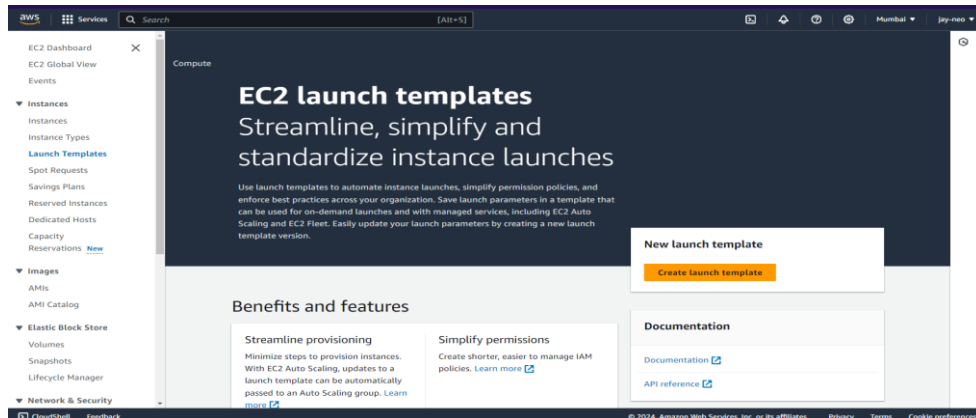


# Assignment: 11

**Problem Statement:** Build scaling plans in AWS that balance the load on different EC2 instance.

## Procedure:

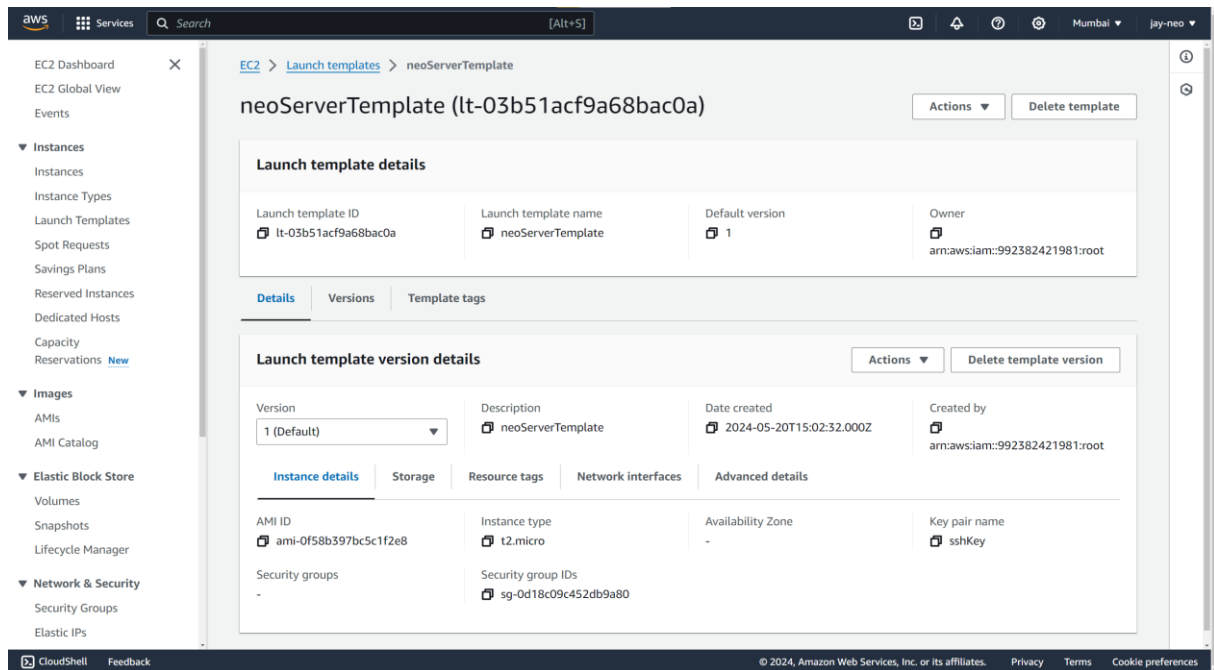
**Step 1:** Go to the Launch Template under the section on EC2 Dashboard, then click **Create launch template**



