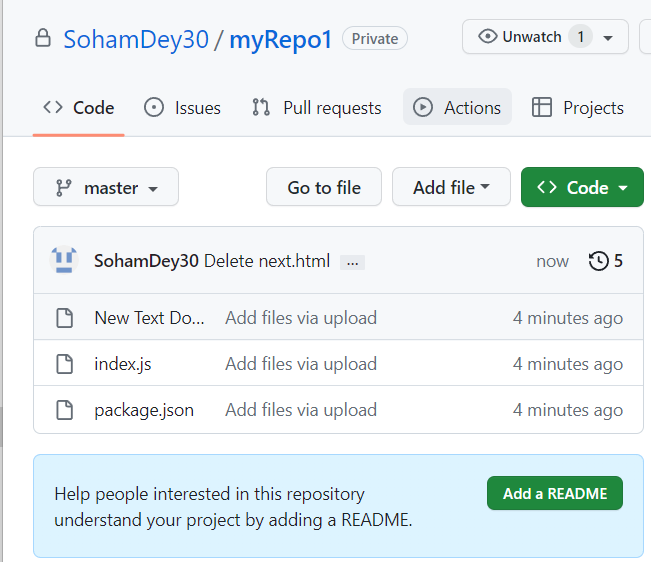
ASSIGNMENT – 11

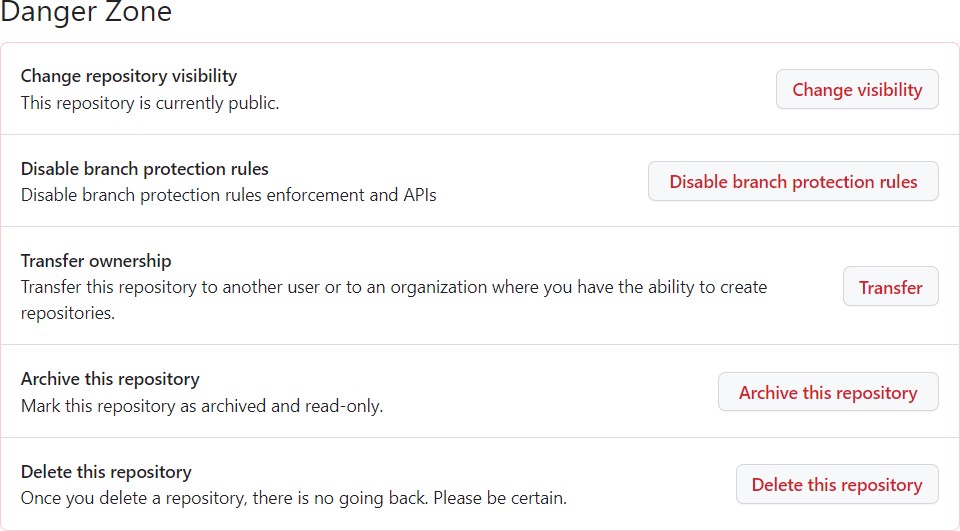
**Problem Statement**: *Build Scaling plans in AWS that balance load on different EC2 instances.*

**Step 1:** Sign-in to GitHub. Make sure the Repository which will be cloned is made public or not.

* For This, select the “settings” tab of your repository.



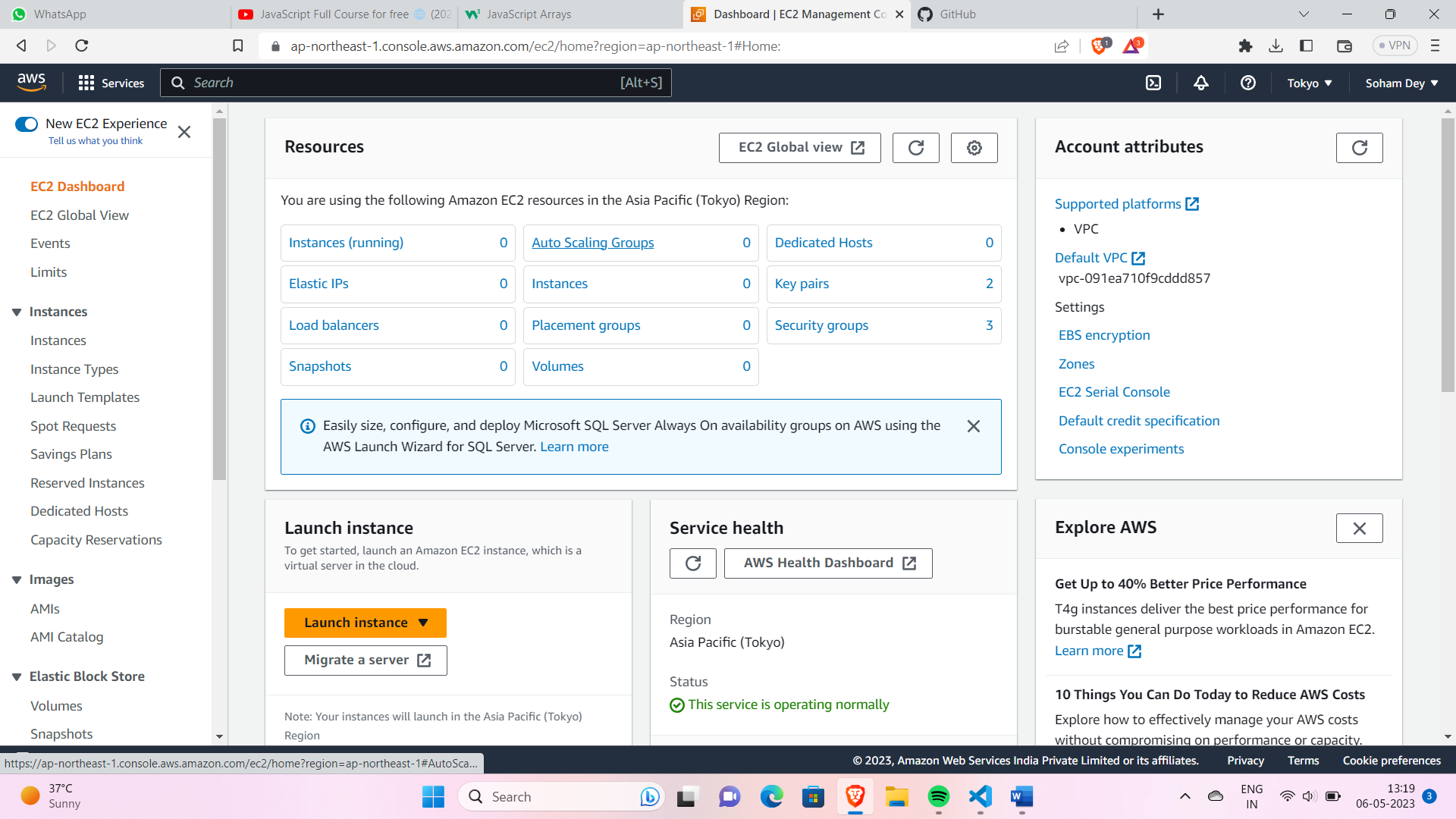
* Next Scroll Down until you reach the danger-zone section.



