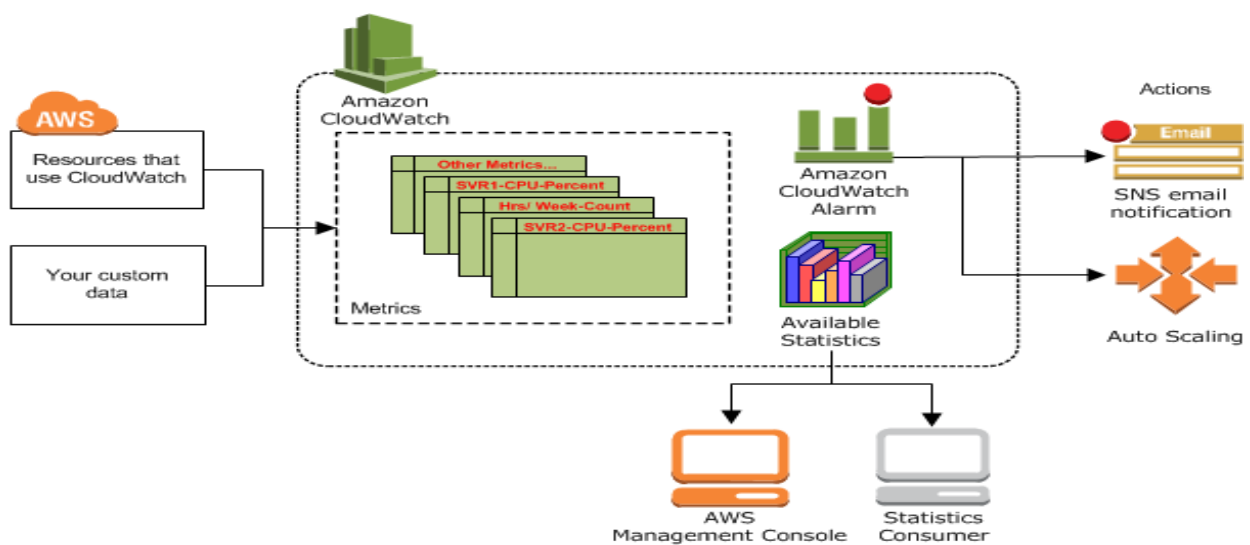
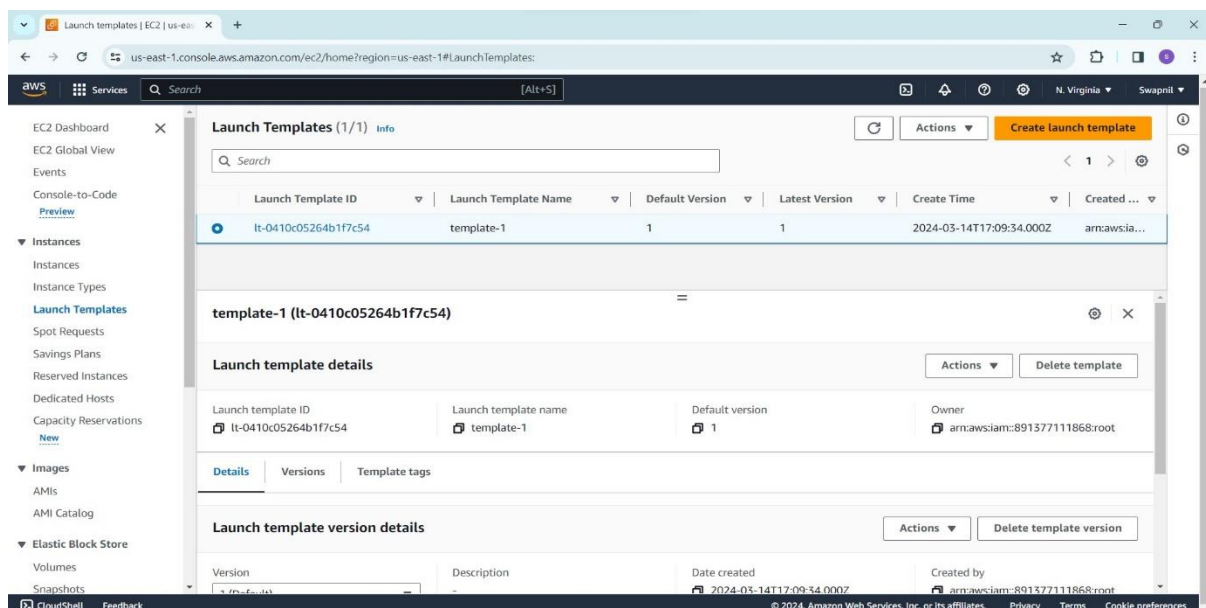




