CS 524 Intro to Cloud computing

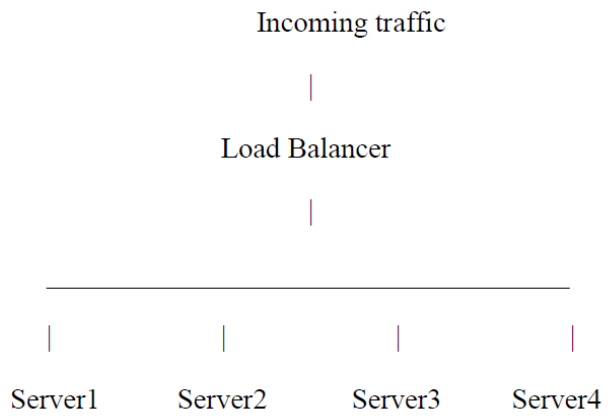
Dhruvit Patel (CWID: 10404032)

Lab Assignment 2

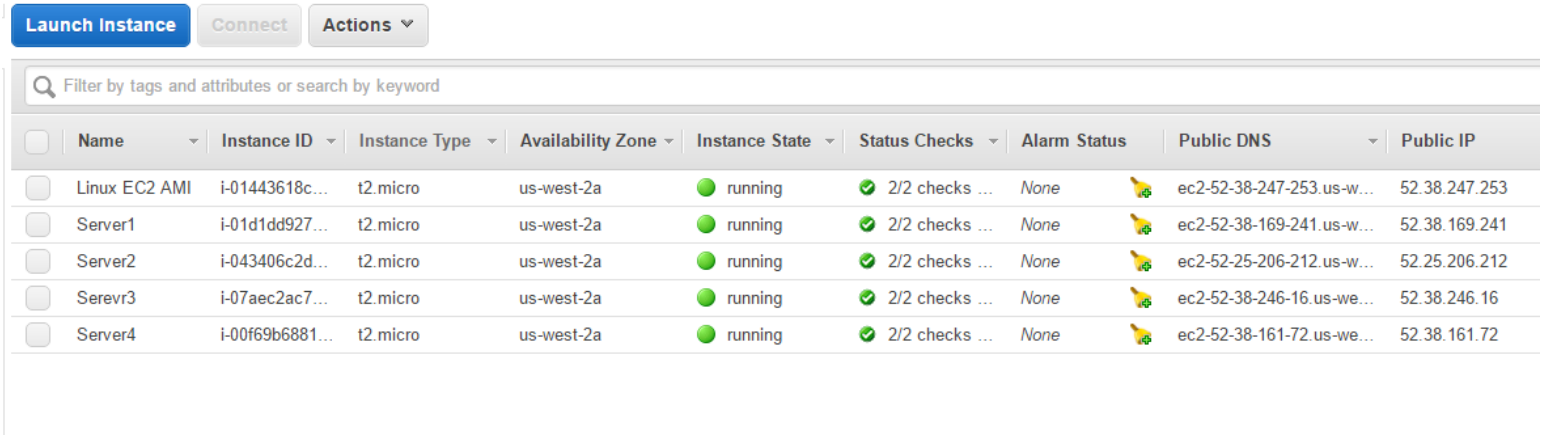
I followed the step-by-step procedures to accomplish this assignment.

Step 1: Create the Amazon EC2 instances

You need to create *five* EC2 instances of the same time you have created in the previous homework. One of these instances will act as a load balancer; the other four will act as web servers named **Server1, Server2, Server3, Server4**, as demonstrated below:

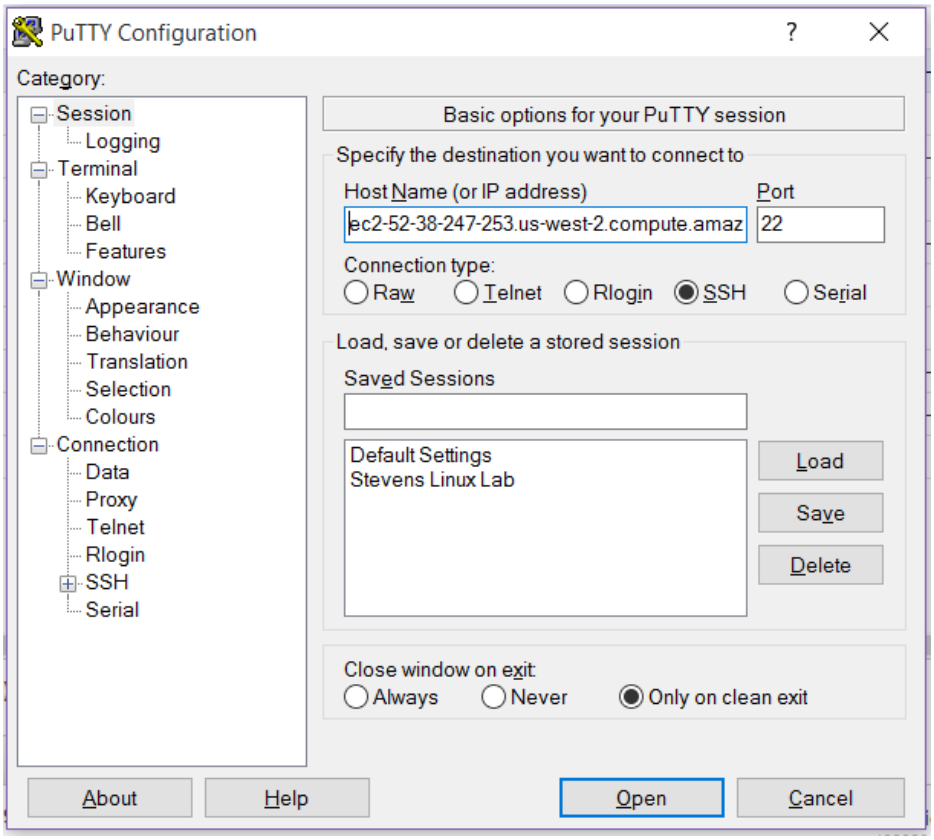


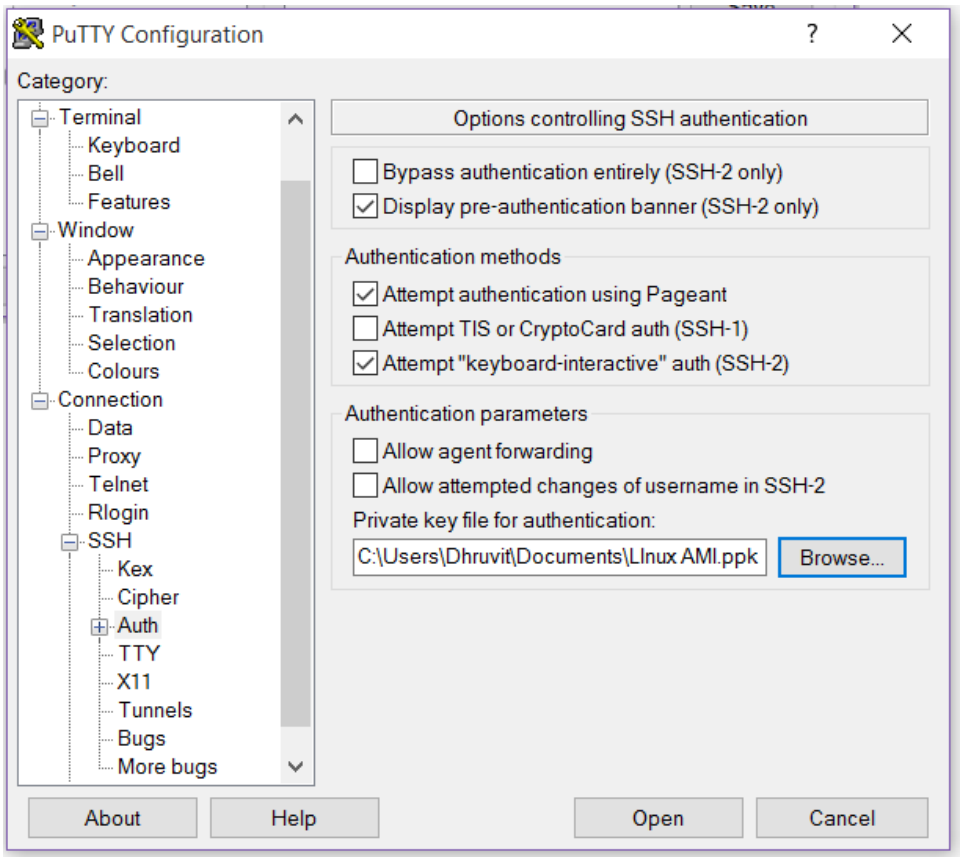
First, I have created five instances and one of them act as load balancer (Linux EC2 AMI instance).



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS	Public IP
Linux EC2 AMI	i-01443618c...	t2.micro	us-west-2a	running	2/2 checks ...	None	ec2-52-38-247-253.us-w...	52.38.247.253
Server1	i-01d1dd927...	t2.micro	us-west-2a	running	2/2 checks ...	None	ec2-52-38-169-241.us-w...	52.38.169.241
Server2	i-043406c2d...	t2.micro	us-west-2a	running	2/2 checks ...	None	ec2-52-25-206-212.us-w...	52.25.206.212
Serevr3	i-07aec2ac7...	t2.micro	us-west-2a	running	2/2 checks ...	None	ec2-52-38-246-16.us-we...	52.38.246.16
Server4	i-00f69b6881...	t2.micro	us-west-2a	running	2/2 checks ...	None	ec2-52-38-161-72.us-we...	52.38.161.72

I changed security group for my all instance and add rules for HTTP port 80.
I used PuTTY to connect all created EC2 instances.


PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Serial
Basic options for your PuTTY session
Specify the destination you want to connect to
Host Name (or IP address)
Port
ec2-52-38-247-253.us-west-2.compute.amaz
22
Connection type:
Raw
Telnet
Rlogin
SSH
Serial
Load, save or delete a stored session
Saved Sessions
Default Settings
Stevens Linux Lab
Load
Save
Delete
Close window on exit:
Always
Never
Only on clean exit
About
Help
Open
Cancel


PuTTY Configuration
Category:
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Kex
Cipher
Auth
TTY
X11
Tunnels
Bugs
More bugs
Options controlling SSH authentication
Bypass authentication entirely (SSH-2 only)
Display pre-authentication banner (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:
C:\Users\Dhruvit\Documents\LInux AMI.ppk
Browse...
About
Help
Open
Cancel

ec2-52-38-247-253.us-west-2.compute.amazonaws.com - PuTTY

```
login as: ec2-user
```

ec2-user@ip-172-31-35-243:~

```
login as: ec2-user
Authenticating with public key "imported-openssh-key"
Last login: Tue Apr 12 16:43:46 2016 from 155.246.210.100

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2016.03-release-notes/
4 package(s) needed for security, out of 6 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-35-243 ~]$
```

Step 2: Install *Nginx* on each instance

After launching the instances, use *yum* (the Amazon Linux native software manager) to install *Nginx* on every instance and start the *Nginx* service.

