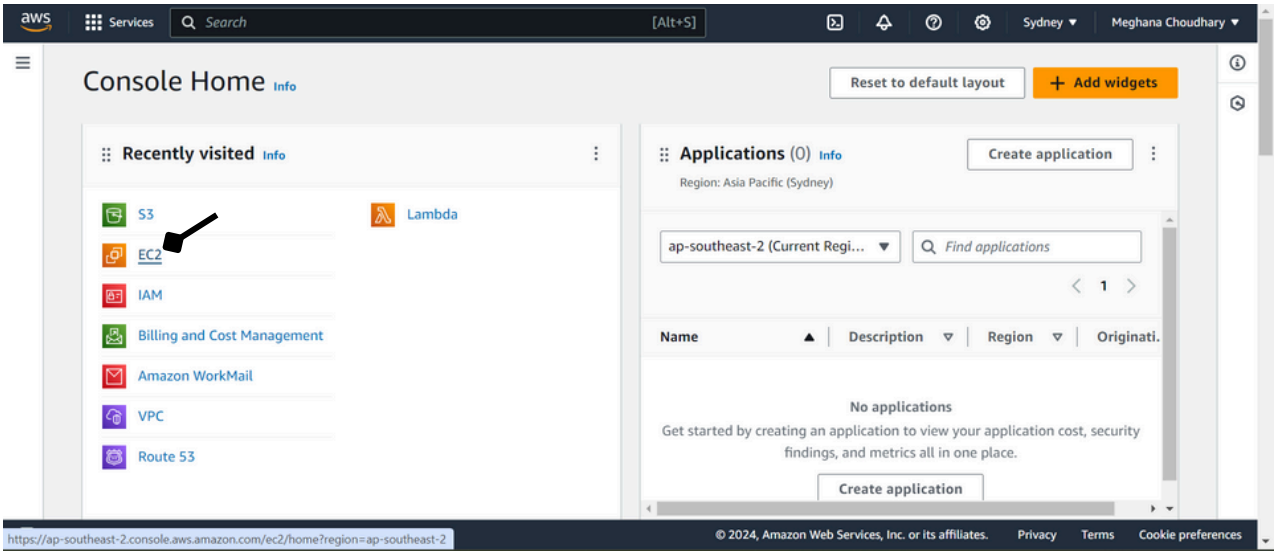
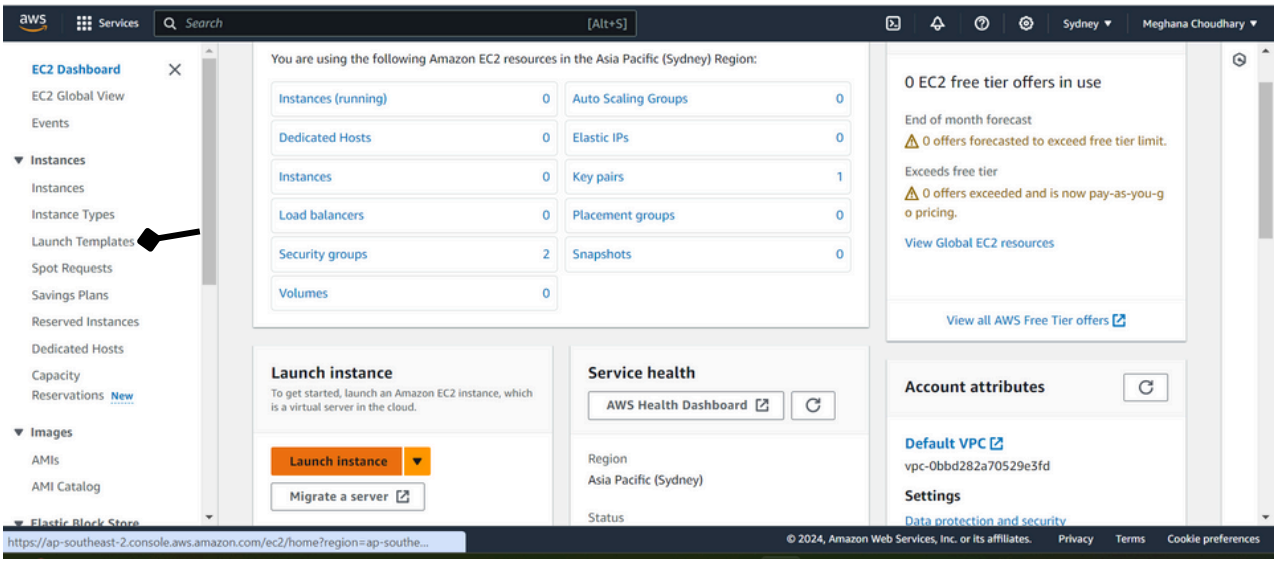
11. PROBLEM STATEMENT:

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

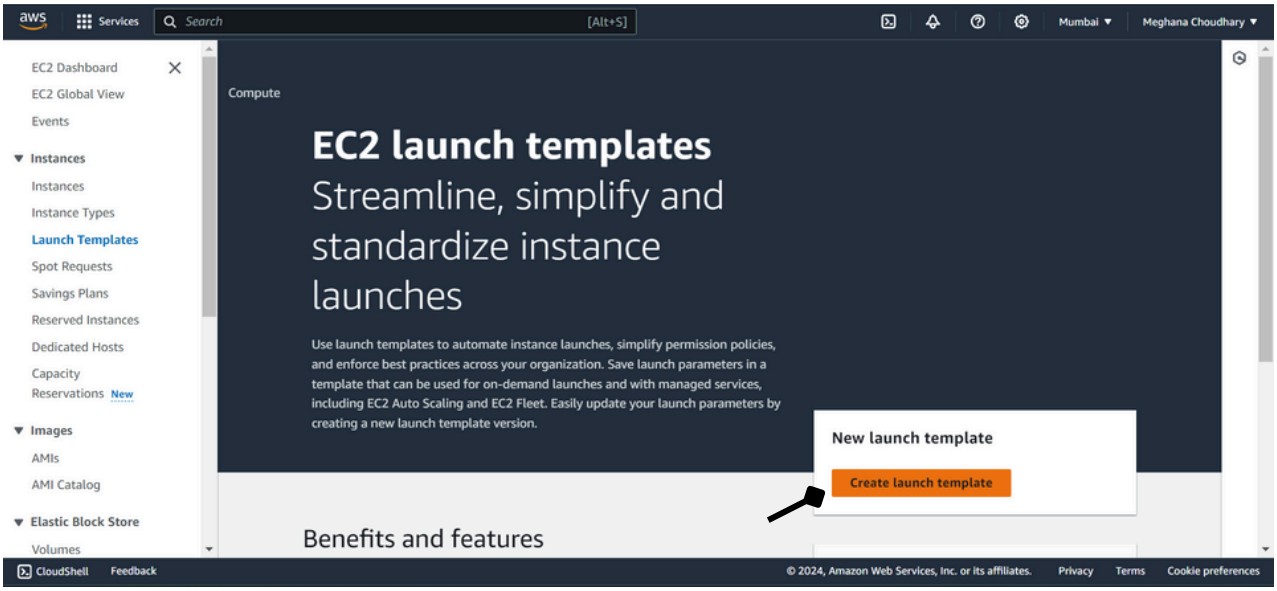
1. Log into AWS console and click on “EC2”.



