AWS AUTO SCALING

AWS Auto Scaling is a service that automatically adjusts the number of instances in your AWS setup based on the conditions you define.

AWS Auto Scaling helps you maintain the availability and performance of your applications by automatically adjusting the number of EC2 instances or other resources based on demand. When demand increases, it adds more instances; when demand decreases, it removes instances, ensuring you're not paying for resources you don't need.

Types of Auto Scaling:

1. **Manual Scaling:** *In the AWS Auto Scaling console, you can create scaling policies that specify the conditions under which scaling actions should be taken. These policies can be triggered based on metrics like CPU utilization, network traffic, or custom metrics that you define. By configuring these policies manually, you effectively implement manual scaling.*

2. **Dynamic Scaling:** *When creating scaling policies, you can choose to use dynamic scaling by configuring scaling actions based on CloudWatch alarms or other metric data. These policies will automatically adjust the number of instances based on real-time demand, effectively implementing dynamic scaling.*

3. **Scheduled Scaling:** *In the AWS Auto Scaling console, you can create scheduled actions that define specific times for scaling actions to occur. These actions allow you to anticipate changes in demand and adjust your resources accordingly, implementing scheduled scaling.*

4. **Predictive Scaling:** *While not explicitly labeled in the console, you can leverage AWS Auto Scaling features like predictive scaling by enabling predictive scaling within your scaling policies. AWS Auto Scaling uses machine learning algorithms to forecast future demand based on historical data and adjusts resources proactively, optimizing performance and cost.*

   ***Note**: while the AWS Management Console may not have distinct options labeled as "Auto Scaling Types," you can implement different types of auto scaling by configuring your scaling policies and settings according to your specific requirements and preferences.*

***Vertical and horizontal scaling** are indeed approaches to scaling infrastructure but are not typically referred to as types of auto scaling within AWS.*

1. ***Vertical Auto Scaling:***
   - *Also known as scaling up or scaling vertically.*
   - *Involves increasing the capacity of individual instances by adding more resources to them, such as CPU, memory, or storage.*
   - *Example: Instead of adding more instances to your fleet, you might vertically scale by upgrading the instance type to a larger one with more CPU and memory.*
2. ***Horizontal Auto Scaling:***
   - *Also known as scaling out or scaling horizontally.*
   - *Involves adding more instances to your application to distribute the load across multiple servers.*
   - *Example: When demand increases, horizontal scaling automatically spins up additional instances of your application to handle the increased load. Conversely, when demand decreases, it reduces the number of instances to save costs.*

Both vertical and horizontal scaling have their advantages and use cases, and in many scenarios, a combination of both is employed to ensure optimal performance and resource utilization in response to changing demands.

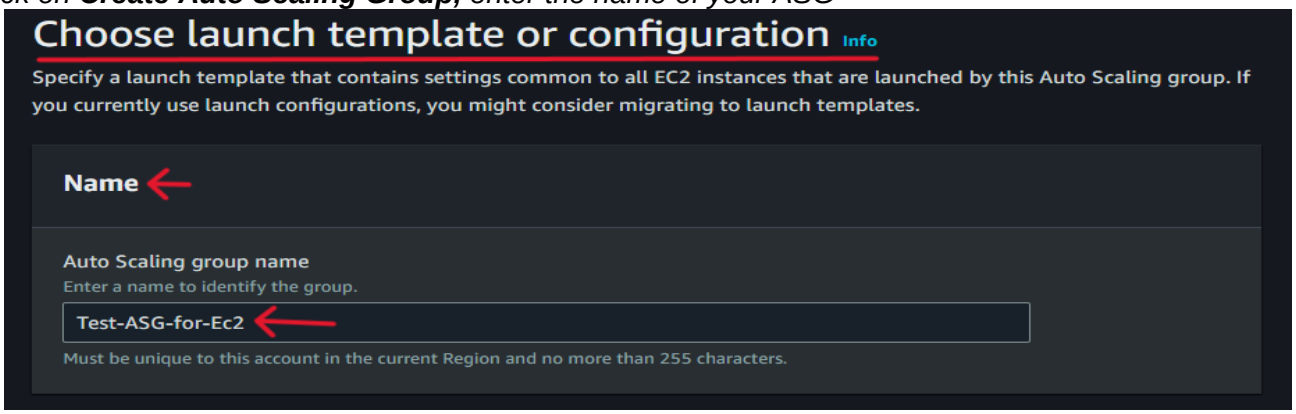
Now we will set up auto scaling for an application, we can set auto scaling for our running instance or even create a new instance using a template.

We have a pre-configured Instance template, and we need to create a Load balancer, and Target group then we create an Auto Scaling group by configuring our template and load balancer to it.