To verify the *Nginx* is working, visit the hosted default webpage through instance's public DNS from an internet browser. You will see the Welcome message (if *Nginx* http server works properly).

First of all, I updated “yum” installer, which used to install any pkg in Linux machine.

```
ec2-user@ip-172-31-40-35:~
login as: ec2-user
Authenticating with public key "imported-openssh-key"

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2016.03-release-notes/
4 package(s) needed for security, out of 6 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-40-35 ~]$ sudo yum update
Loaded plugins: priorities, update-motd, upgrade-helper
amzn-main/latest                                          | 2.1 kB     00:00
amzn-updates/latest                                       | 2.3 kB     00:00
Resolving Dependencies
--> Running transaction check
---> Package libssh2.x86_64 0:1.4.2-1.10.amzn1 will be updated
---> Package libssh2.x86_64 0:1.4.2-2.13.amzn1 will be an update
---> Package nano.x86_64 0:2.3.1-10.16.amzn1 will be updated
---> Package nano.x86_64 0:2.5.3-1.19.amzn1 will be an update
---> Package openssh.x86_64 0:6.6.1p1-23.60.amzn1 will be updated
---> Package openssh.x86_64 0:6.6.1p1-25.61.amzn1 will be an update
---> Package openssh-clients.x86_64 0:6.6.1p1-23.60.amzn1 will be updated
---> Package openssh-clients.x86_64 0:6.6.1p1-25.61.amzn1 will be an update
---> Package openssh-server.x86_64 0:6.6.1p1-23.60.amzn1 will be updated
---> Package openssh-server.x86_64 0:6.6.1p1-25.61.amzn1 will be an update
---> Package sysctl-defaults.noarch 0:1.0-1.0.amzn1 will be updated
---> Package sysctl-defaults.noarch 0:1.0-1.1.amzn1 will be an update
--> Finished Dependency Resolution
```

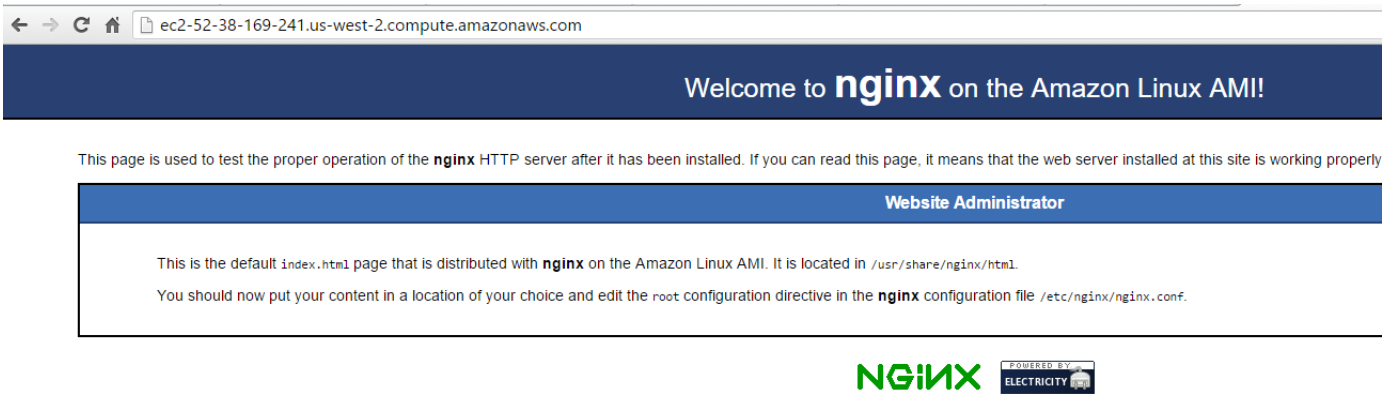
I installed nginx with the use of “yum” Linux installer.

```
ec2-user@ip-172-31-40-35:~
[ec2-user@ip-172-31-40-35 ~]$ sudo yum install nginx
Loaded plugins: priorities, update-motd, upgrade-helper
Resolving Dependencies
--> Running transaction check
---> Package nginx.x86_64 1:1.8.1-1.26.amzn1 will be installed
--> Processing Dependency: gd for package: 1:nginx-1.8.1-1.26.amzn1.x86_64
--> Processing Dependency: GeoIP for package: 1:nginx-1.8.1-1.26.amzn1.x86_64
--> Processing Dependency: libprofiler.so.0()(64bit) for package: 1:nginx-1.8.1-1.26.amzn1.x86_64
--> Processing Dependency: libGeoIP.so.1()(64bit) for package: 1:nginx-1.8.1-1.26.amzn1.x86_64
--> Processing Dependency: libgd.so.2()(64bit) for package: 1:nginx-1.8.1-1.26.amzn1.x86_64
--> Running transaction check
---> Package GeoIP.x86_64 0:1.4.8-1.5.amzn1 will be installed
---> Package gd.x86_64 0:2.0.35-11.10.amzn1 will be installed
--> Processing Dependency: libXpm.so.4()(64bit) for package: gd-2.0.35-11.10.amzn1.x86_64
---> Package gperftools-libs.x86_64 0:2.0-11.5.amzn1 will be installed
--> Processing Dependency: libunwind.so.8()(64bit) for package: gperftools-libs-2.0-11.5.amzn1.x86_6
4
--> Running transaction check
---> Package libXpm.x86_64 0:3.5.10-2.9.amzn1 will be installed
---> Package libunwind.x86_64 0:1.1-10.8.amzn1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================================
 Package                  Arch            Version                      Repository          Size
================================================================================================
Installing:
 nginx                    x86_64          1:1.8.1-1.26.amzn1           amzn-main           557 k
```

After then, I start and reload nginx server to serve the functionality provided by it.

```
ec2-user@ip-172-31-42-39:~
Transaction test succeeded
Running transaction
  Installing : libXpm-3.5.10-2.9.amzn1.x86_64                          1/6
  Installing : gd-2.0.35-11.10.amzn1.x86_64                            2/6
  Installing : libunwind-1.1-10.8.amzn1.x86_64                         3/6
  Installing : gperftools-libs-2.0-11.5.amzn1.x86_64                   4/6
  Installing : GeoIP-1.4.8-1.5.amzn1.x86_64                            5/6
  Installing : 1:nginx-1.8.1-1.26.amzn1.x86_64                         6/6
  Verifying  : gperftools-libs-2.0-11.5.amzn1.x86_64                   1/6
  Verifying  : GeoIP-1.4.8-1.5.amzn1.x86_64                            2/6
  Verifying  : 1:nginx-1.8.1-1.26.amzn1.x86_64                         3/6
  Verifying  : libunwind-1.1-10.8.amzn1.x86_64                         4/6
  Verifying  : gd-2.0.35-11.10.amzn1.x86_64                            5/6
  Verifying  : libXpm-3.5.10-2.9.amzn1.x86_64                          6/6

Installed:
  nginx.x86_64 1:1.8.1-1.26.amzn1

Dependency Installed:
  GeoIP.x86_64 0:1.4.8-1.5.amzn1            gd.x86_64 0:2.0.35-11.10.amzn1
  gperftools-libs.x86_64 0:2.0-11.5.amzn1   libXpm.x86_64 0:3.5.10-2.9.amzn1
  libunwind.x86_64 0:1.1-10.8.amzn1

Complete!
[ec2-user@ip-172-31-42-39 ~]$ sudo service nginx start
Starting nginx:                                            [  OK  ]
[ec2-user@ip-172-31-42-39 ~]$ sudo service nginx reload
Reloading nginx:                                           [  OK  ]
[ec2-user@ip-172-31-42-39 ~]$
```

This is by default index.html webpage provided by nginx to verify, it is working or not.



Now, we are going to modify this index.html webpage to depict our custom information.
Visit directory, who contained this index.html by “cd /usr/share/nginx/html”.

```
ec2-user@ip-172-31-42-39:/usr/share/nginx/html
  Installing : gd-2.0.35-11.10.amzn1.x86_64                                2/6
  Installing : libunwind-1.1-10.8.amzn1.x86_64                             3/6
  Installing : gperftools-libs-2.0-11.5.amzn1.x86_64                       4/6
  Installing : GeoIP-1.4.8-1.5.amzn1.x86_64                                5/6
  Installing : 1:nginx-1.8.1-1.26.amzn1.x86_64                             6/6
  Verifying  : gperftools-libs-2.0-11.5.amzn1.x86_64                       1/6
  Verifying  : GeoIP-1.4.8-1.5.amzn1.x86_64                                2/6
  Verifying  : 1:nginx-1.8.1-1.26.amzn1.x86_64                             3/6
  Verifying  : libunwind-1.1-10.8.amzn1.x86_64                             4/6
  Verifying  : gd-2.0.35-11.10.amzn1.x86_64                                5/6
  Verifying  : libXpm-3.5.10-2.9.amzn1.x86_64                              6/6

Installed:
  nginx.x86_64 1:1.8.1-1.26.amzn1

Dependency Installed:
  GeoIP.x86_64 0:1.4.8-1.5.amzn1             gd.x86_64 0:2.0.35-11.10.amzn1
  gperftools-libs.x86_64 0:2.0-11.5.amzn1    libXpm.x86_64 0:3.5.10-2.9.amzn1
  libunwind.x86_64 0:1.1-10.8.amzn1

Complete!
[ec2-user@ip-172-31-42-39 ~]$ sudo service nginx start
Starting nginx:                                            [  OK  ]
[ec2-user@ip-172-31-42-39 ~]$ sudo service nginx reload
Reloading nginx:                                           [  OK  ]
[ec2-user@ip-172-31-42-39 ~]$ cd /usr/share/nginx/html
[ec2-user@ip-172-31-42-39 html]$ ls
404.html  50x.html  index.html  nginx-logo.png  poweredby.png
[ec2-user@ip-172-31-42-39 html]$
```

Open the index.html by “sudo vi index.html” and I wrote following custom information.

