



LOYALIST COLLEGE IN TORONTO

WINP2000 - Cloud Management

Week5 – In class lab 3

Cloud Automation

Assignment Submitted by:

Student Name: Srinidhi Sivakumar

Student ID: 500237144

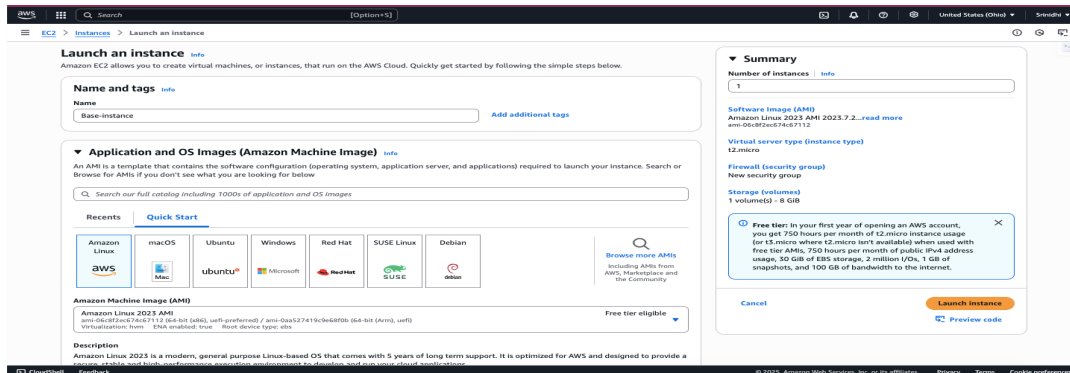
Course Code: WINP2000

Instructor Name: Sergio Loza

Phase 1: Create a Custom AMI with NGINX and Instance ID Page

Step 1: Launch a Base EC2 Instance Manually

- Go to **AWS Console > EC2 > Launch Instance**
- AMI: Amazon Linux 2
- Instance type: t2.micro
- Security Group: allow **HTTP (80)** and **SSH (22)**
- Key Pair: Select or create a new one
- Launch the instance



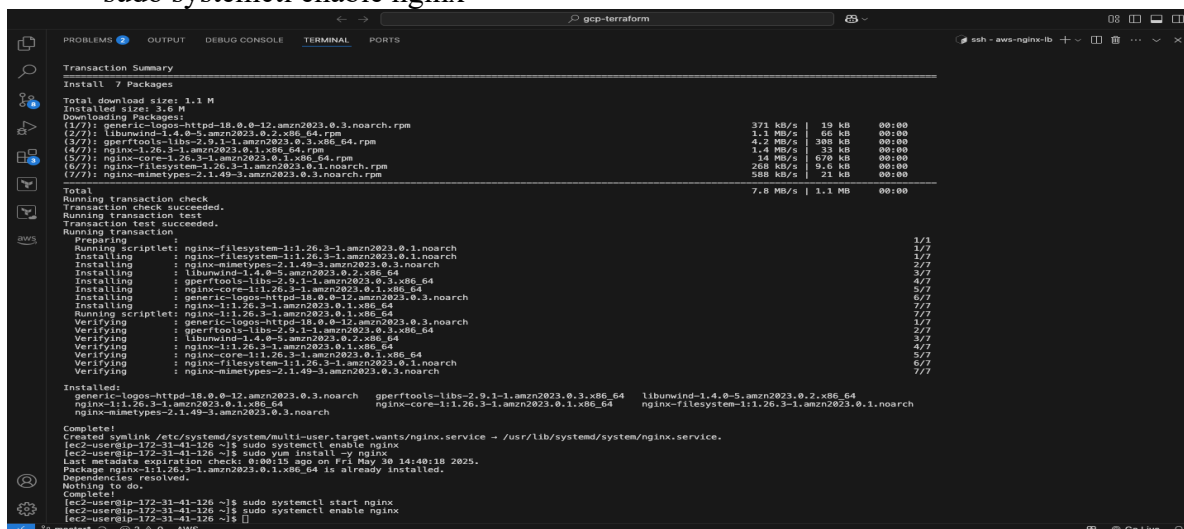
Step 2: Connect to Instance via SSH and Install and Configure NGINX