2. Click on ‘Launch Templates’.

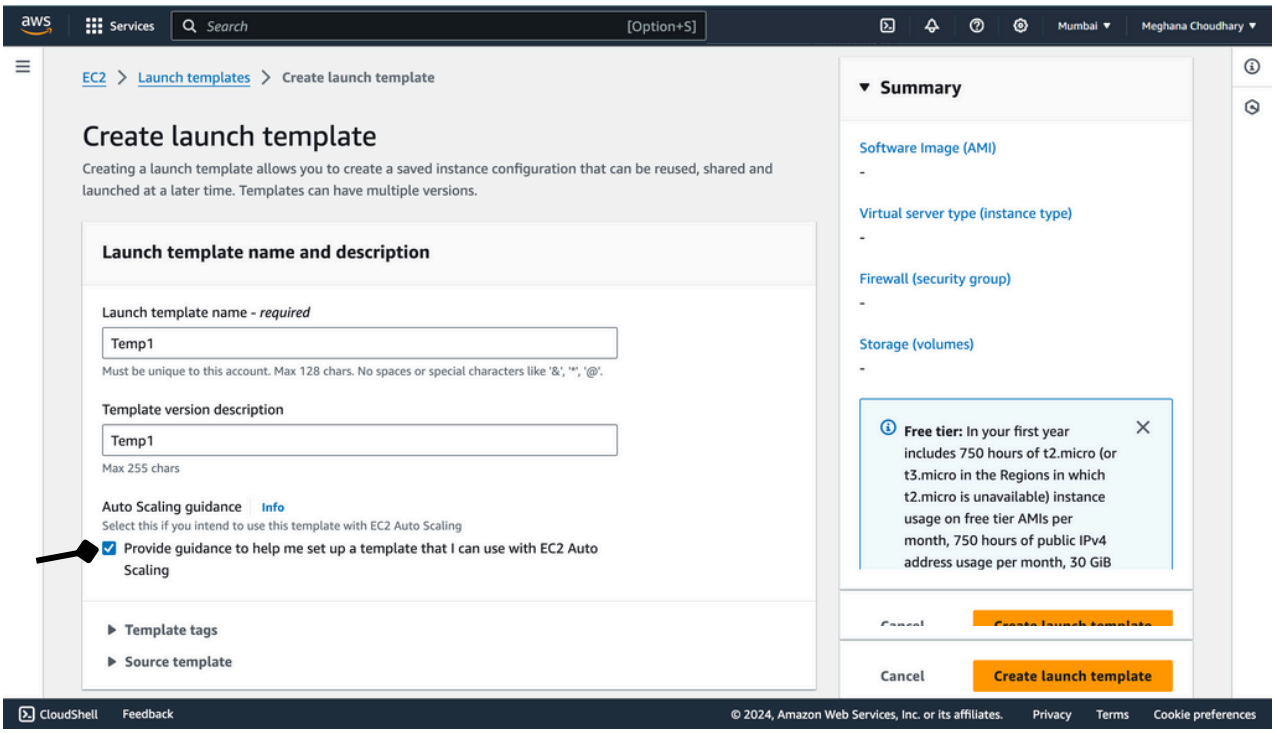


3. Create launch templates.

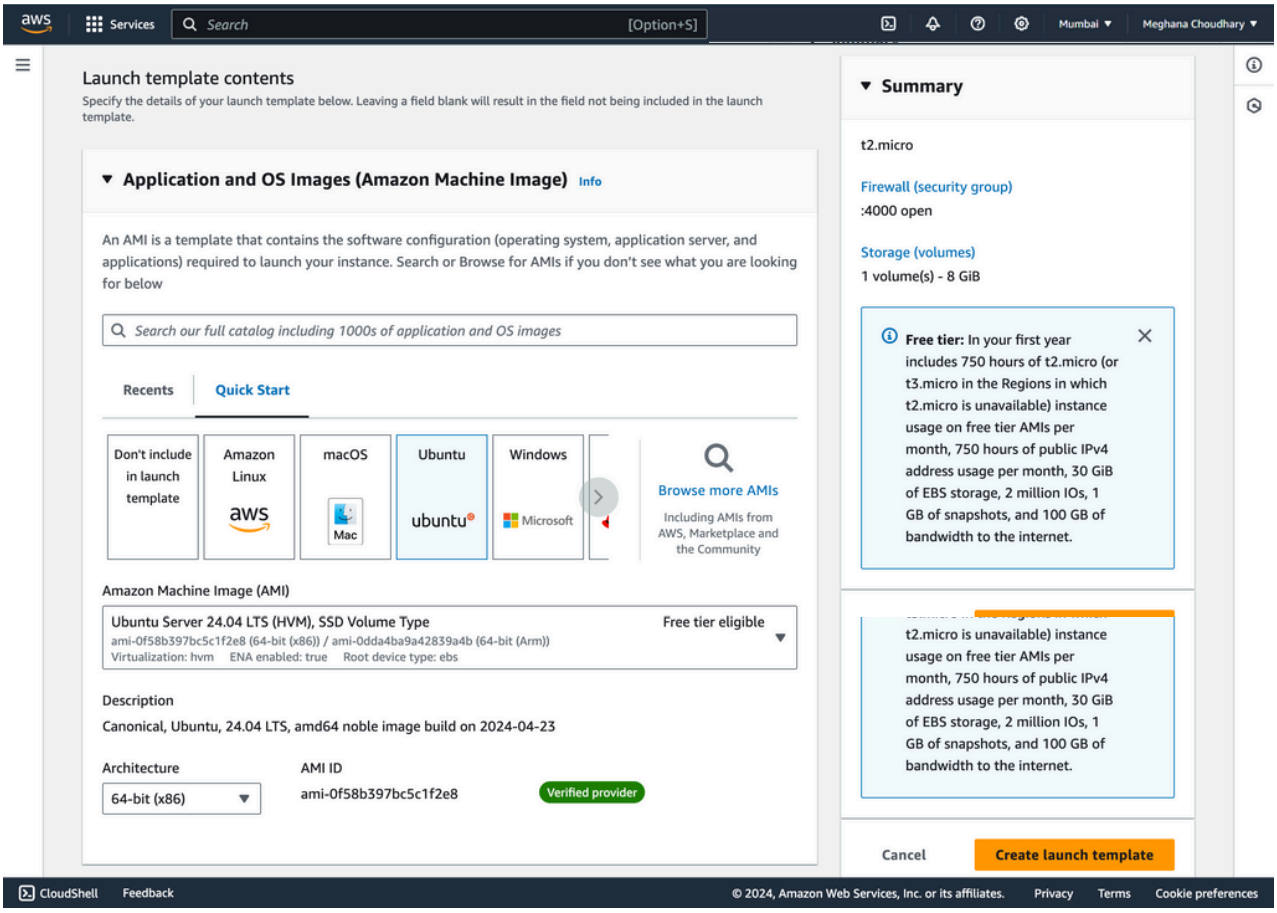


4. Enter necessary details.

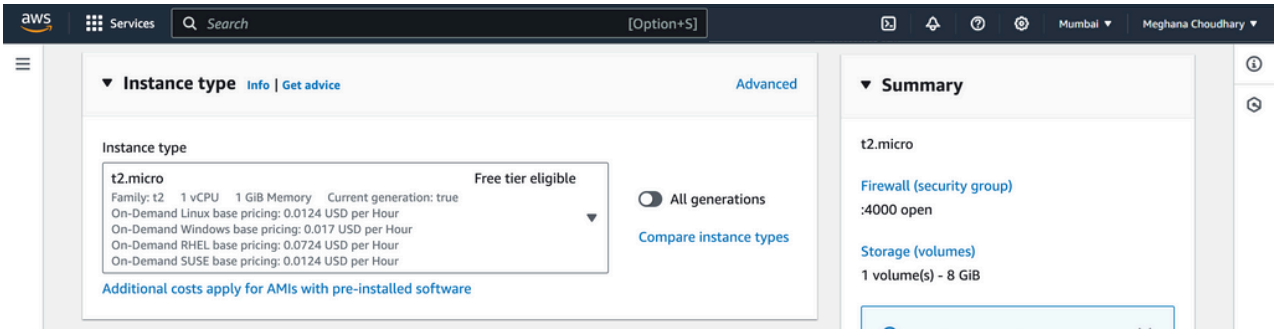
In ‘Launch templet name and description’ give a name and description to your launch templet and select the check box for ‘Provide guidance to help me set up a template that I can use with EC2 Auto Scaling’



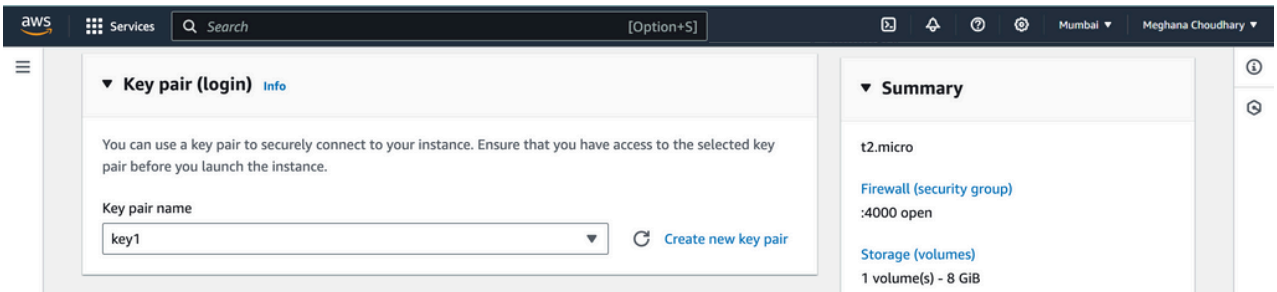
For OS Image select ‘Ubuntu Server’



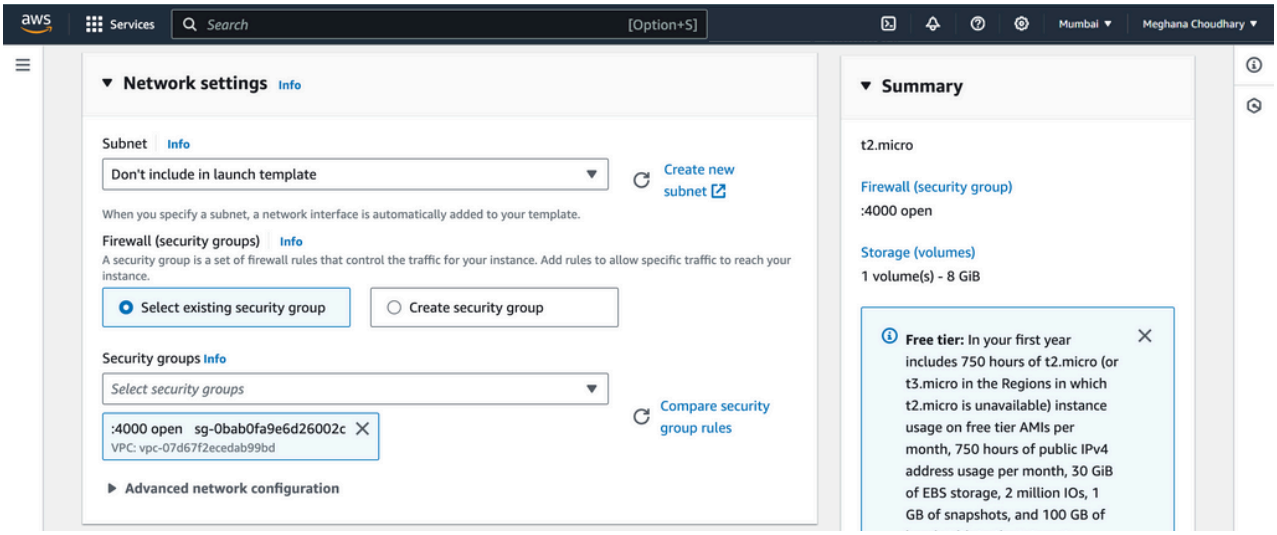
For instance type, select ‘t2.micro’ or any other free tire eligible instance type.