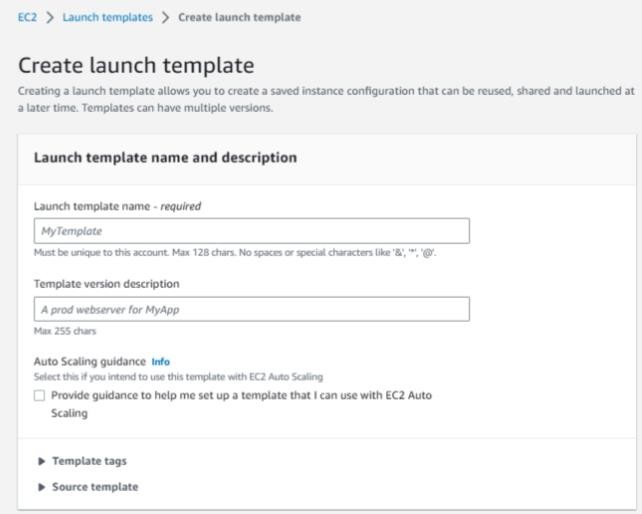
* In here, check The Change Repository visibility option. Here, we can see the repository is currently Public.
* If it shows Private then click on the Change Visibility option and follow the on- screen Instructions to make the repository Public.

**Step 2:** Now in another tab open AWS and Sign-in to your console. Now go to your EC2 dashboard.

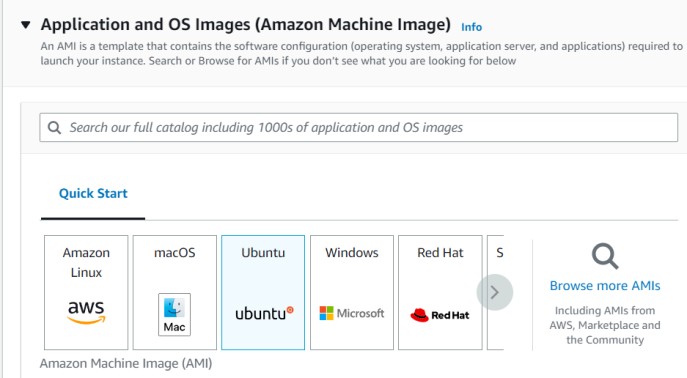
From the Left Side Nav-bar Go to your Instances Section. Under it Click the launch templates button.



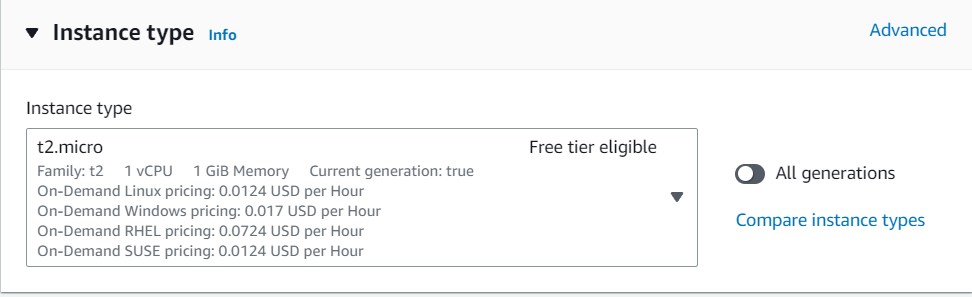
**Step 3:** Now click on the Create Launch Templates button. Give a name and description for your EC2 template you are about to create. Here we gave the same for both the fields. Next, Check the “Provide Guidance” box.



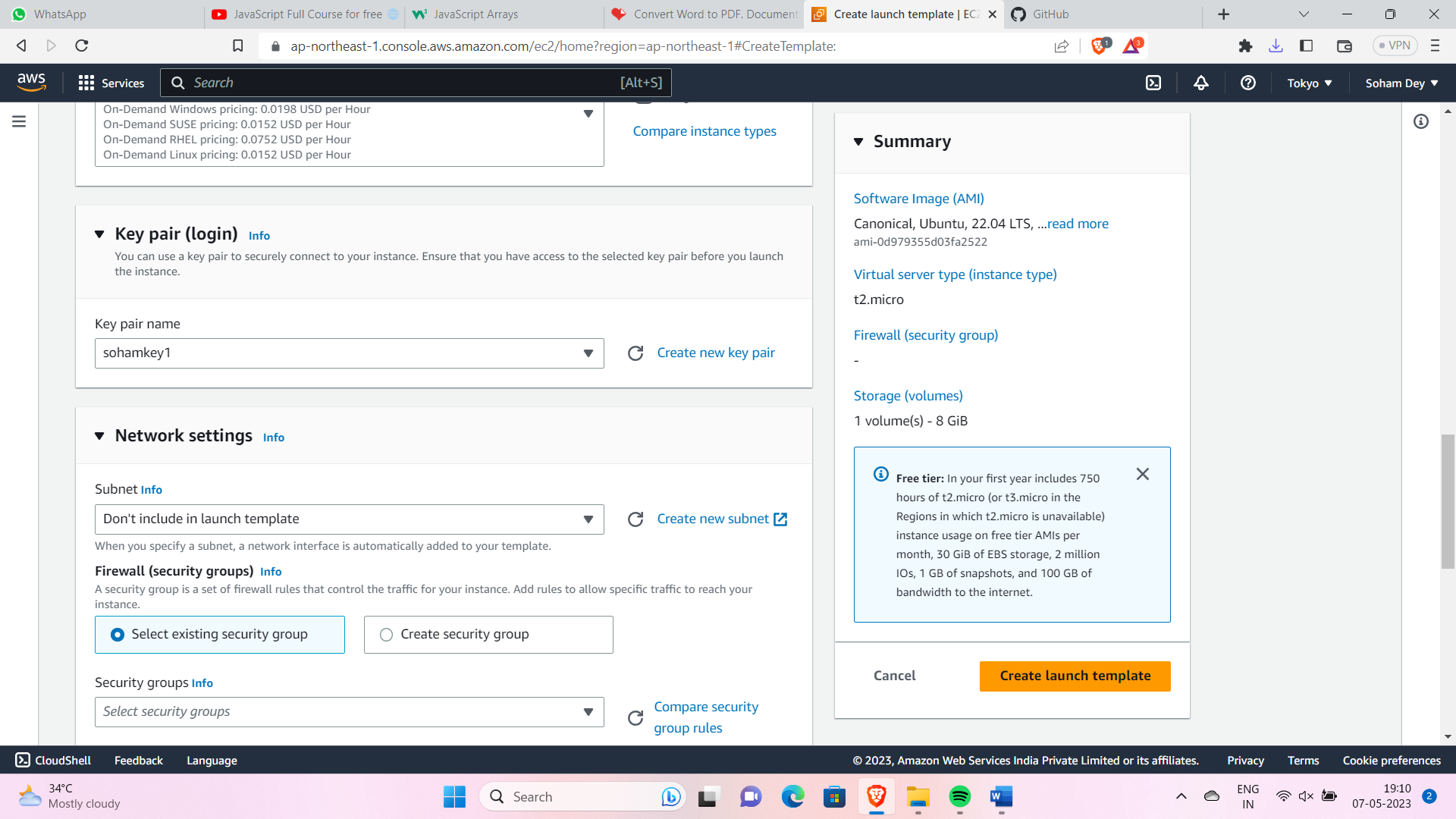
**Step 4:** Next, under the quick start menu select Ubuntu as the OS.