Connect to the Base Instance:

```
ssh -i nginx-lb.pem ec2-user@3.139.73.226
```

Once you entered to instance, run below commands:

```
sudo yum update -y
sudo yum install -y nginx
sudo systemctl start nginx
sudo systemctl enable nginx
```



Step 3: Add Dynamic Page with Instance ID

Once you're inside the EC2 terminal:

```
Vi gen-page.sh
```

Paste the following into the file:

```
#!/bin/bash
INSTANCE_ID=$(curl -s http://169.254.169.254/latest/meta-data/instance-id)
echo "<html>
<head>
<title>Welcome to Srinidhi's website of nginx and load balancer project</title>
</head>
<body style=\"font-family:Arial; text-align:center; margin-top:100px; background-color:#f0f8ff;\">
<h1 style=\"color:#2e8b57;\">Welcome to Srinidhi's website of nginx and load balancer project</h1>
<h2>Running on Instance: <span style=\"color:#ff4500;\">${INSTANCE_ID}</span></h2>
</body>
</html>\" | sudo tee /usr/share/nginx/html/index.html
```

Run the file:

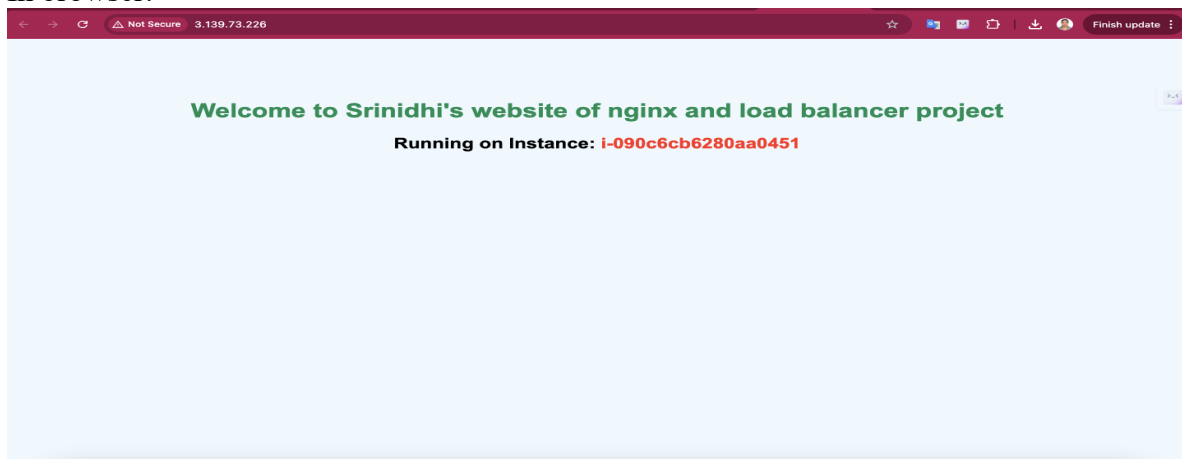
```
./gen-page.sh
```

Check it works:

In Terminal: curl localhost

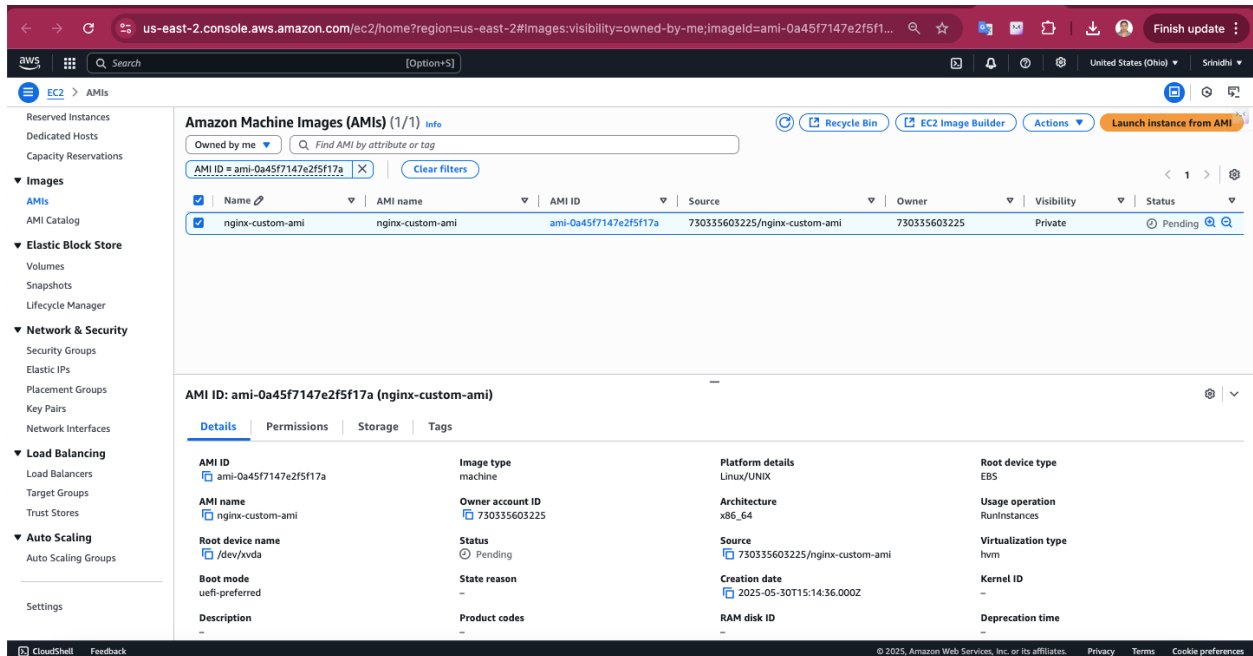
```
[ec2-user@ip-172-31-41-126 ~]$ vi gen-page.sh
[ec2-user@ip-172-31-41-126 ~]$ ./gen-page.sh
<html>
<head>
<title>Welcome to Srinidhi's website of nginx and load balancer project</title>
</head>
<body style="font-family:Arial; text-align:center; margin-top:100px; background-color:#f0f8ff;">
<h1 style="color:#2e8b57;">Welcome to Srinidhi's website of nginx and load balancer project</h1>
<h2>Running on Instance: <span style="color:#ff4500;"></span></h2>
</body>
</html>
[ec2-user@ip-172-31-41-126 ~]$ curl -s http://169.254.169.254/latest/meta-data/instance-id
[ec2-user@ip-172-31-41-126 ~]$ curl -s http://169.254.169.254/latest/meta-data/instance-id
[ec2-user@ip-172-31-41-126 ~]$ curl http://169.254.169.254/latest/meta-data/
[ec2-user@ip-172-31-41-126 ~]$ curl http://169.254.169.254/latest/meta-data/
[ec2-user@ip-172-31-41-126 ~]$ curl -s http://169.254.169.254/latest/meta-data/instance-id
i-090c6cb6280aa0451[ec2-user@ip-172-31-41-126 ~]$
[ec2-user@ip-172-31-41-126 ~]$ ./gen-page.sh
<html>
<head>
<title>Welcome to Srinidhi's website of nginx and load balancer project</title>
</head>
<body style="font-family:Arial; text-align:center; margin-top:100px; background-color:#f0f8ff;">
<h1 style="color:#2e8b57;">Welcome to Srinidhi's website of nginx and load balancer project</h1>
<h2>Running on Instance: <span style="color:#ff4500;">i-090c6cb6280aa0451</span></h2>
</body>
</html>
[ec2-user@ip-172-31-41-126 ~]$ exit
logout
```