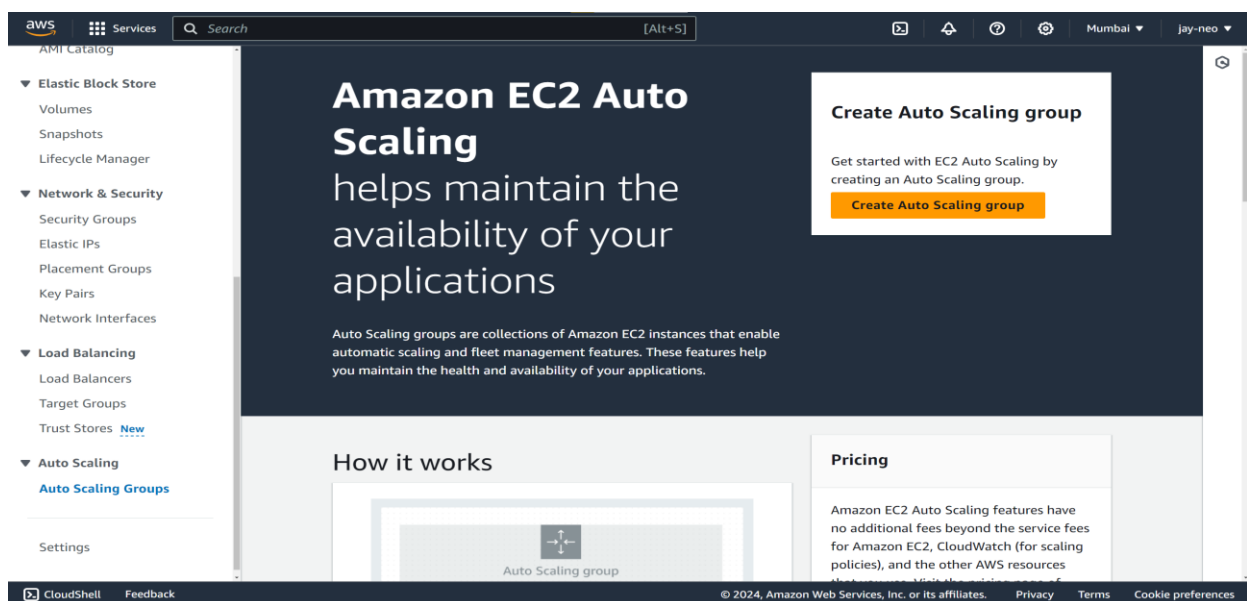
**Step 2:** Fill the required boxes following below,

Then go to the Editor section exits in Advanced details and paste the following bash code, next click **Create launch instance**.

```
#!/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 http://github.com/jay-neo/IT-Workshop-AWS-10.git
cd IT-Workshop-AWS-10
npm install
node index.js
```



**Step 3:** Go to the **Auto Scaling Groups** under the section of EC2 Dashboard and click **Create Auto Scaling group** button.



**Step 4:** Write After clicking the **Auto Scaling group** button fille all the necessary details following bellow,

**Step 4.1:** Give a name of Auto scaling group and select the Launch template that created at Step-2.

aws

Services

Search

[Alt+S]

Mumbai

jay-neo

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1  
Choose launch template

Step 2  
Choose instance launch options

Step 3 - optional  
Configure advanced options

Step 4 - optional  
Configure group size and scaling

Step 5 - optional  
Add notifications

Step 6 - optional  
Add tags

Step 7  
Review

## 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.

neoScaling

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.

neoServerTemplate

Create a launch template

Version

Default (1)

Create a launch template version

Description	Launch template	Instance type
neoServerTemplate	neoServerTemplate	t2.micro
	lt-03b51acf9a68bac0a	
AMI ID	Security groups	Request Spot Instances
ami-0f58b397bc5c1f2e8	-	No
Key pair name	Security group IDs	
sshKey	sg-0d18c09c452db9a80	

Additional details

Storage (volumes)	Date created
-	Mon May 20 2024 20:32:32 GMT+0530 (India Standard Time)

Cancel

Next

CloudShell

Feedback

© 2024, Amazon Web Services, Inc. or its affiliates.