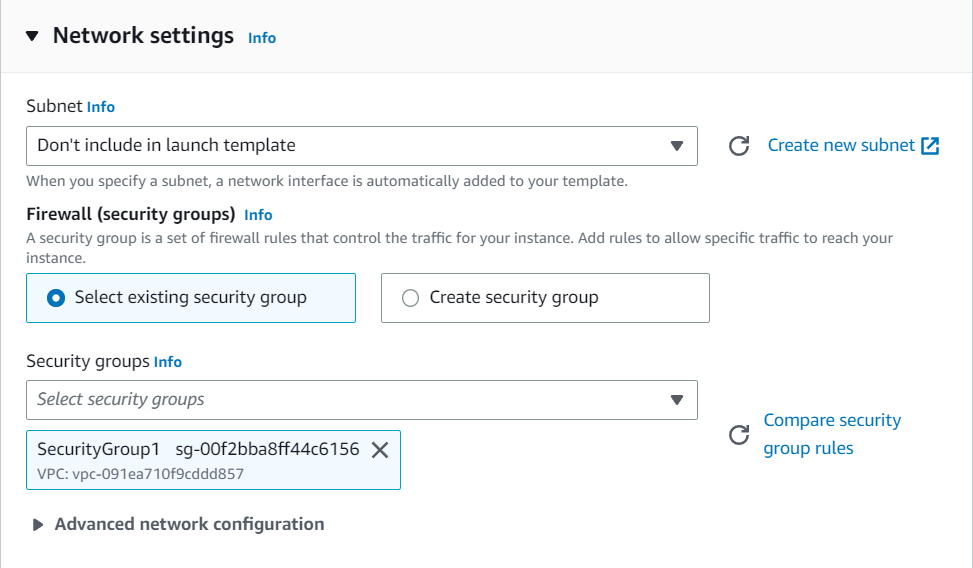
**Step 5:** Under Instance type select t2.micro type of configuration.



**Step 6:** Select Existing Key-Pair and Security Group and if not applicable then Generate or Create a Key-Pair or Security Group wherever required.

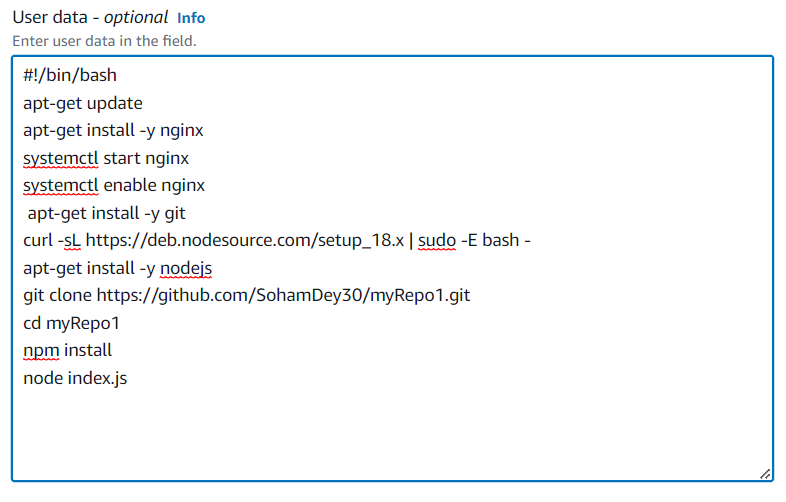


**Step 7:** Now, Click on the Advanced Group Section at the bottom. Scroll Down to User Data Section and paste the following commands in the provided box. Then click on the Create Launch Template button.