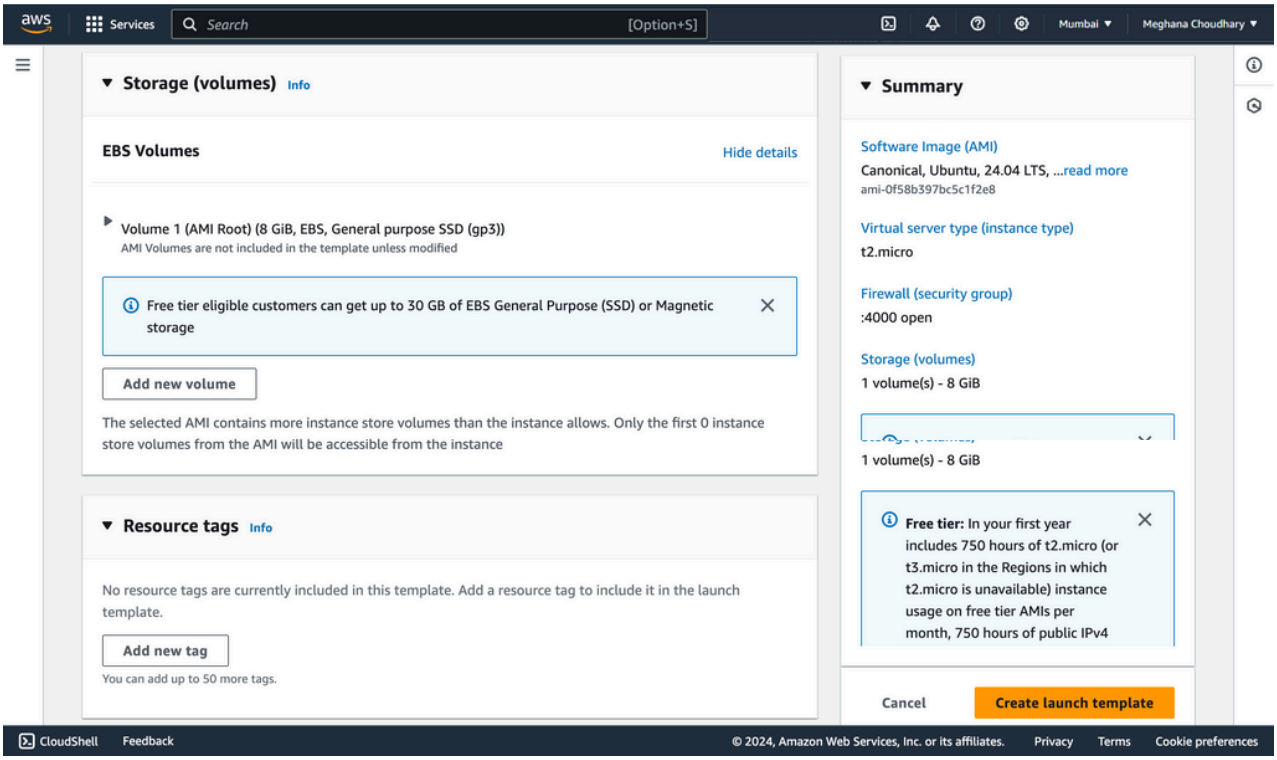
For key pair, select an existing key pair or crate a new key pair.



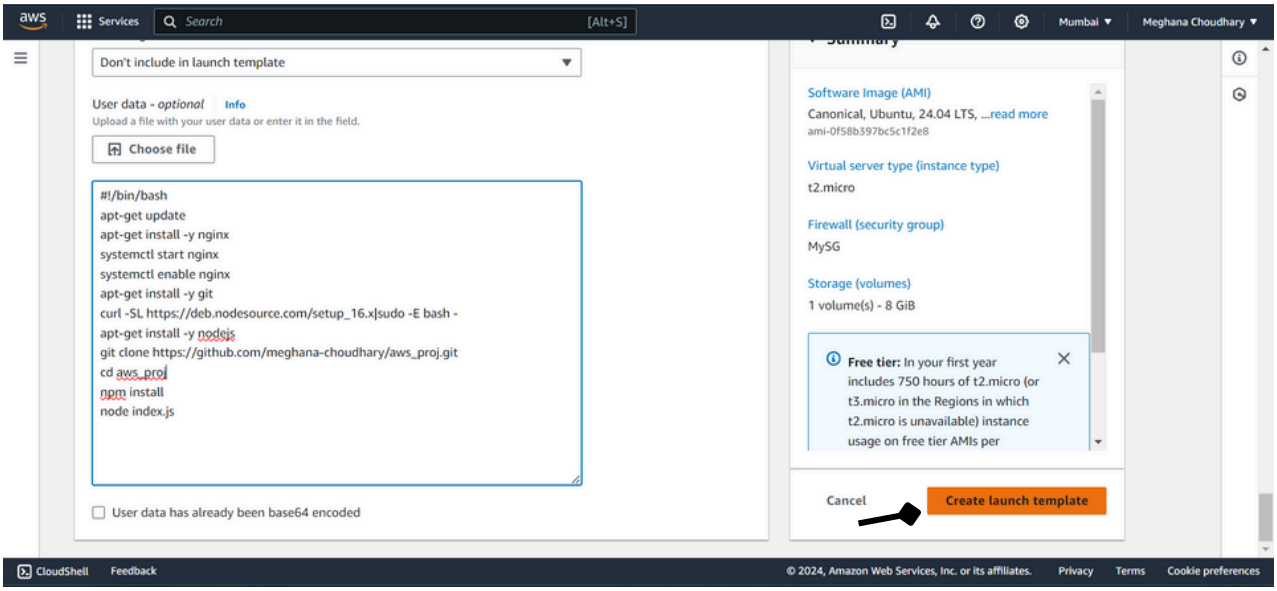
For security group select an existing security group where port 4000 is open for incomeing traffic.



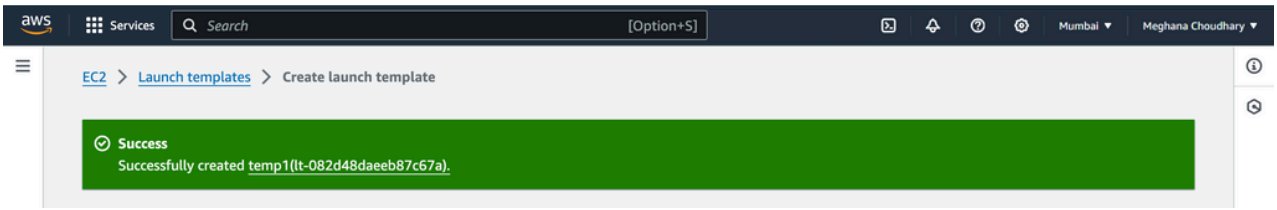
Keep Storage and Resource Tag as default.



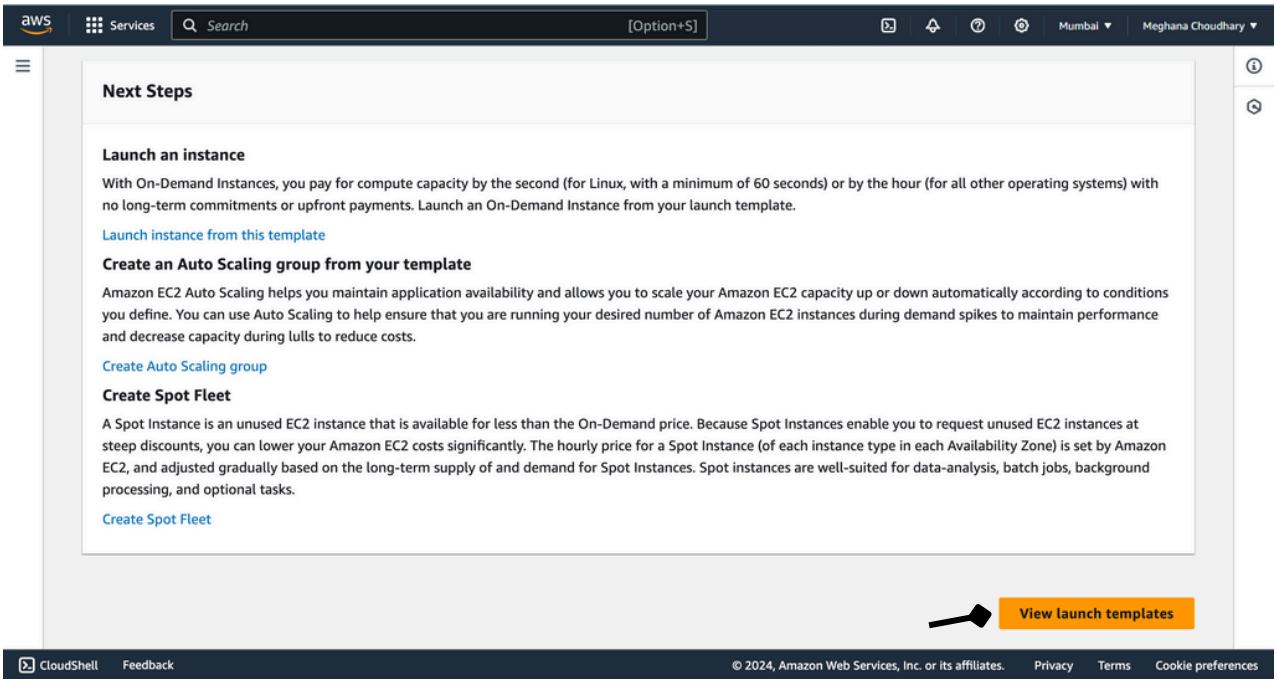
In Advanced details scroll down to the bottom and add the text to the user data. Then click on Create launch template.