Privacy

Terms

Cookie preferences

Step 4.2: Select the default VPC and choose all the Availability Zones and subnets.

aws

Services

Search

[Alt+S]

Mumbai

jay-neo

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1  
Choose launch template

Step 2  
Choose instance launch options

Step 3 - optional  
Configure advanced options

Step 4 - optional  
Configure group size and scaling

Step 5 - optional  
Add notifications

Step 6 - optional  
Add tags

Step 7  
Review

## Choose instance launch options Info

Choose the VPC network environment that your instances are launched into, and customize the instance types and purchase options.

**Instance type requirements** Info

Override launch template

You can keep the same instance attributes or instance type from your launch template, or you can choose to override the launch template by specifying different instance attributes or manually adding instance types.

Launch template	Version	Description
neoServerTemplate	Default	neoServerTemplate
lt-03b51acf9a68bac0a		
Instance type		
t2.micro		

**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-003907d798e946506

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-04a981ab6ceb02d6c

172.31.32.0/20

Default

ap-south-1b | subnet-03163847d17a5cf17

172.31.0.0/20

Default

ap-south-1c | subnet-0c5ed6afd309f5f98

172.31.16.0/20

Default

Create a subnet

Your requested instance type (t2.micro) is not available in 1 Availability Zone. You may need to change the instance type or choose other Availability Zones for better resiliency. [Learn more](#)

Cancel

Skip to review

Previous

Next

CloudShell

Feedback

© 2024, Amazon Web Services, Inc. or its affiliates.

Privacy

Terms

Cookie preferences

Step 4.3: Fill the configure advanced options following bellow,

**Load balancing:** Attach to a new load, **Load balancer type:** Application Load Balancer, **Load balancer scheme:** Internet-facing, **Protocol:** HTTP 4000, **Target:** neo-scale-1 | HTTP  
Check the **Turn on Elastic Load Balancing health checks**, **Health check grace period:** 240

aws

Services

Search

[Alt+S]

Mumbai

jay-neo

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1  
Choose launch template

Step 2  
Choose instance launch options

Step 3 - optional  
Configure advanced options

Step 4 - optional  
Configure group size and scaling

Step 5 - optional  
Add notifications

Step 6 - optional  
Add tags

Step 7  
Review

### 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.

#### 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.

neoScaling-1

##### Load balancer scheme

Scheme cannot be changed after the load balancer is created.

☐ Internal

☒ Internet-facing

##### Network mapping

Your new load balancer will be created using the same VPC and Availability Zone selections as your Auto Scaling group. You can select different subnets and add subnets from additional Availability Zones.

VPC  
vpc-003907d798e946506 [info](#)

##### Availability Zones and subnets

You must select a single subnet for each Availability Zone enabled. Only public subnets are available for selection to support DNS resolution.

☒ ap-south-1a

subnet-04a981ab6ceb02d6c

☒ ap-south-1b

subnet-03163847d17a5cf17

☒ ap-south-1c

subnet-0c5ed6afd309f5f98

##### 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)  
neo-scale-1 | HTTP

##### 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

#### VPC Lattice integration options [info](#)

To improve networking capabilities and scalability, integrate your Auto Scaling group with VPC Lattice. VPC Lattice facilitates communications between AWS services and helps you connect and manage your applications across compute services in AWS.

##### Select VPC Lattice service to attach

☒ No VPC Lattice service  
VPC Lattice will not manage your Auto Scaling group's network access and connectivity with other services.

☐ Attach to VPC Lattice service  
Incoming requests associated with specified VPC Lattice target groups will be routed to your Auto Scaling group.

