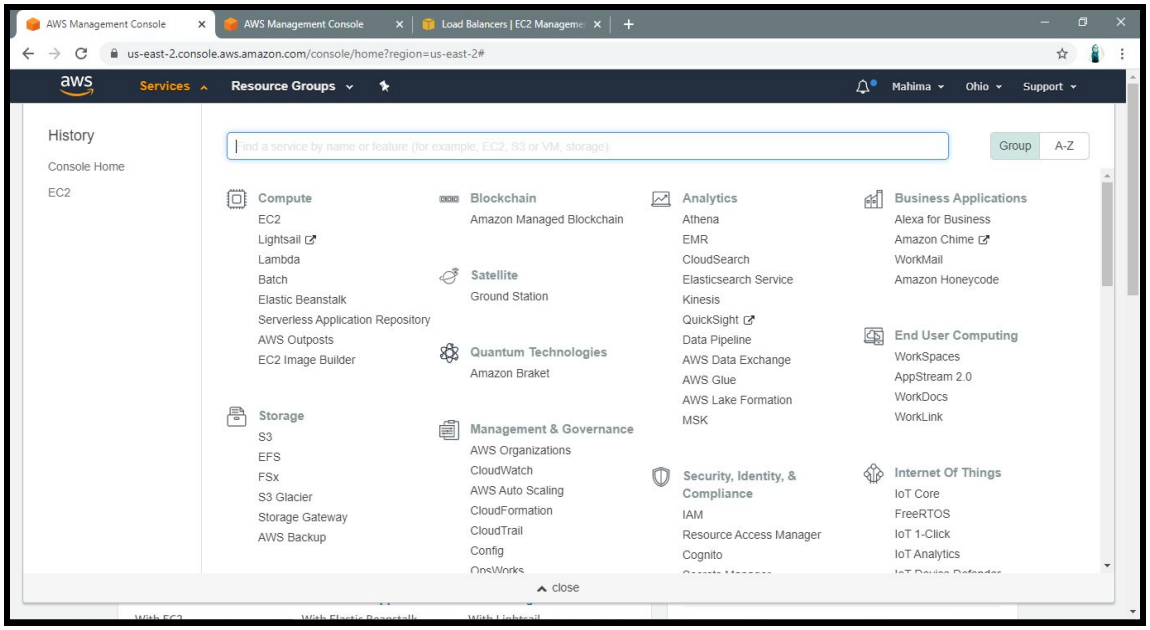
PROJECT - III

CREATE AN APPLICATION LOAD BALANCER WITH TWO INSTANCES AS TARGETS AND CHECK FUNCTIONING OF ELB

TASK - I: Create two Linux Instances - Amazon Linux 2 AMI (HVM), SSD Volume Type

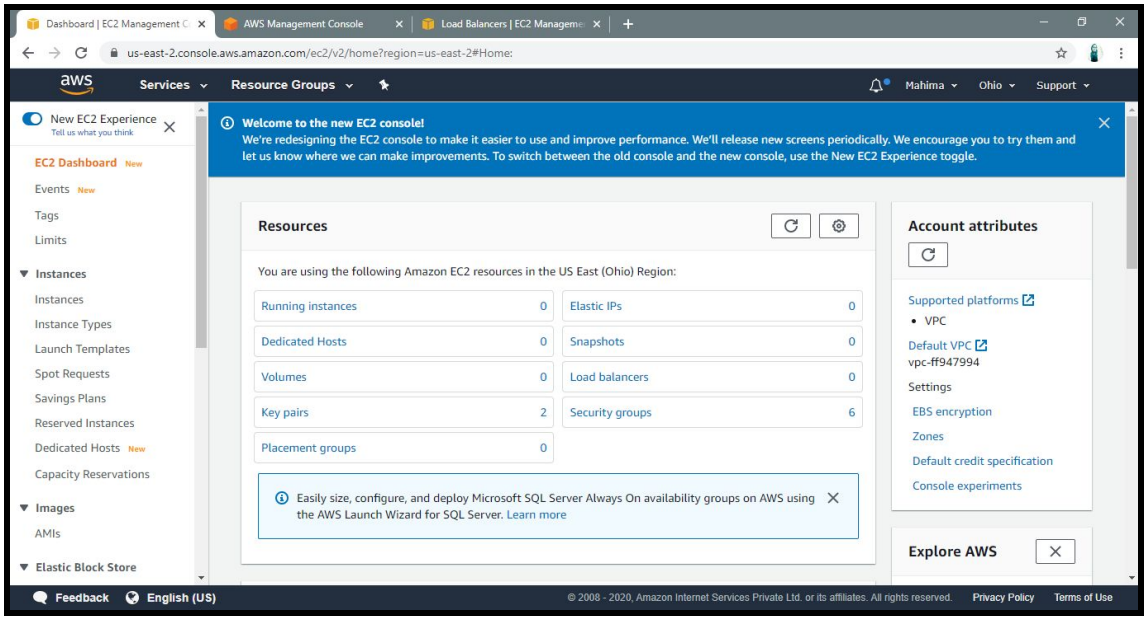
Step - 1:

Go to the AWS Management Console, click Services Tab and select EC2 from Compute service.



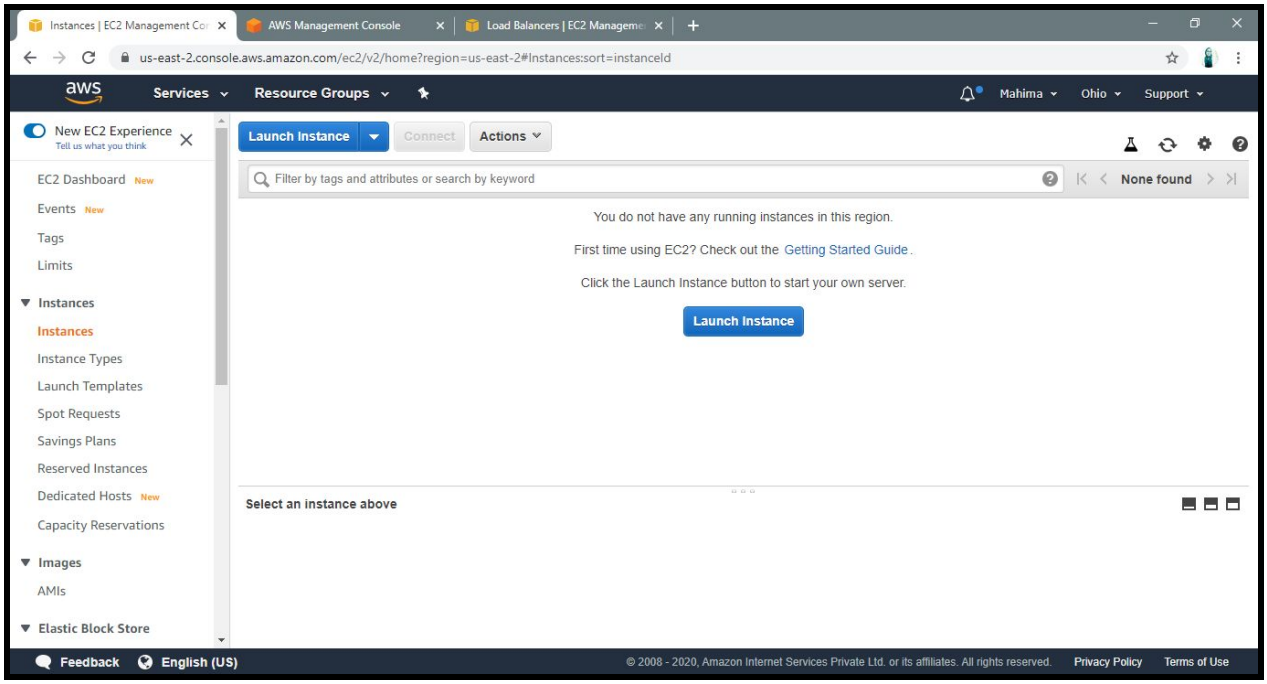
Step - 2:

On EC2 Dashboard, click on Running Instances displayed under the Resources Header.



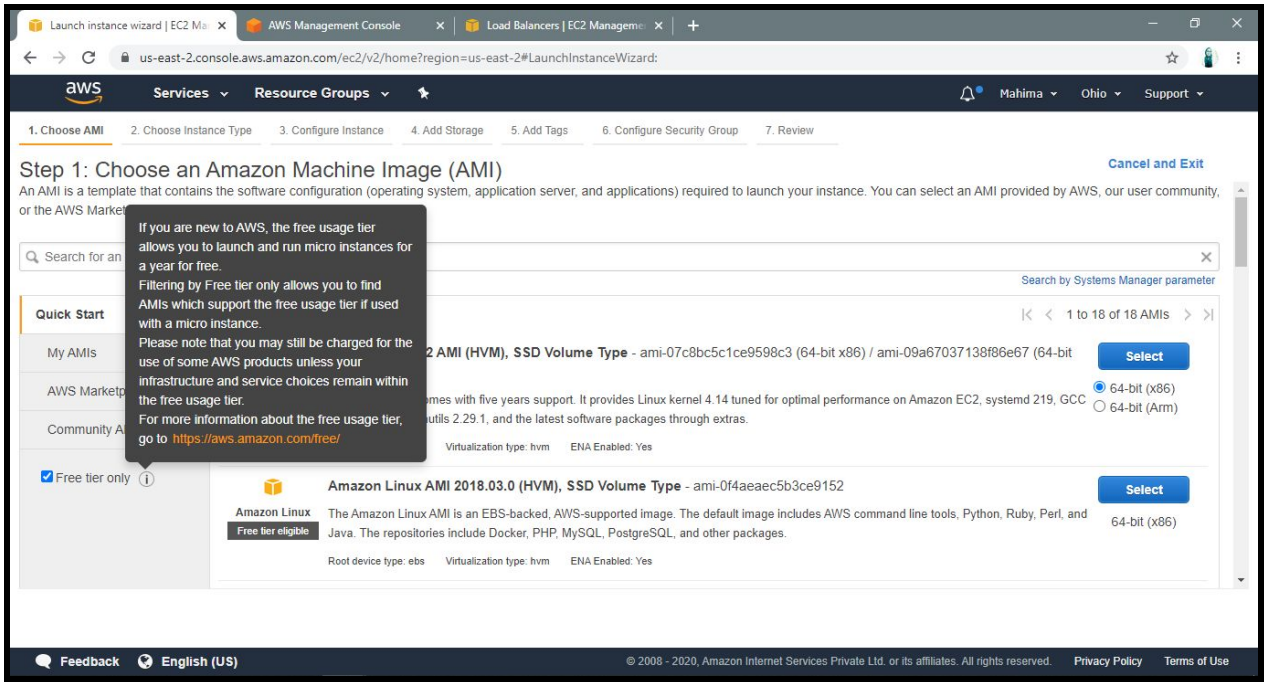
Step - 3:

Now, click on Launch Instance to create a new Virtual Machine i.e. **Amazon Linux 2 AMI** in our case.



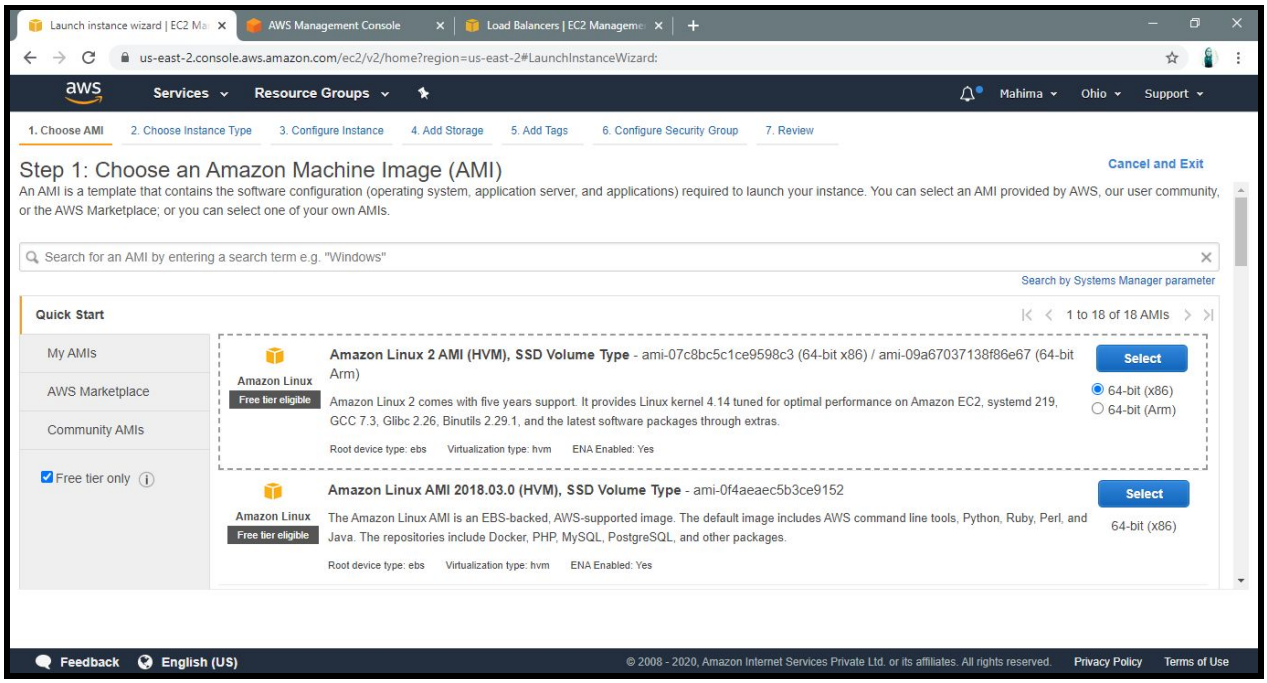
Step - 4:

First step in launching a machine is “Choose an Amazon Machine Image (AMI)” from a plethora of pre-existing images. From the left aligned menu bar, enable the “Free tier only” checkbox to list only free tier eligible resources available from AMI and avoid any additional charges from your account.



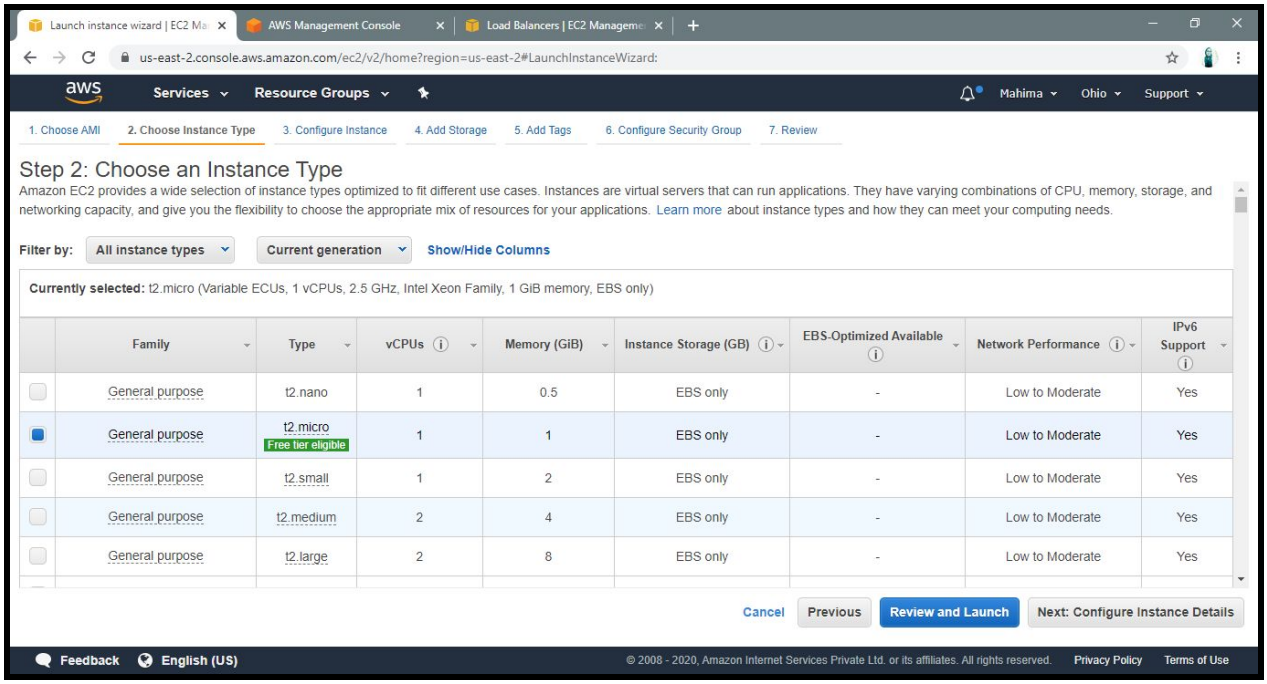
Step - 5:

Select “Amazon Linux 2 AMI (HVM), SSD Volume Type” eligible under the free-tier. On selecting it will be redirected to “Next: Choose Instance Type”.

