LOAD BALANCER SOLUTION WITH APACHE

Load balancers are a special type of server farm that helps in efficiently distributing incoming network traffic across a group of backend servers. This helps in serving hundred of thousands of servers if not millions of server of concurrent request from users and returns the correct text, image ,video and application data in a very fast and reliable manner.

In this project we are implementing, we are going to configure Apache as the load balancer that act as a single point of access and turns the traffics to 2 webservers.

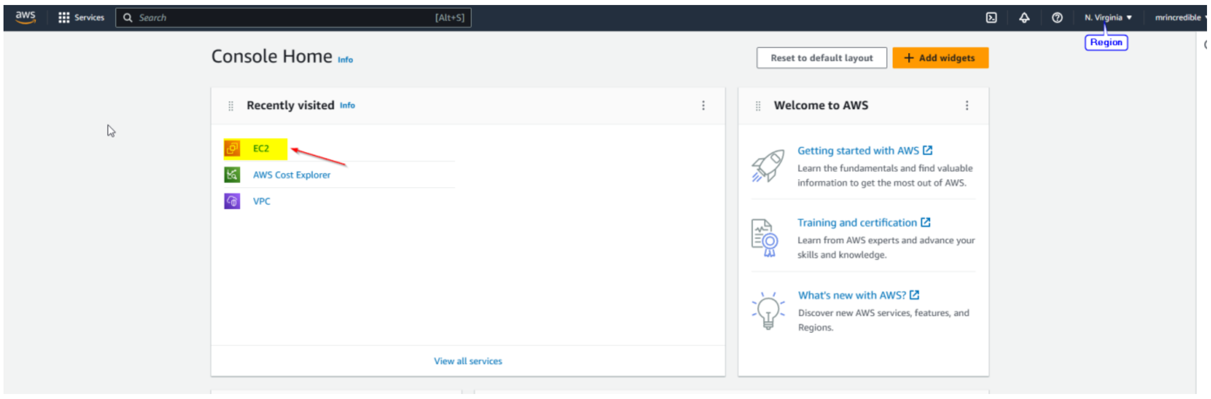
Pre-requisite for the projects is the following.

1) Fundamental Knowledge of Installing and downloading software
2) Basic Understanding of Linux Commands
3) AWS account login with EC2 instances
4) 2 Webserver Linux: Red Hat Enterprise Linux 9
5) Database Server: On Ubuntu 20.04+ MySQL
6) Storage Server: Red Hat Enterprise Linux 9 (NFS Server)
7) Internet connection

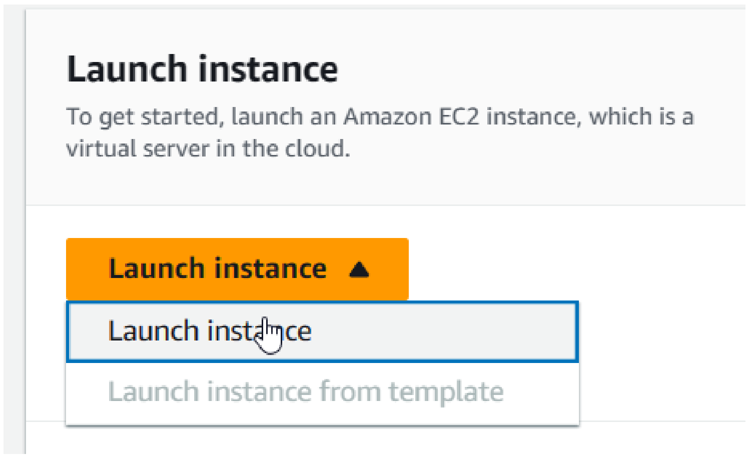
IMPLEMENTATION STEPS: Set up of all EC-2 instances.

i) Ensure you login with your details to your AWS console via the https://aws.amazon.com
ii) Click on the EC2 link and spin up an EC2 instance and make sure they are set up with the operating systems below

