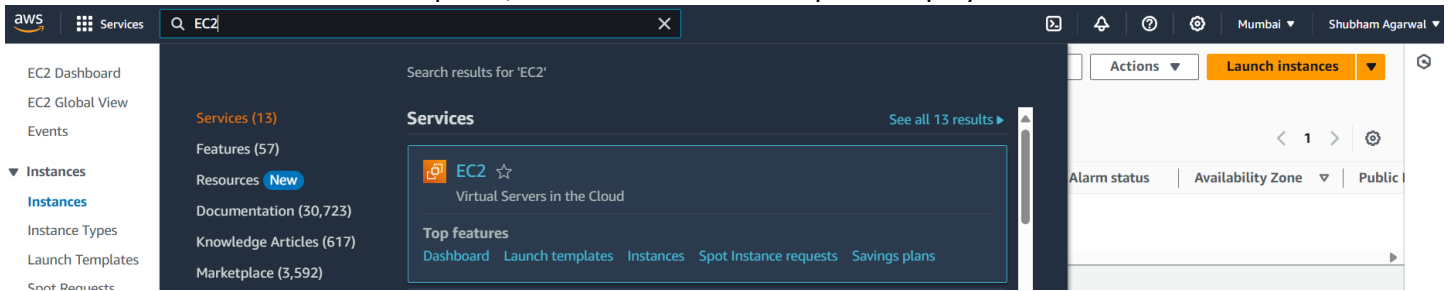


Assignment No : 11

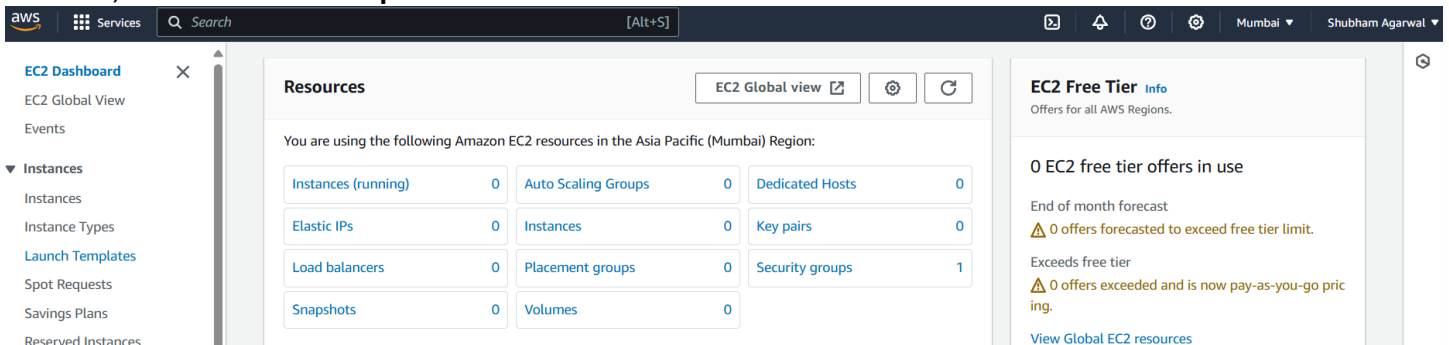
Problem Statement : Build scaling plans in AWS that balance load on different EC2 instances.

Steps:-

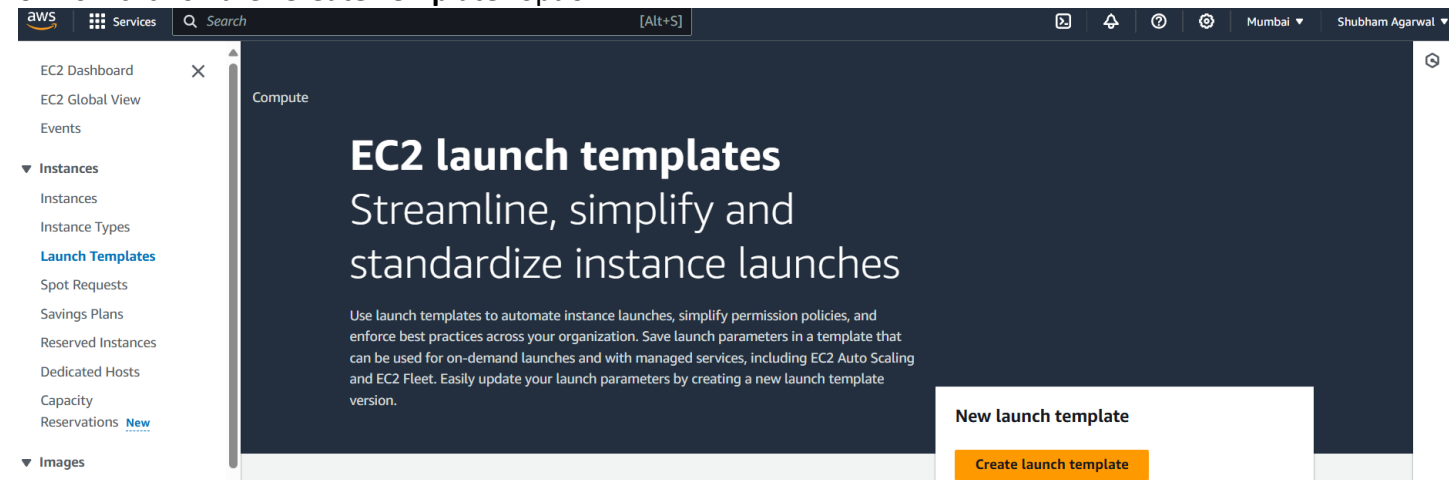
1. Please access AWS and look up EC2, then select the initial option displayed.