Create new VPC Lattice service [info](#)

#### 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

#### Additional settings

##### Monitoring [info](#)

☐ Enable group metrics collection within CloudWatch

##### Default instance warmup [info](#)

The amount of time that CloudWatch metrics for new instances do not contribute to the group's aggregated instance metrics, as their usage data is not reliable yet.

☐ Enable default instance warmup

Cancel

Skip to review

Previous

Next

CloudShell

Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

**Step 4.3:** In Configure group size and scaling, **Desired capacity: 2, Min desired capacity: 2, Max desired capacity: 3, select Target tracking scaling policy, Instance warmup: 240**

The screenshot shows the 'Configure group size and scaling' step in the AWS Management Console. The left sidebar lists steps from 1 to 7, with Step 4 selected. The main content area is titled 'Configure group size and scaling - optional' and includes the following sections:

- Group size:** Set the initial size of the Auto Scaling group. The 'Desired capacity type' is set to 'Units (number of instances)'. The 'Desired capacity' is set to 2.
- Scaling:** 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' is 2 and 'Max desired capacity' is 3.
  - Automatic scaling - optional:** Choose whether to use a target tracking policy. The 'Target tracking scaling policy' is selected. The 'Scaling policy name' is 'Target Tracking Policy'. The 'Metric type' is 'Average CPU utilization' and the 'Target value' is 50. The 'Instance warmup' is set to 240 seconds.
- Instance maintenance policy:** Control your Auto Scaling group's availability during instance replacement events. Choose a replacement behavior depending on your availability requirements:
  - Mixed behavior:** No policy (selected).
  - Prioritize availability:** Launch before terminating.
  - Control costs:** Terminate and launch.
  - Flexible:** Custom behavior.
- Instance scale-in protection:** Scale-in protection prevents newly launched instances from being terminated by scaling activities. The 'Enable instance scale-in protection' checkbox is unchecked.

At the bottom, there are buttons for 'Cancel', 'Skip to review', 'Previous', and 'Next'.

**Step 4.4:** In Add Notifications no change just click Next button,

The screenshot shows the 'Add notifications' step in the AWS Management Console. The left sidebar lists steps from 1 to 7, with Step 5 selected. The main content area is titled 'Add notifications - optional' and includes the following sections:

- Add notification:** Send notifications to SNS topics whenever Amazon EC2 Auto Scaling launches or terminates the EC2 instances in your Auto Scaling group. There is an 'Add notification' button.

At the bottom, there are buttons for 'Cancel', 'Skip to review', 'Previous', and 'Next'.

