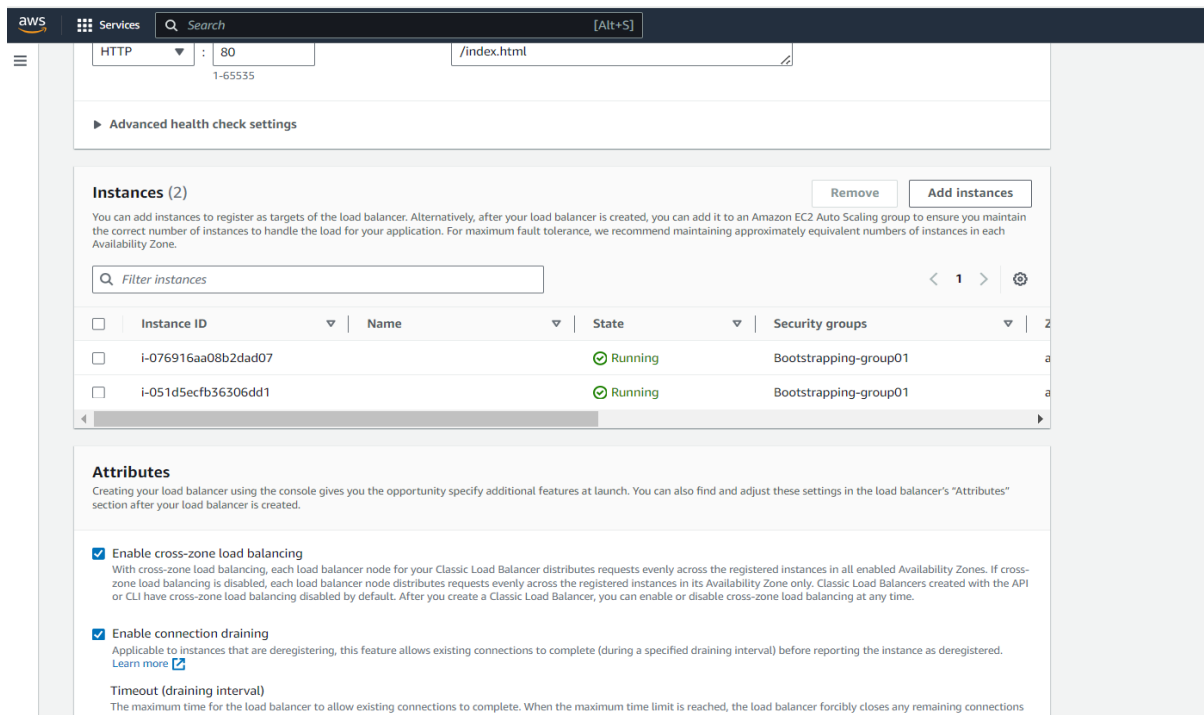
☒ **Internet-facing**
An internet-facing load balancer routes requests from clients over the internet to targets. Requires a public subnet. [Learn more](#)

☐ **Internal**
An internal load balancer routes requests from clients to targets using private IP addresses.

Network mapping [Info](#)
The load balancer routes traffic to targets in the selected subnets, and in accordance with your network settings.

VPC [Info](#)
Select the virtual private cloud (VPC) for your targets or you can [create a new VPC](#). Only VPCs with an internet gateway are available for selection. The selected VPC cannot be changed after the load balancer is created. When selecting a VPC for your load balancer, ensure each subnet has a CIDR block with at least a /27 bitmask and at least 8 free IP addresses. [Learn more](#)

2. Select security group and add running Instances



aws Services Search [Alt+S]

HTTP : 80 /index.html 1-65535

► **Advanced health check settings**

Instances (2)

You can add instances to register as targets of the load balancer. Alternatively, after your load balancer is created, you can add it to an Amazon EC2 Auto Scaling group to ensure you maintain the correct number of instances to handle the load for your application. For maximum fault tolerance, we recommend maintaining approximately equivalent numbers of instances in each Availability Zone.

< 1 > ⚙

<input type="checkbox"/>	Instance ID	Name	State	Security groups
<input type="checkbox"/>	i-076916aa08b2dad07		Running	Bootstrapping-group01
<input type="checkbox"/>	i-051d5ecfb36306dd1		Running	Bootstrapping-group01

◀ ▶

Attributes

Creating your load balancer using the console gives you the opportunity specify additional features at launch. You can also find and adjust these settings in the load balancer's "Attributes" section after your load balancer is created.

☒ **Enable cross-zone load balancing**
With cross-zone load balancing, each load balancer node for your Classic Load Balancer distributes requests evenly across the registered instances in all enabled Availability Zones. If cross-zone load balancing is disabled, each load balancer node distributes requests evenly across the registered instances in its Availability Zone only. Classic Load Balancers created with the API or CLI have cross-zone load balancing disabled by default. After you create a Classic Load Balancer, you can enable or disable cross-zone load balancing at any time.

☒ **Enable connection draining**
Applicable to instances that are deregistering, this feature allows existing connections to complete (during a specified draining interval) before reporting the instance as deregistered. [Learn more](#)

Timeout (draining interval)
The maximum time for the load balancer to allow existing connections to complete. When the maximum time limit is reached, the load balancer forcibly closes any remaining connections and marks the instance as deregistered.

3. Load Balancer have been created and instances are in service

Successfully created load balancer: **AutoScalingCLB**
It might take a few minutes for your load balancer to be fully set up and ready to route traffic. Targets will also take a few minutes to complete the registration process and pass initial health checks.

EC2 > Load balancers > AutoScalingCLB

Load balancers (1/1)
Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Filter load balancers: 1 match

AutoScalingCLB Clear filters

<input checked="" type="checkbox"/>	Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
<input checked="" type="checkbox"/>	AutoScalingCLB	AutoScalingCLB-1204055...	-	vpc-01ff4ab41ecc0386a	2 Availability Zones	classic	September 10, 2024, 18:33 (UTC+05:...