Template Created:-





**#!/bin/bash apt-get update**

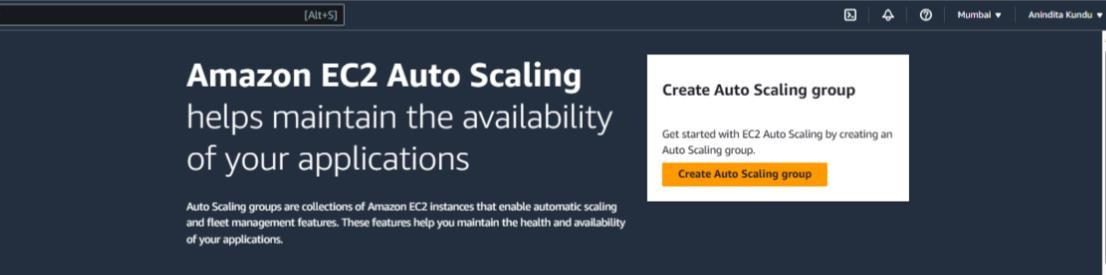
**apt-get install -y nginx systemctl start nginx systemctl enable nginx apt-get install -y git**

**curl -sL https://deb.nodesource.com/setup\_18.x | sudo -E bash - apt-get install -y nodejs**

**git clone YourRepositoryURLhere cd YourRepositoryNamehere/ npm install**

**node index.js**

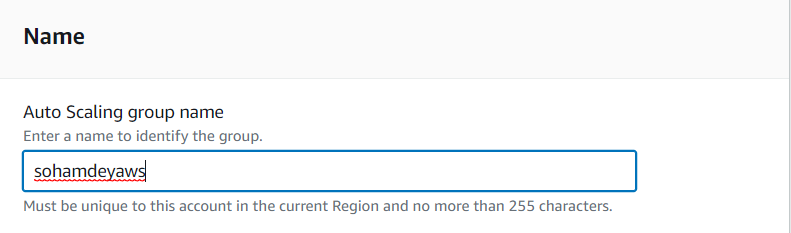
**Step 8**: Now from the Left side Nav Bar go to Auto Scaling Groups under Auto-Scaling section. Then click on Create Auto Scaling Group.

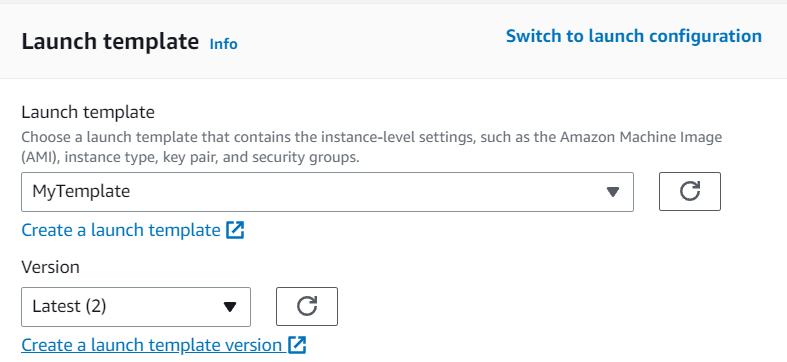


**Step 9:** Now, Give a unique name to your new Auto Scaling Group. Also select the Launch Template that we recently created by using the drop-down menu under Launch

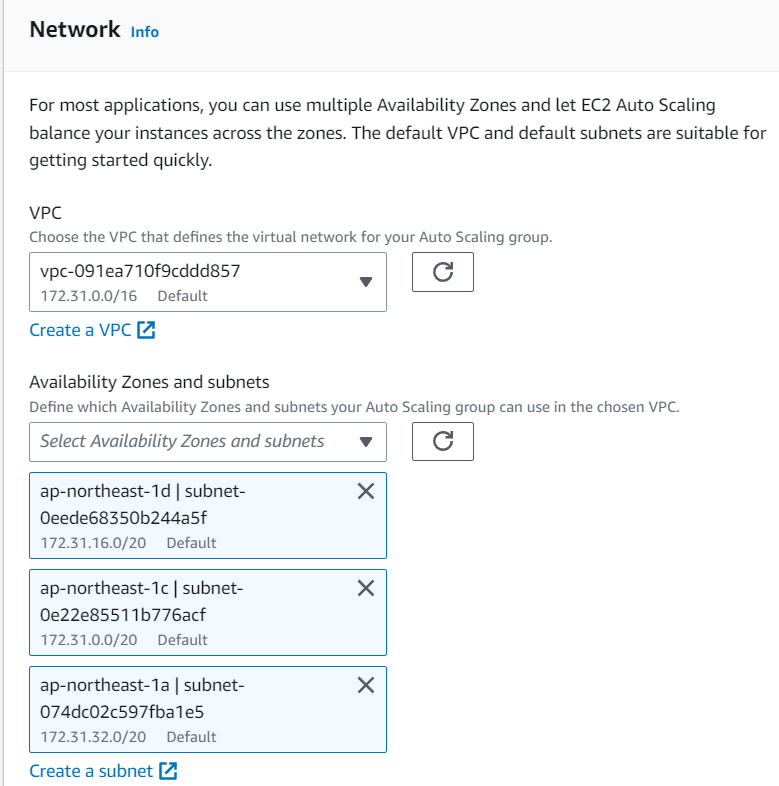
Templates section.

* Under the selected Launch Template click on the version option and select Latest.

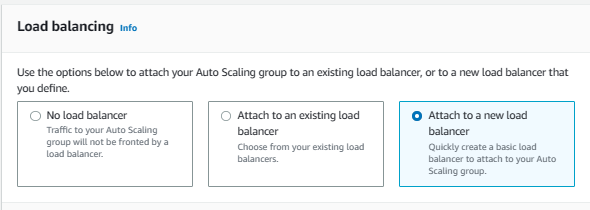




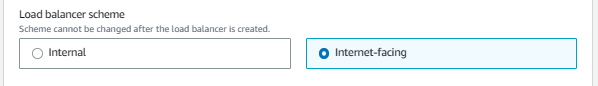
**Step 10:** Now click on the Next button. After that, Under Availability Zones and Subnets select all the zones that appear.



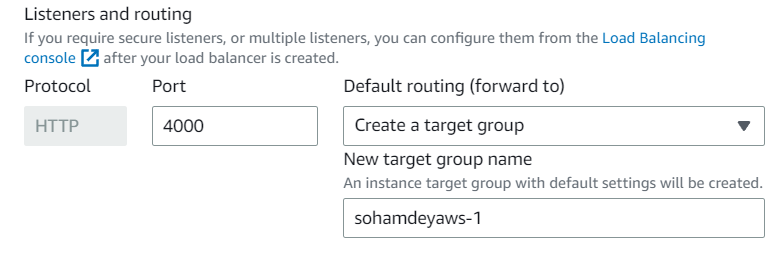
**Step 11:** Again, click on the Next button. Now Under Load Balancers select the Attach to a New Load balancer option.