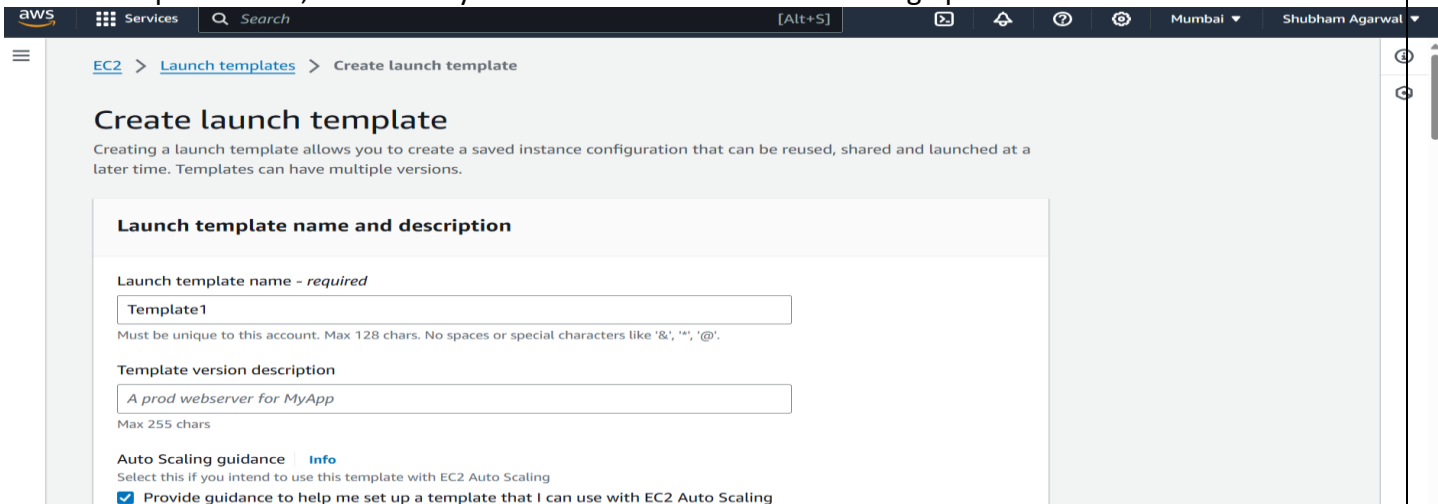
2. Next, select "Launch Template" from the menu on the left side.



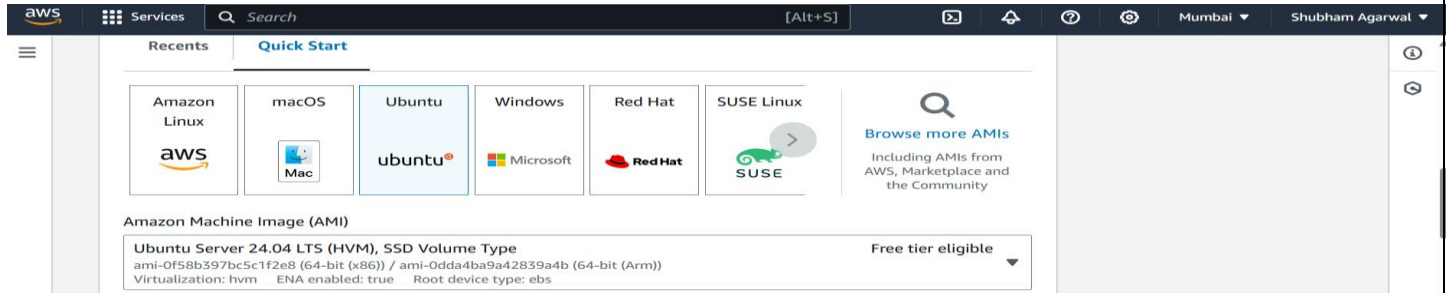
3. Now click on the "Create Template" option.



