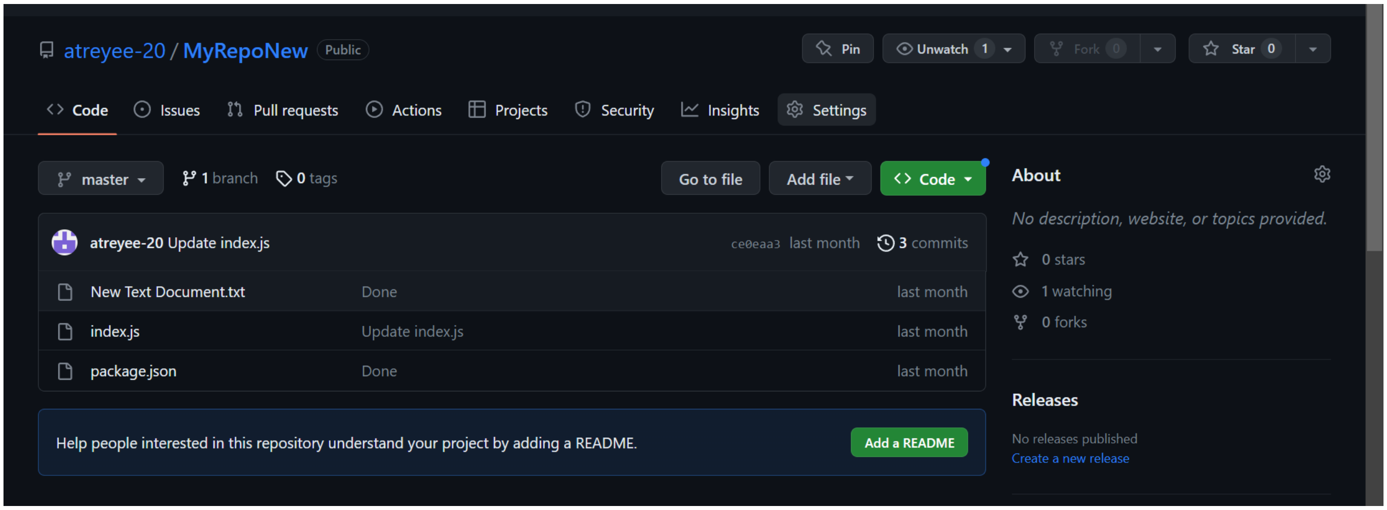
ASSIGNMENT-11

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

Step 1: Sign in to your GitHub account. Go to your repository. Then go to settings.



atreyee-20 Update index.js ce0eaa3 last month 3 commits

New Text Document.txt	Done	last month
index.js	Update index.js	last month
package.json	Done	last month

Help people interested in this repository understand your project by adding a README.

About

No description, website, or topics provided.

0 stars

1 watching

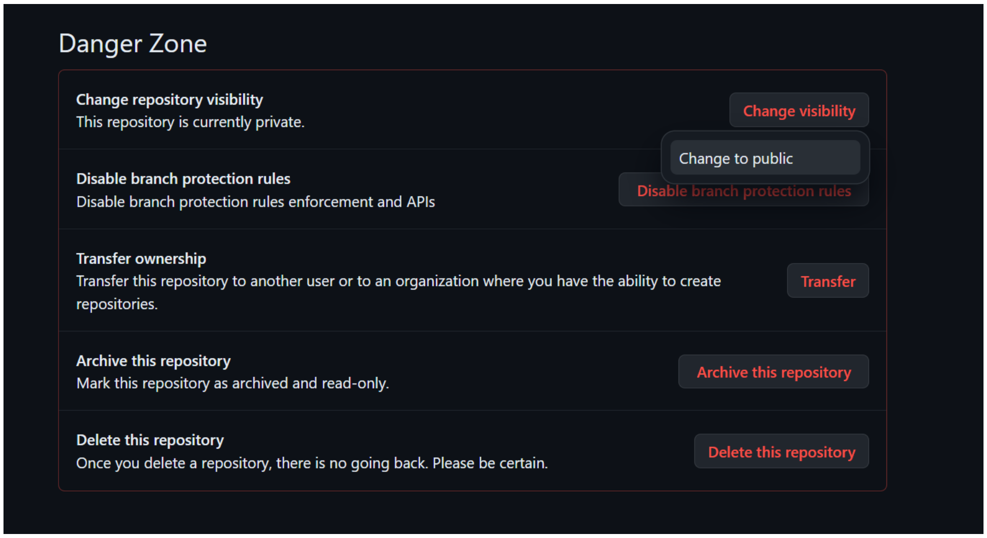
0 forks

Releases

No releases published

Create a new release

Step 2: Scroll down to Danger Zone. Click on Change visibility. Then Change to public.



Danger Zone

Change repository visibility
This repository is currently private.

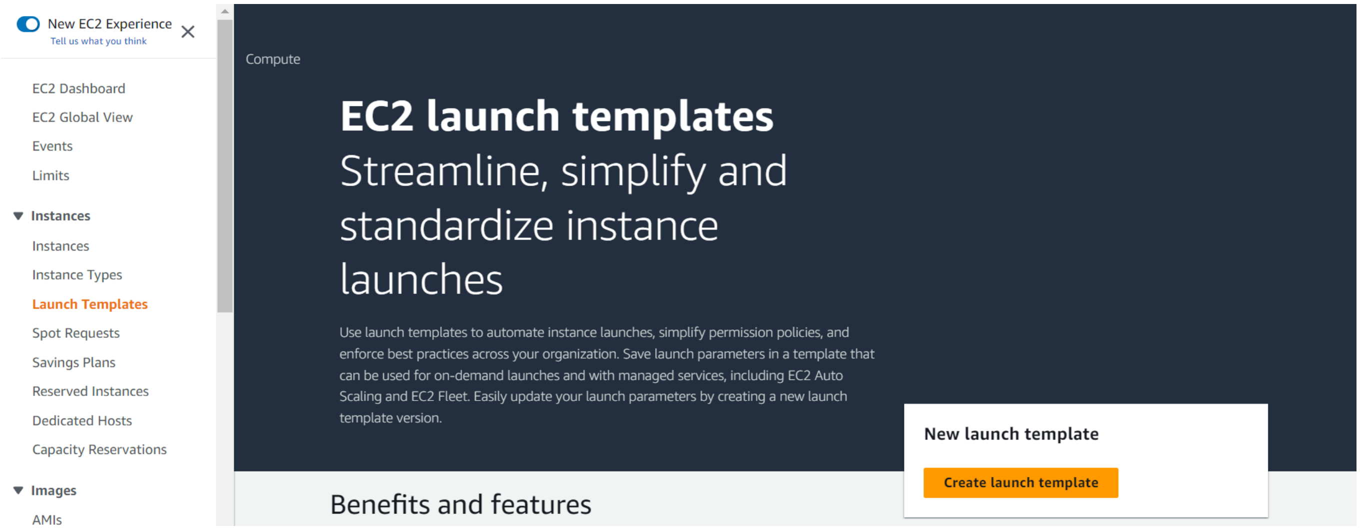
Disable branch protection rules
Disable branch protection rules enforcement and APIs

Transfer ownership
Transfer this repository to another user or to an organization where you have the ability to create repositories.

Archive this repository
Mark this repository as archived and read-only.

Delete this repository
Once you delete a repository, there is no going back. Please be certain.

Step 3: Now Sign in to your AWS account as a root user. Then go to EC2 Dashboard. Click on Launch Templates.



New EC2 Experience
Tell us what you think

EC2 Dashboard
EC2 Global View
Events
Limits

▼ Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations

▼ Images
AMIs

Compute

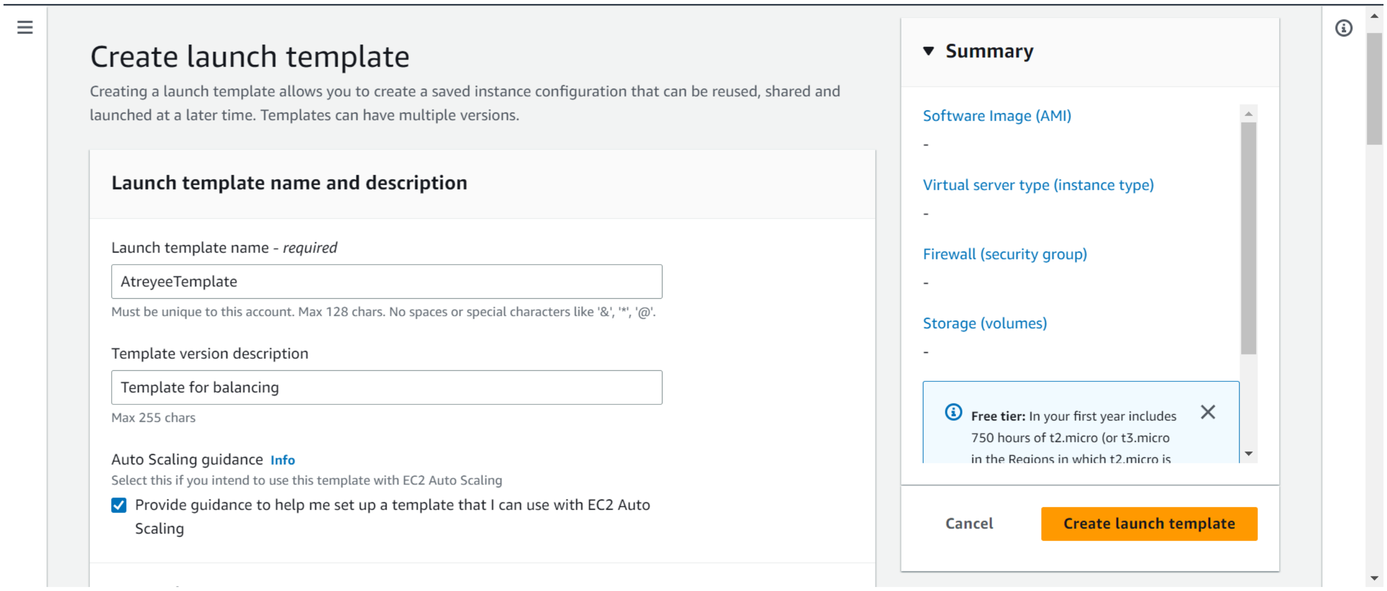
EC2 launch templates
Streamline, simplify and standardize instance launches

Use launch templates to automate instance launches, simplify permission policies, and enforce best practices across your organization. Save launch parameters in a template that can be used for on-demand launches and with managed services, including EC2 Auto Scaling and EC2 Fleet. Easily update your launch parameters by creating a new launch template version.

