1. Launch an AWS Instance for Servers
   * 1. Name: Server1
     2. AMI: Amazon Linux 2023 AMI
     3. Architecture: 64bit (x86)

A screenshot of a computer

Description automatically generated

* + 1. Instance Type: t2.micro (Free tier eligible)
    2. Key Pair Name: Owais\_L02 (this was reused from previous lab)
    3. Security group with the below config:

A screenshot of a computer

Description automatically generated

With this configuration:

There are three types of inbound rules:

1. SSH at port 22 from anywhere on the internet (0.0.0.0/0)
2. HTTP at port 80 from anywhere on the internet (0.0.0.0/0)
3. HTTPS at port 443 from anywhere on the internet (0.0.0.0/0)
   * 1. Launch the instance, using the aws cli command   
        ``` aws ec2 run-instances --image-id ami-0900fe555666598a2 --security-group-ids sg-056a4fe84079135c7 --count 5 --instance-type t2.micro --key-name Owais\_L02 ```  
        After launching, Connect to the launched instance using SSH or AWS EC2 instance connect and execute the following commands:

sudo dnf install nginx -y

sudo cd /usr/share/nginx/html

sudo sed -i 's/Welcome to nginx/Owais Nginx SERVER\_NUMBER/g' ./index.html

sudo systemctl start nginx

sudo systemctl enable nginx

sudo systemctl status nginx

Step by step output of each command is:

1. sudo dnf install nginx -y #this command will install nginx on instance

A screenshot of a computer

Description automatically generated

1. sudo cd /usr/share/nginx/html #this site holds the host page of the html server when accessed from the internet. Update the title and h1 tag with the Server name tag to distinguish between different servers from the load balancer.
2. sudo sed -i 's/Welcome to nginx/Owais Nginx SERVER\_NAME/g' ./index.html # this command replaces the Welcome to nginx with Owais Nginx SERVER\_NAME in the file index.html  
   A screenshot of a computer screen

   Description automatically generated
3. sudo systemctl start nginx # this command starts the nginx service
4. sudo systemctl enable nginx #this commands enable the nginx service everytime the instance reboots



After executing the above code, your nginx service is accessible on the public IP address of your EC2 instance (screenshot attached). Repeat the above steps for configuring Server2, Server3, Server4.

A screenshot of a computer

Description automatically generated

Load Balancer configuration has a slightly different procedure which is given below:

1. execute the command ```sudo dnf install nginx -y```
2. Use any text editor and edit the file located at ```/etc/nginx/nginx.conf```, and make the following changes:
   1. Change the value of server connections to 768. (This can be found under the events indentation)
   2. Add an upsteam block under http indentation  
      upstream myapp {  
       #ip\_hash  
       server SERVER1\_PUBLIC\_DNS weight=1  
       server SERVER2\_PUBLIC\_DNS weight=1  
       server SERVER3\_PUBLIC\_DNS weight=1  
       server SERVER4\_PUBLIC\_DNS weight=1  
      }
   3. Replace the server name under server indentation as myapp.com and add a location block as   
      location / {  
       proxy\_pass http://myapp;  
      }
3. After modification the file should look something as:A screenshot of a computer

   Description automatically generated
4. Execute the command ```sudo systemctl start nginx && sudo systemctl enable nginx```
5. Load Balancer is now configured.  
   NOTE: SERVER\_1 SERVER\_2 SERVER\_3 SERVER\_4 should be replaced with the respective IP Address or Public DNS Name of each EC2 instance.

**METRICS COLLECTION:**

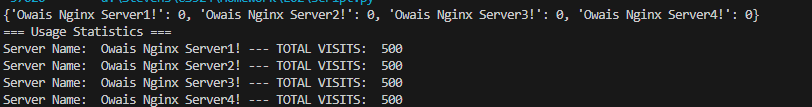
The following code was used to collect metrics, the code is written in Python:

import lxml.html  
counter = {  
 "Owais Nginx Server1!": 0,  
 "Owais Nginx Server2!": 0,  
 "Owais Nginx Server3!": 0,  
 "Owais Nginx Server4!": 0,  
}  
print(counter)  
for i in range(2000):  
 t = lxml.html.parse("http://ec2-3-14-8-219.us-east-2.compute.amazonaws.com")  
 title = t.find(".//title").text  
 counter[title]+=1   
print("=== Usage Statistics ===")  
for key in counter:   
 print("Server Name: ", key, '--- TOTAL VISITS: ', counter[key])

for changing weights in each distribution execute the following commands and edit the weight value as specified in upstream block against each server:

1. sudo vi /etc/config/nginx.conf #modify the weights in this file
2. sudo systemctl reload nginx #this command will restart the service of nginx
3. wait for a few minutes so that the changes are propagated and execute the above python code to collect metrics.
4. **CONFIG 1: (EQUAL WEIGHT DISTRIBUTION)**

**A screen shot of a computer program

Description automatically generated**

1. **CONFIG 2: (WEIGHT=1 FOR SERVER1, WEIGHT=2 FOR SERVER2, WEIGHT=3 FOR SERVER3, WEIGHT=4 FOR SERVER4)**

**A screenshot of a computer program

Description automatically generatedA screenshot of a computer screen

Description automatically generated**

1. **CONFIG 3: (WEIGHT=1 FOR SERVER 1 AND SERVER3, WEIGHT=2 FOR SERVER2 AND SERVER4)**

**A computer screen with white text

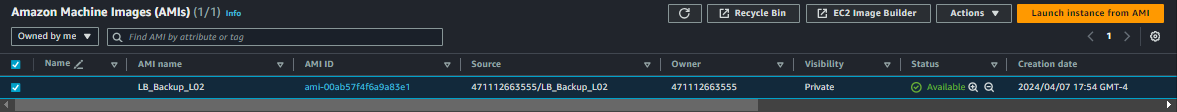
Description automatically generatedA screenshot of a computer

Description automatically generated**

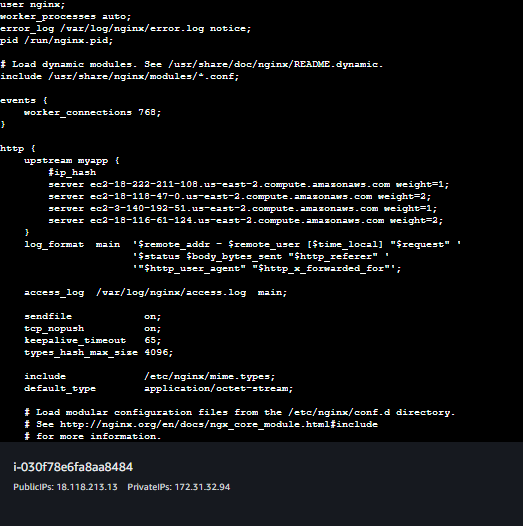
**Backup & Restore an AMI:**

1. Select the EC2 instance to backup, under the Actions Menu, Go to Image and templates, and Create Image, fill in the desired field.  
   A screenshot of a computer

   Description automatically generated
2. Now to restore instance from an AMI, under AMI sections of EC2 instance, select the AMI you want to restore from and click on Launch Instance from AMI. Provide Network & Security information i.e. Key Pair and Security Group and Click on Launch instance. All instance related information will be made available from the AMI backup



1. Verify that all the files are already present. Connect to the newly launched instance, and we will cross check if the last configuration is already present for the nginx load balancer.



All configuration was backed up, it means the restore and backup was successful.