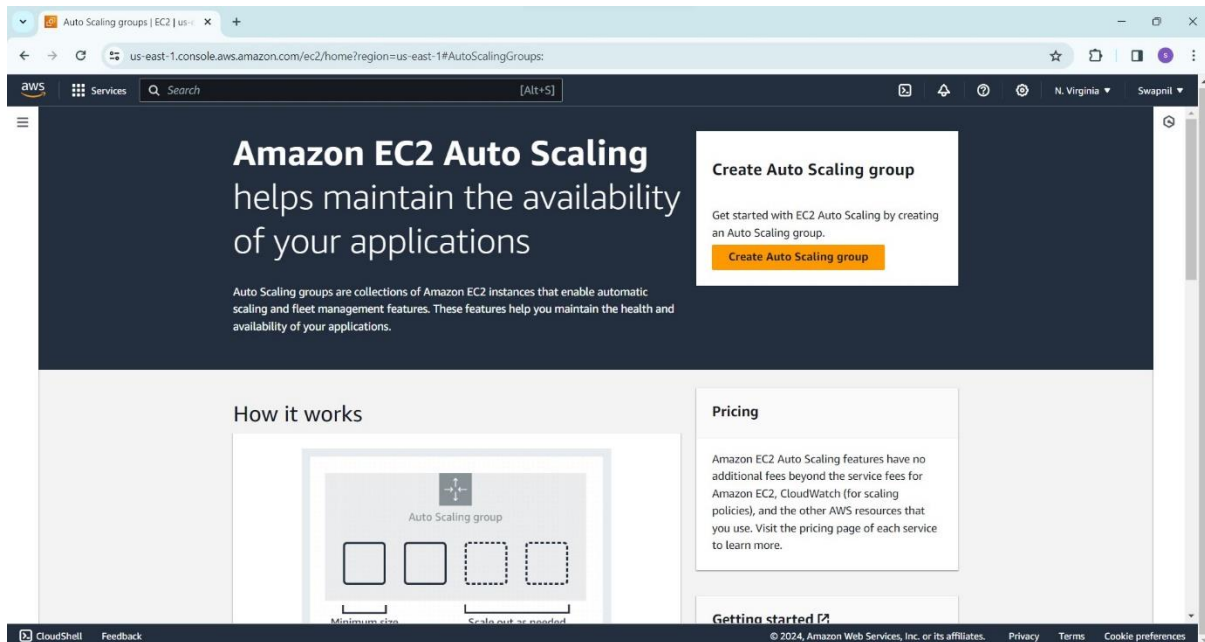
Monitor CloudWatch metrics for your Auto Scaling groups.



Step 1: Create launch template.

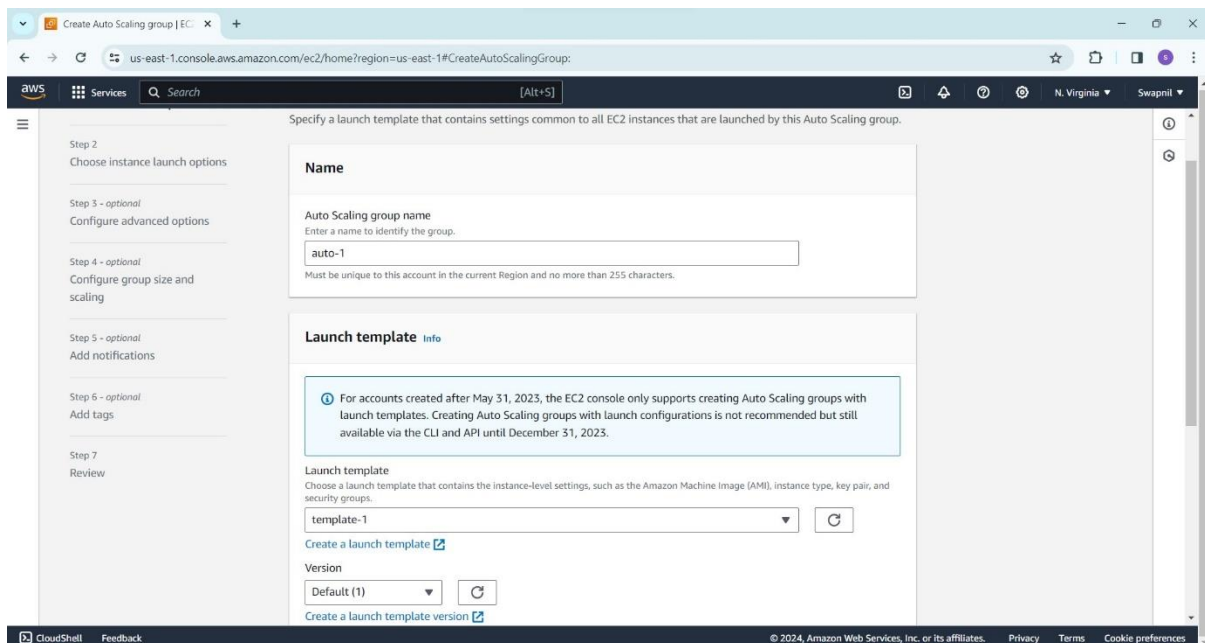


Step 2: Create auto scaling group.



