Cloud Computing Lab

Name: Arooj Saleem

Roll # 2023-BSE-013

Submitted To: Engr. Muhammad Shoaib

Lab Title: Terraform Provisioners, Modules & Nginx Reverse Proxy/Load Balancer

Lab # 12

Task 0 Lab Setup (Codespace & GH CLI)

All actions below should be executed inside the Codespace shell.

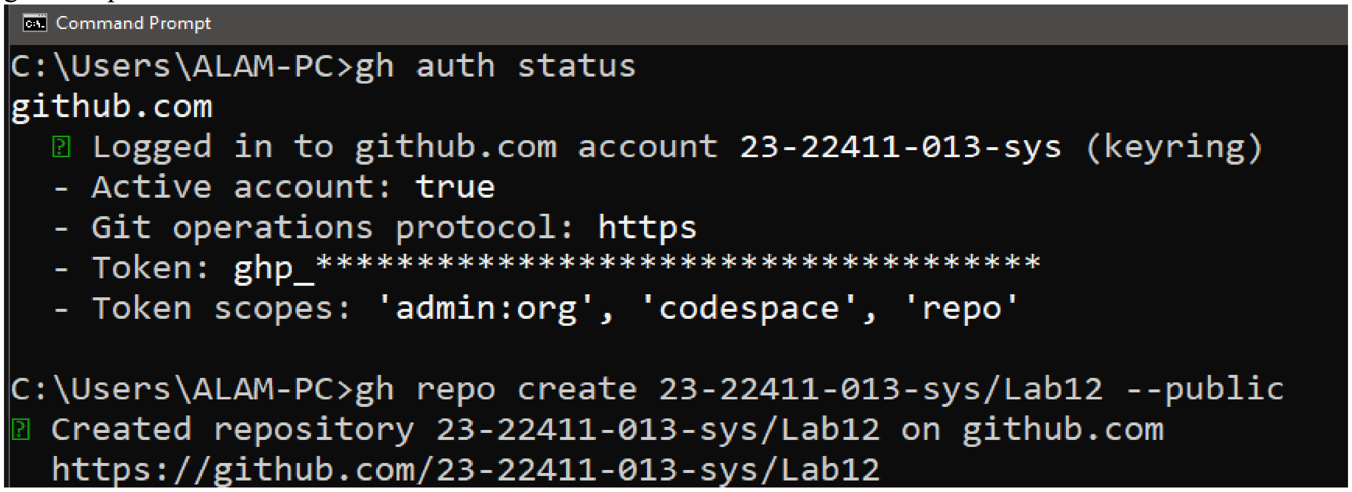
Create Codespace & connect:

create or open codespace via GH CLI (example)

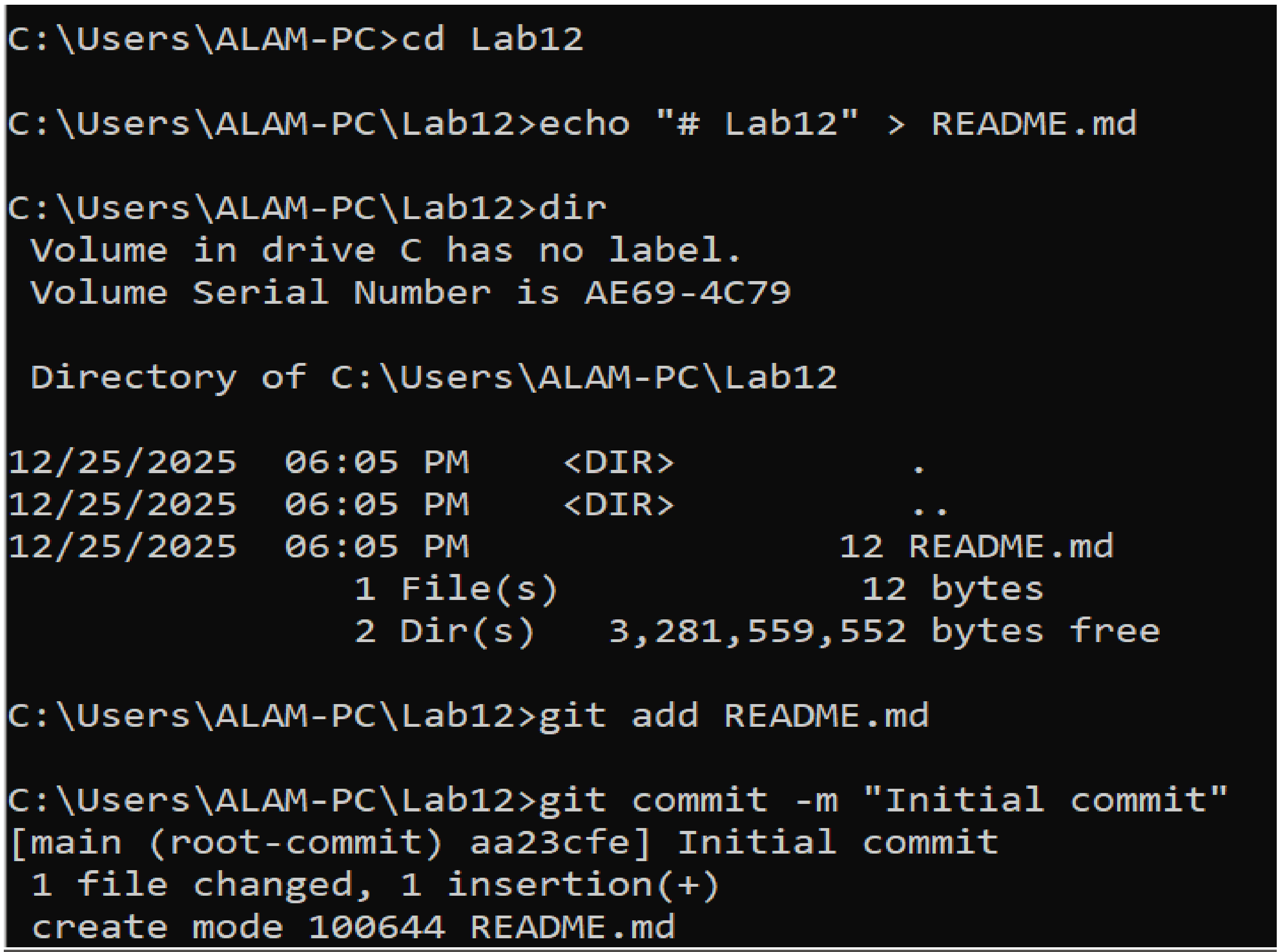
gh repo create CC_<YourName>_<YourRollNumber>/Lab12 –public

gh codespace create --repo <user_name>/Lab12

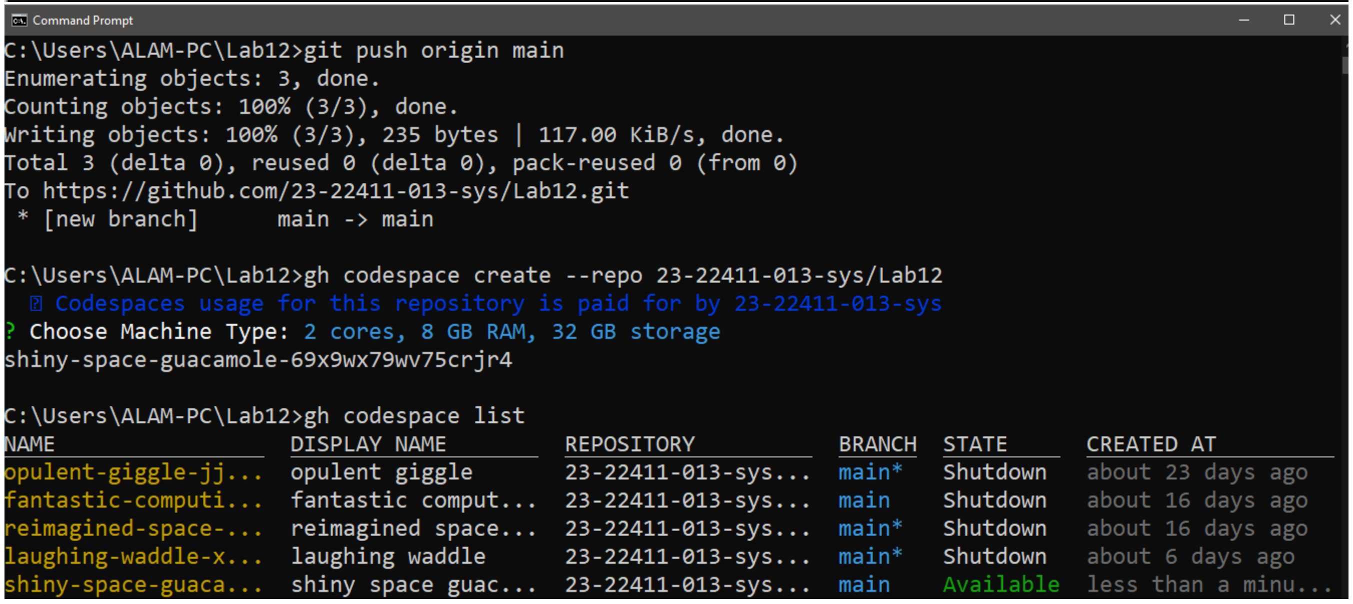
gh codespace list



```
C:\Users\ALAM-PC>gh auth status
github.com
  ✓ Logged in to github.com account 23-22411-013-sys (keyring)
  - Active account: true
  - Git operations protocol: https
  - Token: ghp_************************************
  - Token scopes: 'admin:org', 'codespace', 'repo'

C:\Users\ALAM-PC>gh repo create 23-22411-013-sys/Lab12 --public
✓ Created repository 23-22411-013-sys/Lab12 on github.com
  https://github.com/23-22411-013-sys/Lab12
```