New launch template
Create launch template

Benefits and features

Step 4: Click on Create Launch Template. Give name and description. Then check Auto Scaling guidance box.



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

Template version description
Template for balancing
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

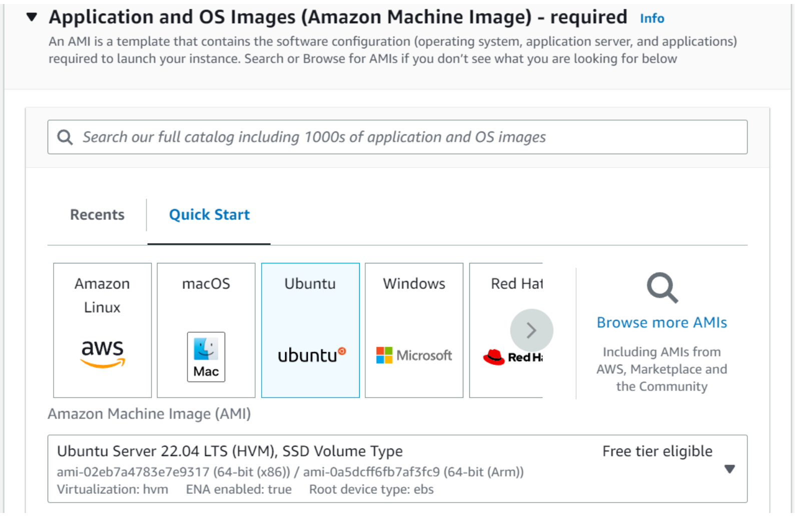
▼ Summary

Software Image (AMI)
-
Virtual server type (instance type)
-
Firewall (security group)
-
Storage (volumes)
-

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is

Cancel
Create launch template

Step 5: Select Ubuntu as OS.



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

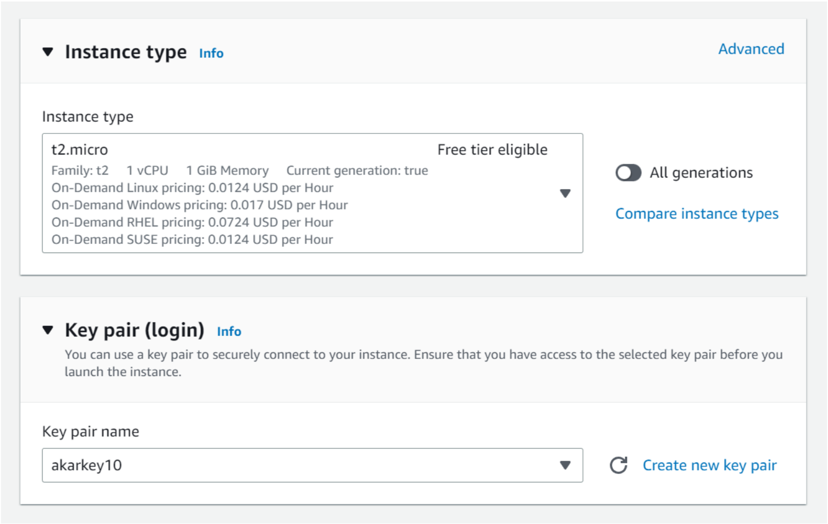
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

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

Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-02eb7a4783e7e9317 (64-bit (x86)) / ami-0a5dcff6fb7af3fc9 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Step 6: Select t2.micro as Instance type and provide a key pair.



▼ Instance type Info

Instance type

t2.micro — Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Linux pricing: 0.0124 USD per Hour
On-Demand Windows pricing: 0.017 USD per Hour
On-Demand RHEL pricing: 0.0724 USD per Hour
On-Demand SUSE pricing: 0.0124 USD per Hour

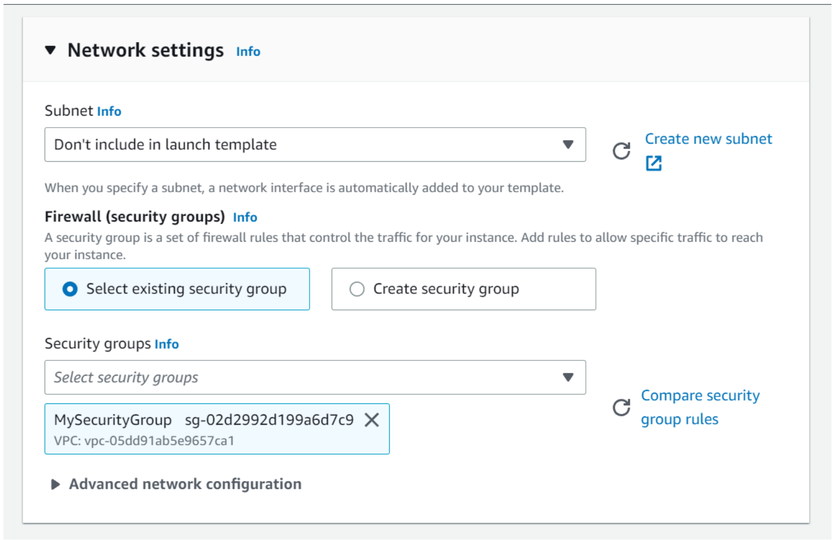
▼ Key pair (login) Info

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

akarkey10

Step 7: Either select existing security group (If there is any) or create new Security Group.



▼ Network settings

Subnet

Don't include in launch template ▼

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

Firewall (security groups)

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.

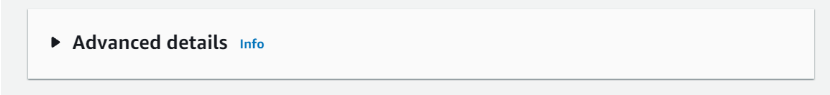
● Select existing security group ○ Create security group

Security groups

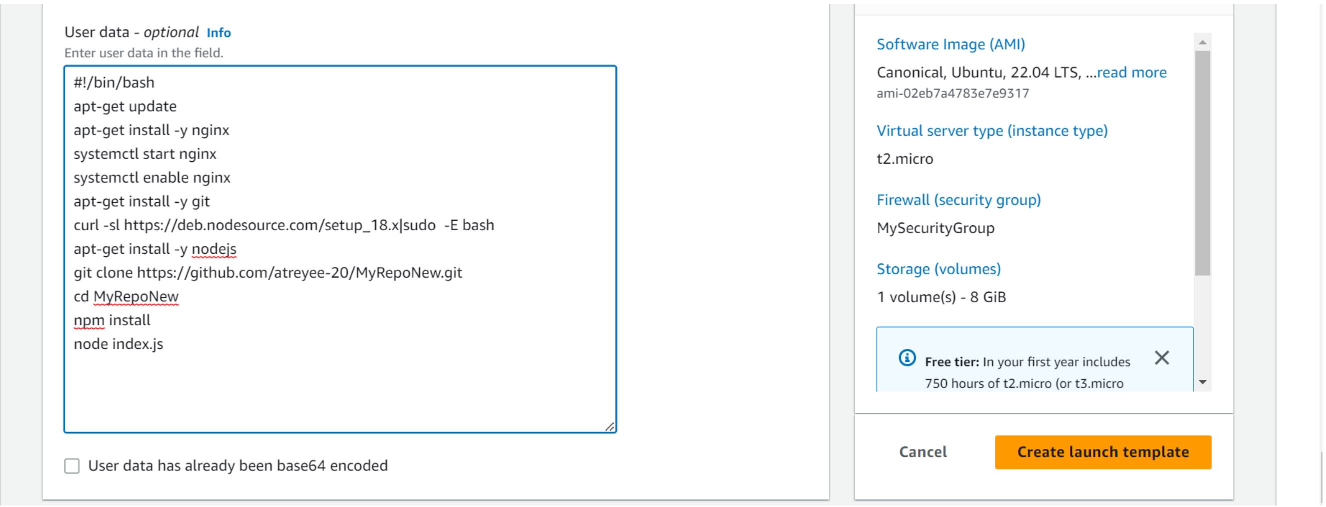
Select security groups ▼

MySecurityGroup sg-02d2992d199a6d7c9 ✕
VPC: vpc-05dd91ab5e9657ca1

Step 8: Go to Advanced details section and then scroll down to User data.



Step 9: Provide the following commands in User data as shown:



User data - *optional*

Enter user data in the field.

```
#!/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_18.x|sudo  -E bash
apt-get install -y nodejs
git clone https://github.com/atreyee-20/MyRepoNew.git
cd MyRepoNew
npm install
node index.js
```

□ User data has already been base64 encoded

Software Image (AMI)
Canonical, Ubuntu, 22.04 LTS, ...read more
ami-02eb7a4783e7e9317

Virtual server type (instance type)
t2.micro

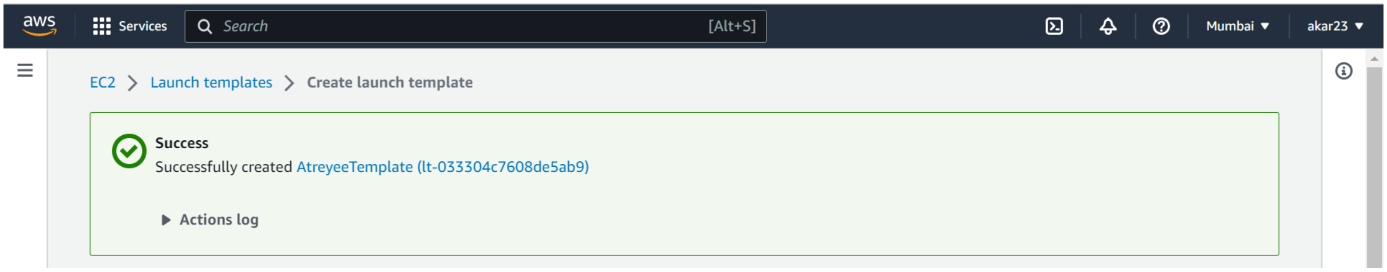
Firewall (security group)
MySecurityGroup

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

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro

Then Click on Create launch template.