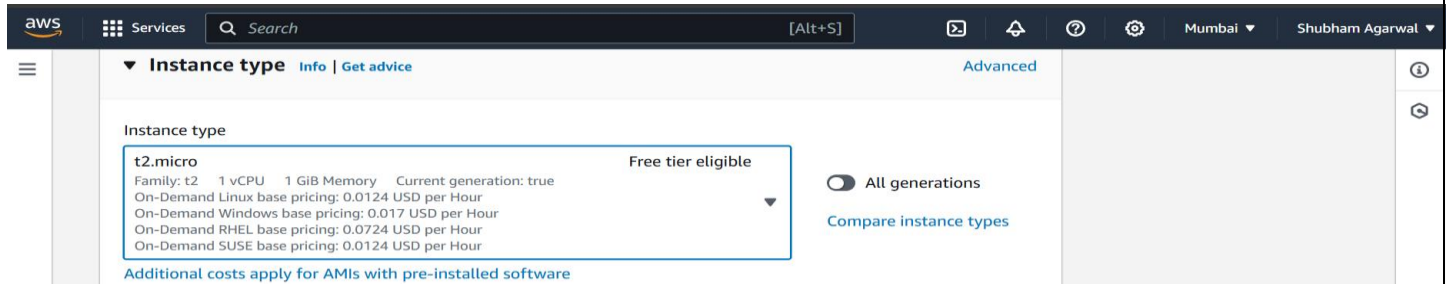
4. Enter a template name, such as "Sky" and check the box for autoscaling options.



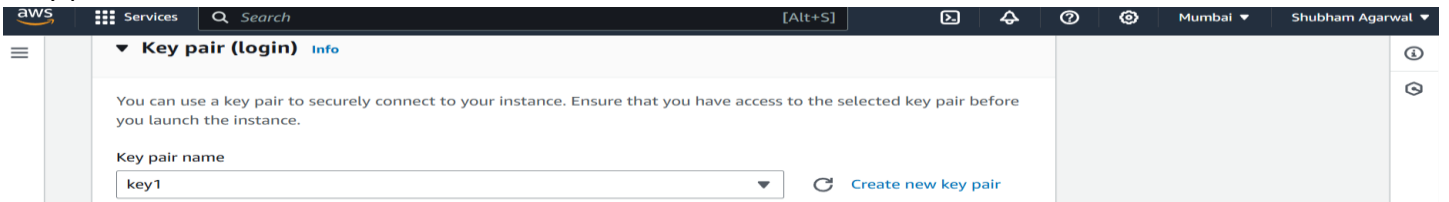
5. Navigate to "Quick Start" and choose "Ubuntu" from the list of available AMIs.



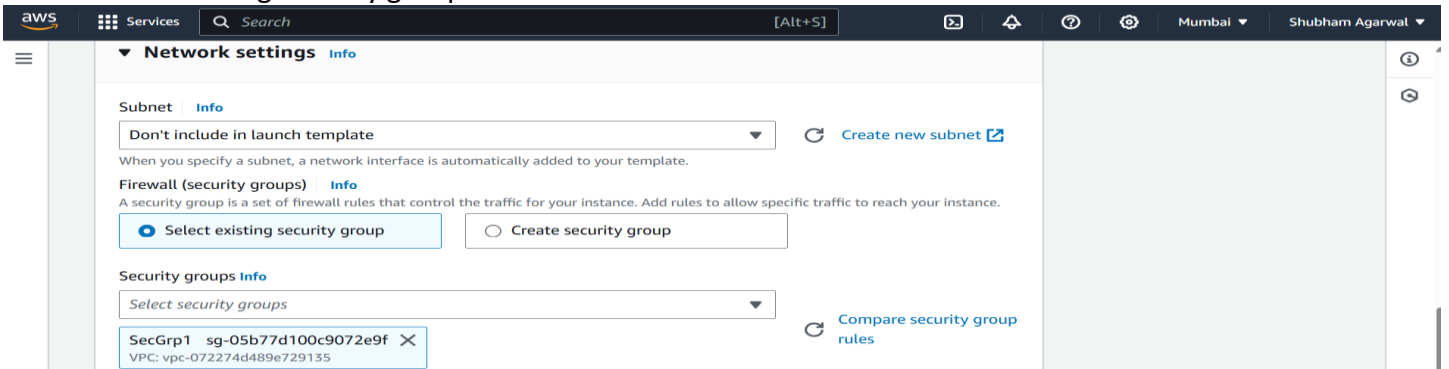
6. Next, select the instance type—either t2.micro or t3.micro, both of which are free tier eligible.