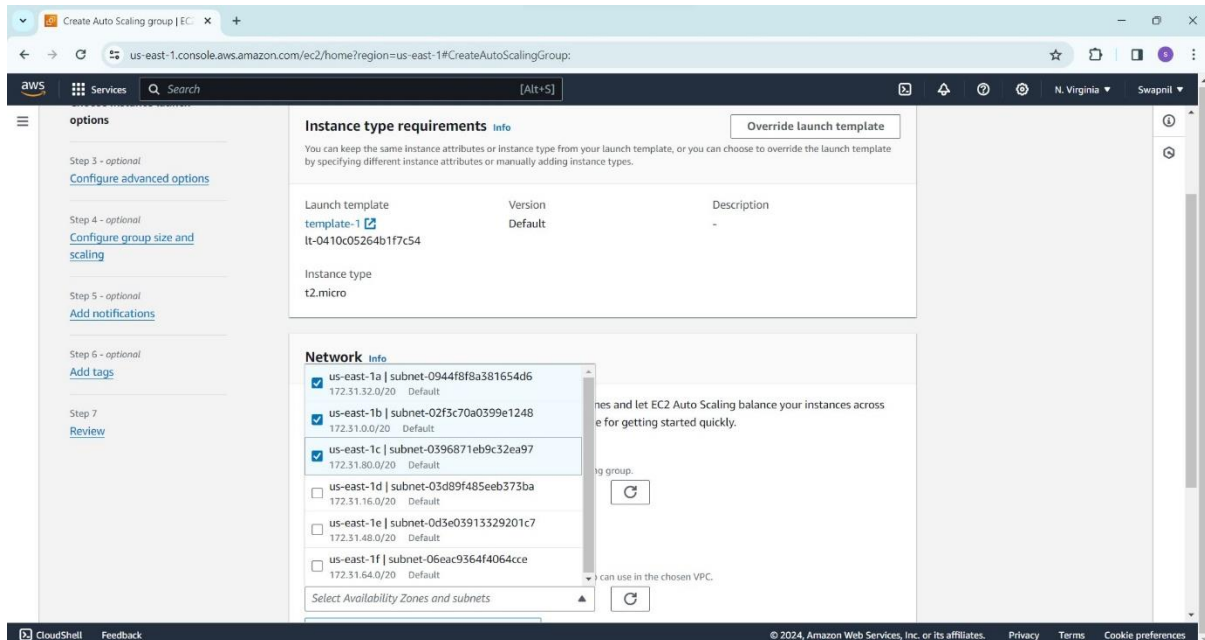
Step 3: Choose launch template.

- Enter the auto scaling group name.
- Select launch template.