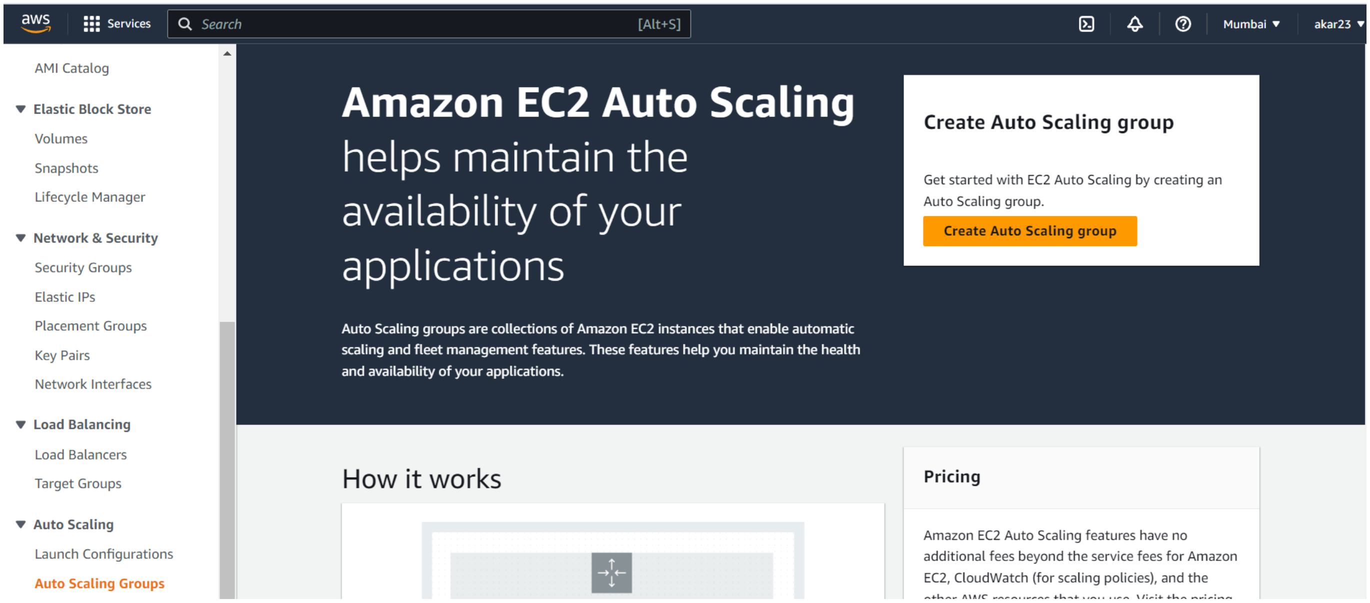
Step 10: New template is created.



Success

Successfully created AtreyeeTemplate (lt-033304c7608de5ab9)

Step 11: Again go to EC2 dashboard and click on Auto Scaling Groups. Then 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

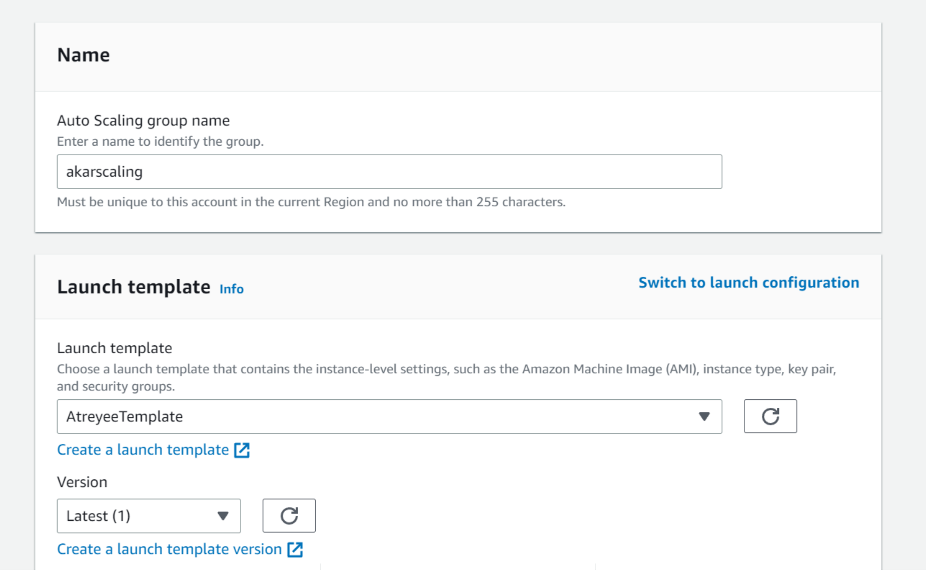
Get started with EC2 Auto Scaling by creating an Auto Scaling group.

How it works

Pricing

Amazon EC2 Auto Scaling features have no additional fees beyond the service fees for Amazon EC2, CloudWatch (for scaling policies), and the other AWS resources that you use. Visit the pricing

Step 12: Give a name. Select the template you just created and choose its latest version. Then click on Next.


Info
Switch to launch configuration
Create a launch template
Create a launch template version

Name

Auto Scaling group name

Enter a name to identify the group.

akarscaling

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

Launch template

Launch template

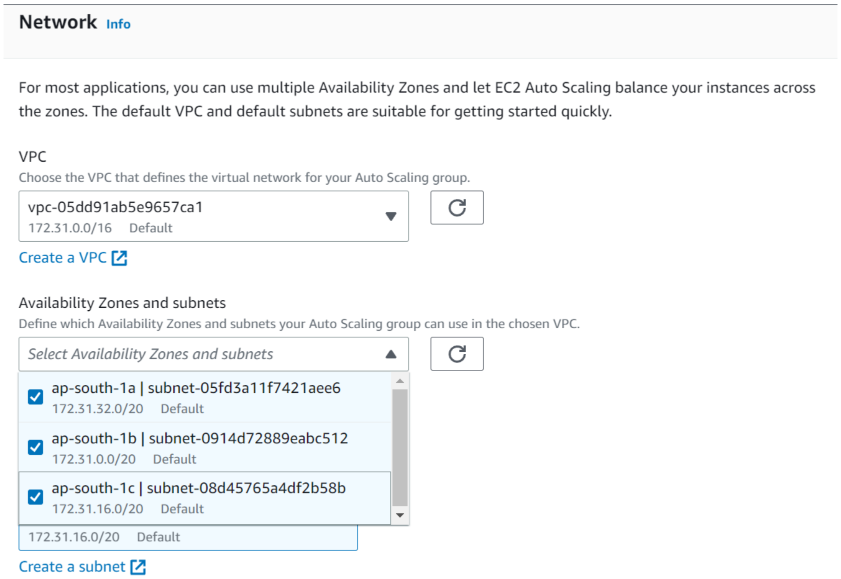
Choose a launch template that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.

AtreyeeTemplate ▼

Version

Latest (1) ▼

Step 13: Now go to Availability Zones and subnets and select all. Then click on Next.



Network Info

For most applications, you can use multiple Availability Zones and let EC2 Auto Scaling balance your instances across the zones. The default VPC and default subnets are suitable for getting started quickly.

VPC

Choose the VPC that defines the virtual network for your Auto Scaling group.

vpc-05dd91ab5e9657ca1
172.31.0.0/16 Default

Create a VPC

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

☑ ap-south-1a | subnet-05fd3a11f7421aee6
172.31.32.0/20 Default

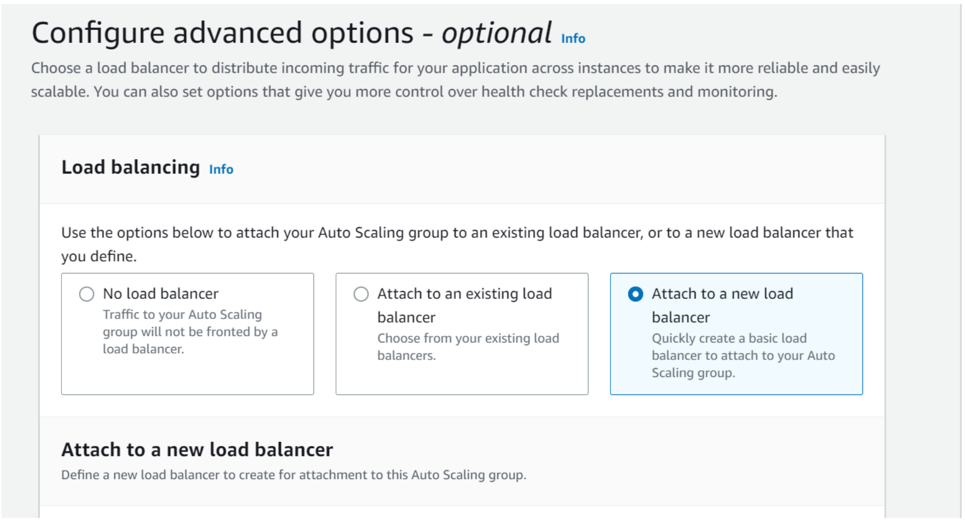
☑ ap-south-1b | subnet-0914d72889eabc512
172.31.0.0/20 Default

☑ ap-south-1c | subnet-08d45765a4df2b58b
172.31.16.0/20 Default

172.31.16.0/20 Default

Create a subnet

Step 14: Select Attach a new load balancer.



Configure advanced options - *optional* Info

Choose a load balancer to distribute incoming traffic for your application across instances to make it more reliable and easily scalable. 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.

