

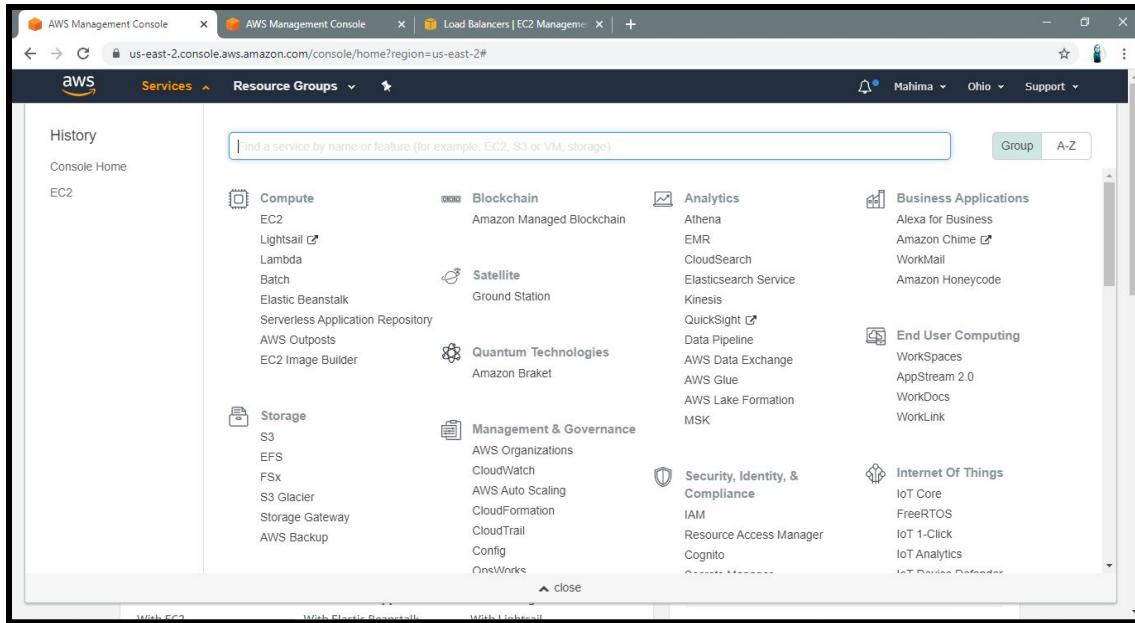
PROJECT - III

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

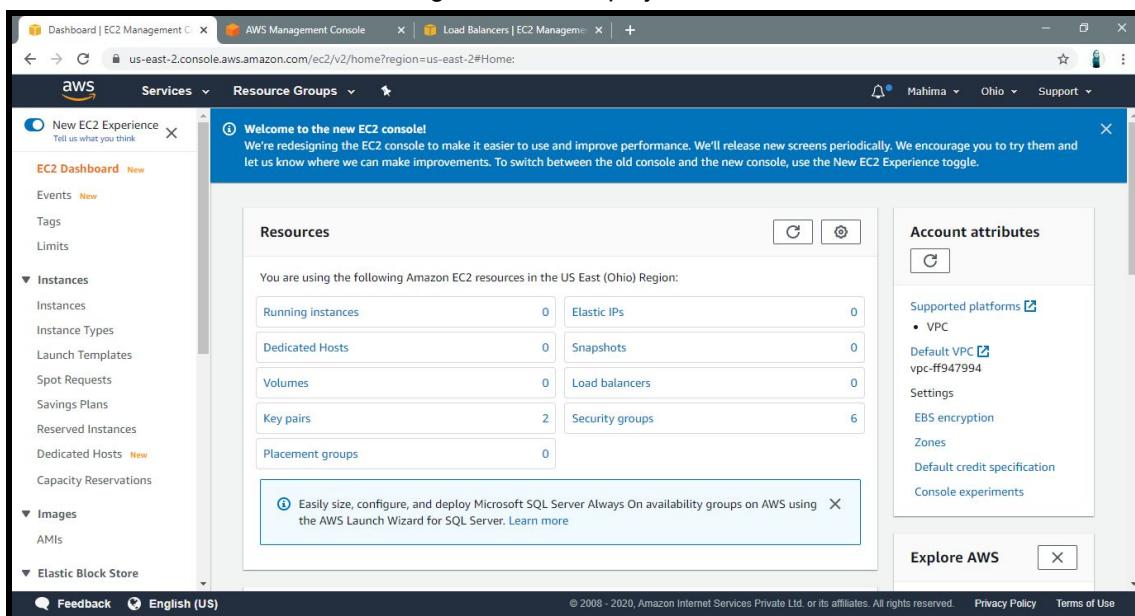
Step - 1:

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.

The screenshot shows the AWS Management Console interface for the EC2 service. The left sidebar has a tree view with 'Instances' selected, which further branches into 'Instances', 'Instance Types', 'Launch Templates', etc. The main content area is titled 'Launch Instance' and contains a message: 'You do not have any running instances in this region.' Below this, there's a link to the 'Getting Started Guide' and a large blue 'Launch Instance' button. At the bottom, it says 'Select an instance above' with three small icons. The footer includes links for 'Feedback', 'English (US)', and copyright information from 2008-2020.

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.

The screenshot shows the 'Launch instance wizard' for the EC2 service, specifically Step 1: Choose an Amazon Machine Image (AMI). The top navigation bar shows the steps: 1. Choose AMI, 2. Choose Instance Type, 3. Configure Instance, 4. Add Storage, 5. Add Tags, 6. Configure Security Group, 7. Review. The main content area has a heading 'Step 1: Choose an Amazon Machine Image (AMI)' and a sub-instruction: 'An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace.' A tooltip for the 'Free tier only' checkbox is visible, explaining that it filters for free usage tier AMIs. Below this, a search bar and a 'Quick Start' sidebar are shown. The main list shows various AMI options, with one highlighted: 'Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-0f4aeaec5b3ce9152'. This entry includes details like 'Virtualization type: hvm', 'ENI Enabled: Yes', and two radio buttons for '64-bit (x86)' and '64-bit (Arm)'. A 'Select' button is next to the highlighted item. The footer includes 'Feedback', 'English (US)', and copyright information.

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 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Search for an AMI by entering a search term e.g. "Windows"

Quick Start

My AMIs

AWS Marketplace

Community AMIs

Free tier only (1)

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-07c8bc5c1ce9598c3 (64-bit x86) / ami-09a67037138f86e67 (64-bit Arm)
Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Gilbc 2.26, Binutils 2.29.1, and the latest software packages through extras.
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes
 64-bit (x86) 64-bit (Arm)
Select

Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-0f4aeac5b3ce9152
The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes
64-bit (x86)
Select

Feedback English (US)

© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use

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 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All instance types Current generation Show/Hide Columns

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
<input checked="" type="checkbox"/>	General purpose	t2.micro	1	1	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.large	2	8	EBS only	-	Low to Moderate	Yes

Cancel Previous Review and Launch Next: Configure Instance Details

Feedback English (US)

© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use

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 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of Instances Launch into Auto Scaling Group [\(i\)](#)

You may want to consider launching these instances into an Auto Scaling Group to help you maintain application availability and for easy scaling in the future. Learn how Auto Scaling can help your application stay healthy and cost effective.

Purchasing option [\(i\)](#) Request Spot instances

Network [\(i\)](#) [Create new VPC](#)

Subnet [\(i\)](#) [Create new subnet](#)
4091 IP Addresses available

Auto-assign Public IP [\(i\)](#)

Placement group [\(i\)](#) Add instance to placement group

Capacity Reservation [\(i\)](#)

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Add Storage](#)

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 3: Configure Instance Details

▼ Network interfaces [\(i\)](#)

Device	Network Interface	Subnet	Primary IP	Secondary IP addresses	IPv6 IPs
eth0	New network interface (i)	subnet-9b0a1ff3 (i)	Auto-assign	Add IP	Add IP

[Add Device](#)

▼ Advanced Details

Metadata accessible [\(i\)](#)

Metadata version [\(i\)](#)

Metadata token response hop limit [\(i\)](#)

User data [\(i\)](#) As text As file Input is already base64 encoded

(Optional)

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Add Storage](#)

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 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more](#) about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encryption
Root	/dev/xvda	snap-00a3ac8046ab803ef	30	General Purpose SSD (gp2)	100 / 3000	N/A	<input checked="" type="checkbox"/>	Not Encrypted

Add New Volume

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. [Learn more](#) about free usage tier eligibility and usage restrictions.

Cancel Previous Review and Launch Next: Add Tags

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 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.
A copy of a tag can be applied to volumes, instances or both.
Tags will be applied to all instances and volumes. [Learn more](#) about tagging your Amazon EC2 resources.

Key	(128 characters maximum)	Value	(256 characters maximum)	Instances	Volumes
This resource currently has no tags					

Choose the Add tag button or [click to add a Name tag](#).
Make sure your IAM policy includes permissions to create tags.

Add Tag (Up to 50 tags maximum)

Cancel Previous Review and Launch Next: Configure Security Group

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”.

The screenshot shows the AWS Management Console interface for launching an instance. The current step is "6. Configure Security Group". The security group "launch-wizard-6" is created with one rule: "All traffic" on port range 0-65535 from "Anywhere" to "0.0.0.0/0". The description is "Servers for Load Balancers". A warning message at the bottom states: "Warning: Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only." At the bottom right, there are "Cancel", "Previous", and "Review and Launch" buttons.