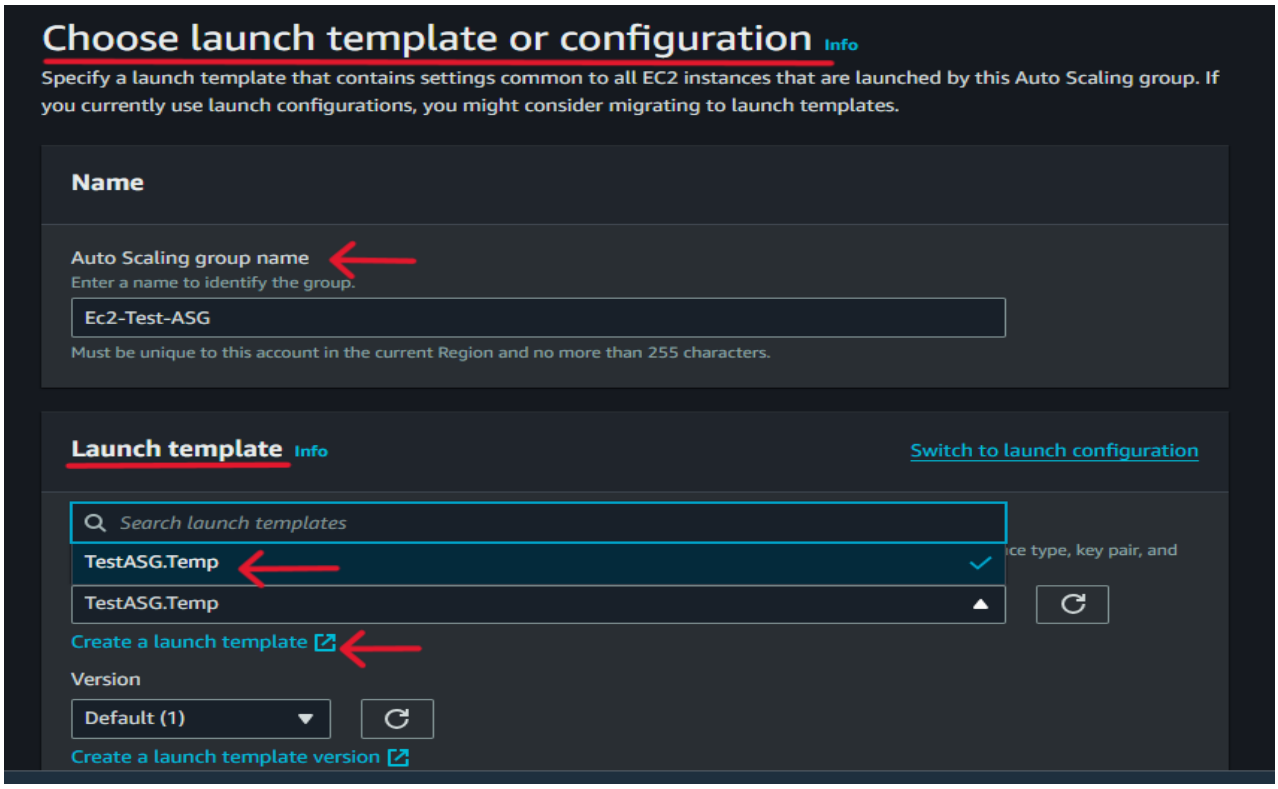
Then we will put artificial stress on our servers to test whether our Auto Scaling is working properly.

Steps to Setting up Auto Scaling

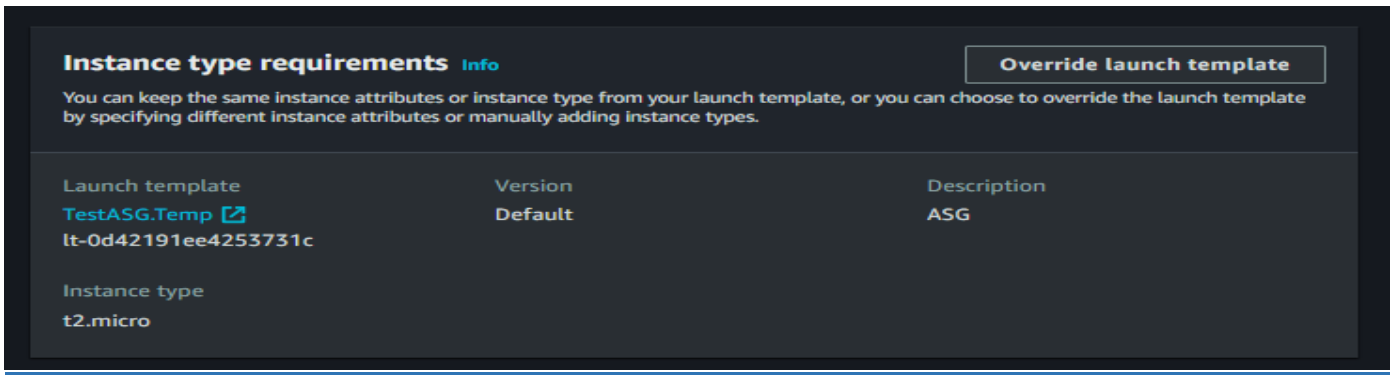
*Step 1)Go to **Ec2 Dashboard**'s Auto Scaling Menu, select **Auto Scaling Groups,***
*Step 2)Click on **Create Auto Scaling Group,** enter the name of your ASG*



