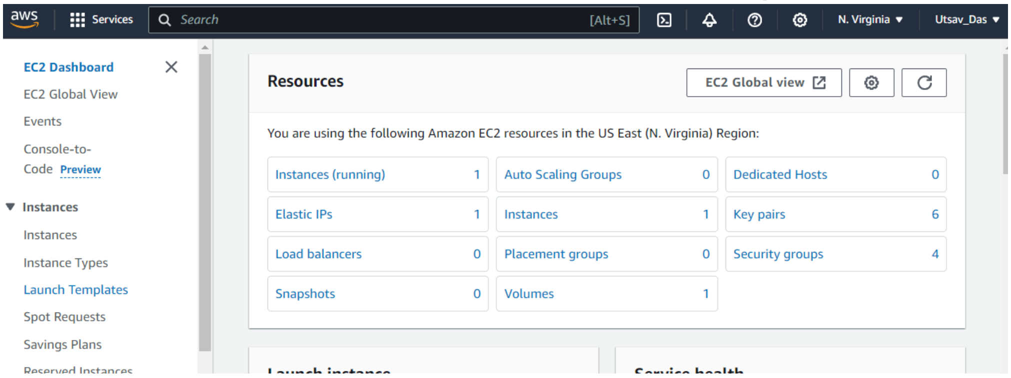
Assignment – 11

Problem Statement:

Build scaling plans in AWS that balance the load on different EC2 instances.

1) Go to EC2 then in Instances click on Launch Templates.



Resources

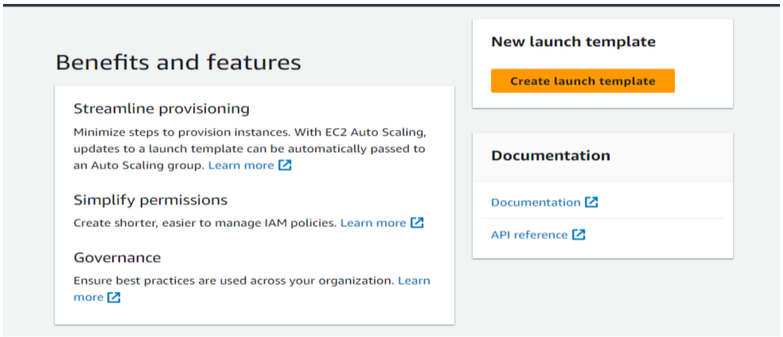
EC2 Global view

You are using the following Amazon EC2 resources in the US East (N. Virginia) Region:

Instances (running)	1	Auto Scaling Groups	0	Dedicated Hosts	0
Elastic IPs	1	Instances	1	Key pairs	6
Load balancers	0	Placement groups	0	Security groups	4
Snapshots	0	Volumes	1		

48

2) Click on Create launch template.


Benefits and features

Streamline provisioning

Minimize steps to provision instances. With EC2 Auto Scaling, updates to a launch template can be automatically passed to an Auto Scaling group. Learn more

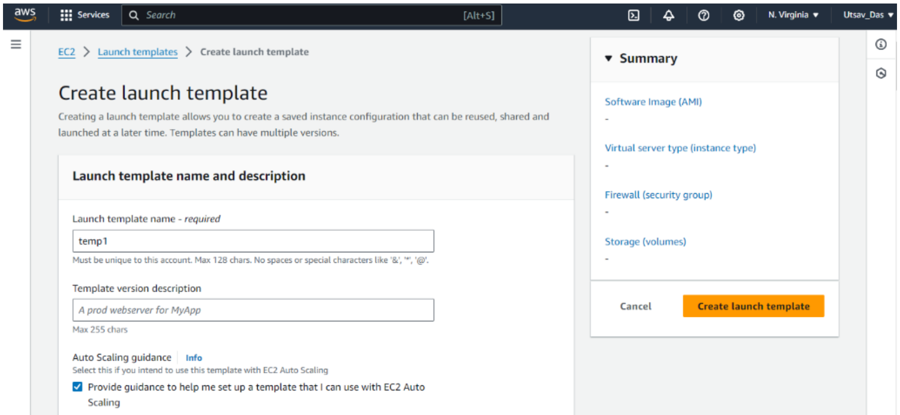
Simplify permissions

Create shorter, easier to manage IAM policies. Learn more

Governance

Ensure best practices are used across your organization. Learn more

3) Give Launch template name and then click on checkbox for Auto Scaling guidance.



Create launch template

Creating a launch template allows you to create a saved instance configuration that can be reused, shared and launched at a later time. Templates can have multiple versions.

Launch template name and description

Launch template name - *required*

temp1

Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '*', '@'.