Ubuntu Server 20.04 LTS

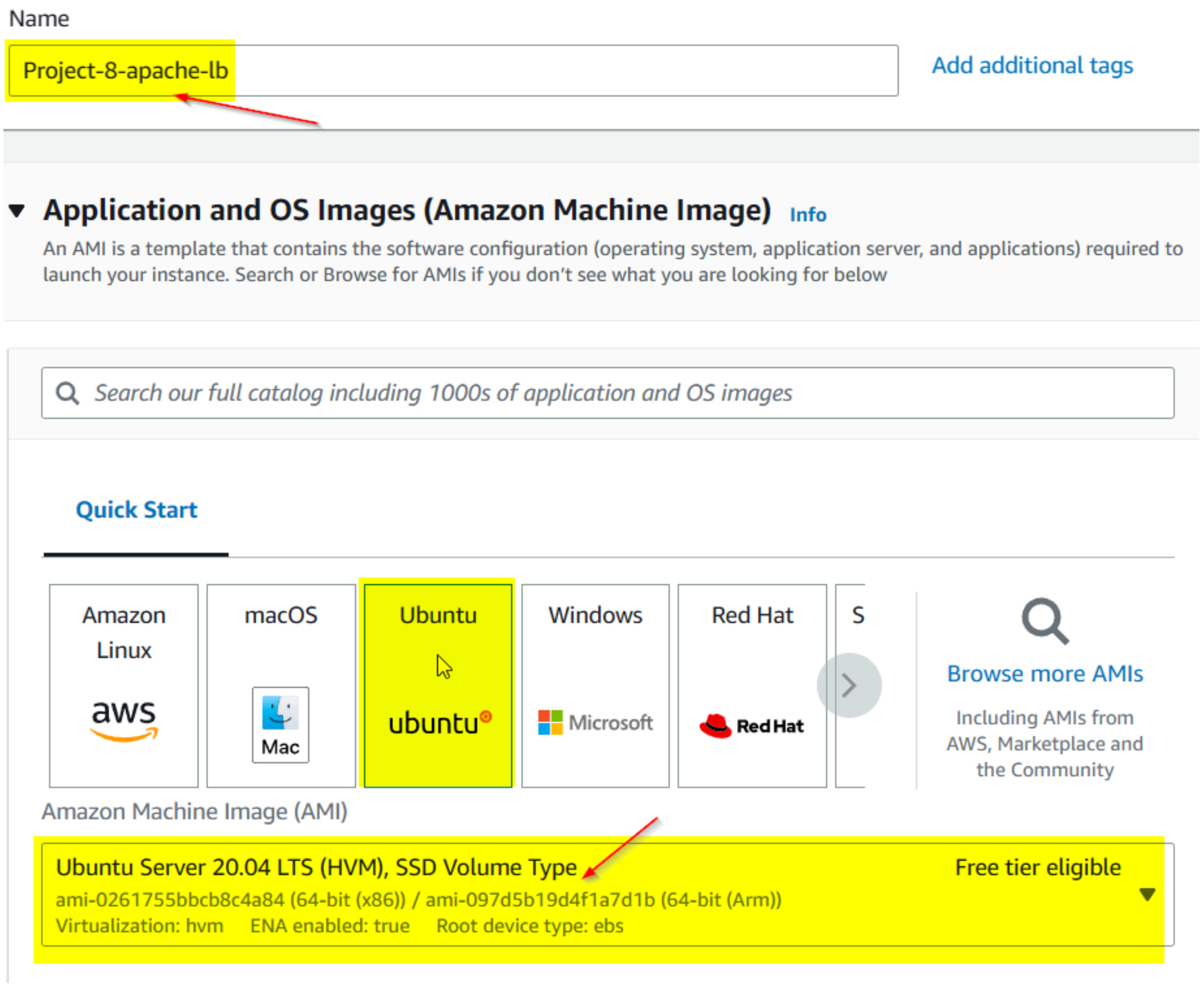


Click on launch instance dropdown button and select launch instance.


Launch instance

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Select Ubuntu from the quick start option and note that amazon machine image selection varies from user to user. Select Ubuntu Server 20.04 LTS (HVM) , SSD Volume type .


Name: Project-8-apache-lb

Application and OS Images (Amazon Machine Image) Info

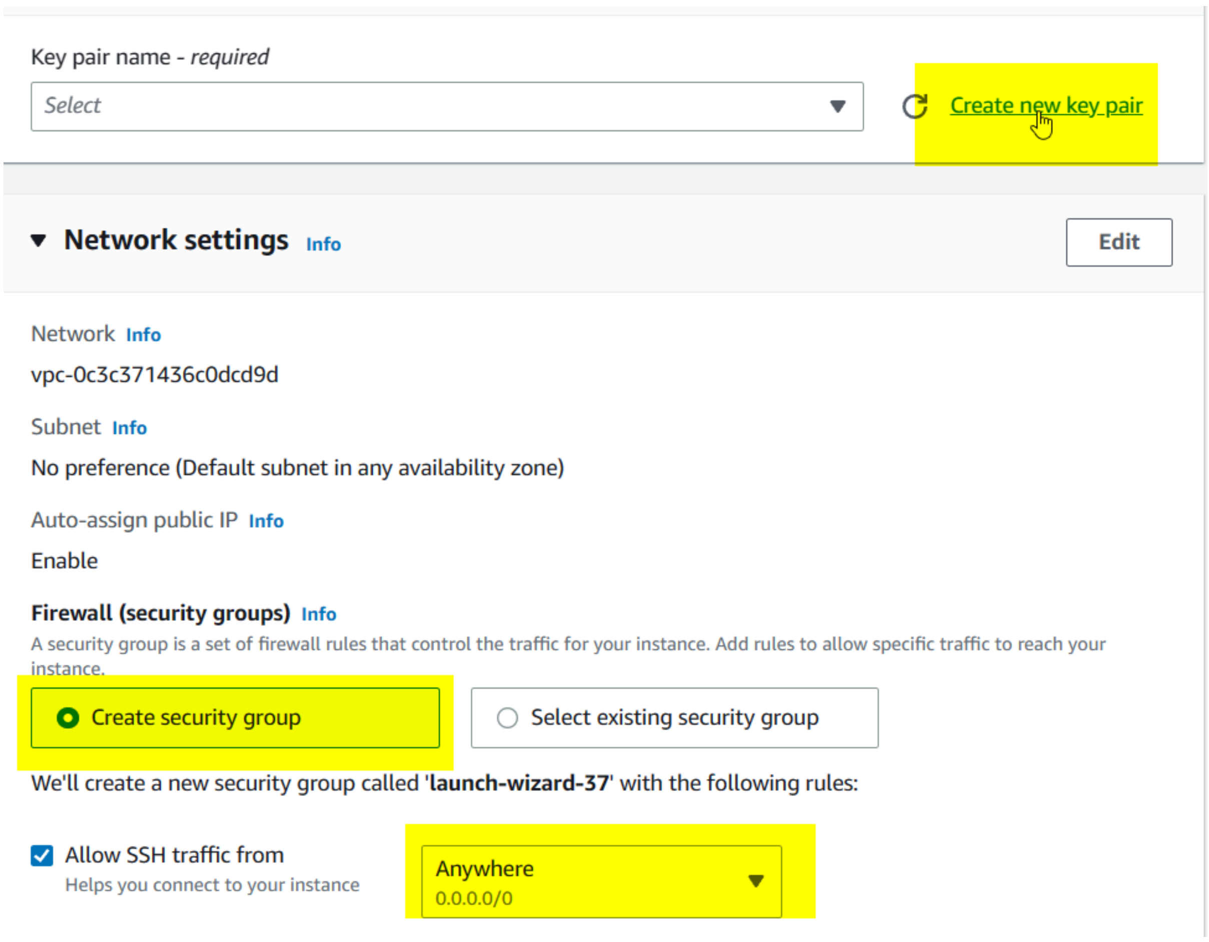
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images
Quick Start
Amazon Linux
macOS
Ubuntu
Windows
Red Hat
S
Browse more AMIs
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 20.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-0261755bbcb8c4a84 (64-bit (x86)) / ami-097d5b19d4f1a7d1b (64-bit (Arm))
Virtualization: hvm   ENA enabled: true   Root device type: ebs

Click on the "Create new key pair" link and ensure the Checkbox remains unchanged on the "Create security group.