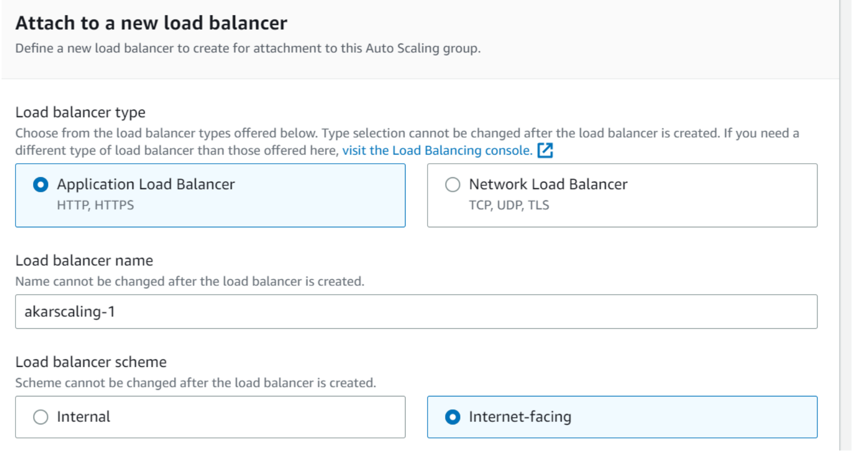
○ Attach to an existing load balancer
Choose from your existing load balancers.

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

Attach to a new load balancer

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

Step 15: In Attach a new load balancer, select balancer type as Application Load Balancer and Load Balancer scheme as Internet-facing.



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.

akarscaling-1

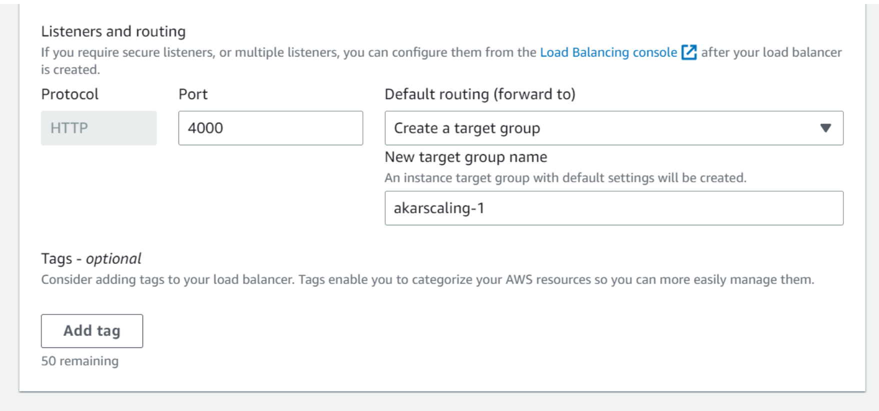
Load balancer scheme

Scheme cannot be changed after the load balancer is created.

○ Internal

● Internet-facing

Step 16: In Listeners and routing give the port number (For my project, it is 4000). Set the Default routing as "Create a target group". Automatically New target group name will appear. Then click on Next.



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: HTTP

Port: 4000

Default routing (forward to): Create a target group

New target group name

An instance target group with default settings will be created.

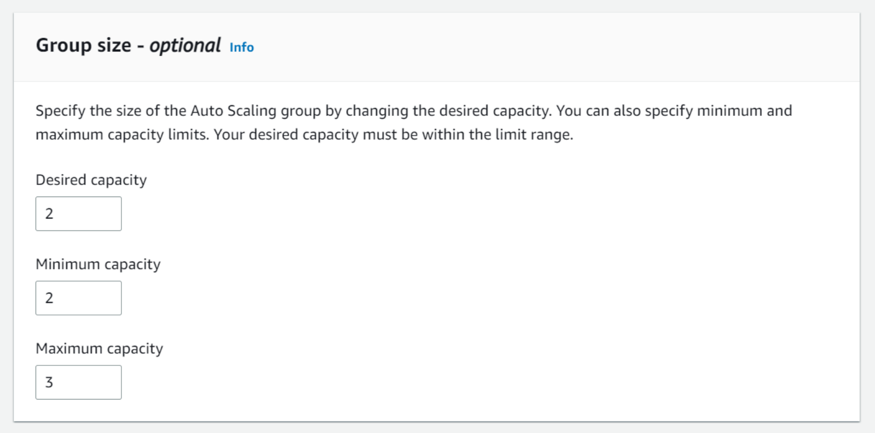
akarscaling-1

Tags - *optional*

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

50 remaining

Step 17: Set Group size.



Group size - *optional* Info

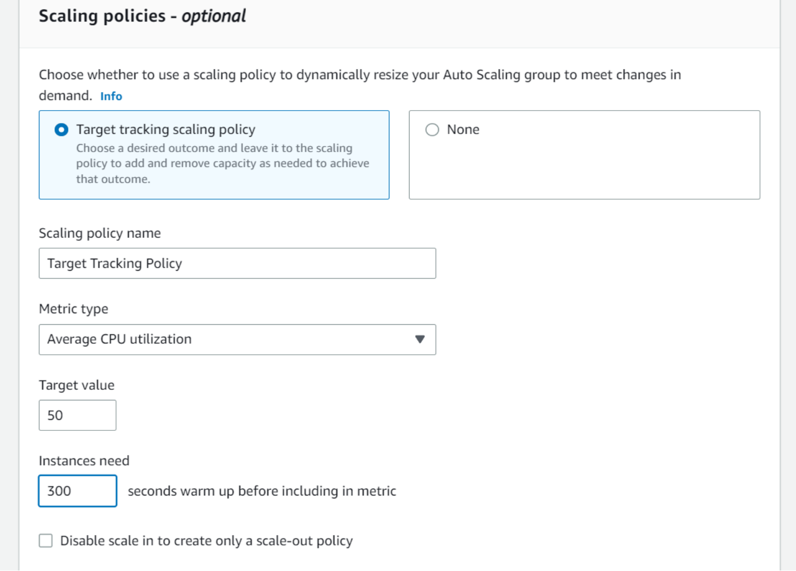
Specify the size of the Auto Scaling group by changing the desired capacity. You can also specify minimum and maximum capacity limits. Your desired capacity must be within the limit range.

Desired capacity: 2

Minimum capacity: 2

Maximum capacity: 3

Step 18: Set scaling policy as Target tracking scaling policy and provide the time needed for instances. Then click on Next.



Scaling policies - *optional*

Choose whether to use a scaling policy to dynamically resize your Auto Scaling group to meet changes in demand. Info

● Target tracking scaling policy
Choose a desired outcome and leave it to the scaling policy to add and remove capacity as needed to achieve that outcome.

○ None

Scaling policy name: Target Tracking Policy

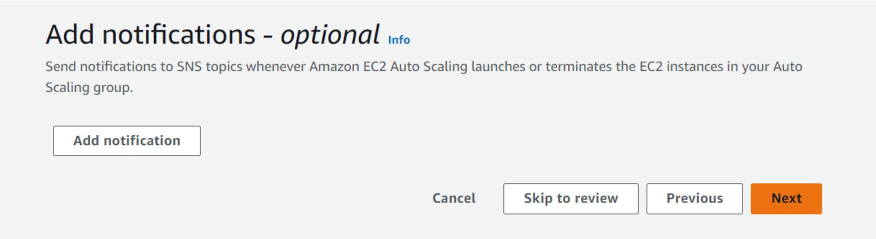
Metric type: Average CPU utilization

Target value: 50

Instances need 300 seconds warm up before including in metric

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

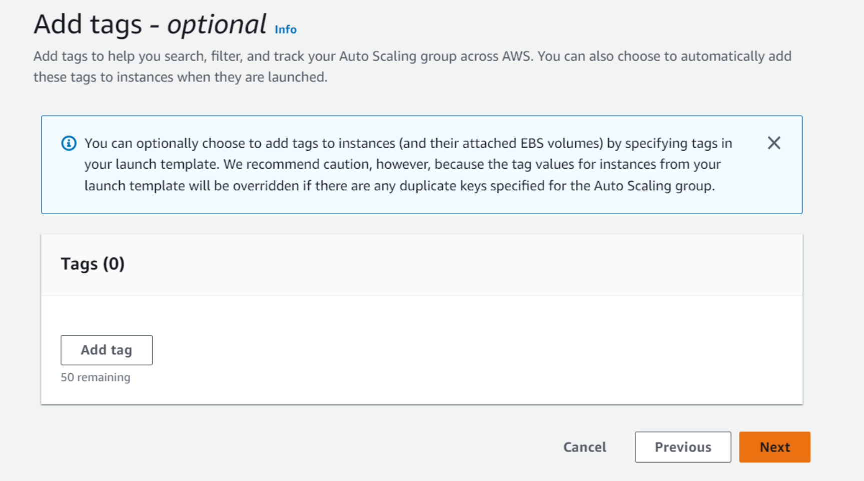
Step 19: Again click on Next.



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.

Step 20: Again click on Next.



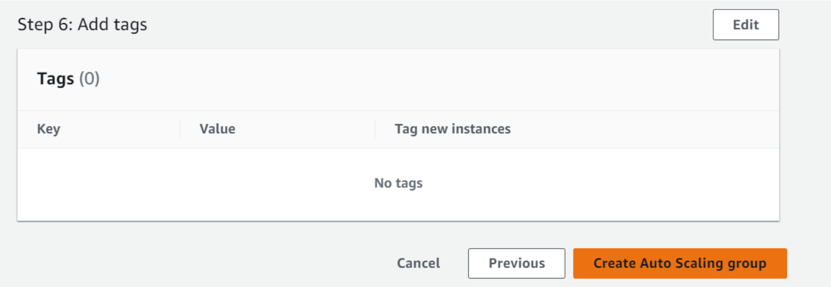
Add tags - *optional* Info

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)

Step 21: Review everything and click on “Create Auto Scaling group”.