In browser:



Step 4: Create Custom AMI

- Go to **EC2 > Instances**
- Right-click your instance → **Image > Create Image**
- Name: **nginx-custom-ami**
- Click **Create Image**
- Wait for the AMI to be "available" under **EC2 > AMIs**
- Copy the **AMI ID** : [ami-0a45f7147e2f5f17a](#)
-



The screenshot shows the AWS Management Console interface for Amazon Machine Images (AMIs). The left sidebar contains navigation links for EC2, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, Elastic Block Store, Network & Security, Load Balancing, and Auto Scaling. The main content area is titled 'Amazon Machine Images (AMIs) (1/1)' and includes a search bar and a table of AMIs. The table lists one AMI: 'nginx-custom-ami' with ID 'ami-0a45f7147e2f5f17a'. Below the table, the 'Details' tab is selected, displaying various attributes for the AMI.

AMI ID	Image type	Platform details	Root device type
ami-0a45f7147e2f5f17a	machine	Linux/UNIX	EBS

AMI name	Owner account ID	Architecture	Usage operation
nginx-custom-ami	730335603225	x86_64	RunInstances

Root device name	Status	Source	Virtualization type
/dev/xvda	Pending	730335603225/nginx-custom-ami	hvm

Boot mode	State reason	Creation date	Kernel ID
uefi-preferred	-	2025-05-30T15:14:36.000Z	-

Description	Product codes	RAM disk ID	Deprecation time
-	-	-	-

Phase 2: Terraform Project Setup

Step 1: Create Project Folder and Terraform Files

You'll need:

- main.tf
- variables.tf
- outputs.tf
- gen-page.sh

Link to check Terraform Code: <https://github.com/srinidhisivakumar/aws-nginx-lb>