Key pair name - *required*

Select

Network settings Info

Network Info

vpc-0c3c371436c0dcd9d

Subnet Info

No preference (Default subnet in any availability zone)

Auto-assign public IP Info

Enable

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

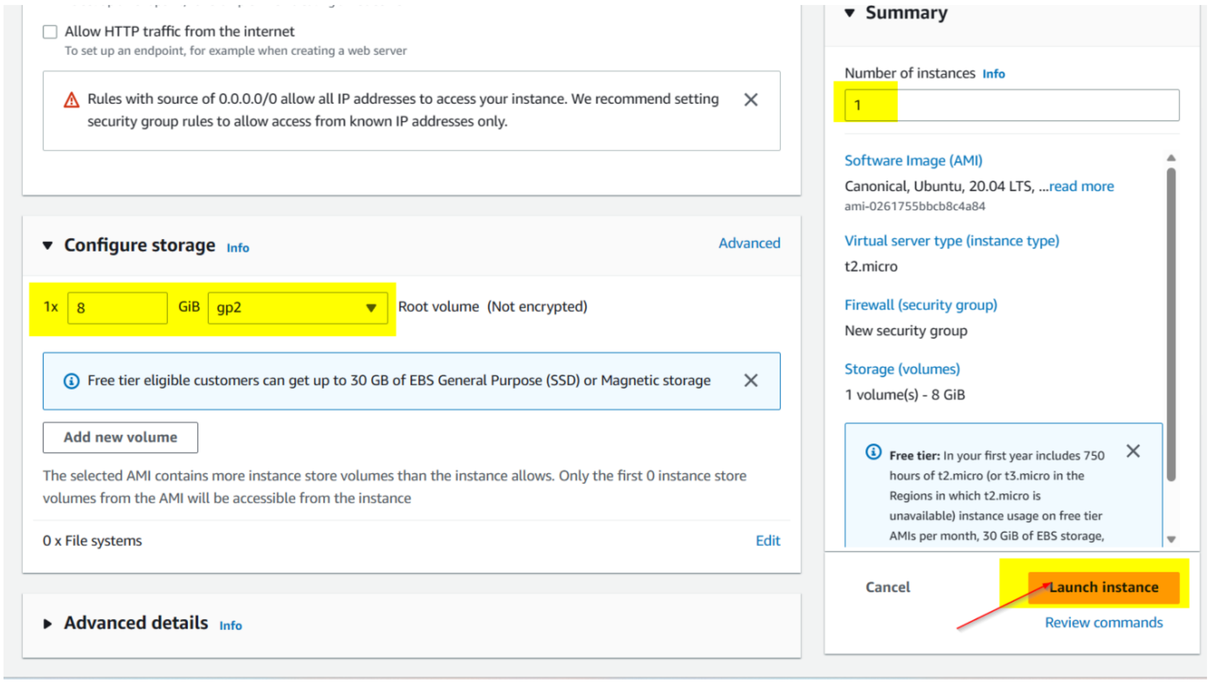
[✓ Create security group] Select existing security group

We'll create a new security group called '**launch-wizard-37**' with the following rules:

✓ Allow SSH traffic from
Helps you connect to your instance

Anywhere
0.0.0.0/0

Select 1 Instance and launch it.


Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

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.

Configure storage Info

1x 8 GiB gp2 Root volume (Not encrypted)

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage

The selected AMI contains more instance store volumes than the instance allows. Only the first 0 instance store volumes from the AMI will be accessible from the instance

0 x File systems

Advanced details Info

Summary

Number of instances Info

1

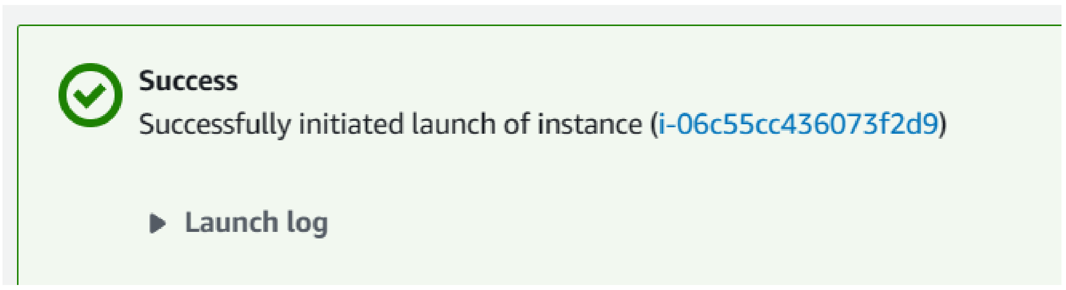
Software Image (AMI)
Canonical, Ubuntu, 20.04 LTS, ...read more
ami-0261755bbcb8c4a84

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

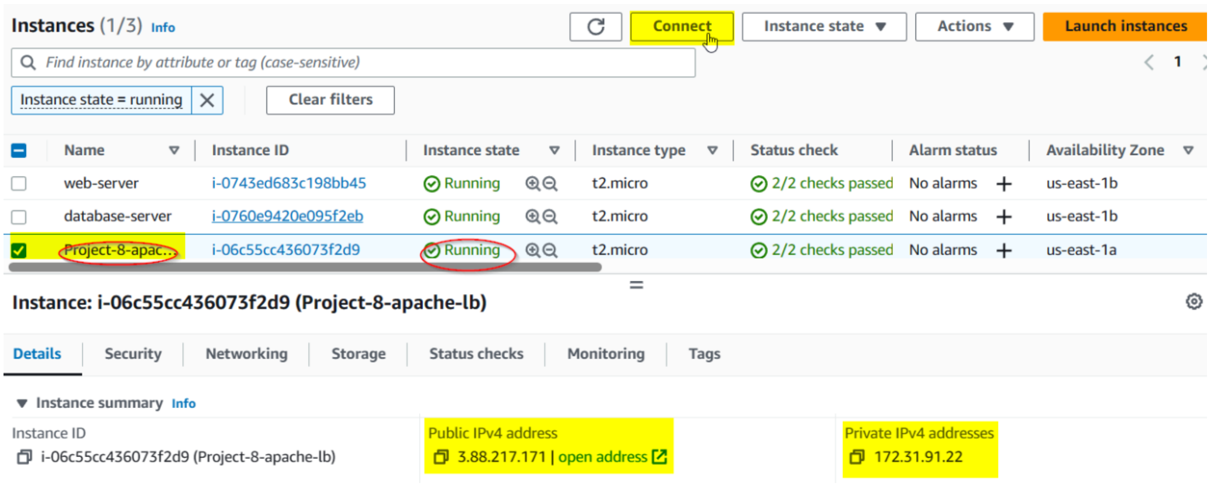
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage,



Success
Successfully initiated launch of instance (i-06c55cc436073f2d9)