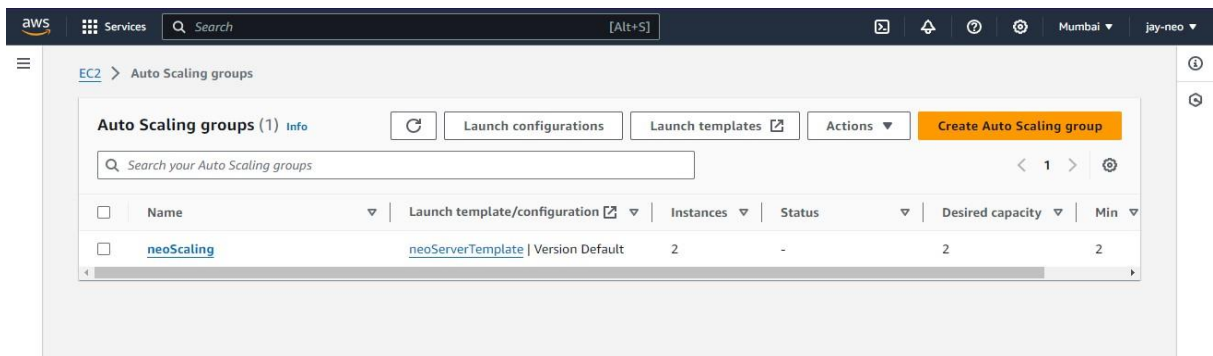


Step 4.5: In Add tags section no change just click Next button,

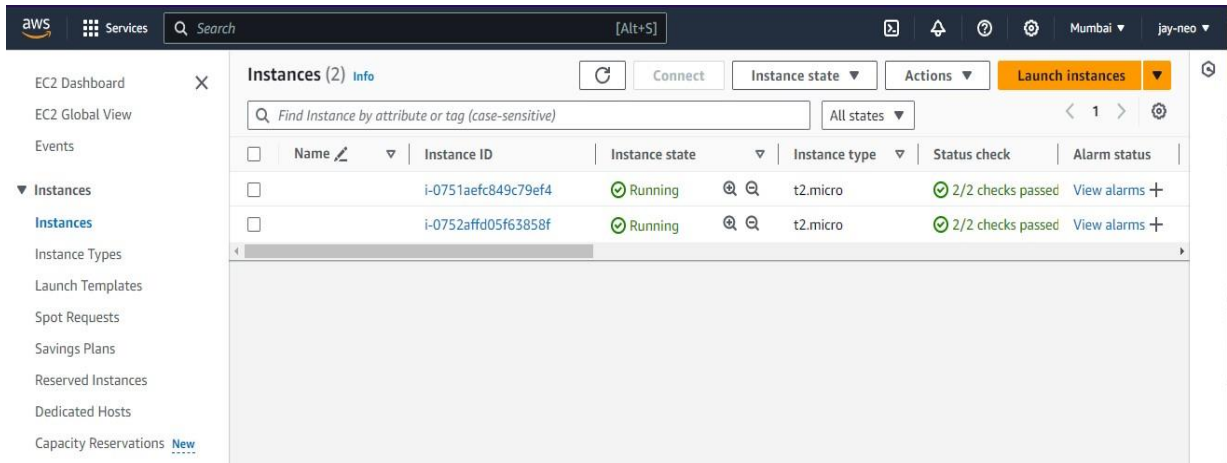
The screenshot shows the 'Add tags - optional' step in the AWS Management Console. The left sidebar contains a navigation menu with steps 1 through 7. The main content area has a title 'Add tags - optional' with an 'Info' icon. Below the title is a paragraph explaining that tags can be used to search, filter, and track the Auto Scaling group. A blue information box contains a note about overriding duplicate keys. Below this is a section titled 'Tags (0)' with an 'Add tag' button and a note '50 remaining'. At the bottom right are 'Cancel', 'Previous', and 'Next' buttons.