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
    <body>
        <h1>[SERVER_ID]</h1>
    </body>
</html>
```

Here, I wrote Server 1 instead of [SERVER_ID].

ec2-user@ip-172-31-42-39:/usr/share/nginx/html

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<body>
<h1>SERVER_1</h1>
</body>
</html>
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"index.html" 5L, 99C
```

I visited public DNS of server 1 EC2 instance, it opened the following webpage.

ec2-52-38-169-241.us-west-2.compute.amazonaws.com

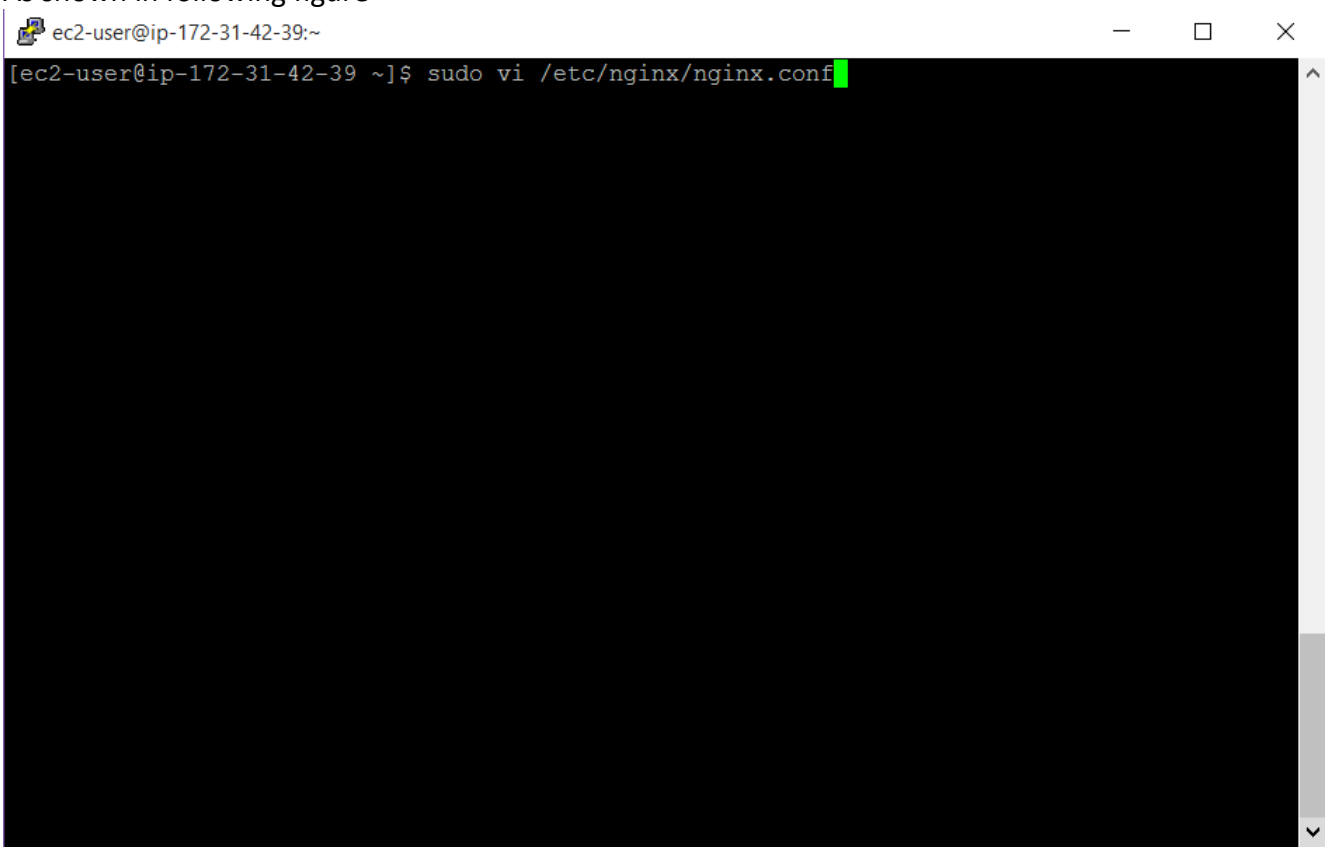
SERVER_1

I did the same thing (changing the index.html and visiting public DNS of that instances) for other 3 instances.

Step 3: Configure the Load balancer

Make load balancer and configure it.
To config load balancer I open nginx.config file using /etc/nginx/nginx.conf

As shown in following figure



```
ec2-user@ip-172-31-42-39:~
# For more information on configuration, see:
#   * Official English Documentation: http://nginx.org/en/docs/
#   * Official Russian Documentation: http://nginx.org/ru/docs/

user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log;
pid /var/run/nginx.pid;

events {
    worker_connections 1024;
}

http {
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    access_log  /var/log/nginx/access.log  main;

    sendfile            on;
    tcp_nopush          on;
    tcp_nodelay         on;
    keepalive_timeout   65;
    types_hash_max_size 2048;

    include             /etc/nginx/mime.types;
"/etc/nginx/nginx.conf" 120L, 3593C
```

Then I replace following code with existing code and this is shown in script.

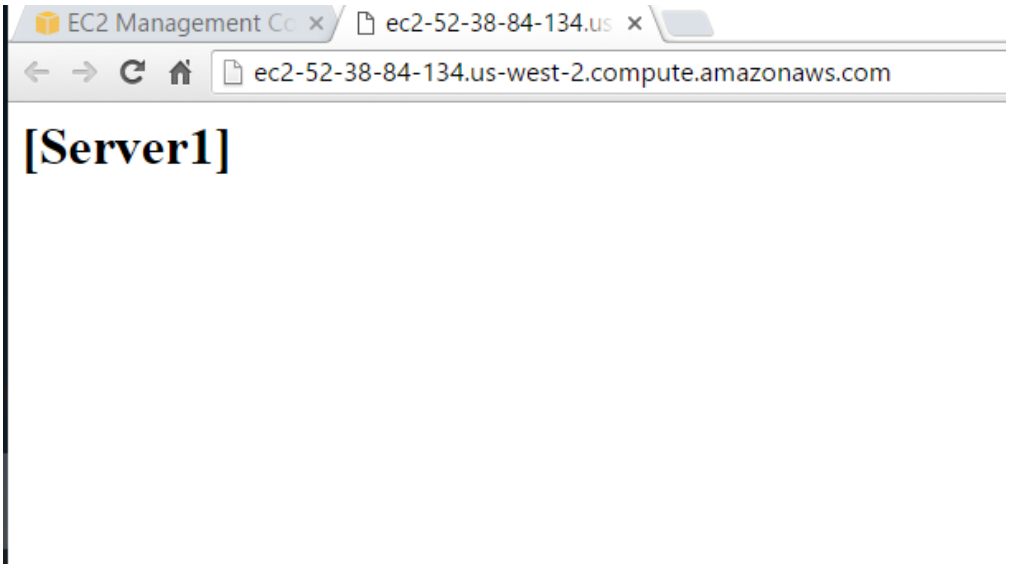
```
events {
        worker_connections 768;
}
http {
        upstream myapp {
        #ip_hash;
                server ec2-52-34-1-31.us-west-2.compute.amazonaws.com weight=1;
                server ec2-52-27-133-122.us-west-2.compute.amazonaws.com weight=1;
                server ec2-52-38-236-64.us-west-2.compute.amazonaws.com weight=1;
                server ec2-52-38-225-253.us-west-2.compute.amazonaws.com weight=1;
        }
        server {
        listen 80;
        server_name myapp.com;
                location / {
                proxy_pass http://myapp;
                }
        }
}
```

```
ec2-user@ip-172-31-35-243:~
  GNU nano 2.5.3          File: /etc/nginx/nginx.conf

events {
worker_connections 768;
}
http {
upstream myapp {
#ip_hash;
server ec2-52-34-1-31.us-west-2.compute.amazonaws.com weight=1;
server ec2-52-27-133-122.us-west-2.compute.amazonaws.com weight=1;
server ec2-52-38-236-64.us-west-2.compute.amazonaws.com weight=1;
server ec2-52-38-225-253.us-west-2.compute.amazonaws.com weight=1;
}
server {
listen 80;
server_name myapp.com;
location / {
proxy_pass http://myapp;
}
}
}
                              [ Read 22 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text^T To Spell  ^_ Go To Line
```

As all of the instances weights are same server by default select any instance and give the one by one other instances' output

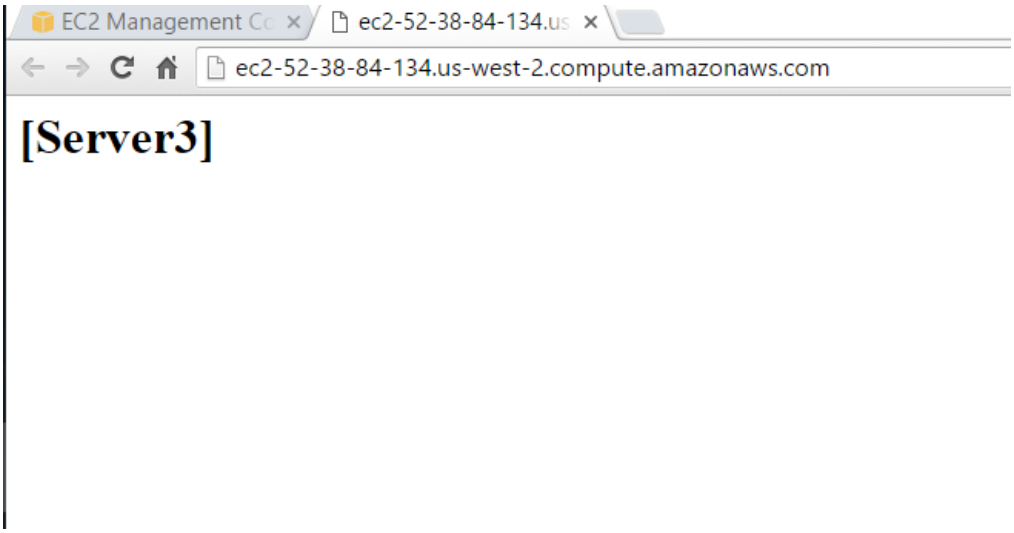
Now, I execute curl ec2-52-38-84-134.us-west-2.compute.amazonaws.com and this curl command distribute traffic among the servers.

Now, I execute this load balancer in browser then it's give following output



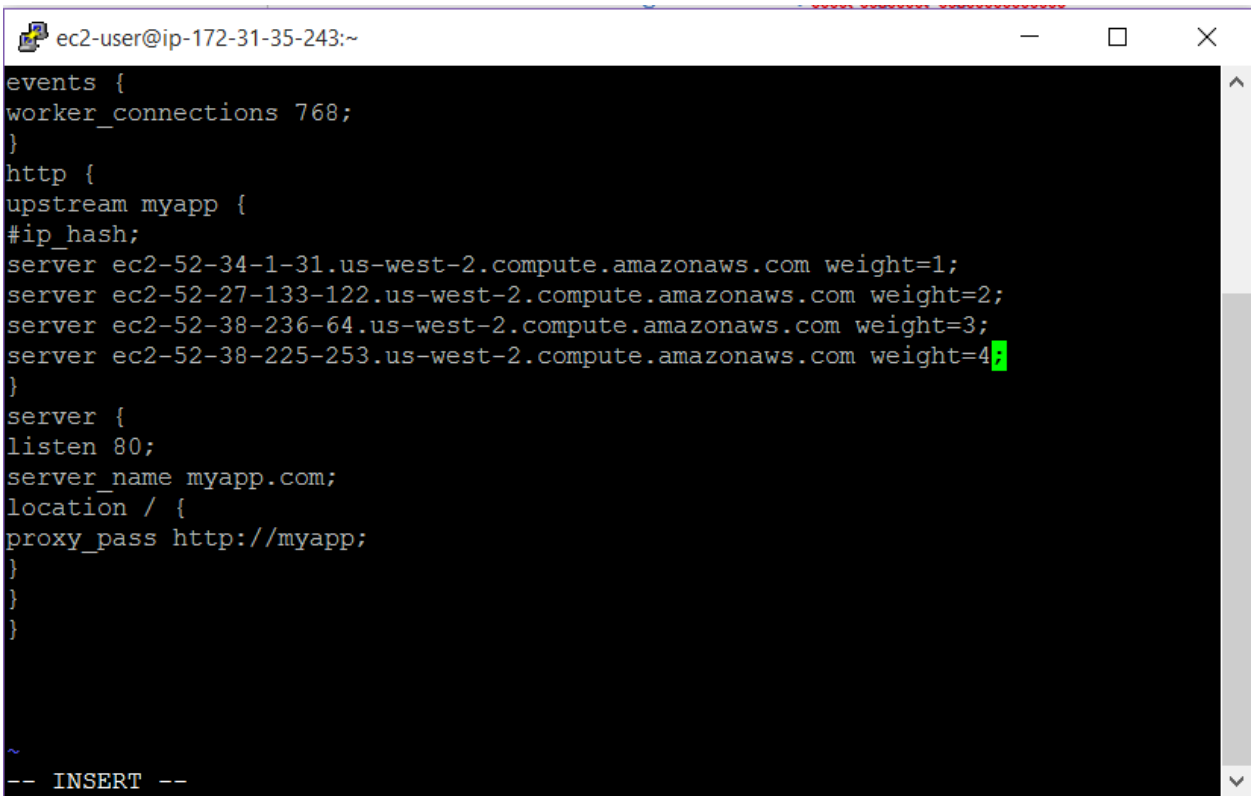
Now when I refresh the same web page it gives following output.