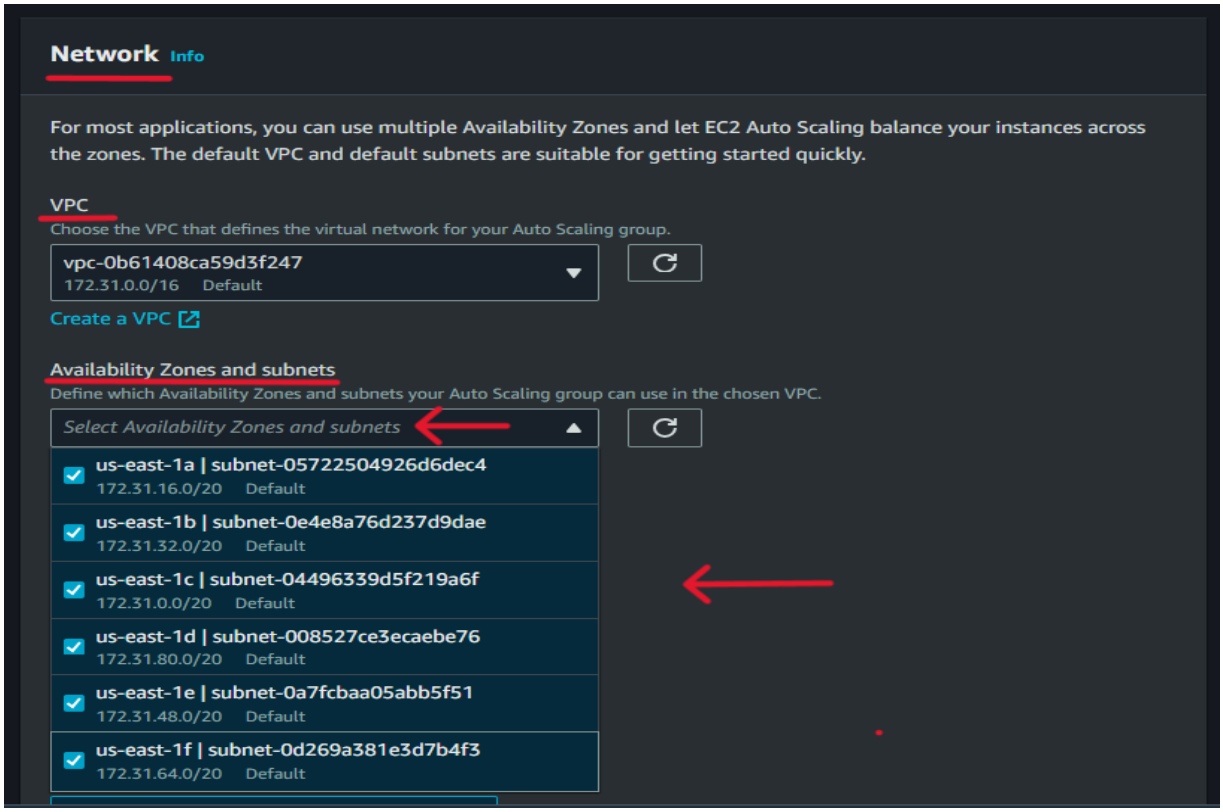
Step 3) Select Template, and click on the next button. (if the pre-configured template is not created click on Create a Launch Template, and a new tab will open Launch it then proceed next step)



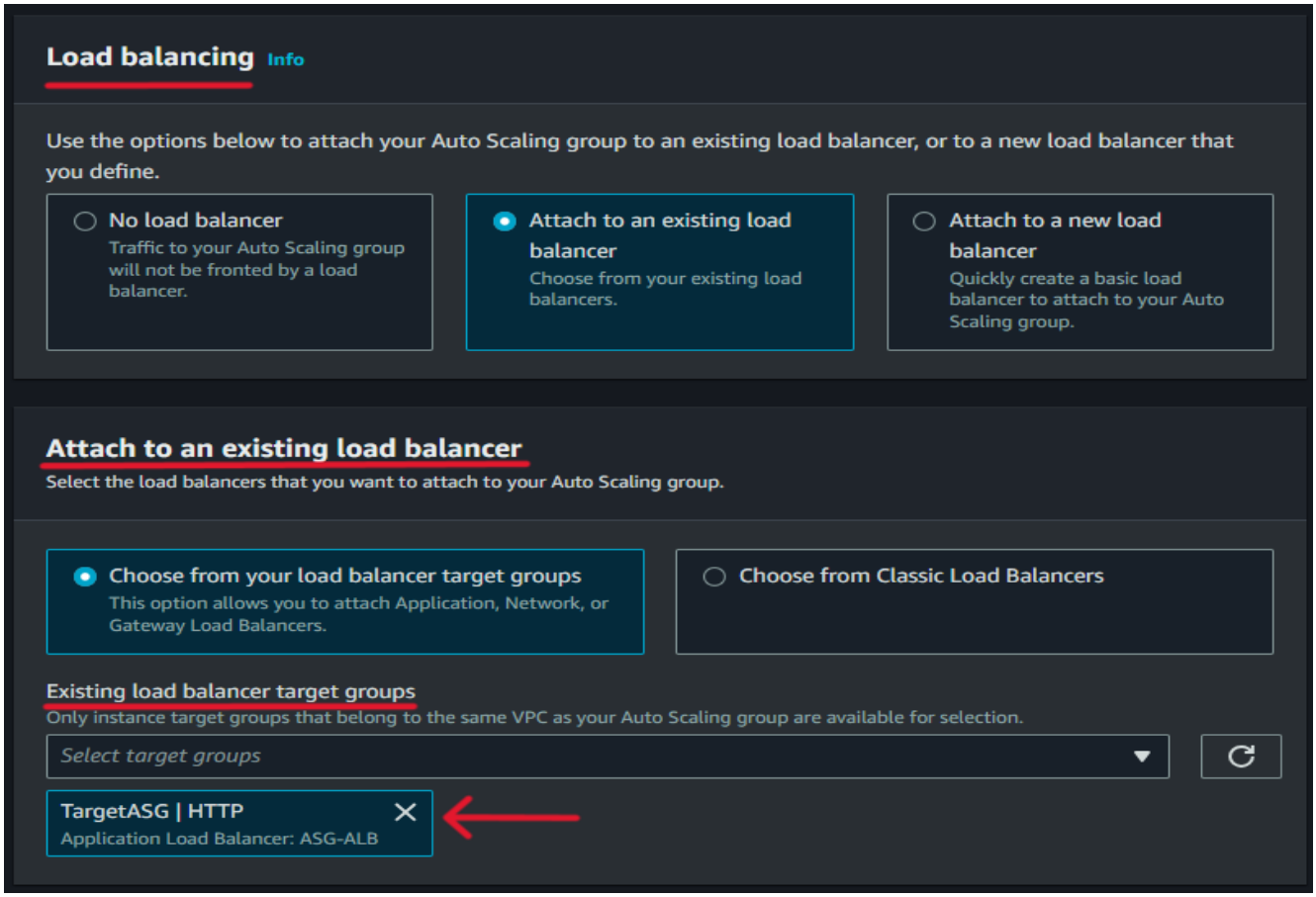
Step 4) In the Instance type requirement, you can either use the same instance settings as your launch template or specify different settings manually. Now we go with the template setting.