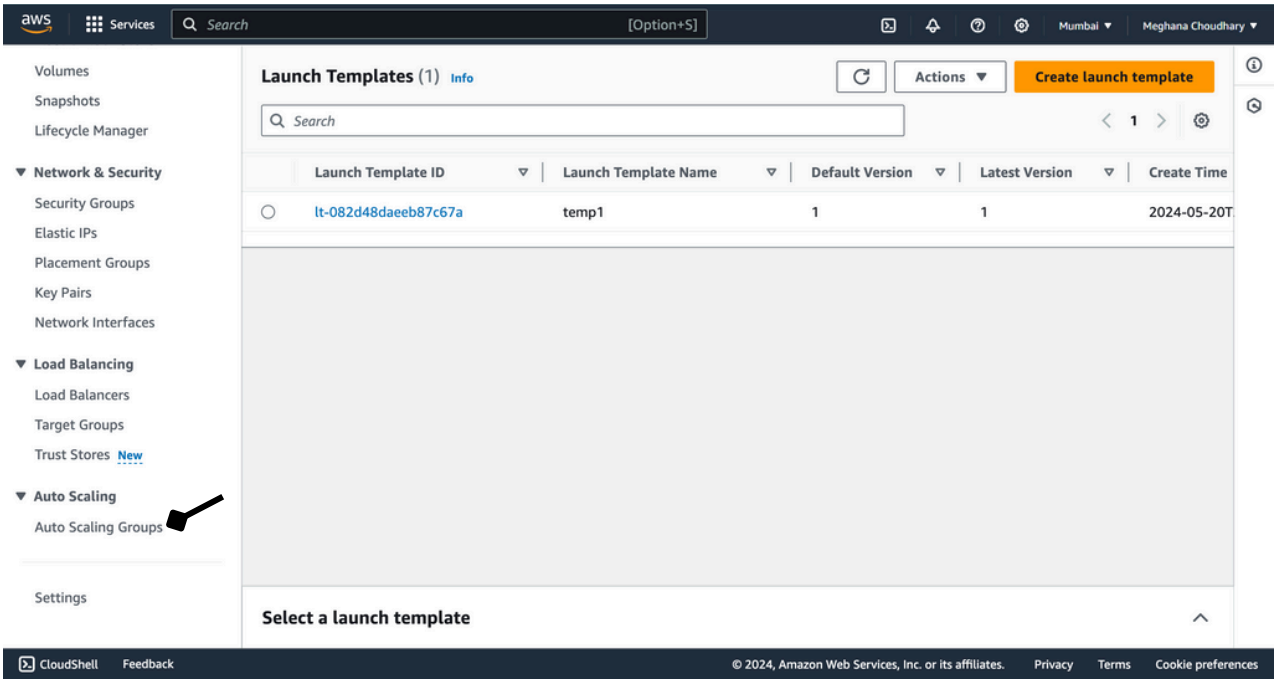
Launch Templates created successfully.



5. Scroll down and click on View launch templates.



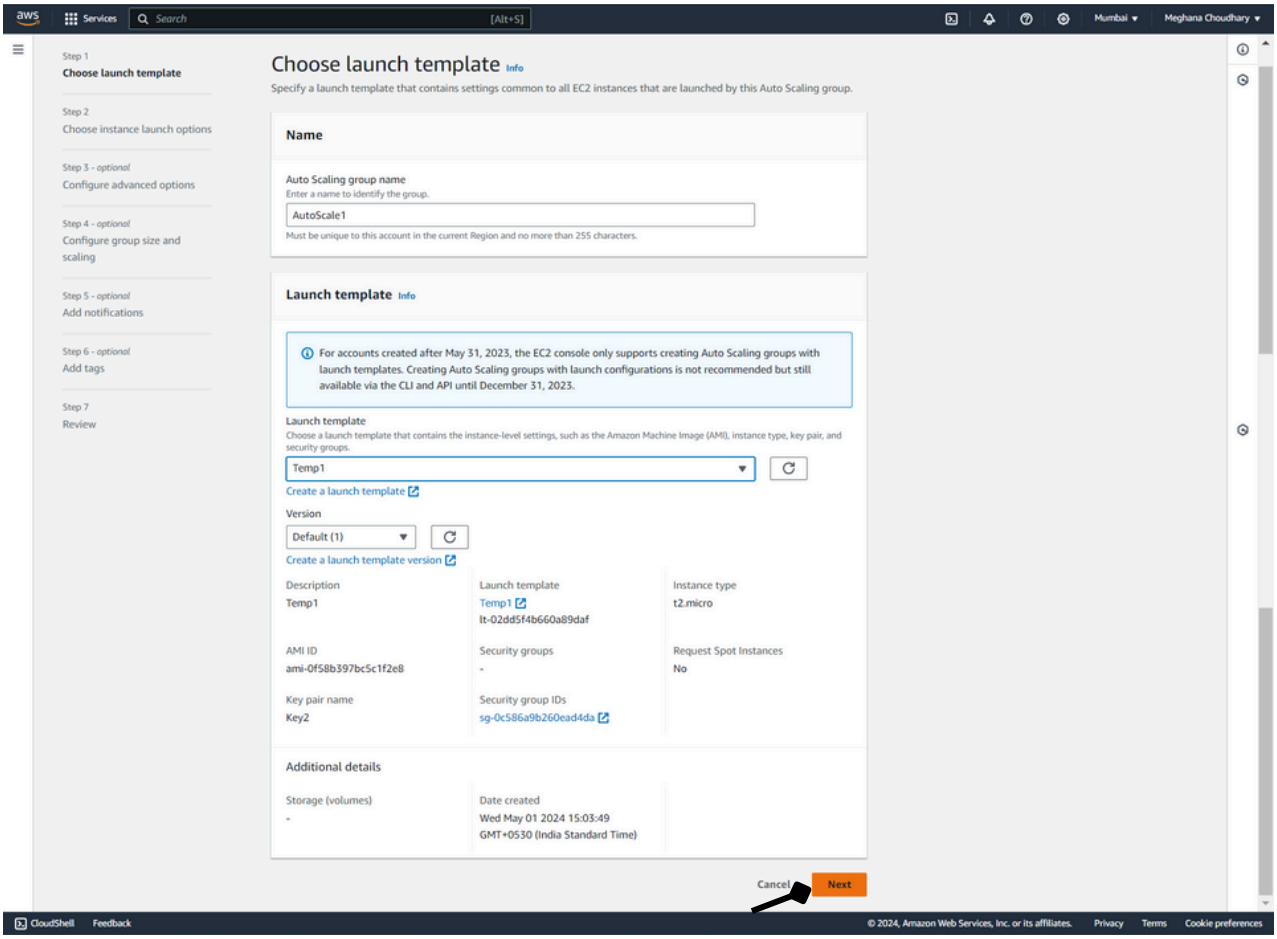
6. Now go to ‘Auto Scaling Groups’



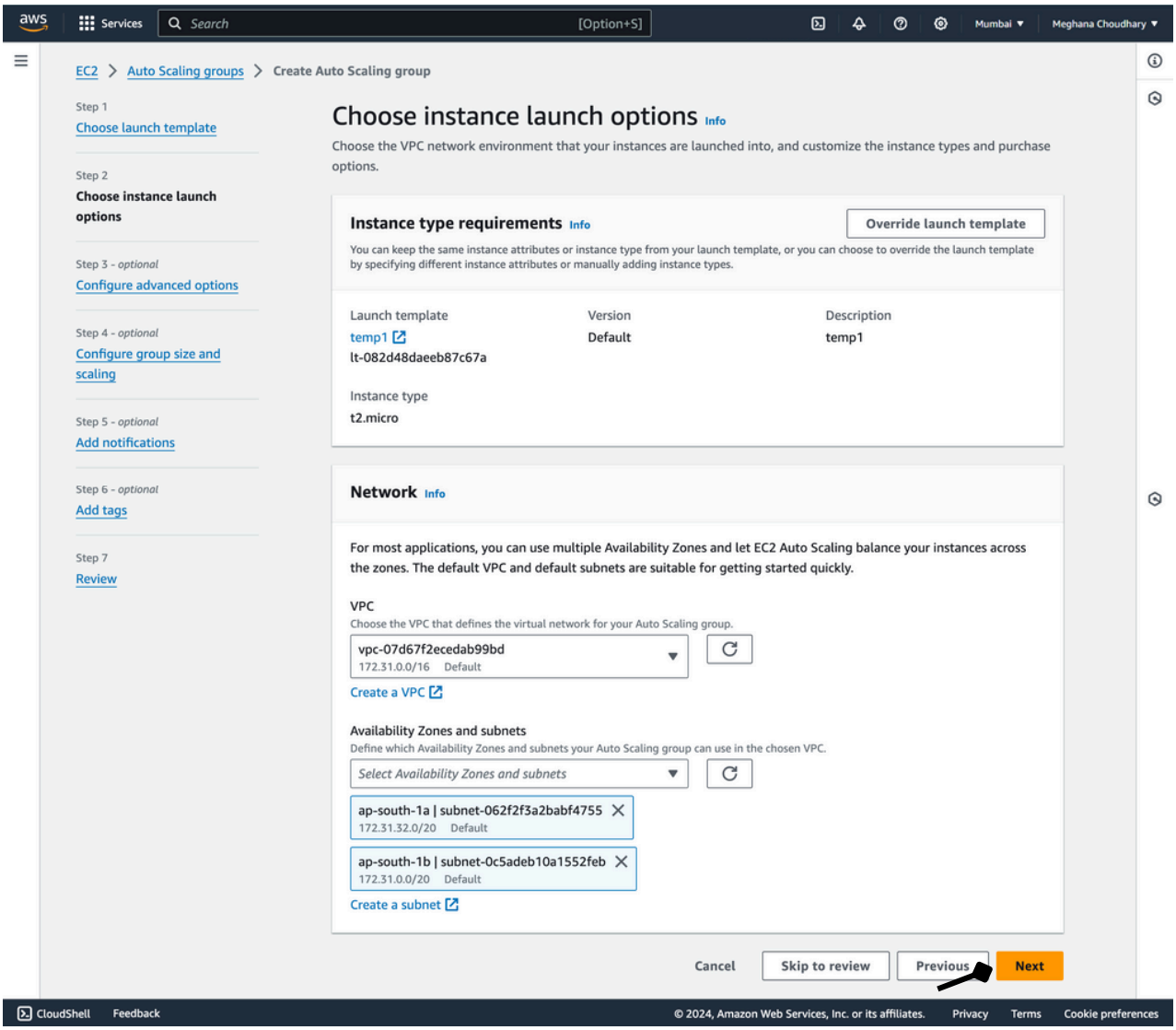
7. Add Details for the Auto Scaling Group.