Template version description

A prod webserver for MyApp

Max 255 chars

Auto Scaling guidance Info

Select this if you intend to use this template with EC2 Auto Scaling

☑ Provide guidance to help me set up a template that I can use with EC2 Auto Scaling

▼ Summary

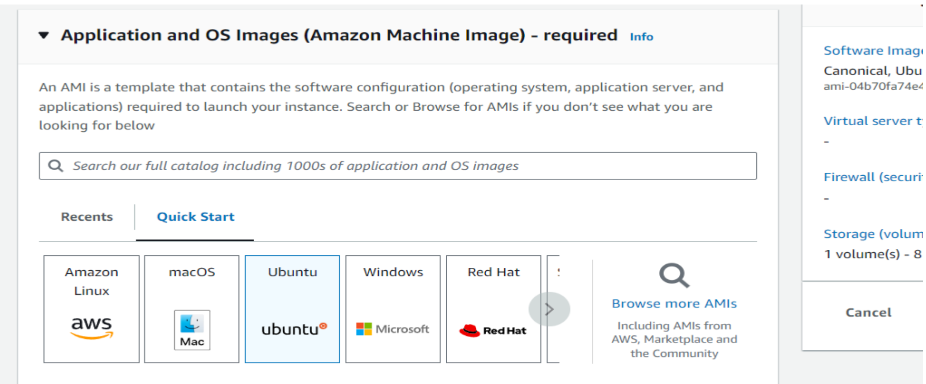
Software Image (AMI)
-

Virtual server type (instance type)
-

Firewall (security group)
-

Storage (volumes)
-

4) In Quick Start select Ubuntu.


▼ Application and OS Images (Amazon Machine Image) - required Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

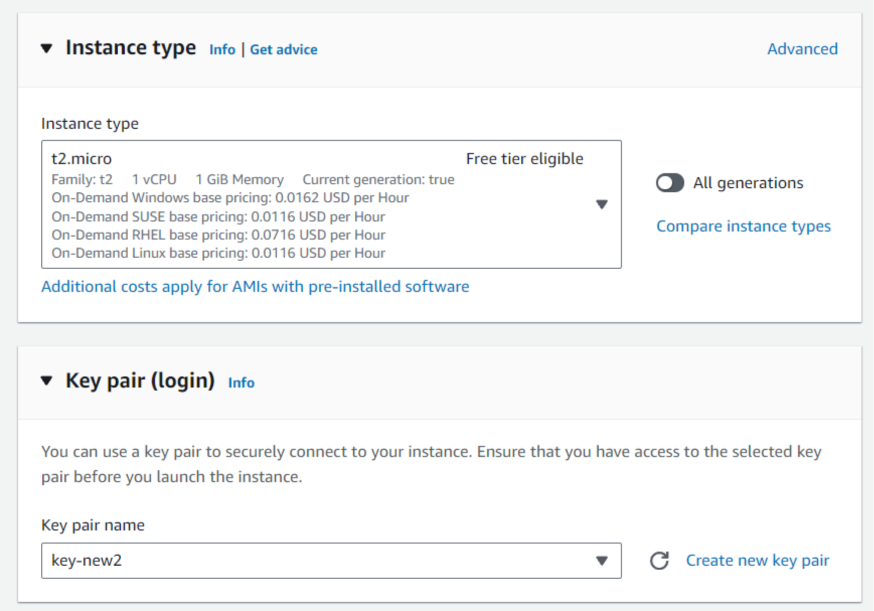
Software Imag
Canonical, Ubu
ami-04b70fa74e4

Virtual server t
-

Firewall (securi
-

Storage (volum
1 volume(s) - 8

5) Select Instance type – either t2.micro or t3.micro. Create new key pair.



▼ Instance type Info | Get advice — Advanced

Instance type

t2.micro — Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour
On-Demand RHEL base pricing: 0.0716 USD per Hour
On-Demand Linux base pricing: 0.0116 USD per Hour

All generations

Compare instance types

Additional costs apply for AMIs with pre-installed software

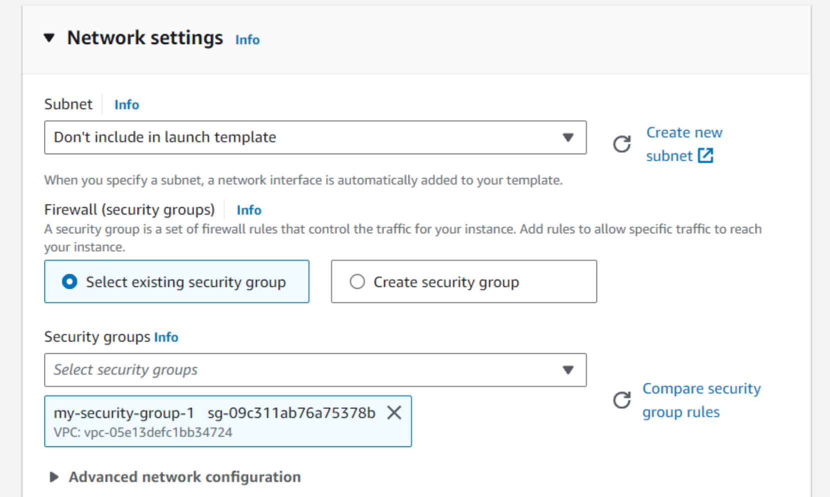
▼ Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name: key-new2

Create new key pair

6) Select existing security group.



▼ Network settings Info

Subnet Info: Don't include in launch template

Create new subnet

When you specify a subnet, a network interface is automatically added to your template.

Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

● Select existing security group ○ Create security group

Security groups Info: Select security groups

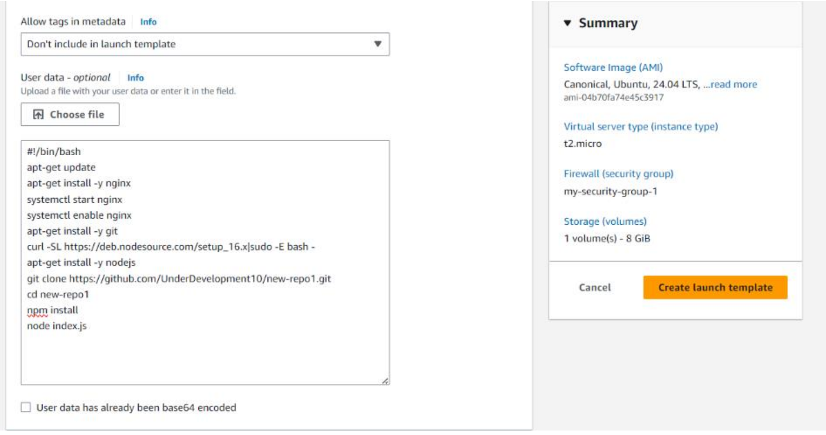
my-security-group-1 sg-09c311ab76a75378b ✕
VPC: vpc-05e13defc1bb34724

Compare security group rules

▶ Advanced network configuration

7) Now go to Advanced details and then in User data write given statements and then Create launch template.

```
#!/bin/bash
apt -get update
apt -get install -y nginx
systemctl start nginx
systemctl enable nginx
apt -get install -y git
curl -SL https://deb.nodesource.com/setup_16.x|sudo -E bash -
apt -get install -y nodejs
git clone https://github.com/UnderDevelopment10/new-repo1.git
cd new-repo1
npm install
node index.js
```



Allow tags in metadata Info

Don't include in launch template

User data - *optional* Info

Upload a file with your user data or enter it in the field.

Choose file

```
#!/bin/bash
apt-get update
apt-get install -y nginx
systemctl start nginx
systemctl enable nginx
apt-get install -y git
curl -SL https://deb.nodesource.com/setup_16.x|sudo -E bash -
apt-get install -y nodejs
git clone https://github.com/UnderDevelopment10/new-repo1.git
cd new-repo1
npm install
node index.js
```

□ User data has already been base64 encoded

Summary

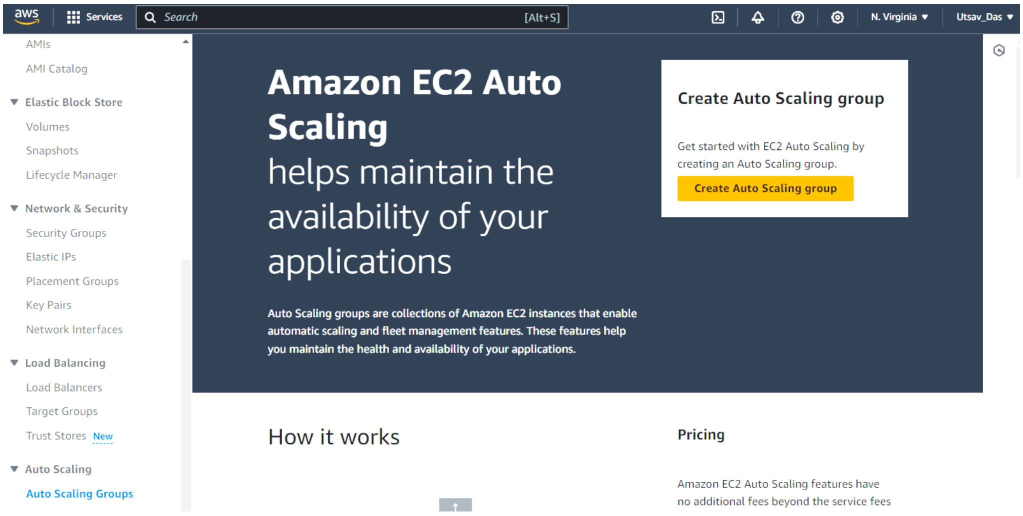
Software Image (AMI)
Canonical, Ubuntu, 24.04 LTS, ...read more
ami-04b70fa74e45c3917

Virtual server type (instance type)
t2.micro

Firewall (security group)
my-security-group-1

Storage (volumes)
1 volume(s) - 8 GiB

8) After creating Launch template, click on Auto Scaling Groups in left pane. Click on Create Auto Scaling group.



Amazon EC2 Auto Scaling helps maintain the availability of your applications

Auto Scaling groups are collections of Amazon EC2 instances that enable automatic scaling and fleet management features. These features help you maintain the health and availability of your applications.

Create Auto Scaling group

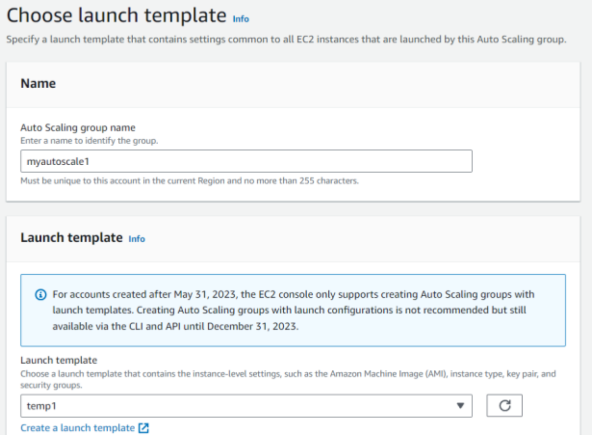
Get started with EC2 Auto Scaling by creating an Auto Scaling group.