▶ Launch log

Click to connect to ssh



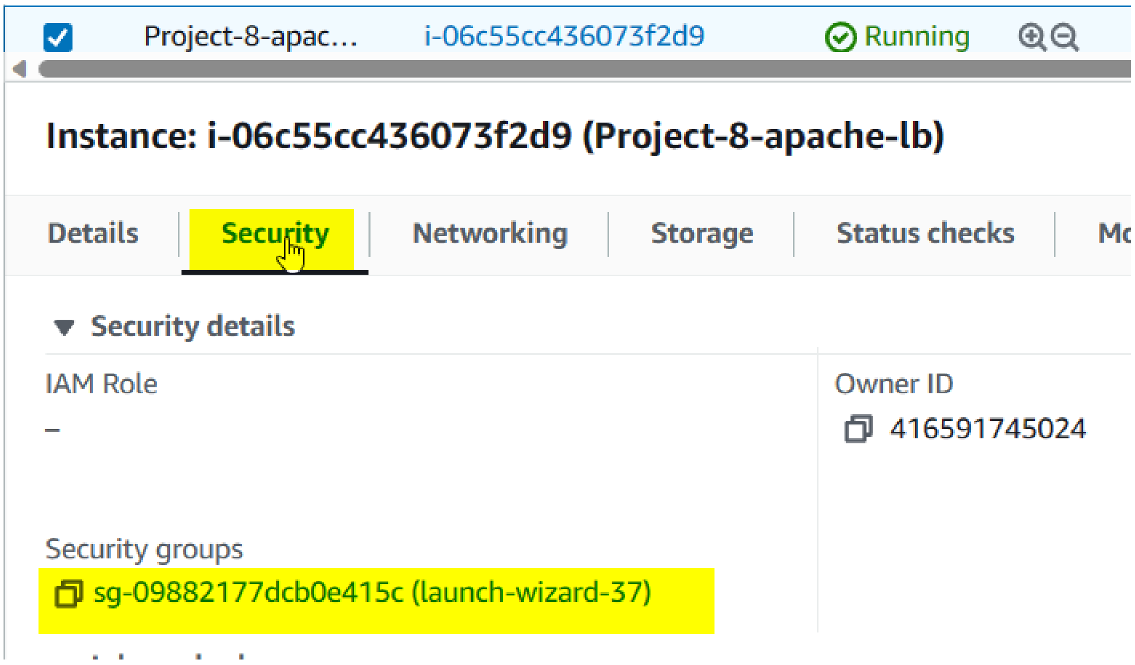
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
☐	web-server	i-0743ed683c198bb45	Running	t2.micro	2/2 checks passed	No alarms +	us-east-1b
☐	database-server	i-0760e9420e095f2eb	Running	t2.micro	2/2 checks passed	No alarms +	us-east-1b
☑	Project-8-apac…	i-06c55cc436073f2d9	Running	t2.micro	2/2 checks passed	No alarms +	us-east-1a

Instance: i-06c55cc436073f2d9 (Project-8-apache-lb)

▼ Instance summary Info

Instance ID	Public IPv4 address	Private IPv4 addresses
i-06c55cc436073f2d9 (Project-8-apache-lb)	3.88.217.171 | open address	172.31.91.22

We need to navigate back to the security group on the platform to add a new rule for TCP port 80 which is the default for web browsers. Click on security button.



Project-8-apac… i-06c55cc436073f2d9 Running

Instance: i-06c55cc436073f2d9 (Project-8-apache-lb)

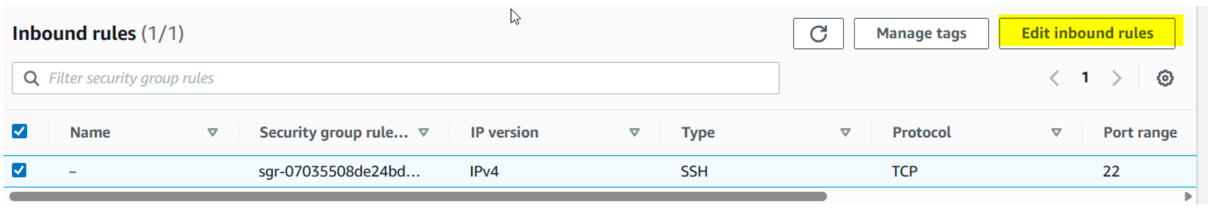
▼ Security details

IAM Role
–

Owner ID
416591745024

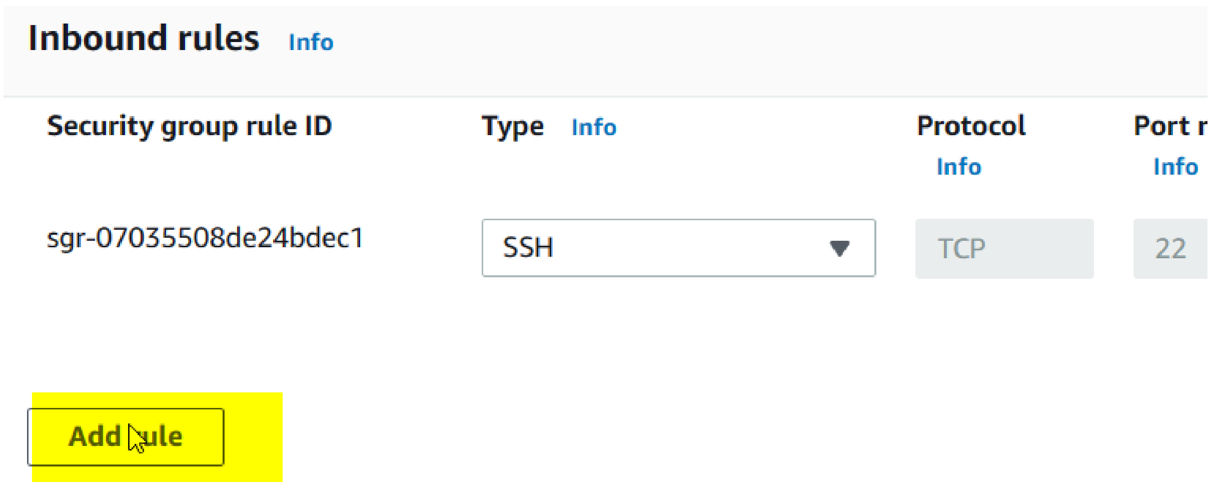
Security groups
sg-09882177dcb0e415c (launch-wizard-37)