Step 1: Choose launch templates- Give an name to the Auto Scaling Group and choose the Launch template you just created as the launch template.

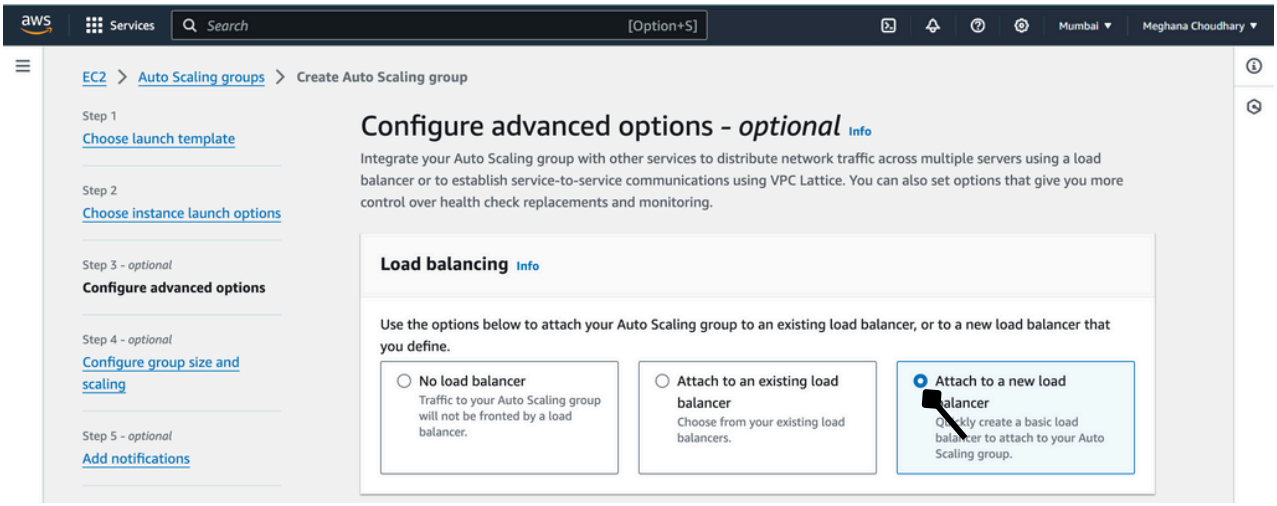
Click on Next.



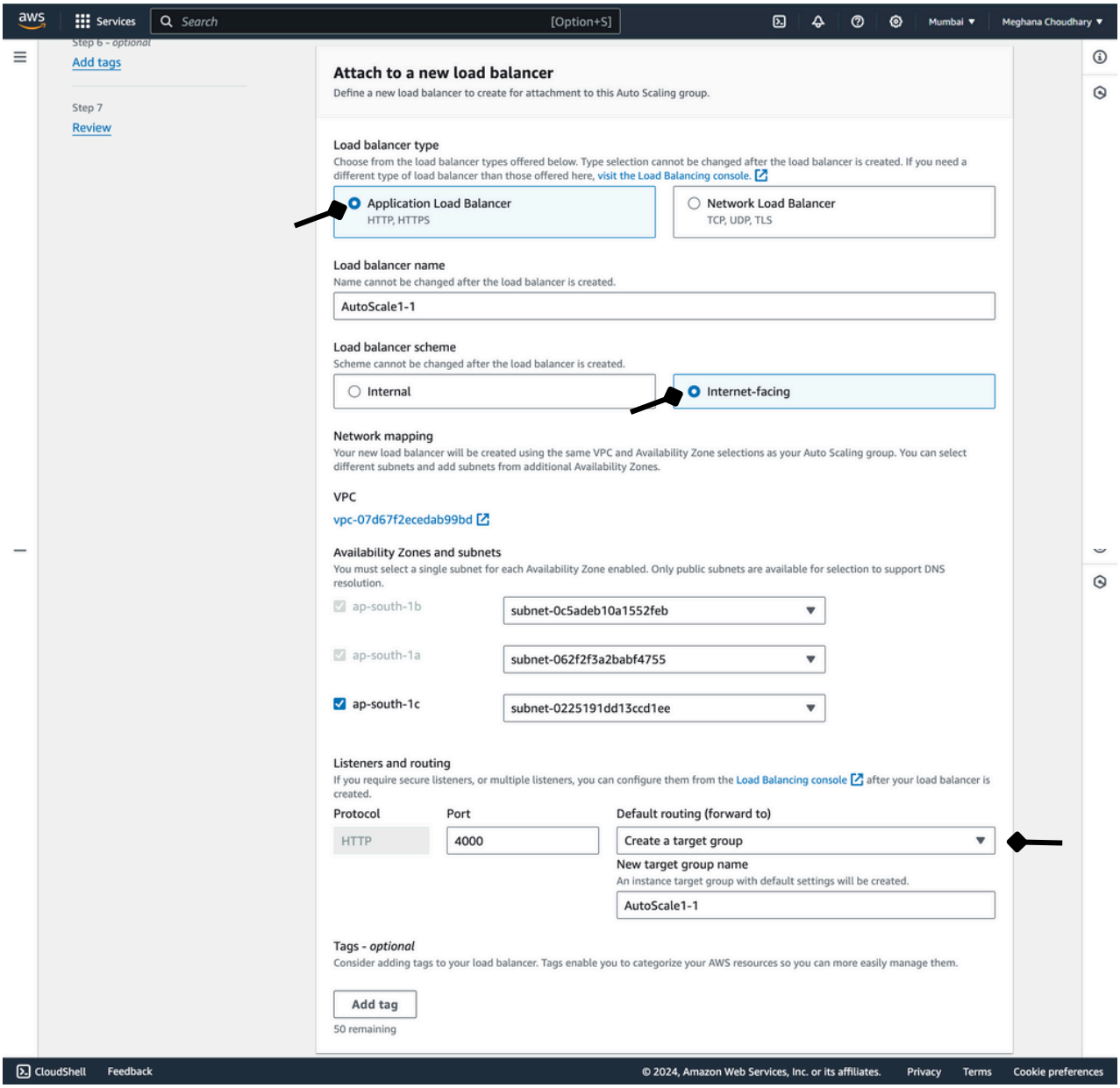
Step 2: Choose instance launch options- Choose all the ‘Availability Zones’ available.
Click on Next.