How it works

Pricing

Amazon EC2 Auto Scaling features have no additional fees beyond the service fees

9) Give Auto scaling group name and then select your recently launched template. Then click on Next.



Choose launch template Info

Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group.

Name

Auto Scaling group name
Enter a name to identify the group.

myautoscale1

Must be unique to this account in the current Region and no more than 255 characters.

Launch template Info

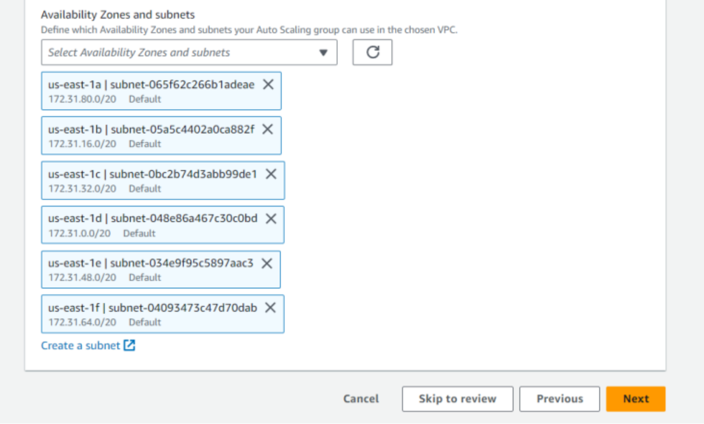
For accounts created after May 31, 2023, the EC2 console only supports creating Auto Scaling groups with launch templates. Creating Auto Scaling groups with launch configurations is not recommended but still available via the CLI and API until December 31, 2023.

Launch template
Choose a launch template that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.

temp1

Create a launch template

10)Select availability zones and subnets then click on Next.



Availability Zones and subnets
Define which Availability Zones and subnets your Auto Scaling group can use in the chosen VPC.

Select Availability Zones and subnets

us-east-1a | subnet-065f62c266b1adeae
172.31.80.0/20 Default

us-east-1b | subnet-05a5c4402a0ca882f
172.31.16.0/20 Default

us-east-1c | subnet-0bc2b74d3abb99de1
172.31.32.0/20 Default

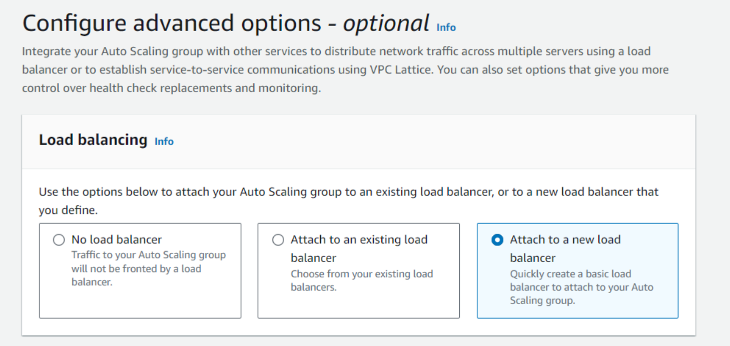
us-east-1d | subnet-048e86a467c30c0bd
172.31.0.0/20 Default

us-east-1e | subnet-034e9f95c5897aac3
172.31.48.0/20 Default

us-east-1f | subnet-04093473c47d70dab
172.31.64.0/20 Default

Create a subnet

11)Click on Attach to a new load balancer.



Configure advanced options - *optional* Info

Integrate your Auto Scaling group with other services to distribute network traffic across multiple servers using a load balancer or to establish service-to-service communications using VPC Lattice. You can also set options that give you more control over health check replacements and monitoring.

Load balancing Info

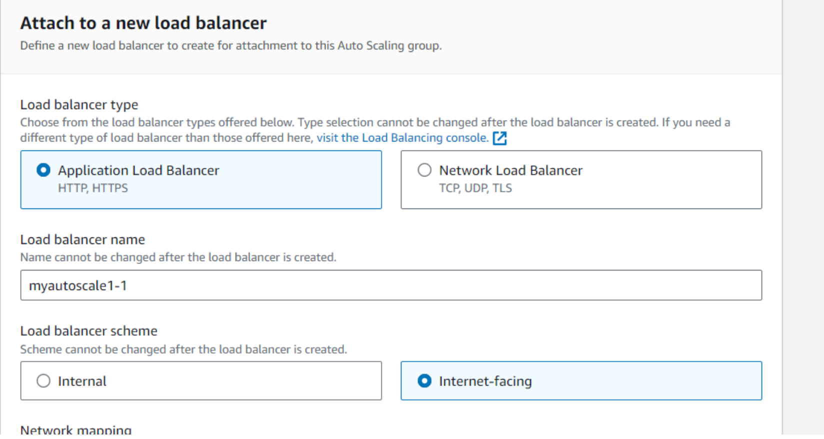
Use the options below to attach your Auto Scaling group to an existing load balancer, or to a new load balancer that you define.

○ No load balancer
Traffic to your Auto Scaling group will not be fronted by a load balancer.

○ Attach to an existing load balancer
Choose from your existing load balancers.

● Attach to a new load balancer
Quickly create a basic load balancer to attach to your Auto Scaling group.