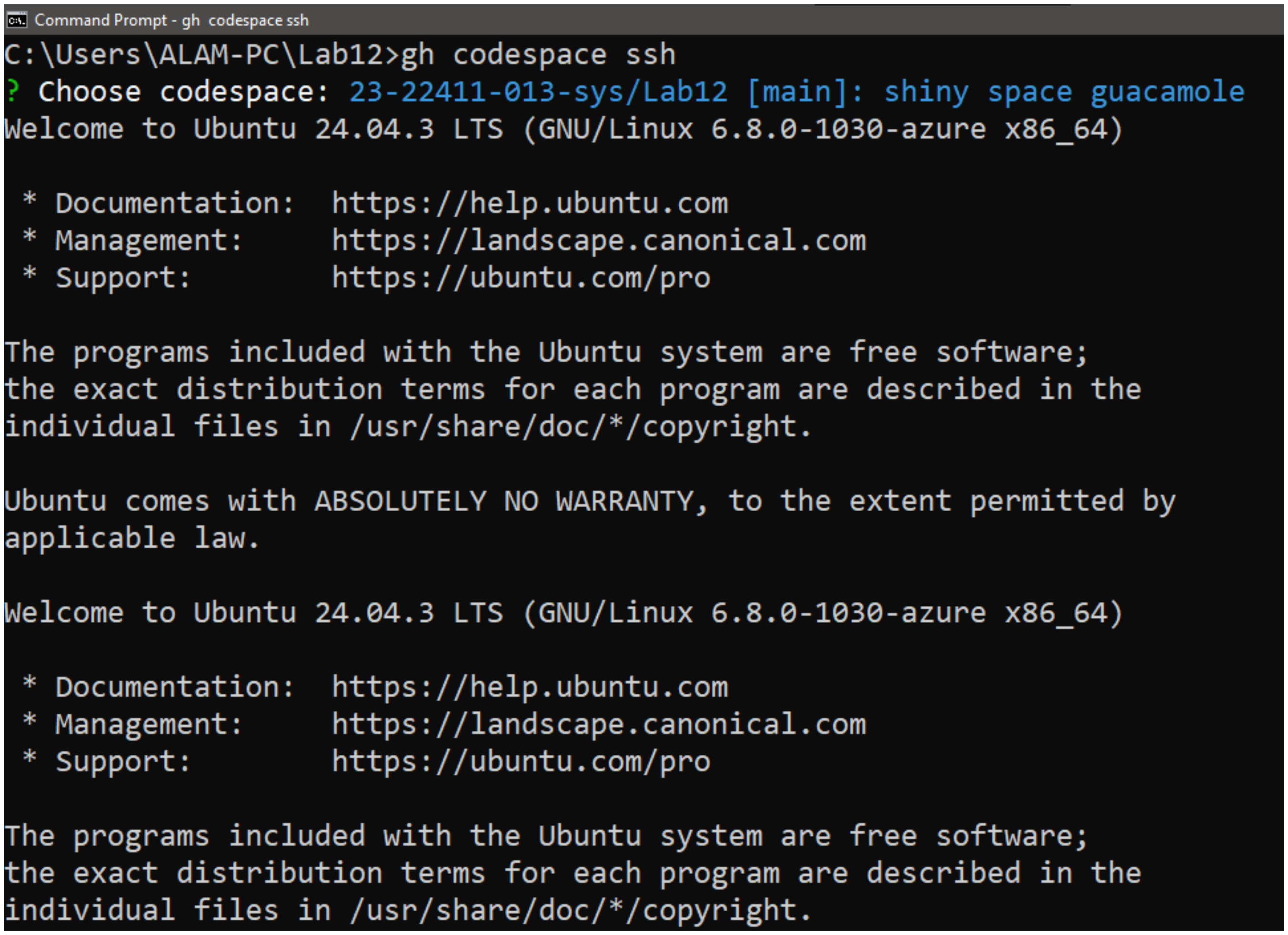
```
C:\Users\ALAM-PC>cd Lab12

C:\Users\ALAM-PC\Lab12>echo "# Lab12" > README.md

C:\Users\ALAM-PC\Lab12>dir
 Volume in drive C has no label.
 Volume Serial Number is AE69-4C79

 Directory of C:\Users\ALAM-PC\Lab12

12/25/2025  06:05 PM    <DIR>          .
12/25/2025  06:05 PM    <DIR>          ..
12/25/2025  06:05 PM                12 README.md
               1 File(s)             12 bytes
               2 Dir(s)   3,281,559,552 bytes free

C:\Users\ALAM-PC\Lab12>git add README.md

C:\Users\ALAM-PC\Lab12>git commit -m "Initial commit"
[main (root-commit) aa23cfe] Initial commit
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
```



Command Prompt

```
C:\Users\ALAM-PC\Lab12>git push origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 235 bytes | 117.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/23-22411-013-sys/Lab12.git
 * [new branch]      main -> main

C:\Users\ALAM-PC\Lab12>gh codespace create --repo 23-22411-013-sys/Lab12
  ▯ Codespaces usage for this repository is paid for by 23-22411-013-sys
? Choose Machine Type: 2 cores, 8 GB RAM, 32 GB storage
shiny-space-guacamole-69x9wx79wv75crjr4

C:\Users\ALAM-PC\Lab12>gh codespace list
NAME                  DISPLAY NAME          REPOSITORY            BRANCH   STATE      CREATED AT
opulent-giggle-jj...  opulent giggle        23-22411-013-sys...   main*    Shutdown   about 23 days ago
fantastic-computi...  fantastic comput...   23-22411-013-sys...   main     Shutdown   about 16 days ago
reimagined-space-...  reimagined space...   23-22411-013-sys...   main*    Shutdown   about 16 days ago
laughing-waddle-x...  laughing waddle       23-22411-013-sys...   main*    Shutdown   about 6 days ago
shiny-space-guaca...  shiny space guac...   23-22411-013-sys...   main     Available  less than a minu...
```

gh codespace ssh -c <your_codespace_name>



```
C:\Users\ALAM-PC\Lab12>gh codespace ssh
? Choose codespace: 23-22411-013-sys/Lab12 [main]: shiny space guacamole
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-1030-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-1030-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
```

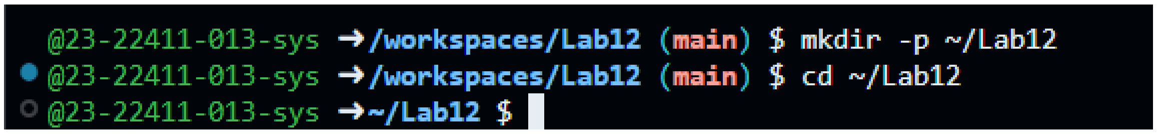
Task 1 — Organize Terraform code into separate files

In this task, you will split a monolithic Terraform configuration into separate, well-organized files following best practices.