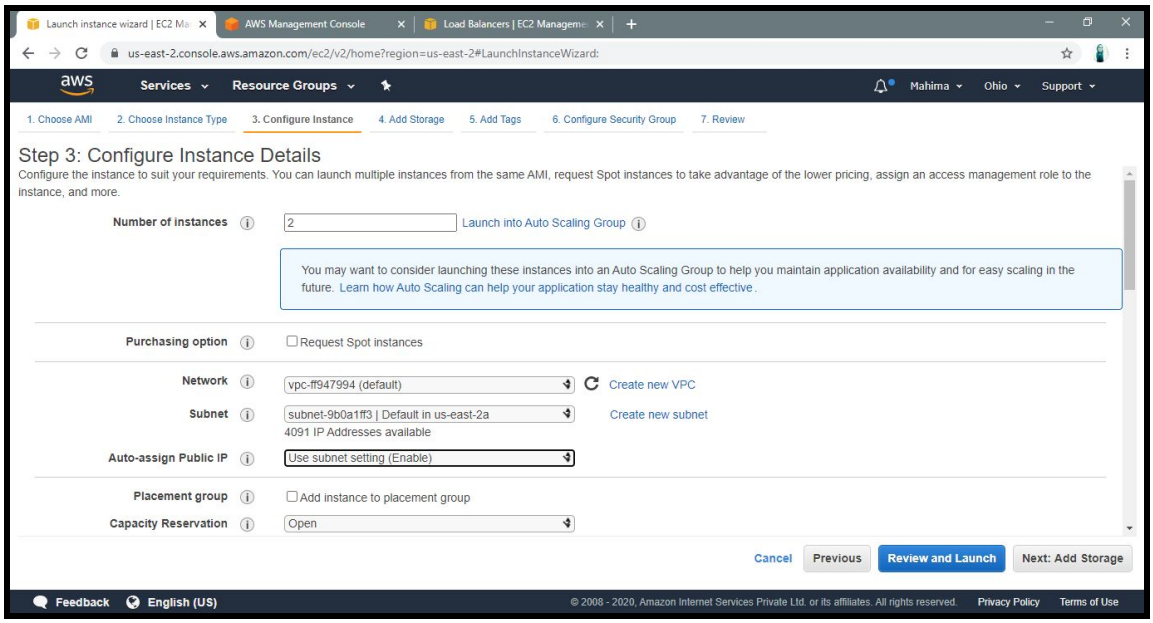


Step - 6:

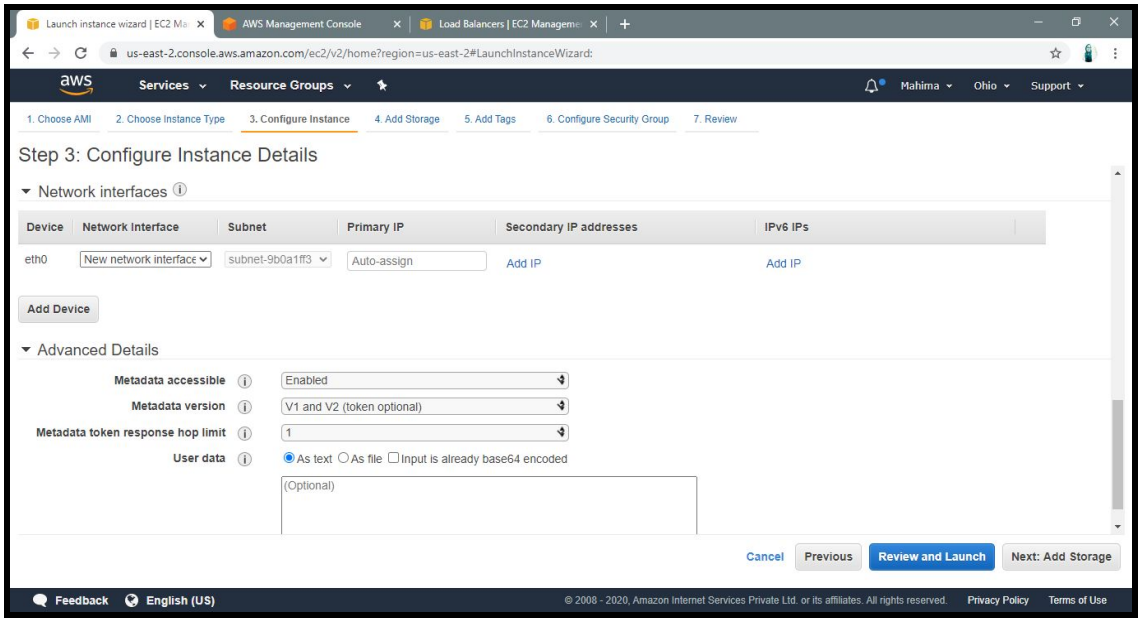
Second step is to “Choose an Instance Type” which has varying combinations of CPU, memory, storage, and networking capacity. We are selecting “General Purpose t2 micro” instance type eligible under the free-tier. Click “Next: Configure Instance Details”.



Step - 7:
Third step is “Configure Instance Details” i.e.,
Number of Instances: 2 (As 2 Instances are required for ELB Lab, we can create them at once of similar configurations)
Network: default (Select a default VPC as of now)
Subnet: us-east-2a (Or any default subnet can be chosen)
Auto-assign Public IP: Use subnet setting (Enable)
Rest leave all the parameters as default and click on the “i” icon corresponding to those parameters to understand the brief of each parameter.

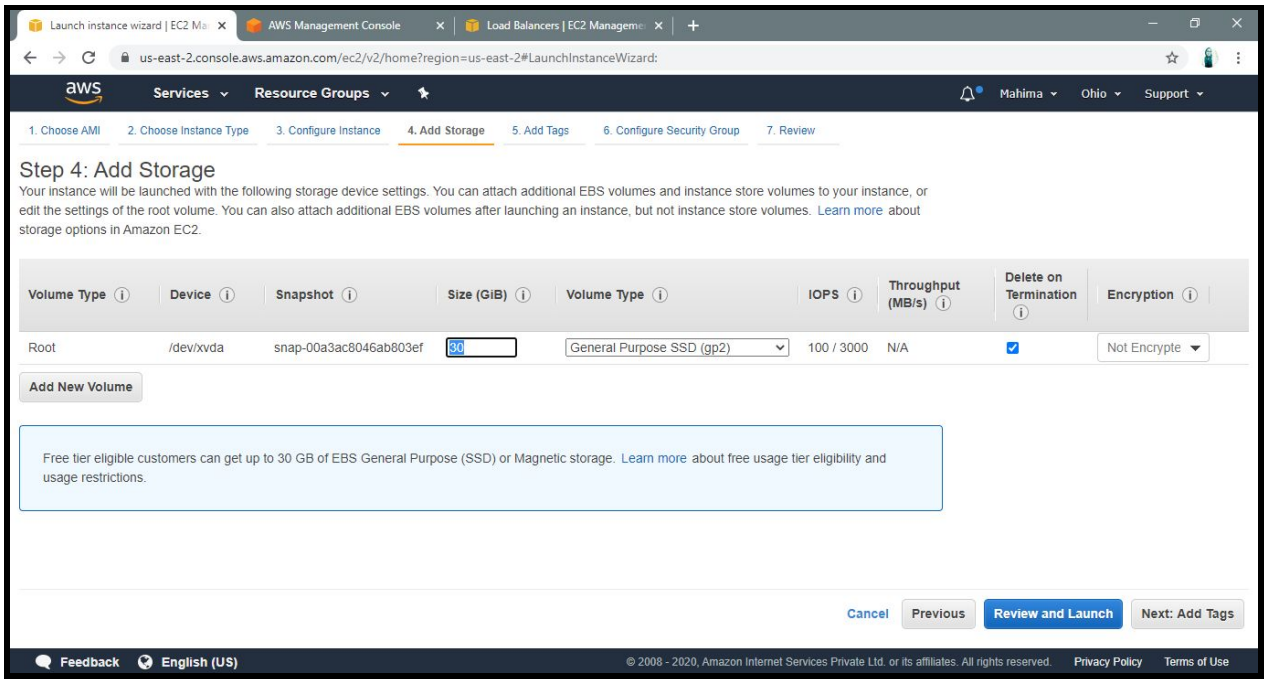


Step - 9:
Now keeping the “Network Interfaces” and “Advanced Details” section as default, click on “Next: Add Storage” to proceed to the next step.



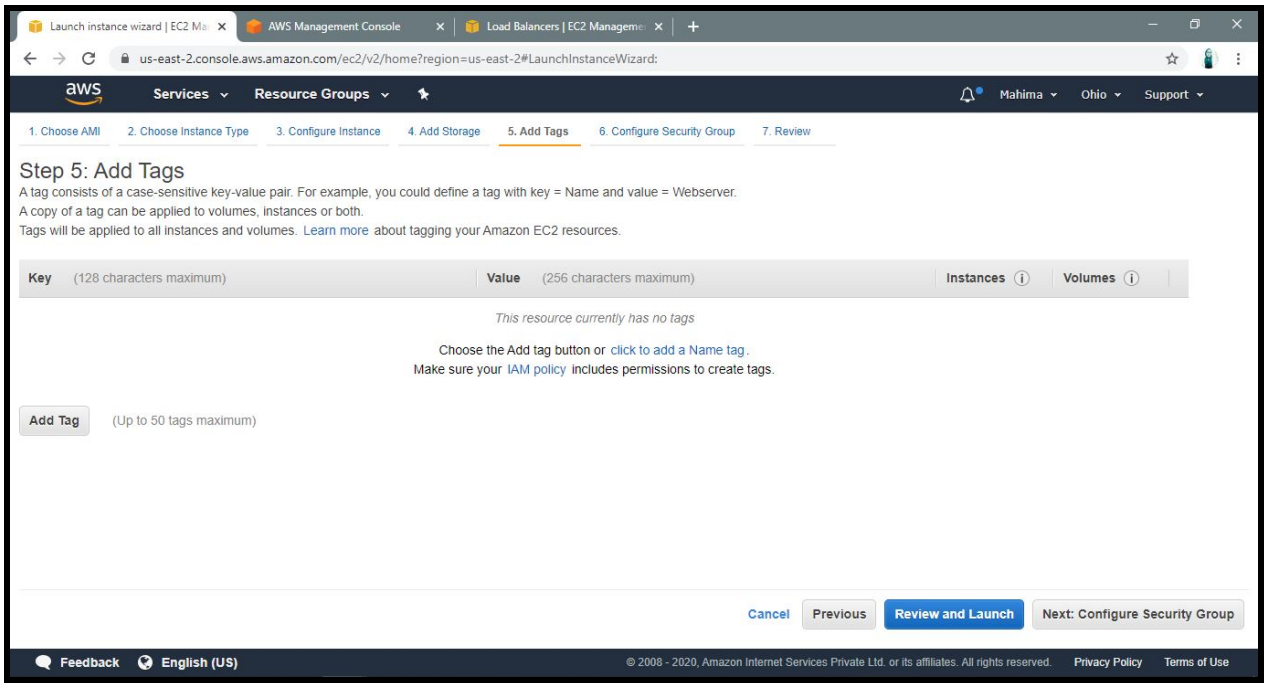
Step - 10:

Fourth step is to update the size of the root volume i.e. “30 GIB” which can be kept as it is as by default i.e. “8 GIB” or modify it to “30 GIB” and then click “Next: Add Tags” to proceed to the next step.



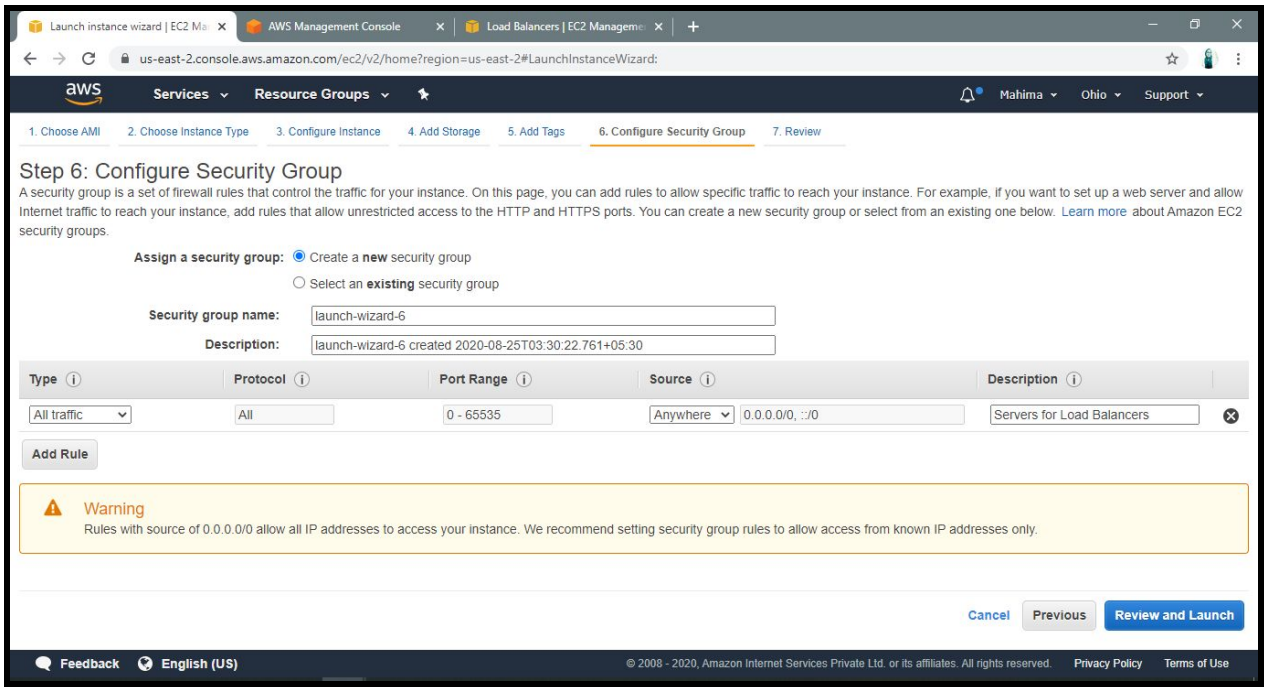
Step - 11:

Fifth step is adding a tag which is mainly used for us to filter out the required instance from a list of n number of Instances created i.e. by giving an apt name to the VM as per our use case. But as of now we can skip this as it’ll assign the same tag to all the Instances created while its launch. Then click “Next: Configure Security Group” to proceed to the next step.



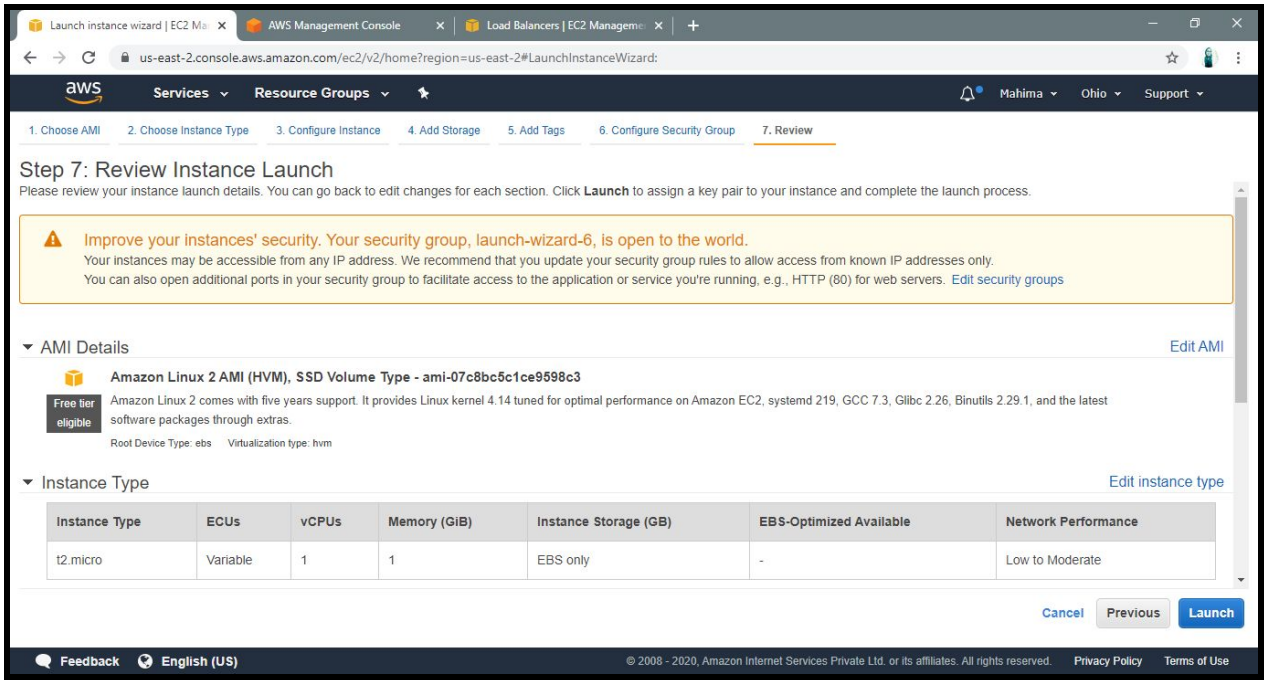
Step - 12:

Sixth step is “Configure Security Group”, we’ll create a new security group with “All traffic” enabled and source as “Anywhere” that means our VM can be accessed from anywhere and by anyone without any specific restrictions. Also give a Description as “Servers for Load Balancers”. Then click “Review and Launch”.



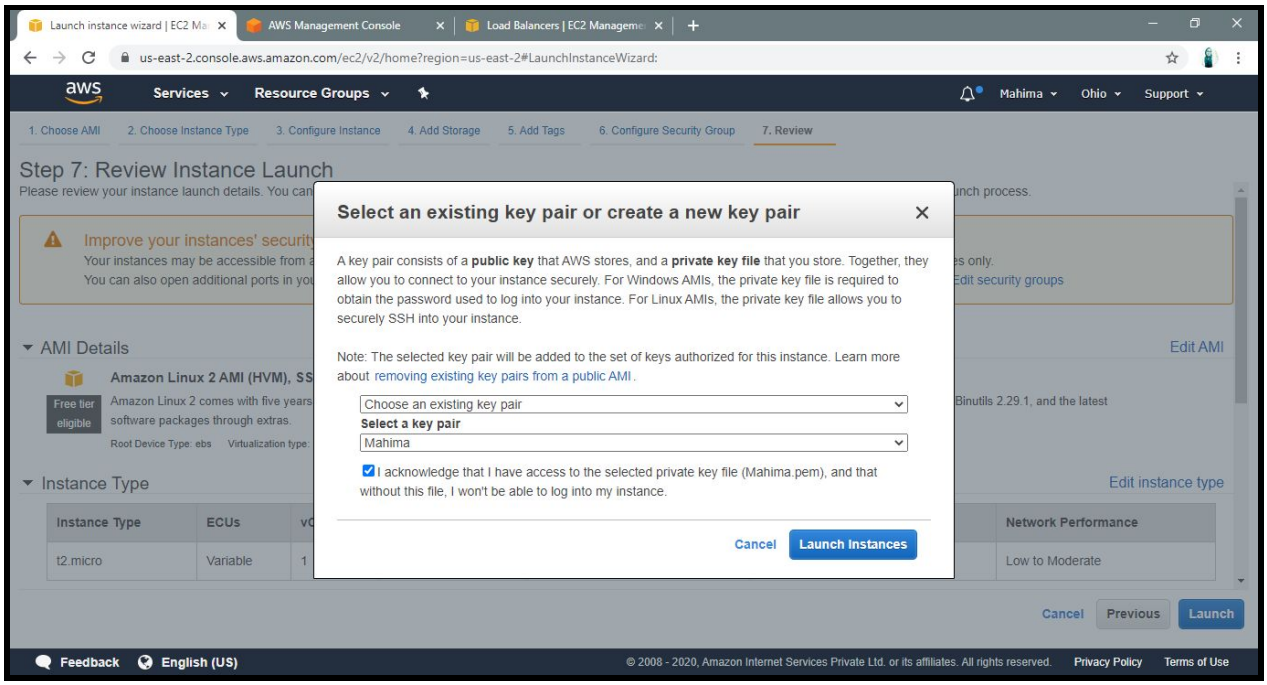
Step - 13:

After reviewing all the configurations selected and given for the VM, we can proceed to the Launch step and click the “Launch” button at the bottom right corner.