Load balancer: AutoScalingCLB

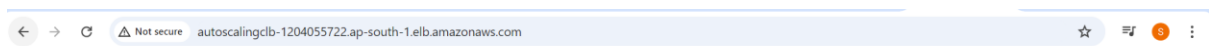
Details | Listeners | Network mapping | Security | Health checks | **Target instances** | Monitoring | Attributes | Tags

Target instances (2)
Instances currently registered to your load balancer are displayed. To deregister instances, select them, then choose Deregister. To register and deregister instances simultaneously, choose Manage instances.

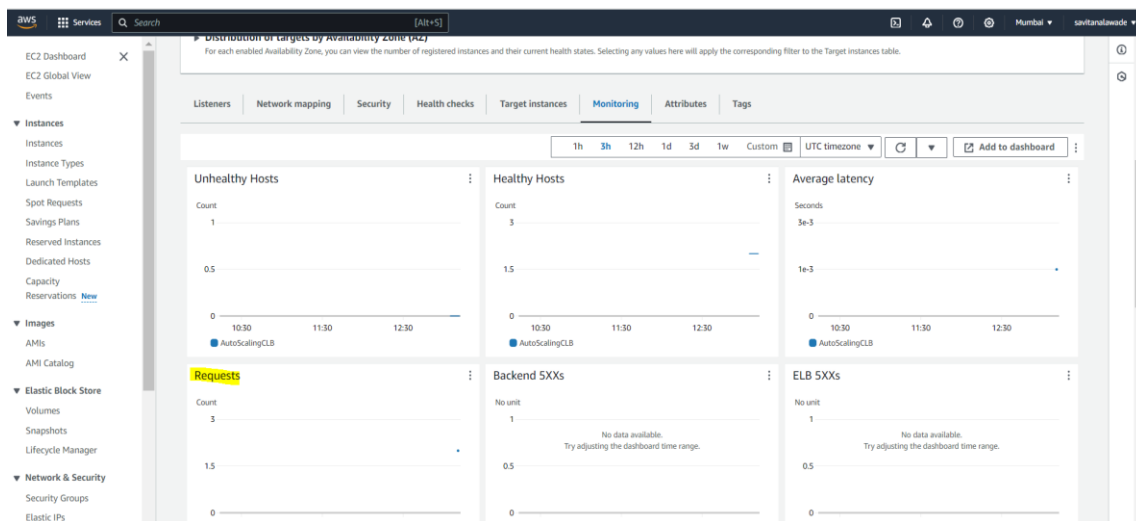
Filter target instances

<input type="checkbox"/>	Instance ID	Name	Health status	Health status description	Security groups	Zone
<input type="checkbox"/>	i-076916aa08b2dad07		In-service	Not applicable	Bootstrapping-group01	ap-south
<input type="checkbox"/>	i-051d5ecfb36306dd1		In-service	Not applicable	Bootstrapping-group01	ap-south

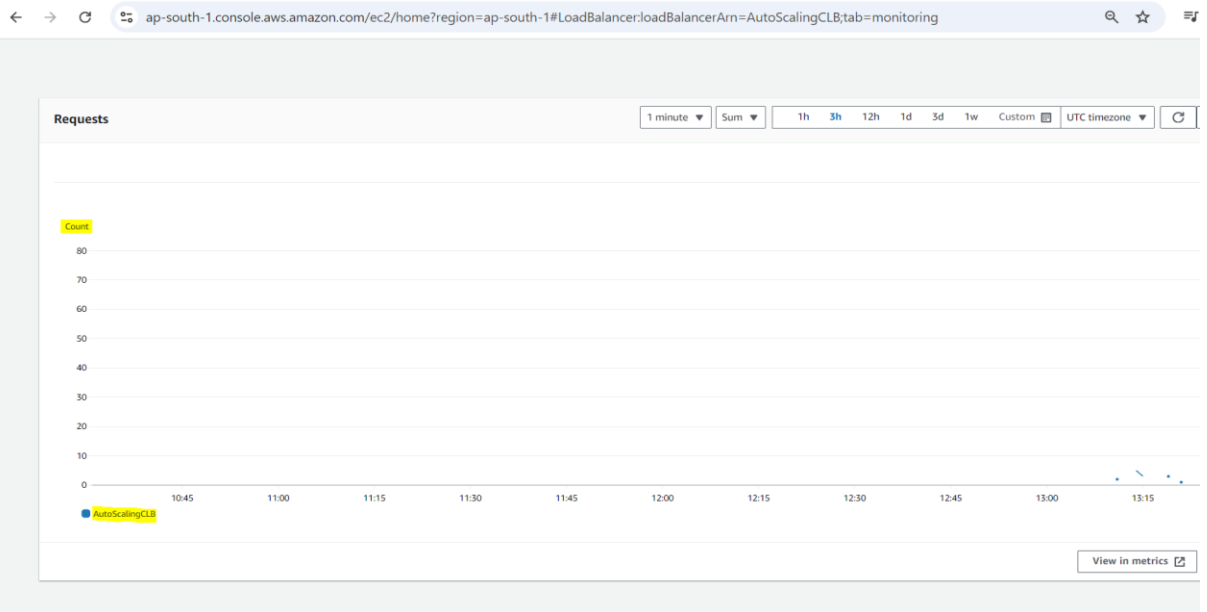
3. Using DNS of load balancer we can view application and if hit multiple times we can see in monitoring of LB



Create and configure the service front-end-service so its accessible through ClusterIP and routes to the existing pod named front-end



After hitting multiple times on browser we can see request are getting increased



Thank you...