# Nginx Scenario Task :

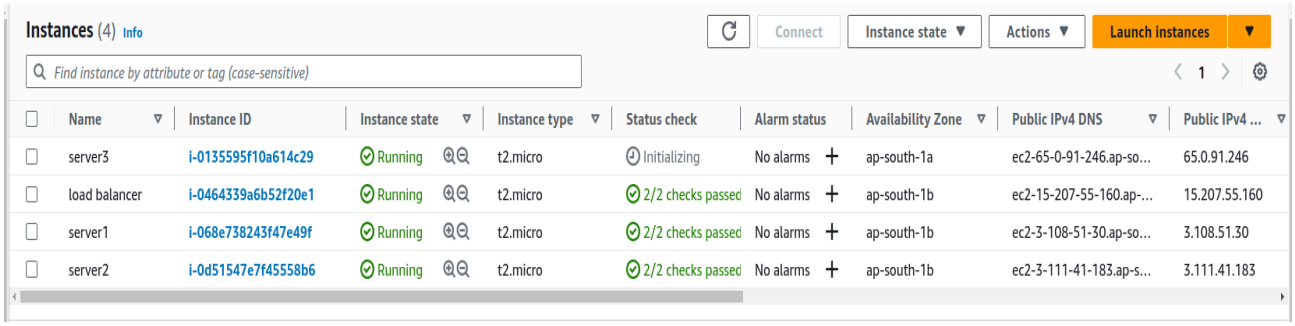
Configure an Nginx server to act as a load balancer and distribute traffic to other backend servers

Load Balancer:

It is a feature or a module that distributes incoming network traffic across multiple servers or backend nodes in order to improve availability and overall performance of a system or application.

STEP 1:

Deploy 4 servers, Load balancer server, server1, server2, server3 being the names of each instance.




STEP 2:

- Configure load balancer server by installing nginx server and then configuring it to act as a load balancer.

→ sudo apt update
→ sudo apt install nginx -y
→ sudo nano /etc/nginx/nginx.conf

- Add the following code inside of nginx.conf in http block-

```
    upstream backend {
        server 3.108.51.30;
        server 3.111.41.183;
        server 65.0.91.246;
    }
 server {
        listen 80;

        location / {
            proxy_pass http://backend;
        }
    }
}
```

- Save and exit the nginx.conf file

- Test the configuration file
  → sudo nginx -t

```
ubuntu@ip-150-0-0-50:~$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
ubuntu@ip-150-0-0-50:~$
```

- Delete the default configuration file in sites-enabled
  → sudo rm /etc/nginx/sites-enabled/default

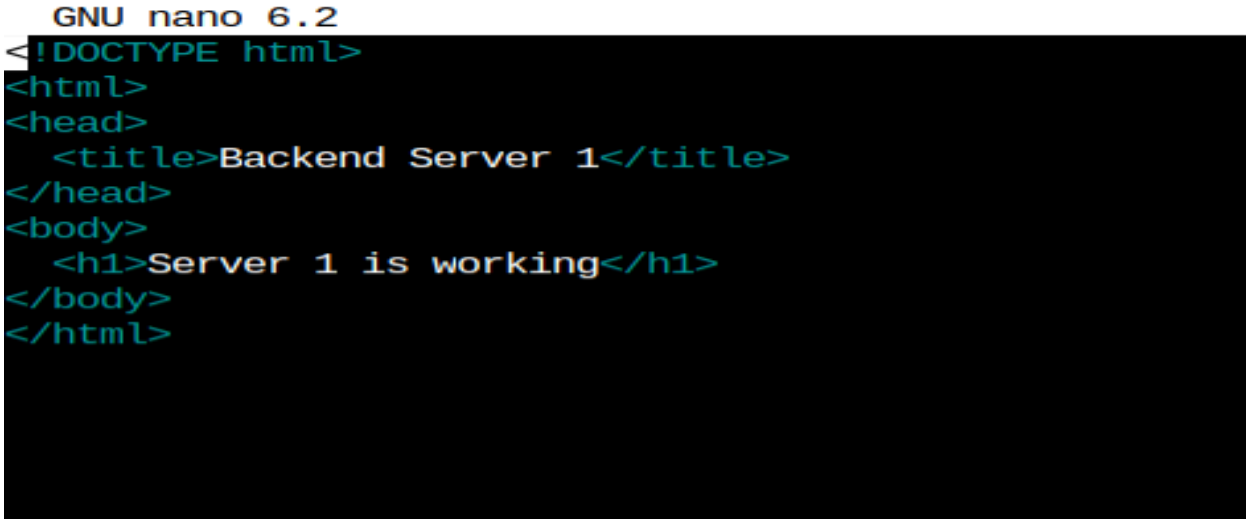
- Restart the nginx server
  →sudo service nginx restart