Step 5) In the Network option, we select VPC and define the availability zone and subnets, where our instance will launch. Then click on the next button.



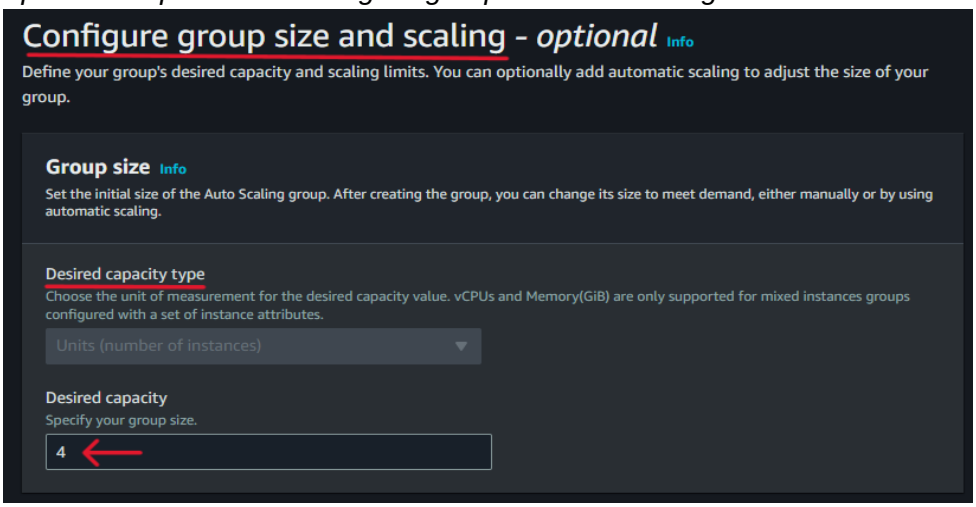
Step 6) In the Load balancing option, attach to an existing load balancer. Select load balancer and target group.