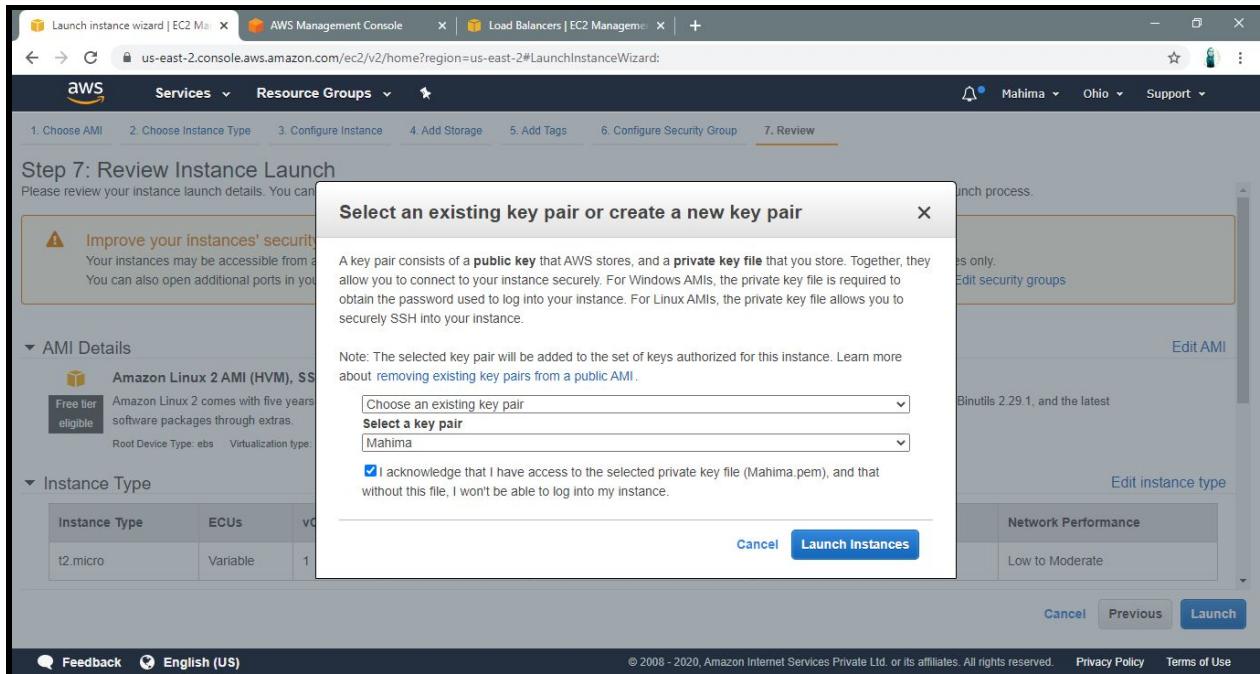
Step - 13:

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.

The screenshot shows the AWS Management Console interface for launching an instance. The current step is "7. Review". The review section includes "AMI Details" (Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-07c8bc5c1ce9598c3) and "Instance Type" (t2.micro). A summary message at the top says: "Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process." A warning message in a box says: "⚠ Improve your instances' security. Your security group, launch-wizard-6, is open to the world. Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow access from known IP addresses only. You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. [Edit security groups](#)". At the bottom right, there are "Cancel", "Previous", and "Launch" buttons.

Step - 14:

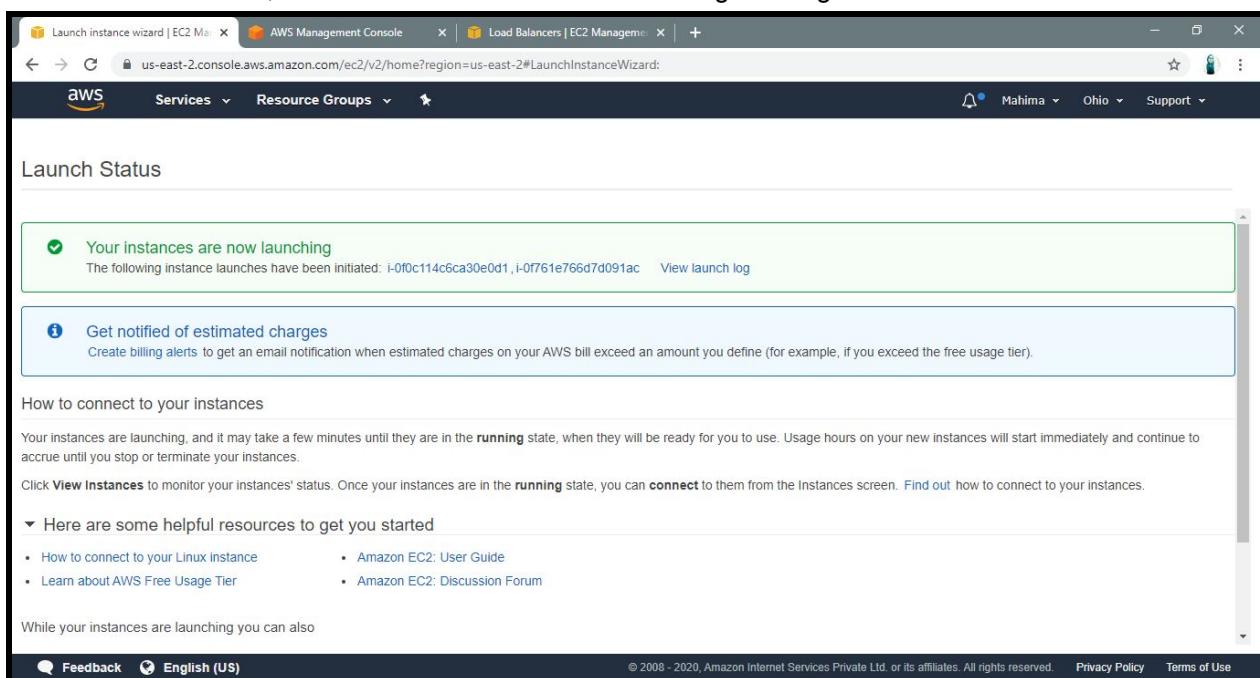
Now, we have to “Select an existing key pair or create a new key pair” as a last step in the process of launching an Instance on AWS. From the dropdown, select “Choose an existing key pair” as we already have it in our system from our previous launch of a Windows Instance. Enable the acknowledgment checkbox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



The screenshot shows the AWS Management Console interface for launching an instance. The main page displays 'Step 7: Review Instance Launch'. On the left, there are sections for 'AMI Details' (Amazon Linux 2 AMI (HVM), SSD Volume Type) and 'Instance Type' (t2.micro). A central modal window titled 'Select an existing key pair or create a new key pair' is open, showing a dropdown menu with 'Select a key pair' and 'Mahima' selected. Below the dropdown is a checkbox for acknowledging access to the private key file. At the bottom right of the modal are 'Cancel' and 'Launch Instances' buttons. The background shows other tabs like 'Launch instance wizard | EC2 Management Console' and 'AWS Management Console'.

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.



The screenshot shows the AWS Management Console interface for monitoring instance launches. The main header says 'Launch Status'. Below it, a green message box states 'Your instances are now launching' with instance IDs i-0fc114c6ca30e0d1 and i-0f761e766d7d091ac, and a link to 'View launch log'. Below this is a blue info box about estimated charges. Further down, there are sections for connecting to instances and helpful resources. The background shows other tabs like 'Launch instance wizard | EC2 Management Console' and 'AWS Management Console'.

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”.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
	i-0f0c114c6ca30e0d1	t2.micro	us-east-2a	running	Initializing	None	ec2-3-15-223-223.us-e...
	i-0f761e766d7d091ac	t2.micro	us-east-2a	running	Initializing	None	ec2-18-223-108-73.us...

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.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
Linux1	i-0f0c114c6ca30e0d1	t2.micro	us-east-2a	running	2/2 checks ...	None	ec2-3-15-223-223.us-e...
Linux2	i-0f761e766d7d091ac	t2.micro	us-east-2a	running	2/2 checks ...	None	ec2-18-223-108-73.us...

The screenshot shows the AWS Management Console interface for the EC2 service. The main area displays a table of running instances. The table includes columns for Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS (IPv4). Two instances are listed: Linux1 (Instance ID: i-0f0c114c6ca30e0d1, t2.micro, us-east-2a, running) and Linux2 (Instance ID: i-0f761e766d7d091ac, t2.micro, us-east-2a, running). The Public DNS for Linux2 is ec2-18-223-108-73.us-east-2.compute.amazonaws.com. The left sidebar contains links for EC2 Dashboard, Events, Tags, Limits, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts (New), Capacity Reservations, Images (AMIs), and Elastic Block Store.