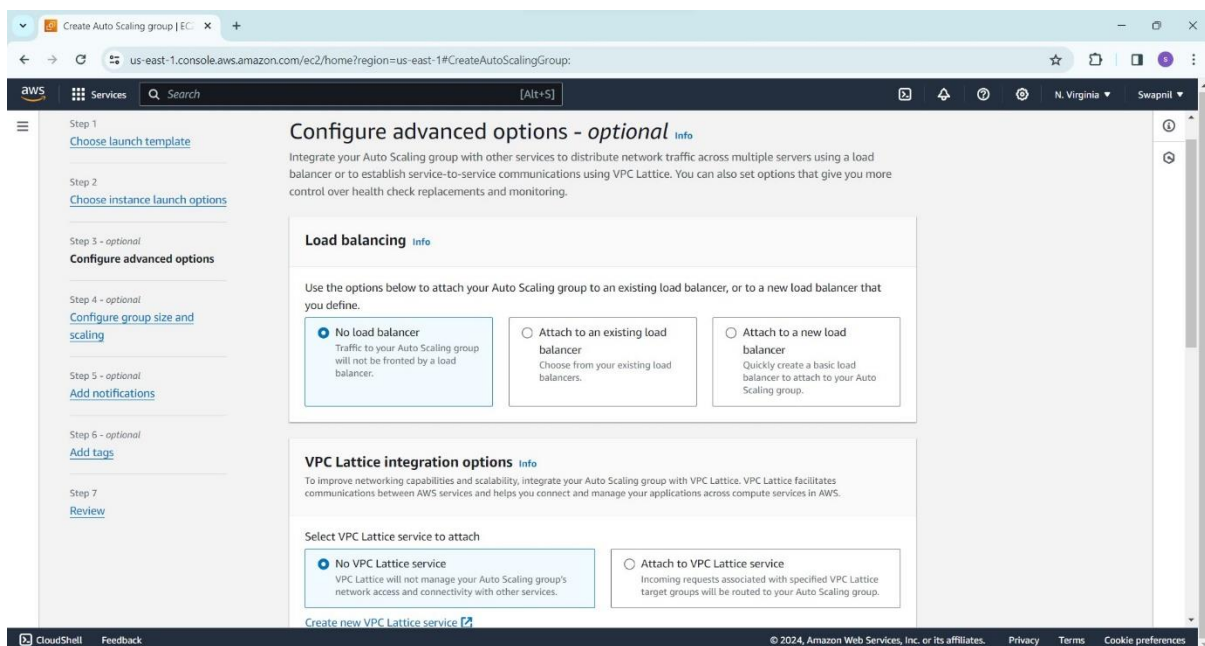
Step 4: Choose instance launch options.

- Select availability zones and subnets.



Step 5: Configure advanced options.

- Select any Load balancer option.



Step 6: Configure group size and scaling.

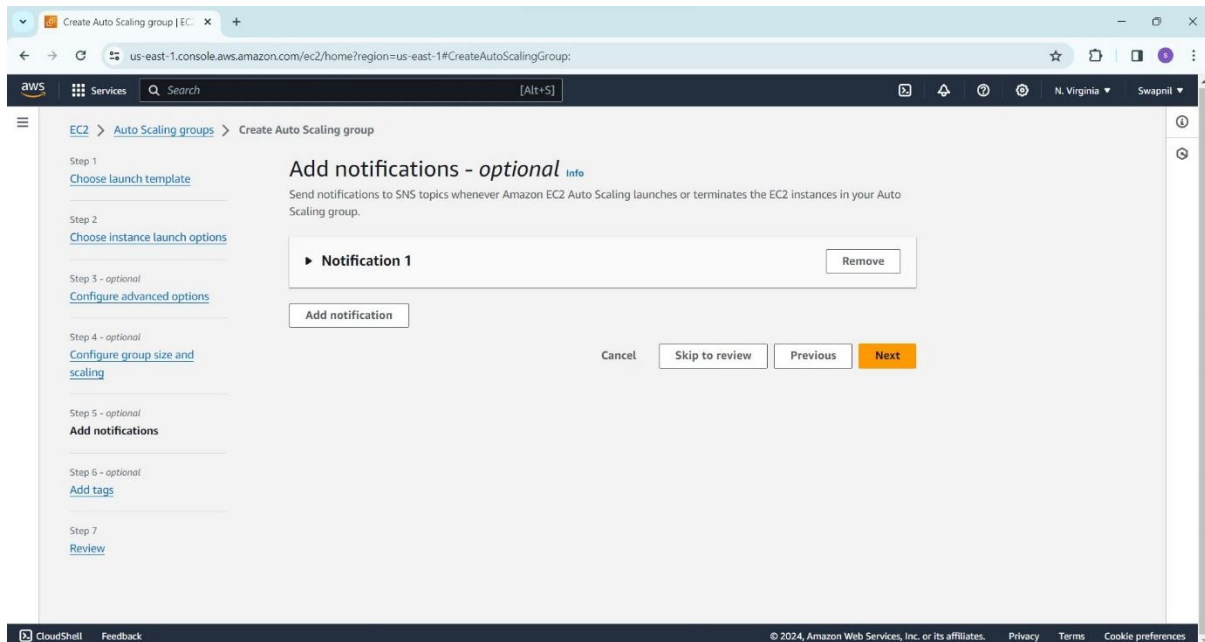
- Select the minimum, desired and maximum capacity.

The screenshot shows the AWS Management Console interface for creating an Auto Scaling group. The left sidebar lists the steps: 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), and Step 7 (Review). The main content area is titled 'Define your group's desired capacity and scaling limits. You can optionally add automatic scaling to adjust the size of your group.' It contains three sections: 'Group size' with a 'Desired capacity type' dropdown set to 'Units (number of instances)' and a 'Desired capacity' input field set to '1'; 'Scaling' with 'Scaling limits' input fields for 'Min desired capacity' (set to '1') and 'Max desired capacity' (set to '5'); and 'Automatic scaling - optional' which is currently collapsed.