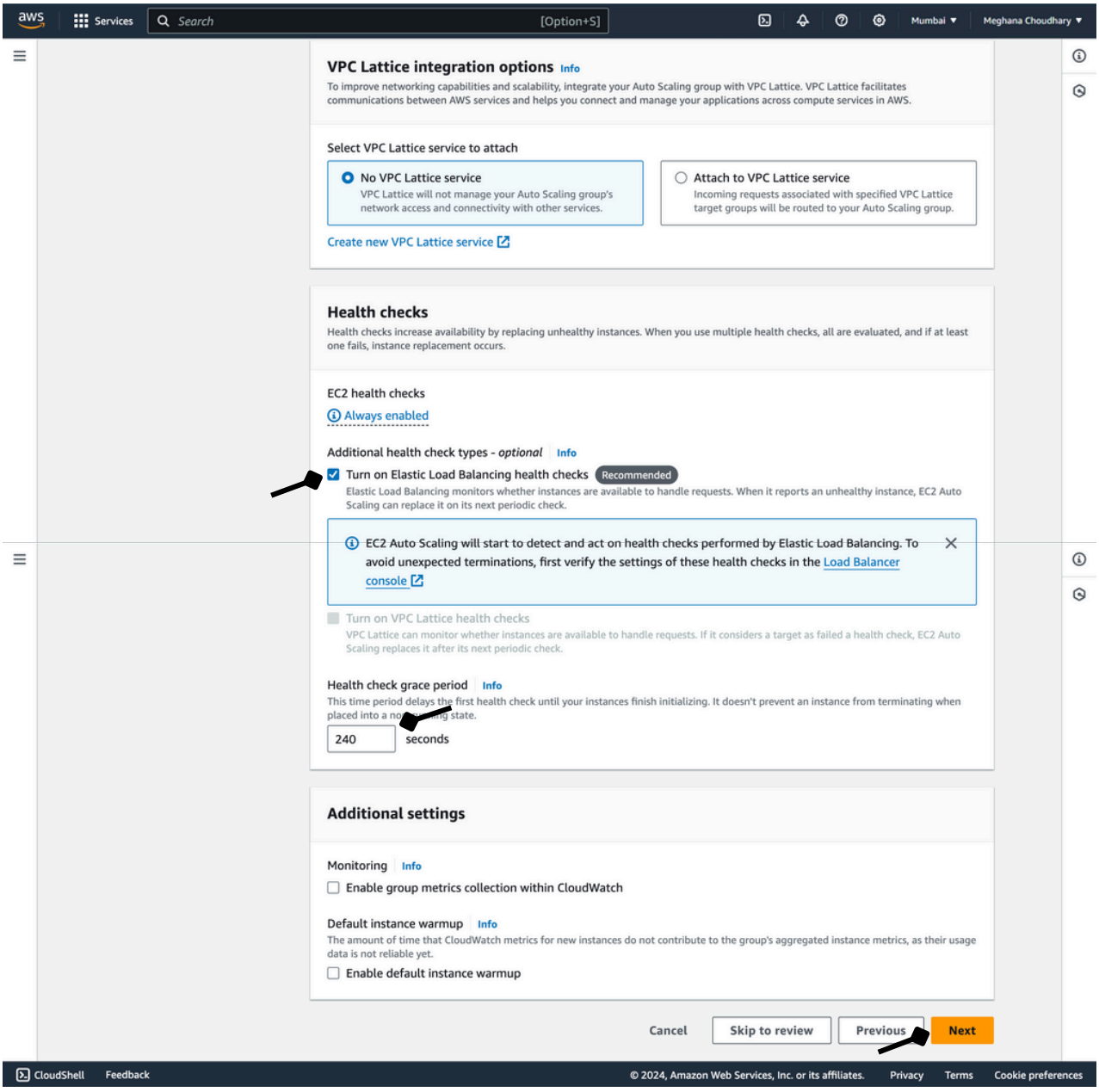
Step 3: Configure advanced options- For Load balancing select ‘Attach to a new load balancer’



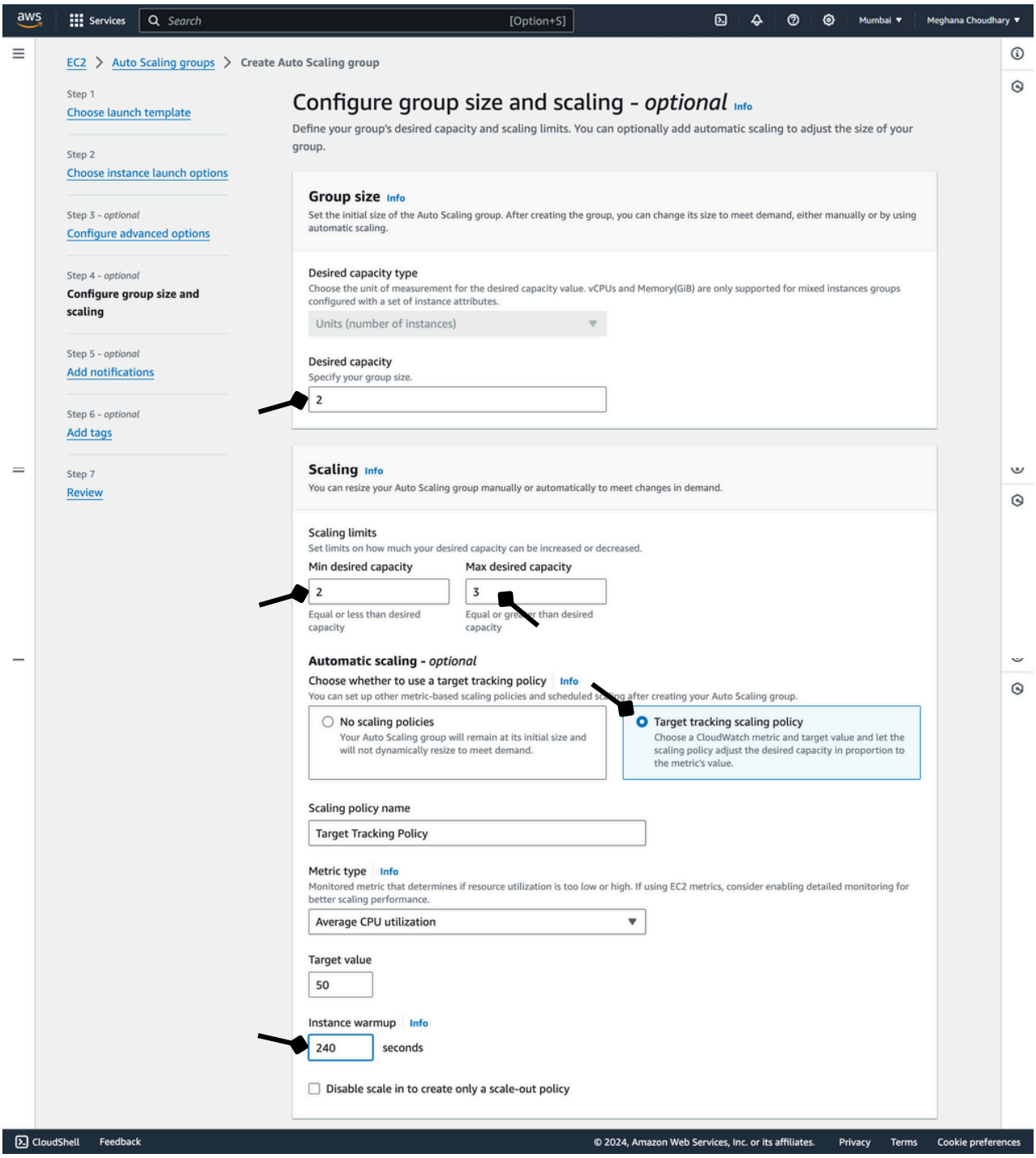
For new load balancer configuration select ‘Application Load Balancer’, ‘Internet-facing’ and make HTTP port from 80 to 4000 while creating a target group.



Health checks tick the check box for ‘Turn on Elastic Load Balancing health checks’ and make the health check grace period 240 seconds.
Click on Next.

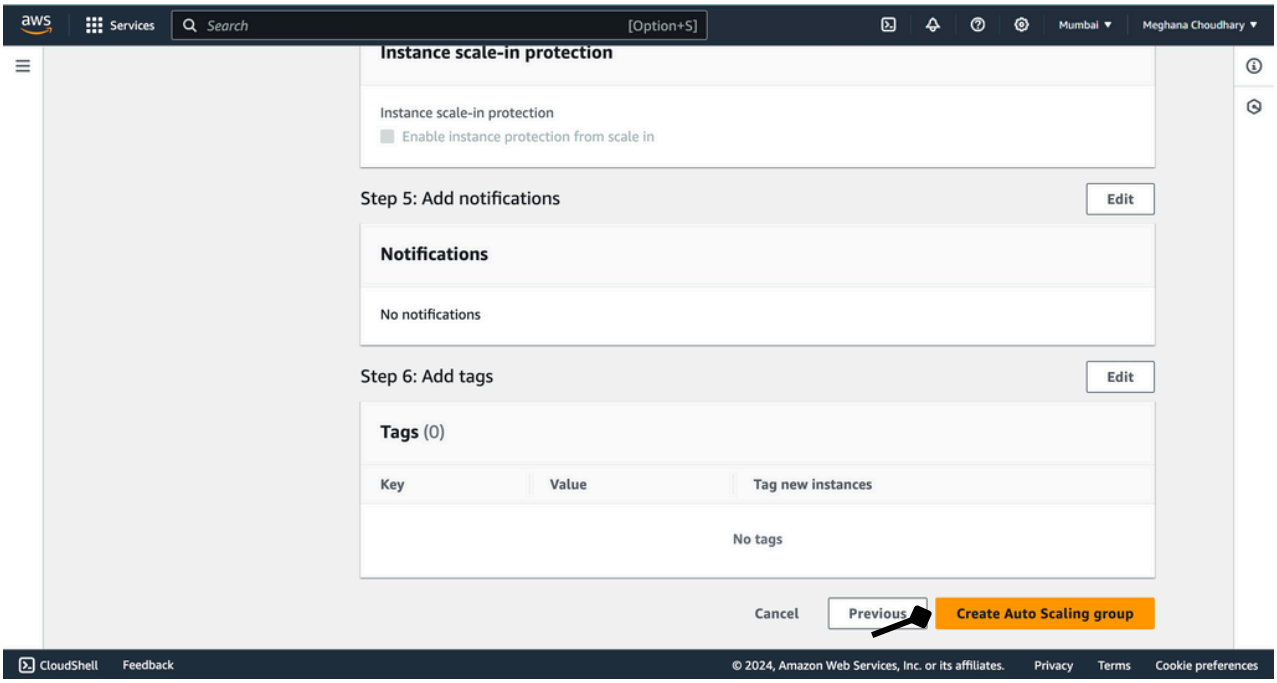


Step 4: Configure group size and scaling- In Group size set Desired capacity as 2. In Scaling set Min desired capacity as 2 and Max desired capacity as 3. In Automatic scaling choose target scaling policy and set the instance warmup as 240 second.
Keeping everything else as default, Click on Next.