12)Select Application Load Balancer as load balancer type and Internet-facing as Load balancer scheme.



Attach to a new load balancer

Define a new load balancer to create for attachment to this Auto Scaling group.

Load balancer type

Choose from the load balancer types offered below. Type selection cannot be changed after the load balancer is created. If you need a different type of load balancer than those offered here, visit the Load Balancing console.

● Application Load Balancer
HTTP, HTTPS

○ Network Load Balancer
TCP, UDP, TLS

Load balancer name

Name cannot be changed after the load balancer is created.

myautoscale1-1

Load balancer scheme

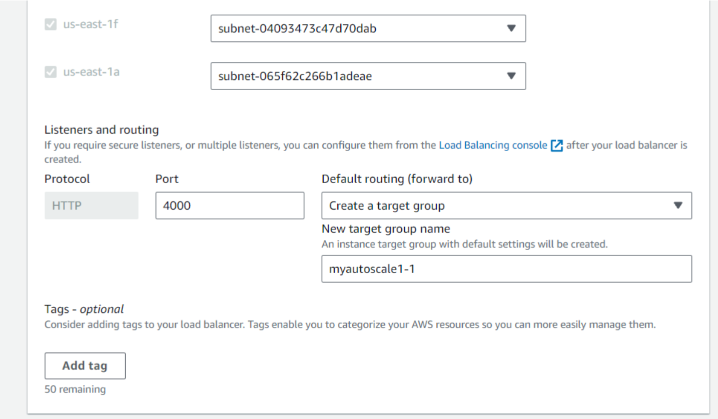
Scheme cannot be changed after the load balancer is created.

○ Internal

● Internet-facing

Network mapping

13)Select port no. 4000 for HTTP and select New target group name.



☑ us-east-1f subnet-04093473c47d70dab ▼

☑ us-east-1a subnet-065f62c266b1adeae ▼

Listeners and routing

If you require secure listeners, or multiple listeners, you can configure them from the Load Balancing console after your load balancer is created.

Protocol	Port	Default routing (forward to)
HTTP	4000	Create a target group ▼

New target group name

An instance target group with default settings will be created.

myautoscale1-1

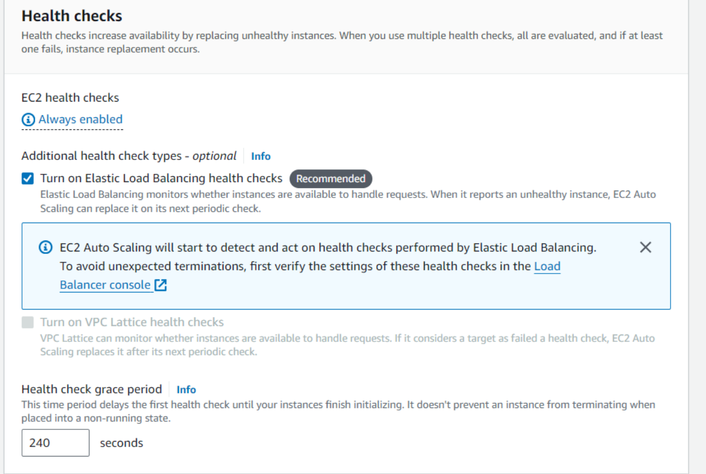
Tags - *optional*

Consider adding tags to your load balancer. Tags enable you to categorize your AWS resources so you can more easily manage them.

Add tag

50 remaining

14)Now turn on Elastic load balancing health check and in health check grace period give 240 seconds. Then click on Next.



Health checks

Health checks increase availability by replacing unhealthy instances. When you use multiple health checks, all are evaluated, and if at least one fails, instance replacement occurs.

EC2 health checks

Always enabled

Additional health check types - *optional* Info

☑ Turn on Elastic Load Balancing health checks Recommended
Elastic Load Balancing monitors whether instances are available to handle requests. When it reports an unhealthy instance, EC2 Auto Scaling can replace it on its next periodic check.

EC2 Auto Scaling will start to detect and act on health checks performed by Elastic Load Balancing. To avoid unexpected terminations, first verify the settings of these health checks in the Load Balancer console ✕

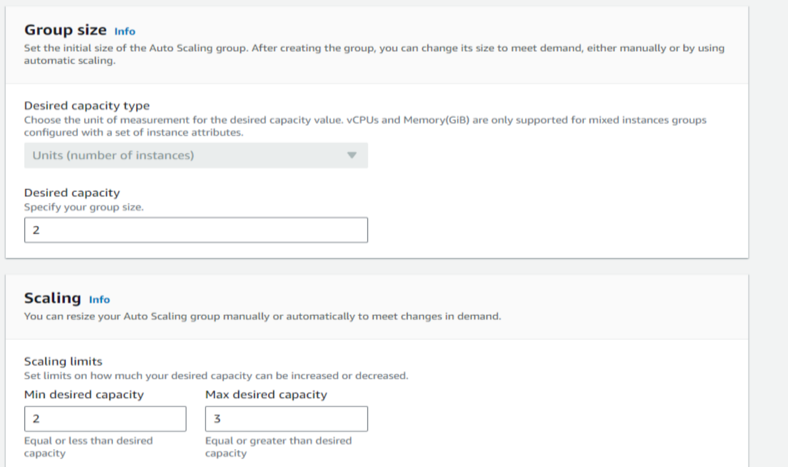
□ Turn on VPC Lattice health checks
VPC Lattice can monitor whether instances are available to handle requests. If it considers a target as failed a health check, EC2 Auto Scaling replaces it after its next periodic check.