Step 7: Select scaling policies.

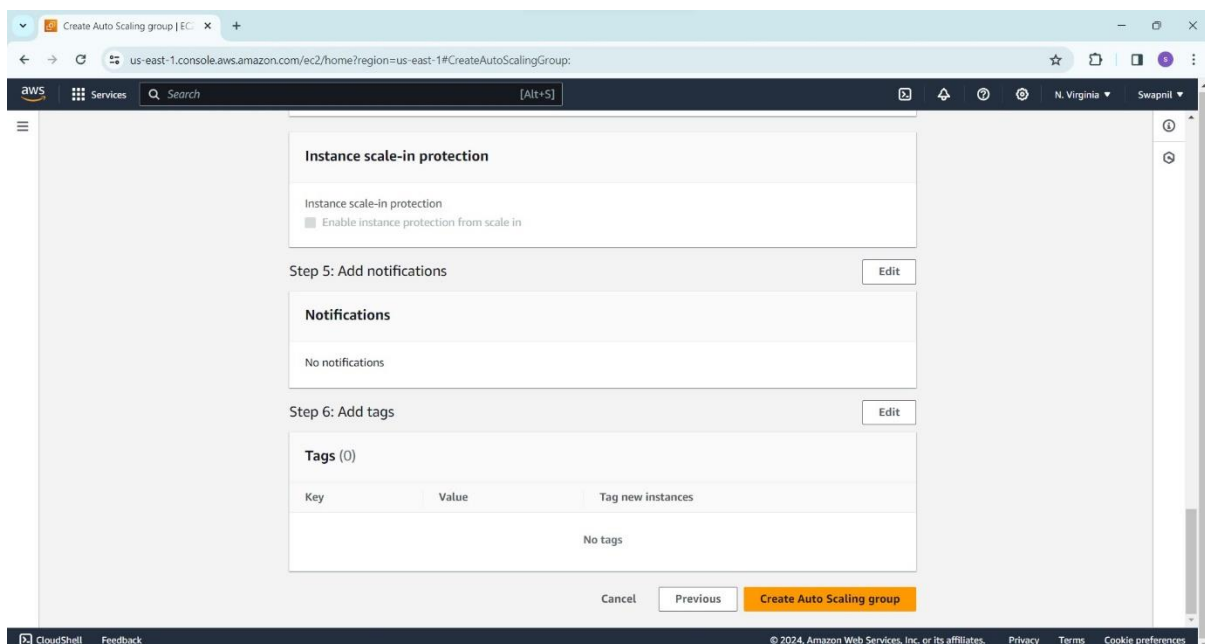
The screenshot shows the AWS Management Console interface for creating an Auto Scaling group, specifically the 'Automatic scaling - optional' section. It prompts the user to 'Choose whether to use a target tracking policy'. Two options are available: 'No scaling policies' (selected with a radio button) and 'Target tracking scaling policy' (selected with a radio button). Below the 'Target tracking scaling policy' option, there are fields for 'Scaling policy name' (set to 'Target Tracking Policy'), 'Metric type' (set to 'Average CPU utilization'), 'Target value' (set to '50'), and 'Instance warmup' (set to '300' seconds). There is also a checkbox for 'Disable scale in to create only a scale-out policy' which is currently unchecked. The bottom section, 'Instance maintenance policy - new', is partially visible.

Step 8: Add notifications.