1. Create the initial project structure:

mkdir -p ~/Lab12

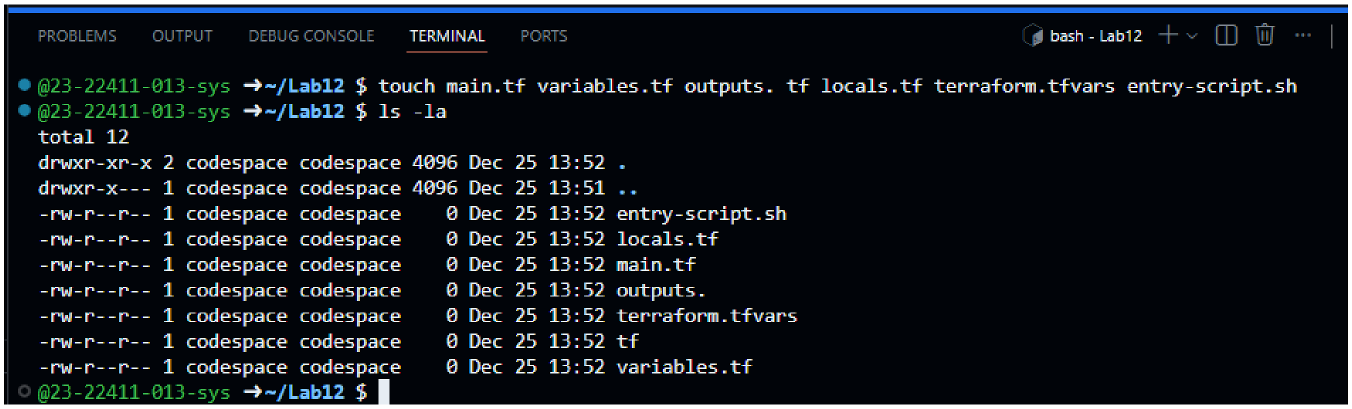
cd ~/Lab12



```
@23-22411-013-sys →/workspaces/Lab12 (main) $ mkdir -p ~/Lab12
@23-22411-013-sys →/workspaces/Lab12 (main) $ cd ~/Lab12
@23-22411-013-sys →~/Lab12 $
```

2. Create all required files:

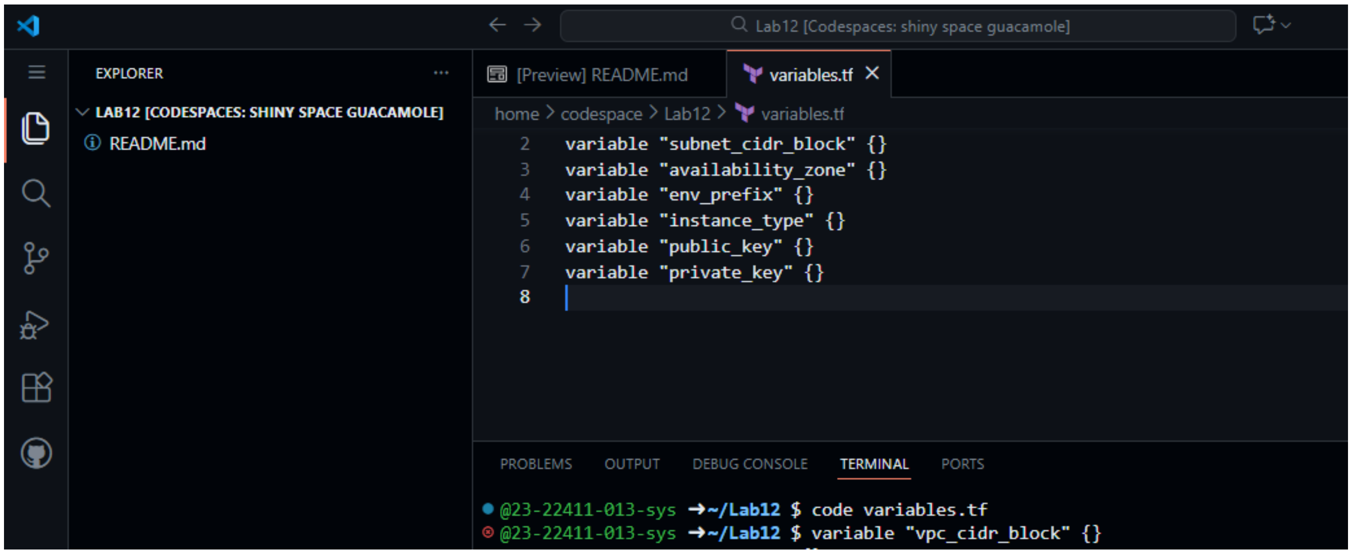
touch main.tf variables.tf outputs. tf locals.tf terraform.tfvars entry-script.sh



```
@23-22411-013-sys ➜~/Lab12 $ touch main.tf variables.tf outputs. tf locals.tf terraform.tfvars entry-script.sh
@23-22411-013-sys ➜~/Lab12 $ ls -la
total 12
drwxr-xr-x 2 codespace codespace 4096 Dec 25 13:52 .
drwxr-x--- 1 codespace codespace 4096 Dec 25 13:51 ..
-rw-r--r-- 1 codespace codespace    0 Dec 25 13:52 entry-script.sh
-rw-r--r-- 1 codespace codespace    0 Dec 25 13:52 locals.tf
-rw-r--r-- 1 codespace codespace    0 Dec 25 13:52 main.tf
-rw-r--r-- 1 codespace codespace    0 Dec 25 13:52 outputs.
-rw-r--r-- 1 codespace codespace    0 Dec 25 13:52 terraform.tfvars
-rw-r--r-- 1 codespace codespace    0 Dec 25 13:52 tf
-rw-r--r-- 1 codespace codespace    0 Dec 25 13:52 variables.tf
@23-22411-013-sys ➜~/Lab12 $
```

3. Create variables.tf with the following content:

```
variable "vpc_cidr_block" {}

variable "subnet_cidr_block" {}

variable "availability_zone" {}

variable "env_prefix" {}

variable "instance_type" {}

variable "public_key" {}

variable "private_key" {}
```



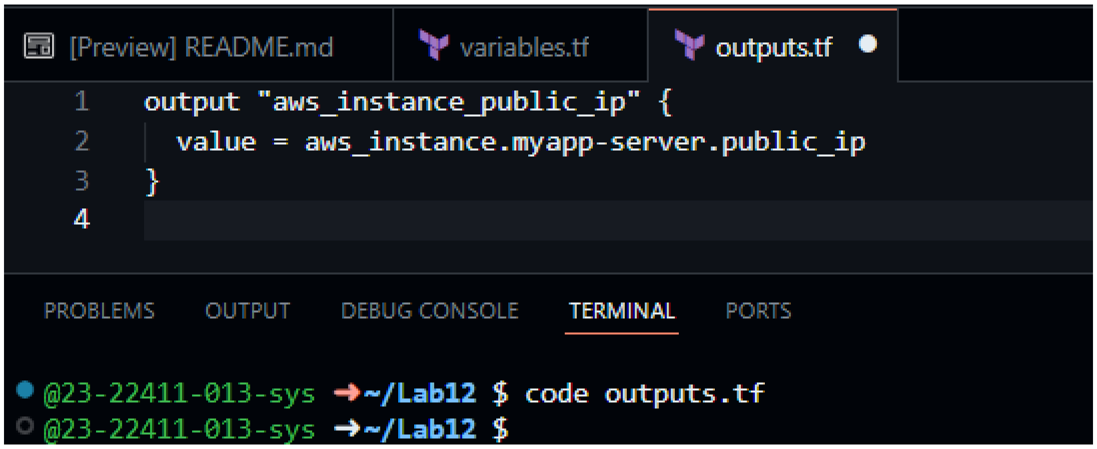
Lab12 [Codespaces: shiny space guacamole]
EXPLORER
LAB12 [CODESPACES: SHINY SPACE GUACAMOLE]
README.md
[Preview] README.md
variables.tf
home > codespace > Lab12 > variables.tf
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS

```
variable "subnet_cidr_block" {}
variable "availability_zone" {}
variable "env_prefix" {}
variable "instance_type" {}
variable "public_key" {}
variable "private_key" {}
```

```
@23-22411-013-sys ➜~/Lab12 $ code variables.tf
@23-22411-013-sys ➜~/Lab12 $ variable "vpc_cidr_block" {}
```

4. Create outputs.tf with the following content:

```
output "aws_instance_public_ip" {
  value = aws_instance.myapp-server.public_ip
}
```