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

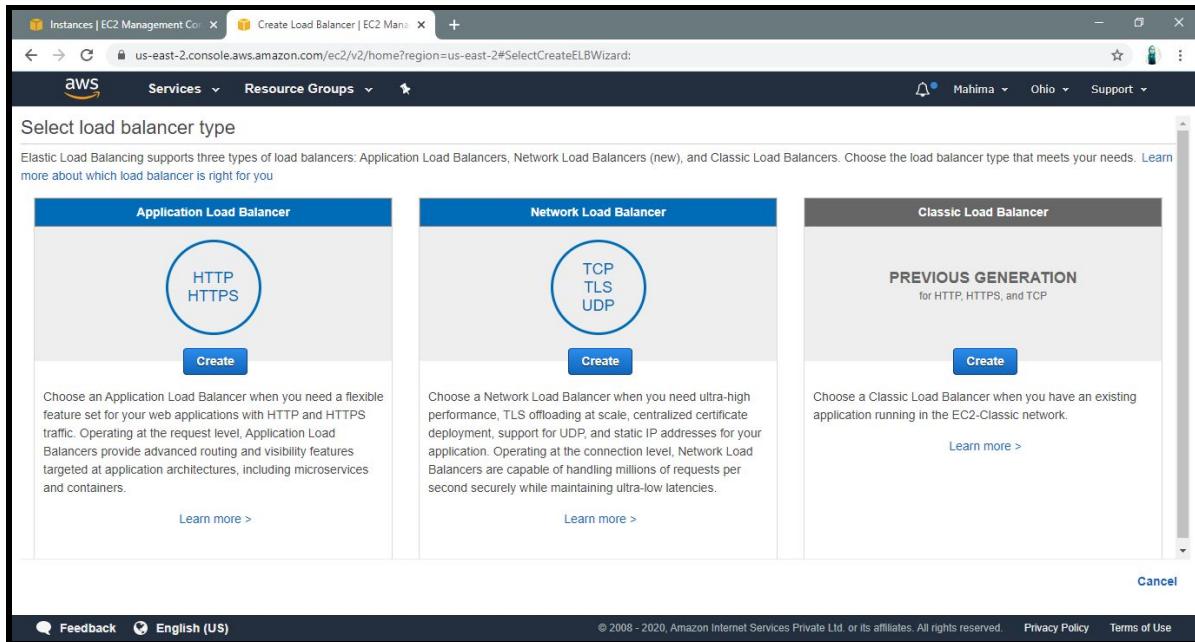
Step - 18:

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.

The screenshot shows the AWS Management Console interface for the Load Balancers service. The main area displays a table with the message "You do not have any load balancers in this region." The left sidebar contains links for Volumes, Snapshots, Lifecycle Manager, Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs), Load Balancing (selected), Target Groups (New), Auto Scaling (Launch Configurations, Auto Scaling Groups), and Feedback. The language is set to English (US).

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.,

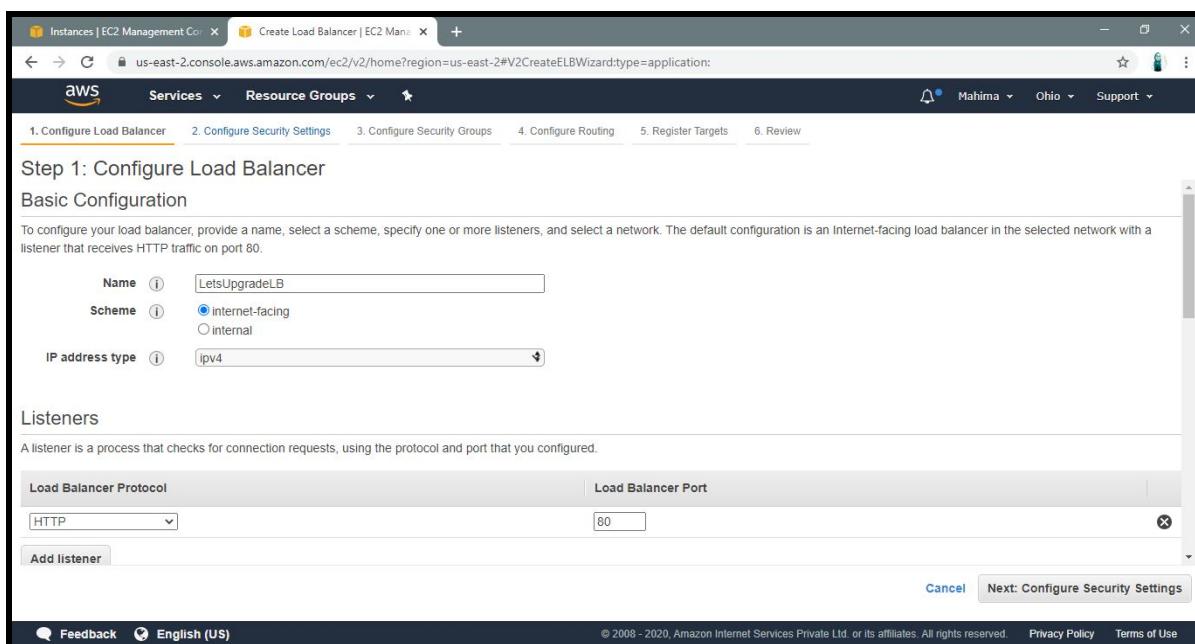
Name: LetsUpgradeLB

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)

The screenshot shows the 'Create Load Balancer' wizard at Step 1: Configure Load Balancer. In the 'Availability Zones' section, 'us-east-2a' and 'us-east-2b' are selected, each associated with a specific subnet. Other availability zones ('us-east-2c') are listed but not selected.

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.

The screenshot shows the 'Create Load Balancer' wizard at Step 1: Configure Load Balancer. The 'Add-on services' section includes an option for 'AWS Global Accelerator'. The 'Create an accelerator' checkbox is unselected. A note below explains that the Accelerator will be created with a customizable name and can be managed via the Global Accelerator console. An input field for 'Accelerator name' is present, with a note stating 'Maximum 64 characters. Letters and numbers only.'

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”.

The screenshot shows the AWS Create Load Balancer wizard at Step 2: Configure Security Settings. A warning message in a yellow box states: "⚠ Improve your load balancer's security. Your load balancer is not using any secure listener. If your traffic to the load balancer needs to be secure, use the HTTPS protocol for your front-end connection. You can go back to the first step to add/configure secure listeners under Basic Configuration section. You can also continue with current settings." Below the message, there are tabs for 1. Configure Load Balancer, 2. Configure Security Settings (which is selected), 3. Configure Security Groups, 4. Configure Routing, 5. Register Targets, and 6. Review. At the bottom, there are buttons for Cancel, Previous, Next: Configure Security Groups, Feedback, English (US), and links to Privacy Policy and Terms of Use.

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”.

The screenshot shows the AWS Create Load Balancer wizard at Step 3: Configure Security Groups. It asks to assign a security group, with the option to create a new one selected. A new security group named "load-balancer-wizard-1" is being created. The description field shows "load-balancer-wizard-1 created on 2020-08-25T03:52:19.455+05:30". Below this, a table lists a single rule: "Type: All traffic", "Protocol: All", "Port Range: 0 - 65535", and "Source: Anywhere". An "Add Rule" button is available. At the bottom, there are buttons for Cancel, Previous, Next: Configure Routing, Feedback, English (US), and links to Privacy Policy and Terms of Use.

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 4: Configure Routing

Your load balancer routes requests to the targets in this target group using the protocol and port that you specify, and performs health checks on the targets using these health check settings. Note that each target group can be associated with only one load balancer.

Target group

Target group: New target group

Name: ELB-Target-Group

Target type: Instance

Protocol: HTTP

Port: 80

Health checks

Protocol: HTTP

Path: /

Cancel **Previous** **Next: Register Targets**

Step - 26:

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

Step 4: Configure Routing

Protocol: HTTP

Port: 80

Health checks

Protocol: HTTP

Path: /

Advanced health check settings

Port: traffic port

Healthy threshold: 5

Unhealthy threshold: 2

Timeout: 5 seconds

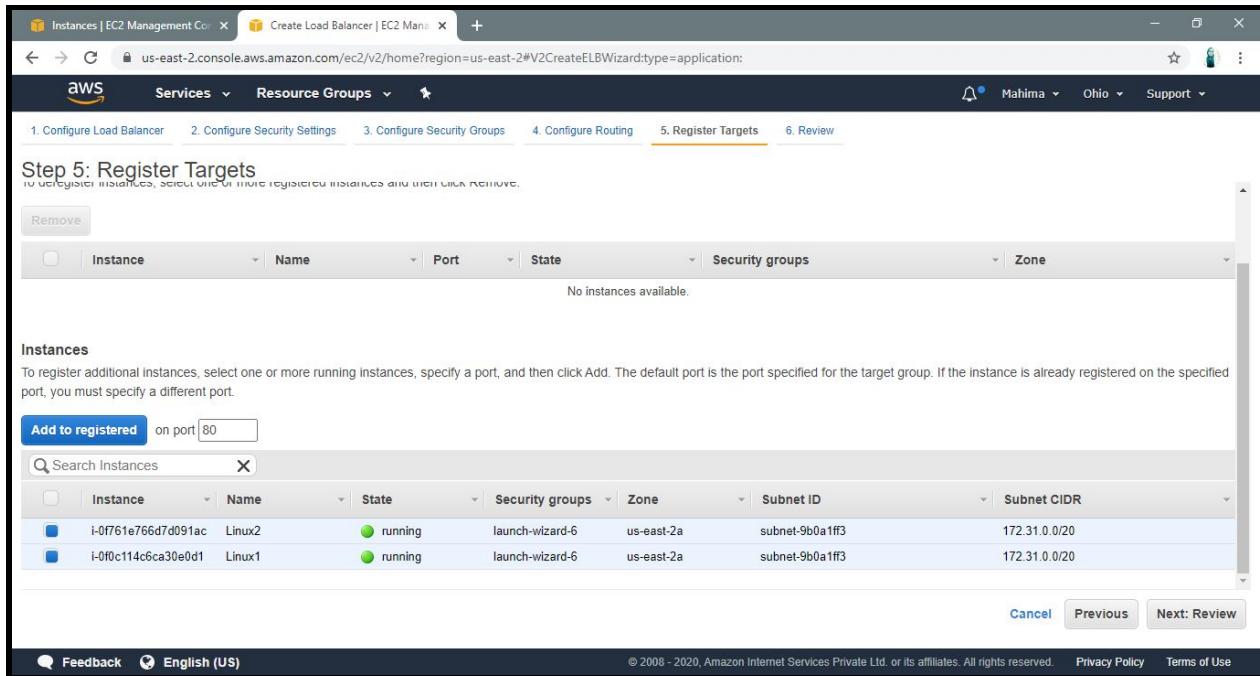
Interval: 30 seconds

Success codes: 200

Cancel **Previous** **Next: Register Targets**

Step - 27:

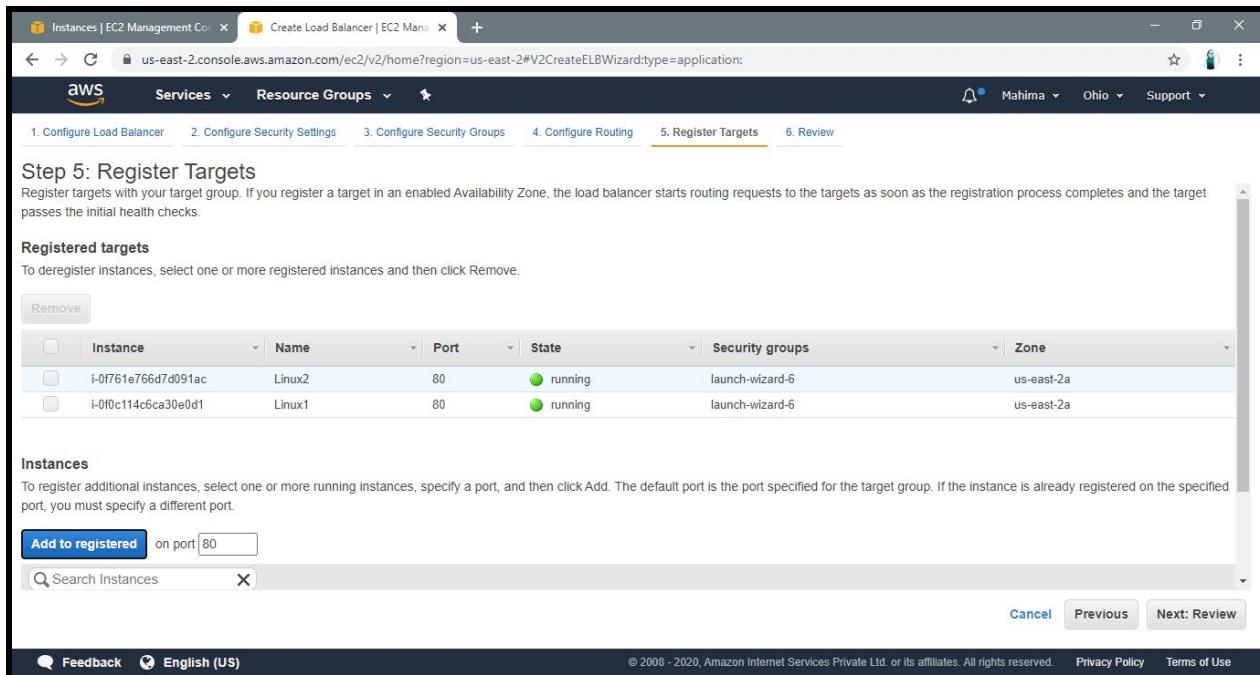
Select both the Instances in the Gridlist i.e. Linux1 and Linux2 and thereafter click “Add to registered” button.



The screenshot shows the AWS EC2 Management Console with the URL <https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#V2CreateELBWizard:type=application>. The page is titled "Step 5: Register Targets". It displays a table of instances with columns: Instance, Name, Port, State, Security groups, and Zone. Two instances are listed: "Linux2" and "Linux1", both in the "running" state. Below the table is a search bar labeled "Search Instances". A blue button labeled "Add to registered" is visible, with a dropdown menu showing "on port 80". At the bottom right are buttons for "Cancel", "Previous", and "Next: Review".

Step - 28:

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



The screenshot shows the AWS EC2 Management Console with the same URL as the previous step. The "Registered targets" section now lists both "Linux2" and "Linux1" under the "Instances" column. The "Instances" section below it also lists "Linux2" and "Linux1". The "Add to registered" button is highlighted again. The bottom right buttons are "Cancel", "Previous", and "Next: Review".

Step - 29:

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

The screenshot shows the AWS Create Load Balancer wizard at Step 6: Review. The page displays the configuration details for a new Application Load Balancer named 'LetsUpgradeLB'. The configuration includes:

- Name:** LetsUpgradeLB
- Scheme:** Internet-facing
- Listeners:** Port:80 - Protocol:HTTP
- IP address type:** ipv4
- VPC:** vpc-f947994
- Subnets:** subnet-9b0a1ff3, subnet-ba8ccfc0
- Tags:** None listed

Below the main configuration, there are sections for Security groups (containing 'load-balancer-wizard-1') and Routing (containing a target group 'ELB-Target-Group' with port 80). At the bottom right, there are 'Cancel', 'Previous', and 'Create' buttons, along with links for Feedback, English (US), Privacy Policy, and Terms of Use.

Step - 30:

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

The screenshot shows the AWS Load Balancer Creation Status screen. It displays a success message: "Successfully created load balancer. Load balancer LetsUpgradeLB was successfully created. Note: It might take a few minutes for your load balancer to be fully set up and ready to route traffic, and for the targets to complete the registration process and pass the initial health checks." Below this message, under "Suggested next steps", are two items:

- Discover other services that you can integrate with your load balancer. Visit the [Integrated services](#) tab within LetsUpgradeLB
- Consider using AWS Global Accelerator to further improve the availability and performance of your applications. [AWS Global Accelerator console](#)

At the bottom right, there is a '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.

Name	DNS name	State	VPC ID	Availability Zones	Type
LetsUpgradeLB	LetsUpgradeLB-473747217....	provisioning	vpc-f947994	us-east-2a, us-east-2b	application

Step - 32:

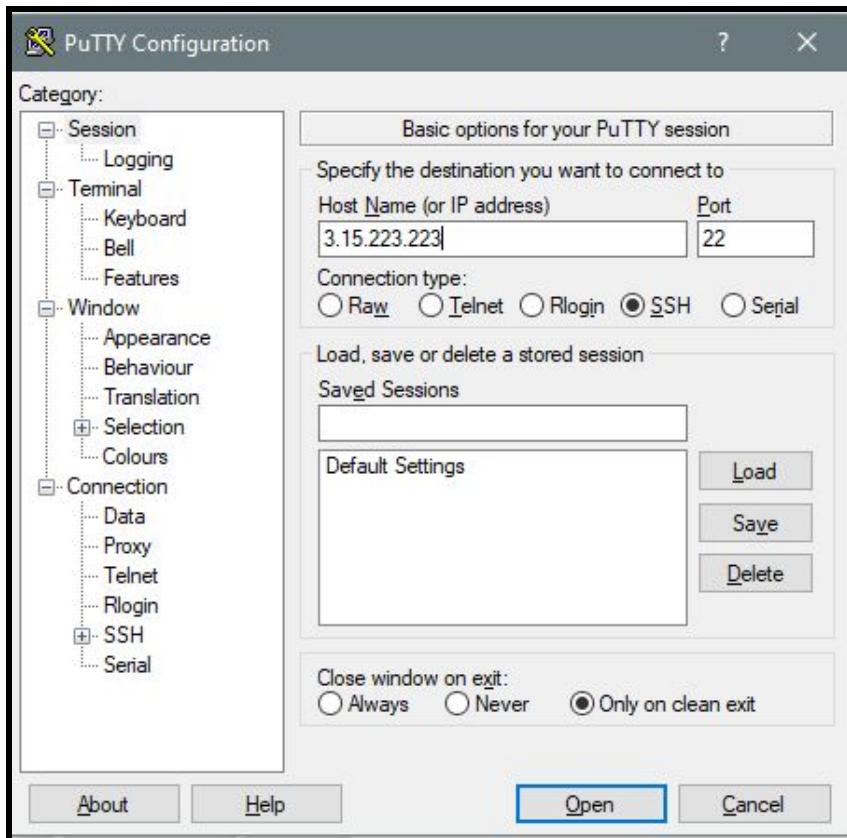
After pasting the DNS in a Browser, it shows us “502 Bad Gateway” as we have not yet hosted a web page on the servers. We’ll hit it again after hosting the web pages on the Servers.