Phase 3: Run Terraform

Step 1: Initialize

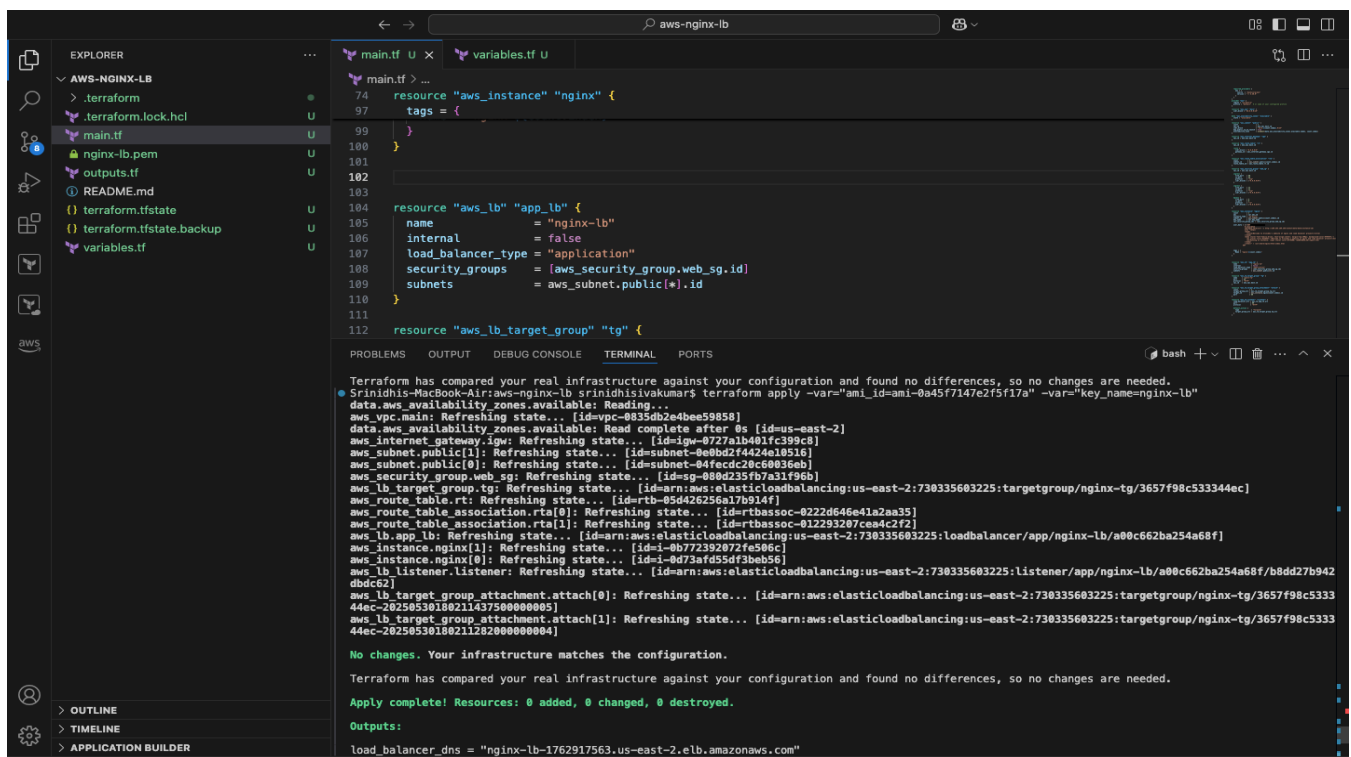
```
terraform init
```

Step 2: Plan

```
terraform plan -var="ami_id= ami-0a45f7147e2f5f17a " -var="key_name=nginx-lb.pem"
```