Step 4.5: Finally Review the Auto Scaling Group and Create Auto Scaling group

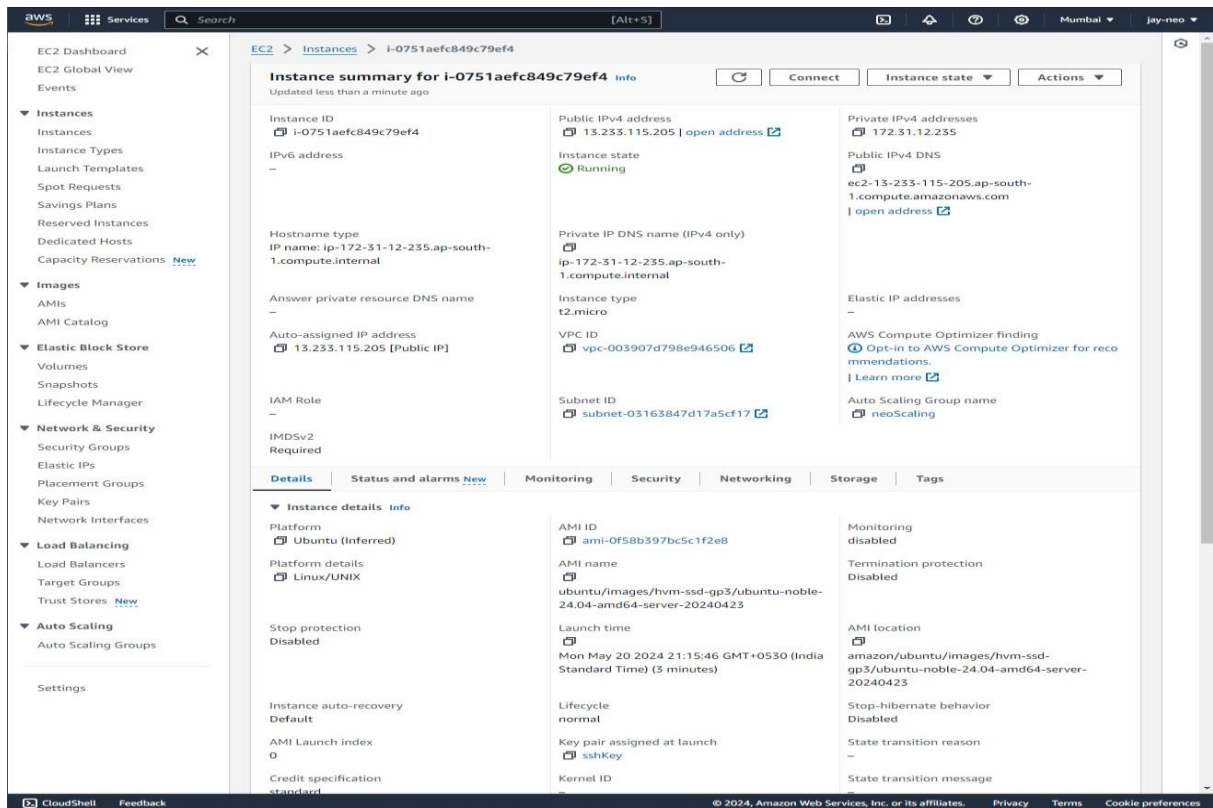
The screenshot shows the 'Review' step in the AWS Management Console. The left sidebar contains a navigation menu with steps 1 through 7. The main content area has a title 'Review' with an 'Info' icon. Below the title is a section titled 'Step 1: Choose launch template' with an 'Edit' button. This section contains 'Group details' (Auto Scaling group name: neoScaling) and 'Launch template' (Launch template: neoServerTemplate, Version: Default, Description: neoServer Template). Below this is 'Step 2: Choose instance launch options' with an 'Edit' button. This section contains 'Network' (VPC: vpc-d03907d798be946506, Availability Zone: ap-south-1a, Subnet: subnet-04a981ab6ceb02d6c, IP: 172.31.32.0/20) and 'Instance type requirements' (This Auto Scaling group will adhere to the launch template.). Below this is 'Step 3: Configure advanced options' with an 'Edit' button. This section contains 'Load balancing' (Load balancer 1: neoScaling-1, Type: Application/HTTP, Target group: neo-scale-1), 'VPC Lattice integration options' (VPC Lattice target groups: -), 'Health checks' (Health check type: EC2, ELB, Health check grace period: 240 seconds), and 'Additional settings' (Monitoring: Disabled, Default instance warmup: Disabled). Below this is 'Step 4: Configure group size and scaling policies' with an 'Edit' button. This section contains 'Group size' (Desired capacity: 2, Desired capacity type: Units (number of instances)) and 'Scaling' (Minimum desired capacity: 2, Maximum desired capacity: 8, Target tracking policy: Target Tracking Policy, Policy type: Target tracking scaling, Take the action: Add or remove capacity units as required, Instances need: 240 seconds to warm up before including in metric, Execute policy when: As required to maintain Average CPU utilization at 50, Scale in: Enabled). Below this is 'Instance maintenance policy' (Replacement behavior: No policy, Min healthy percentage: -, Max healthy percentage: -) and 'Instance scale-in protection' (Instance scale-in protection: -). Below this is 'Step 5: Add notifications' with an 'Edit' button. This section contains 'Notifications' (No notifications). Below this is 'Step 6: Add tags' with an 'Edit' button. This section contains 'Tags (0)' (Key, Value, Tag new instances) and 'No tags'. At the bottom right are 'Cancel', 'Previous', and 'Create Auto Scaling group' buttons.