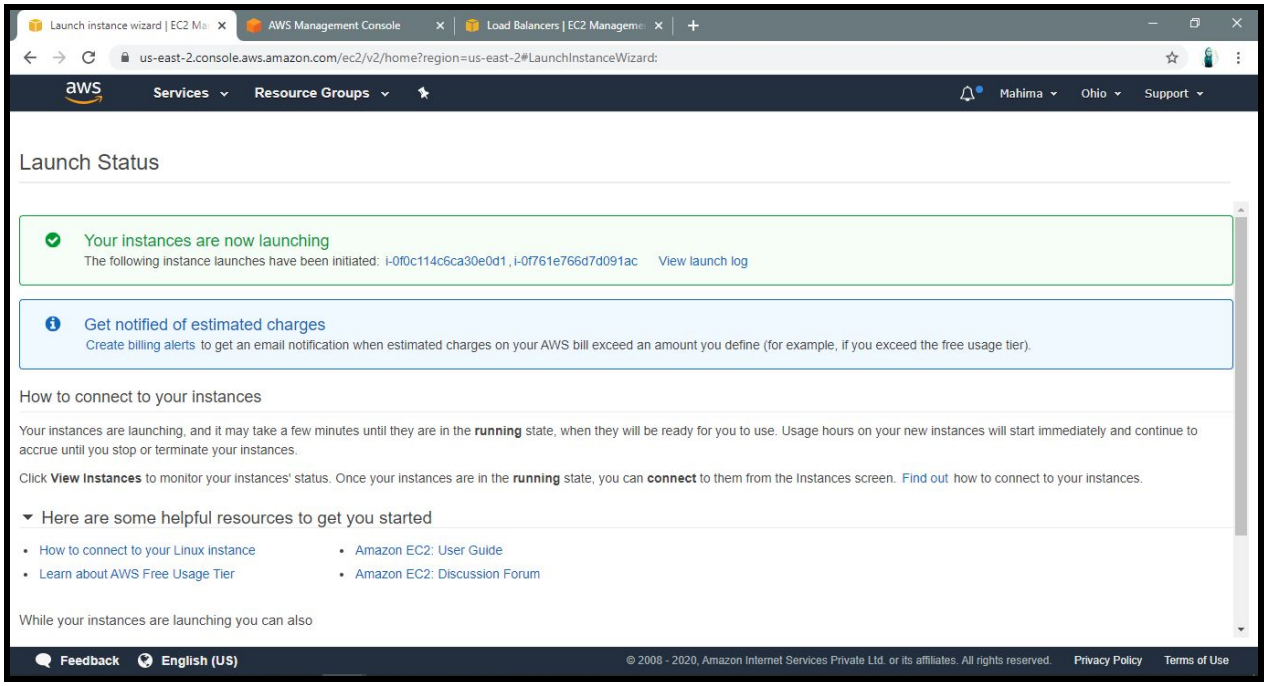
Step - 14:

Now, we have to “Select an existing key pair or create a new key pair” as a last step in the process of launching an Instance on AWS. From the dropdown, select “Choose an existing key pair” as we already have it in our system from our previous launch of a Windows Instance. Enable the acknowledgment checkbox and click on “Launch Instance”.



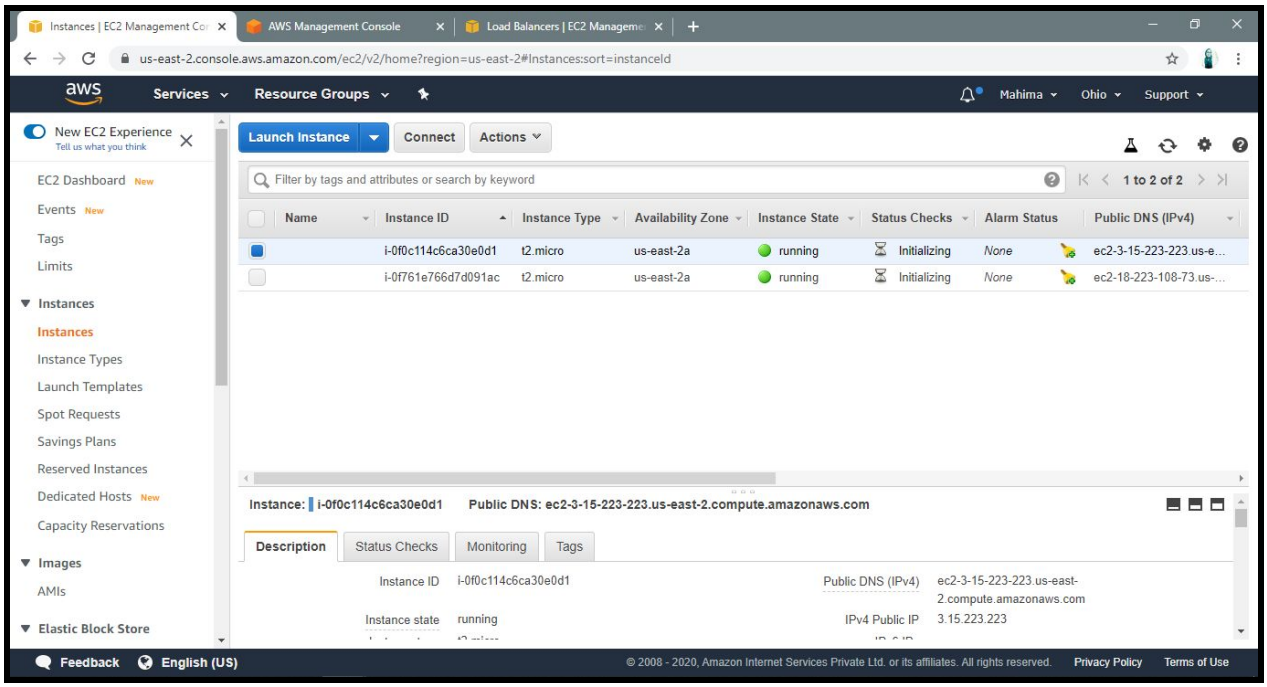
Step - 15:

Instance has started launching now, click on any Instance ID just beside View launch log displayed under Launch Status Header, also we can see 2 Instance ID’s being creating.



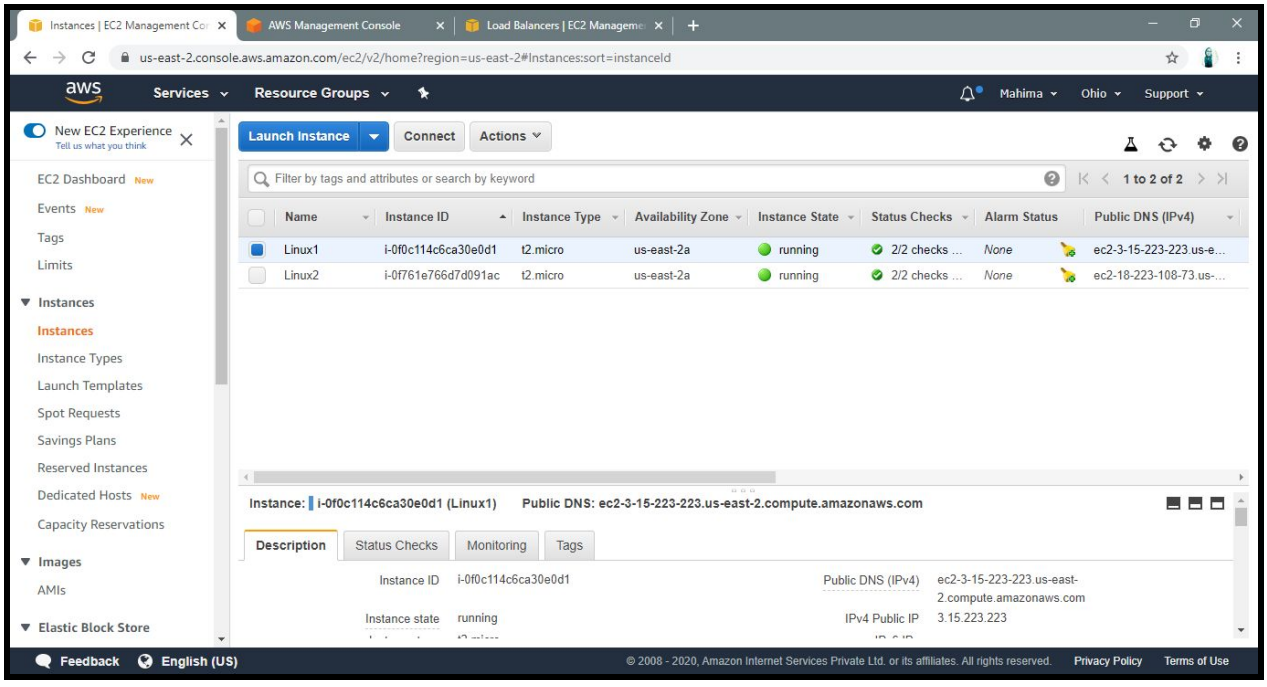
Step - 16:

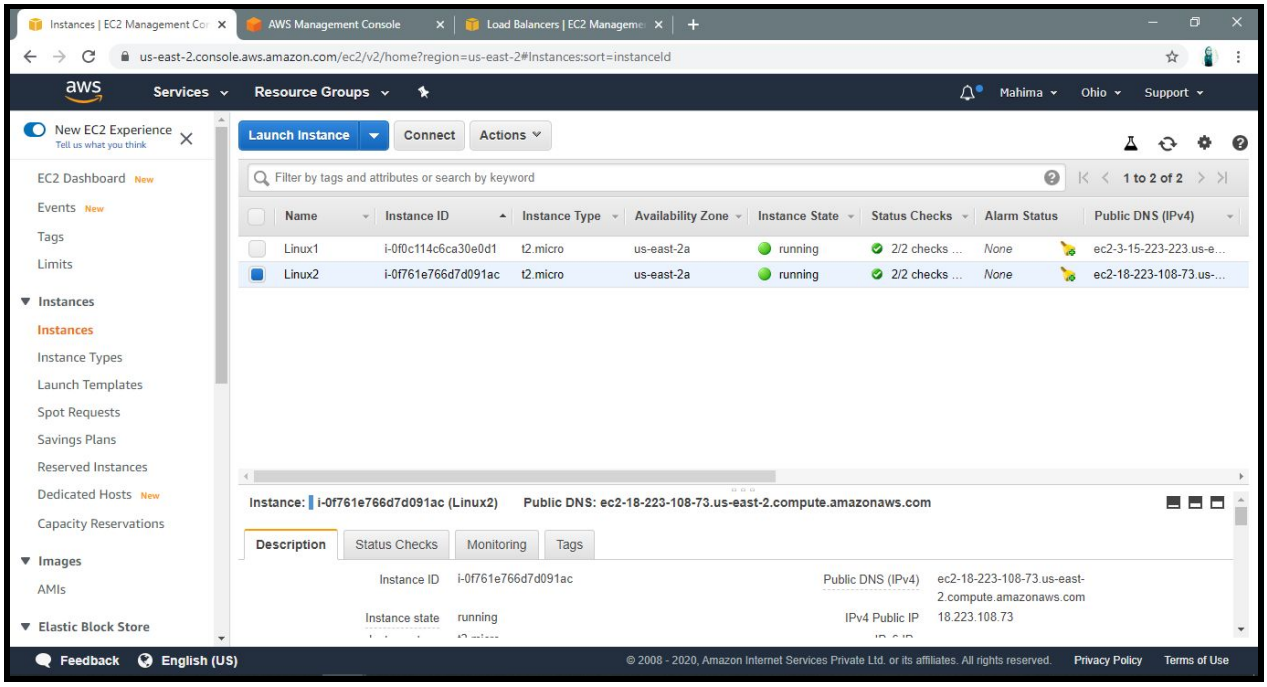
Observe Status Checks would be “Initializing” at the beginning for the corresponding Instance ID, wait till the Status Checks becomes “2/2 checks”.



Step - 17:

We observe 2/2 status checks on the console, hence we are good to go and connect to the respective VM. Also observe the IPv4 Public IP under the Description Header on the screenshot below for corresponding to Linux Instances created.

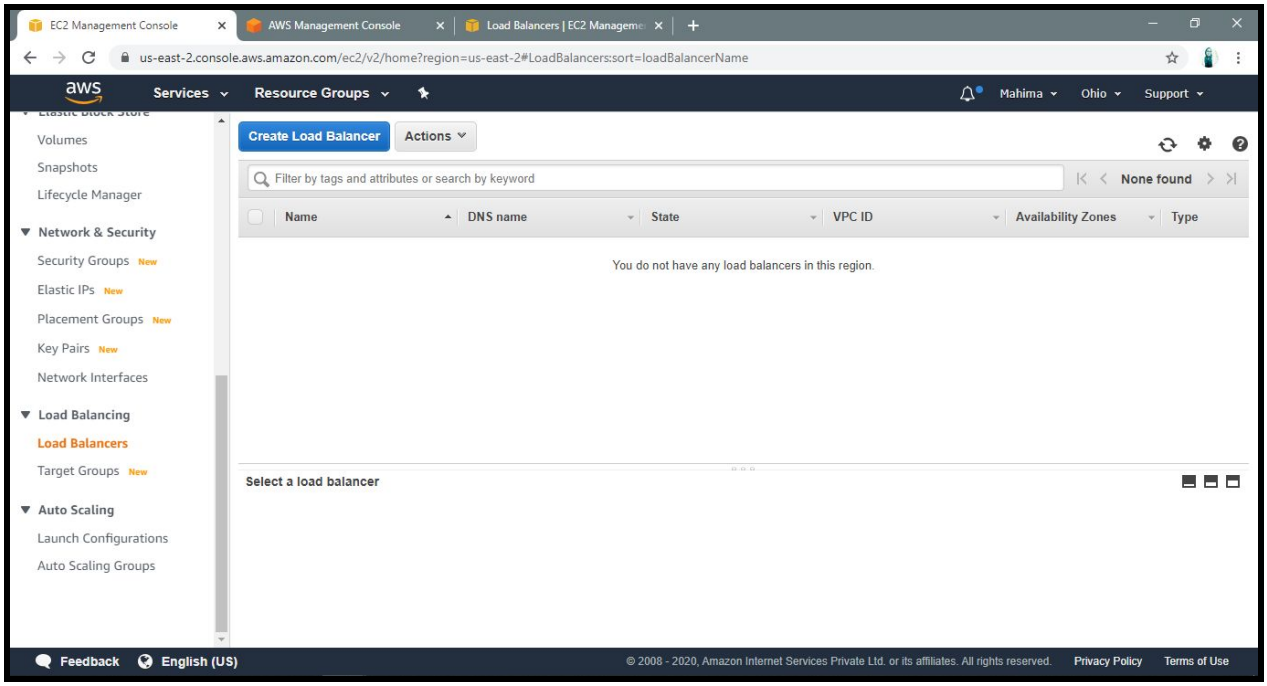




TASK - II: Create an Application Load Balancer with the above two Instances as target

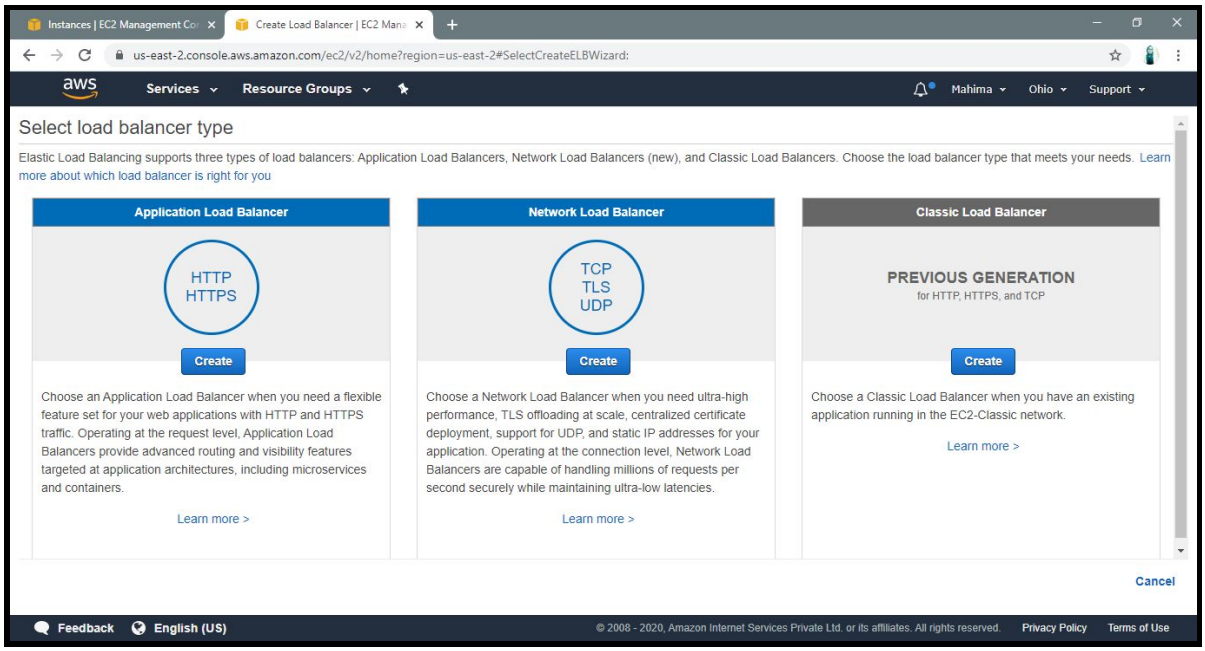
Step - 18:

From the left navigation pane, drop down to the bottom and under the Load Balancing header select “Load Balancers”. Then click on Create Load Balancer Button.



Step - 19:

Select a type of Load Balancer, in our case we would select “Application Load Balancer” as we have to deploy web applications with HTTP or HTTPS traffic and click Create.