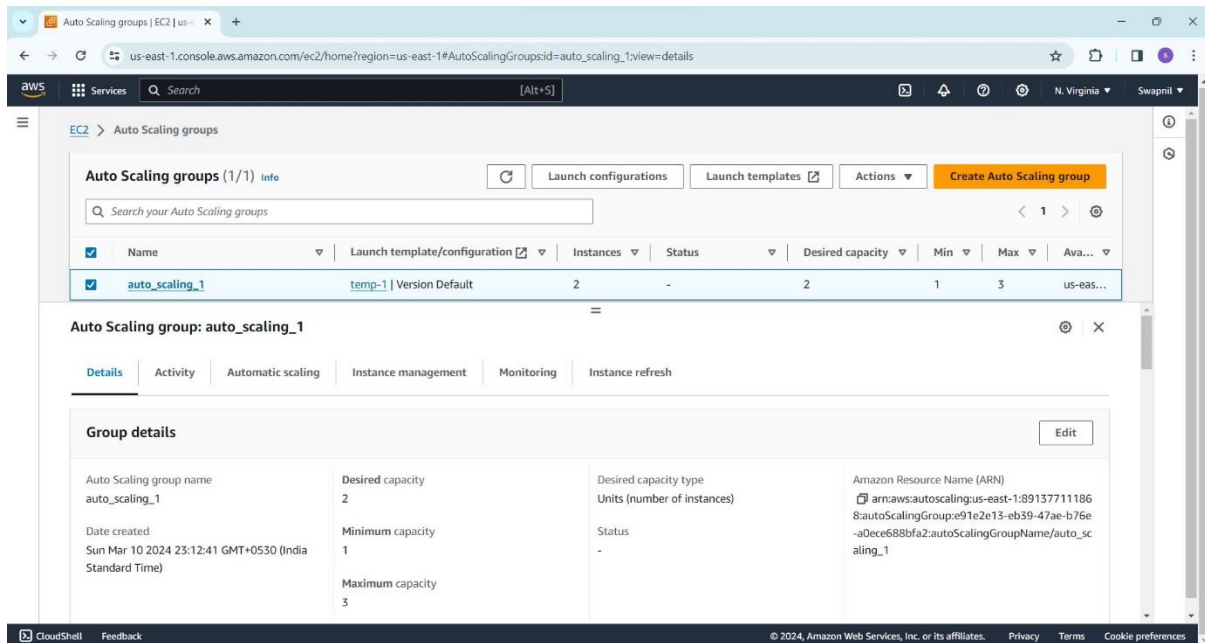


Step 9: Add tags and review.

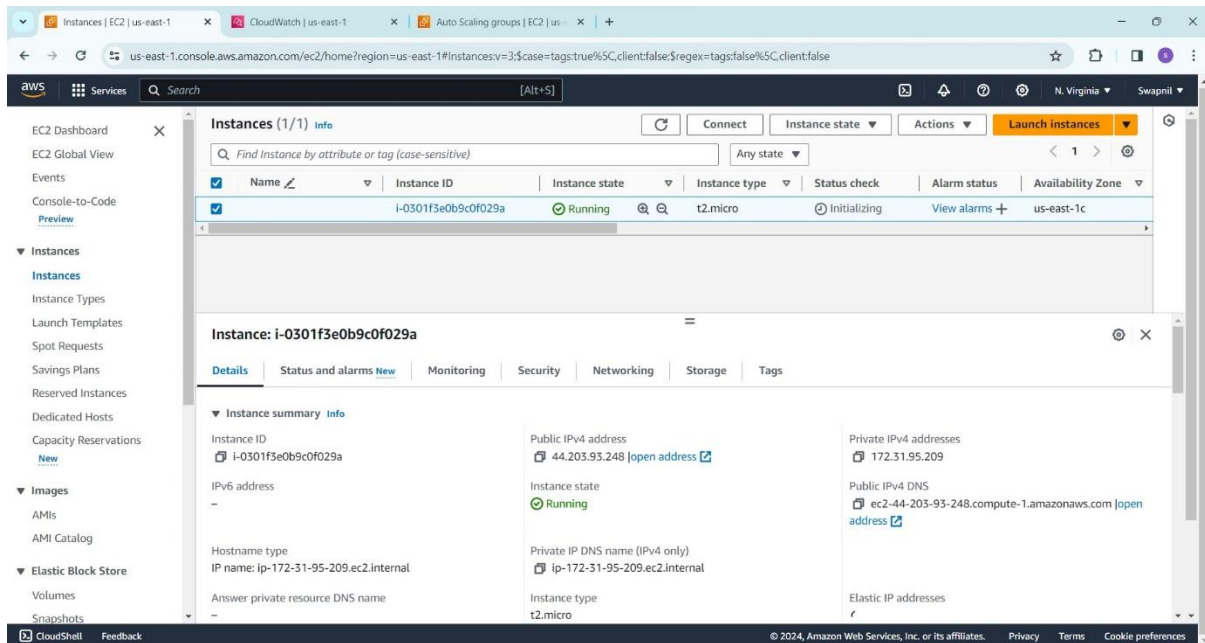
➤ Click on create Auto Scaling groups.



Step 10: Auto scaling group is created.