**Step 5:** After creating Auto Scaling group namely neoScaling, go to the Instance page under EC2 dashboard



**Step 6:** Copy public IPv4 of one instance from the any two and connect it using ssh command from terminal



```
>> ssh ssh -i ~/Downloads/sshKey.pem ubuntu@13.233.115.205
```

Step 7: Next, create a Bash script to implement an infinite loop program that monitors the server CPU utilization. When it reaches its limit, the auto scaling policy should create another instance to balance the load of the auto-scaling group.

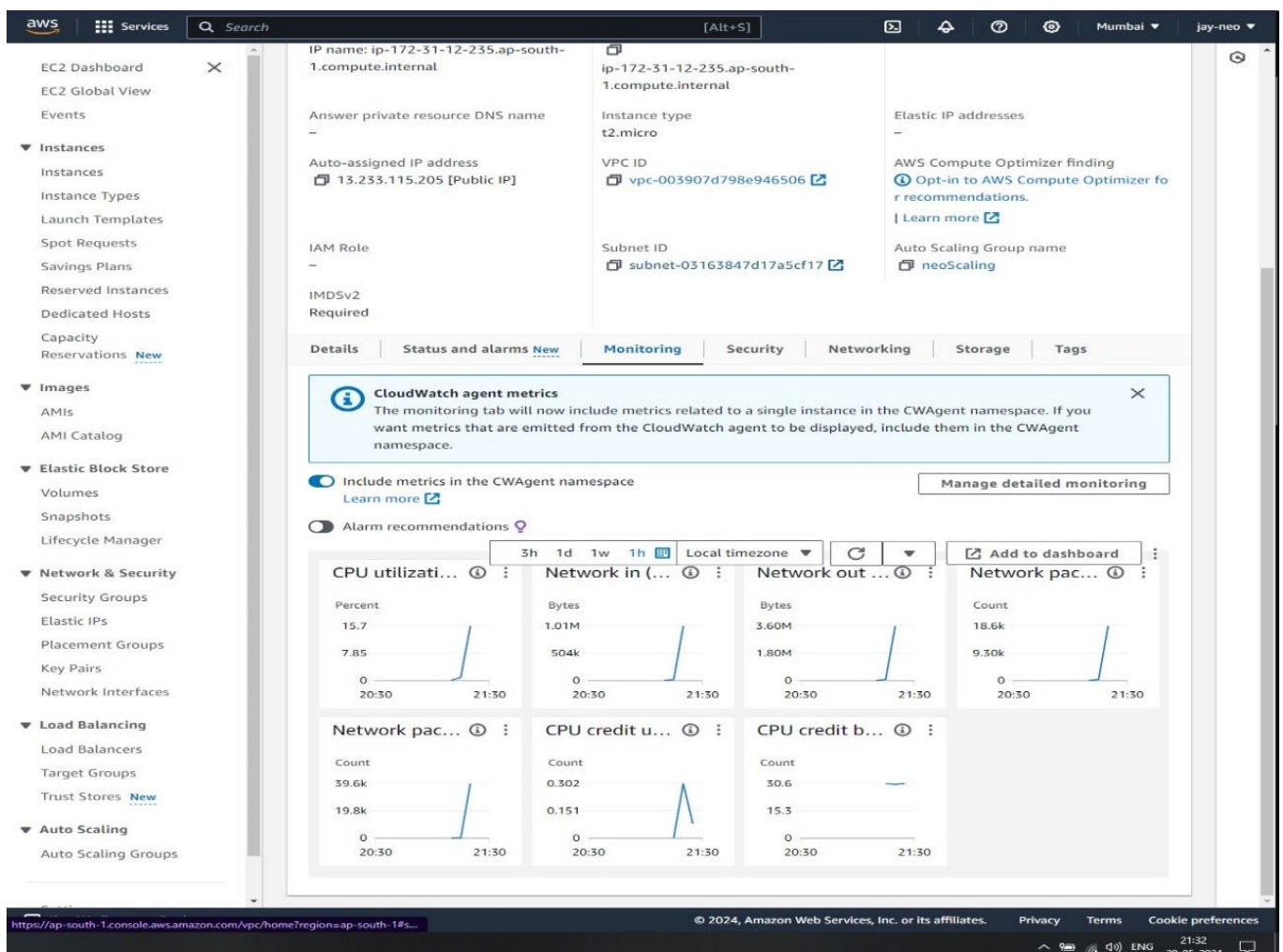
The bash script code in the **infy.sh** file is:

```
#!/bin/bash
while(true)
do
    echo "Inside loop ..."
done
```