Step 3: Apply

```
terraform apply -var="ami_id= ami-0a45f7147e2f5f17a " -var="key_name=nginx-lb.pem"
```



```
main.tf U X variables.tf U
main.tf > ...
74 resource "aws_instance" "nginx" {
97   tags = {
99   }
100 }
101
102 resource "aws_lb" "app_lb" {
103   name           = "nginx-lb"
104   internal       = false
105   load_balancer_type = "application"
106   security_groups = [aws_security_group.web_sg.id]
107   subnets       = aws_subnet.public[*].id
108 }
109
110 resource "aws_lb_target_group" "tg" {
111 }
112

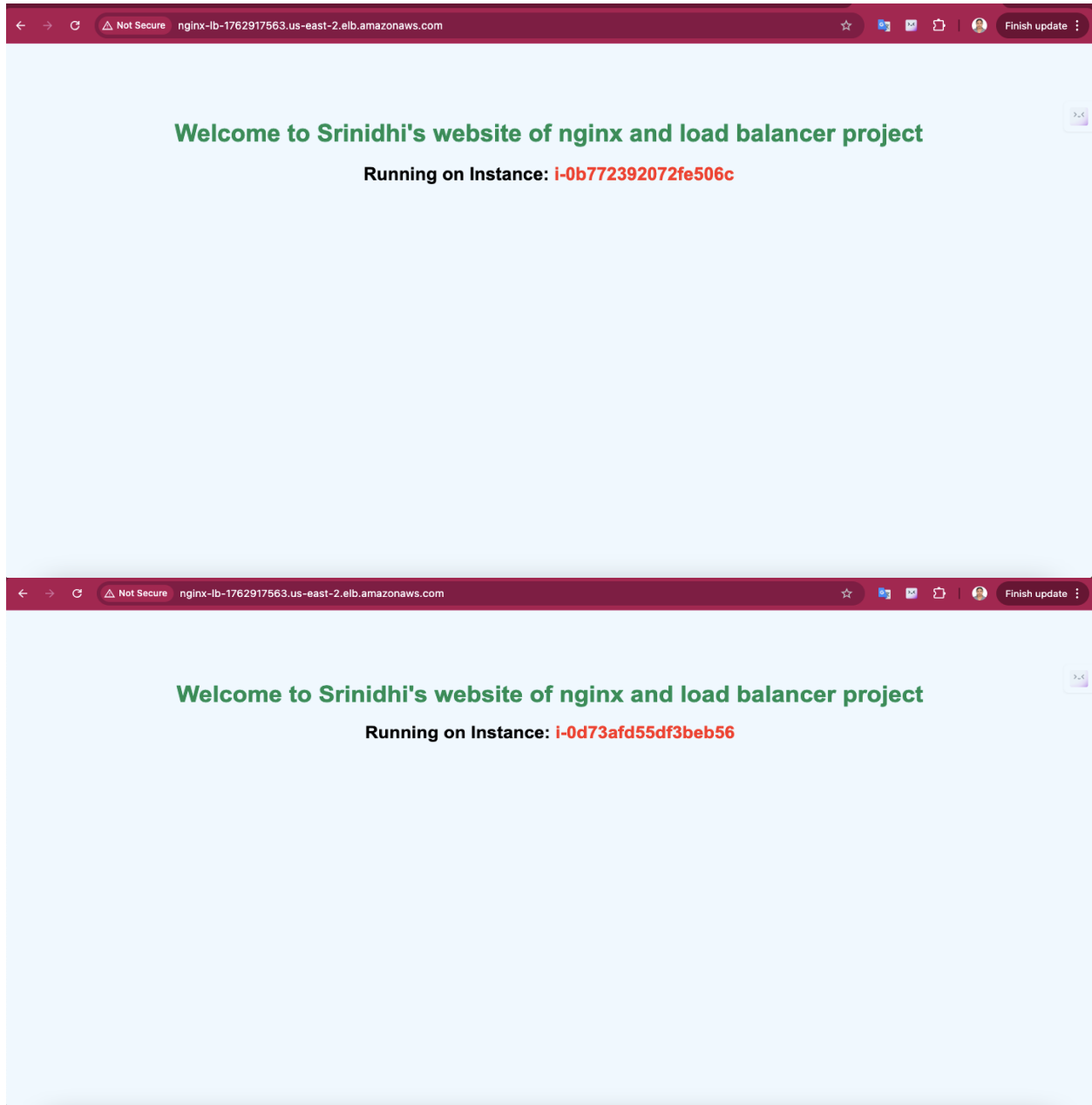
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.
Srinidhis-MacBook-Air:aws-nginx-lb srinidhisivakumar$ terraform apply -var="ami_id=ami-0a45f7147e2f5f17a" -var="key_name=nginx-lb"
data.aws_availability_zones.available: Reading...
aws_vpc.main: Refreshing state... [id=vpc-0835db2e4bee59858]
data.aws_availability_zones.available: Read complete after 0s [id=us-east-2]
aws_internet_gateway.igw: Refreshing state... [id=igw-0727a1b401fc399c8]
aws_subnet.public[1]: Refreshing state... [id=subnet-0e0bd2f442de0516]
aws_subnet.public[0]: Refreshing state... [id=subnet-04fedc28ce0836eb]
aws_security_group.web_sg: Refreshing state... [id=sg-080d235fb7a31f96b]
aws_lb_target_group.tg: Refreshing state... [id=arn:aws:elasticloadbalancing:us-east-2:730335603225:targetgroup/nginx-tg/3657f98c533344ec]
aws_route_table.rt: Refreshing state... [id=rtb-05d426256a17b914f]
aws_route_table_association.rta[0]: Refreshing state... [id=rtbassoc-0222d646e41a2aa35]
aws_route_table_association.rta[1]: Refreshing state... [id=rtbassoc-012293207cea4c2f2]
aws_lb.app_lb: Refreshing state... [id=arn:aws:elasticloadbalancing:us-east-2:730335603225:loadbalancer/app/nginx-lb/a00c662ba254a68f]
aws_instance.nginx[1]: Refreshing state... [id=i-0b77239207fe506c]
aws_instance.nginx[0]: Refreshing state... [id=i-0d73af65d5df2be856]
aws_lb_listener.listener: Refreshing state... [id=arn:aws:elasticloadbalancing:us-east-2:730335603225:listener/app/nginx-lb/a00c662ba254a68f/b8dd27b942dbdc62]
aws_lb_target_group_attachment.attach[0]: Refreshing state... [id=arn:aws:elasticloadbalancing:us-east-2:730335603225:targetgroup/nginx-tg/3657f98c533344ec-20250530180211437500000005]
aws_lb_target_group_attachment.attach[1]: Refreshing state... [id=arn:aws:elasticloadbalancing:us-east-2:730335603225:targetgroup/nginx-tg/3657f98c533344ec-20250530180211437500000004]

No changes. Your infrastructure matches the configuration.
Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.
Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
Outputs:
load_balancer_dns = "nginx-lb-1762917563.us-east-2.elb.amazonaws.com"
```