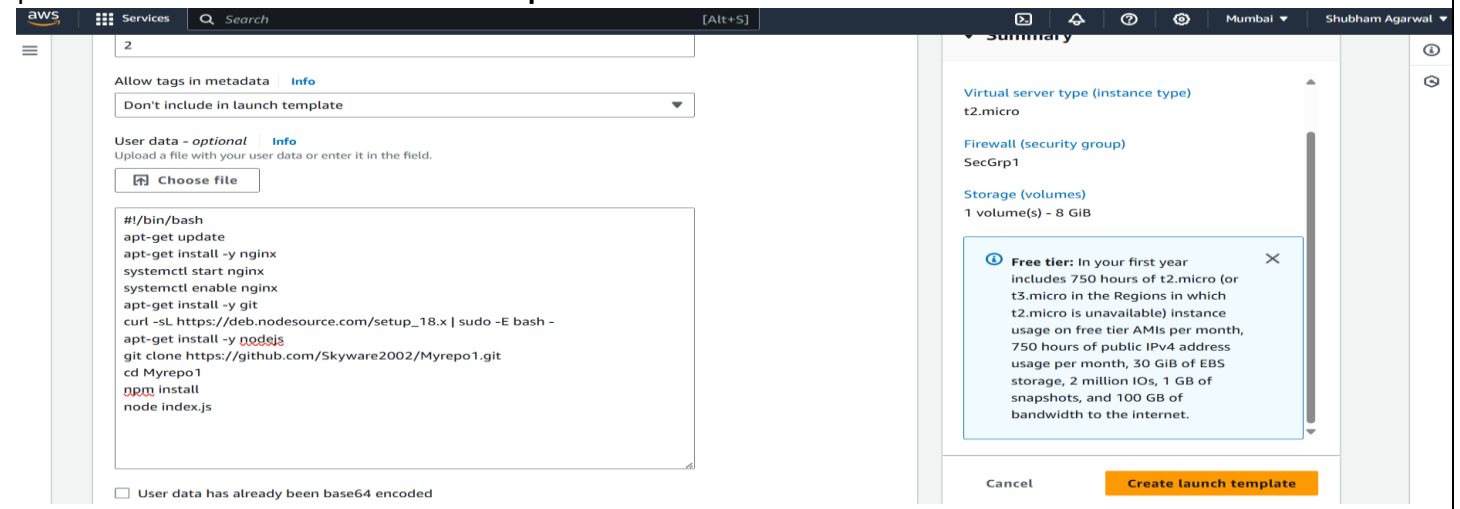
7. Choose either an existing key pair or create a new one if it doesn't exist. For example, use "keypair1" as the key pair name.



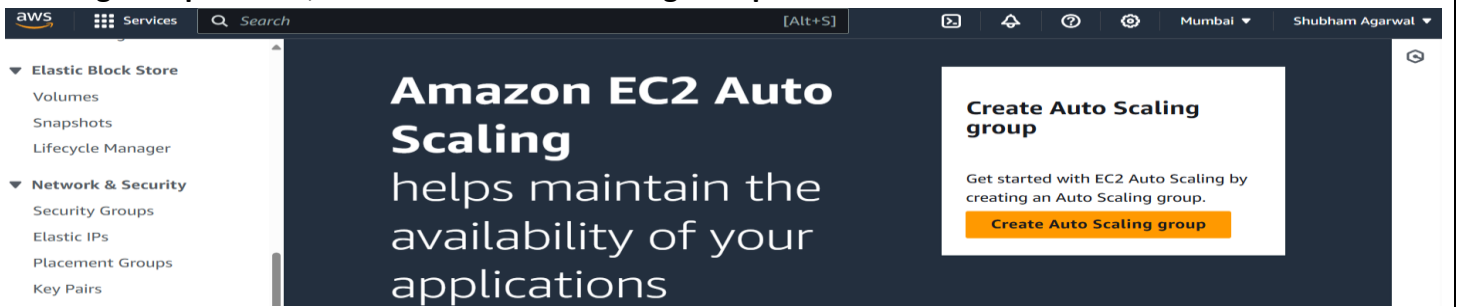
8. Select an existing security group.



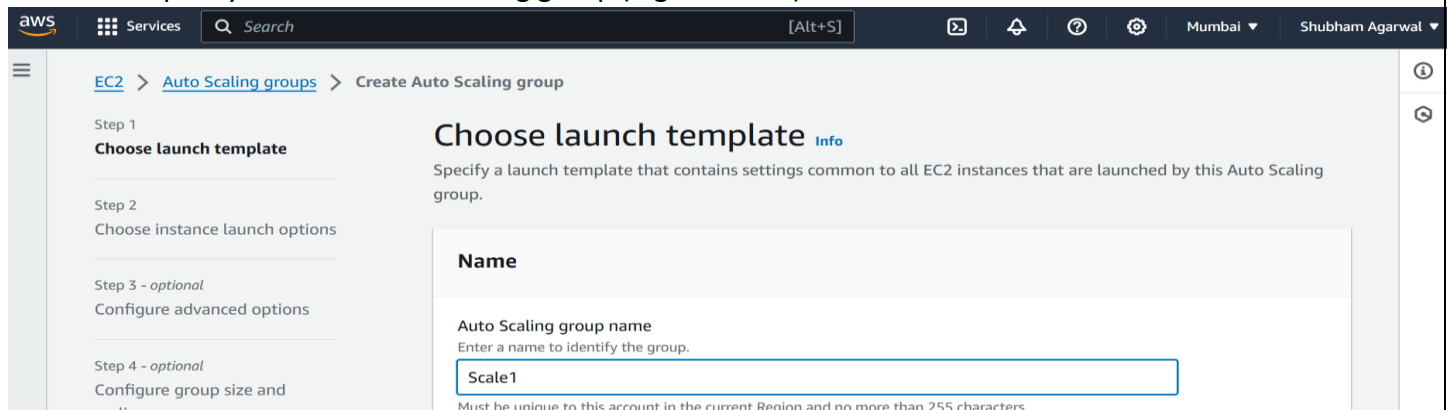
9. Expand the "Advanced details" section, navigate to "User data", and input the provided code. Then proceed to click on "Create launch template".