Health check grace period Info

This time period delays the first health check until your instances finish initializing. It doesn't prevent an instance from terminating when placed into a non-running state.

240 seconds

15)Select 2 in Desired capacity in group size and in Scaling 2 as Min desired capacity and 3 as Max desired capacity.



Group size Info

Set the initial size of the Auto Scaling group. After creating the group, you can change its size to meet demand, either manually or by using automatic scaling.

Desired capacity type

Choose the unit of measurement for the desired capacity value. vCPUs and Memory(GiB) are only supported for mixed instances groups configured with a set of instance attributes.

Units (number of instances)

Desired capacity

Specify your group size.

2

Scaling Info

You can resize your Auto Scaling group manually or automatically to meet changes in demand.

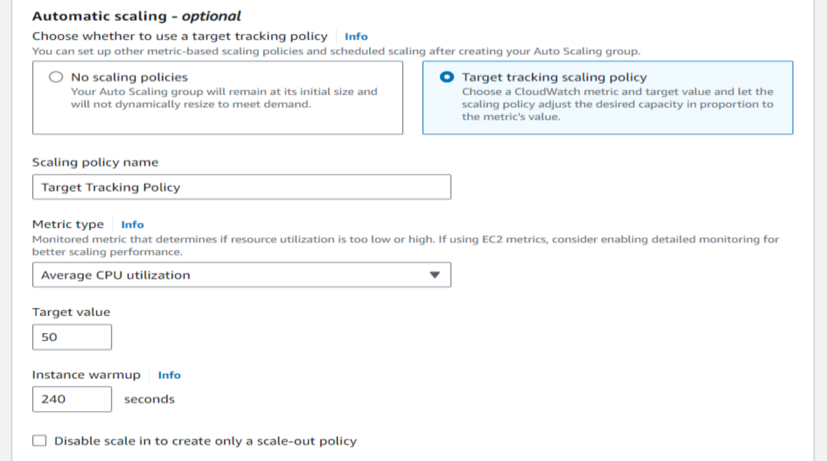
Scaling limits

Set limits on how much your desired capacity can be increased or decreased.

Min desired capacity: 2 (Equal or less than desired capacity)

Max desired capacity: 3 (Equal or greater than desired capacity)

16)Next in Automatic scaling select Target tracking scaling policy and give 240 in Instance warmup.



Automatic scaling - *optional*

Choose whether to use a target tracking policy Info

You can set up other metric-based scaling policies and scheduled scaling after creating your Auto Scaling group.

○ No scaling policies
Your Auto Scaling group will remain at its initial size and will not dynamically resize to meet demand.

● Target tracking scaling policy
Choose a CloudWatch metric and target value and let the scaling policy adjust the desired capacity in proportion to the metric's value.

Scaling policy name

Target Tracking Policy

Metric type Info

Monitored metric that determines if resource utilization is too low or high. If using EC2 metrics, consider enabling detailed monitoring for better scaling performance.

Average CPU utilization

Target value

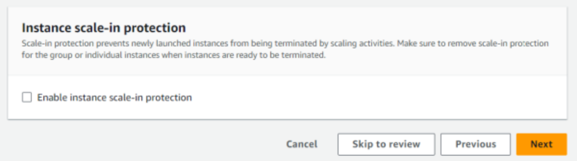
50

Instance warmup Info

240 seconds

□ Disable scale in to create only a scale-out policy

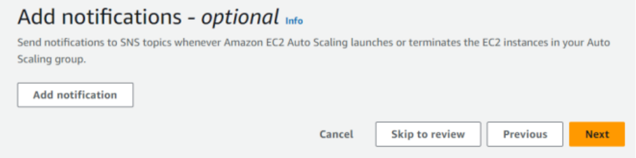
17)Then click on Next, then again next.



Instance scale-in protection

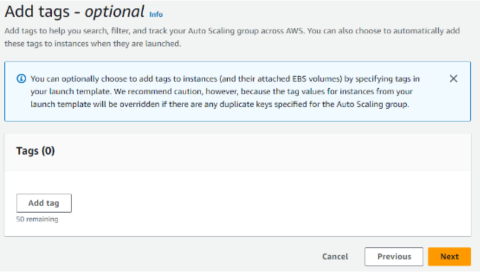
Scale-in protection prevents newly launched instances from being terminated by scaling activities. Make sure to remove scale-in protection for the group or individual instances when instances are ready to be terminated.

□ Enable instance scale-in protection



Add notifications - *optional* Info

Send notifications to SNS topics whenever Amazon EC2 Auto Scaling launches or terminates the EC2 instances in your Auto Scaling group.