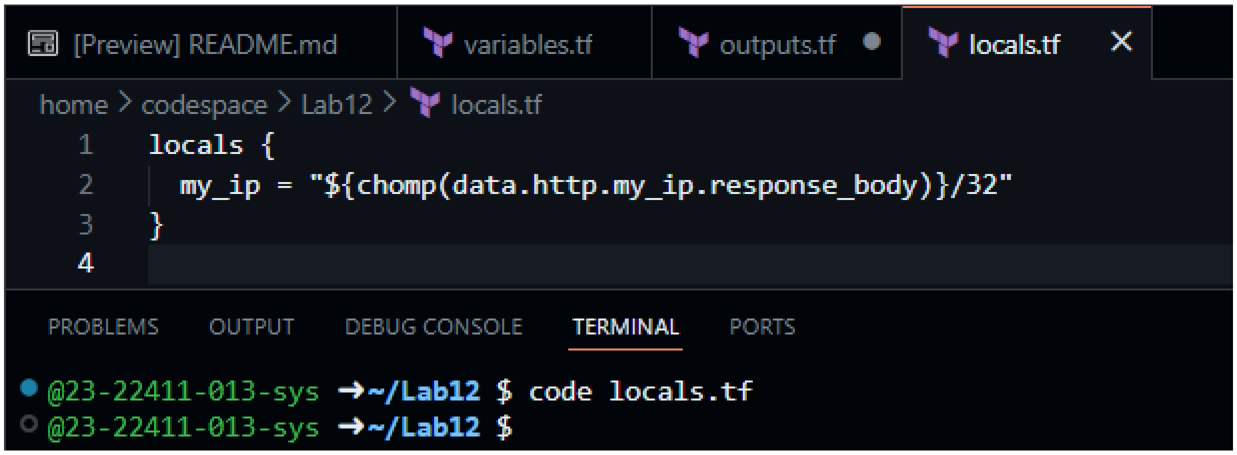


```
output "aws_instance_public_ip" {
  value = aws_instance.myapp-server.public_ip
}

```

```
@23-22411-013-sys ➜~/Lab12 $ code outputs.tf
@23-22411-013-sys ➜~/Lab12 $
```

5. Create locals.tf with the following content:

```
locals {
  my_ip = "${chomp(data.http.my_ip.response_body)}/32"
}
```



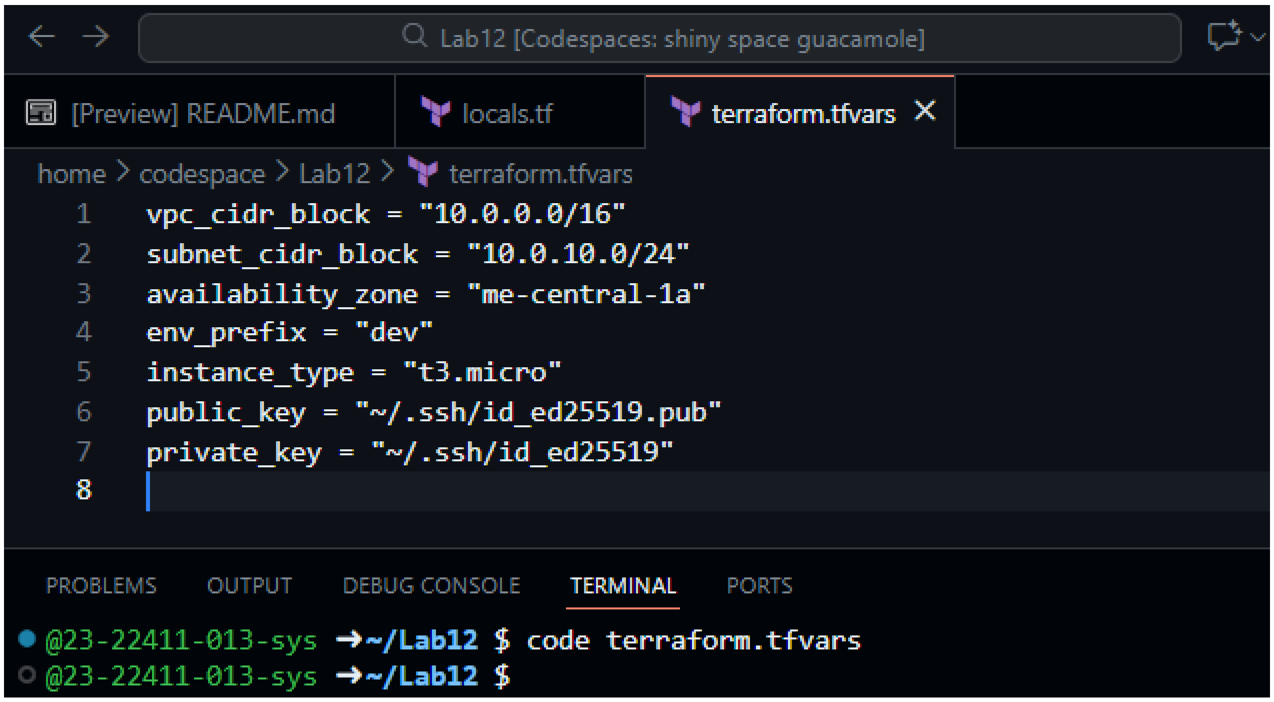
```
locals {
  my_ip = "${chomp(data.http.my_ip.response_body)}/32"
}

```

```
@23-22411-013-sys ➜~/Lab12 $ code locals.tf
@23-22411-013-sys ➜~/Lab12 $
```

6. Create terraform.tfvars with the following content:

```
vpc_cidr_block = "10.0.0.0/16"
subnet_cidr_block = "10.0.10.0/24"
availability_zone = "me-central-1a"
env_prefix = "dev"
instance_type = "t3.micro"
public_key = "~/.ssh/id_ed25519.pub"
private_key = "~/.ssh/id_ed25519"
```

- **Save screenshot as:** task1_terraform_tfvars. png — content of terraform.tfvars file.


Lab12 [Codespaces: shiny space guacamole]
[Preview] README.md
locals.tf
terraform.tfvars
home > codespace > Lab12 > terraform.tfvars
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS

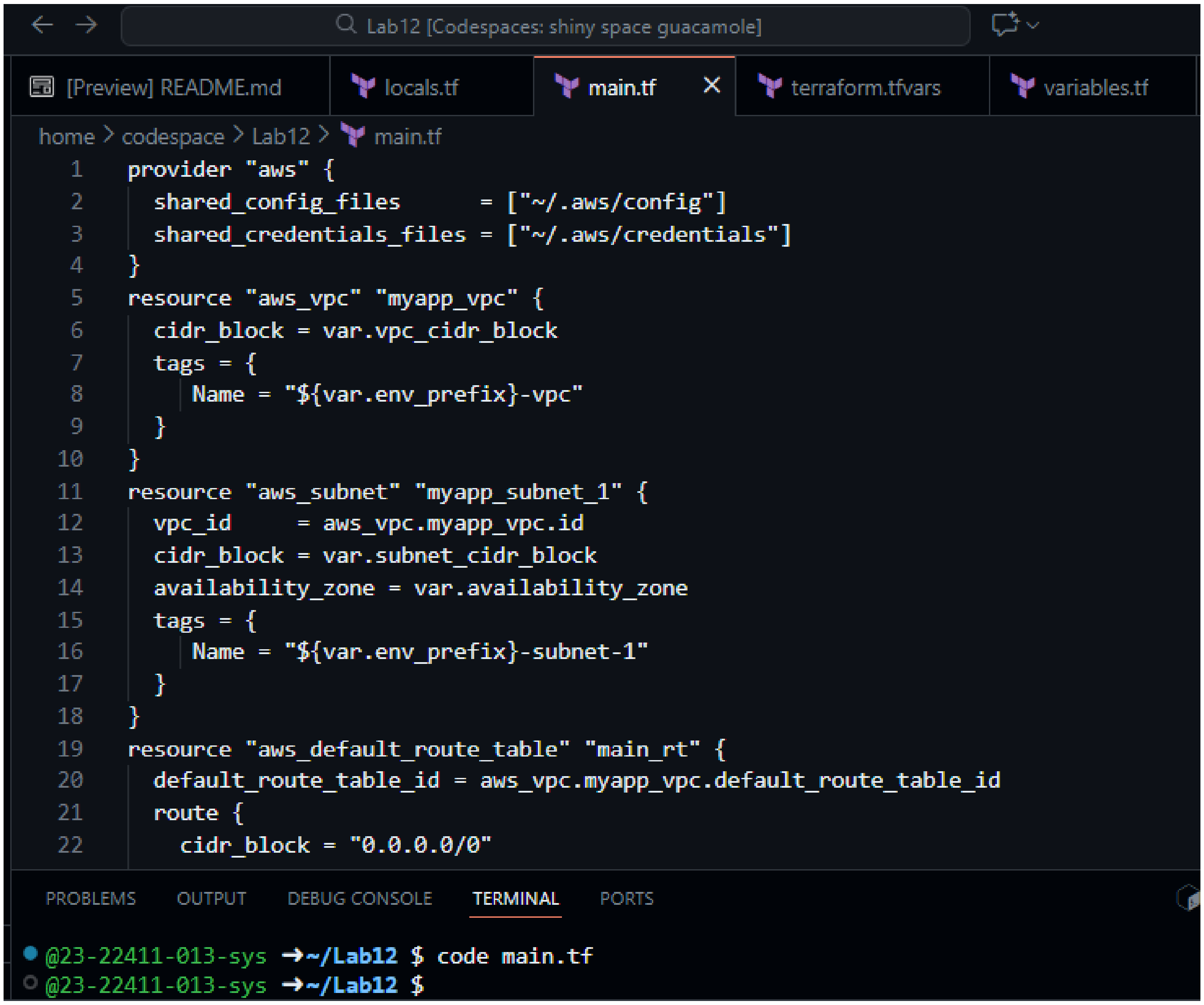
```
vpc_cidr_block = "10.0.0.0/16"
subnet_cidr_block = "10.0.10.0/24"
availability_zone = "me-central-1a"
env_prefix = "dev"
instance_type = "t3.micro"
public_key = "~/.ssh/id_ed25519.pub"
private_key = "~/.ssh/id_ed25519"
```

```
@23-22411-013-sys ➜~/Lab12 $ code terraform.tfvars
@23-22411-013-sys ➜~/Lab12 $
```