- Close the load balancer server

STEP 3:

- Connect to Server1 and configure nginx to set up a sample html website
    → sudo apt update
    → sudo apt install nginx -y
- Create Index.html file inside of new directory called server1
    → cd /var/www
    → sudo mkdir server1
    → sudo nano server1/index.html

- Add the following code inside the index.html file




- Save and exit the html file and then create the configuration file for the server1.
    → sudo nano /etc/nginx/sites-available/server1

- Add the following code inside the server1 configuration file

```
  GNU nano 6.2
server {
    listen 80;
    root /var/www/server1;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

- Save the file and exit

- Check for syntax errors in the file
  → sudo nginx -t

```
ubuntu@ip-150-0-0-47:~$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
ubuntu@ip-150-0-0-47:~$
```

- Enable the server1 site for nginx
  →sudo ln -s /etc/nginx/sites-available/server1 /etc/nginx/sites-enabled/

- The server1 site will then be available on sites-enabled.

```
ubuntu@ip-150-0-0-47:~$ cd /etc/nginx/sites-enabled/
ubuntu@ip-150-0-0-47:/etc/nginx/sites-enabled$ ls
server1
ubuntu@ip-150-0-0-47:/etc/nginx/sites-enabled$
```

- Delete the default configuration file in sites-enabled
  → sudo rm /etc/nginx/sites-enabled/default

STEP 5:

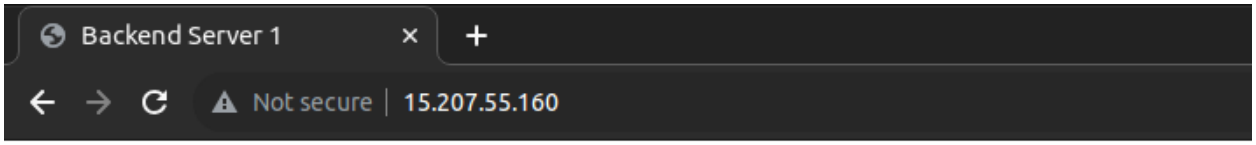
- Follow the same procedure for other 2 servers as well
- Edit the index.html with “Backend server 2 ” and “Backend server 3” respectively for the other 2 servers.
- Also edit the configuration file of both server2 and server3 inside of sites-available with the necessary path root as “/var/www/server2” and “/var/www/server3” respectively.

STEP 6:

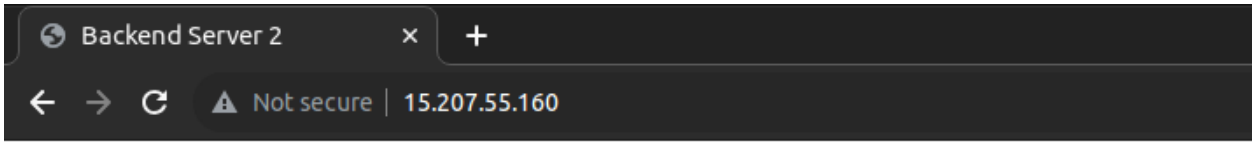
- After configuring all the 3 servers, copy the ip-address of the load balancer server and paste it in the address bar of the browser.

- We will see that, with the same ip-address of the load balancer server, we are able to access the contents of both server1 ,server2 and server3

- This proves that nginx load balancer server is distributing traffic evenly to all the 3 servers.




# Server 1 is working




# Server 2 is working