Click on “Edit inbound rules “in order to add a new rule for port 80

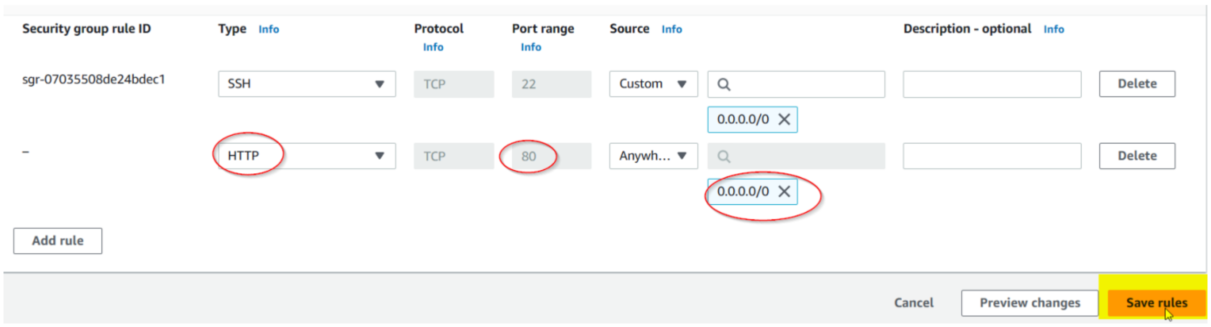


	Name	Security group rule...	IP version	Type	Protocol	Port range
☑	–	sgr-07035508de24bd...	IPv4	SSH	TCP	22

Add rule

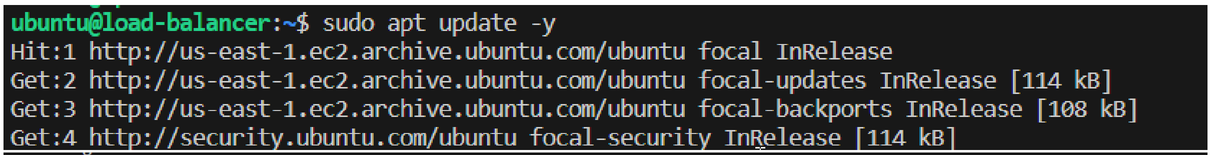


Security group rule ID	Type Info	Protocol Info	Port r Info
sgr-07035508de24bdec1	SSH	TCP	22

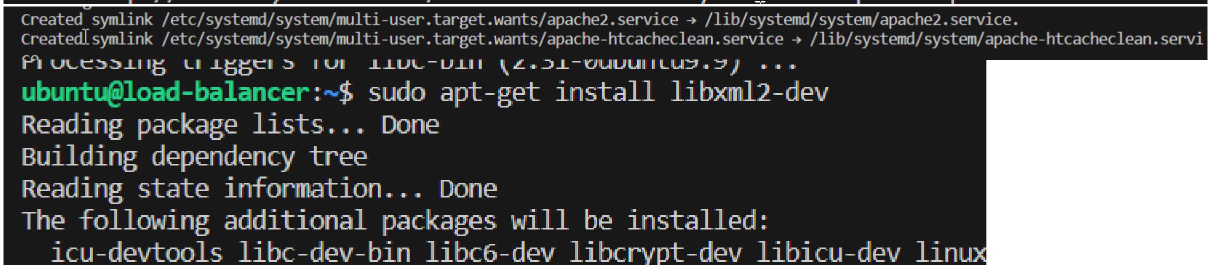


Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-07035508de24bdec1	SSH	TCP	22	Custom	0.0.0.0/0		Delete
–	HTTP	TCP	80	Anywh...	0.0.0.0/0		Delete

We are updating the packages in the package manager and installing Apache2 as seen below.

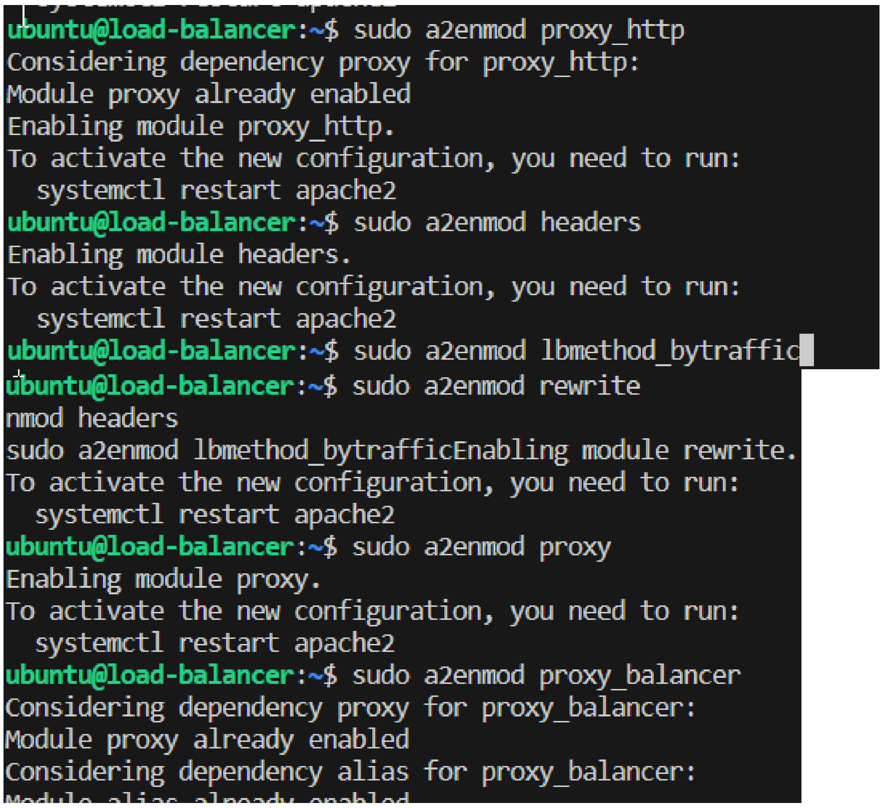


```
ubuntu@load-balancer:~$ sudo apt update -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:4 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
```