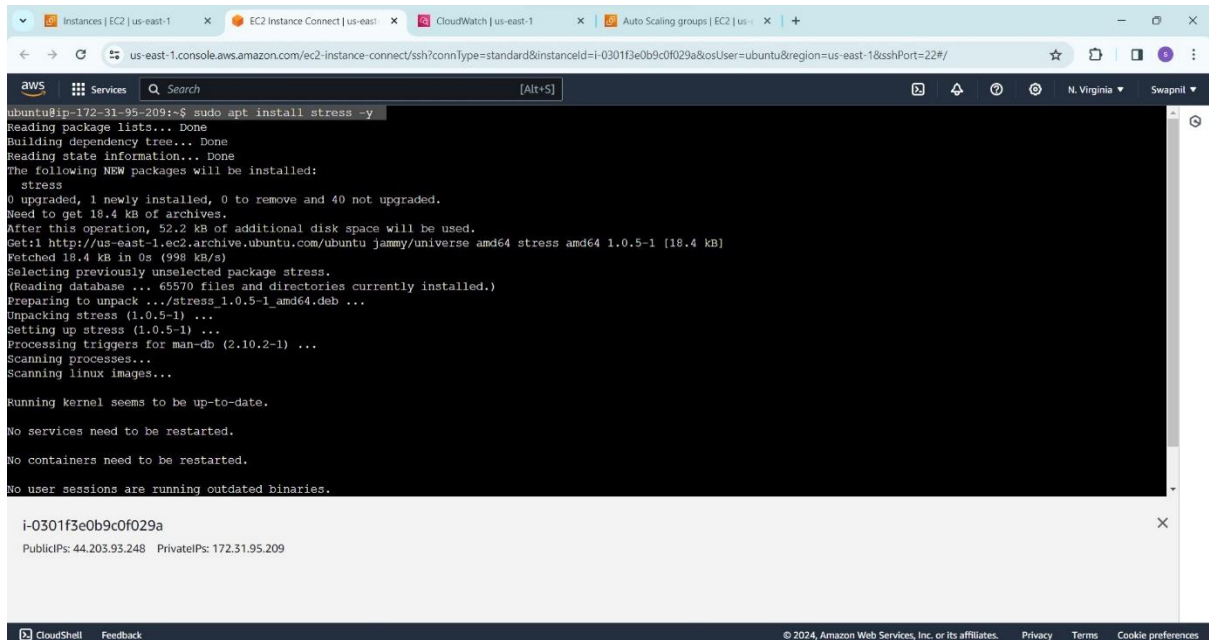


Step 11: As describe one instance is launched



Step 12: Connect to the instance and install the stress packet

➤ Sudo apt install stress -y



The screenshot shows the AWS Management Console with the EC2 Instance Connect terminal open. The terminal displays the output of the command `sudo apt install stress -y`. The output indicates that the stress package (version 1.0.5-1) is being installed. It shows the package lists, dependency tree, and state information. The package is fetched from the Ubuntu archive and installed. The terminal also shows that no services, containers, or user sessions need to be restarted. The instance ID is `i-0301f3e0b9c0f029a` and the public IP is `44.203.93.248`.

```
ubuntu@ip-172-31-95-209:~$ sudo apt install stress -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  stress
0 upgraded, 1 newly installed, 0 to remove and 40 not upgraded.
Need to get 18.4 kB of archives.
After this operation, 52.2 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 stress amd64 1.0.5-1 [18.4 kB]
Fetched 18.4 kB in 0s (998 kB/s)
Selecting previously unselected package stress.
(Reading database ... 65570 files and directories currently installed.)
Preparing to unpack .../stress_1.0.5-1_amd64.deb ...
Unpacking stress (1.0.5-1) ...
Setting up stress (1.0.5-1) ...
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

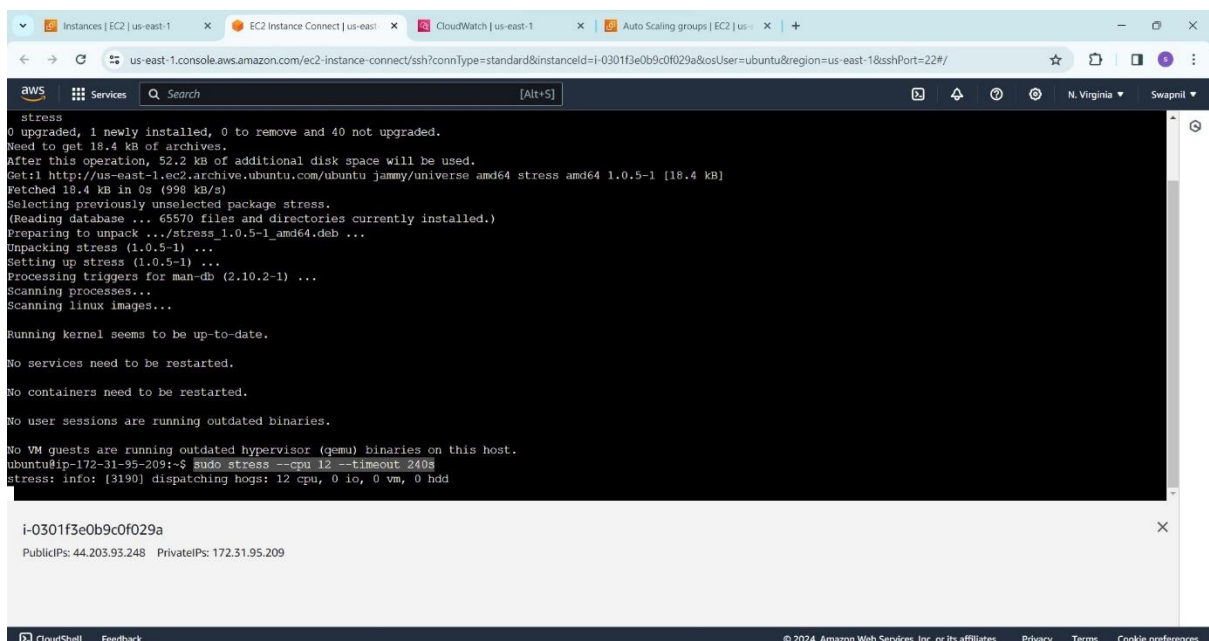
No containers need to be restarted.

No user sessions are running outdated binaries.

i-0301f3e0b9c0f029a
PublicIPs: 44.203.93.248  PrivateIPs: 172.31.95.209
```