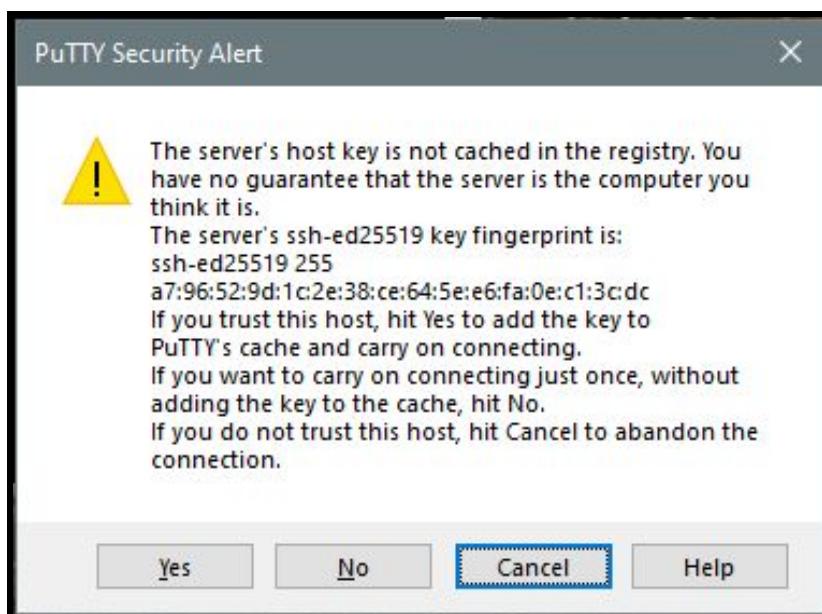
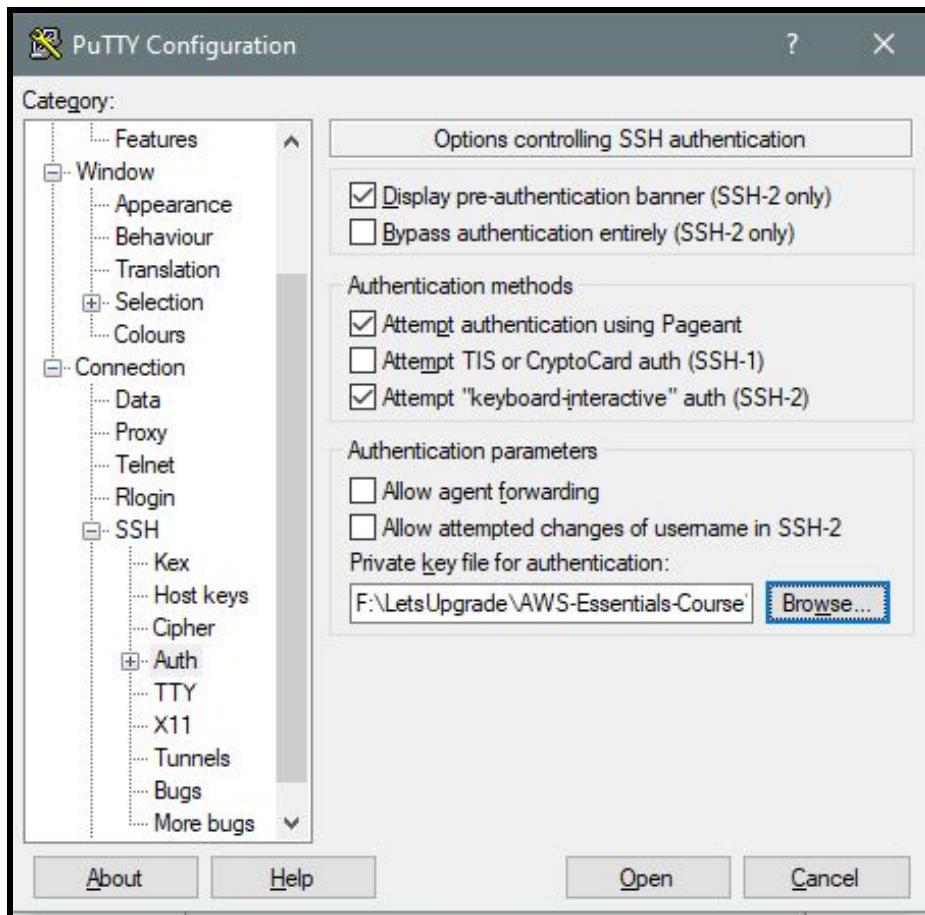
TASK - III: Launch both the Instances using MobaXterm Portable Edition or Putty

Step - 33:

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

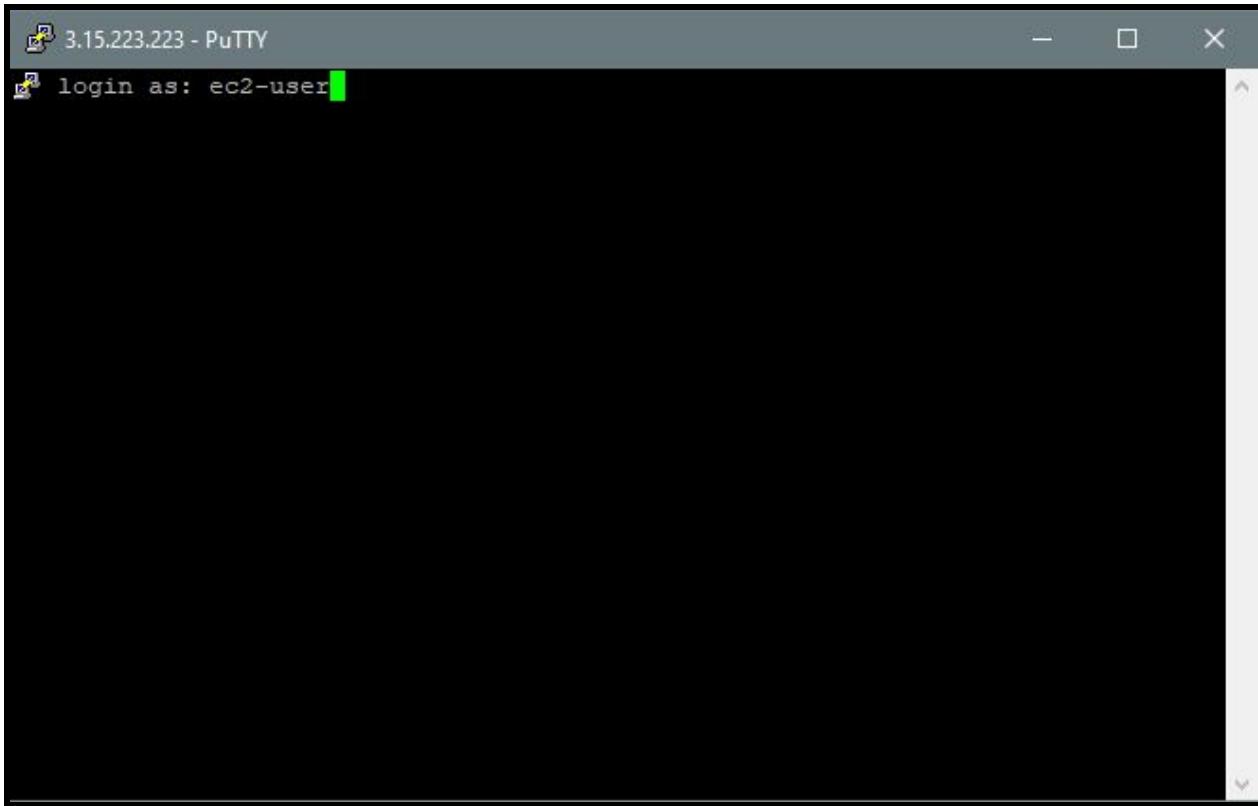
The screenshot shows the AWS EC2 Management Console interface. On the left, there's a navigation sidebar with options like New EC2 Experience, EC2 Dashboard, Events, Tags, Limits, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, and Elastic Block Store. The main content area displays a table of instances. The table has columns for Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS (IPv4). There are two entries: Linux1 (Instance ID: i-0f0c114c6ca30e0d1, Instance Type: t2.micro, State: running, Public DNS: ec2-3-15-223-223.us-east-2.compute.amazonaws.com) and Linux2 (Instance ID: i-0f761e766d7d091ac, Instance Type: t2.micro, State: running, Public DNS: ec2-18-223-108-73.us-east-2.compute.amazonaws.com). Below the table, a detailed view for Linux1 is shown with tabs for Description, Status Checks, Monitoring, and Tags. The Description tab shows the instance ID, public DNS, instance state (running), and IPv4 public IP (3.15.223.223).





Step - 34:

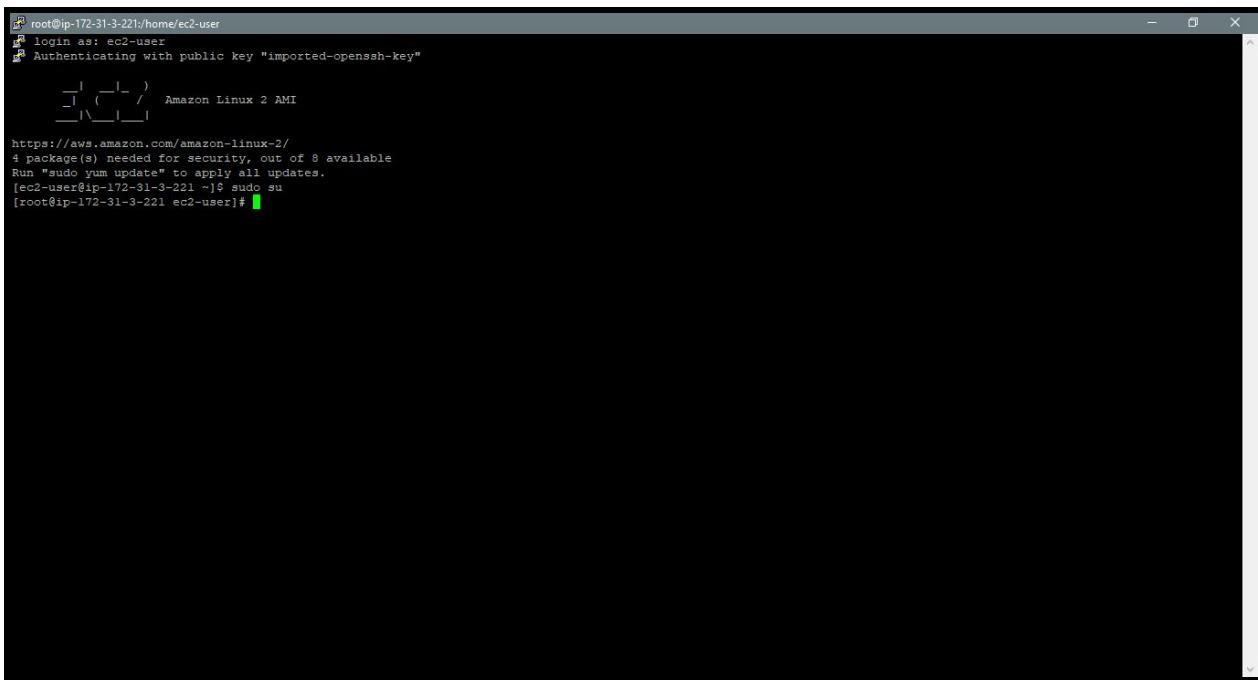
For Linux Servers, the username is “**ec2-user**”.