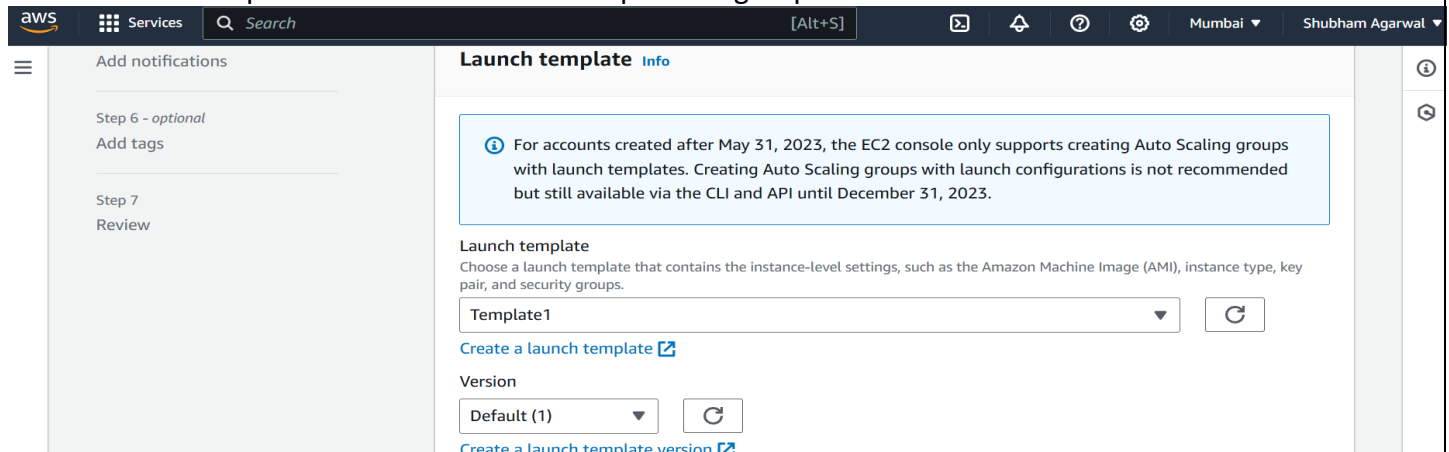
10. Once the launch template has been successfully created, navigate to the left pane and search for "Auto Scaling Groups". Then, select "Create Auto Scaling Group".