**Step 12:** Now select Internet-Facing under Load balancer scheme.

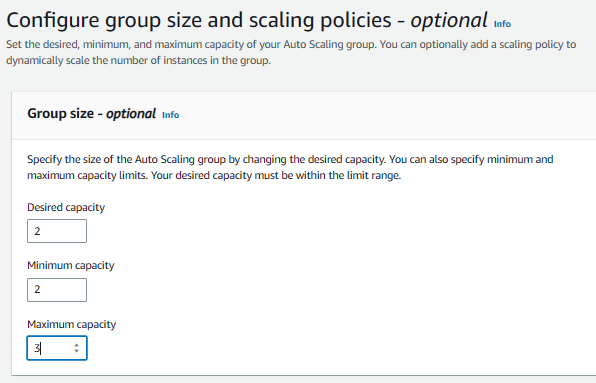


**Step 13:** Under Listeners and Routing enter the port no. of the project and select Create target group followed by giving the target group a name.



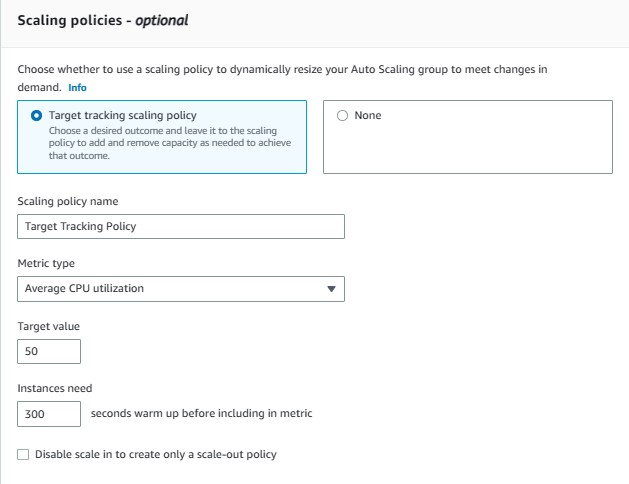
**Step 14:** Now click on the next button. After clicking on the Next button, a new page will open. Under Group Size mention:

* Desired Capacity = 2
* Minimum Capacity = 2
* Maximum Capacity = 3

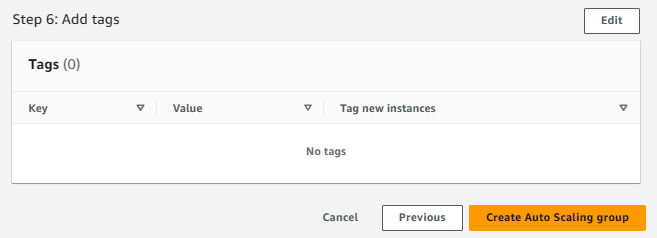


**Step 15:** Now under Scaling policies Choose the Target Tracking Scaling policy option.

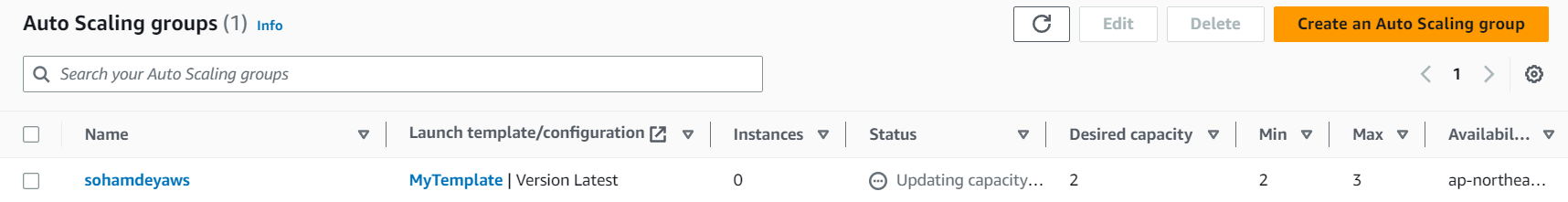
* Select the metric type as Average CPU utilization.
* Set Target Value to 50.
* Set Warm-Up time to 300 seconds under Instances Need.



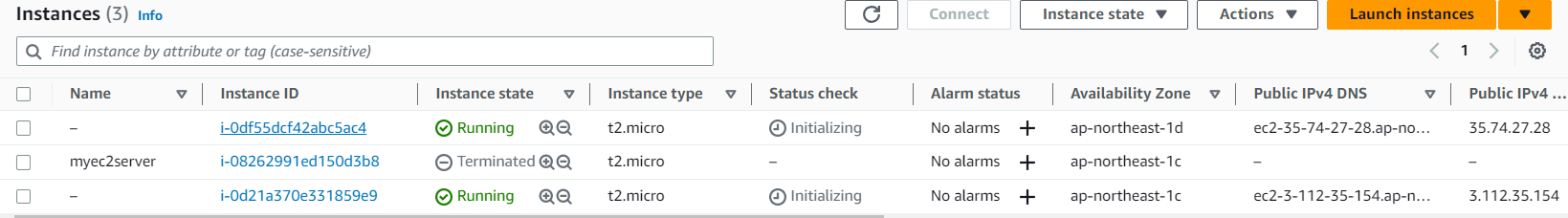
**Step 16:** Then click on next. Nothing to do in Notifications page. So again, click on the Next button. No tags needed. Again, click on the Next button. Now Review your Auto- Scaling Group you are going to create. Now click on the Create Auto-Scaling Group button.



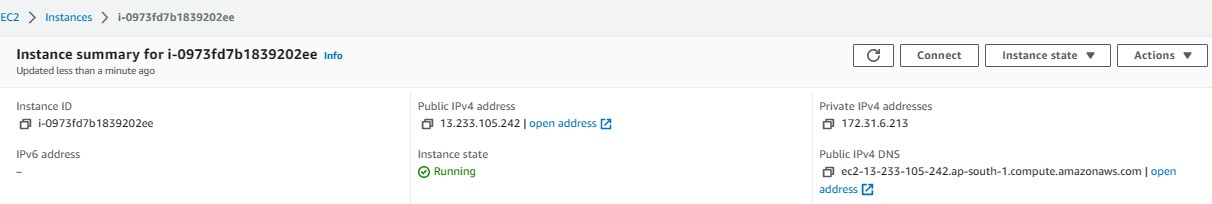
**Step 17:** Now we can go to the Auto Scaling Groups section and find our newly created Auto Scaling Group.