Step - 20:

First step is “Configure Load Balancer” and provide all the required Basic Configuration and Listeners i.e.,

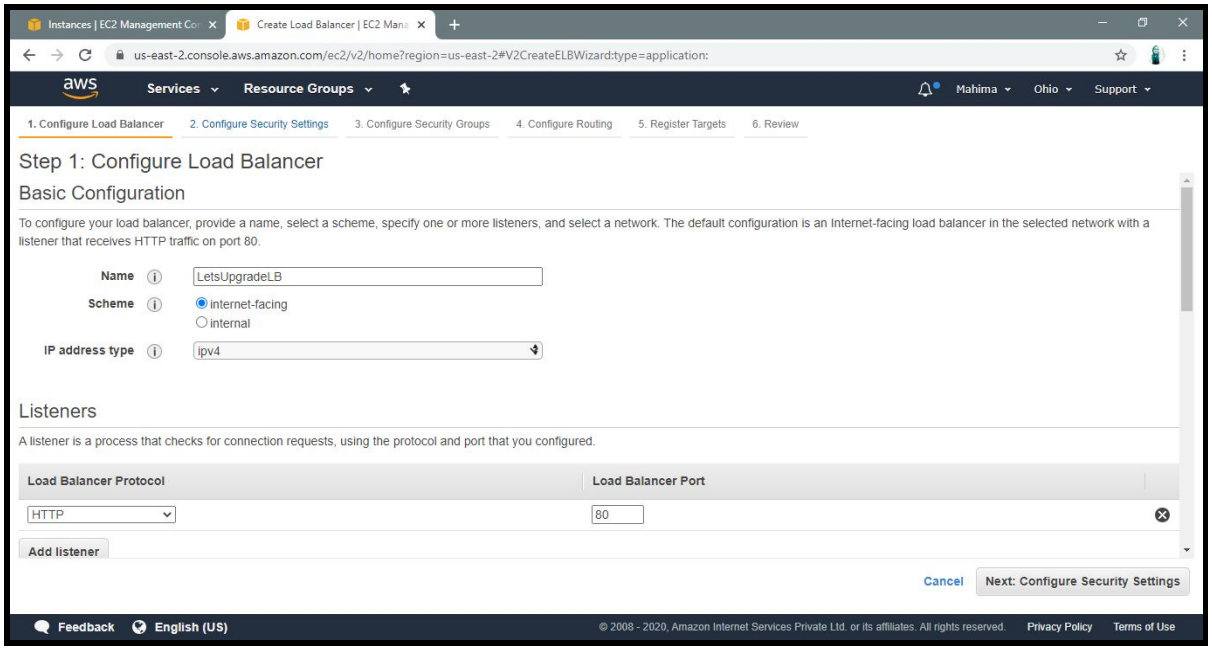
Name: LetsUpgradeLB

Scheme: internet-facing

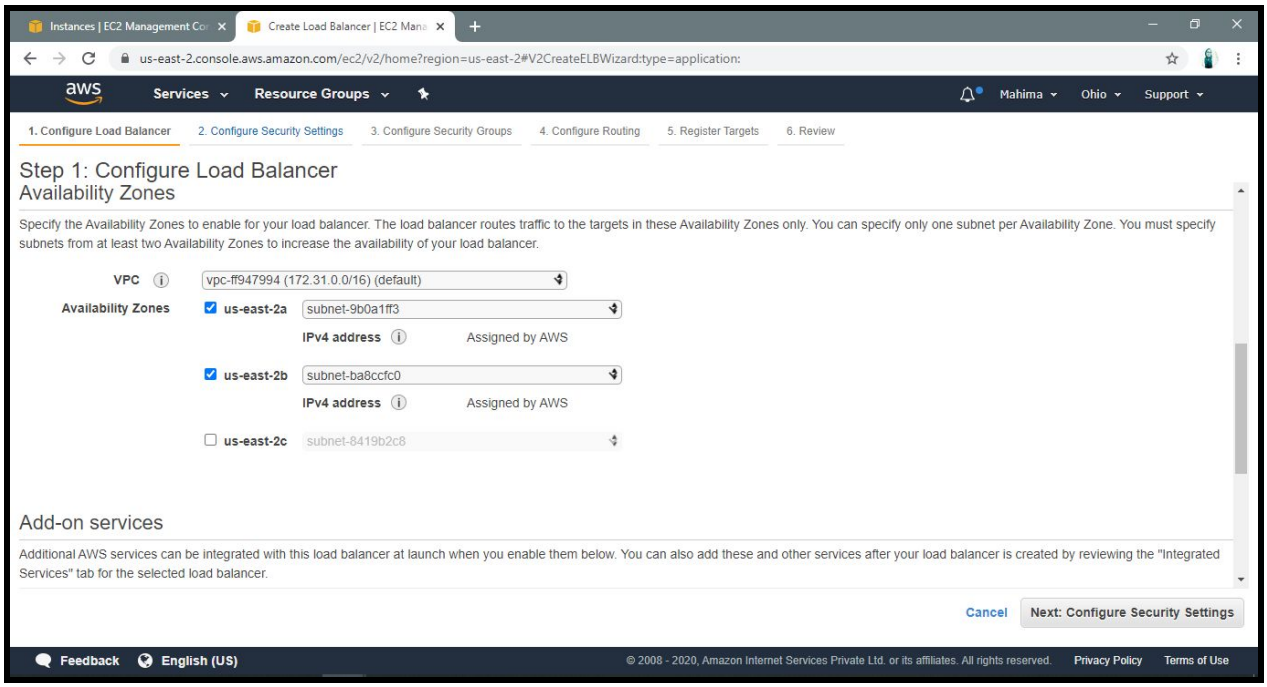
IP address type: ipv4

Load Balancer Protocol: HTTP

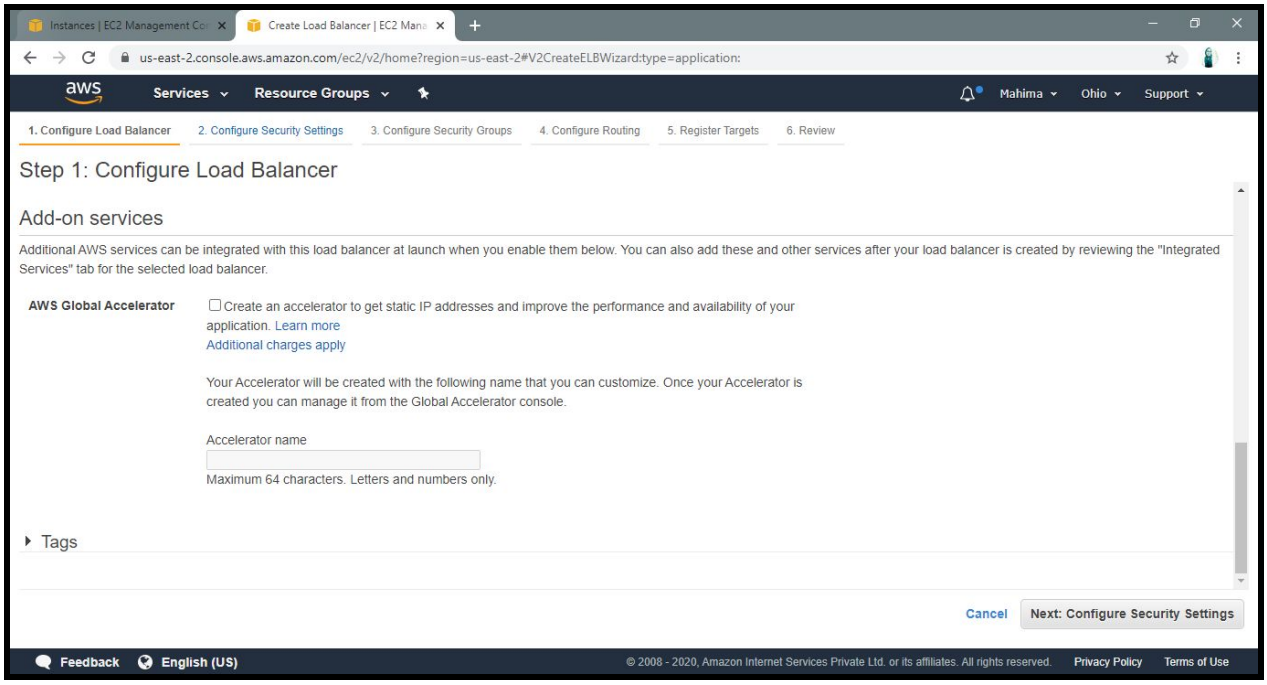
Load Balancer Port: 80



Step - 21:
Provide the details under Availability Zones header i.e.,
VPC: default (default VPC would suffice)
Availability Zones: us-east-2a, us-east-2b (select any two AZs)

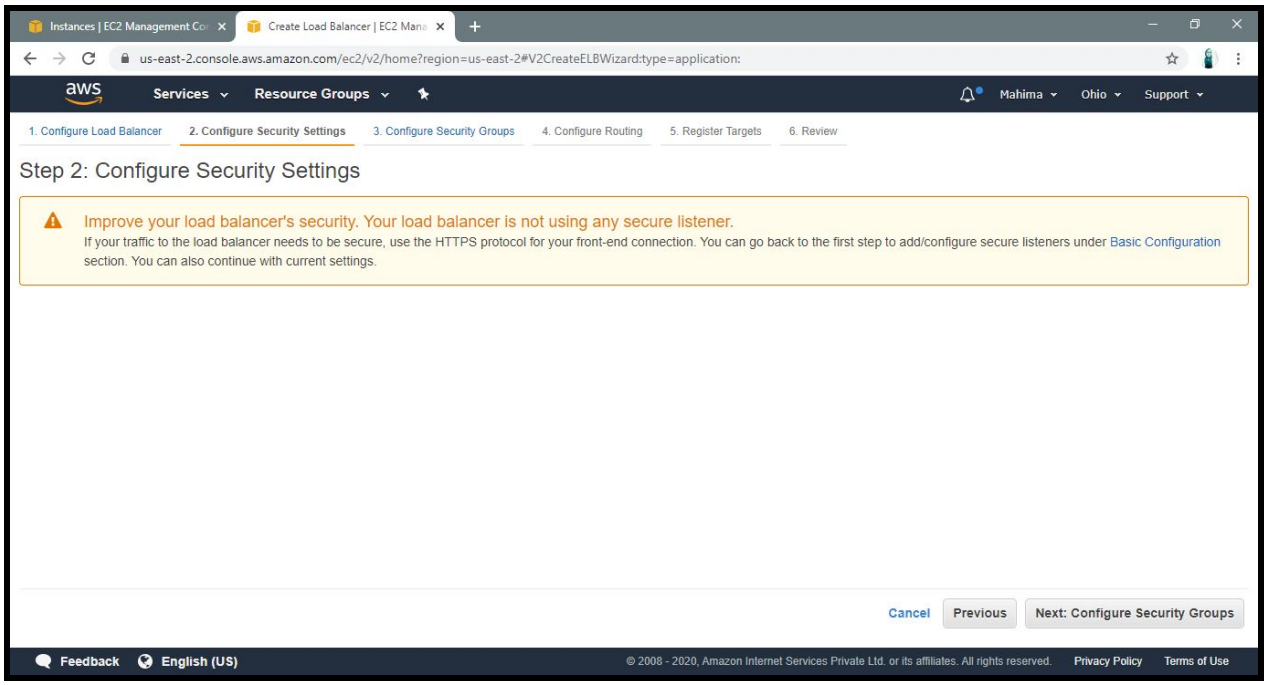


Step - 22:
Keeping the rest of the fields as default if required we can click an ”i” icon in front of the labels to understand a brief about the same. Then click “Configure Security Settings” to proceed to the next step.



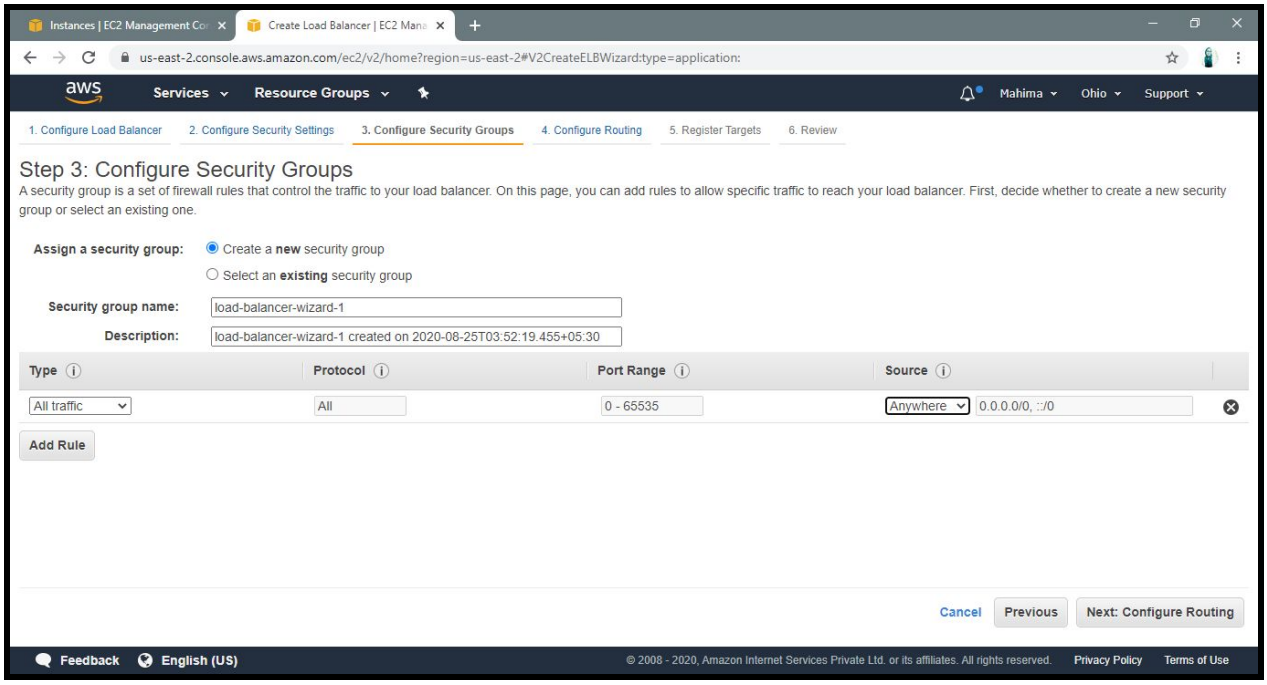
Step - 23:

We can skip this as we are using HTTP Protocol and not HTTPS Protocol which is more secure hence requires some sought of certificate and proceed to the next step “Configure Security Groups”.

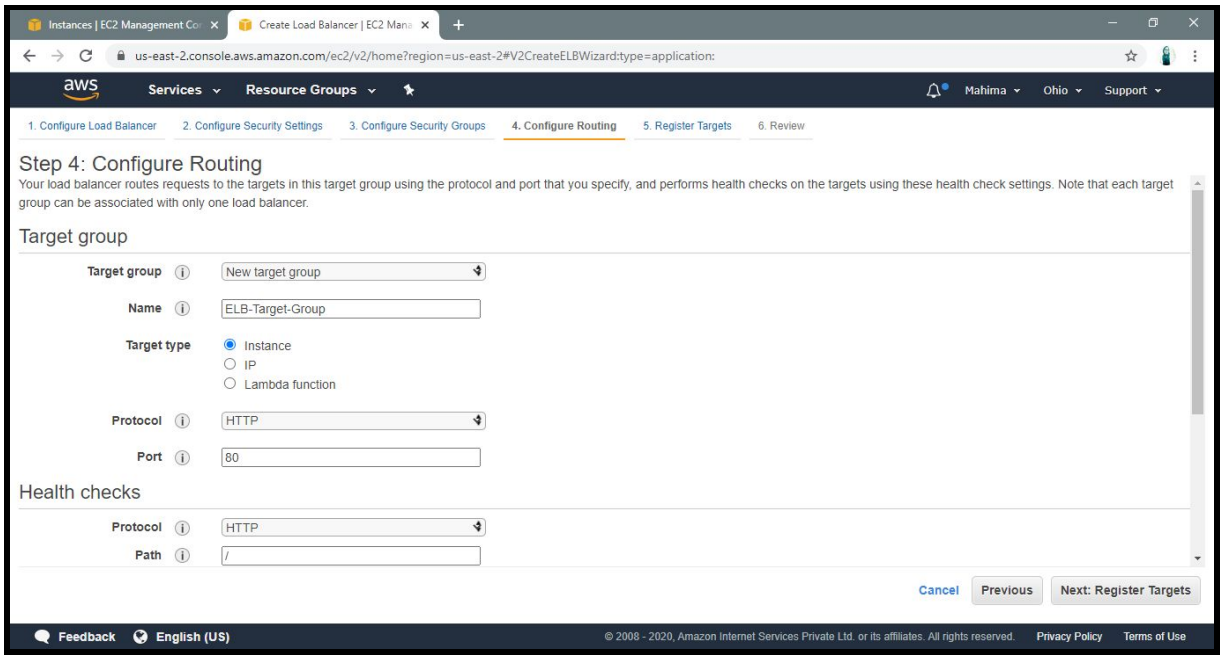


Step - 24:

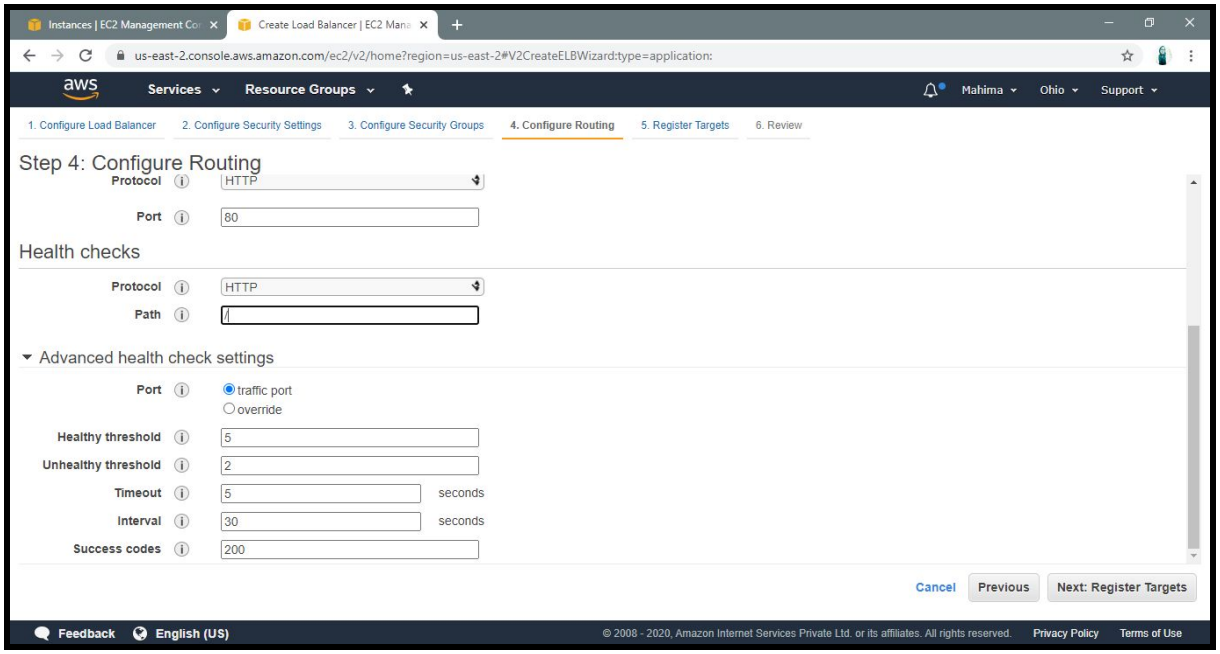
Third step is “Configure Security Groups” for the Load Balancer as well i.e. in our case creating a new security group with “All traffic” enabled and source as “Anywhere”. Then click “Next: Configure Routing”.