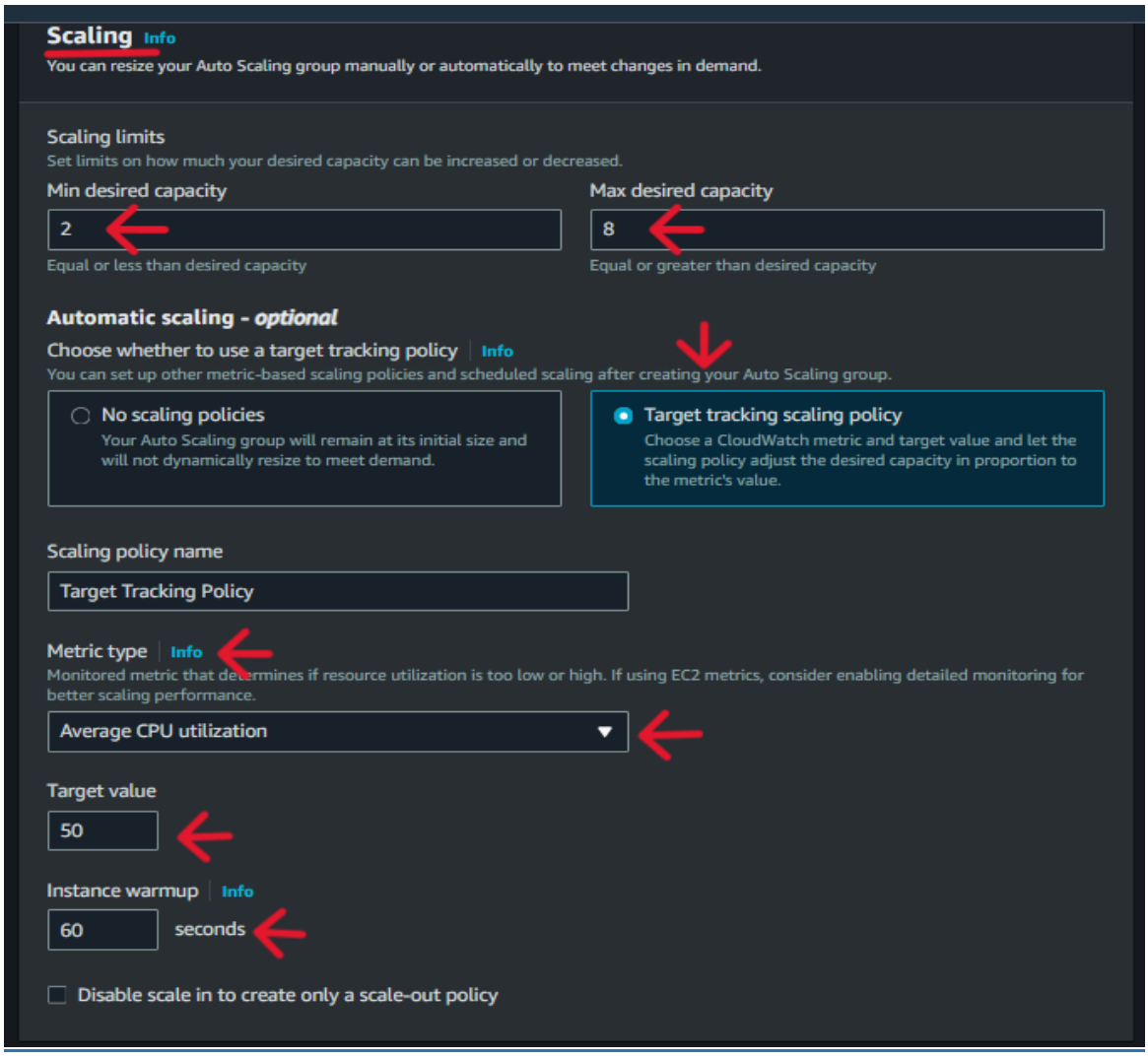


Also, to have more options as shown in the image we can create a new load balancer, and in this have the option to create target groups.

Other settings rest as by default.

Step 7) Now this is our important step here we configure group size and scaling. Shown as in the image.





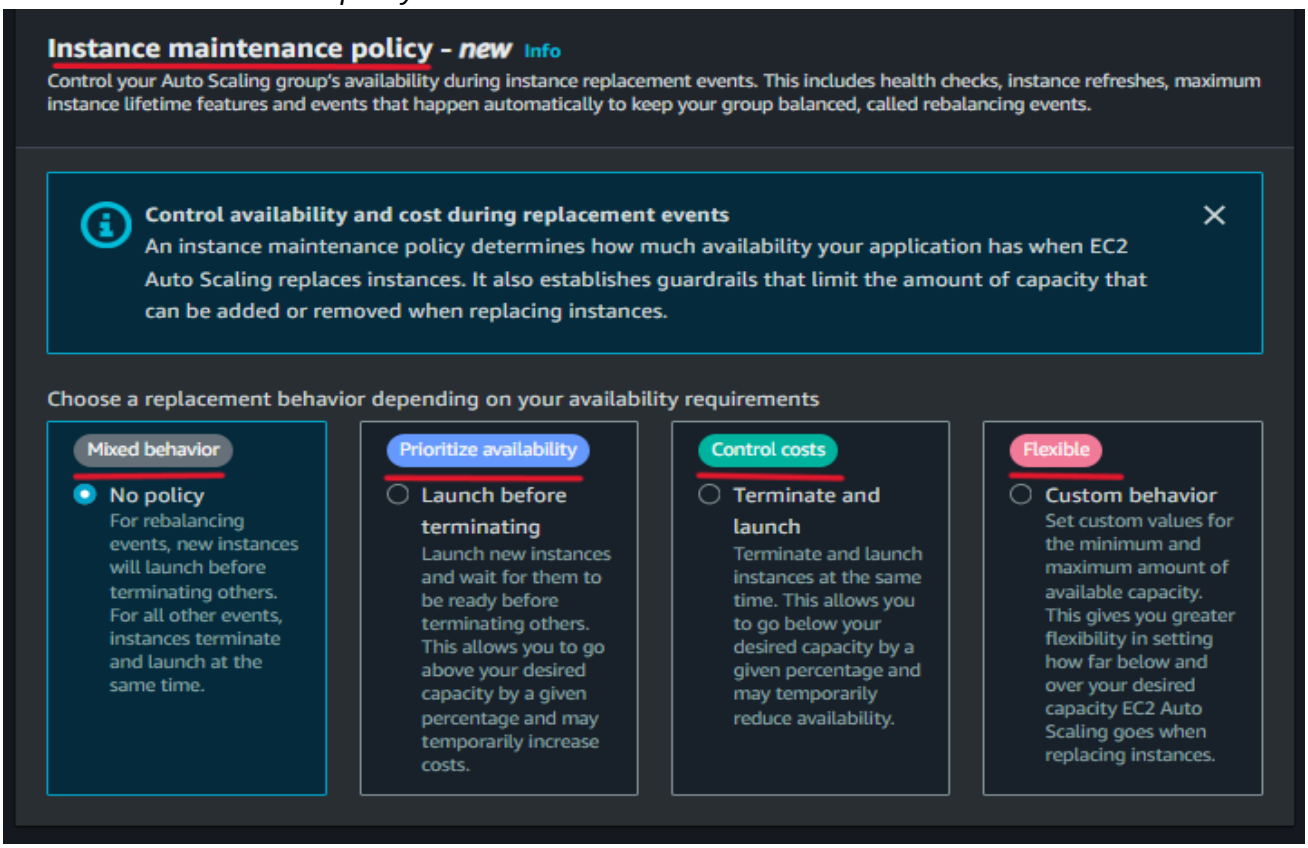
<u>Define desired capacity</u>: Set the number of instances you want to maintain in your Auto Scaling group.