A screenshot of a PuTTY terminal window. The title bar says "3.15.223.223 - PuTTY". The session name is "login as: ec2-user". The terminal window is mostly blank, indicating no further output.

Step - 35:

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



```
root@ip-172-31-3-221:/home/ec2-user
[ec2-user@ip-172-31-3-221 ~]$ sudo su
[root@ip-172-31-3-221 ec2-user]#
```

A screenshot of a terminal window. The title bar shows the user is root. The command "sudo su" has been entered and executed, switching the user to root. The prompt now shows the root user's directory: "/root".

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.

```

root@ip-172-31-3-221:/var/www/html| - □
(3/9): apr-util-bdb-1.6.1-5.amzn2.0.2.x86_64.rpm | 19 kB 00:00:00
(4/9): generic-logos-htpd-18.0.0-4.amzn2.noarch.rpm | 19 kB 00:00:00
(5/9): httpd-filesystem-2.4.43-1.amzn2.noarch.rpm | 23 kB 00:00:00
(6/9): httpd-tools-2.4.43-1.amzn2.x86_64.rpm | 87 kB 00:00:00
(7/9): httpd-2.4.43-1.amzn2.x86_64.rpm | 1.3 MB 00:00:00
(8/9): mailcap-2.1.41-2.amzn2.noarch.rpm | 31 kB 00:00:00
(9/9): mod_http2-1.15.3-2.amzn2.x86_64.rpm | 146 kB 00:00:00

Total 7.7 MB/s | 1.6 MB 00:00:00

Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : apr-1.6.3-5.amzn2.0.2.x86_64 1/9
  Installing : apr-util-bdb-1.6.1-5.amzn2.0.2.x86_64 2/9
  Installing : apr-util-1.6.1-5.amzn2.0.2.x86_64 3/9
  Installing : httpd-tools-2.4.43-1.amzn2.x86_64 4/9
  Installing : generic-logos-htpd-18.0.0-4.amzn2.noarch 5/9
  Installing : mailcap-2.1.41-2.amzn2.noarch 6/9
  Installing : httpd-filesystem-2.4.43-1.amzn2.noarch 7/9
  Installing : mod_http2-1.15.3-2.amzn2.x86_64 8/9
  Installing : apr-2.4.43-1.amzn2.x86_64 9/9
  Verifying : apr-util-1.6.1-5.amzn2.0.2.x86_64 1/9
  Verifying : apr-util-bdb-1.6.1-5.amzn2.0.2.x86_64 2/9
  Verifying : httpd-2.4.43-1.amzn2.x86_64 3/9
  Verifying : mod_http2-1.15.3-2.amzn2.x86_64 4/9
  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-htpd-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

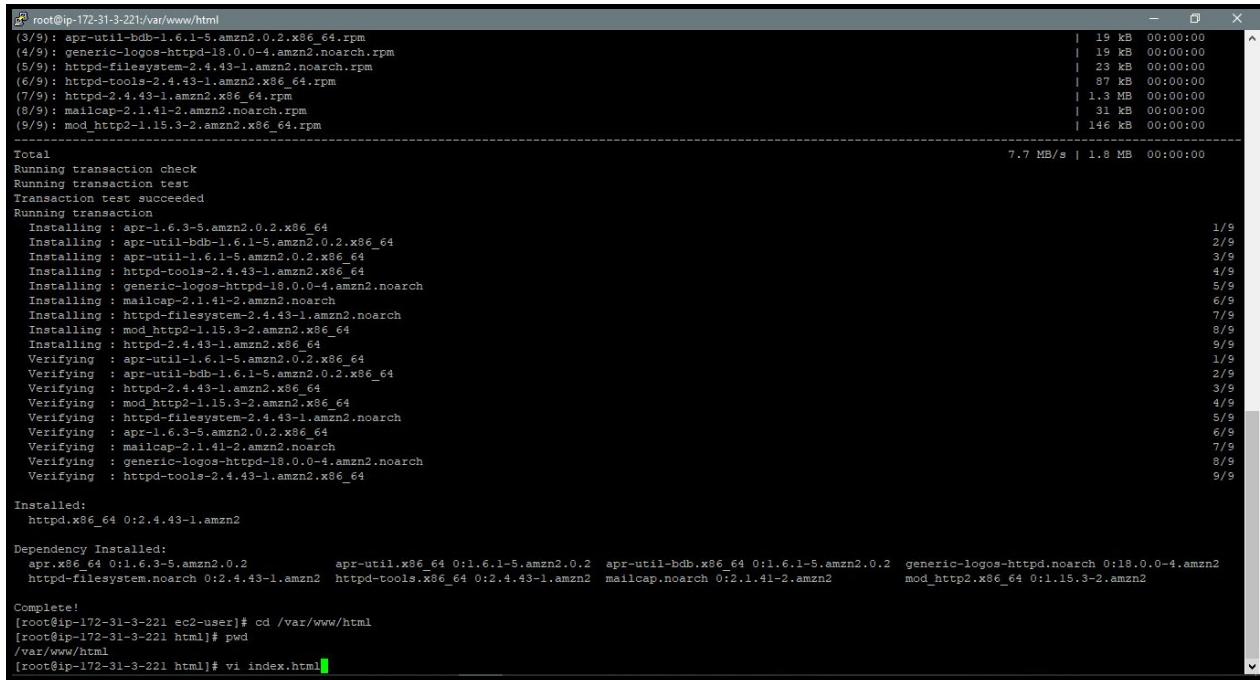
Dependency Installed:
  apr.x86_64 0:1.6.3-5.amzn2.0.2      apr-util.x86_64 0:1.6.1-5.amzn2.0.2      apr-util-bdb.x86_64 0:1.6.1-5.amzn2.0.2      generic-logos-htpd.noarch 0:18.0.0-4.amzn2
  httpd-filesystem.noarch 0:2.4.43-1.amzn2  httpd-tools.x86_64 0:2.4.43-1.amzn2  mailcap.noarch 0:2.1.41-2.amzn2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

```

Step - 38:

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



```
[root@ip-172-31-3-221 ~]# rpm -q --all | grep httpd
(3/9): apr-util-bdb-1.6.1-5.amzn2.0.2.x86_64.rpm
(4/9): generic-logos-httpd-18.0.0-4.amzn2.noarch.rpm
(5/9): httpd-filesystem-2.4.43-1.amzn2.noarch.rpm
(6/9): httpd-tools-2.4.43-1.amzn2.x86_64.rpm
(7/9): httpd-2.4.43-1.amzn2.x86_64.rpm
(8/9): mailcap-2.1.41-2.amzn2.noarch.rpm
(9/9): mod_http2-1.15.3-2.amzn2.x86_64.rpm

Total
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
    Installing : apr-1.6.3-5.amzn2.0.2.x86_64          1/9
    Installing : apr-util-bdb-1.6.1-5.amzn2.0.2.x86_64  2/9
    Installing : apr-util-1.6.1-5.amzn2.0.2.x86_64      3/9
    Installing : httpd-tools-2.4.43-1.amzn2.x86_64      4/9
    Installing : generic-logos-httpd-18.0.0-4.amzn2.noarch 5/9
    Installing : mailcap-2.1.41-2.amzn2.noarch          6/9
    Installing : httpd-filesystem-2.4.43-1.amzn2.noarch 7/9
    Installing : mod_http2-1.15.3-2.amzn2.x86_64        8/9
    Installing : httpd-2.4.43-1.amzn2.x86_64           9/9
Verifying : apr-util-bdb-1.6.1-5.amzn2.0.2.x86_64      1/9
Verifying : apr-util-1.6.1-5.amzn2.0.2.x86_64          2/9
Verifying : httpd-2.4.43-1.amzn2.x86_64              3/9
Verifying : mod_http2-1.15.3-2.amzn2.x86_64           4/9
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
```

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">
        
    </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 ~]# more index.html
<form action="action_page.php" method="post">
    <div class="imgcontainer">
        
    </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>
:wg
```

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 ~]# more index.html
Verifying : httpd-filesystem-2.4.43-1.amzn2.noarch
Verifying : apr-1.6.3-5.amzn2.0.2.x86_64
Verifying : mailcap-2.1.41-2.amzn2.noarch
Verifying : generic-logos-httpd-18.0.0-4.amzn2.noarch
Verifying : httpd-tools-2.4.43-1.amzn2.x86_64

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 ~]# 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">
        
    </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]#
```