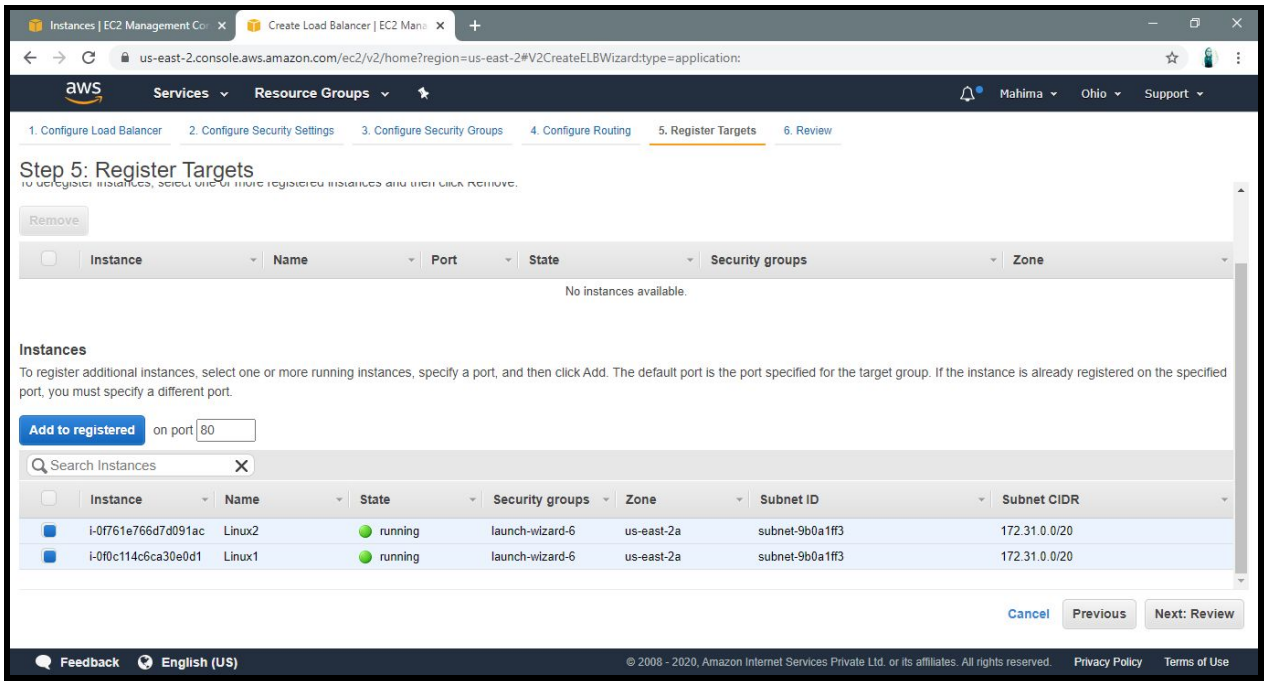
Step - 25:
Fourth step is “Configure Routing” i.e. provide a name to the Target Group i.e.,
Target group: New target group
Name: ELB-Target-Group
Target type: Instance
Protocol: HTTP
Port: 80



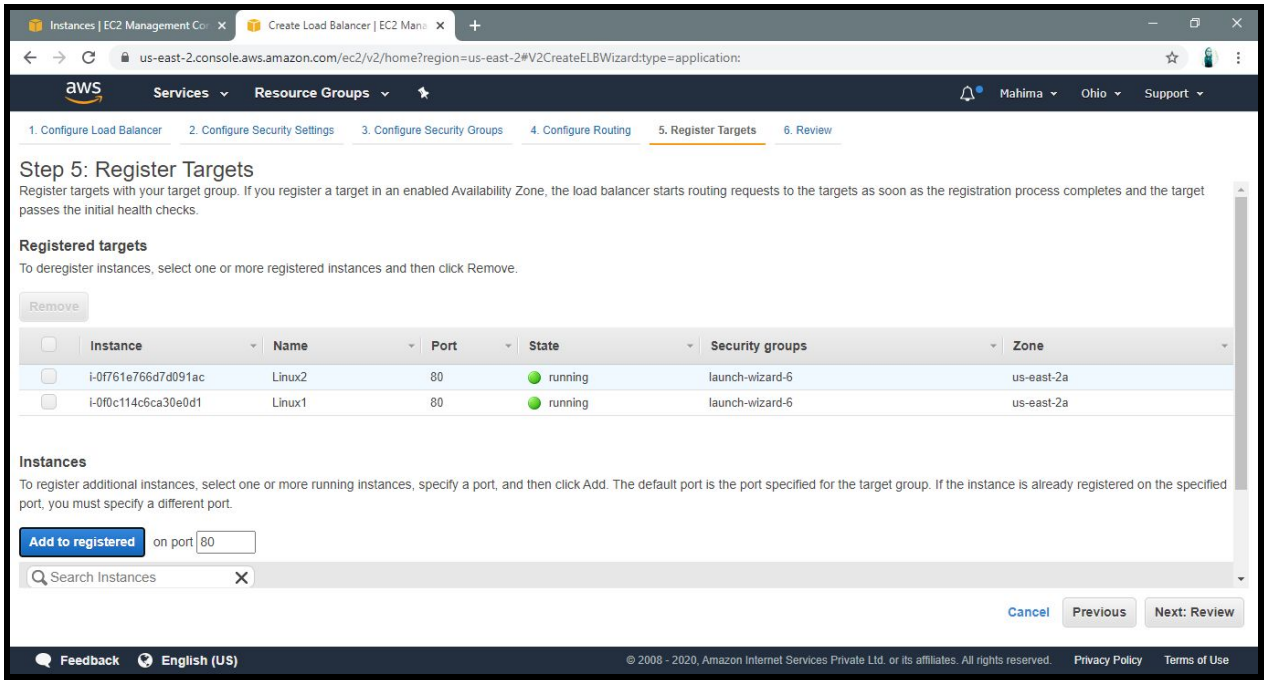
Step - 26:
Keep the Health Checks and Advances Health Check Settings as default only and click “Next: Register Targets”.



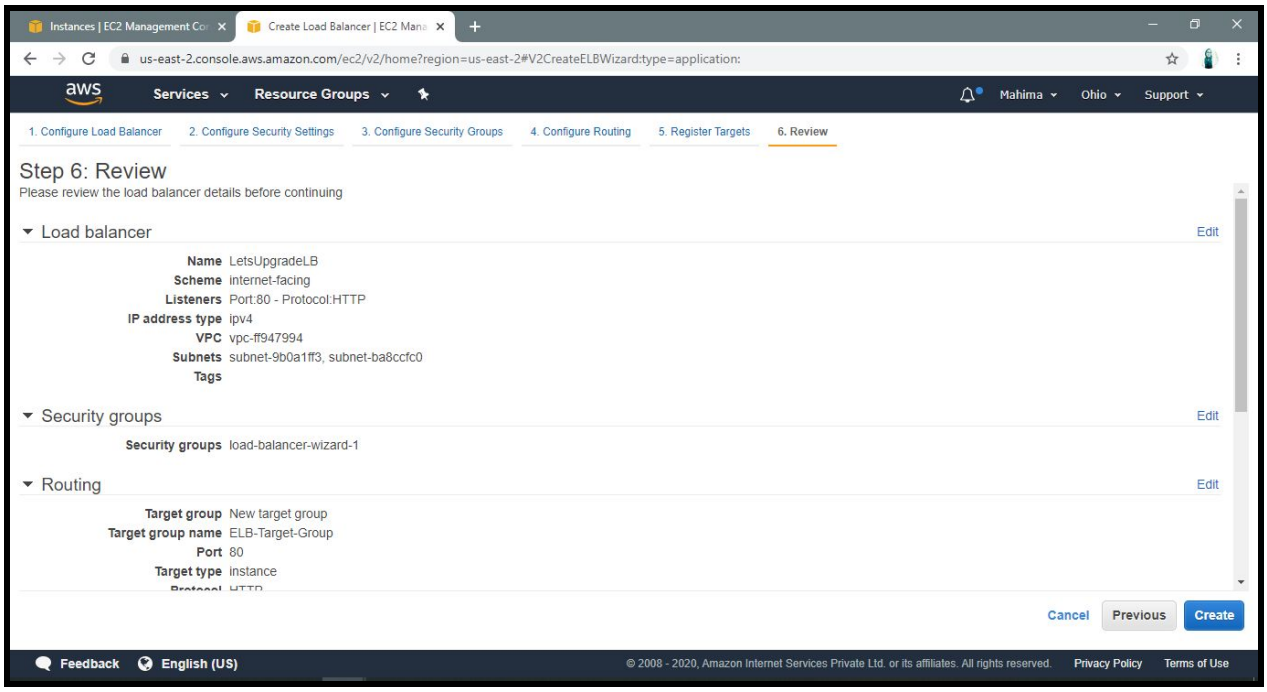
Step - 27:
Select both the Instances in the Gridlist i.e. Linux1 and Linux2 and thereafter click “Add to registered” button.



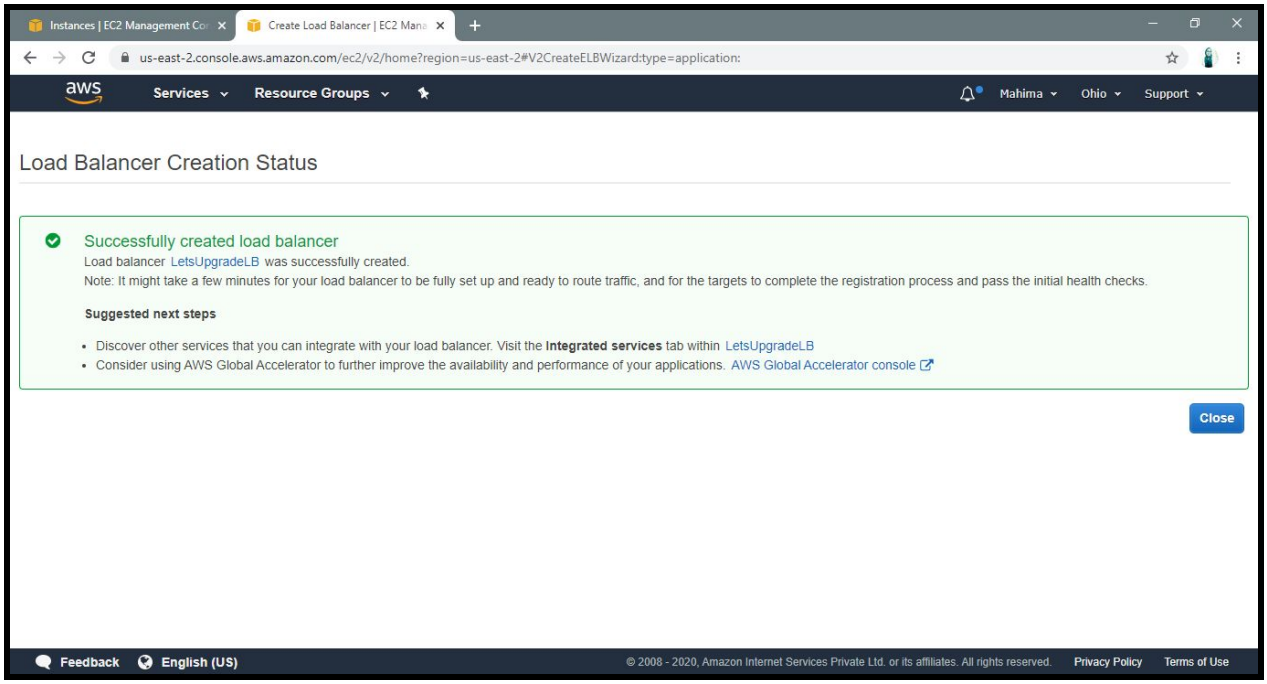
Step - 28:
Both the Instances selected should be displayed under the registered targets list and hence click “Next: Review” to proceed further.



Step - 29:
After reviewing all the details entered for configuring an Application Load Balancer, click on the “Create” button.

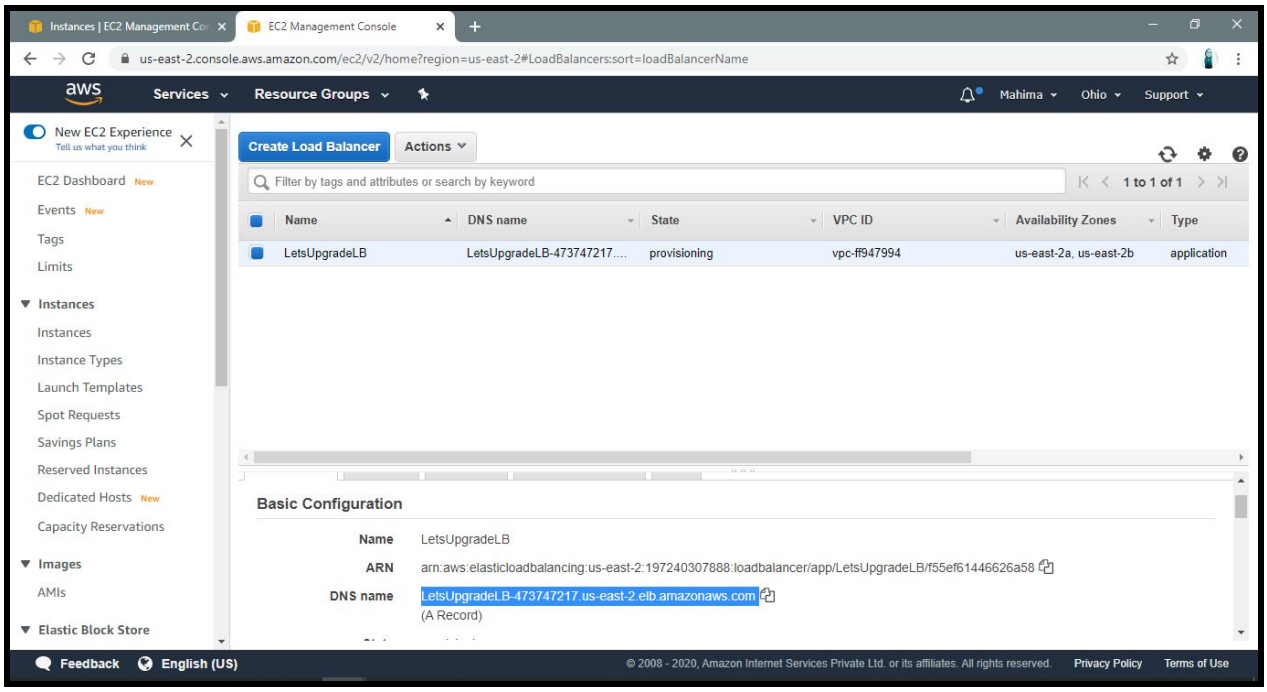


Step - 30:
Load Balancer Creation Status should be displayed on the screen as “Successful” and click on close button.



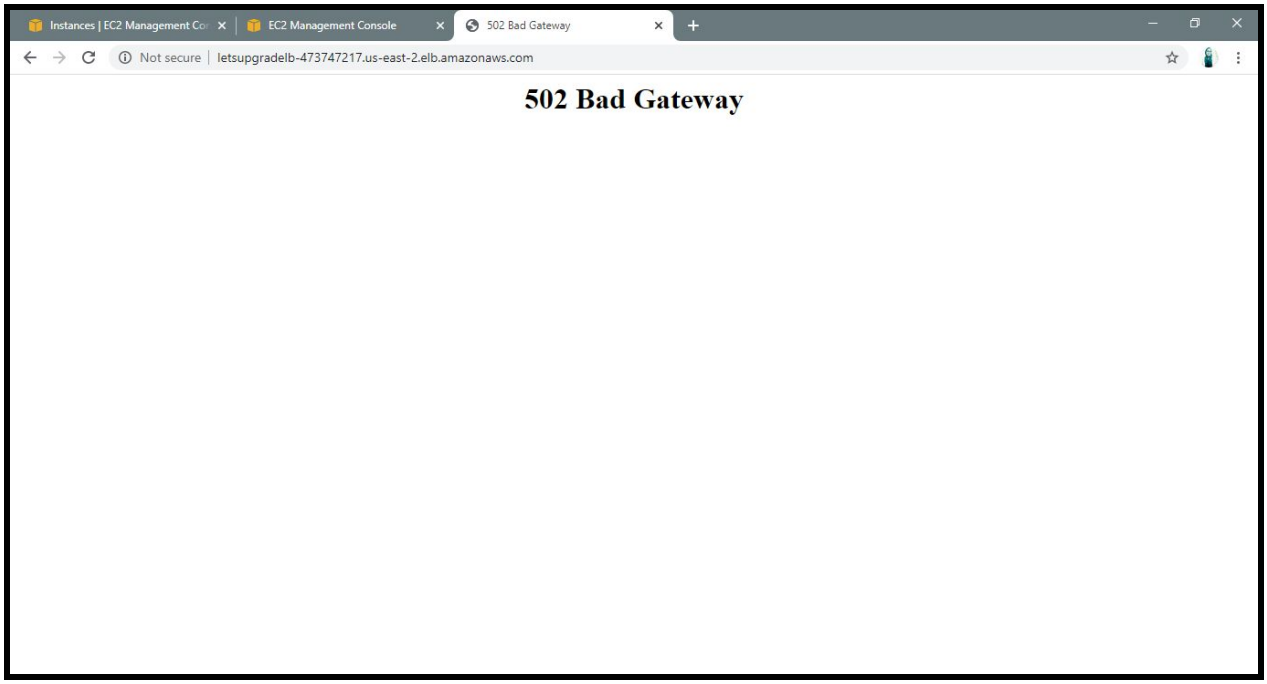
Step - 31:

You can observe the Load Balancer State under “Provisioning” and it might take a while to come into Active state. Observe the DNS Name under Basic Configuration for the created Load Balancer.



Step - 32:

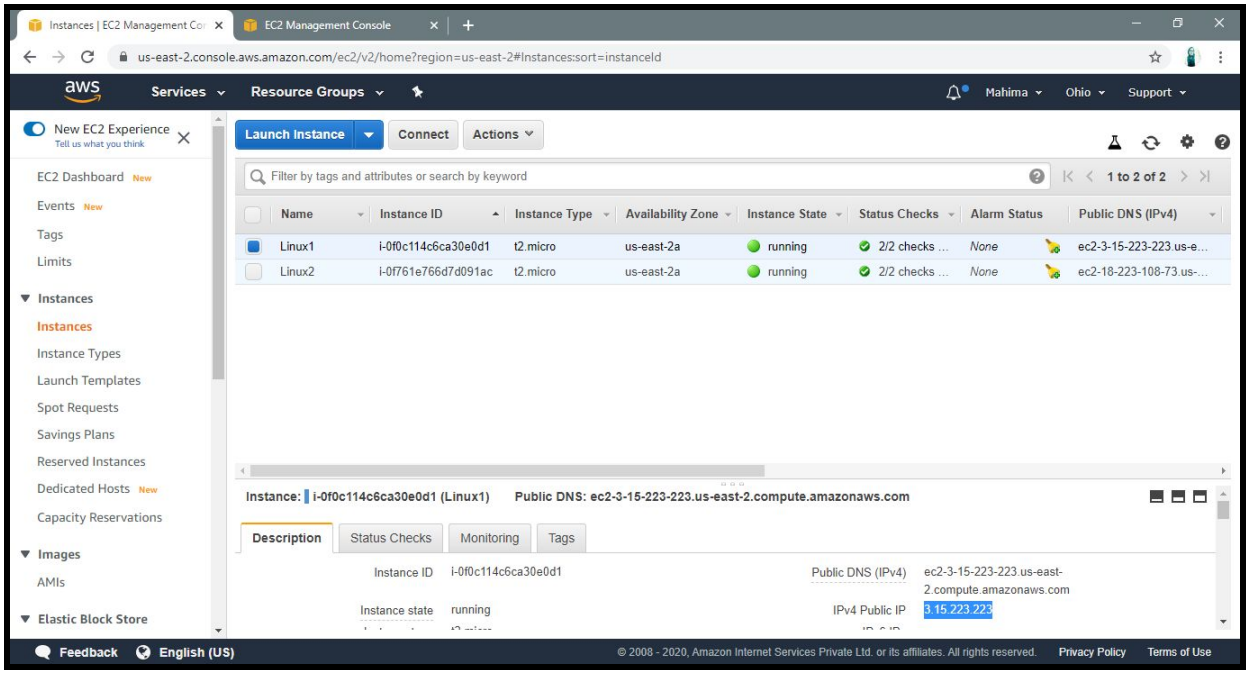
After pasting the DNS in a Browser, it shows us “502 Bad Gateway” as we have not yet hosted a web page on the servers. We’ll hit it again after hosting the web pages on the Servers.