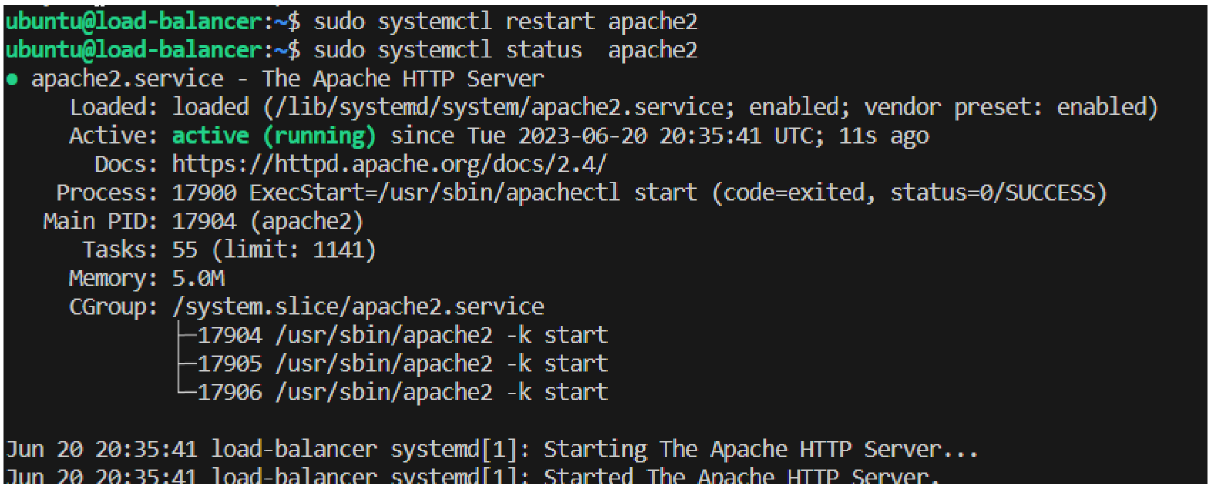
```
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/systemd/system/apache-htcacheclean.servi
Processing triggers for libc-bin (2.31-0ubuntu9.9) ...
ubuntu@load-balancer:~$ sudo apt-get install libxml2-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  icu-devtools libc-dev-bin libc6-dev libcrypt-dev libicu-dev linux
```

We proceed to enable the following module below



```
ubuntu@load-balancer:~$ sudo a2enmod proxy_http
Considering dependency proxy for proxy_http:
Module proxy already enabled
Enabling module proxy_http.
To activate the new configuration, you need to run:
  systemctl restart apache2
ubuntu@load-balancer:~$ sudo a2enmod headers
Enabling module headers.
To activate the new configuration, you need to run:
  systemctl restart apache2
ubuntu@load-balancer:~$ sudo a2enmod lbmethod_bytraffic
ubuntu@load-balancer:~$ sudo a2enmod rewrite
nmod headers
sudo a2enmod lbmethod_bytrafficEnabling module rewrite.
To activate the new configuration, you need to run:
  systemctl restart apache2
ubuntu@load-balancer:~$ sudo a2enmod proxy
Enabling module proxy.
To activate the new configuration, you need to run:
  systemctl restart apache2
ubuntu@load-balancer:~$ sudo a2enmod proxy_balancer
Considering dependency proxy for proxy_balancer:
Module proxy already enabled
Considering dependency alias for proxy_balancer:
Module alias already enabled
```

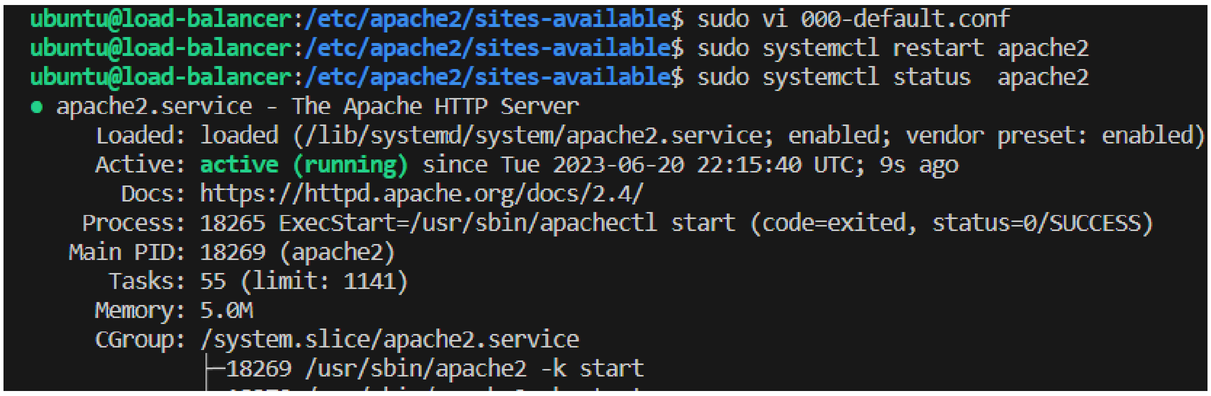
We proceed to restart Apache and check the status to verify it is running and enabled.



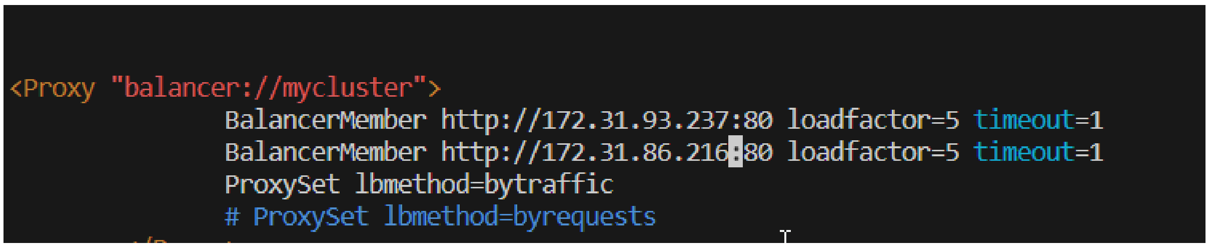
```
ubuntu@load-balancer:~$ sudo systemctl restart apache2
ubuntu@load-balancer:~$ sudo systemctl status  apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2023-06-20 20:35:41 UTC; 11s ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 17900 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 17904 (apache2)
      Tasks: 55 (limit: 1141)
     Memory: 5.0M
     CGroup: /system.slice/apache2.service
             ├─17904 /usr/sbin/apache2 -k start
             ├─17905 /usr/sbin/apache2 -k start
             └─17906 /usr/sbin/apache2 -k start

Jun 20 20:35:41 load-balancer systemd[1]: Starting The Apache HTTP Server...
Jun 20 20:35:41 load-balancer systemd[1]: Started The Apache HTTP Server.
```

We then configure the load balancer by editing its file in such a way that the Apache server maps out the private ip address and add to the file and save the file .