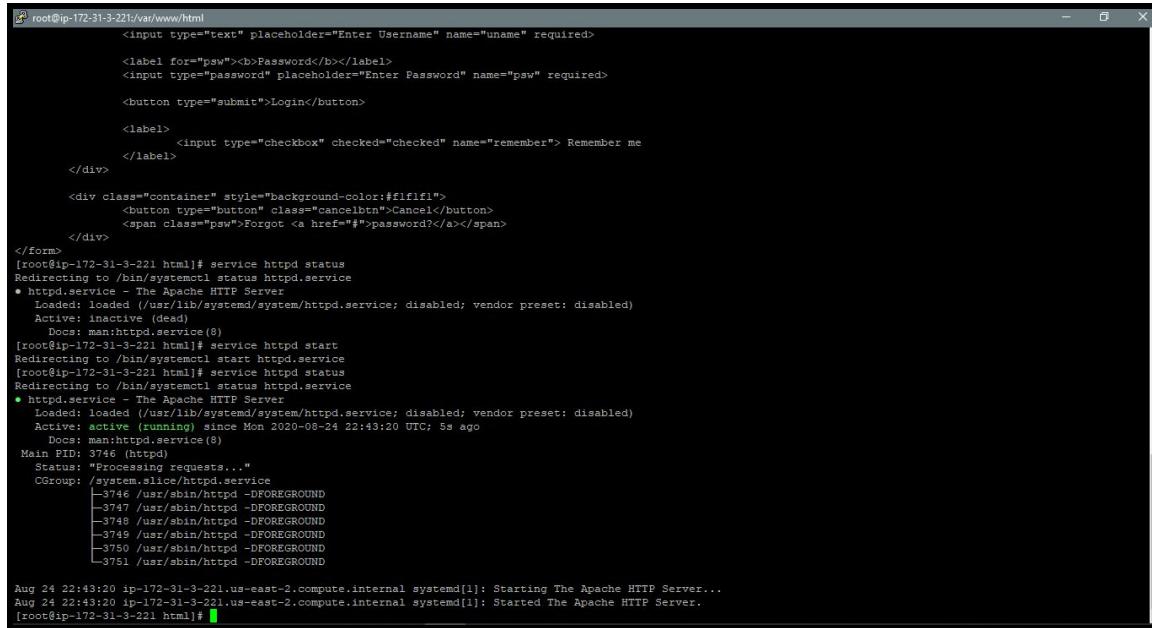
Step - 41:

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



```
<input type="text" placeholder="Enter Username" name="uname" required>
<label form="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]# service httpd status
Redirecting to /bin/systemctl status httpd.service
● httpd.service - The Apache HTTP Server
    Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
      Active: inactive (dead)
        Docs: man:httpd.service(8)
[root@ip-172-31-3-221 html]# service httpd start
Redirecting to /bin/systemctl start httpd.service
[root@ip-172-31-3-221 html]# service httpd status
Redirecting to /bin/systemctl status httpd.service
● httpd.service - The Apache HTTP Server
    Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
      Active: active (running) since Mon 2020-08-24 22:49:20 UTC; 5s ago
        Docs: man:httpd.service(8)
Main PID: 3746 (httpd)
    Status: "Processing requests..."
   CGrou...
```

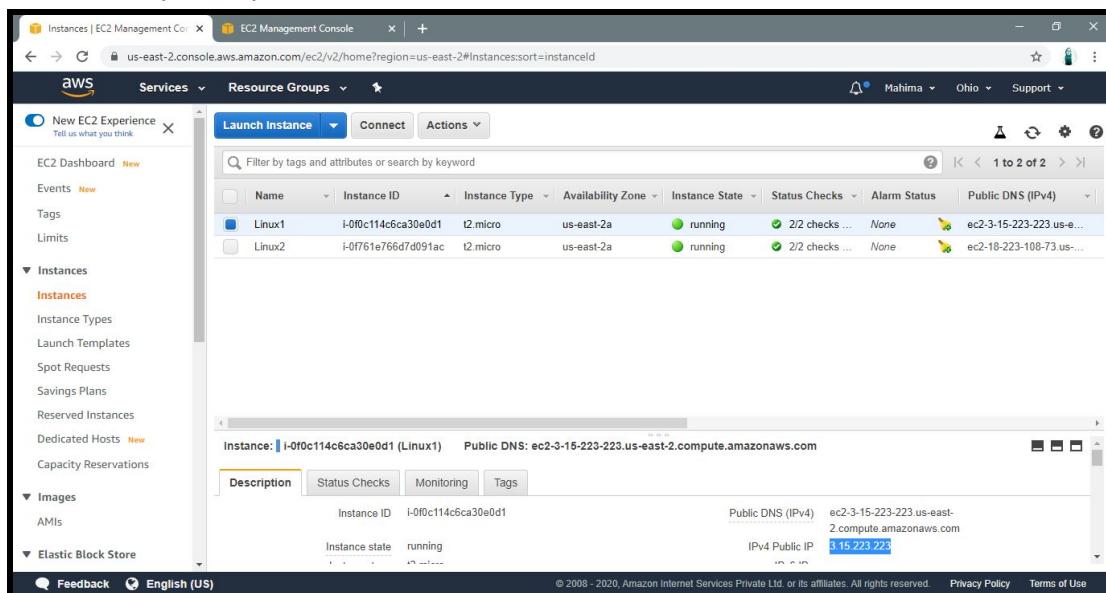
Main PID: 3746 (httpd)
Status: "Processing requests..."
CGroup: /system.slice/httpd.service
└─3746 /usr/sbin/httpd -DFOREGROUND
 ├─3747 /usr/sbin/httpd -DFOREGROUND
 ├─3748 /usr/sbin/httpd -DFOREGROUND
 ├─3749 /usr/sbin/httpd -DFOREGROUND
 ├─3750 /usr/sbin/httpd -DFOREGROUND
 └─3751 /usr/sbin/httpd -DFOREGROUND

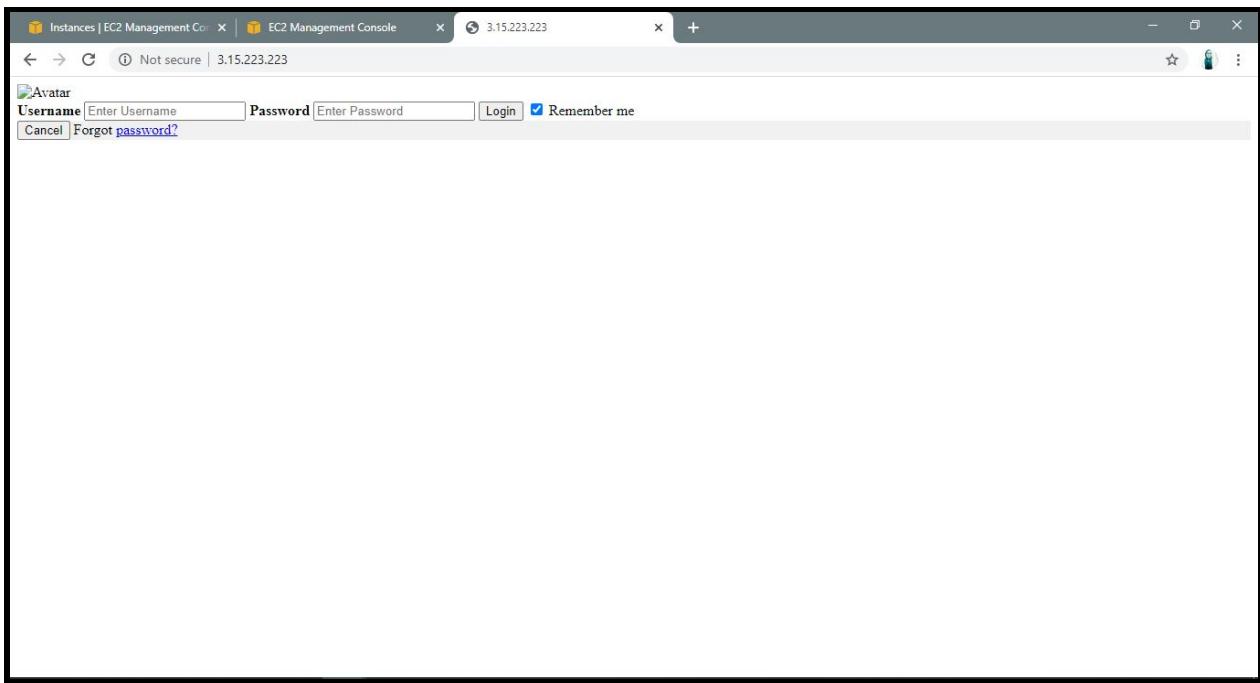
Aug 24 22:49:20 ip-172-31-3-221.us-east-2.compute.internal systemd[1]: Starting The Apache HTTP Server...
Aug 24 22:49:20 ip-172-31-3-221.us-east-2.compute.internal systemd[1]: Started The Apache HTTP Server.
[root@ip-172-31-3-221 html]#

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)