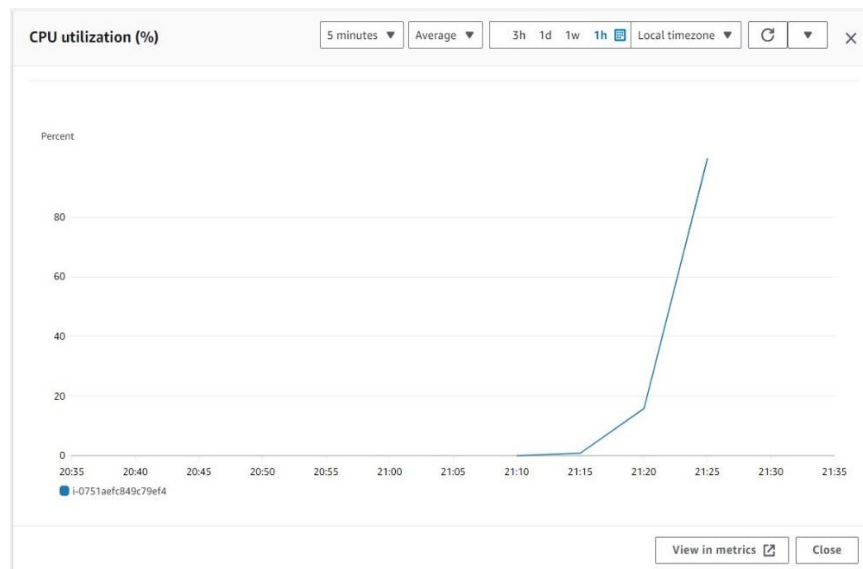
and the following commands run in the terminal is

```
ubuntu@ip-172-31-12-235: ~
ubuntu@ip-172-31-12-235:~$ sudo nano infy.sh
ubuntu@ip-172-31-12-235:~$ sudo chmod +x infy.sh
ubuntu@ip-172-31-12-235:~$ sh infy.sh
Inside loop ...
Inside loop ...
Inside loop ...
Inside loop ...
Inside loop ...
Inside loop ...
Inside loop ...
Inside loop ...
Inside loop ...
Inside loop ...
Inside loop ...
```

Step 8: Then monitor the CPU utilization graph of the selected instance where the code is running







The screenshot shows the AWS Management Console interface. On the left is a navigation menu with categories like EC2 Dashboard, Instances, Images, Elastic Block Store, Network & Security, Load Balancing, and Auto Scaling. The main content area is titled "Instances (3)" and shows a table of three running EC2 instances. Below the table is a "Select an instance" dialog box. The bottom of the screen shows a footer with copyright information, links to Privacy, Terms, and Cookie preferences, and system status information.

Name	Instance ID	Instance state	Instance type	Status check
	i-0751aefc849c79ef4	Running	t2.micro	2/2 checks passed
	i-0752affd05f63858f	Running	t2.micro	2/2 checks passed
	i-0f1d06ed82890bc5d	Running	t2.micro	2/2 checks passed

**Step 9:** To delete the instances permanently delete the Auto Scaling group as well as Load Balancers, Target Groups and Launch Template.