Again I refresh the same web page and it gives following output.



Again I refresh the same web page and it gives following output

Here, in this scenario I give different weight 1,2,3,4 to server.



EC2 Management Co × ec2-52-38-84-134.us × ec2-52-38-84-134.us ×

ec2-52-38-84-134.us-west-2.compute.amazonaws.com

[Server4]

EC2 Management Co × ec2-52-38-84-134.us × ec2-52-38-84-134.us ×

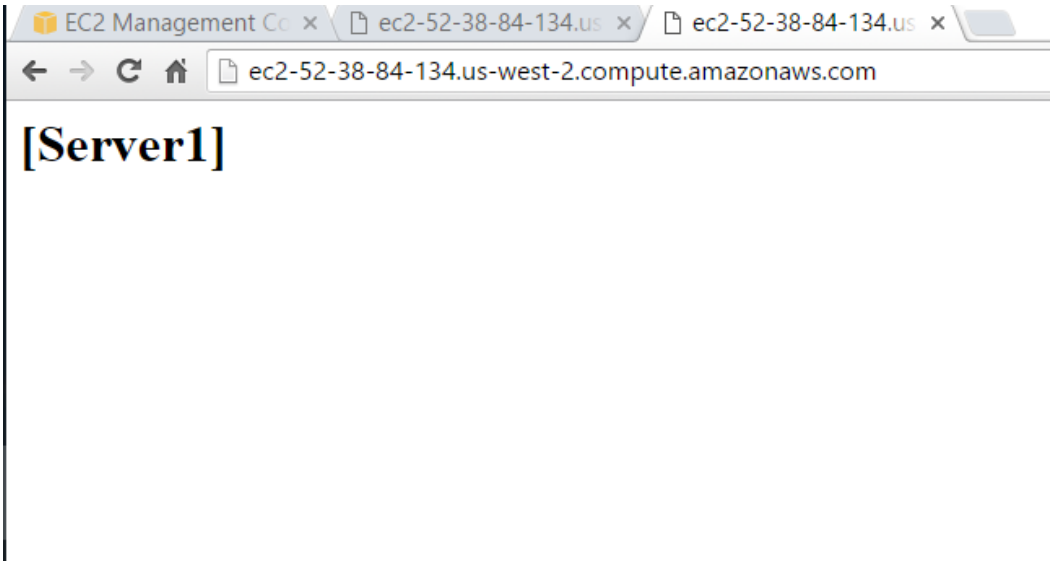
ec2-52-38-84-134.us-west-2.compute.amazonaws.com

[Server3]

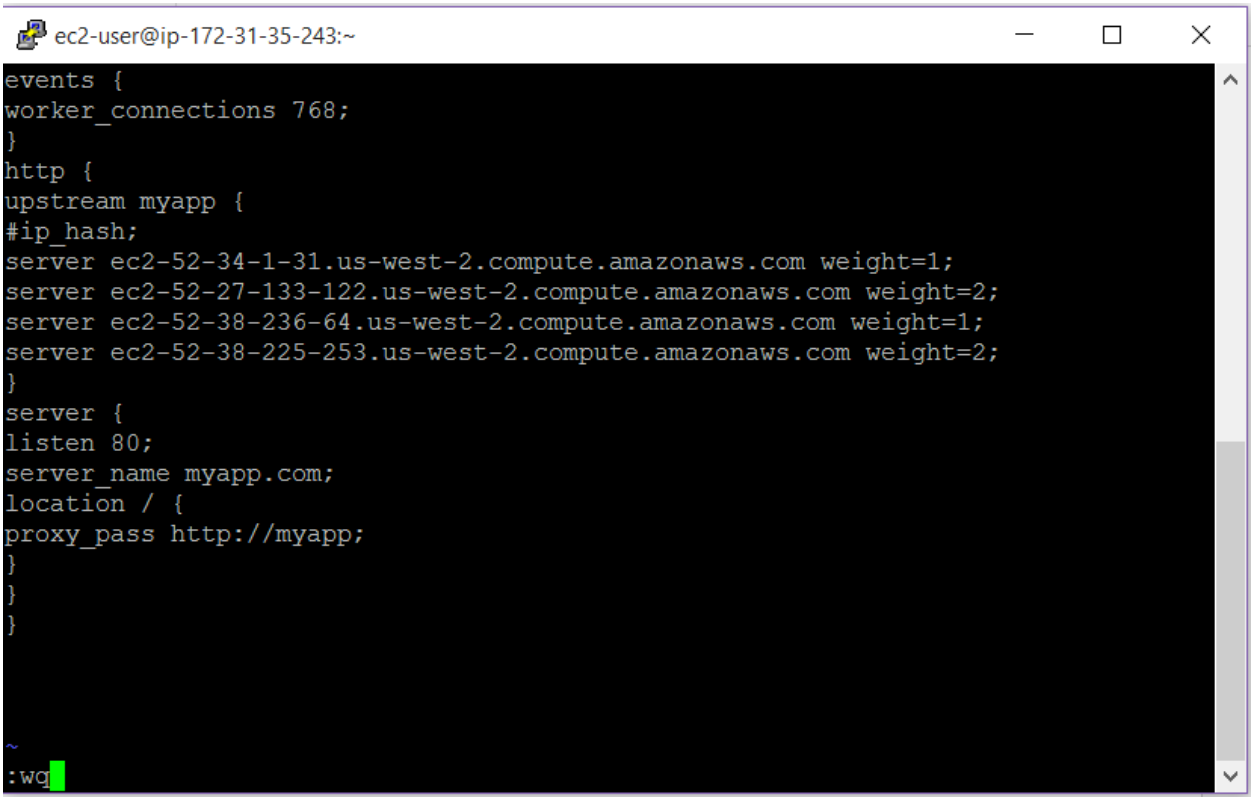
EC2 Management Co × ec2-52-38-84-134.us × ec2-52-38-84-134.us ×

ec2-52-38-84-134.us-west-2.compute.amazonaws.com

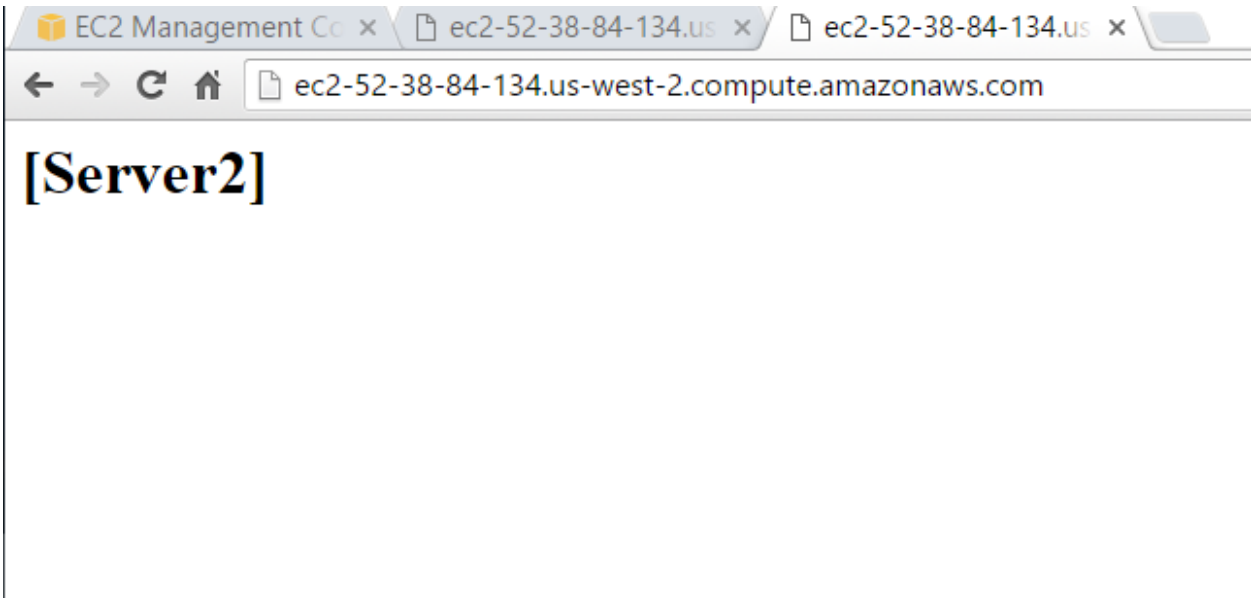
[Server2]