<u>Create scaling policies</u>: Configure rules to automatically adjust instance count based on metrics like CPU usage or network traffic.

<u>Set minimum and maximum instances</u>: Specify the range of instances allowed to ensure scalability and cost control.

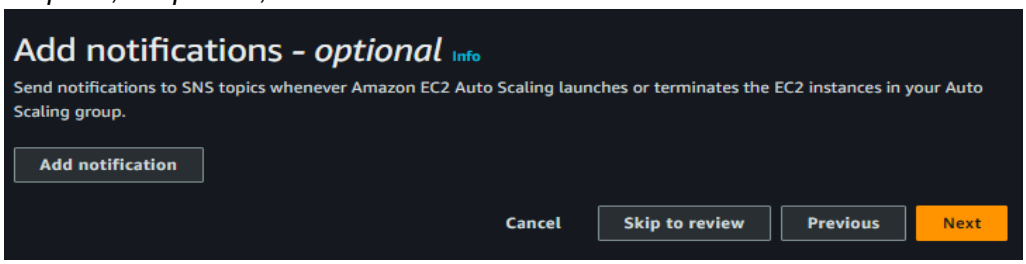
<u>Set cooldown periods:</u> Add cooldown periods to prevent rapid scaling actions in response to fluctuations.

Step 8) Select Instance maintenance policy. click on the next button



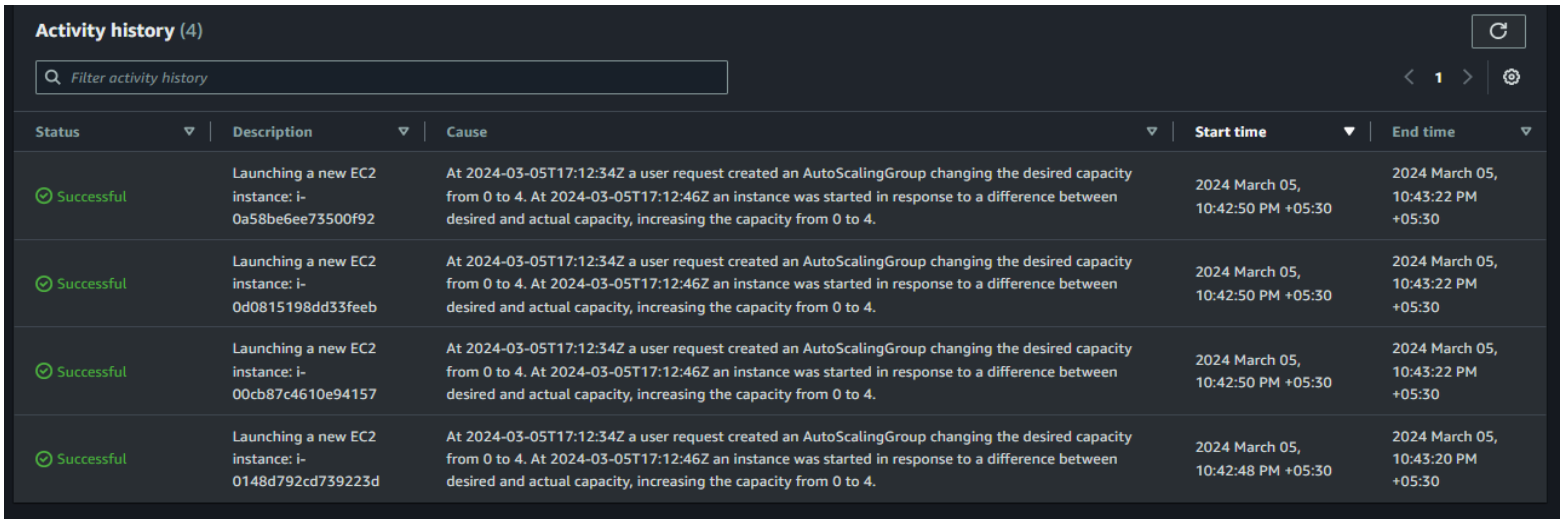
The instance maintenance policy defines how instances should be managed during scheduled events or when a problem occurs. These policies ensure that your applications remain available and responsive even when instances are being replaced or taken out of service.

Step 9)Add notifications option, is optional, click on the next button.