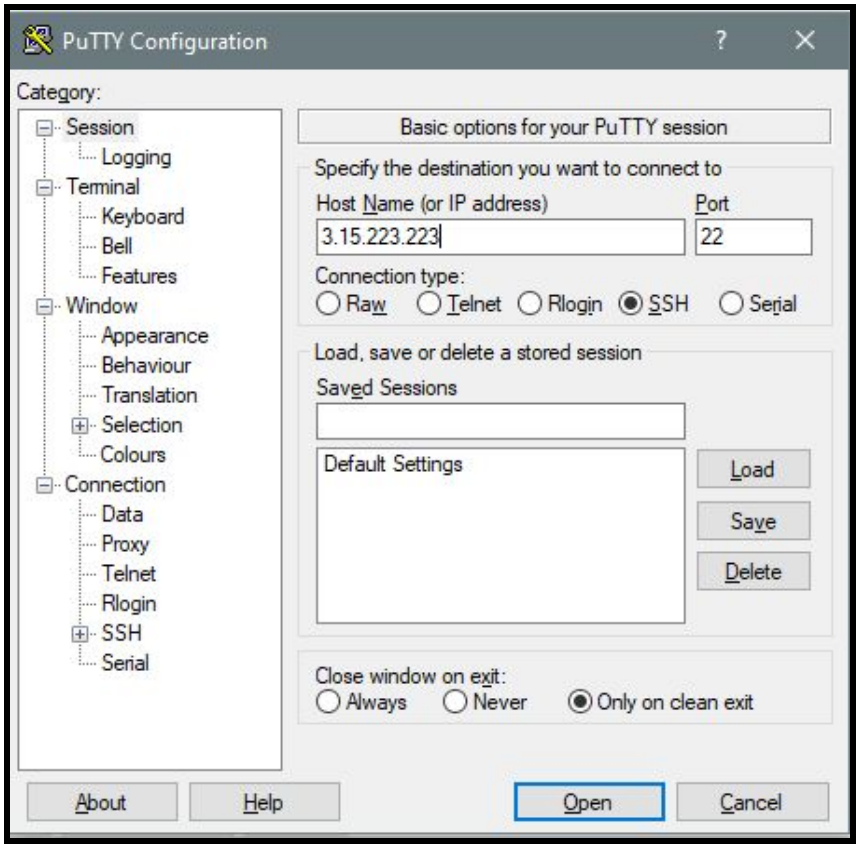


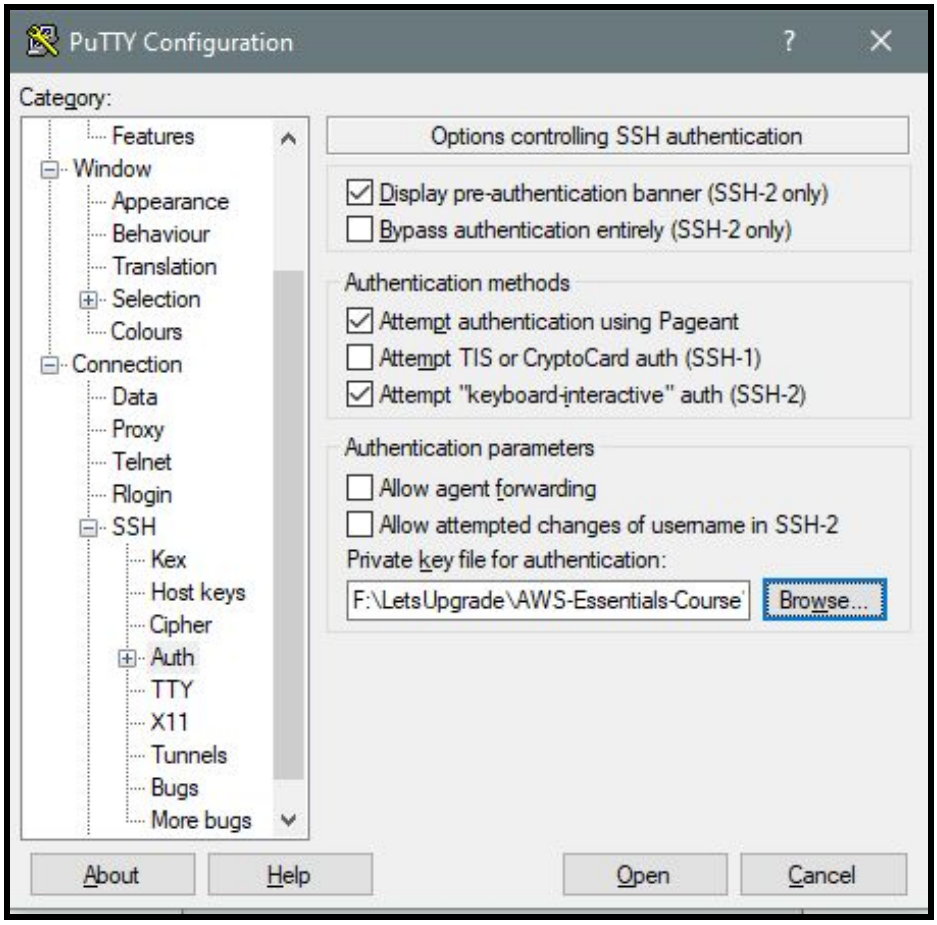
TASK - III: Launch both the Instances using MobaXterm Portable Edition or Putty

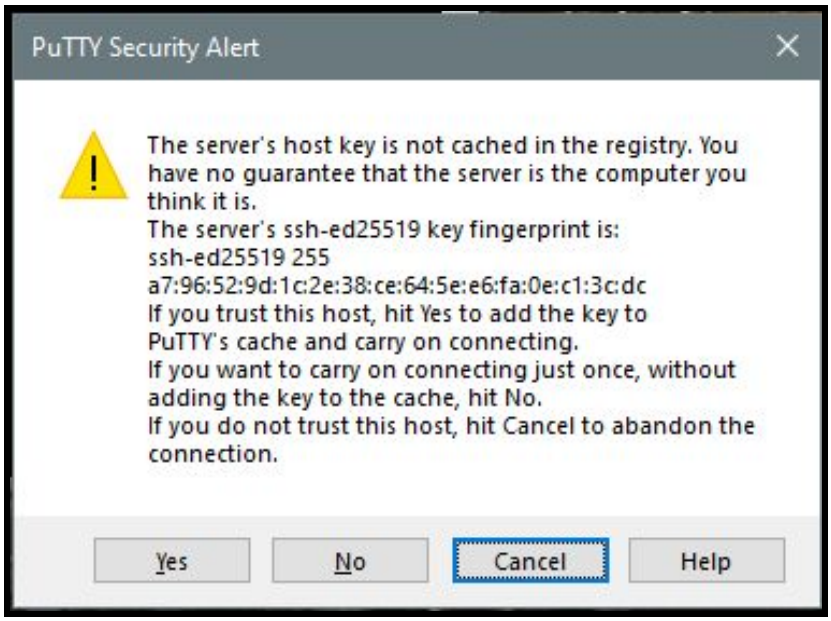
Step - 33:

Firstly launch and host HTML webpage on Instance 1 i.e. Linux1.



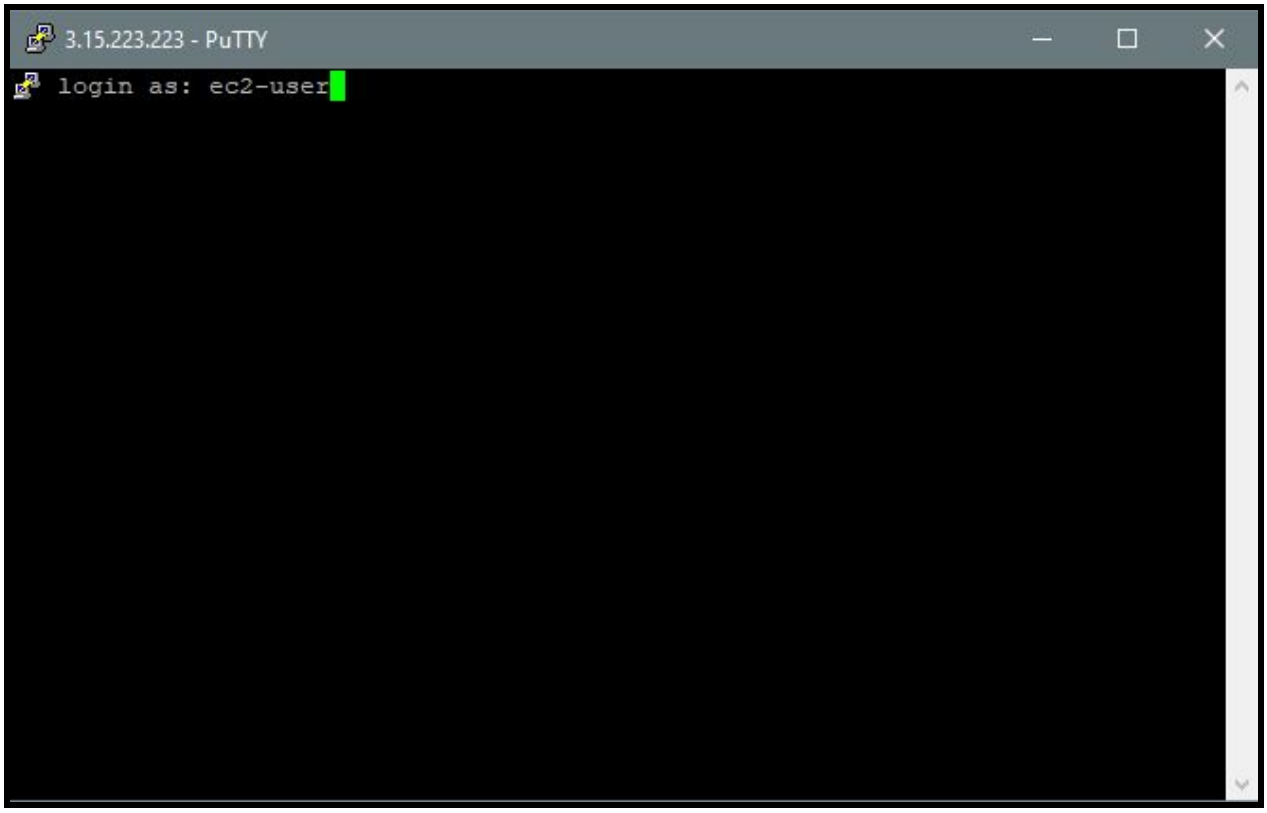



PuTTY Configuration
Category:
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Kex
Host keys
Cipher
Auth
TTY
X11
Tunnels
Bugs
More bugs
Options controlling SSH authentication
Display pre-authentication banner (SSH-2 only)
Bypass authentication entirely (SSH-2 only)
Authentication methods
Attempt authentication using Pageant
Attempt TIS or CryptoCard auth (SSH-1)
Attempt "keyboard-interactive" auth (SSH-2)
Authentication parameters
Allow agent forwarding
Allow attempted changes of username in SSH-2
Private key file for authentication:
F:\LetsUpgrade\AWS-Essentials-Course
Browse...
About
Help
Open
Cancel


PuTTY Security Alert
The server's host key is not cached in the registry. You have no guarantee that the server is the computer you think it is.
The server's ssh-ed25519 key fingerprint is:
ssh-ed25519 255
a7:96:52:9d:1c:2e:38:ce:64:5e:e6:fa:0e:c1:3c:dc
If you trust this host, hit Yes to add the key to PuTTY's cache and carry on connecting.
If you want to carry on connecting just once, without adding the key to the cache, hit No.
If you do not trust this host, hit Cancel to abandon the connection.
Yes
No
Cancel
Help

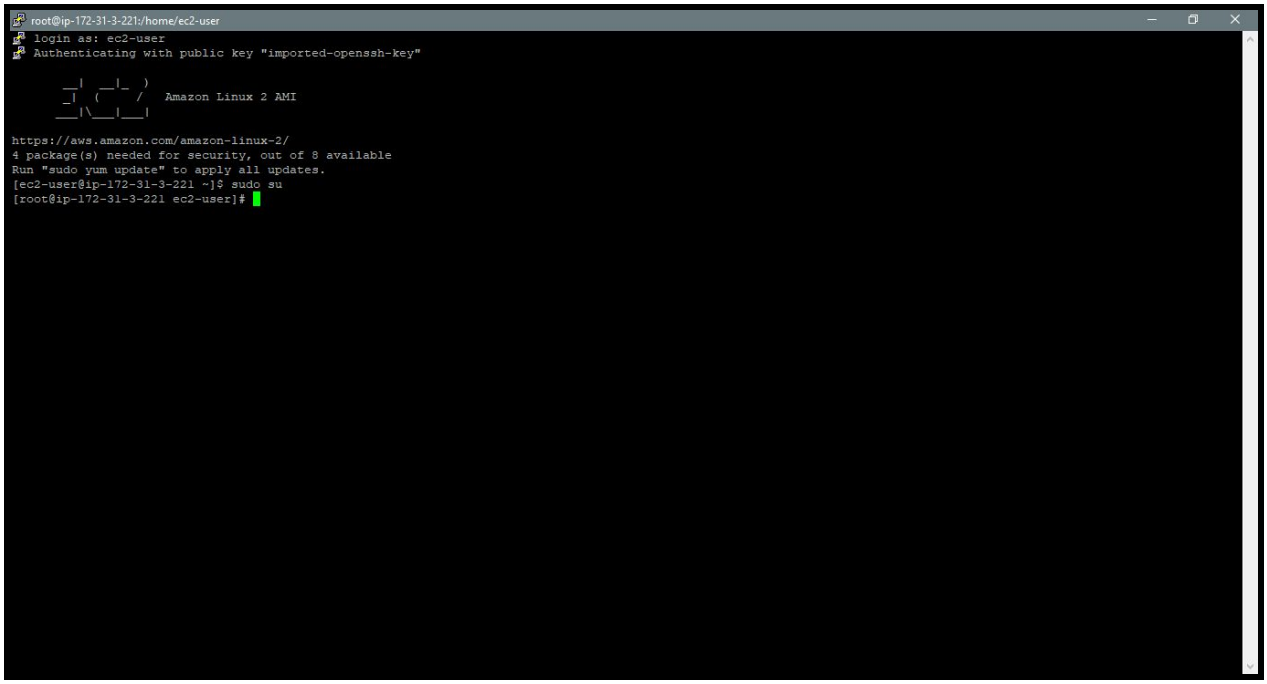
Step - 34:

For Linux Servers, the username is **“ec2-user”.**



Step - 35:

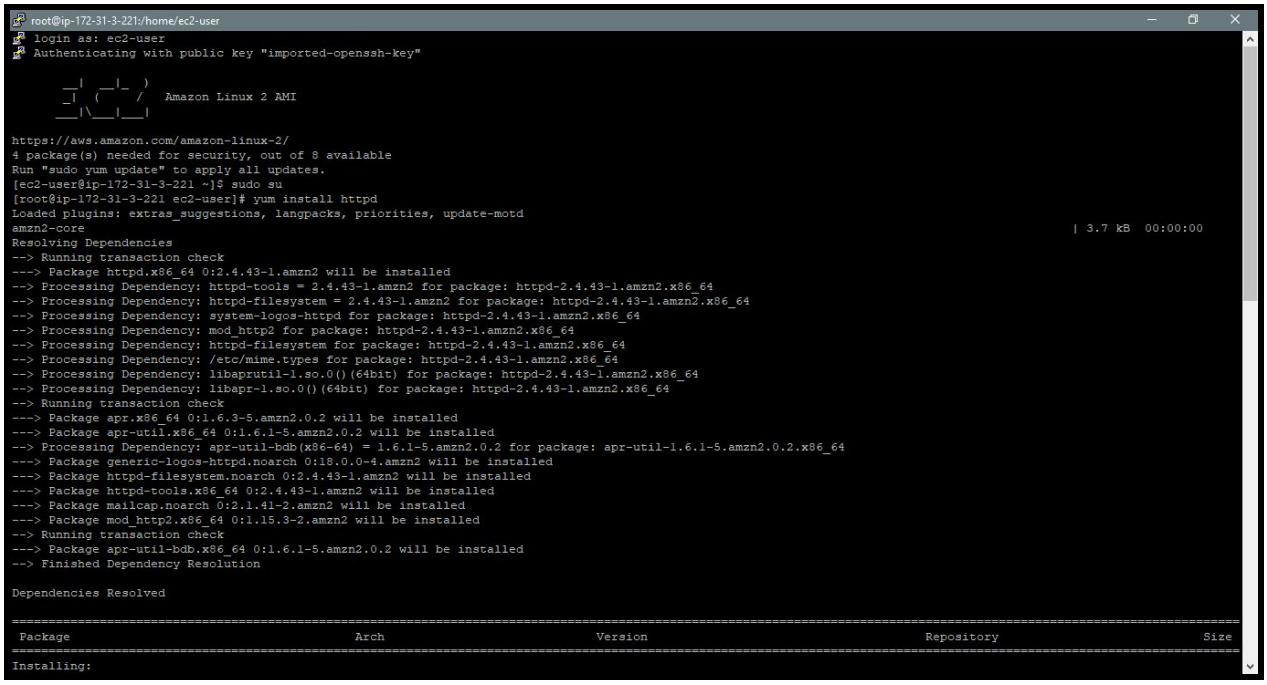
Go to the root user via Terminal using the command as **“sudo su”.**



TASK - IV: Host HTML Login Web page on both the Servers

Step - 36:

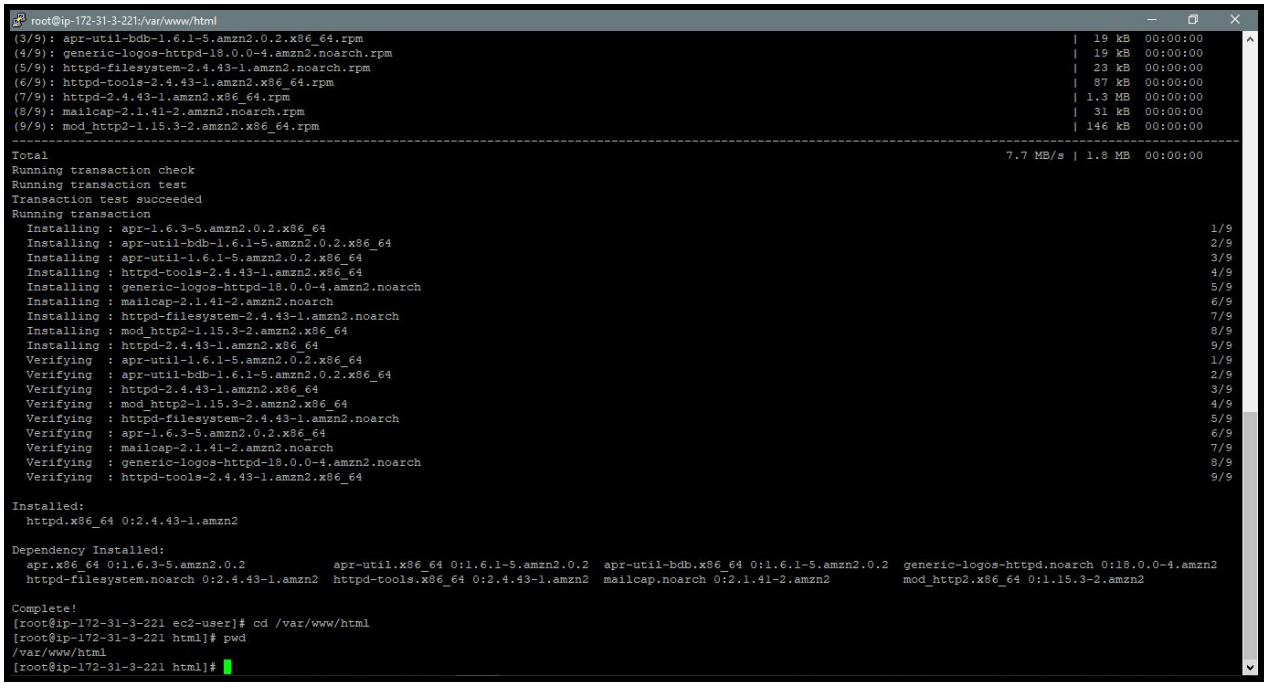
Install httpd for the hosting of a web page on the server using the command **“yum install httpd”.**



Step - 37:

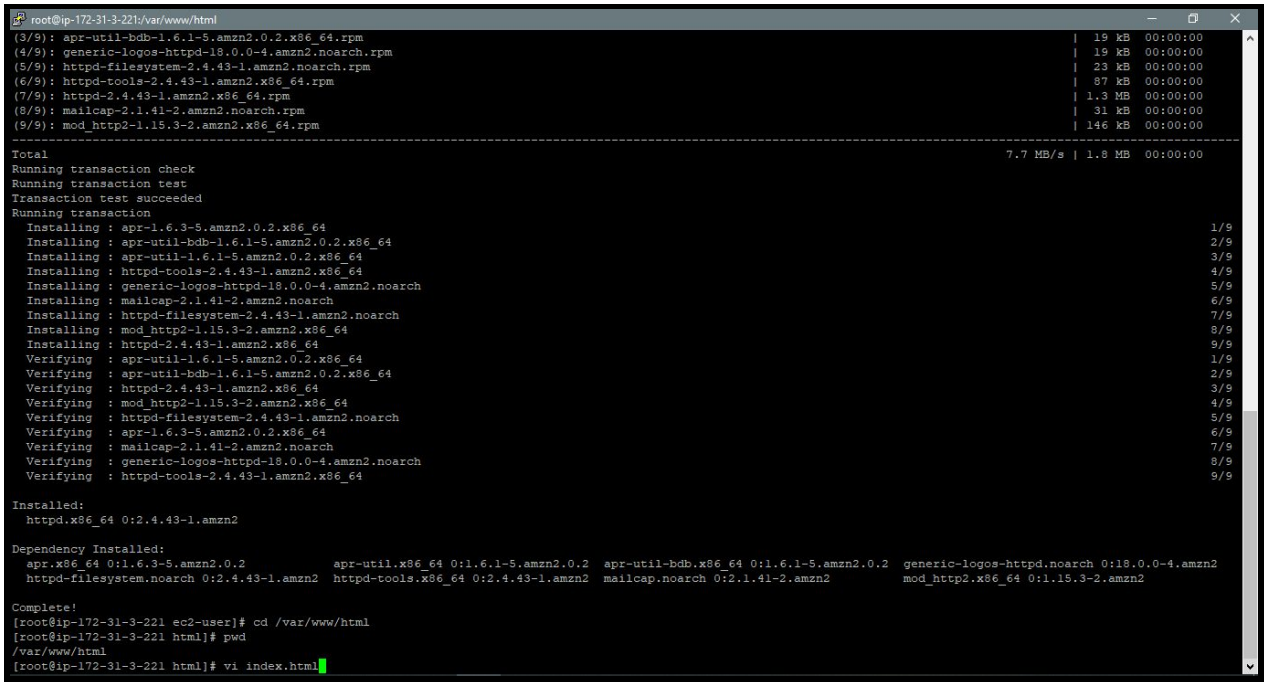
Press ‘y’ from the keyboard to complete the installation for httpd. Then go to location using the command **“cd /var/www/html”.**

Using the command **“pwd”,** we can verify what’s the users present or current working directory.



Step - 38:

Create a file **index.html** using the command **“vi index.html”.**



Step - 39:

Paste the HTML code into the editor and save the file i.e. index.html:

```
<form action="action_page.php" method="post">
	<div class="imgcontainer">
		<img src="img_avatar2.png" alt="Avatar" class="avatar">
	</div>

	<div class="container">
		<label for="uname"><b>Username</b></label>
		<input type="text" placeholder="Enter Username" name="uname" required>

		<label for="psw"><b>Password</b></label>
		<input type="password" placeholder="Enter Password" name="psw" required>

		<button type="submit">Login</button>
		<label>
			<input type="checkbox" checked="checked" name="remember"> Remember me
		</label>
	</div>

	<div class="container" style="background-color:#f1f1f1">
		<button type="button" class="cancelbtn">Cancel</button>
		<span class="psw">Forgot <a href="#">password?</a></span>
	</div>
</form>
```

```
root@ip-172-31-3-221:/var/www/html
<form action="action_page.php" method="post">
        <div class="imgcontainer">
                <img src="img_avatar2.png" alt="Avatar" class="avatar">
        </div>

        <div class="container">
                <label for="uname"><b>Username</b></label>
                <input type="text" placeholder="Enter Username" name="uname" required>

                <label for="psw"><b>Password</b></label>
                <input type="password" placeholder="Enter Password" name="psw" required>

                <button type="submit">Login</button>

                <label>
                        <input type="checkbox" checked="checked" name="remember"> Remember me
                </label>
        </div>

        <div class="container" style="background-color:#f1f1f1">
                <button type="button" class="cancelbtn">Cancel</button>
                <span class="psw">Forgot <a href="#">password?</a></span>
        </div>
</form>
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```

Step - 40:

Using the command **"more index.html",** we can verify the html code written in the respective file.

```
root@ip-172-31-3-221:/var/www/html
  Verifying  : httpd-filesystem-2.4.43-1.amzn2.noarch                                                                 5/9
  Verifying  : apr-1.6.3-5.amzn2.0.2.x86_64                                                                           6/9
  Verifying  : mailcap-2.1.41-2.amzn2.noarch                                                                          7/9
  Verifying  : generic-logos-httpd-18.0.0-4.amzn2.noarch                                                              8/9
  Verifying  : httpd-tools-2.4.43-1.amzn2.x86_64                                                                      9/9

Installed:
  httpd.x86_64 0:2.4.43-1.amzn2

Dependency Installed:
  apr.x86_64 0:1.6.3-5.amzn2.0.2           apr-util.x86_64 0:1.6.1-5.amzn2.0.2  apr-util-bdb.x86_64 0:1.6.1-5.amzn2.0.2  generic-logos-httpd.noarch 0:18.0.0-4.amzn2
  httpd-filesystem.noarch 0:2.4.43-1.amzn2  httpd-tools.x86_64 0:2.4.43-1.amzn2  mailcap.noarch 0:2.1.41-2.amzn2          mod_http2.x86_64 0:1.15.3-2.amzn2

Complete!
[root@ip-172-31-3-221 ec2-user]# cd /var/www/html
[root@ip-172-31-3-221 html]# pwd
/var/www/html
[root@ip-172-31-3-221 html]# vi index.html
[root@ip-172-31-3-221 html]# more index.html
<form action="action_page.php" method="post">
        <div class="imgcontainer">
                <img src="img_avatar2.png" alt="Avatar" class="avatar">
        </div>

        <div class="container">
                <label for="uname"><b>Username</b></label>
                <input type="text" placeholder="Enter Username" name="uname" required>

                <label for="psw"><b>Password</b></label>
                <input type="password" placeholder="Enter Password" name="psw" required>

                <button type="submit">Login</button>

                <label>
                        <input type="checkbox" checked="checked" name="remember"> Remember me
                </label>
        </div>