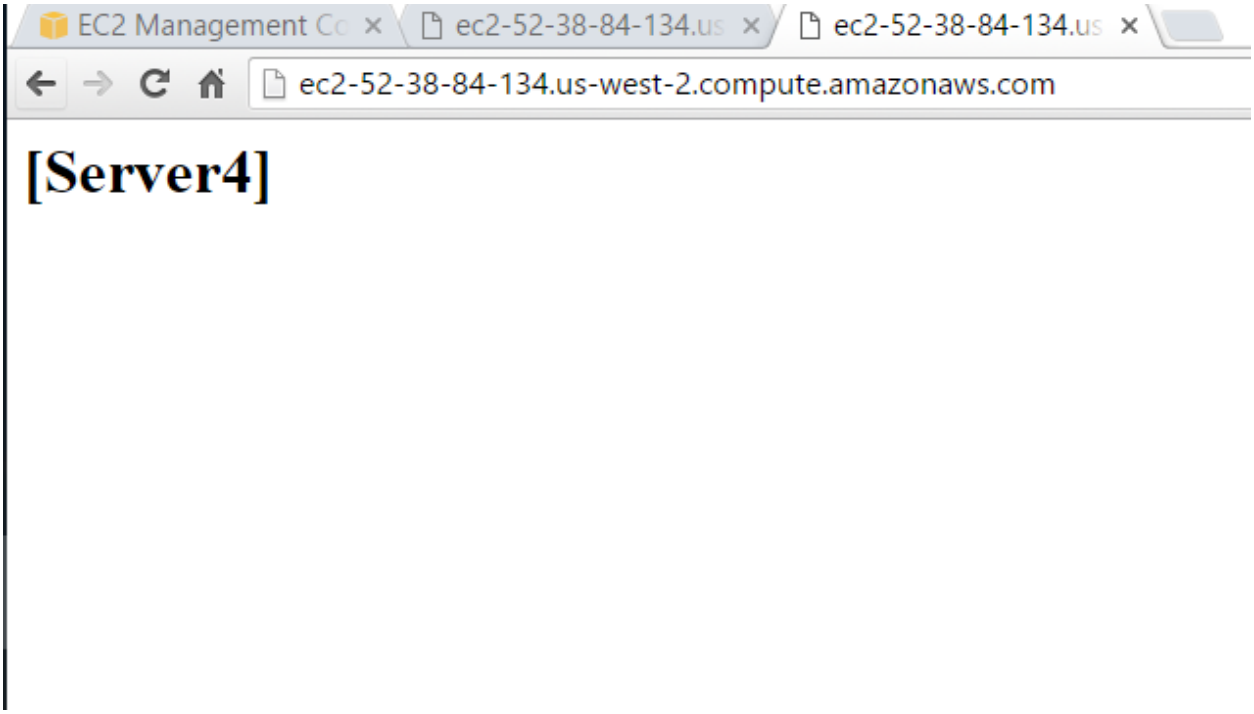


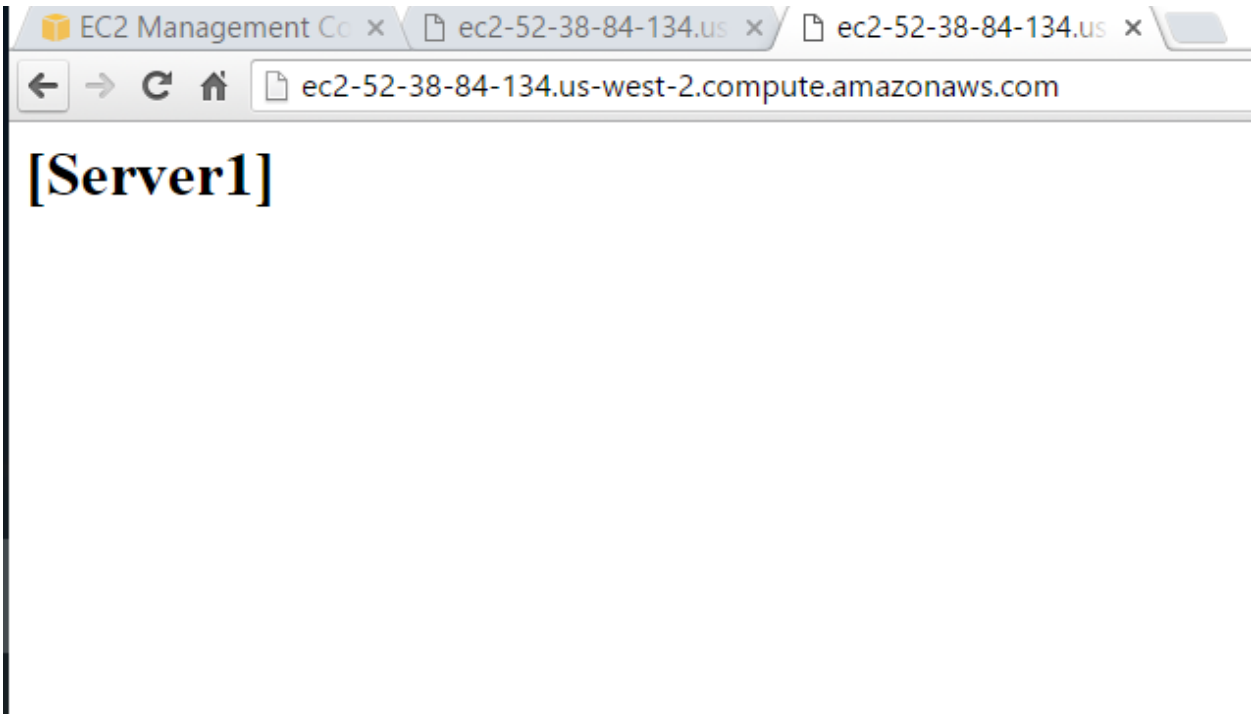
In this scenario I give weight 2 to the server number 2 and 4 and I give weight 1 to server 1 and 3.

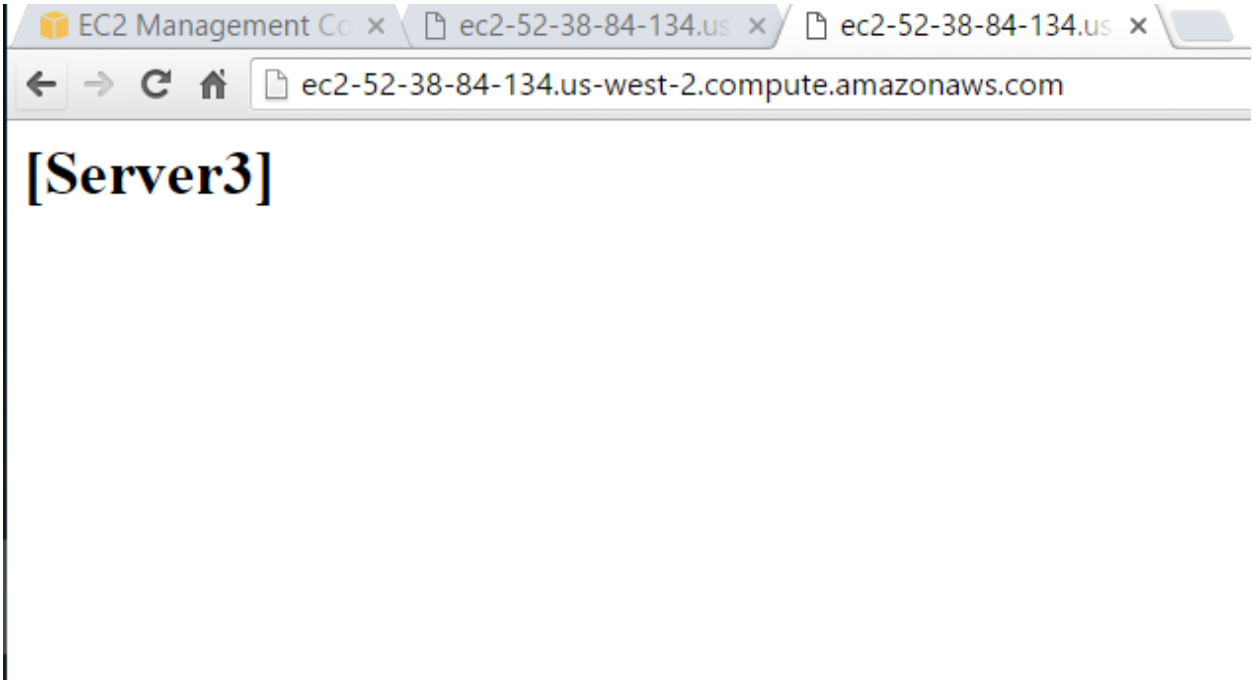


When I execute this load balancer in web browser it display first server 2 and then when refresh the page it displays server 4.
After that continuously refresh pages then I get output for server 3 and then get output for server 1


EC2 Management Co
ec2-52-38-84-134.us
ec2-52-38-84-134.us
ec2-52-38-84-134.us-west-2.compute.amazonaws.com
[Server2]


EC2 Management Co
ec2-52-38-84-134.us
ec2-52-38-84-134.us
ec2-52-38-84-134.us-west-2.compute.amazonaws.com
[Server4]

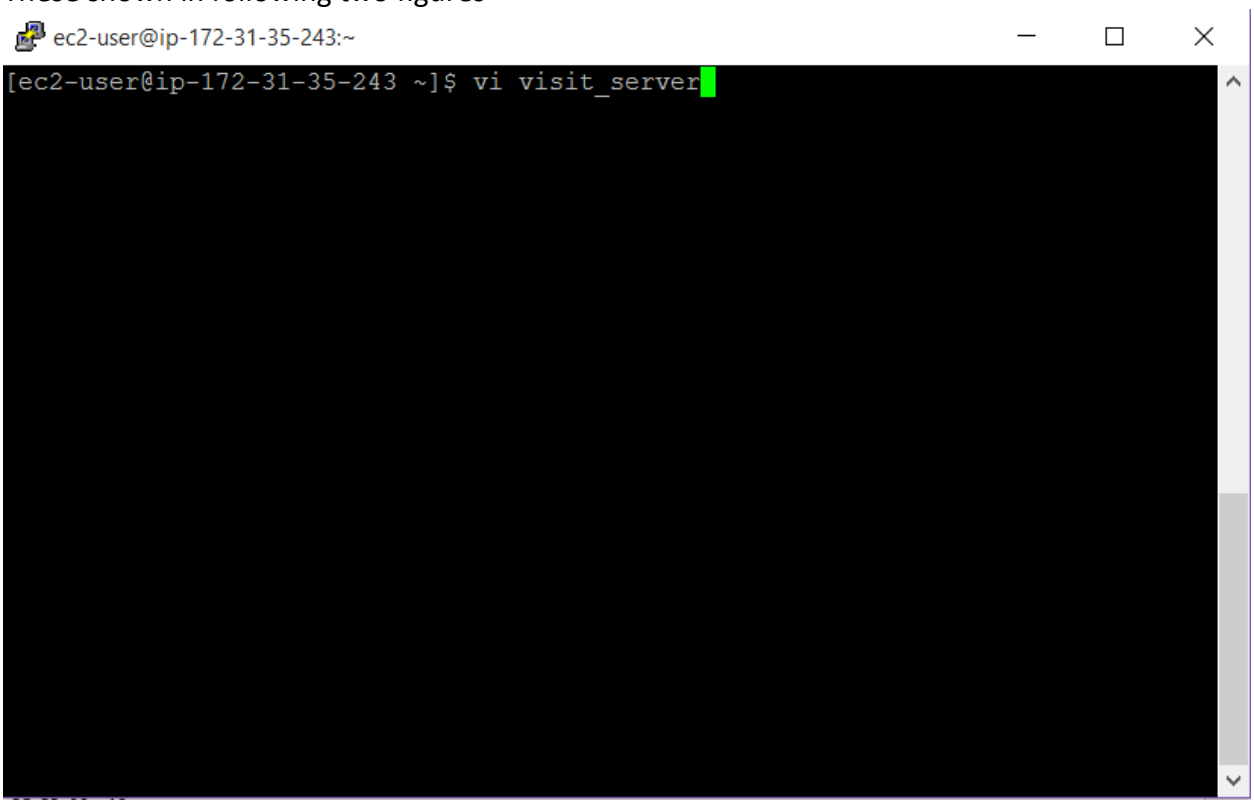




Step 4: Collect the information on visits to your site

You can use the *visit server* tool, provided in the Appendix, or write your own tool, to track the distribution of the load. The tool visits the cluster 2000 times and returns the visit count on each server.