Phase 4: Test & Capture Outputs (Uploaded video in GitHub)

1. Copy the DNS from Terraform output.
2. Paste it into your browser.
3. Refresh the page multiple times to see both instance IDs appear

LB DNS: <http://nginx-lb-1762917563.us-east-2.elb.amazonaws.com/>



Video Link: <https://github.com/srinidhisivakumar/aws-nginx-lb/tree/master/video-output>

VPC console:

VPC dashboard

EC2 Global View

Filter by VPC:

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only Internet gateways

DHCP option sets

Elastic IPs

Managed prefix lists

NAT gateways

Peering connections

Route servers

Security

Network ACLs

Security groups

PrivateLink and Lattice

Getting started

Endpoints

Endpoint services

Service networks

Lattice services

VPC ID

vpc-0835db2e4bee59858

DNS resolution

Enabled

Main network ACL

acl-0a91dfe3a3260960c

IPv6 CIDR

-

State

Available

Tenancy

default

Default VPC

No

Network Address Usage metrics

Disabled

Block Public Access

Off

DHCP option set

dopt-0a6876c7367beaa1e

IPv4 CIDR

10.0.0.0/16

Route 53 Resolver DNS Firewall rule groups

-

DNS hostnames

Disabled

Main route table

rtb-028f3899d7b68c65a

IPv6 pool

-

Owner ID

730335603225

Resource map

CIDRs

Flow logs

Tags

Integrations

Resource map

VPC

Subnets (2)

Route tables (2)

Network connections (1)

Security Group inbound rules:

Type	Protocol	Port	Source
SSH	TCP	22	My IP
HTTP	TCP	80	0.0.0.0/0

EC2

Dashboard

EC2 Global View

Events

Instances

Instance Types

Launch Templates

Spot Requests

Savings Plans

Reserved Instances

Dedicated Hosts

Capacity Reservations

Images

AMIs

AMI Catalog

Elastic Block Store

Volumes

Snapshots

Lifecycle Manager

Network & Security

Security Groups

Elastic IPs

Placement Groups

Key Pairs

sg-080d235fb7a31f96b - terraform-20250530180146113500000001

Details

Security group name

terraform-20250530180146113500000001

Security group ID

sg-080d235fb7a31f96b

Description

Managed by Terraform

VPC ID

vpc-0835db2e4bee59858

Owner

730335603225

Inbound rules count

2 Permission entries

Outbound rules count

1 Permission entry

Inbound rules

Outbound rules

Sharing - new

VPC associations - new

Tags

Inbound rules (2)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	Description
-	sgr-015068405f1160c20	IPv4	HTTP	TCP	80	0.0.0.0/0	-
-	sgr-0dd21f8a13f974309	IPv4	SSH	TCP	22	0.0.0.0/0	-