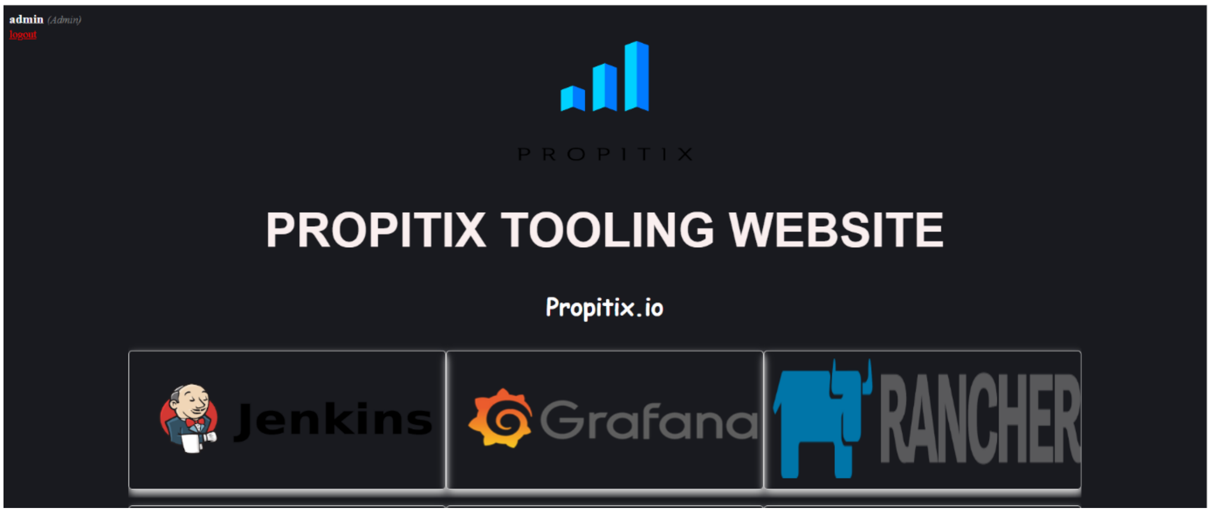
```
ubuntu@load-balancer:/etc/apache2/sites-available$ sudo vi 000-default.conf
ubuntu@load-balancer:/etc/apache2/sites-available$ sudo systemctl restart apache2
ubuntu@load-balancer:/etc/apache2/sites-available$ sudo systemctl status  apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2023-06-20 22:15:40 UTC; 9s ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 18265 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 18269 (apache2)
      Tasks: 55 (limit: 1141)
     Memory: 5.0M
     CGroup: /system.slice/apache2.service
             ├─18269 /usr/sbin/apache2 -k start
```



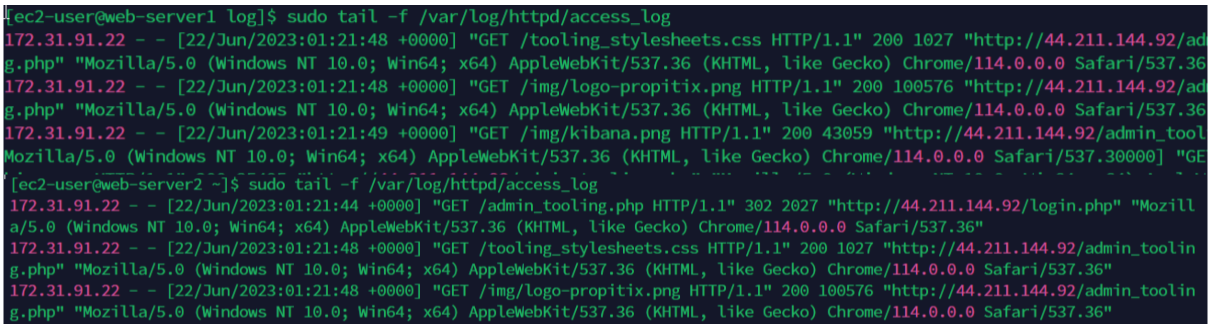
```
<Proxy "balancer://mycluster">
                BalancerMember http://172.31.93.237:80 loadfactor=5 timeout=1
                BalancerMember http://172.31.86.216:80 loadfactor=5 timeout=1
                ProxySet lbmethod=bytraffic
                # ProxySet lbmethod=byrequests
```

As we know there are different types of load balancing, we would be using the by-traffic methods which would distribute incoming between your servers according to the current traffic load .It can be controlled by load factor parameter with the proportion in which the traffic must be distributed.

We are now suppose to launch the website to verify the configuration works as shown below.



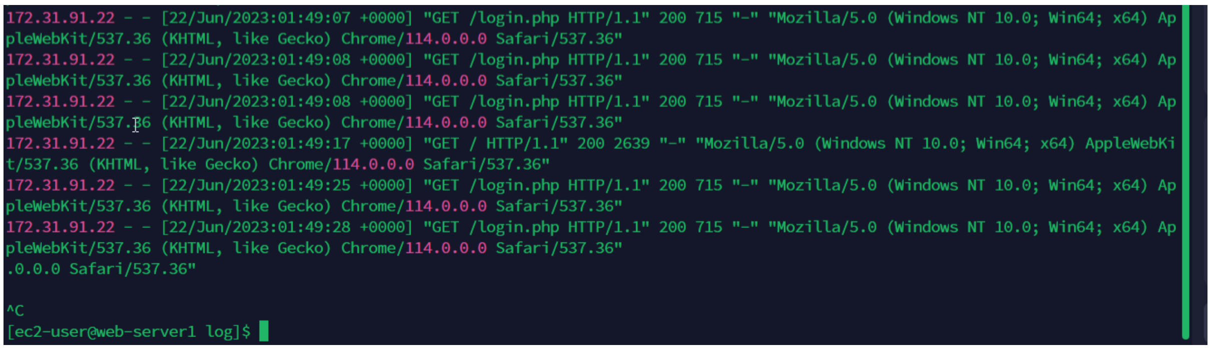
Once this is done we then check both servers to make sure they have their separate log directory and by running the command below we can see the access logs that displays on both webserver terminals.



```
[ec2-user@web-server1 log]$ sudo tail -f /var/log/httpd/access_log
172.31.91.22 - - [22/Jun/2023:01:21:48 +0000] "GET /tooling_stylesheets.css HTTP/1.1" 200 1027 "http://44.211.144.92/ad
g.php" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36
172.31.91.22 - - [22/Jun/2023:01:21:48 +0000] "GET /img/logo-propitix.png HTTP/1.1" 200 100576 "http://44.211.144.92/ad
g.php" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36
172.31.91.22 - - [22/Jun/2023:01:21:49 +0000] "GET /img/kibana.png HTTP/1.1" 200 43059 "http://44.211.144.92/admin_tool
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.30000] "GE
```

```
[ec2-user@web-server2 ~]$ sudo tail -f /var/log/httpd/access_log
172.31.91.22 - - [22/Jun/2023:01:21:44 +0000] "GET /admin_tooling.php HTTP/1.1" 302 2027 "http://44.211.144.92/login.php" "Mozill
a/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36"
172.31.91.22 - - [22/Jun/2023:01:21:48 +0000] "GET /tooling_stylesheets.css HTTP/1.1" 200 1027 "http://44.211.144.92/admin_toolin
g.php" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36"
172.31.91.22 - - [22/Jun/2023:01:21:48 +0000] "GET /img/logo-propitix.png HTTP/1.1" 200 100576 "http://44.211.144.92/admin_toolin
g.php" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36"
```