7. Create main.tf with the following content:

```
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}
resource "aws_vpc" "myapp_vpc" {
  cidr_block = var.vpc_cidr_block
  tags = {
    Name = "${var.env_prefix}-vpc"
  }
}
resource "aws_subnet" "myapp_subnet_1" {
  vpc_id     = aws_vpc.myapp_vpc.id
  cidr_block = var.subnet_cidr_block
  availability_zone = var.availability_zone
  tags = {
    Name = "${var.env_prefix}-subnet-1"
  }
}
```

```
resource "aws_default_route_table" "main_rt" {
  default_route_table_id = aws_vpc.myapp_vpc.default_route_table_id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.myapp_igw.id
  }
    tags = {
     Name = "${var.env_prefix}-rt"
  }
}
resource "aws_internet_gateway" "myapp_igw" {
  vpc_id = aws_vpc.myapp_vpc.id
    tags = {
     Name = "${var.env_prefix}-igw"
  }
}
resource "aws_default_security_group" "default_sg" {
  vpc_id      = aws_vpc.myapp_vpc.id
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = [local.my_ip]
  }
  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
    prefix_list_ids = []
  }
  tags = {
     Name = "${var.env_prefix}-default-sg"
  }
}
resource "aws_key_pair" "ssh-key" {
  key_name = "serverkey"
  public_key = file(var.public_key)
}
resource "aws_instance" "myapp-server" {
  ami           = "ami-05524d6658fcf35b6" # Amazon Linux 2023 Kernel 6.1 AMI
  instance_type = var.instance_type
  subnet_id     = aws_subnet.myapp_subnet_1.id
  security_groups = [aws_default_security_group.default_sg. id]
  availability_zone = var.availability_zone
  associate_public_ip_address = true
  key_name = aws_key_pair.ssh-key. key_name
  user_data = file("./entry-script.sh")
  tags = {
     Name = "${var.env_prefix}-ec2-instance"
  }
}
data "http" "my_ip" {
  url = "https://icanhazip.com"
}
```

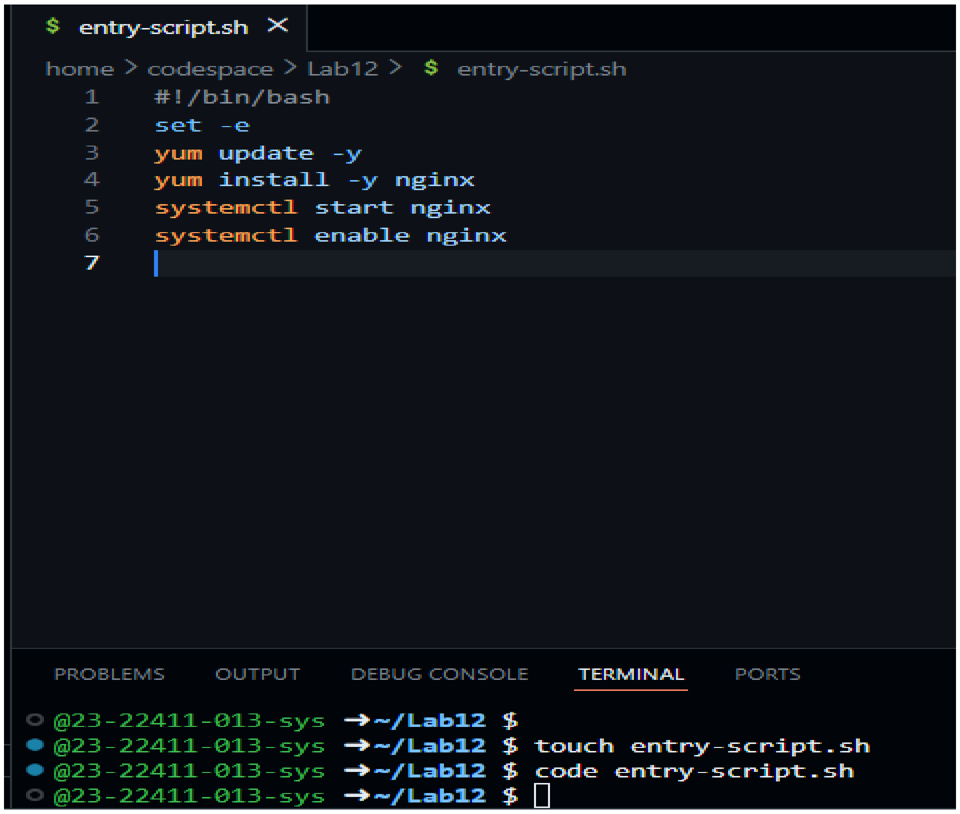


```
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}
resource "aws_vpc" "myapp_vpc" {
  cidr_block = var.vpc_cidr_block
  tags = {
     Name = "${var.env_prefix}-vpc"
  }
}
resource "aws_subnet" "myapp_subnet_1" {
  vpc_id     = aws_vpc.myapp_vpc.id
  cidr_block = var.subnet_cidr_block
  availability_zone = var.availability_zone
  tags = {
     Name = "${var.env_prefix}-subnet-1"
  }
}
resource "aws_default_route_table" "main_rt" {
  default_route_table_id = aws_vpc.myapp_vpc.default_route_table_id
  route {
    cidr_block = "0.0.0.0/0"
```

```
@23-22411-013-sys →~/Lab12 $ code main.tf
@23-22411-013-sys →~/Lab12 $
```

8. Create entry-script.sh with the following content:

```
#!/bin/bash
set -e
yum update -y
yum install -y nginx
systemctl start nginx
systemctl enable nginx
```