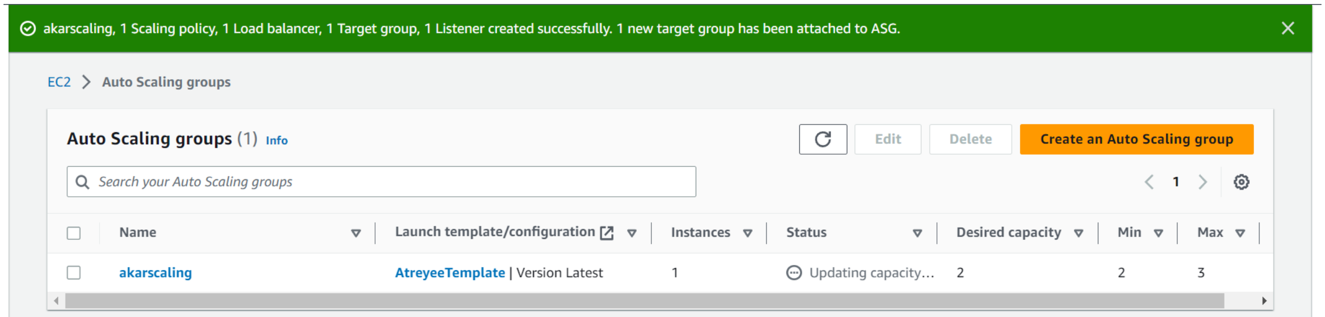


Step 6: Add tags

Tags (0)

Key	Value	Tag new instances
	No tags	

Step 22: Now Auto Scaling group is created.



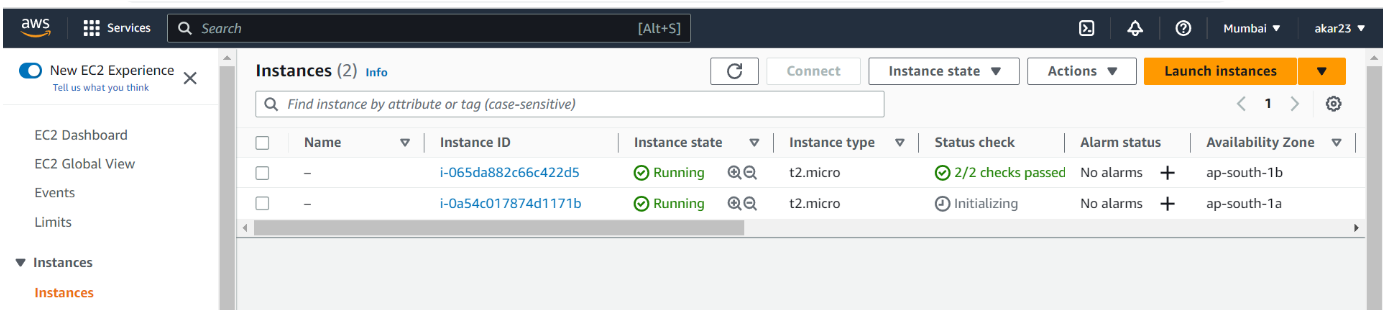
akarscaling, 1 Scaling policy, 1 Load balancer, 1 Target group, 1 Listener created successfully. 1 new target group has been attached to ASG.

EC2 > Auto Scaling groups

Auto Scaling groups (1) Info

□	Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max
□	akarscaling	AtreyeeTemplate | Version Latest	1	Updating capacity...	2	2	3

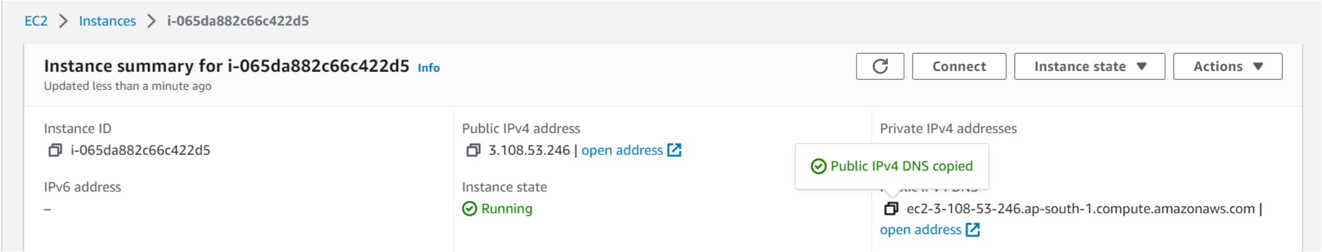
Step 23: Again go to EC2 dashboard. Then go to Instance. You can see that two instance is already created.



Instances (2) Info

□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
□	–	i-065da882c66c422d5	✓ Running	t2.micro	✓ 2/2 checks passed	No alarms +	ap-south-1b
□	–	i-0a54c017874d1171b	✓ Running	t2.micro	Initializing	No alarms +	ap-south-1a

Step 24: Open any of the instance and copy the Public IPv4 DNS.



Instance summary for i-065da882c66c422d5 Info
Updated less than a minute ago

Instance ID: i-065da882c66c422d5

Public IPv4 address: 3.108.53.246 | open address

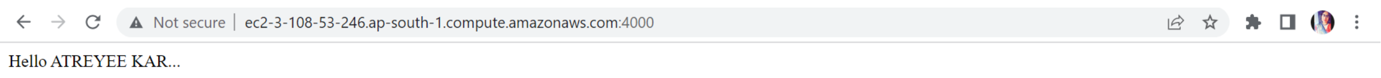
Private IPv4 addresses

IPv6 address: –

Instance state: ✓ Running

Public IPv4 DNS: ec2-3-108-53-246.ap-south-1.compute.amazonaws.com | open address

Step 25: Open it in a new browser. Give the port number (:4000) after the address. The content can be seen.

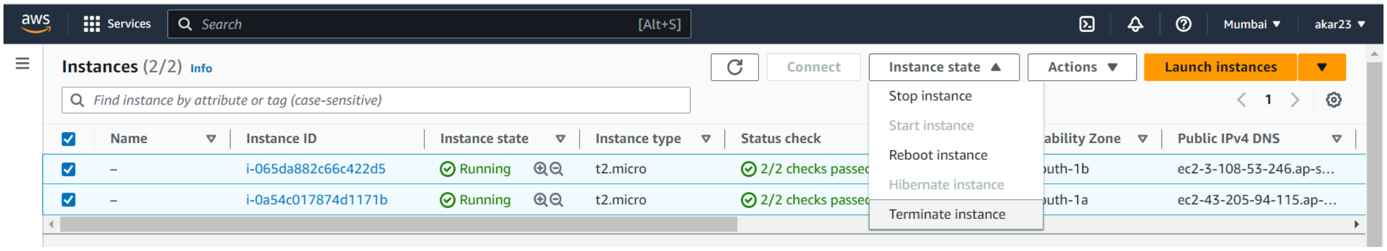

Not secure | ec2-3-108-53-246.ap-south-1.compute.amazonaws.com:4000

Hello ATREYEE KAR...

To test whether our Auto-Scaling Group actually works we need to crash or overload the existing instance servers.
Then only our Auto-Scaling Group will provide fresh instance servers automatically in case of crash or it can provide extra servers to handle overloads.

We will now **CRASH THE SERVER INSTANCES** manually by terminating them.

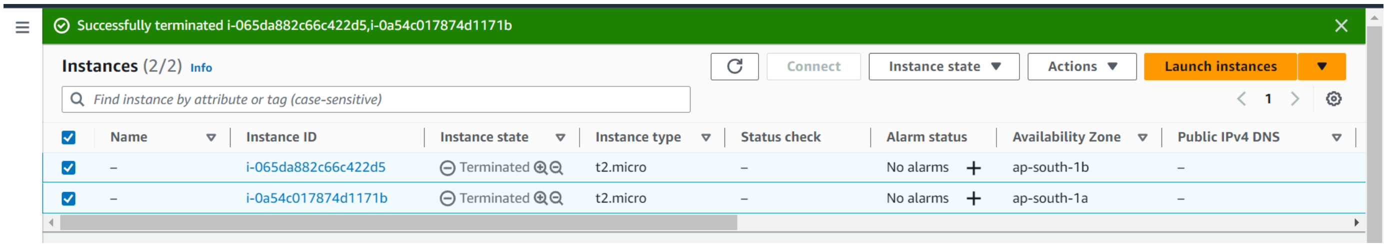
Step 1: Select both the instances. Go to Instance state and select Terminate instance.



Instances (2/2) Info

☑	Name	Instance ID	Instance state	Instance type	Status check	…ability Zone	Public IPv4 DNS
☑	–	i-065da882c66c422d5	✓ Running	t2.micro	✓ 2/2 checks passed	…uth-1b	ec2-3-108-53-246.ap-s…
☑	–	i-0a54c017874d1171b	✓ Running	t2.micro	✓ 2/2 checks passed	…uth-1a	ec2-43-205-94-115.ap-…

Step 2: Both the instances are terminated.

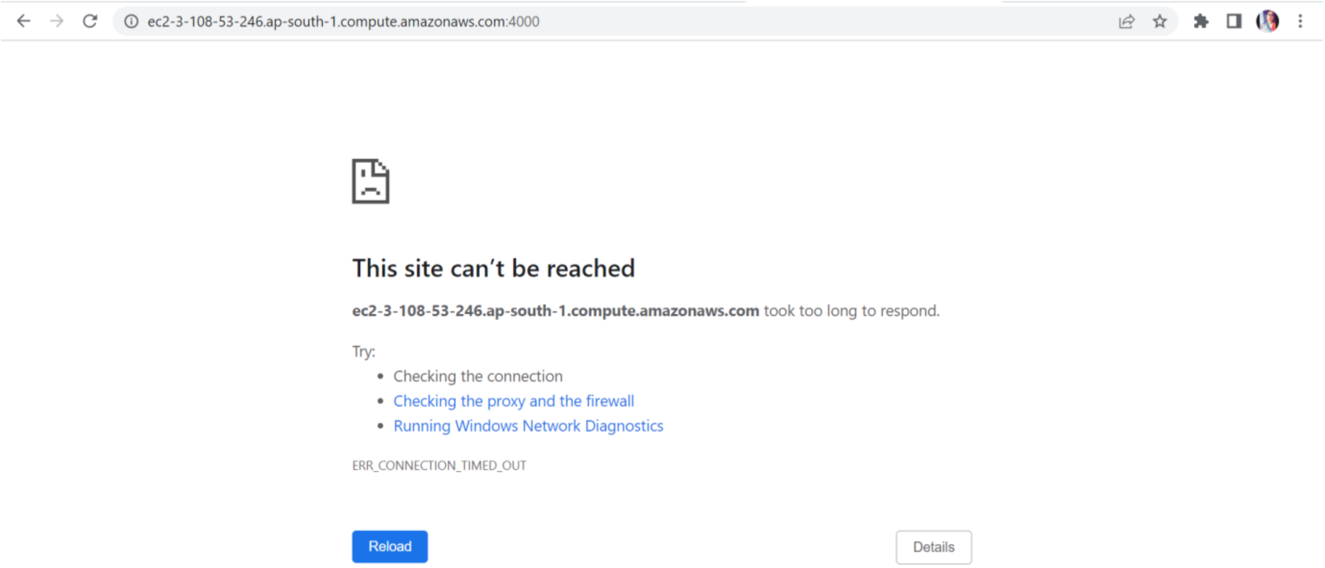


Successfully terminated i-065da882c66c422d5,i-0a54c017874d1171b

Instances (2/2) Info

☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
☑	–	i-065da882c66c422d5	Terminated	t2.micro	–	No alarms	ap-south-1b	–
☑	–	i-0a54c017874d1171b	Terminated	t2.micro	–	No alarms	ap-south-1a	–