Add tags - *optional* Info

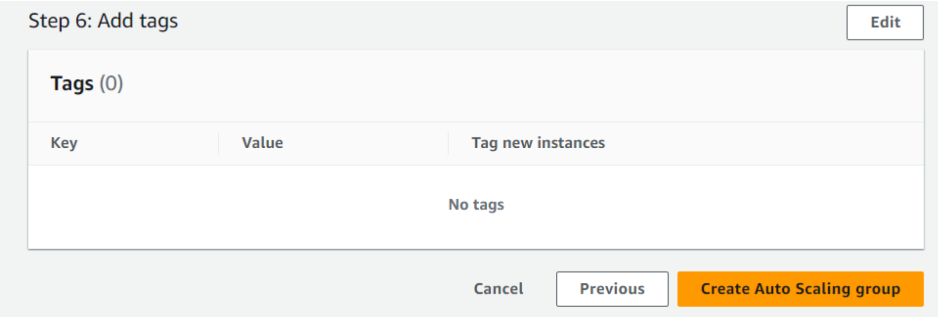
Add tags to help you search, filter, and track your Auto Scaling group across AWS. You can also choose to automatically add these tags to instances when they are launched.

You can optionally choose to add tags to instances (and their attached EBS volumes) by specifying tags in your launch template. We recommend caution, however, because the tag values for instances from your launch template will be overridden if there are any duplicate keys specified for the Auto Scaling group.

Tags (0)

50 remaining

18)Now click on Create Auto Scaling group.

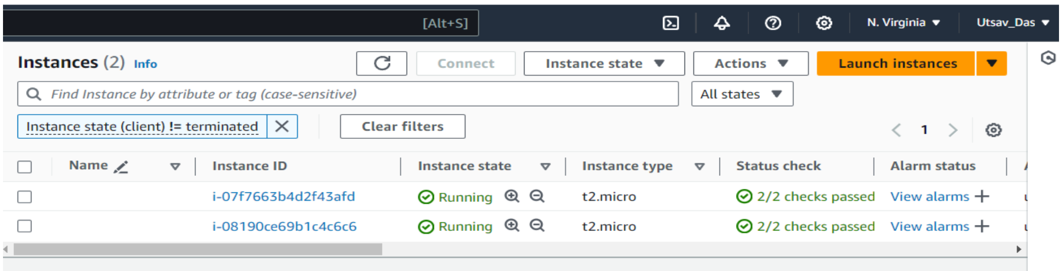


Step 6: Add tags

Tags (0)

Key	Value	Tag new instances
	No tags	

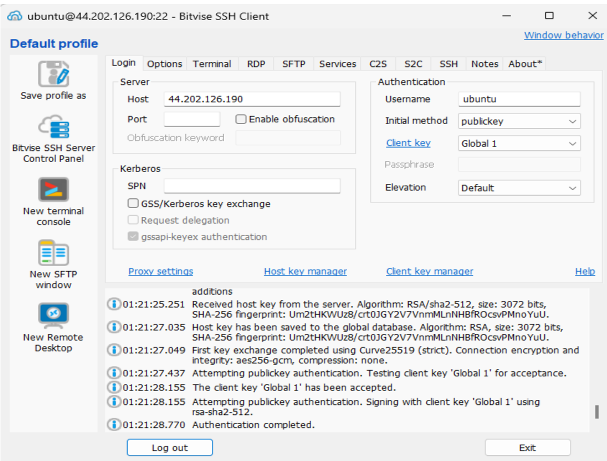
19)Come back to EC2 dashboard and go to Instance and here you can see 2 instances running as minimum capacity 2 is chosen.



☐	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
☐		i-07f7663b4d2f43afd	Running	t2.micro	2/2 checks passed	View alarms +
☐		i-08190ce69b1c4c6c6	Running	t2.micro	2/2 checks passed	View alarms +

20)Click on any instance and then copy public IPv4 address.

21)Open Bitvise SSH client and then paste that public IPv4 address and then in client key manager import the generated key and then Login.



Server
Host 44.202.126.190
Port
☐ Enable obfuscation
Obfuscation keyword

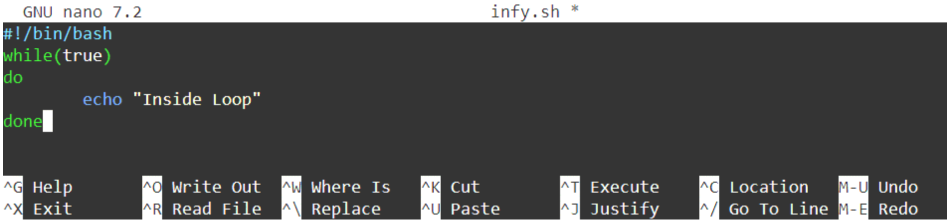
Kerberos
SPN
☐ GSS/Kerberos key exchange
☐ Request delegation
☑ gssapi-keyex authentication

Authentication
Username ubuntu
Initial method publickey
Client key Global 1
Passphrase
Elevation Default

```
                additions
01:21:25.251  Received host key from the server. Algorithm: RSA/sha2-512, size: 3072 bits,
              SHA-256 fingerprint: Um2tHKWUz8/crt0JGY2V7VnmMLnNHBfROcsvPMnoYuU.
01:21:27.035  Host key has been saved to the global database. Algorithm: RSA, size: 3072 bits,
              SHA-256 fingerprint: Um2tHKWUz8/crt0JGY2V7VnmMLnNHBfROcsvPMnoYuU.
01:21:27.049  First key exchange completed using Curve25519 (strict). Connection encryption and
              integrity: aes256-gcm, compression: none.
01:21:27.437  Attempting publickey authentication. Testing client key 'Global 1' for acceptance.
01:21:28.155  The client key 'Global 1' has been accepted.
01:21:28.155  Attempting publickey authentication. Signing with client key 'Global 1' using
              rsa-sha2-512.
01:21:28.770  Authentication completed.
```