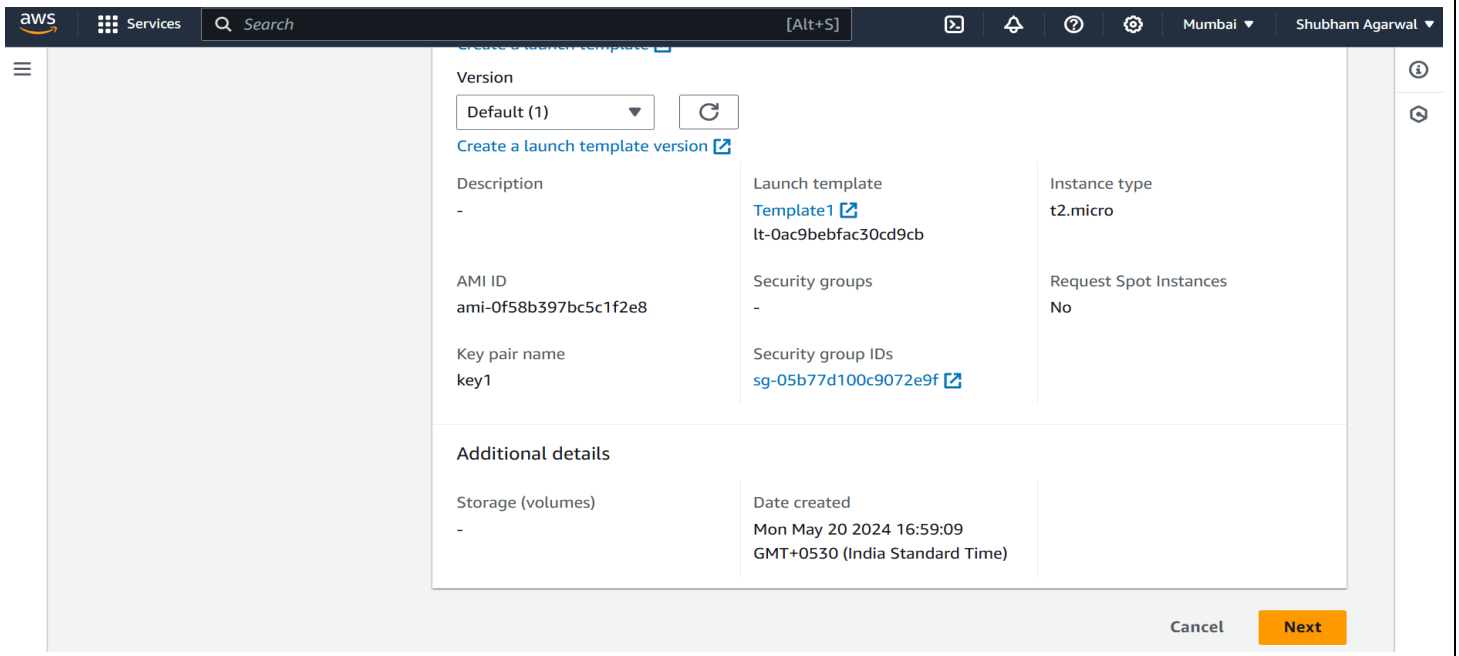
11. Please specify a name for the scaling group (e.g. "Scale1").



12. Select the template that was created in the preceding steps.



13. Proceed to click on "Next".



14. In the following step, choose all available availability zones and subnets, then proceed by clicking "Next".

The screenshot shows the 'Choose the VPC that defines the virtual network for your Auto Scaling group' step in the AWS Management Console. The 'vpc-072274d489e729135' is selected. Under 'Availability Zones and subnets', three options are selected: 'ap-south-1a | subnet-08b94ca84f4955bed', 'ap-south-1b | subnet-091003b3f4d031d43', and 'ap-south-1c | subnet-0bc15b1f07f62ca22'. A warning message states: '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'. The 'Next' button is highlighted.

15. In the subsequent step, begin by selecting "Attach to a new load balancer".

The screenshot shows the 'Load balancing' step. The 'Attach to a new load balancer' option is selected. The description for this option is: 'Quickly create a basic load balancer to attach to your Auto Scaling group.' The 'No load balancer' and 'Attach to an existing load balancer' options are also visible but not selected.

16. Select "Application Load Balancer" as the load balancer type and "Internet-facing" as the load balancer scheme.

The screenshot shows the 'Attach to a new load balancer' step. The 'Application Load Balancer' type is selected. The 'Load balancer name' is 'Scale1-1'. The 'Load balancer scheme' is 'Internet-facing'. The 'Network mapping' section is also visible.

17. Modify the HTTP port number from 80 to 4000 and designate the scaling group created for default routing.

The screenshot shows the 'Tags' step. The 'Protocol' is 'HTTP', the 'Port' is '4000', and the 'Default routing (forward to)' is 'AutoScaling1-1 | HTTP'. The 'Add tag' button is highlighted.

18. Enable the checkbox to activate health checks and specify a **“health check grace period”**, set here to 224 seconds.

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.
 seconds

19. Without any further modifications, proceed to click on **“Next”**.

Additional settings

Monitoring [Info](#)

☐ **Enable group metrics collection within CloudWatch**
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](#)

20. In this step, specify the desired, minimum, and maximum capacities.

Desired capacity
Specify your group size.

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

Equal or less than desired capacity

Max desired capacity

Equal or greater than desired capacity

21. Next, opt for the **"Target Tracking Scaling Policy"** and configure the **CPU utilization target value** to 50. Additionally, set the **instance warm-up time** to 240 seconds.

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

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.

Target value

Instance warmup [Info](#)
 seconds

22. Proceed by clicking "Next" without making any changes, and finally, select "Create auto Scaling Group".

The first screenshot shows the 'Add notifications - optional' step. It includes a sidebar with steps 1 through 6, and a main area with a description of notifications and an 'Add notification' button. The second screenshot shows the 'Add tags - optional' step. It includes a sidebar with steps 1 through 6, and a main area with a description of tags, a 'Tags (0)' section, and an 'Add tag' button. The third screenshot shows the final 'Create Auto Scaling group' step. It includes a sidebar with steps 1 through 6, and a main area with 'Instance scale-in protection', 'Step 5: Add notifications', and 'Step 6: Add tags'.

23. After the auto-scaling group is created, return to the EC2 dashboard and navigate to the "Instances" section for running instances.