**Step 18:** Return to the Instances Page using the Left side Nav bar.



**Step 19:** Click on the First instance. Copy its public IPV4 DNS.

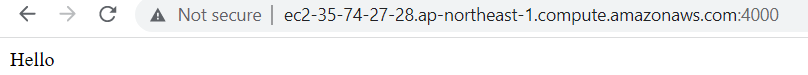


**Step 20:** Paste it in another browser.



We can successfully access the webpage.

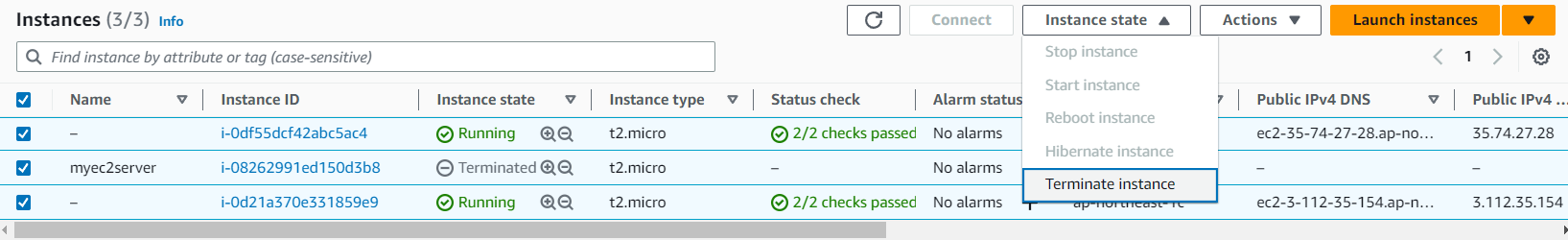
**Step 21:** Now to access our project webpage we need to append the port no. (4000) of our project with a “:”



To test 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.

**Step 22:** We will now **CRASH THE SERVER INSTANCES** by terminating them. Go to the instances page. Select the server instances.

Now click on the Instance State button up top. From the drop-down select the Terminate instance option.



**Step 23:** Refresh the instances the page from time to time. After few seconds the instances will get terminated.

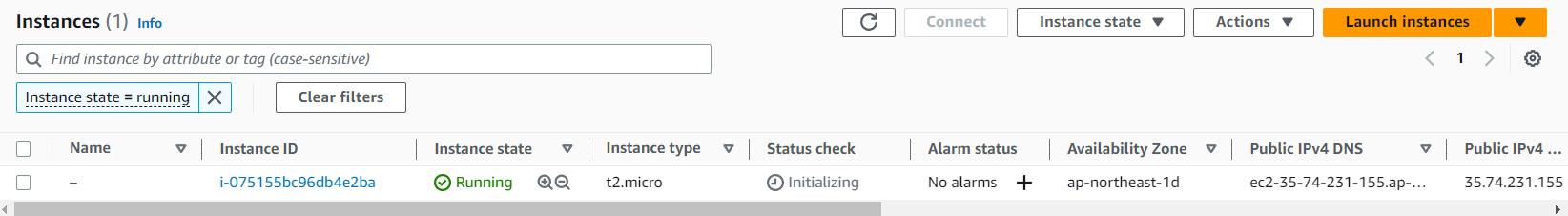
Wait for some time. Keep refreshing using the refresh button on the left side of Instance state button.

After some you will notice a new instance server will appear automatically! To help finding it more easily we need to activate the instance running filter.

Click on the search box below the Instances Section Heading. Start typing running.

Select the option “Instance state = running option in the suggestion dropdown. The filter will be activated.

After some few seconds of refreshing we will be able to see two new servers are running.



Now again copy paste the new public IPv4 DNS of the first instance and port no. in the other browser to see if the instances are working. It will be working. So, our Auto-Scaling Group can handle instance crashing by providing new fresh instances.

We will now OVERLOAD THE SERVER INSTANCES by running scripts and increasing CPU utilization value above the threshold that we specified during Configuration of the Auto scaling group.

**Step 24:** For it we will use:

* Use Bitvise SSH client for instance 1.
* Use direct connect terminal in AWS for instance 2.

**Step 25:** For Instance-1:

* Copy the public IPv4 address
* Open Bitvise SSH client.
* Paste the IP and select/specify the necessary options. (Refer Ass7)
* Now Log-In to your server.
* Open the new Terminal.
* Now enter the command:

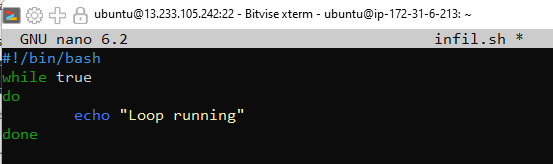
# nano infil.sh

* After the command a new nano Editor window will open. Type the following in it.\

**#!/bin/bash while true do**

**echo “Loop running”**

**done**



* Now, to save and close the shell script we need to press the following shortcuts and keys sequentially:

# Ctrl+X Y

**Enter**

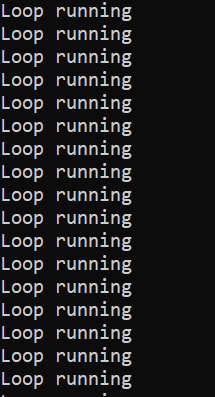
* Now you will be returned back to the terminal.
* Now type the following commands:

# chmod +x infil.sh



* + **./infil.sh**

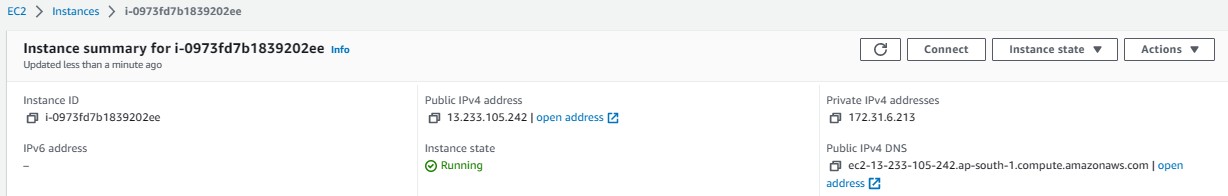
(Used to execute the infil.sh script)



* Now the script will start running infinitely!
* Do not close the terminal. Keep it minimized.

**Step 26:** For Instance-2:

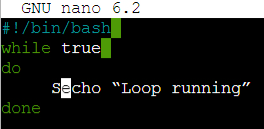
Click on the instance 2. Now click on the connect button.