Collect the information on visits on my site using visit server tool and apply the given 3 scenario
For this first I made visit_server named file and then typed visit server program.
These shown in following two figures



I have replaced the code with given Ruby code in homework.

```
ec2-user@ip-172-31-35-243:~
#!/usr/bin/env ruby
#
# This program is used for collecting web server visit information.
#
# Author: A. Genius
#
require 'optparse'
def print_usage
puts "USAGE: visit_server -d DNS_NAME"
exit
end

# add option switch and handler
options = {}
option_parser = OptionParser.new do |opts|

# DNS_NAME argument
options[:dns_name] = nil
opts.on('-d', '--dns-name DNS_NAME', 'Specify a DNS NAME') { |dns_name| options[
:dns_name] = dns_name }

# HELP argument
options[:help] = nil
-- INSERT --                                                    1,1           Top
```

With the chmod command we can change the access permissions of files and folders.
For example **chmod 770 visit_server**

```
ec2-user@ip-172-31-35-243:~
[ec2-user@ip-172-31-35-243 ~]$ vi visit_server
[ec2-user@ip-172-31-35-243 ~]$ ls
visit_server
[ec2-user@ip-172-31-35-243 ~]$ chmod 770 visit_server
```

Then run ls –l command to traverse subdirectories to create a list of files with the pathname which gives following information:

mode
A one byte entry type followed by three bytes each of Owner, Group, and Other permissions which is rwxrwx--- in following output.

link count
A count of the number of symbolic links to a file which is 1 in following output.

owner
The login or user name that owns a file or directory which is ec2-user in following output.

group
The group name with which permissions are associated which is ec2-user in following output.

size
The size (in bytes) of a file or directory which is 1356 in following output.

date
Date of last modification (month and day, provided in the form mmm nn) which is Apr 13 in following output.

time
Time of last modification, provided in the form hh:mm (for files greater than six months old,hh:mm is replaced with the year in which the file was last modified, provided as nnnn) which is 21:29 in following output.

name
The name of a file or directory which is visit_server in following output.

```
ec2-user@ip-172-31-35-243:~
[ec2-user@ip-172-31-35-243 ~]$ vi visit_server
[ec2-user@ip-172-31-35-243 ~]$ ls
visit_server
[ec2-user@ip-172-31-35-243 ~]$ chmod 770 visit_server
[ec2-user@ip-172-31-35-243 ~]$ ls -l
total 4
-rwxrwx--- 1 ec2-user ec2-user 1356 Apr 13 21:29 visit_server
[ec2-user@ip-172-31-35-243 ~]$
```

Here, I execute this command for SCENARIO 1 in which all the server have same weight 1.

```
ec2-user@ip-172-31-35-243:~
events {
worker_connections 768;
}
http {
upstream myapp {
#ip_hash;
server ec2-52-34-1-31.us-west-2.compute.amazonaws.com weight=1;
server ec2-52-27-133-122.us-west-2.compute.amazonaws.com weight=1;
server ec2-52-38-236-64.us-west-2.compute.amazonaws.com weight=1;
server ec2-52-38-225-253.us-west-2.compute.amazonaws.com weight=1;
}
server {
listen 80;
server_name myapp.com;
location / {
proxy_pass http://myapp;
}
}
}



~
-- INSERT --
```

```
ec2-user@ip-172-31-35-243:~
elsif o =~ /server\s*3/i
server3_visit_count += 1
elsif o =~ /server\s*4/i
server4_visit_count += 1
end
print "."
end

# redirect output to stdout
#$stdout = orig_stdout
# print visit information

puts
puts '-----------------------------------------'
puts ' SUMMARY '
puts '-----------------------------------------'
puts "Server1 visit counts: " + server1_visit_count.to_s
puts "Server2 visit counts: " + server2_visit_count.to_s
puts "Server3 visit counts: " + server3_visit_count.to_s
puts "Server4 visit counts: " + server4_visit_count.to_s
puts "Total visit counts: " + (server1_visit_count + server2_visit_count + server3_visit_count + server
4_visit_count ).to_s

-- INSERT --                                                    81,7          Bot
```

```
ec2-user@ip-172-31-35-243:~
[ec2-user@ip-172-31-35-243 ~]$ sudo ./visit_server -d ec2-52-38-84-134.us-west-2.compute.amazonaws.com
Starting to visit load balancing server
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................
-----------------------------------------
 SUMMARY
-----------------------------------------
Server1 visit counts: 500
Server2 visit counts: 500
Server3 visit counts: 500
Server4 visit counts: 500
Total visit counts: 2000
[ec2-user@ip-172-31-35-243 ~]$
```

Next execute the visit_server –d [MY LOAD BALANCER DNS NAME] for the SCENARIO 2 in which server 1, 2, 3, 4 has weight 1, 2, 3, 4 respectively.

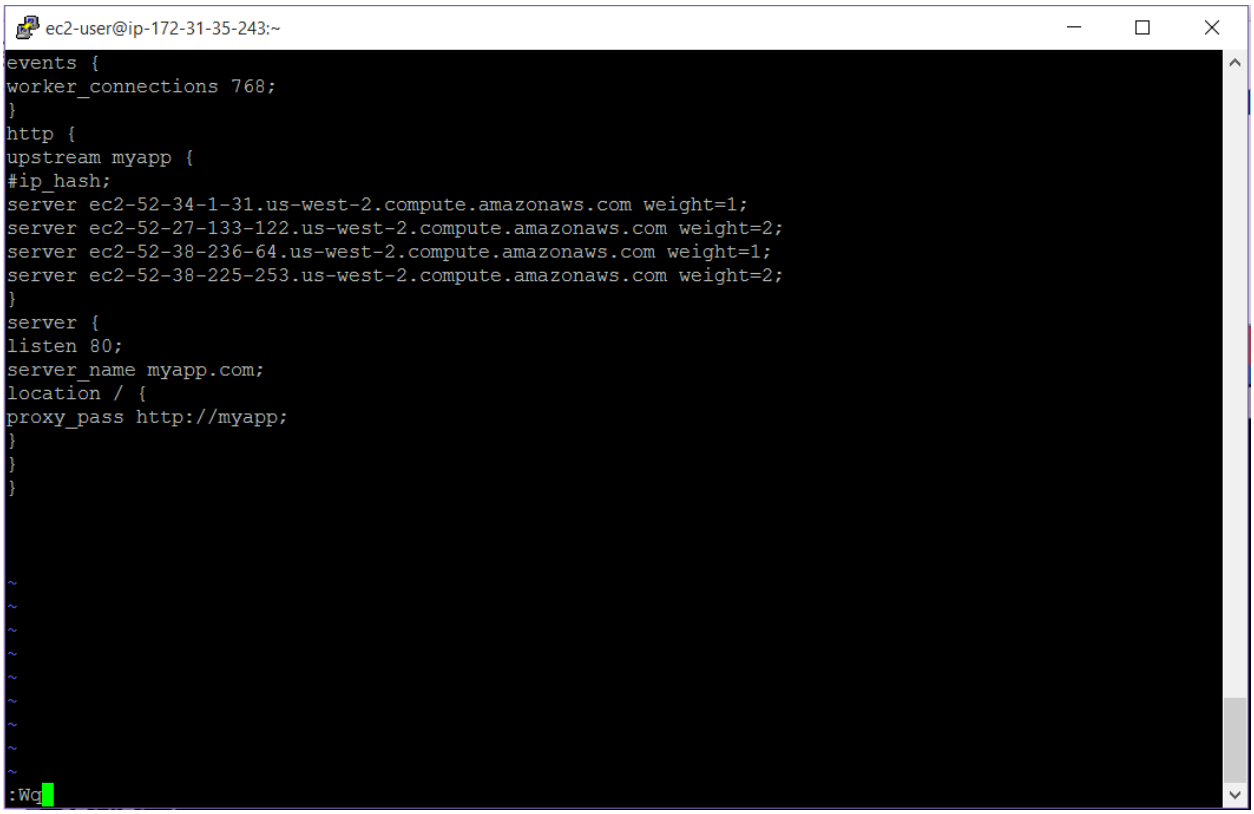
ec2-user@ip-172-31-35-243:~

```
events {
worker_connections 768;
}
http {
upstream myapp {
#ip_hash;
server ec2-52-34-1-31.us-west-2.compute.amazonaws.com weight=1;
server ec2-52-27-133-122.us-west-2.compute.amazonaws.com weight=2;
server ec2-52-38-236-64.us-west-2.compute.amazonaws.com weight=3;
server ec2-52-38-225-253.us-west-2.compute.amazonaws.com weight=4;
}
server {
listen 80;
server_name myapp.com;
location / {
proxy_pass http://myapp;
}
}
}



~
~
~
~
~
~
~
~
~
:wq
```

ec2-user@ip-172-31-35-243:~

```
[ec2-user@ip-172-31-35-243 ~]$ sudo ./visit_server -d ec2-52-38-84-134.us-west-2.compute.amazonaws.com
Starting to visit load balancing server
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................................................................................
.....................................
-----------------------------------------
 SUMMARY
-----------------------------------------
Server1 visit counts: 200
Server2 visit counts: 400
Server3 visit counts: 600
Server4 visit counts: 800
Total visit counts: 2000
[ec2-user@ip-172-31-35-243 ~]$
```

Next execute the visit_server –d [MY LOAD BALANCER DNS NAME] for the SCENARIO 3 in which server 1 and 3 has weight 1; and server 2 and 4 has weight 2.



ec2-user@ip-172-31-35-243:~

```
[ec2-user@ip-172-31-35-243 ~]$ sudo ./visit_server -d ec2-52-38-84-134.us-west-2.compute.amazonaws.com
Starting to visit load balancing server
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................
-----------------------------------------
 SUMMARY
-----------------------------------------
Server1 visit counts: 333
Server2 visit counts: 667
Server3 visit counts: 333
Server4 visit counts: 667
Total visit counts: 2000
[ec2-user@ip-172-31-35-243 ~]$
```

ec2-user@ip-172-31-42-39:/usr/share/nginx

```
[ec2-user@ip-172-31-42-39 html]$ ls -l
total 16
-rw-r--r-- 1 root root 3696 Feb 19 22:52 404.html
-rw-r--r-- 1 root root 3738 Feb 19 22:52 50x.html
-rw-r--r-- 1 root root 4012 Apr 14 15:27 index.html
-rw-r--r-- 1 root root  368 Feb 19 22:52 nginx-logo.png
lrwxrwxrwx 1 root root   32 Apr 13 02:42 poweredby.png -> /usr/share/pixmaps/pow
eredby.png
[ec2-user@ip-172-31-42-39 html]$ cd ..
[ec2-user@ip-172-31-42-39 nginx]$ ls -l
total 4
drwxr-xr-x 2 root root 4096 Apr 14 15:27 html
[ec2-user@ip-172-31-42-39 nginx]$ ^C
[ec2-user@ip-172-31-42-39 nginx]$ chmod 777 html/
chmod: changing permissions of 'html/': Operation not permitted
[ec2-user@ip-172-31-42-39 nginx]$ sudo chmod 777 html/
[ec2-user@ip-172-31-42-39 nginx]$ ls -l
total 4
drwxrwxrwx 2 root root 4096 Apr 14 15:27 html
[ec2-user@ip-172-31-42-39 nginx]$ sudo chmod 755 html/
[ec2-user@ip-172-31-42-39 nginx]$ ls -l
total 4
drwxr-xr-x 2 root root 4096 Apr 14 15:48 html
[ec2-user@ip-172-31-42-39 nginx]$
```