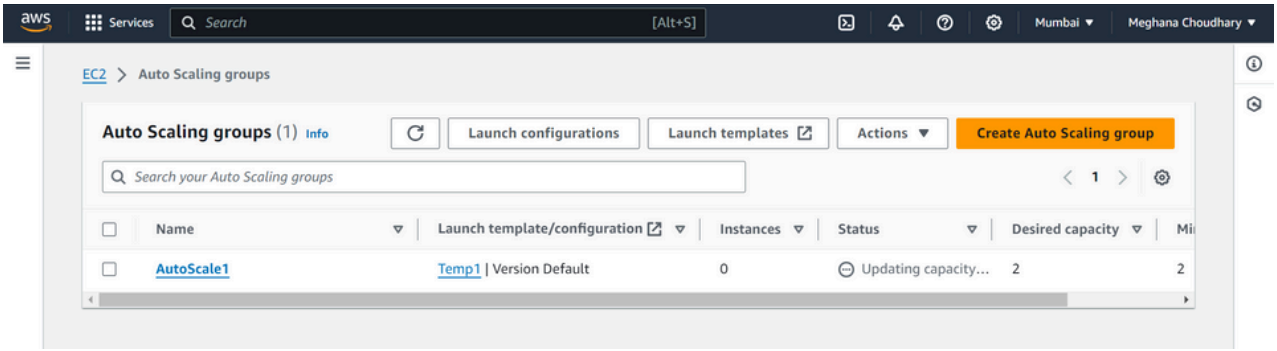


Skip through Step 5 and Step 6 by clicking Next.

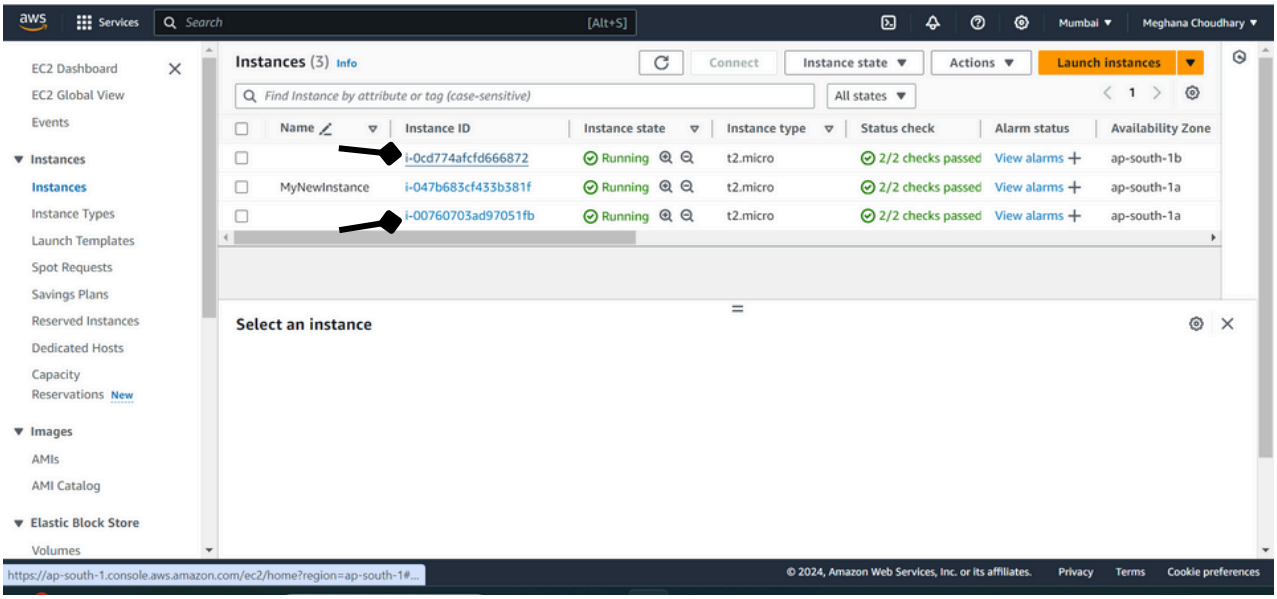
Finally in Step 7 Click on Create Auto Scaling Group.



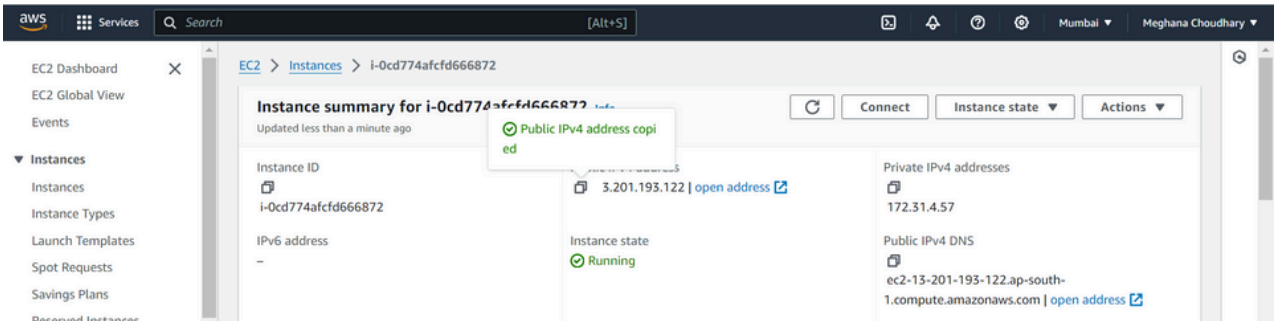
Auto Scaling Group is created.



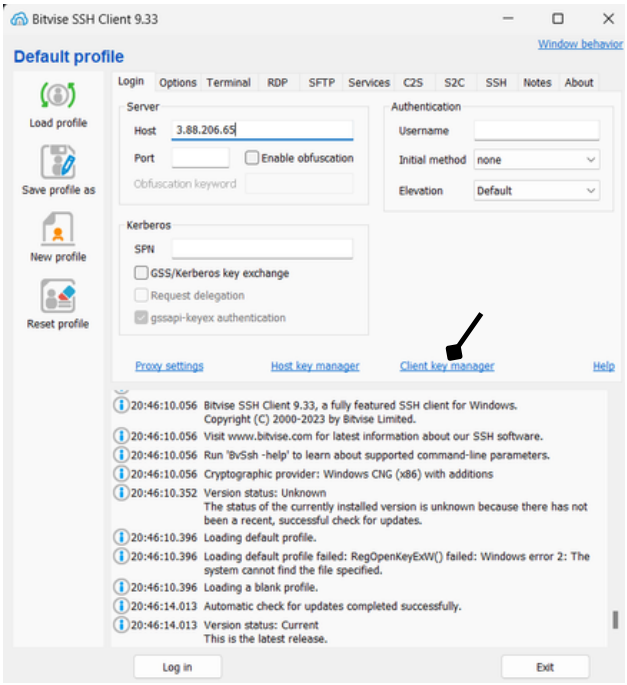
8. Going back to instances we can see two unnamed instances are created.



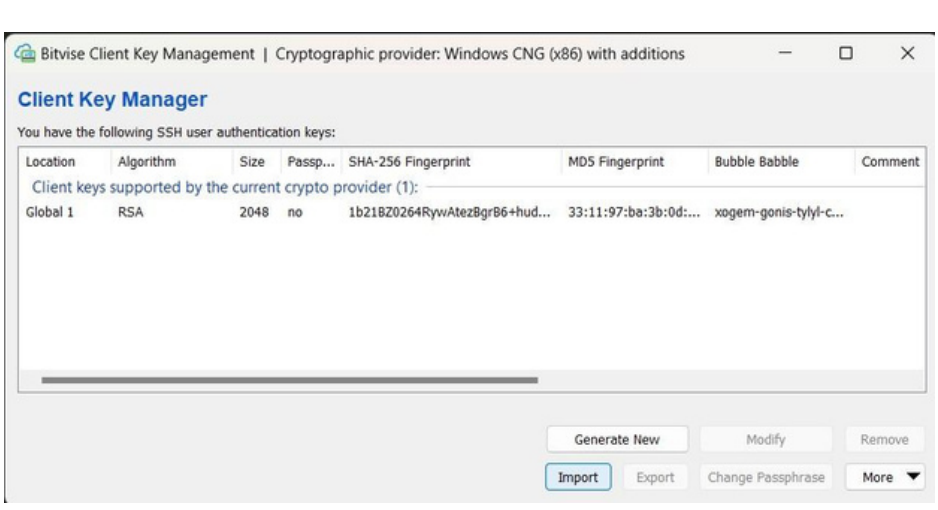
8. Copy the public IPv4 address of any one of the two instance.