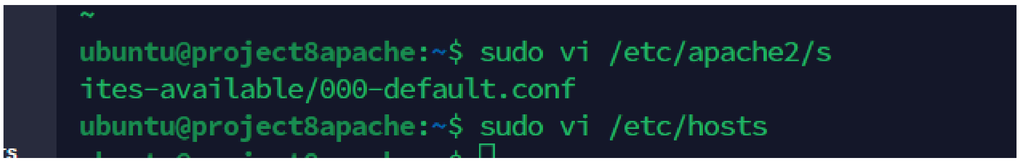
After several refreshing of the browsers it can be noticed that both servers receives HTTP GET requests and the traffic is distributed evenly because of the load factor we inputted in the 000-default.conf file .



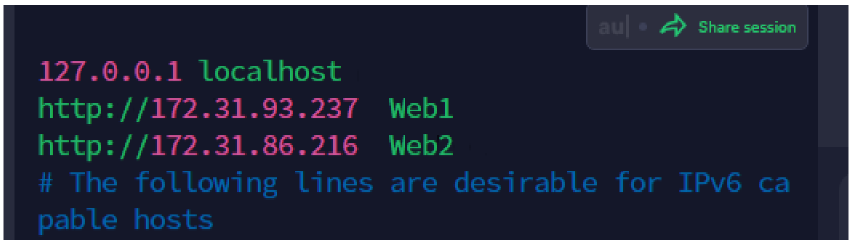
```
172.31.91.22 - - [22/Jun/2023:01:49:07 +0000] "GET /login.php HTTP/1.1" 200 715 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) Ap
pleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36"
172.31.91.22 - - [22/Jun/2023:01:49:08 +0000] "GET /login.php HTTP/1.1" 200 715 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) Ap
pleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36"
172.31.91.22 - - [22/Jun/2023:01:49:08 +0000] "GET /login.php HTTP/1.1" 200 715 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) Ap
pleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36"
172.31.91.22 - - [22/Jun/2023:01:49:17 +0000] "GET / HTTP/1.1" 200 2639 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKi
t/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36"
172.31.91.22 - - [22/Jun/2023:01:49:25 +0000] "GET /login.php HTTP/1.1" 200 715 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) Ap
pleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36"
172.31.91.22 - - [22/Jun/2023:01:49:28 +0000] "GET /login.php HTTP/1.1" 200 715 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) Ap
pleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36"
.0.0.0 Safari/537.36"

^C
[ec2-user@web-server1 log]$
```

Once all this is done then we can decide to configure our local DNS name resolution by creating a file on the load balance server and tag the webservers private ip address by an arbitrary name .In this case we named it Web1 and Web2 as shown below

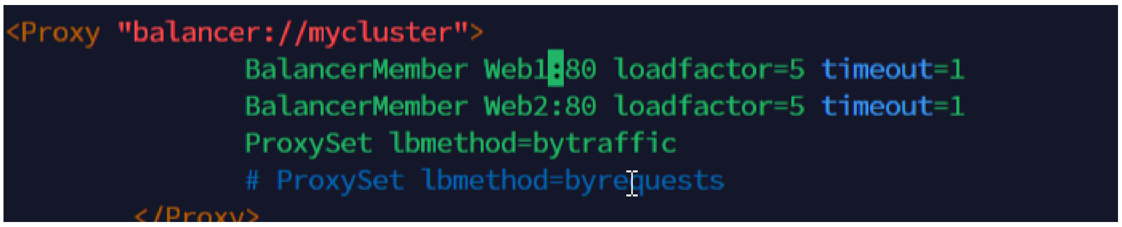


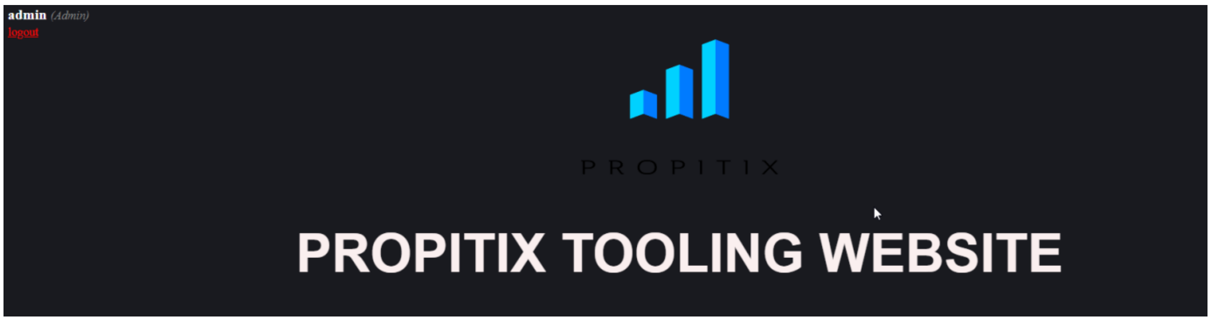
```
~
ubuntu@project8apache:~$ sudo vi /etc/apache2/s
ites-available/000-default.conf
ubuntu@project8apache:~$ sudo vi /etc/hosts
```



```
127.0.0.1 localhost
http://172.31.93.237  Web1
http://172.31.86.216  Web2
# The following lines are desirable for IPv6 ca
pable hosts
```

And then proceed to change it in the Load Balancer config file and once this is done we can curl our webservers from the Load Balancer server locally.


```
<Proxy "balancer://mycluster">
               BalancerMember Web1:80 loadfactor=5 timeout=1
               BalancerMember Web2:80 loadfactor=5 timeout=1
               ProxySet lbmethod=bytraffic
               # ProxySet lbmethod=byrequests
        </Proxy>
```



We have just implemented a Load Balancing Web solution.