The screenshot shows the EC2 Dashboard with the 'Instances' section selected. The 'Resources' section displays a table of EC2 resources in the Asia Pacific (Mumbai) Region:

Resource	Count
Instances (running)	1
Elastic IPs	0
Load balancers	1
Snapshots	0
Auto Scaling Groups	1
Instances	2
Placement groups	0
Volumes	2
Dedicated Hosts	0
Key pairs	1
Security groups	2

24. Since the minimum capacity chosen was 2, two instances have been created.

The screenshot shows the 'Instances (2)' page. It includes a sidebar with the 'Instances' section selected, and a main area with a table of instances:

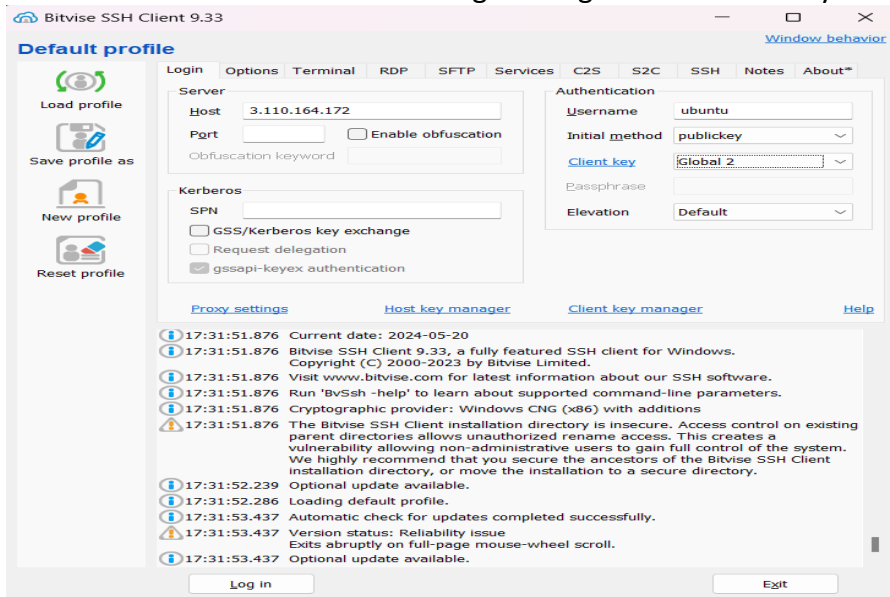
Name	Instance ID	Instance state	Instance type	Status check
	i-040da7fad69b3da27	Running	t2.micro	Initializing
	i-04c55238926811718	Running	t2.micro	Initializing

25. Choose any one of the instance IDs and copy the public IPv4 address.

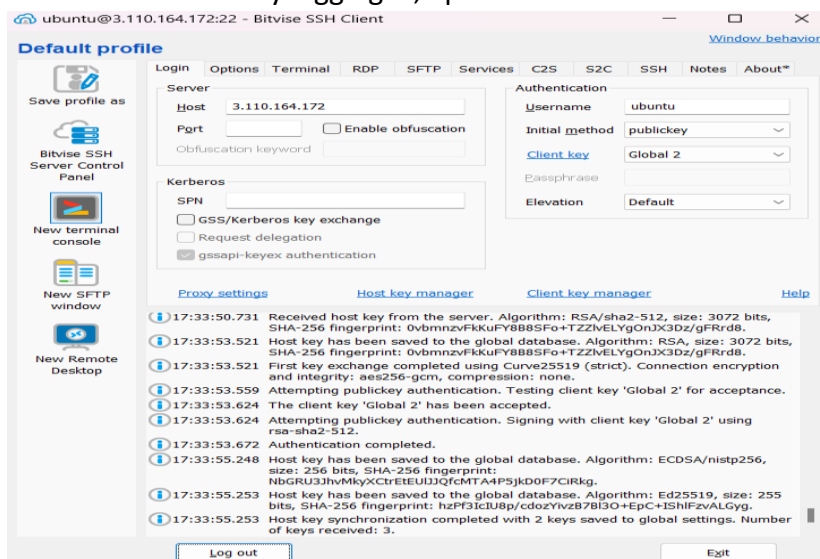
The screenshot shows the 'Instance summary' page for instance i-040da7fad69b3da27. It includes a sidebar with the 'Instances' section selected, and a main area with the instance details:

Instance ID	Public IPv4 address	Private IPv4 addresses
i-040da7fad69b3da27	3.110.164.172 open address	172.31.0.19

26. Launch the Bitwise SSH Client and log in using the IPv4 address you copied earlier.



27. After successfully logging in, open a new terminal console from the left pane.



28. Now write the commands in the terminal as follows: -

→ ***sudo nano infi.sh*** (creates a .sh file)

```
ubuntu@ip-172-31-0-19:~$ sudo nano infi.sh
```

→ Write this code in the file “infi.sh” to run an infinite loop.