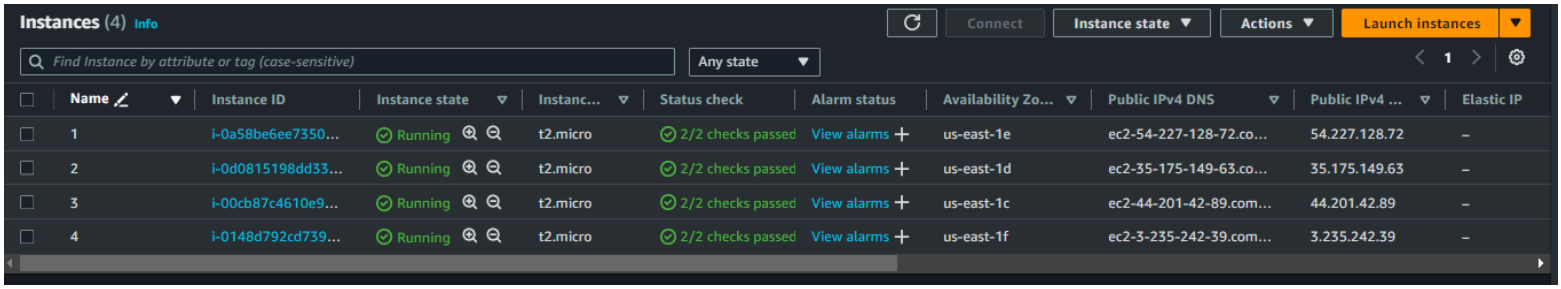


Step 10) Add Tag option, Add or you can skip by clicking the Next button.
Step 11) Review page: check all details, Configurations, and settings are correct according to your use case, then simply click on Create Auto Scaling Group Button.

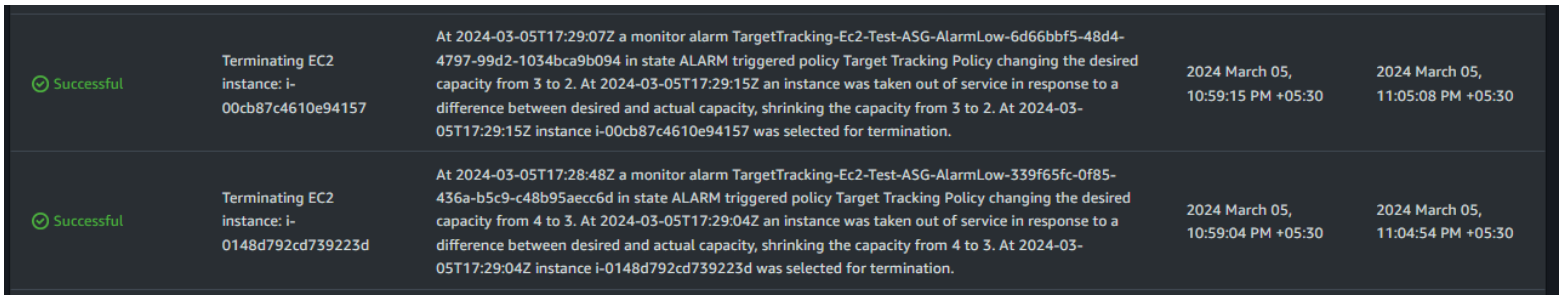
Now you will see your Auto Scaling Group is created.
It will take some time to work, go to the activity, and check the activity history. Our instance will launched successfully.

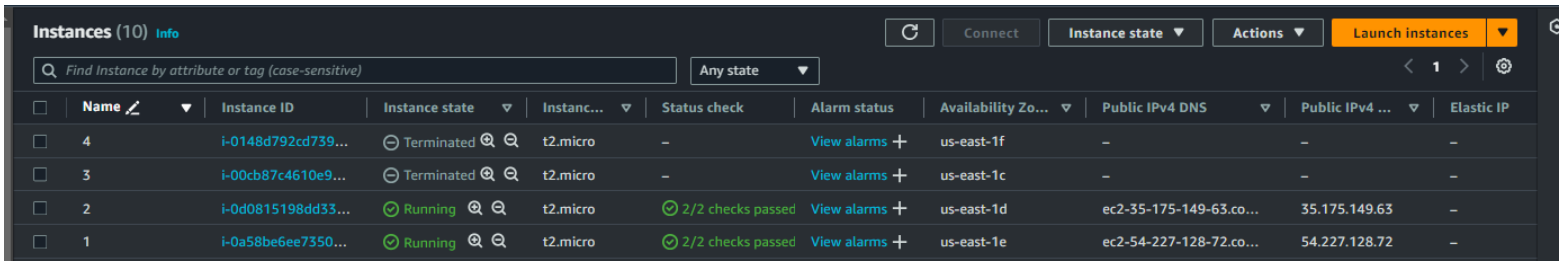


Step 12) Go to the Ec2 Instance dashboard and check whether Instances are properly running or not. We will see our 4 instances are properly running



Step 13) Now do nothing for 15 to 20 min. and then check the activity history in the auto scaling group.



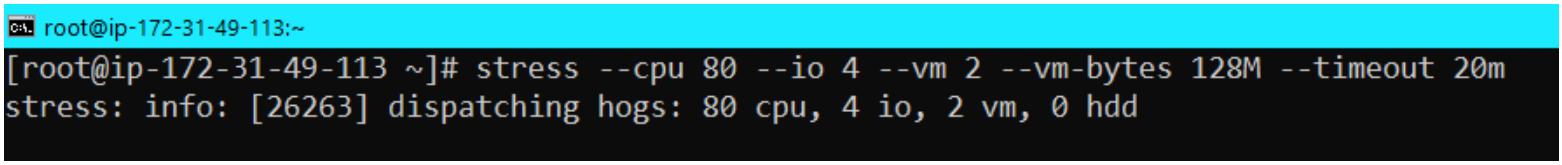


In the above image, we can see that the 3rd and 4th Instances have terminated because we have set the minimum desired capacity at 2 when no more traffic or utilization.

Step 13) Now connect to Instances and put artificial stress into the CPU utilization.