# nano infil.sh

# 

* After the command a new nano Editor window will open. Type the following in it.



**#!/bin/bash while true do**

**echo “Loop running”**

**done**

* Now, to save and close the shell script we need to press the following shortcuts and keys sequentially:

# Ctrl+X Y

**Enter**

* Now you will be returned back to the terminal.
* Now type the following commands:

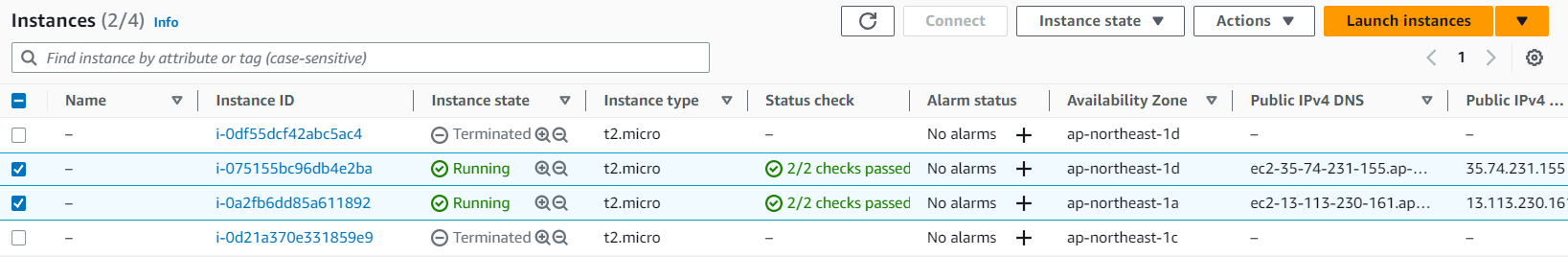
# chmod +x infil.sh

* + **./infil.sh**

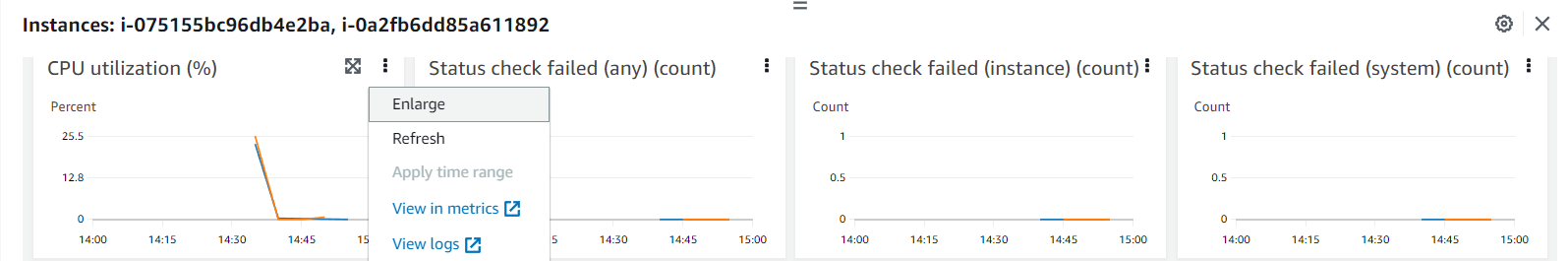
(Used to execute the infil.sh script)

* Now the script will start running infinitely!
* Do not close the terminal. Go back to the previous tab to keep working in AWS.

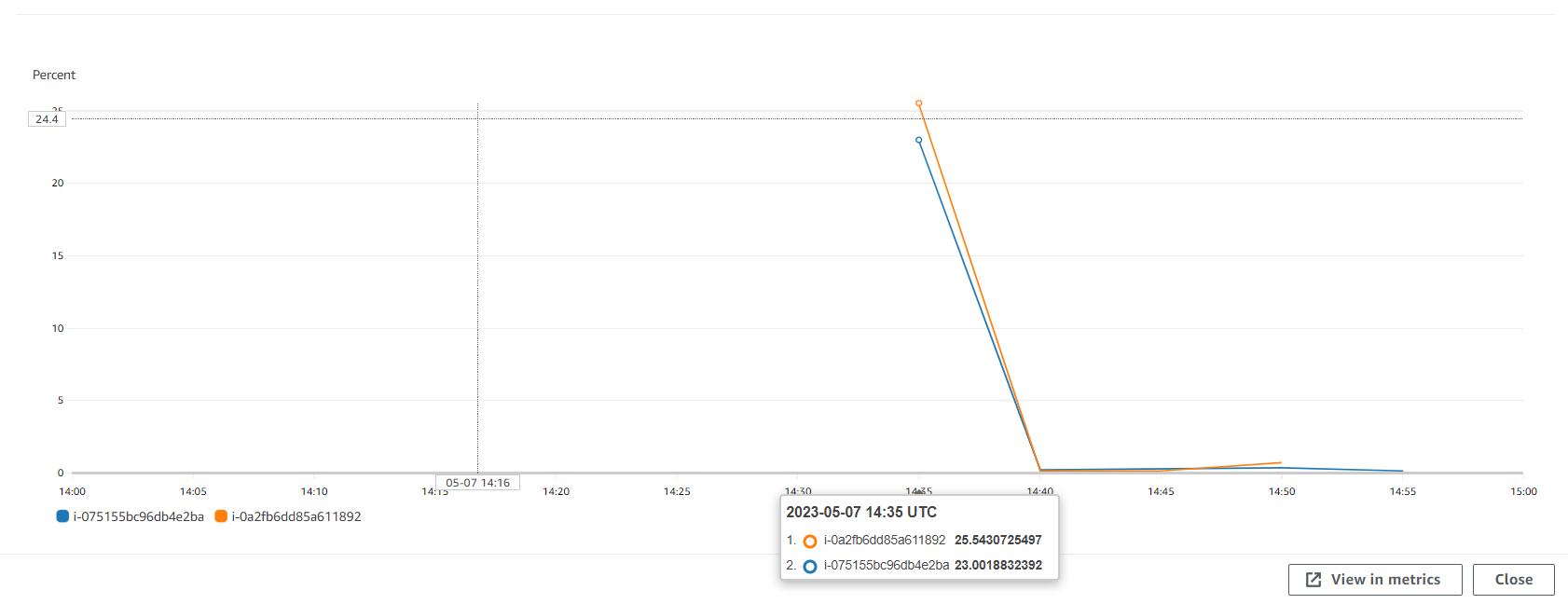
**Step 27:** Now go to the instances page. Select both the instances.