Step 3: Now refresh the browser. Now, we cannot reach the site.



ec2-3-108-53-246.ap-south-1.compute.amazonaws.com:4000

This site can't be reached

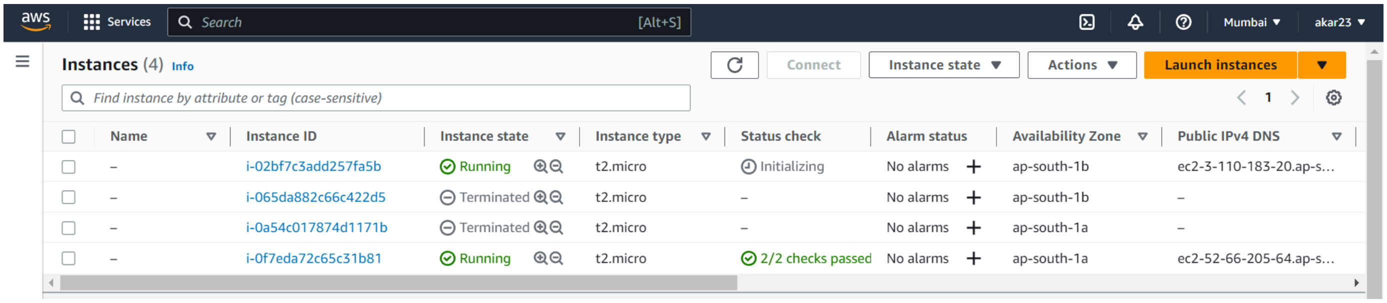
ec2-3-108-53-246.ap-south-1.compute.amazonaws.com took too long to respond.

Try:

- Checking the connection
- Checking the proxy and the firewall
- Running Windows Network Diagnostics

ERR_CONNECTION_TIMED_OUT

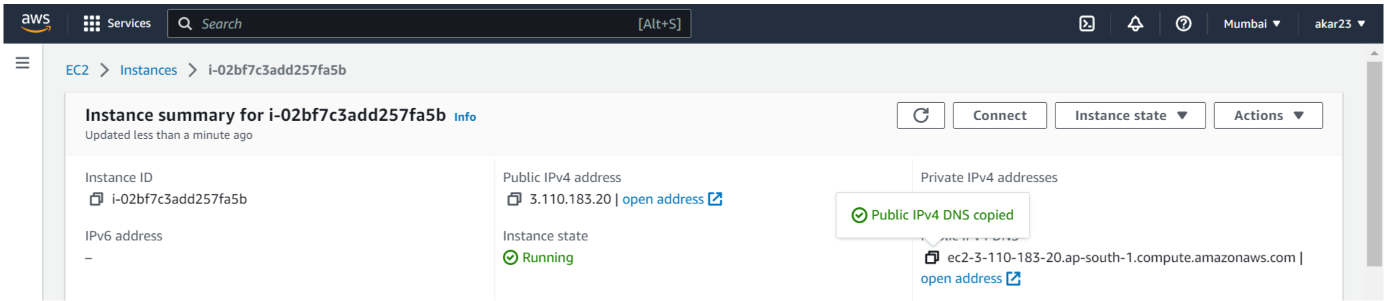
Step 4: Wait for few seconds. Then refresh the Instances page. We can see two new instances are automatically created.



Instances (4) Info

□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
□	–	i-02bf7c3add257fa5b	Running	t2.micro	Initializing	No alarms	ap-south-1b	ec2-3-110-183-20.ap-s...
□	–	i-065da882c66c422d5	Terminated	t2.micro	–	No alarms	ap-south-1b	–
□	–	i-0a54c017874d1171b	Terminated	t2.micro	–	No alarms	ap-south-1a	–
□	–	i-0f7eda72c65c31b81	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1a	ec2-52-66-205-64.ap-s...

Step 5: Now open any of the running instance. Copy the public IPv4 DNS address.



Instance summary for i-02bf7c3add257fa5b Info

Updated less than a minute ago

Instance ID: i-02bf7c3add257fa5b

Public IPv4 address: 3.110.183.20 | open address

Private IPv4 addresses

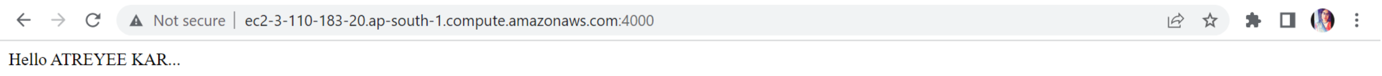
Public IPv4 DNS copied

IPv6 address: –

Instance state: Running

ec2-3-110-183-20.ap-south-1.compute.amazonaws.com | open address

Step 6: Open it in a new browser. Give the port number after the address using ":". Again we can see the content.



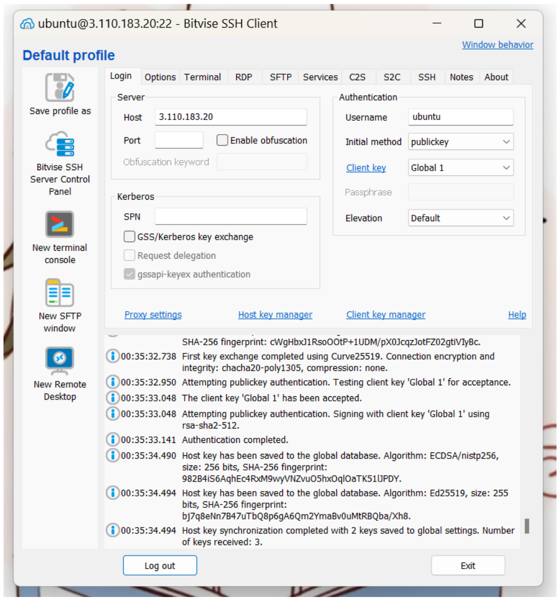
Not secure | ec2-3-110-183-20.ap-south-1.compute.amazonaws.com:4000

Hello ATREYEE KAR...

So, our Auto-Scaling Group can handle instance crashing by providing new fresh instances.

Now, we will **CRASH THE SERVER INSTANCES** by overloading them by running scripts.

Step 1: Now Log in to Bitvise SSH Client.



Server

Host: 3.110.183.20

Port:

□ Enable obfuscation

Obfuscation keyword:

Kerberos

SPN:

□ GSS/Kerberos key exchange

□ Request delegation

☑ gssapi-keyex authentication

Authentication

Username: ubuntu

Initial method: publickey

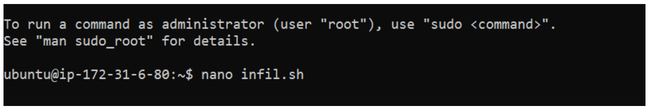
Client key: Global 1

Passphrase:

Elevation: Default

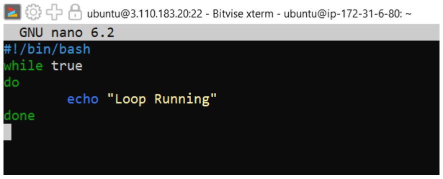
```
                SHA-256 fingerprint: cWgHbxJ1RsoOOtP+1UDM/pX0JcqzJotFZ02gtiVIyBc.
00:35:32.738  First key exchange completed using Curve25519. Connection encryption and
              integrity: chacha20-poly1305, compression: none.
00:35:32.950  Attempting publickey authentication. Testing client key 'Global 1' for acceptance.
00:35:33.048  The client key 'Global 1' has been accepted.
00:35:33.048  Attempting publickey authentication. Signing with client key 'Global 1' using
              rsa-sha2-512.
00:35:33.141  Authentication completed.
00:35:34.490  Host key has been saved to the global database. Algorithm: ECDSA/nistp256,
              size: 256 bits, SHA-256 fingerprint:
              982B4iS6AqhEc4RxM9wyVNZvuO5hxOqlOaTK51lJPDY.
00:35:34.494  Host key has been saved to the global database. Algorithm: Ed25519, size: 255
              bits, SHA-256 fingerprint:
              bj7q8eNn7B47uTbQ8p6gA6Qm2YmaBv0uMtRBQba/Xh8.
00:35:34.494  Host key synchronization completed with 2 keys saved to global settings. Number
              of keys received: 3.
```

Step 2: Now go to New Terminal Console and enter the command : **nano infil.sh**



```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-6-80:~$ nano infil.sh
```

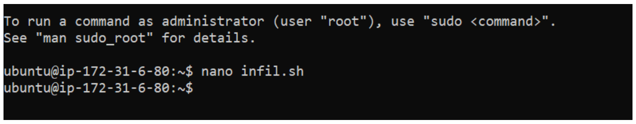
Step 3: A nano Editor will open. Wrile the following commands in it:



ubuntu@3.110.183.20:22 - Bitvise xterm - ubuntu@ip-172-31-6-80: ~

```
  GNU nano 6.2
#!/bin/bash
while true
do
        echo "Loop Running"
done
```

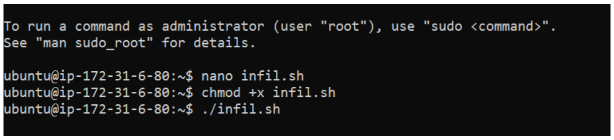
Step 4: Now, to save and close the shell script we need to press the following shortcuts and keys sequentially: **Ctrl X → Y → Enter.** Then you will return to the terminal.



```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-6-80:~$ nano infil.sh
ubuntu@ip-172-31-6-80:~$
```

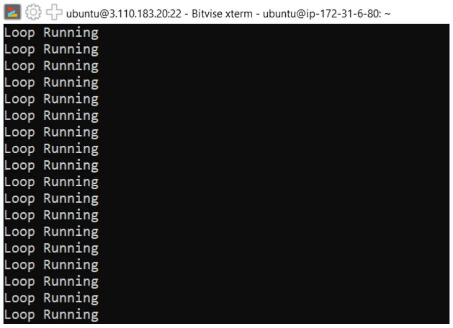
Step 5: Give execute permission to the .sh file. Then execute it.