        <div class="container" style="background-color:#f1f1f1">
                <button type="button" class="cancelbtn">Cancel</button>
                <span class="psw">Forgot <a href="#">password?</a></span>
        </div>
</form>
[root@ip-172-31-3-221 html]#
```

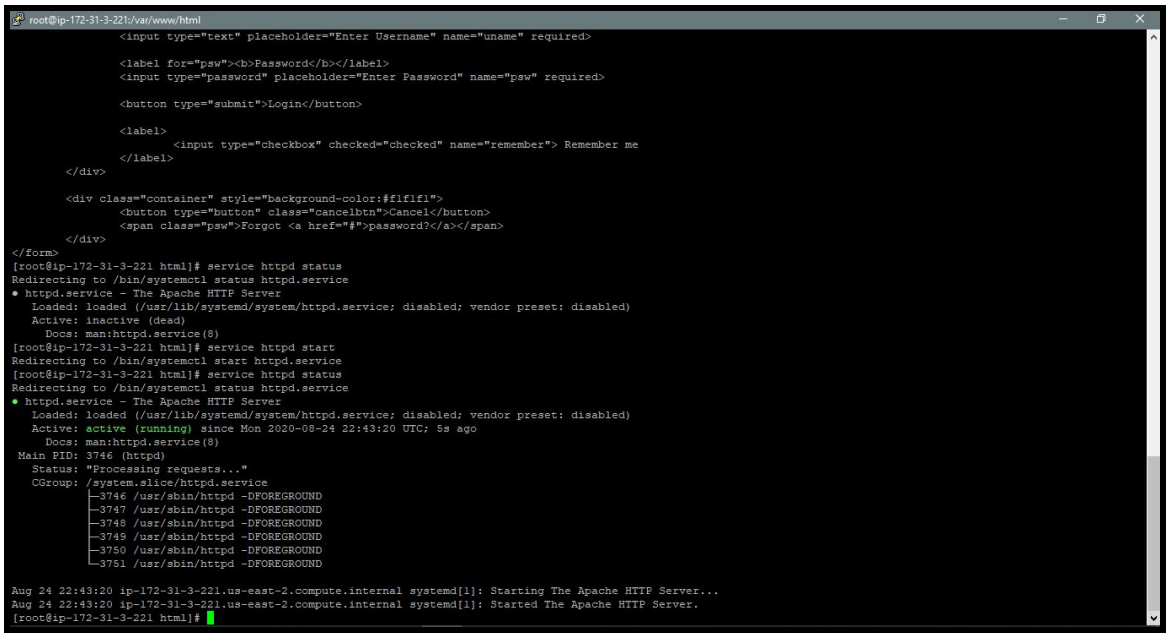
Step - 41:

Use the following command to check the status of “httpd” service and start the service as:

service httpd status

service httpd start

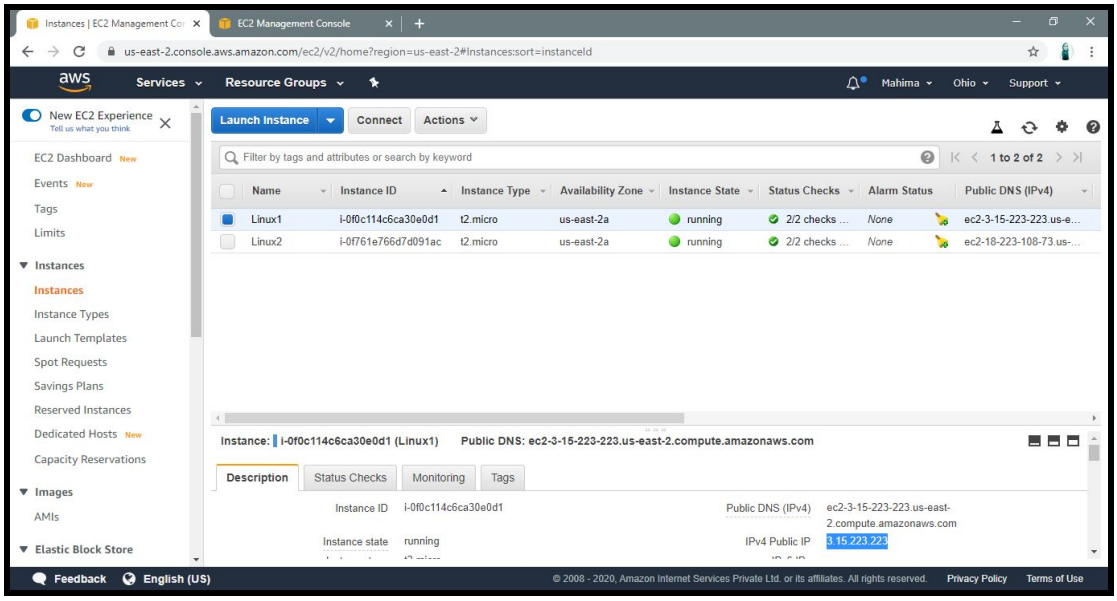
service httpd status

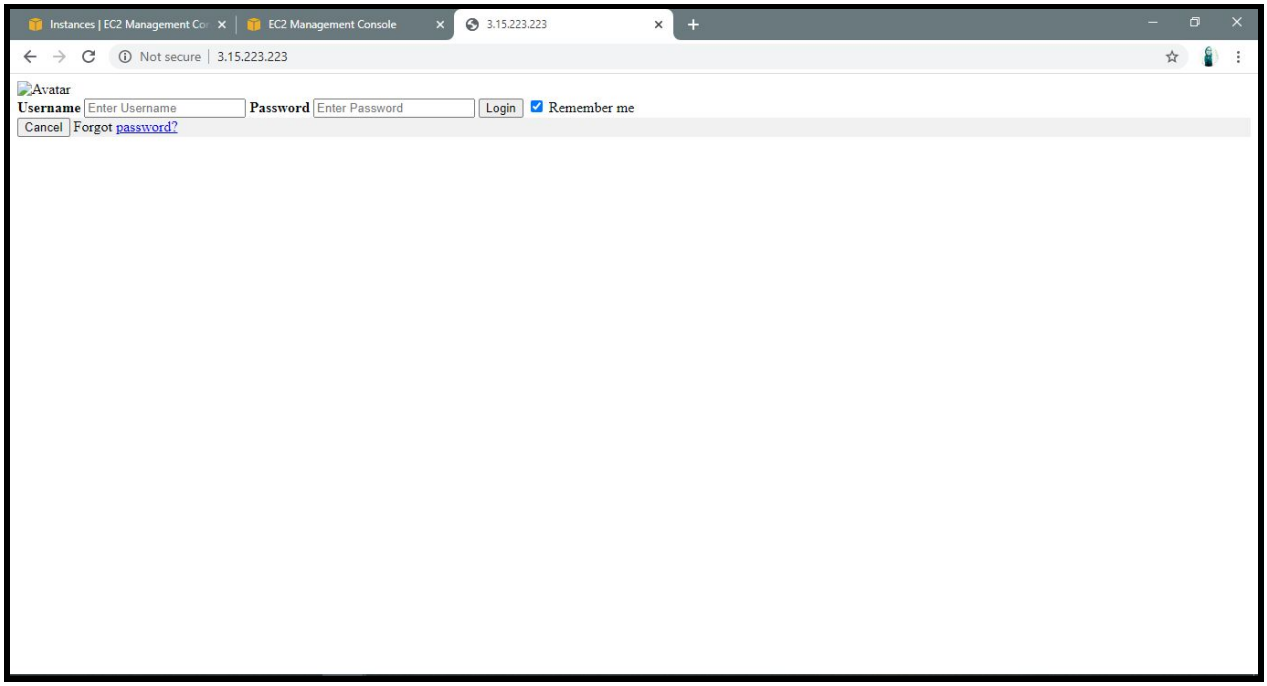


Similarly, deployment can be done on other server html configurations, making note to change the label for Username to UserID and Password to Passkey.

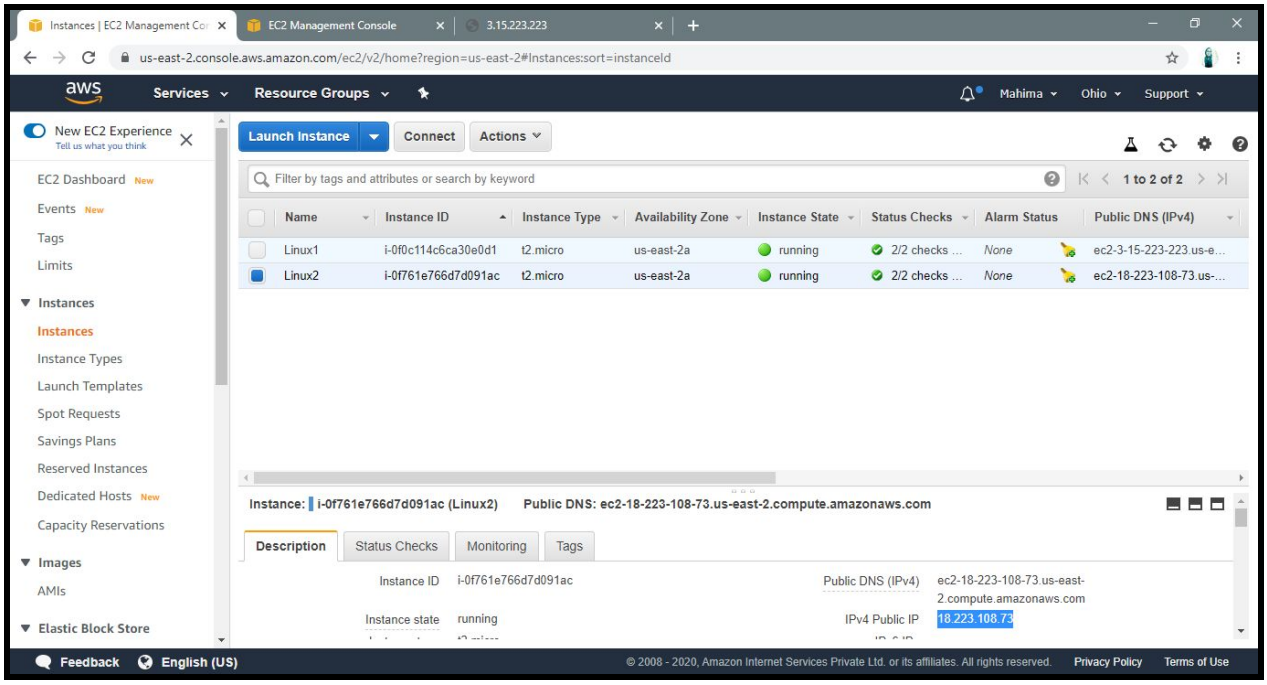
TASK - V: Check if the application is deployed on both servers by copy pasting the Public IP of the servers into the Browser.

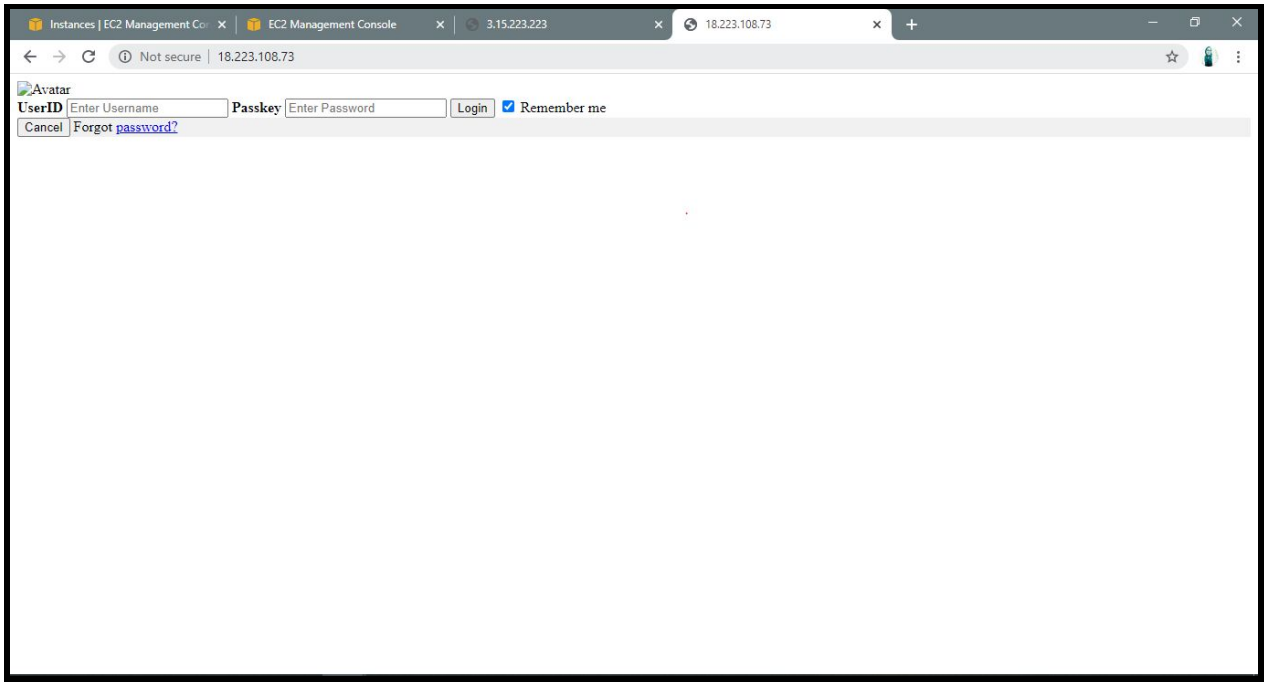
Instance - 1 (Linux1)





Instance - 2 (Linux2)

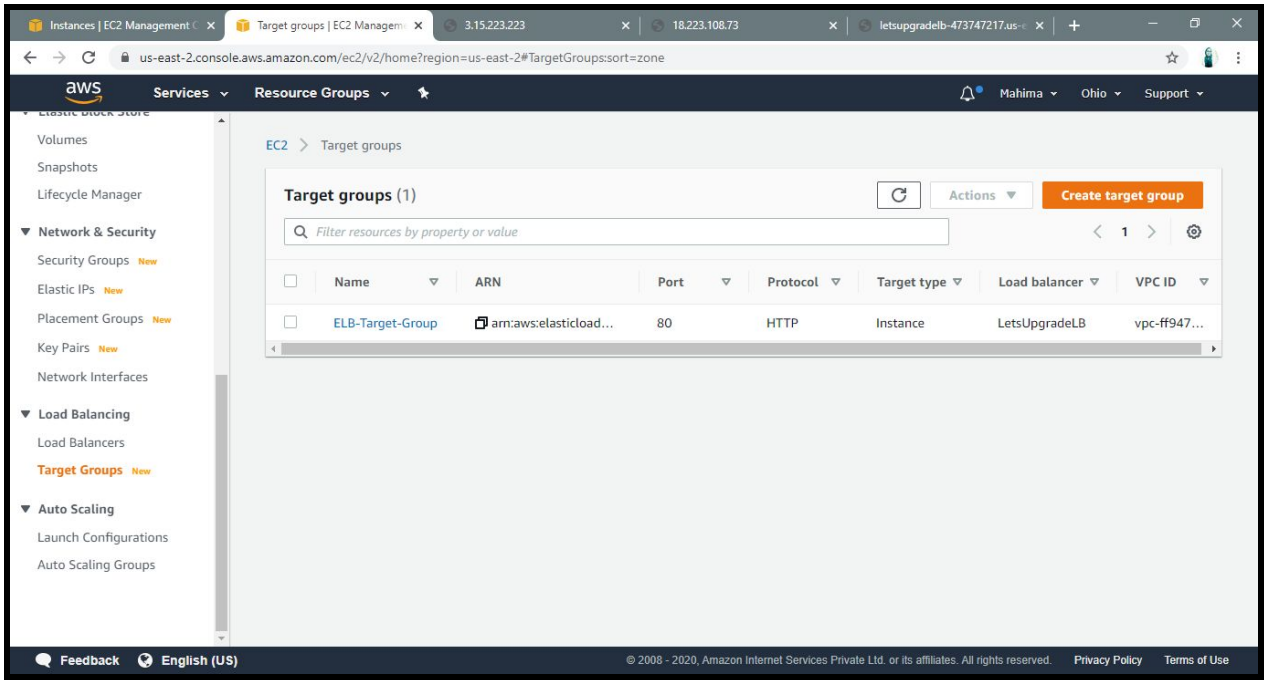




TASK - VI: Check the functionality of ELB (Elastic Load Balancer)

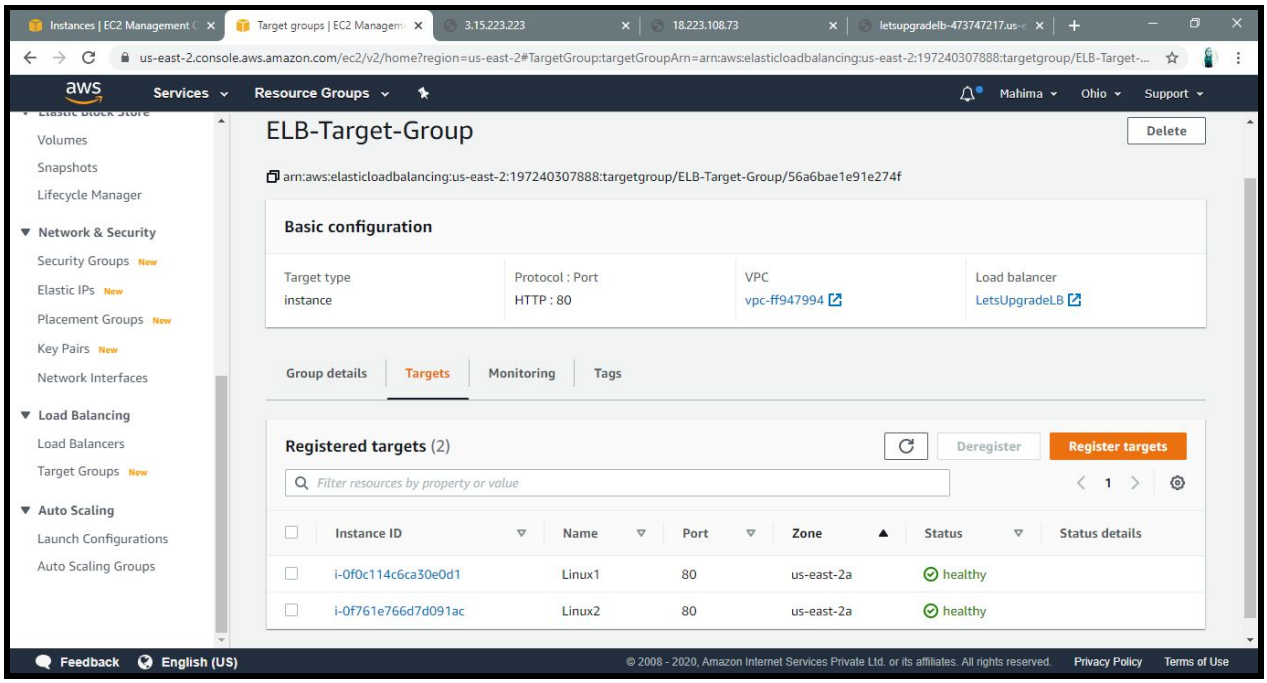
Step - 42:

Go to the Target Groups from the Navigation Pane and click on the Target Group Name to redirect to the detailed description about it.



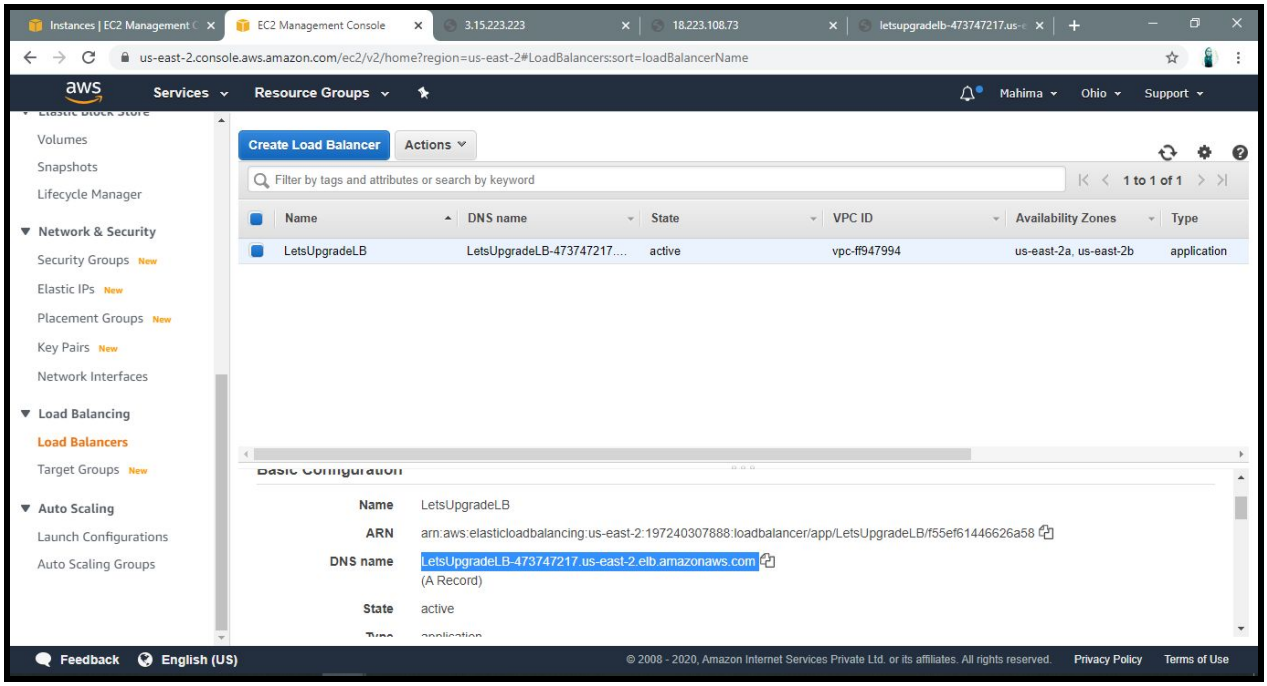
Step - 43:

Under the Target Section below, check for the registered targets status and that should be “healthy”. If not yet “healthy” we might have to wait for sometime to become updated.



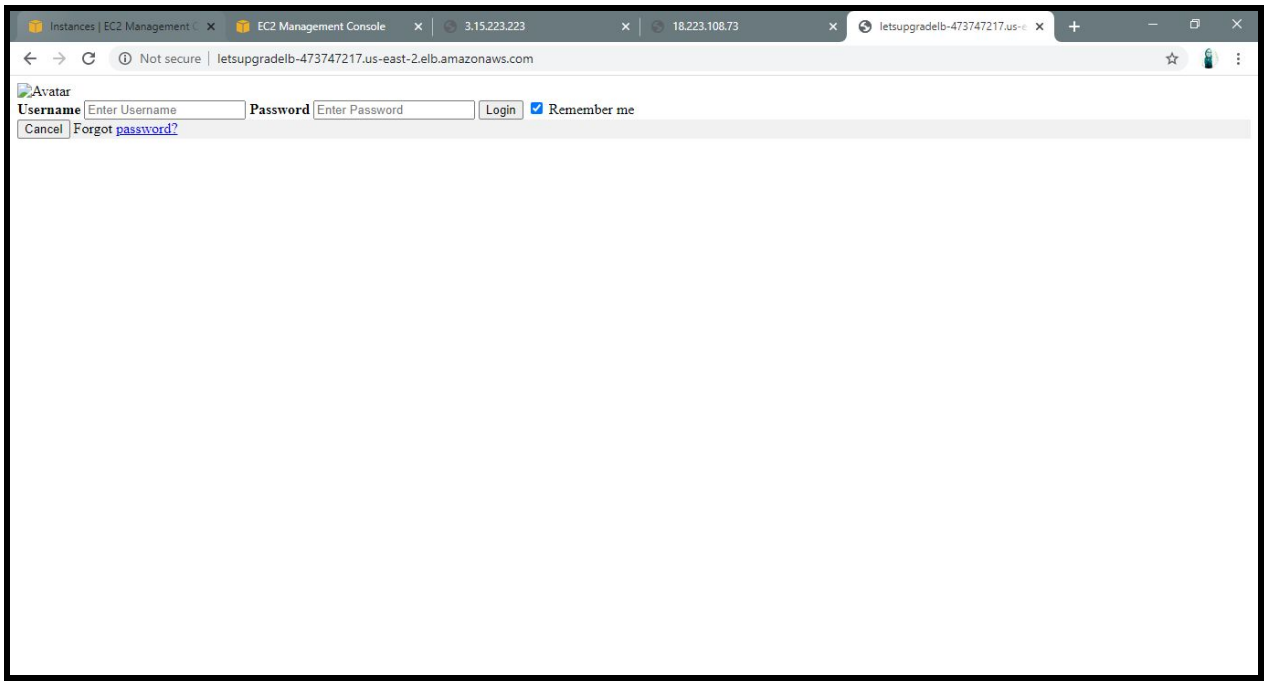
Step - 44:

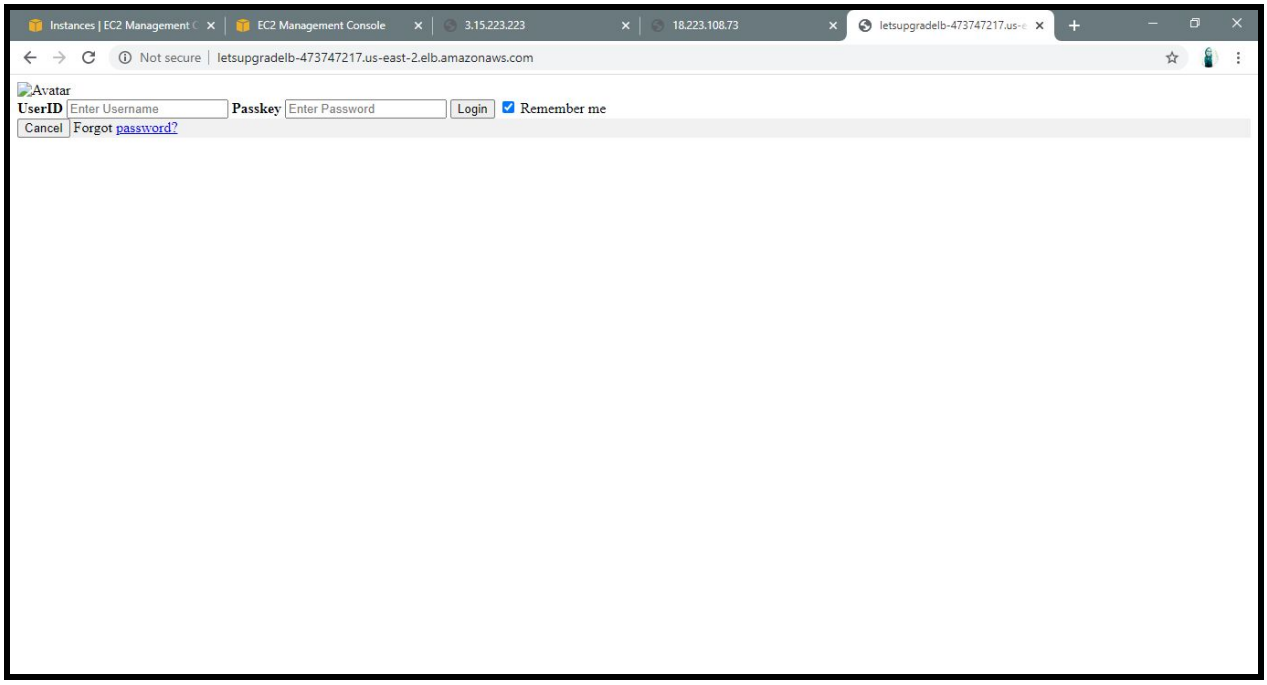
Click on Load Balancers from the Navigation Pane on the left and copy the DNS Name of the created ELB.



Step - 45:

Paste the copied DNS Name on a Browser and refresh it multiple times, it should equalize the load via both the Servers. We can validate it via the label change on the web page displayed on the screen.





Hence, our Elastic Load Balancer is working successfully and the website is seamlessly efficient.