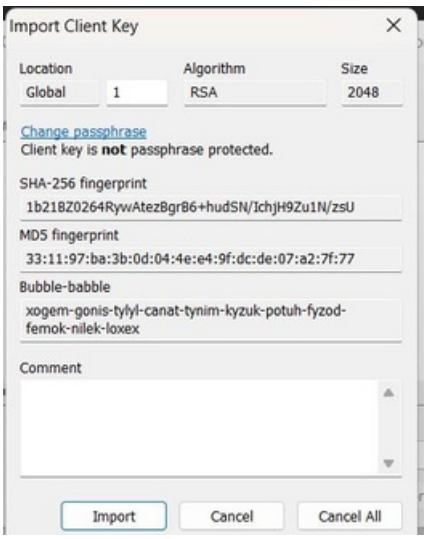


9. In Bitvise SSH Client, paste the 'Public IPV4 address' and click on 'Client key Manager'.

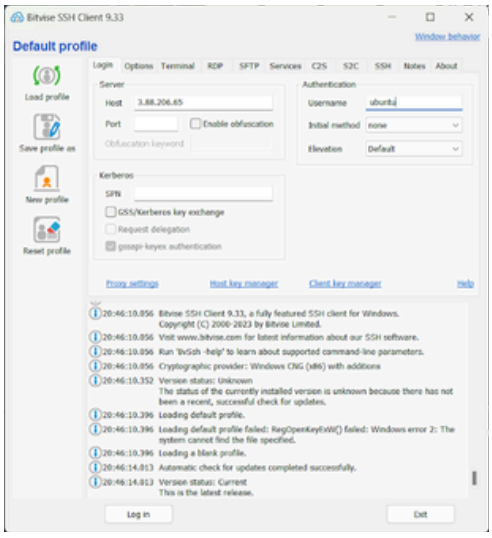


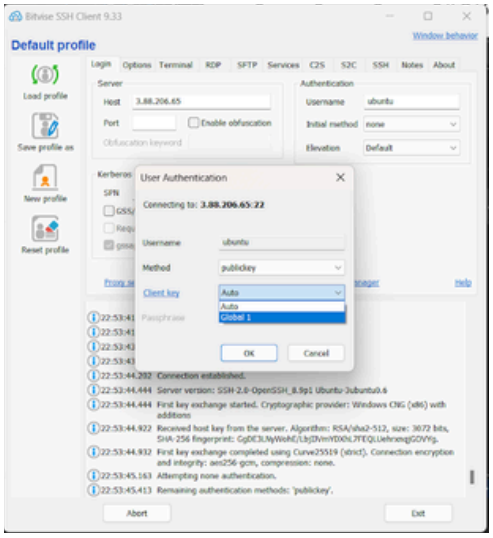
10. Under 'Client key manager', if there is any existing key then remove it and click on 'Import', select the same key that you used while creating the instance, from your file directory and then again click on 'Import' and then the key is successfully imported.

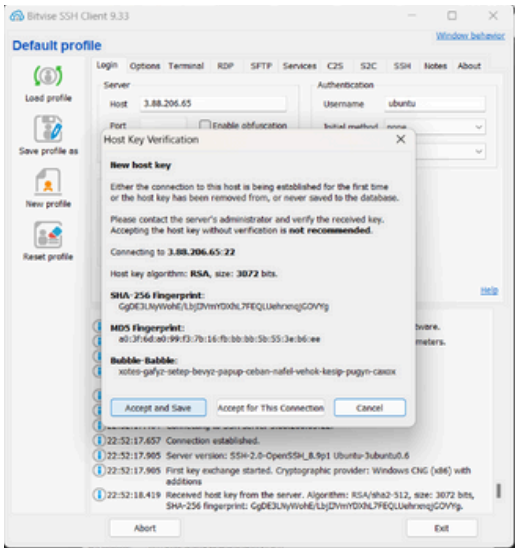




10. In ‘Bitvise SSH Client’, under ‘Default profile’ give the ‘Username’ as ‘ubuntu’ then select ‘Global1’ from ‘Client Key Manager’ then click on ‘Log in’ & ‘Accept and save’.







If the ‘Log In’ button changes to ‘Log Out’, then you are logged in successfully.

11. Click on ‘New terminal consol’, to access the terminal of your instance.

8. Once you are inside the terminal of your instance

To create a infinitely running bash script

```
vim infi.sh
```

```
ubuntu@ip-172-31-12-249:~$ vim infi.sh
```

In the bash script, add the following code.

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

```
#!/bin/bash

while(true)
do
        echo "InsideLoop"
done
```

To make the script executable

```
sudo chmod 777 infi.sh
```

```
ubuntu@ip-172-31-5-108:~$ sudo chmod 777 infi.sh
```

Run the script

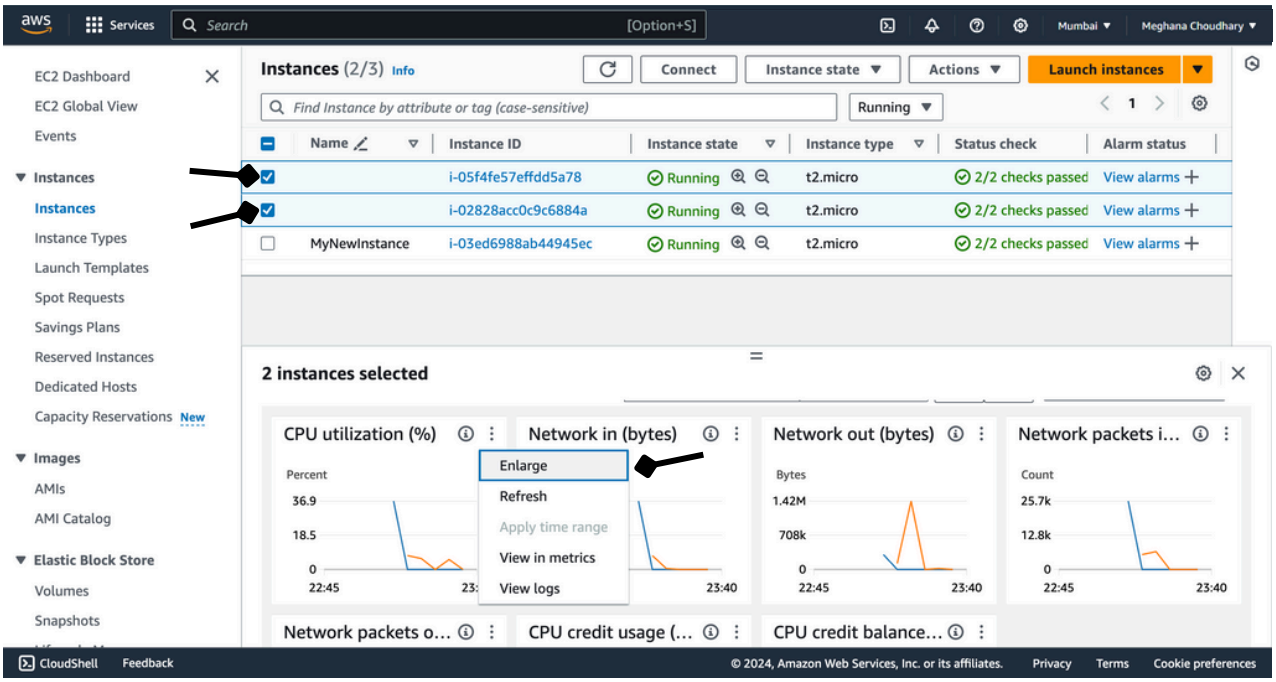
```
sh infi.sh
```

```
ubuntu@ip-172-31-5-108:~$ sh infi.sh
```

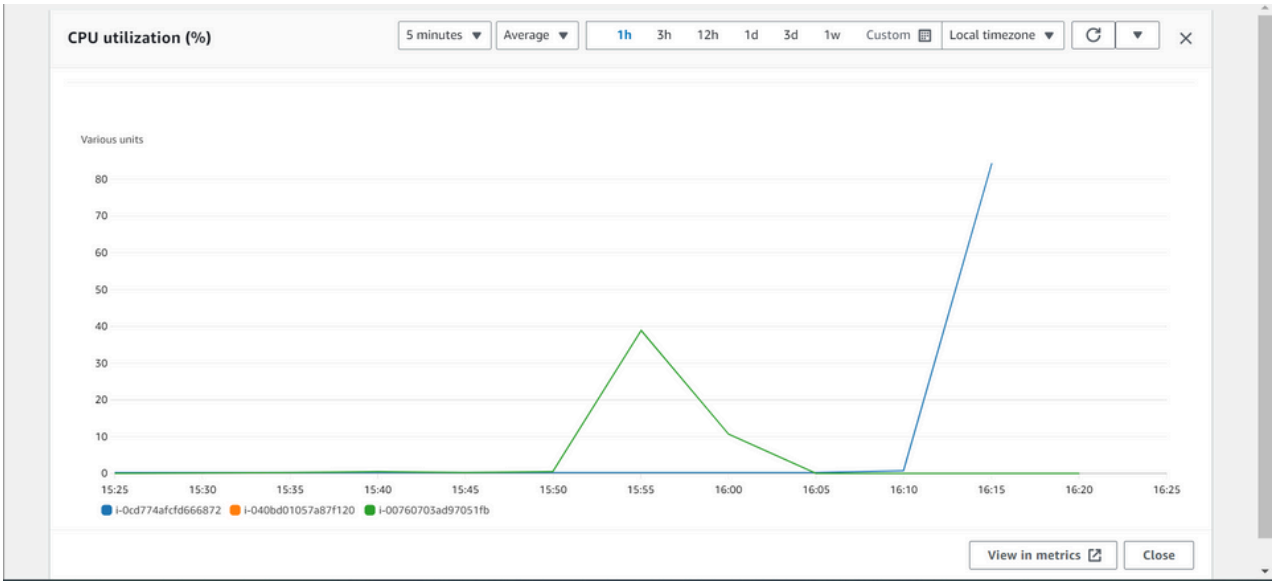
9. An infinite loop starts to run, stressing the cpu of the instance.

```
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
InsideLoop
```

10. Selecting the two instances we can take a look at the cpu utilisation graph.



11. After a while the instance reaches and surpasses it's 50% CPU utilisation threshold.



12. So to take that load off the instance, the auto scaler spins up another instance to balance the load.