All additional perception:

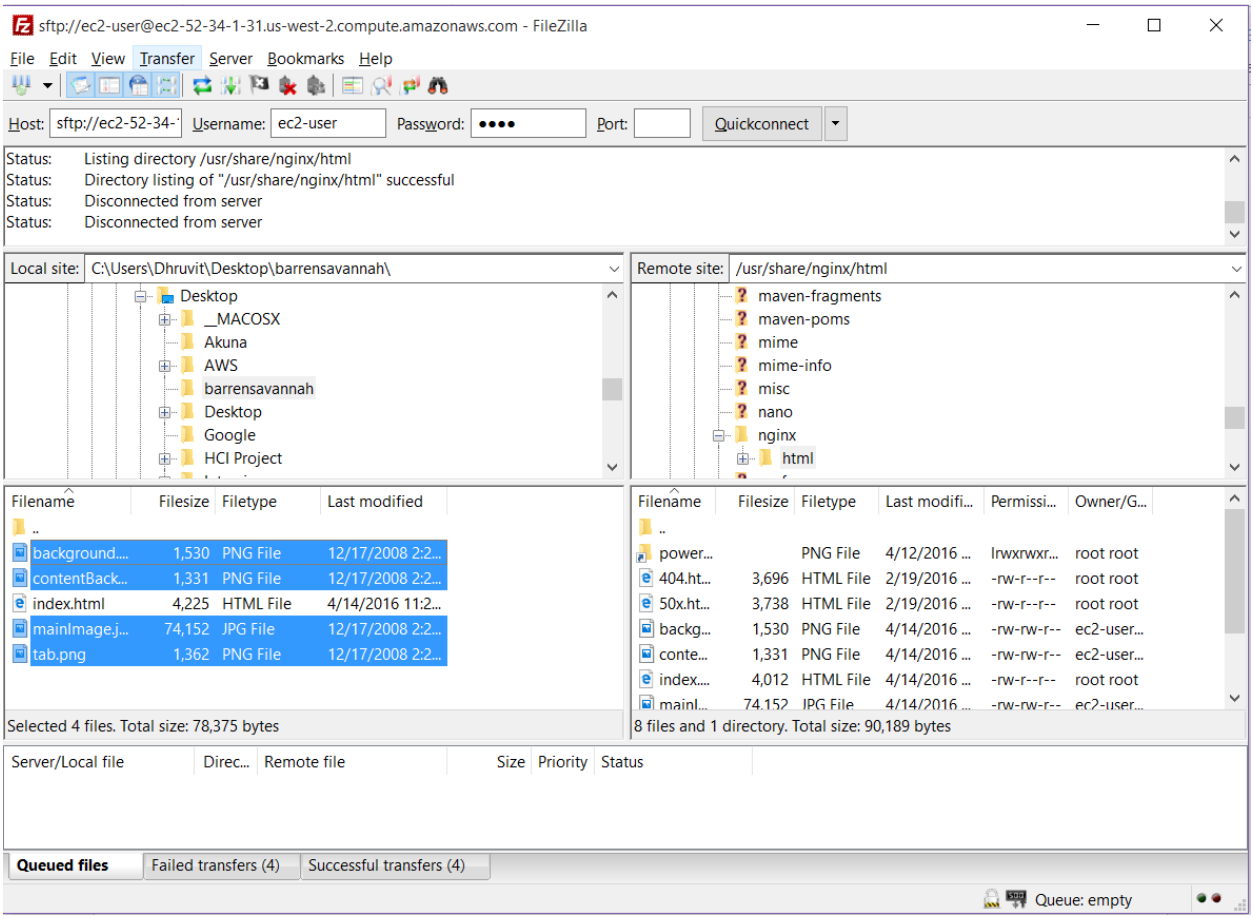
- During this assignment I watched that when I prevent occurrences from running mode and afterward begin it again then Public DNS Name changed inevitably.

- Second thing is that, after configuring load balancer for every yield I simply need to execute load balancer instance in browser.

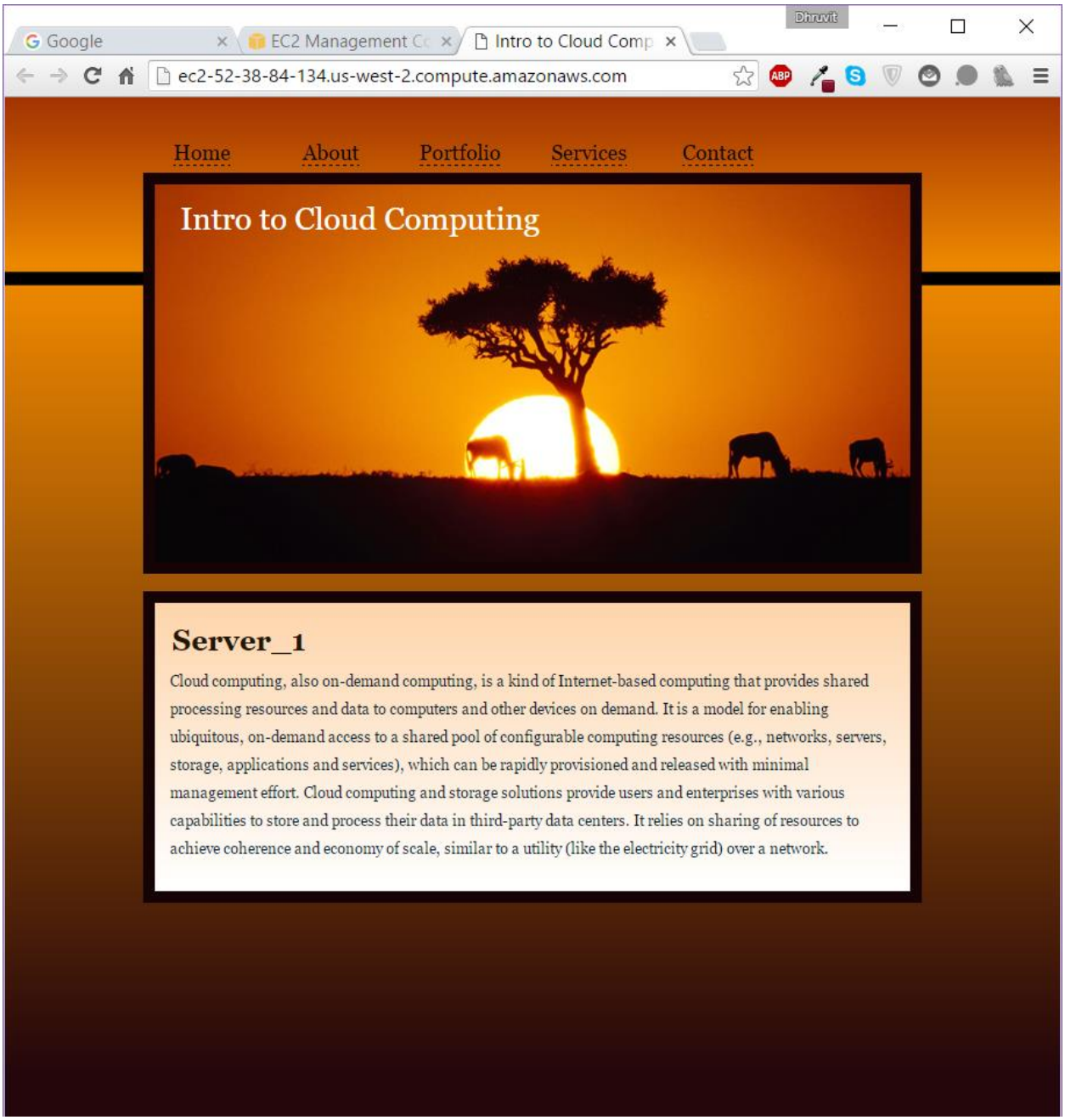
In the wake of completing every one of the steps I make little site using html. In this code I used assorted label like title, div, head, body, ul, h1, h2, p, html and other particular tag.

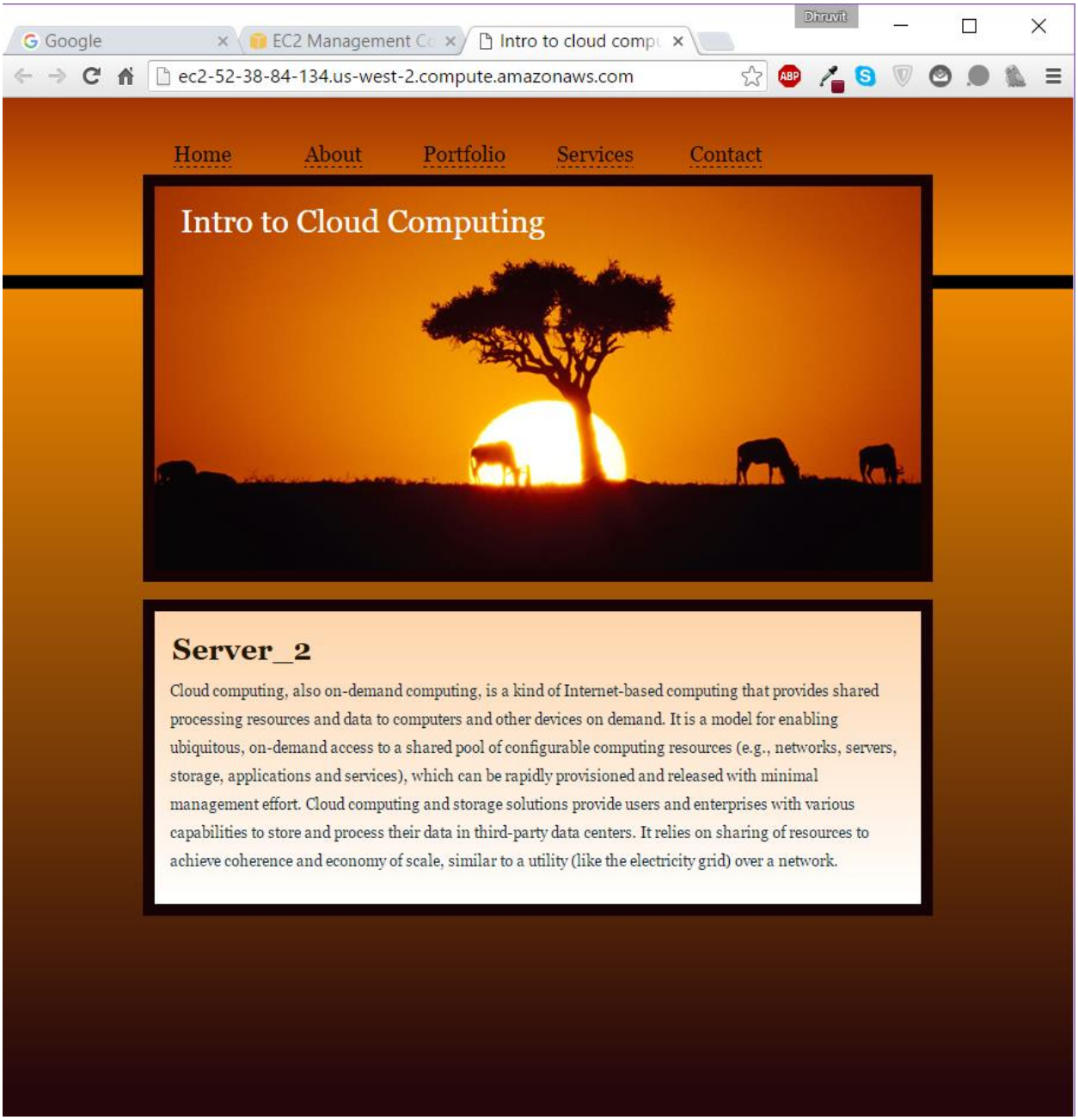
I put this code on Server 1 (instance of Amazon) and run the load balancer record after change outline of load balancer with the objective that piles are flowed in this site and numbers visit server fittingly.