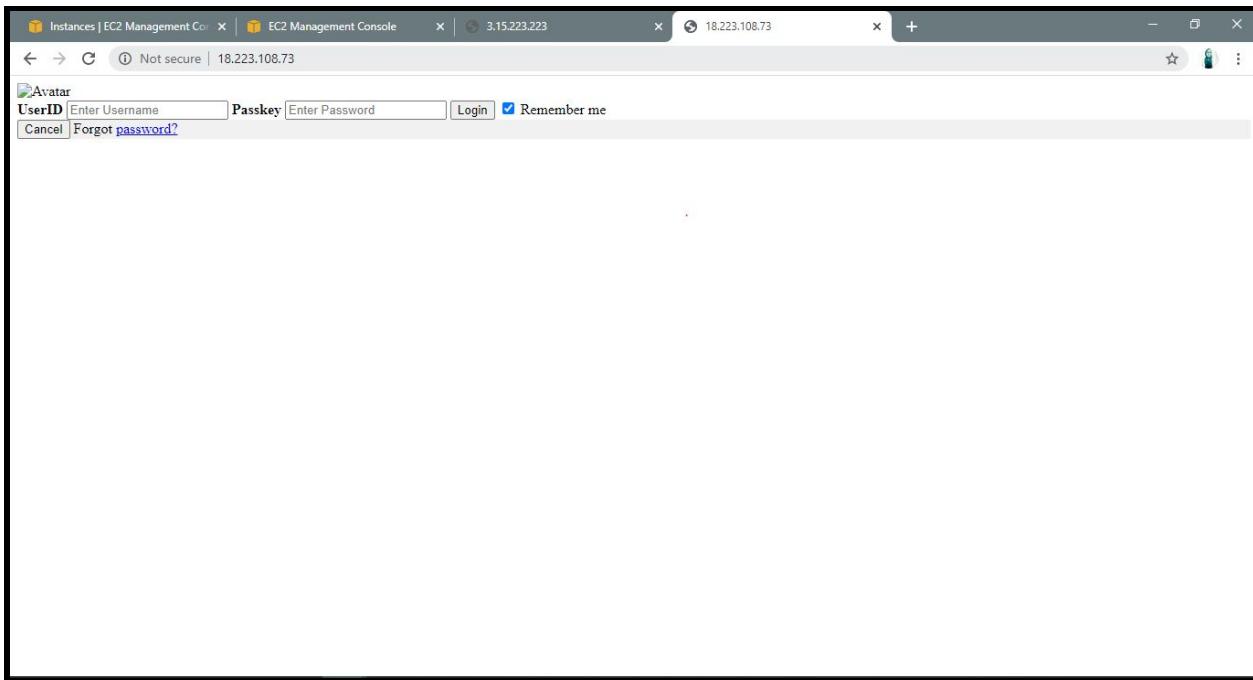
Instance - 2 (Linux2)

The screenshot shows the AWS EC2 Management Console interface. On the left, there's a navigation sidebar with options like New EC2 Experience, EC2 Dashboard, Events, Tags, Limits, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, and Elastic Block Store. The main content area has tabs for Launch Instance, Connect, and Actions. Under Actions, there's a table showing two instances: Linux1 and Linux2. Linux2 is highlighted with a blue selection bar. Below the table, a detailed view for instance i-0f761e766d7d091ac (Linux2) is shown, including its instance ID, public DNS, instance state (running), and public IP address (18.223.108.73).

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
Linux1	i-0f0c114c6ca30e0d1	t2.micro	us-east-2a	running	2/2 checks ...	None	ec2-3-15-223-223.us-e...
Linux2	i-0f761e766d7d091ac	t2.micro	us-east-2a	running	2/2 checks ...	None	ec2-18-223-108-73.us...

Instance: i-0f761e766d7d091ac (Linux2) Public DNS: ec2-18-223-108-73.us-east-2.compute.amazonaws.com

Description	Status Checks	Monitoring	Tags
Instance ID: i-0f761e766d7d091ac	Public DNS (IPv4): ec2-18-223-108-73.us-east-2.compute.amazonaws.com	IPv4 Public IP: 18.223.108.73	
Instance state: running			



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.

Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
ELB-Target-Group	arn:aws:elasticload...	80	HTTP	Instance	LetsUpgradeLB	vpc-ff947...

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.

The screenshot shows the AWS Management Console for the Elastic Load Balancing service. On the left navigation pane, under the 'Load Balancing' section, 'Target Groups' is selected. In the main content area, a target group named 'ELB-Target-Group' is displayed. The 'Targets' tab is selected, showing two registered targets: 'Linux1' and 'Linux2', both of which are marked as 'healthy'. The 'Basic configuration' section at the top shows the target type as 'instance', protocol as 'HTTP : 80', and VPC as 'vpc-f947994'. The ARN of the target group is also visible.

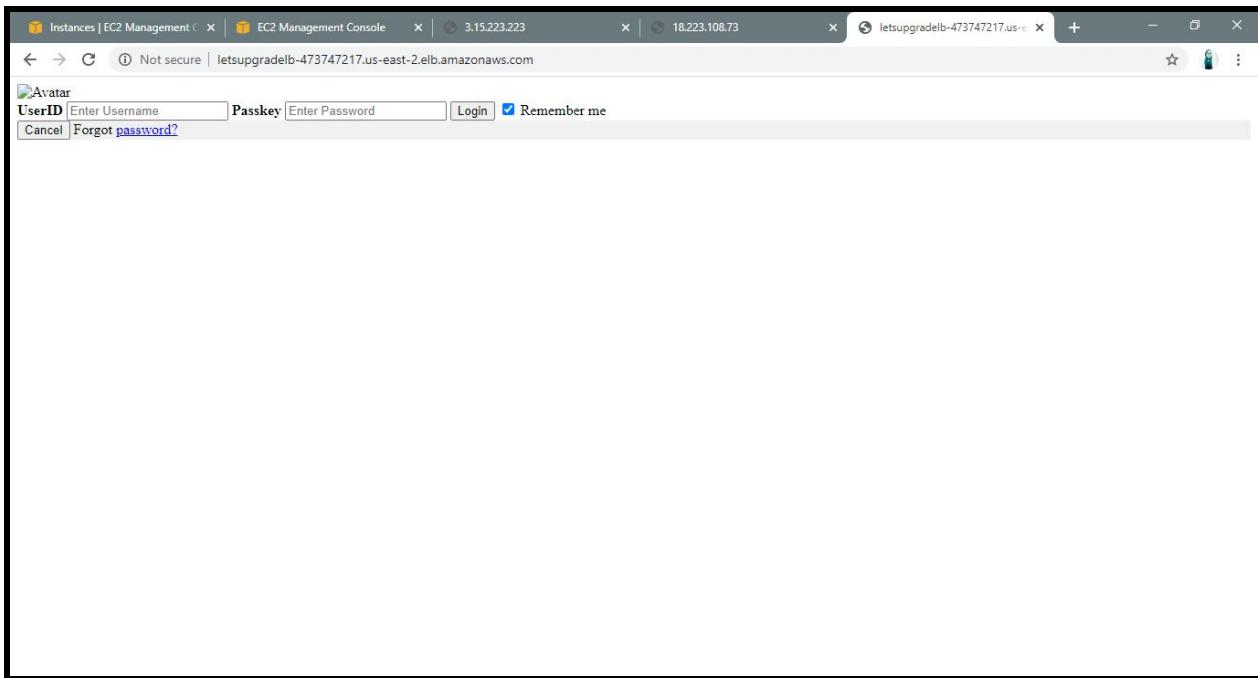
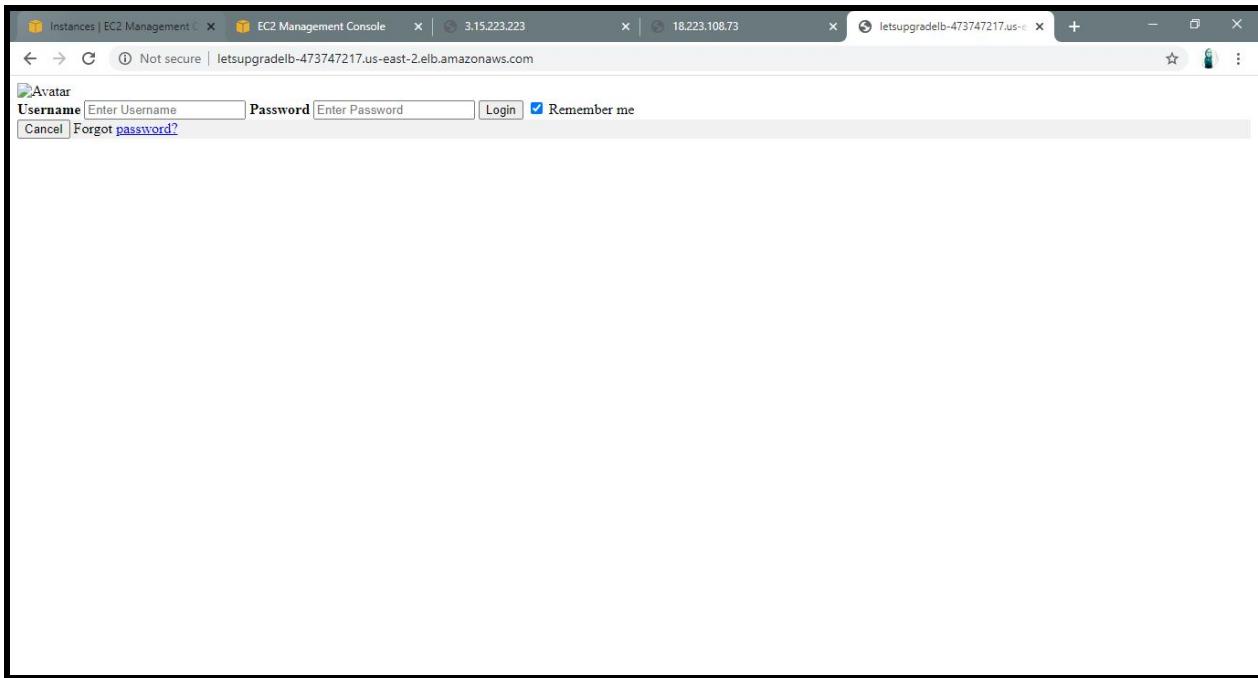
Step - 44:

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

The screenshot shows the AWS Management Console for the Load Balancers service. On the left navigation pane, 'Load Balancers' is selected. In the main content area, a load balancer named 'LetsUpgradeLB' is displayed. The 'Actions' tab is selected, showing a table of load balancers. The 'DNS name' column for 'LetsUpgradeLB' shows the value 'LetsUpgradeLB-473747217.us-east-2.elb.amazonaws.com'. Below the table, the 'Basic Configuration' section provides details about the load balancer, including its name, ARN, and DNS name.

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.