Step 13: Run the stress command.

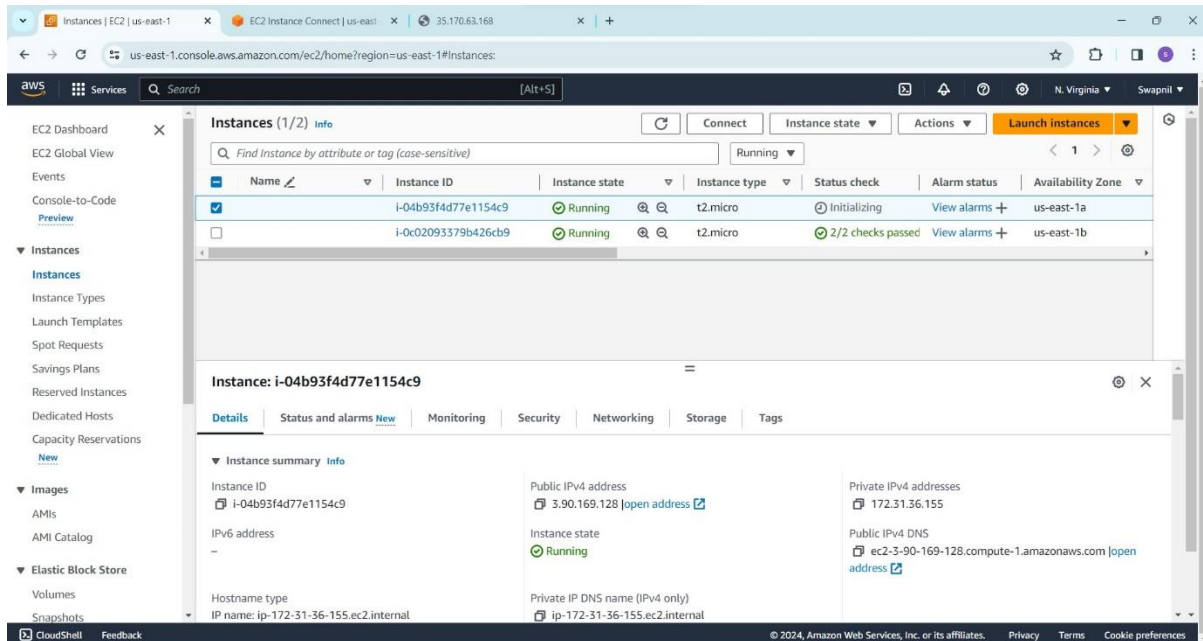
➤ Sudo stress --cpu 12 --timeout 240s.



The screenshot shows the AWS Management Console with the EC2 Instance Connect terminal open. The terminal displays the output of the command `sudo stress --cpu 12 --timeout 240s`. The output shows the stress command being executed with 12 CPU workers for a timeout of 240 seconds. The terminal also shows the same installation output as in Step 12. The instance ID is `i-0301f3e0b9c0f029a` and the public IP is `44.203.93.248`.

```
ubuntu@ip-172-31-95-209:~$ sudo stress --cpu 12 --timeout 240s
stress: info: [3190] dispatching hogs: 12 cpu, 0 io, 0 vm, 0 hdd
```

Step 14: New instance is launched from the auto scaling group.



Step 15: Monitoring from AWS CloudWatch service