**Step 28:** Click on the instances white bar at the bottom of the page. Now drag the two bars to expand the view. We are interested only in the CPU utilization graph. Click on the maximize icon by hovering over the graph as shown in the fig to maximize this graph.



**Step 29:** Our 1st instance has already crossed over 50% utilization. That’s why we can see already a new third instance has been initiated by our auto-scaling group to compensate for the overload.



**Step 30:** There can only be 3 servers running at a time for us as specified in our Auto- Scaling group when we were creating it. Hence, we have reached our maximum limit of instances running concurrently.

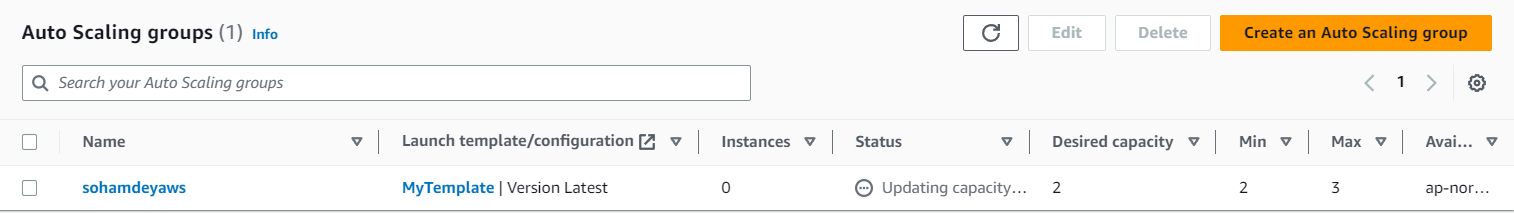
Hence, our Auto-Scaling Group can handle instance overloading by providing new instances to handle the overloading.

***Project webpage was not at all disconnected.***

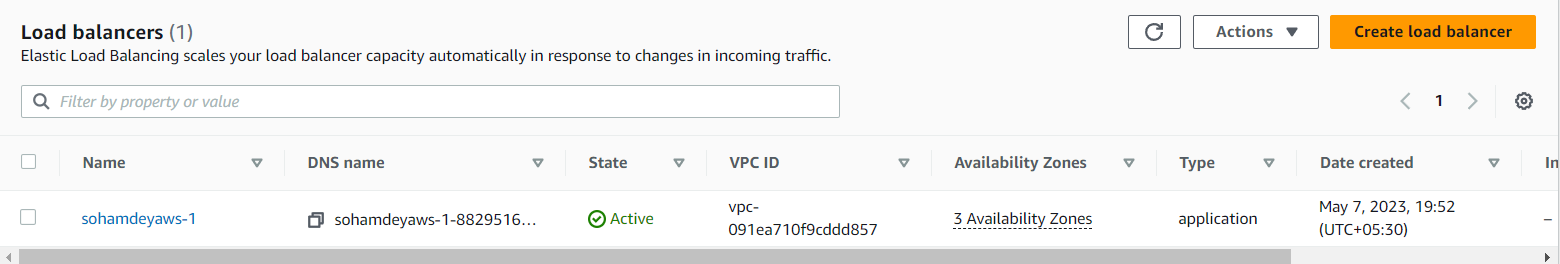
Now observe that whenever we close or terminate any instance then a new instance gets created. Hence, we cannot delete them if we want to delete them finally.

Follow these steps to cleanly remove Auto-Scaling Groups and everything related to it.

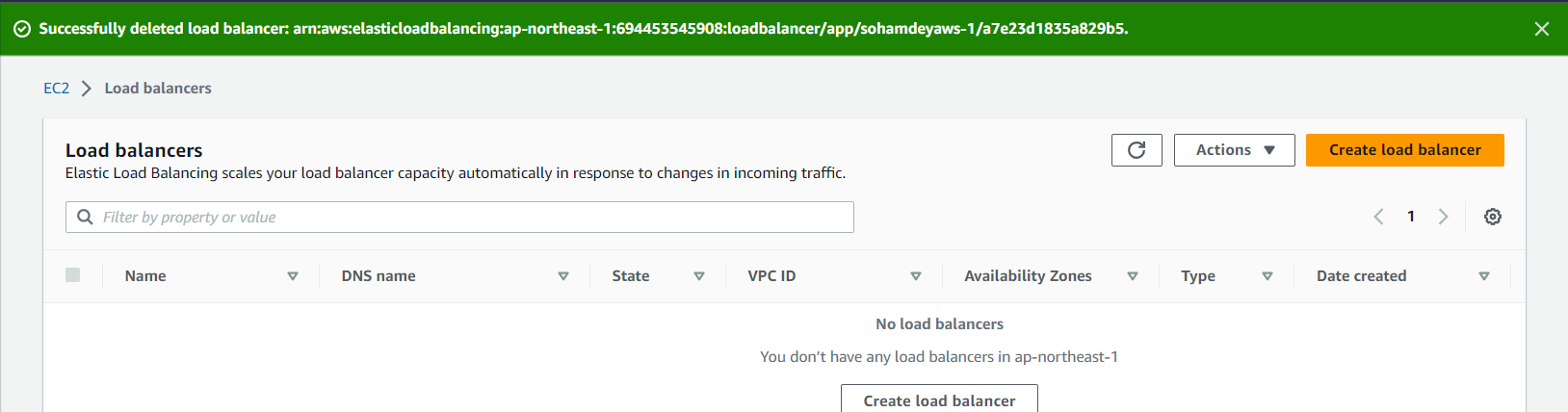
* Go to Auto-Scaling groups and select the one which we are using.



* Now click on the delete button.
* Type delete and finally delete it.
* Notice the status changes to Deleting. It will take some minutes to fully delete. Now go to the Load balancer section from the left side nav bar.



* Now select the load balancer and click on the action button on the top. Now select the Delete Load balancer option to delete it.
* Now go to the Target Groups section. Select the target group and click on the action button on the top. Select the delete option.
* Now go to the instances page.
* You will find that all the instances created by the Auto-Scaling group will automatically be terminated.
* Finally check the Auto-Scaling Group section to see if it is completely removed/deleted.



***Everything has been deleted.***