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

→ Press Ctrl+X and write ‘y’ to save the file.

→ ***sudo chmod 777 infi.sh*** (to provide all permission to the file)

```
ubuntu@ip-172-31-0-19:~$ sudo nano infi.sh
ubuntu@ip-172-31-0-19:~$ sudo chmod 777 infi.sh
```

→ ***sh infi.sh*** (Run the .sh file)

```
ubuntu@3.110.164.172:22 - Bitwise xterm - ubuntu@ip-172-31-0-19: ~
Inside Loop
Inside Loop
Inside Loop
Inside Loop
Inside Loop
Inside Loop
Inside Loop
```

29. Return to AWS and select both running instances. Below, locate the monitoring options, and choose **"CPU utilization"**. Then, enlarge the view.

The screenshot shows the AWS Management Console with two EC2 instances selected. The 'Instances (2/2)' table lists the following:

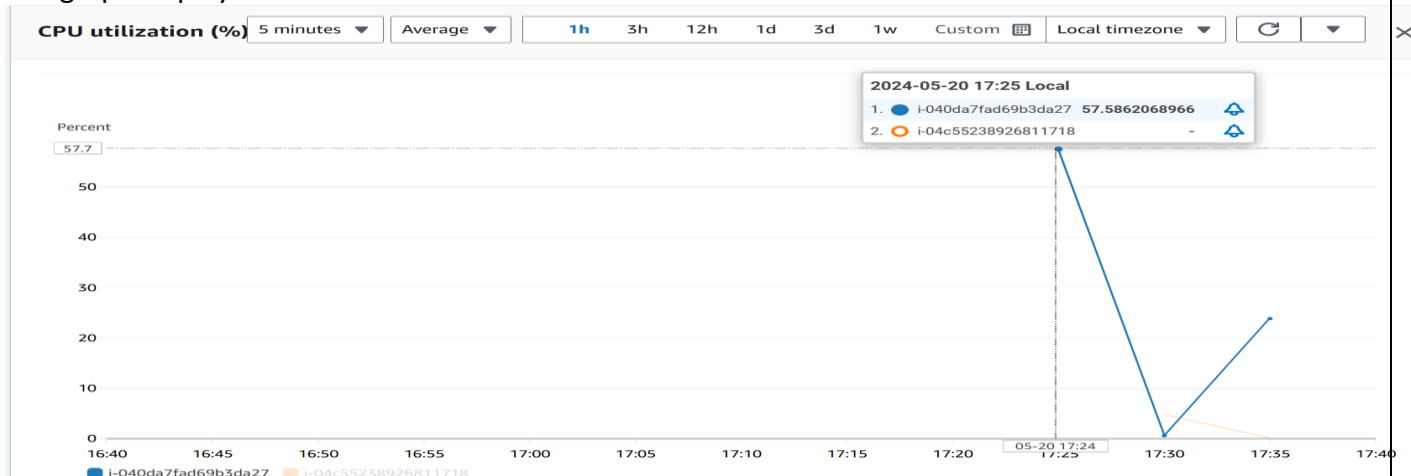
Name	Instance ID	Instance state	Instance type	Status check
i-040da7fad69b3da27	i-040da7fad69b3da27	Running	t2.micro	2/2 checks passed
i-04c55238926811718	i-04c55238926811718	Running	t2.micro	2/2 checks passed

Below the table, the '2 instances selected' section shows four graphs: CPU utilization, Network in (bytes), Network out (bytes), and Network packet count. A context menu is open over the CPU utilization graph, with the 'Enlarge' option highlighted.

30. From the panel above, select **"Local timezone."**

The screenshot shows the 'CPU utilization (%)' graph in the AWS Monitoring console. The graph is set to 'Average' over a '1h' period. The 'Local timezone' dropdown menu is open, showing the 'Local timezone' option selected.

31. The graph displays CPU utilization for both instances.



When the CPU utilization exceeds the limit for both instances, another instance is created, as we have set the maximum capacity to 3.

The screenshot shows the AWS Management Console with three EC2 instances selected. The 'Instances (3/3)' table lists the following:

Name	Instance ID	Instance state	Instance type	Status check
i-0daf1653bcd62ed08	i-0daf1653bcd62ed08	Running	t2.micro	Initializing
i-040da7fad69b3da27	i-040da7fad69b3da27	Running	t2.micro	2/2 checks passed
i-04c55238926811718	i-04c55238926811718	Running	t2.micro	2/2 checks passed

CPU utilization (%)

5 minutes

Average

1h

3h

12h

1d

3d

1w

Custom



UTC timezone



Percent

80

60

40

20

0

11:25

11:30

11:35

11:40

11:45

11:50

11:55

12:00

12:05

12:10

12:15

12:20

12:25

i-0daf1653bcd62ed08

i-040da7fad69b3da27

i-04c55238926811718