- **chmod +x infil.sh –** To give execute permission to the file.
- **./infil.sh –** To execute the file.



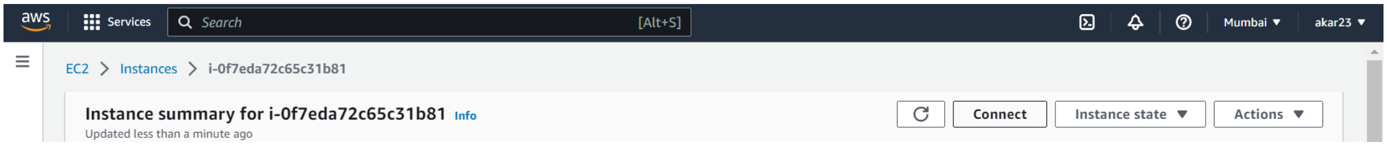
```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-6-80:~$ nano infil.sh
ubuntu@ip-172-31-6-80:~$ chmod +x infil.sh
ubuntu@ip-172-31-6-80:~$ ./infil.sh
```

Step 6: The file is executing. Therefore, the first instance is running infinitely. Keep it as it is.

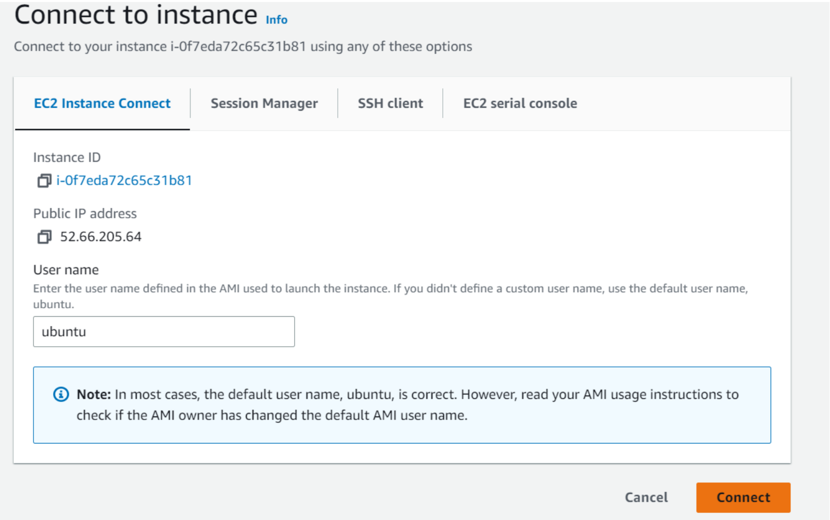

ubuntu@3.110.183.20:22 - Bitvise xterm - ubuntu@ip-172-31-6-80: ~

```
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
```

Step 7: Now go to the other running instance. And click on Connect. Then again click on Connect.



Instance summary for i-0f7eda72c65c31b81
Updated less than a minute ago



Connect to instance Info

Connect to your instance i-0f7eda72c65c31b81 using any of these options

Instance ID

i-0f7eda72c65c31b81

Public IP address

52.66.205.64

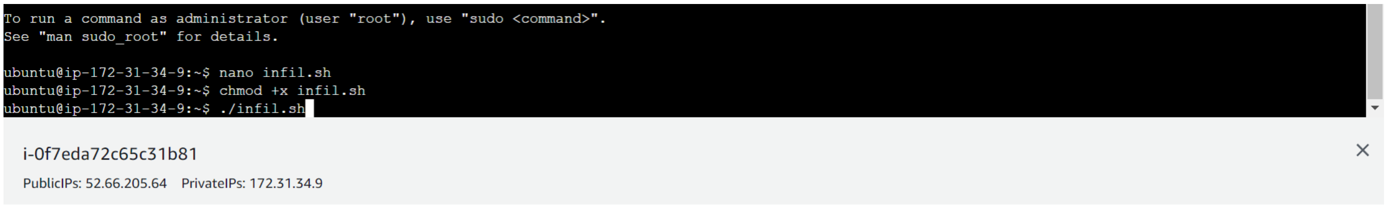
User name

Enter the user name defined in the AMI used to launch the instance. If you didn't define a custom user name, use the default user name, ubuntu.

ubuntu

Note: In most cases, the default user name, ubuntu, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

Step 8: A connect terminal will open. Repeat Step-2 to Step-5 here.

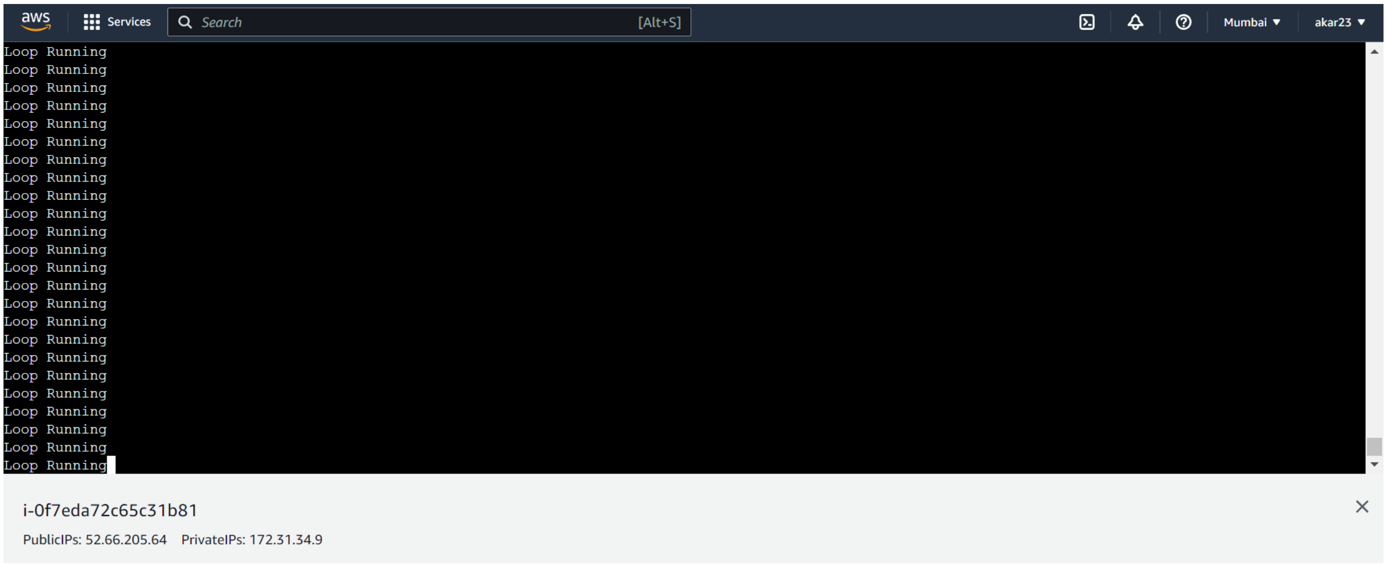


```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-34-9:~$ nano infil.sh
ubuntu@ip-172-31-34-9:~$ chmod +x infil.sh
ubuntu@ip-172-31-34-9:~$ ./infil.sh
```

i-0f7eda72c65c31b81

PublicIPs: 52.66.205.64 PrivateIPs: 172.31.34.9

Step 9: Now the second instance is also running infinitely. Keep it as it is.

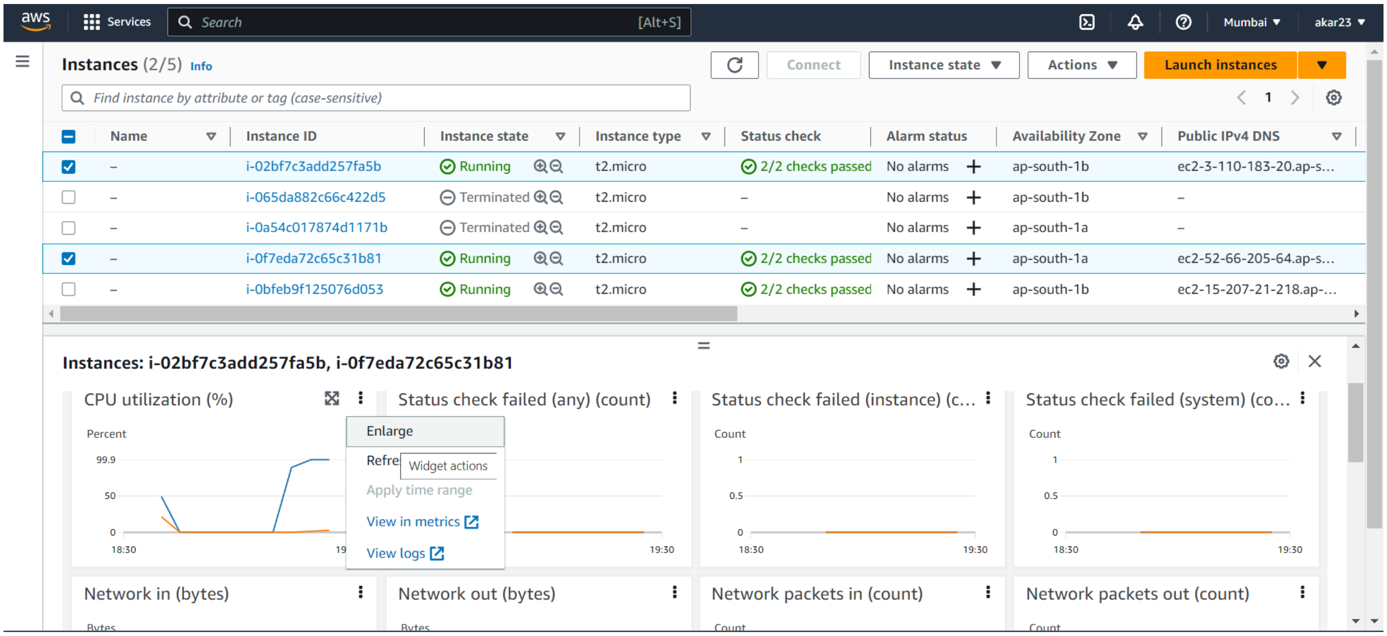


```
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
Loop Running
```

i-0f7eda72c65c31b81

PublicIPs: 52.66.205.64 PrivateIPs: 172.31.34.9

Step 10: Now go to Instance page. Select both the instances. Then scroll down in the "Monitoring" section. Go to CPU Utilization. Click on the three dots. Click on Enlarge.