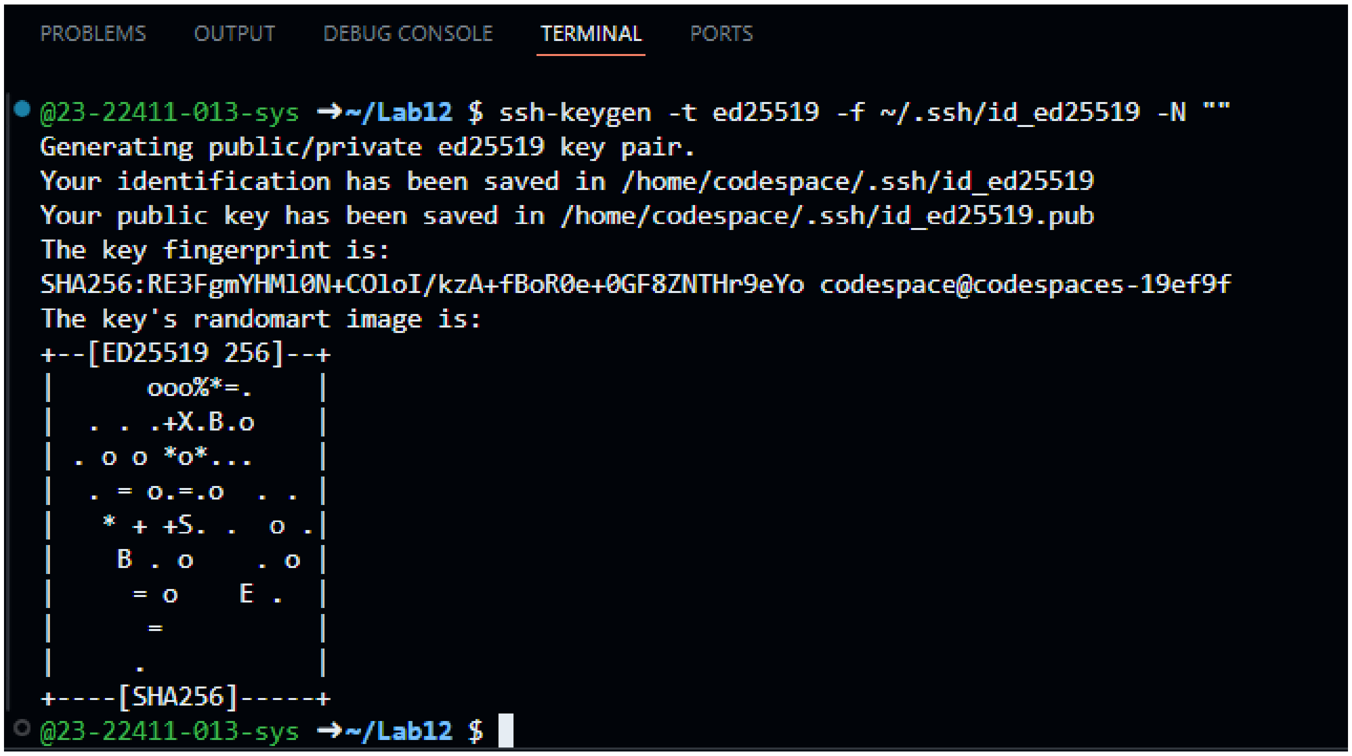


```bash
#!/bin/bash
set -e
yum update -y
yum install -y nginx
systemctl start nginx
systemctl enable nginx

```

```
@23-22411-013-sys →~/Lab12 $
@23-22411-013-sys →~/Lab12 $ touch entry-script.sh
@23-22411-013-sys →~/Lab12 $ code entry-script.sh
@23-22411-013-sys →~/Lab12 $
```

9. Generate SSH key pair if not already exists:

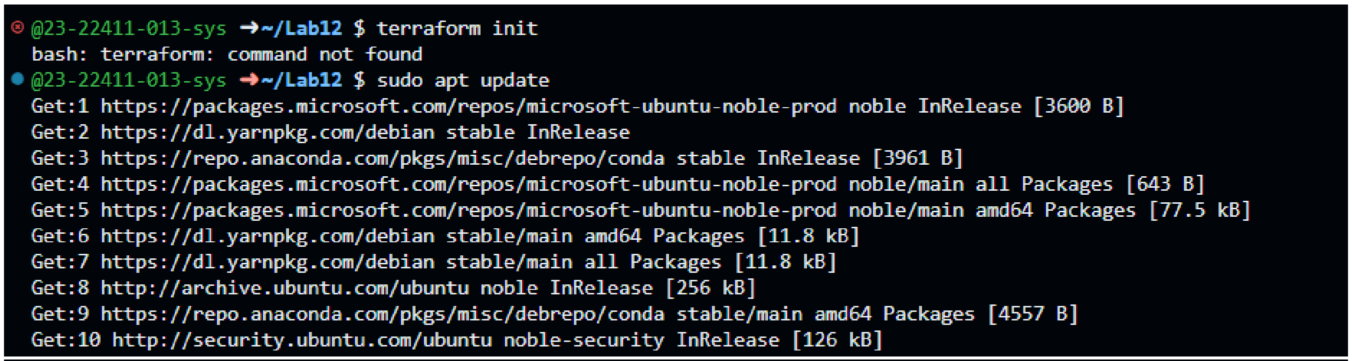
ssh-keygen -t ed25519 -f ~/.ssh/id_ed25519 -N ""



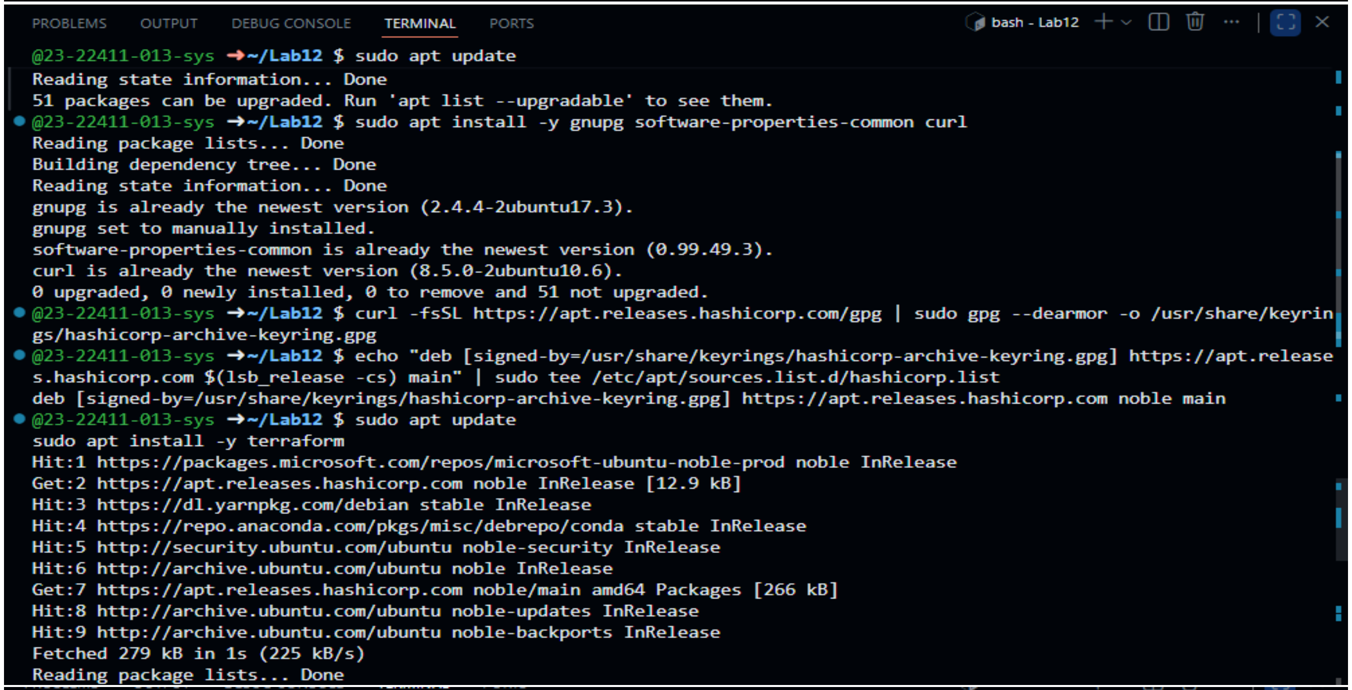
```
@23-22411-013-sys →~/Lab12 $ ssh-keygen -t ed25519 -f ~/.ssh/id_ed25519 -N ""
Generating public/private ed25519 key pair.
Your identification has been saved in /home/codespace/.ssh/id_ed25519
Your public key has been saved in /home/codespace/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:RE3FgmYHM10N+COloI/kzA+fBoR0e+0GF8ZNTHr9eYo codespace@codespaces-19ef9f
The key's randomart image is:
+--[ED25519 256]--+
|      ooo%*=.    |
|  . . .+X.B.o    |
| . o o *o*...    |
|  . = o.=.o  . . |
|   * + +S. .  o .|
|    B . o    . o |
|     = o    E .  |
|      =          |
|     .           |
+----[SHA256]-----+
@23-22411-013-sys →~/Lab12 $
```