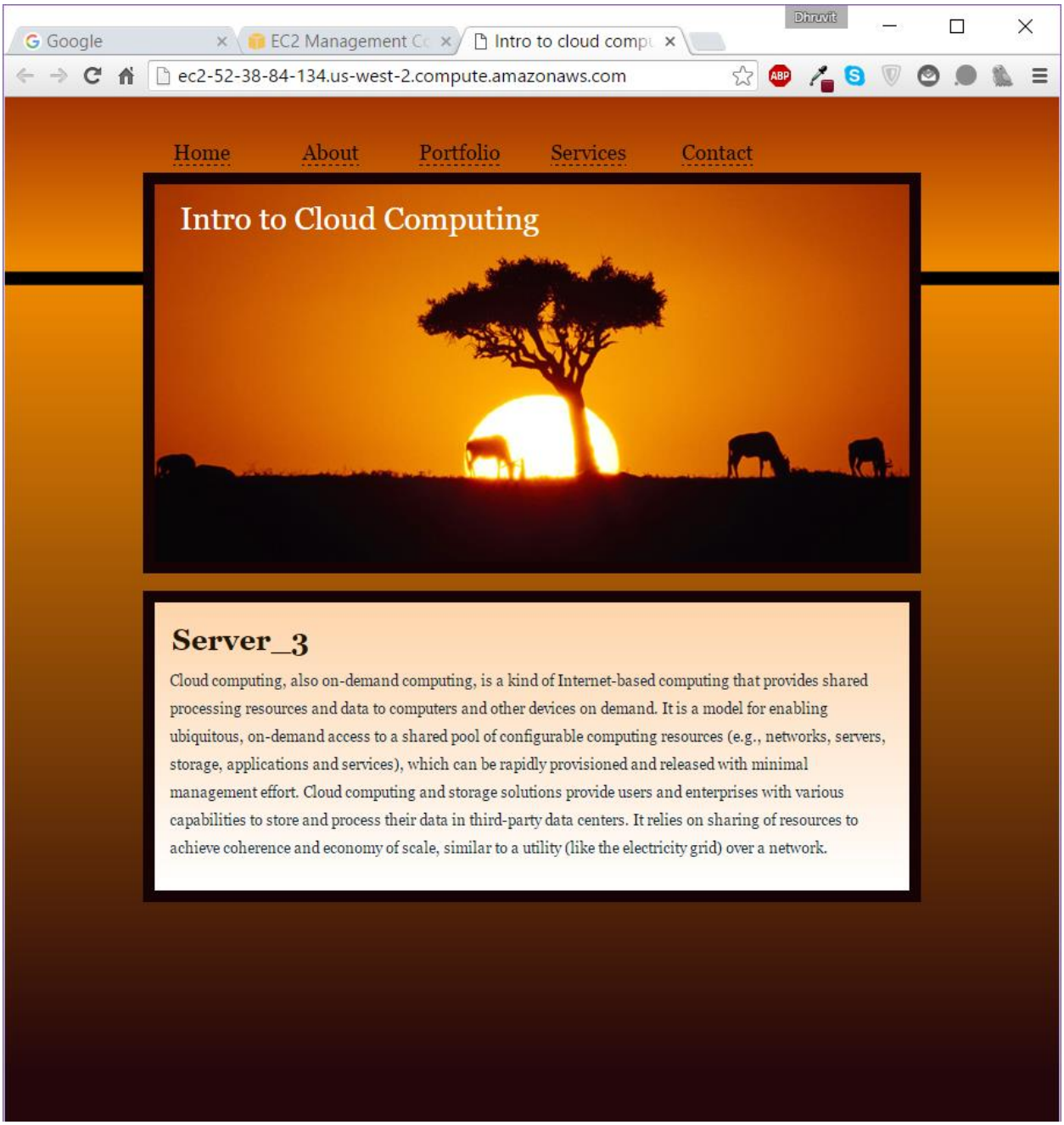
I used FileZilla s/w to transfer my own website to all four instances.

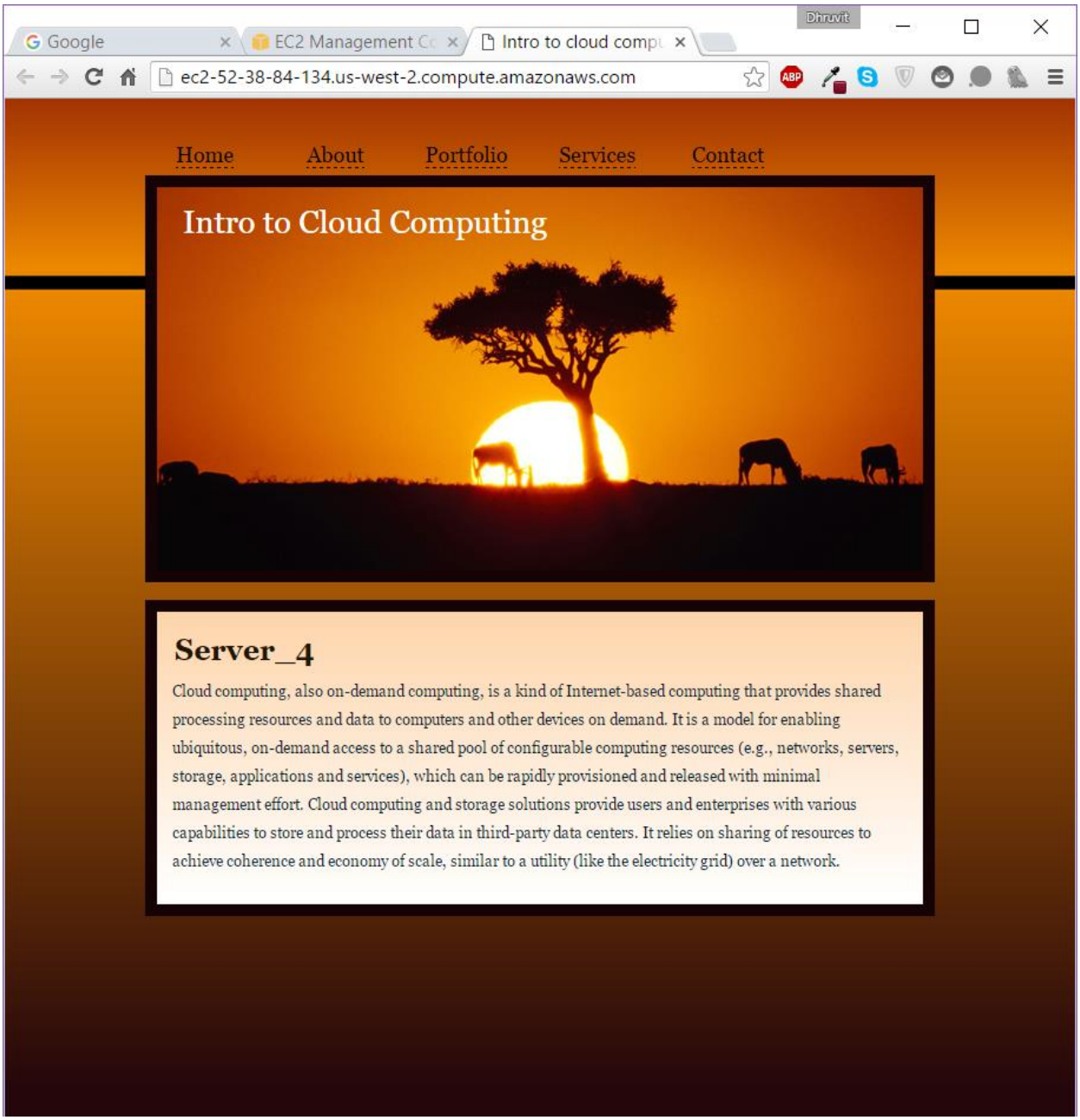


I have put that code on all the server and when I refreshed load balancer I got following output.


Google
EC2 Management Co
Intro to Cloud Comp
Dhruvit
ec2-52-38-84-134.us-west-2.compute.amazonaws.com
Home
About
Portfolio
Services
Contact
Intro to Cloud Computing
Server_1
Cloud computing, also on-demand computing, is a kind of Internet-based computing that provides shared processing resources and data to computers and other devices on demand. It is a model for enabling ubiquitous, on-demand access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services), which can be rapidly provisioned and released with minimal management effort. Cloud computing and storage solutions provide users and enterprises with various capabilities to store and process their data in third-party data centers. It relies on sharing of resources to achieve coherence and economy of scale, similar to a utility (like the electricity grid) over a network.


Google
EC2 Management Co
Intro to cloud compu
Dhruvit
ec2-52-38-84-134.us-west-2.compute.amazonaws.com
Home
About
Portfolio
Services
Contact
Intro to Cloud Computing
Server_2
Cloud computing, also on-demand computing, is a kind of Internet-based computing that provides shared processing resources and data to computers and other devices on demand. It is a model for enabling ubiquitous, on-demand access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services), which can be rapidly provisioned and released with minimal management effort. Cloud computing and storage solutions provide users and enterprises with various capabilities to store and process their data in third-party data centers. It relies on sharing of resources to achieve coherence and economy of scale, similar to a utility (like the electricity grid) over a network.


Google
EC2 Management Co
Intro to cloud compu
Dhruvit
ec2-52-38-84-134.us-west-2.compute.amazonaws.com
Home
About
Portfolio
Services
Contact
Intro to Cloud Computing
Server_3
Cloud computing, also on-demand computing, is a kind of Internet-based computing that provides shared processing resources and data to computers and other devices on demand. It is a model for enabling ubiquitous, on-demand access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services), which can be rapidly provisioned and released with minimal management effort. Cloud computing and storage solutions provide users and enterprises with various capabilities to store and process their data in third-party data centers. It relies on sharing of resources to achieve coherence and economy of scale, similar to a utility (like the electricity grid) over a network.



References:

http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/install-software.html
http://angus.readthedocs.org/en/2014/amazon/transfer-files-between-instance.html