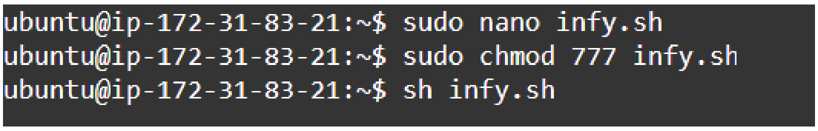
22)Open New terminal console and then write command 'sudo nano infi.sh'. A new .sh file will be created. Now write this code to run an infinite loop.

```
!/bin/bash
while(true)
do
        echo"Inside Loop"
done
```



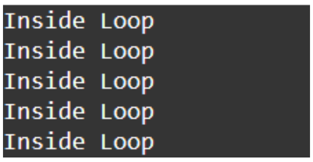
```
  GNU nano 7.2                                  infy.sh *
#!/bin/bash
while(true)
do
        echo "Inside Loop"
done

^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo
```

23)Then do Ctrl+X then Y then click Enter. And write command 'sudo chmod 777 infy.sh' to provide all permissions to file and then to run give 'sh infy.sh'.



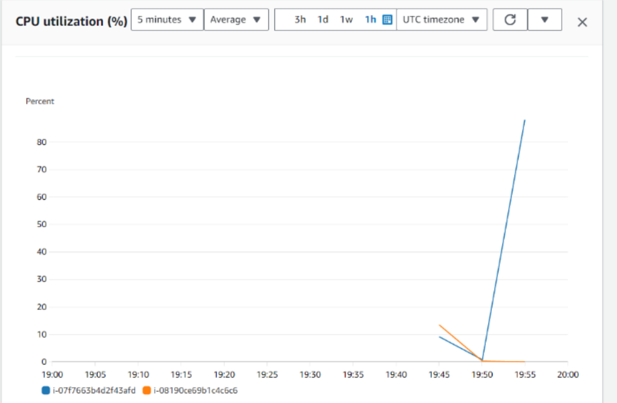
```
ubuntu@ip-172-31-83-21:~$ sudo nano infy.sh
ubuntu@ip-172-31-83-21:~$ sudo chmod 777 infy.sh
ubuntu@ip-172-31-83-21:~$ sh infy.sh
```

24)Here we can see infinite loop running.



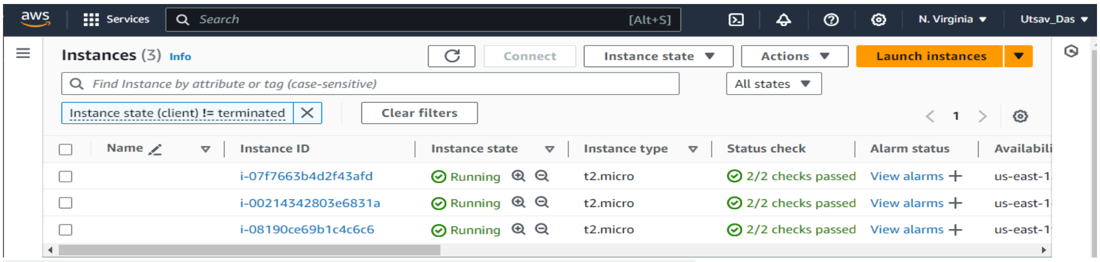
```
Inside Loop
Inside Loop
Inside Loop
Inside Loop
Inside Loop
```

25)Go back to instances and then select both instances and then in bottom pane, go to CPU utilization and select Enlarge.



26)The graph displays CPU utilization for both instances. When CPU utilization for one instance is very high, then another instance will be created as we have set maximum capacity to 3.

27)Finally in Instances, we can see another new instance is created.

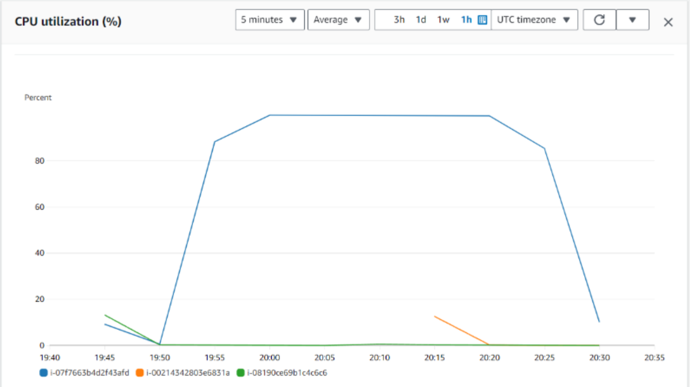


Instances (3) Info

Find Instance by attribute or tag (case-sensitive)

Instance state (client) != terminated

□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availabili
□		i-07f7663b4d2f43afd	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1
□		i-00214342803e6831a	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1
□		i-08190ce69b1c4c6c6	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1


CPU utilization (%)
5 minutes
Average
3h
1d
1w
1h
UTC timezone
Percent
19:40
19:45
19:50
19:55
20:00
20:05
20:10
20:15
20:20
20:25
20:30
20:35
i-07f7663b4d2f43afd
i-00214342803e6831a
i-08190ce69b1c4c6c6