We need to install the Stress Package

Commands to install Stress Package: -> yum install stress –y
To check all options: -> tress --help
Simply copy the example command from the help menu, Paste, and change the 80 % stress level of CPU and period for 20 min then run it.



Do the same with the 2nd instance.

Auto Scaling Group will automatically launch new instances when the instance CPU utilization reaches 80%.

Step 14) Again go to the activity history option of auto scaling group and check the new activity.

Successful	Launching a new EC2 instance: i-07dcfc399d89f5fa9	At 2024-03-05T17:38:59Z a monitor alarm TargetTracking-Ec2-Test-ASG-AlarmHigh-511a2c40-f0d0-4a03-a509-6fe03b75ed6d in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 6 to 8. At 2024-03-05T17:39:11Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 6 to 8.	2024 March 05, 11:09:13 PM +05:30	2024 March 05, 11:10:45 PM +05:30
Successful	Launching a new EC2 instance: i-0be49e767c3f2e245	At 2024-03-05T17:38:59Z a monitor alarm TargetTracking-Ec2-Test-ASG-AlarmHigh-511a2c40-f0d0-4a03-a509-6fe03b75ed6d in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 6 to 8. At 2024-03-05T17:39:11Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 6 to 8.	2024 March 05, 11:09:13 PM +05:30	2024 March 05, 11:10:45 PM +05:30
Successful	Launching a new EC2 instance: i-07ec8c664ef32e606	At 2024-03-05T17:35:59Z a monitor alarm TargetTracking-Ec2-Test-ASG-AlarmHigh-511a2c40-f0d0-4a03-a509-6fe03b75ed6d in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 4 to 6. At 2024-03-05T17:36:04Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 4 to 6.	2024 March 05, 11:06:07 PM +05:30	2024 March 05, 11:07:38 PM +05:30
Successful	Launching a new EC2 instance: i-0e4493345e3a31b94	At 2024-03-05T17:35:59Z a monitor alarm TargetTracking-Ec2-Test-ASG-AlarmHigh-511a2c40-f0d0-4a03-a509-6fe03b75ed6d in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 4 to 6. At 2024-03-05T17:36:04Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 4 to 6.	2024 March 05, 11:06:06 PM +05:30	2024 March 05, 11:07:38 PM +05:30
Successful	Launching a new EC2 instance: i-09a6133c903e91b3a	At 2024-03-05T17:33:59Z a monitor alarm TargetTracking-Ec2-Test-ASG-AlarmHigh-511a2c40-f0d0-4a03-a509-6fe03b75ed6d in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 2 to 4. At 2024-03-05T17:34:01Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 2 to 4.	2024 March 05, 11:04:03 PM +05:30	2024 March 05, 11:05:35 PM +05:30
Successful	Launching a new EC2 instance: i-09e88db157df386ce	At 2024-03-05T17:33:59Z a monitor alarm TargetTracking-Ec2-Test-ASG-AlarmHigh-511a2c40-f0d0-4a03-a509-6fe03b75ed6d in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 2 to 4. At 2024-03-05T17:34:01Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 2 to 4.	2024 March 05, 11:04:03 PM +05:30	2024 March 05, 11:05:35 PM +05:30

We will see that the new 6 instances have been launched successfully.

Step 15) Go to the Ec2 Instance dashboard and check whether Instances are properly running or not. We will see our 6 instances are properly running.

Instances (10) Info

Connect | Instance state | Actions | Launch instances

Find Instance by attribute or tag (case-sensitive) | Any state | 1

Name	Instance ID	Instance state	Instanc...	Status check	Alarm status	Availability Zo...	Public IPv4 DNS	Public IPv4 ...	Elastic IP
4	i-0148d792cd739...	Terminated	t2.micro	–	View alarms	us-east-1f	–	–	–
3	i-00cb87c4610e9...	Terminated	t2.micro	–	View alarms	us-east-1c	–	–	–
2	i-0d0815198dd33...	Running	t2.micro	2/2 checks passed	View alarms	us-east-1d	ec2-35-175-149-63.co...	35.175.149.63	–
1	i-0a58be6ee7350...	Running	t2.micro	2/2 checks passed	View alarms	us-east-1e	ec2-54-227-128-72.co...	54.227.128.72	–
	i-07dcfc399d89f5...	Running	t2.micro	2/2 checks passed	View alarms	us-east-1e	ec2-52-86-72-6.comput...	52.86.72.6	–
	i-07ec8c664ef32e...	Running	t2.micro	2/2 checks passed	View alarms	us-east-1c	ec2-44-204-125-55.co...	44.204.125.55	–
	i-09e88db157df3...	Running	t2.micro	2/2 checks passed	View alarms	us-east-1b	ec2-54-146-108-94.co...	54.146.108.94	–
	i-0be49e767c3f2e...	Running	t2.micro	2/2 checks passed	View alarms	us-east-1f	ec2-44-220-53-83.com...	44.220.53.83	–
	i-0e4493345e3a3...	Running	t2.micro	2/2 checks passed	View alarms	us-east-1f	ec2-44-197-201-80.co...	44.197.201.80	–
	i-09a6133c903e9...	Running	t2.micro	2/2 checks passed	View alarms	us-east-1a	ec2-54-242-142-248.co...	54.242.142.248	–

We set the maximum desired capacity at 6, so as the stress on the Instances increased, 4 more Instances were launched by Auto Scaling Group.