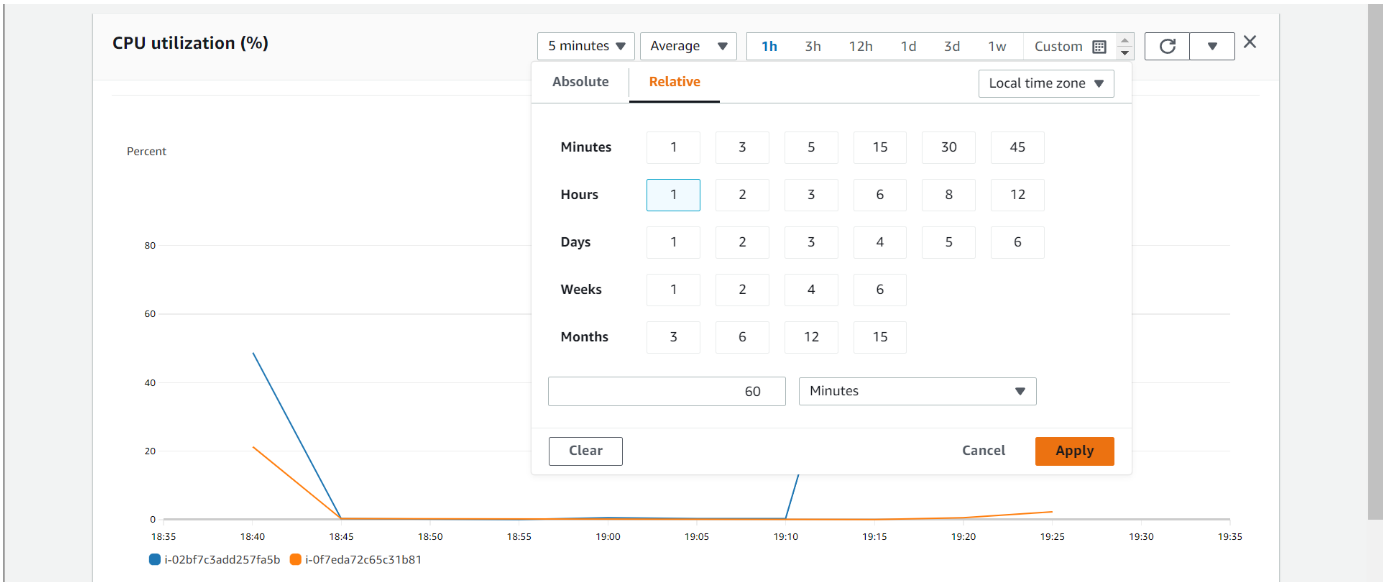
Instances (2/5) Info

Find instance by attribute or tag (case-sensitive)

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
☑	–	i-02bf7c3add257fa5b	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1b	ec2-3-110-183-20.ap-s...
□	–	i-065da882c66c422d5	Terminated	t2.micro	–	No alarms	ap-south-1b	–
□	–	i-0a54c017874d1171b	Terminated	t2.micro	–	No alarms	ap-south-1a	–
☑	–	i-0f7eda72c65c31b81	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1a	ec2-52-66-205-64.ap-s...
□	–	i-0bfeb9f125076d053	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1b	ec2-15-207-21-218.ap-...

Instances: i-02bf7c3add257fa5b, i-0f7eda72c65c31b81

Step 11: Go to Custom. Select Local time zone. Then click on Apply.



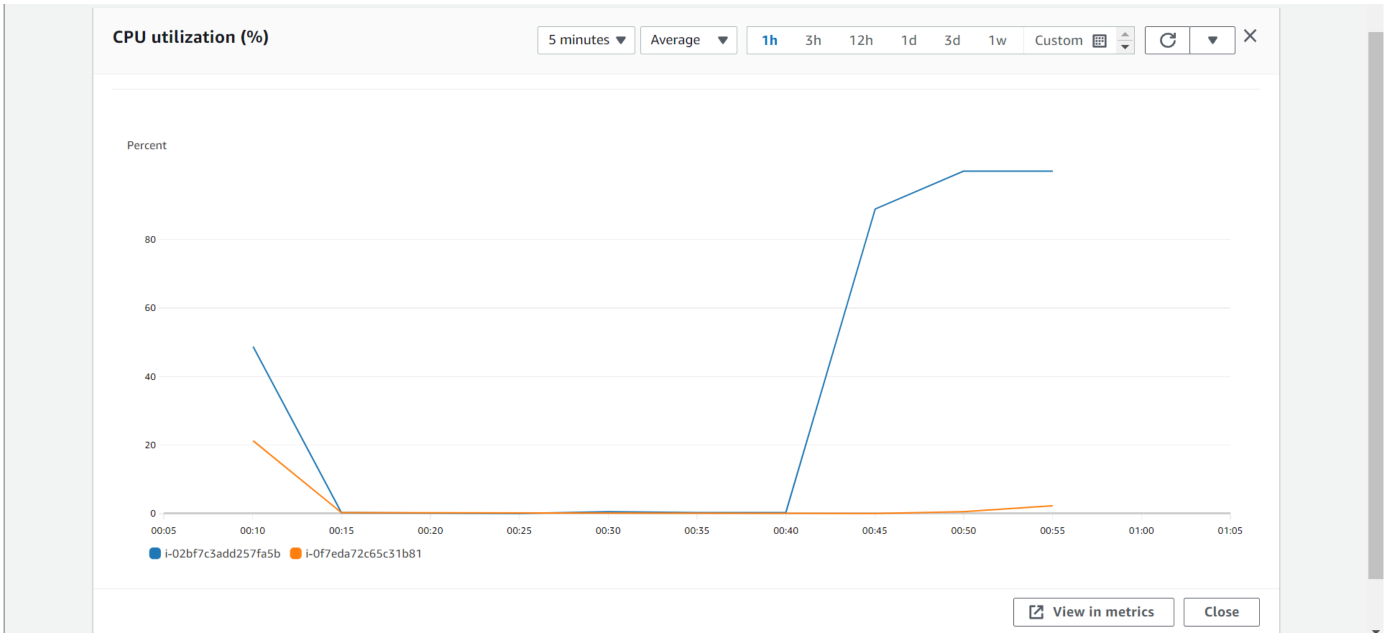
CPU utilization (%)

5 minutes · Average · 1h 3h 12h 1d 3d 1w Custom

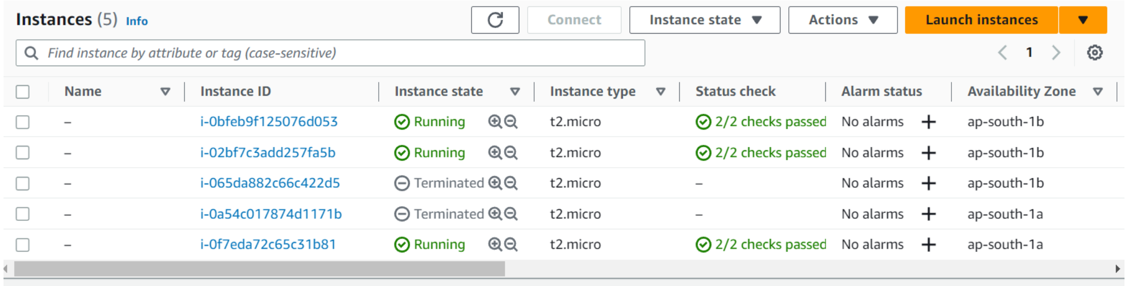
Absolute · Relative · Local time zone

60 Minutes

Step 12: We can see our first instance has already reached 50% utilization.



So, our Auto-scaling group has created another instance to compensate for the overload.



□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
□	–	i-0bfeb9f125076d053	Running	t2.micro	2/2 checks passed	No alarms +	ap-south-1b
□	–	i-02bf7c3add257fa5b	Running	t2.micro	2/2 checks passed	No alarms +	ap-south-1b
□	–	i-065da882c66c422d5	Terminated	t2.micro	–	No alarms +	ap-south-1b
□	–	i-0a54c017874d1171b	Terminated	t2.micro	–	No alarms +	ap-south-1a
□	–	i-0f7eda72c65c31b81	Running	t2.micro	2/2 checks passed	No alarms +	ap-south-1a

Our webpage was not at all disconnected in this whole process. You can check it by refresing the webpage.

If we want to delete the instances permanently, we have to follow the sequence given below. Otherwise, the loop will continue and the instances will be created automatically.

Deleting Sequence:

- Delete Auto-Scaling groups
- Delete Load Balancers
- Delete Target groups
- Delete EC2 Instances

Thus, we have built Scaling plans in AWS that balance load on different EC2 instances.