10. Initialize Terraform:

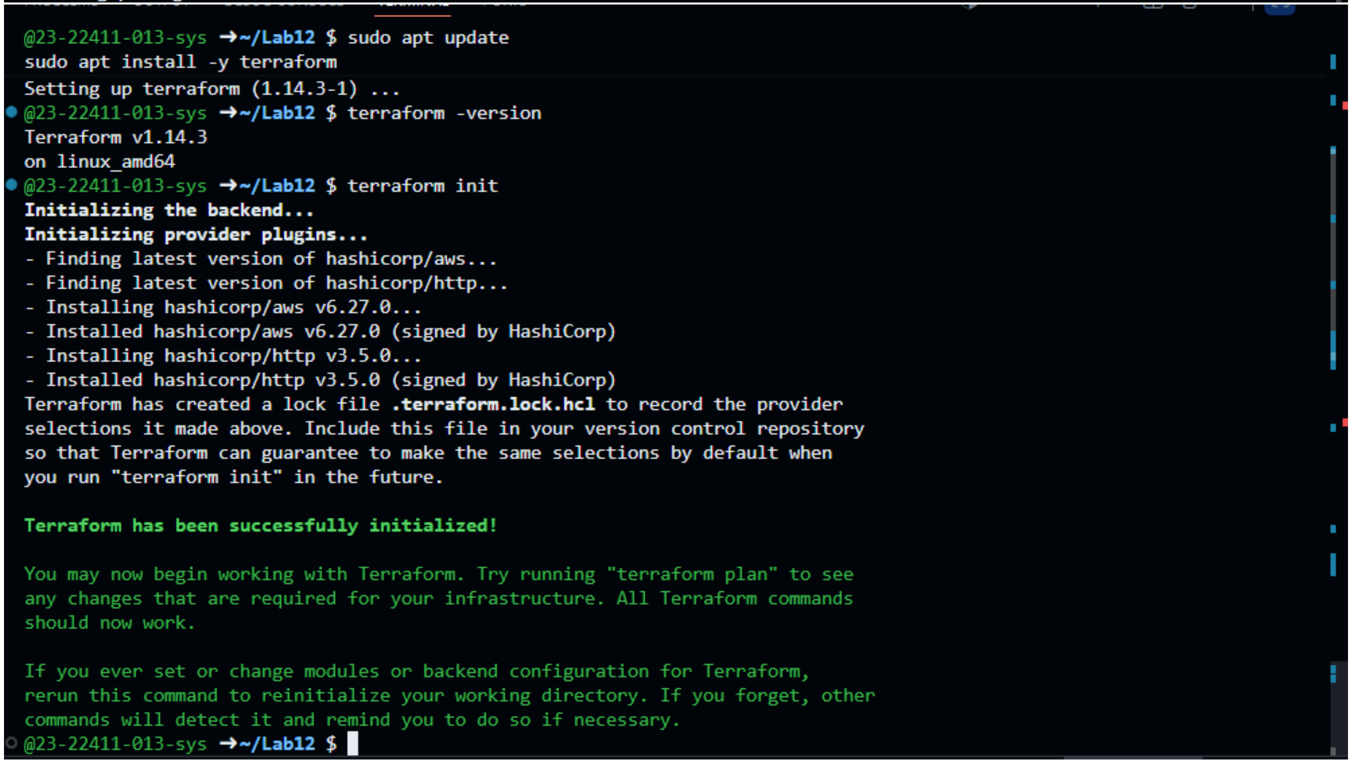
terraform init



```
@23-22411-013-sys ➜~/Lab12 $ terraform init
bash: terraform: command not found
@23-22411-013-sys ➜~/Lab12 $ sudo apt update
Get:1 https://packages.microsoft.com/repos/microsoft-ubuntu-noble-prod noble InRelease [3600 B]
Get:2 https://dl.yarnpkg.com/debian stable InRelease
Get:3 https://repo.anaconda.com/pkgs/misc/debrepo/conda stable InRelease [3961 B]
Get:4 https://packages.microsoft.com/repos/microsoft-ubuntu-noble-prod noble/main all Packages [643 B]
Get:5 https://packages.microsoft.com/repos/microsoft-ubuntu-noble-prod noble/main amd64 Packages [77.5 kB]
Get:6 https://dl.yarnpkg.com/debian stable/main amd64 Packages [11.8 kB]
Get:7 https://dl.yarnpkg.com/debian stable/main all Packages [11.8 kB]
Get:8 http://archive.ubuntu.com/ubuntu noble InRelease [256 kB]
Get:9 https://repo.anaconda.com/pkgs/misc/debrepo/conda stable/main amd64 Packages [4557 B]
Get:10 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
```



```
@23-22411-013-sys ➜~/Lab12 $ sudo apt update
Reading state information... Done
51 packages can be upgraded. Run 'apt list --upgradable' to see them.
@23-22411-013-sys ➜~/Lab12 $ sudo apt install -y gnupg software-properties-common curl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
gnupg is already the newest version (2.4.4-2ubuntu17.3).
gnupg set to manually installed.
software-properties-common is already the newest version (0.99.49.3).
curl is already the newest version (8.5.0-2ubuntu10.6).
0 upgraded, 0 newly installed, 0 to remove and 51 not upgraded.
@23-22411-013-sys ➜~/Lab12 $ curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrin
gs/hashicorp-archive-keyring.gpg
@23-22411-013-sys ➜~/Lab12 $ echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.release
s.hashicorp.com $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com noble main
@23-22411-013-sys ➜~/Lab12 $ sudo apt update
sudo apt install -y terraform
Hit:1 https://packages.microsoft.com/repos/microsoft-ubuntu-noble-prod noble InRelease
Get:2 https://apt.releases.hashicorp.com noble InRelease [12.9 kB]
Hit:3 https://dl.yarnpkg.com/debian stable InRelease
Hit:4 https://repo.anaconda.com/pkgs/misc/debrepo/conda stable InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:6 http://archive.ubuntu.com/ubuntu noble InRelease
Get:7 https://apt.releases.hashicorp.com noble/main amd64 Packages [266 kB]
Hit:8 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:9 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Fetched 279 kB in 1s (225 kB/s)
Reading package lists... Done
```



```
@23-22411-013-sys ➜~/Lab12 $ sudo apt update
sudo apt install -y terraform
Setting up terraform (1.14.3-1) ...
@23-22411-013-sys ➜~/Lab12 $ terraform -version
Terraform v1.14.3
on linux_amd64
@23-22411-013-sys ➜~/Lab12 $ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Finding latest version of hashicorp/http...
- Installing hashicorp/aws v6.27.0...
- Installed hashicorp/aws v6.27.0 (signed by HashiCorp)
- Installing hashicorp/http v3.5.0...
- Installed hashicorp/http v3.5.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
@23-22411-013-sys ➜~/Lab12 $
```

11. Apply the configuration:

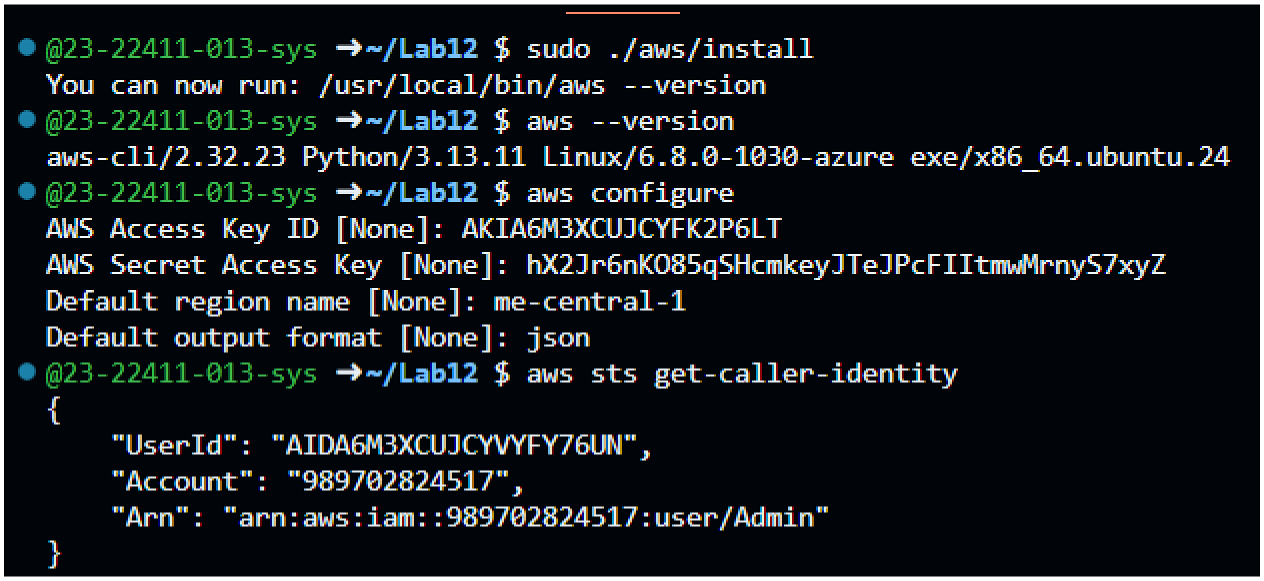
terraform apply -auto-approve

Installing aws cli before terraform apply:

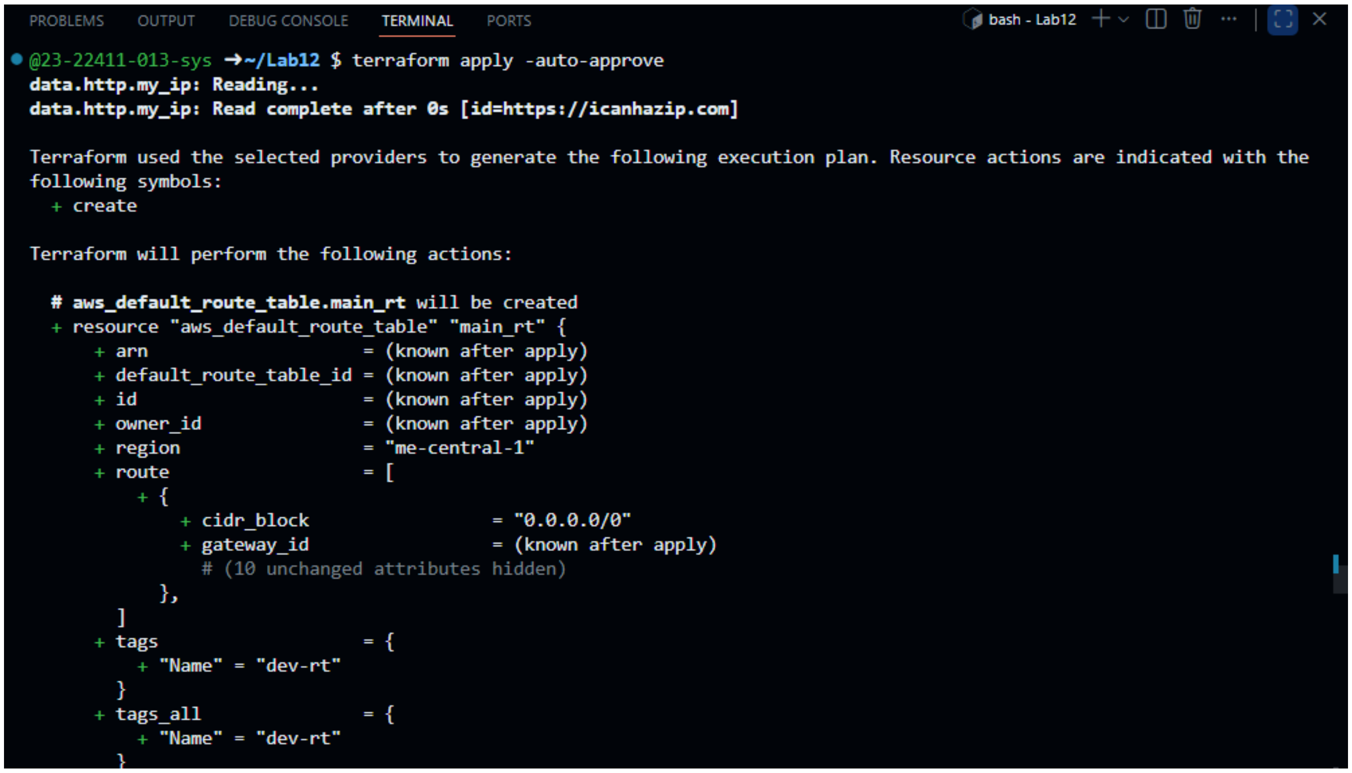
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"

sudo apt install -y unzip

unzip awscliv2.zip



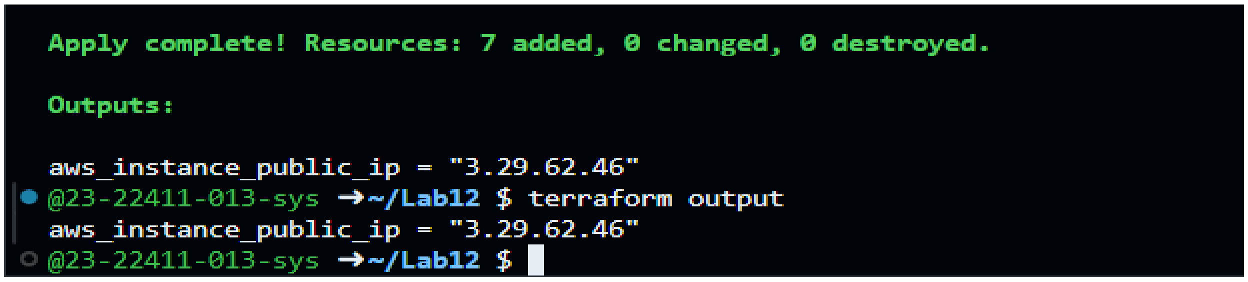
```
● @23-22411-013-sys ➜~/Lab12 $ sudo ./aws/install
  You can now run: /usr/local/bin/aws --version
● @23-22411-013-sys ➜~/Lab12 $ aws --version
  aws-cli/2.32.23 Python/3.13.11 Linux/6.8.0-1030-azure exe/x86_64.ubuntu.24
● @23-22411-013-sys ➜~/Lab12 $ aws configure
  AWS Access Key ID [None]: AKIA6M3XCUJCYFK2P6LT
  AWS Secret Access Key [None]: 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
  Default region name [None]: me-central-1
  Default output format [None]: json
● @23-22411-013-sys ➜~/Lab12 $ aws sts get-caller-identity
  {
      "UserId": "AIDA6M3XCUJCYVYFY76UN",
      "Account": "989702824517",
      "Arn": "arn:aws:iam::989702824517:user/Admin"
  }
```



```
● @23-22411-013-sys ➜~/Lab12 $ terraform apply -auto-approve
  data.http.my_ip: Reading...
  data.http.my_ip: Read complete after 0s [id=https://icanhazip.com]

  Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
  following symbols:
    + create

  Terraform will perform the following actions:

    # aws_default_route_table.main_rt will be created
    + resource "aws_default_route_table" "main_rt" {
        + arn                    = (known after apply)
        + default_route_table_id = (known after apply)
        + id                     = (known after apply)
        + owner_id               = (known after apply)
        + region                 = "me-central-1"
        + route                  = [
            + {
                + cidr_block                 = "0.0.0.0/0"
                + gateway_id                 = (known after apply)
                  # (10 unchanged attributes hidden)
              },
          ]
        + tags                   = {
            + "Name" = "dev-rt"
          }
        + tags_all               = {
            + "Name" = "dev-rt"
          }
```

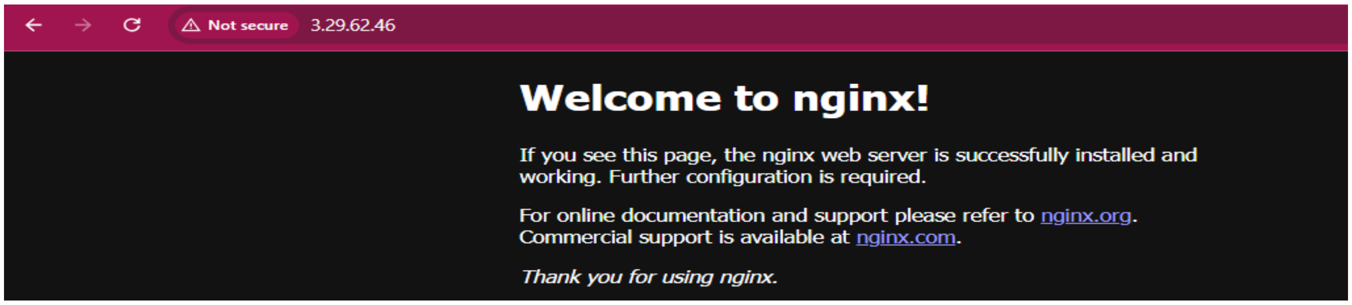
12. Display the output:

terraform output


```
Apply complete! Resources: 7 added, 0 changed, 0 destroyed.

Outputs:

aws_instance_public_ip = "3.29.62.46"
@23-22411-013-sys ➜~/Lab12 $ terraform output
aws_instance_public_ip = "3.29.62.46"
@23-22411-013-sys ➜~/Lab12 $
```

13. Test nginx in browser:

- Open browser and navigate to http://<public-ip>



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

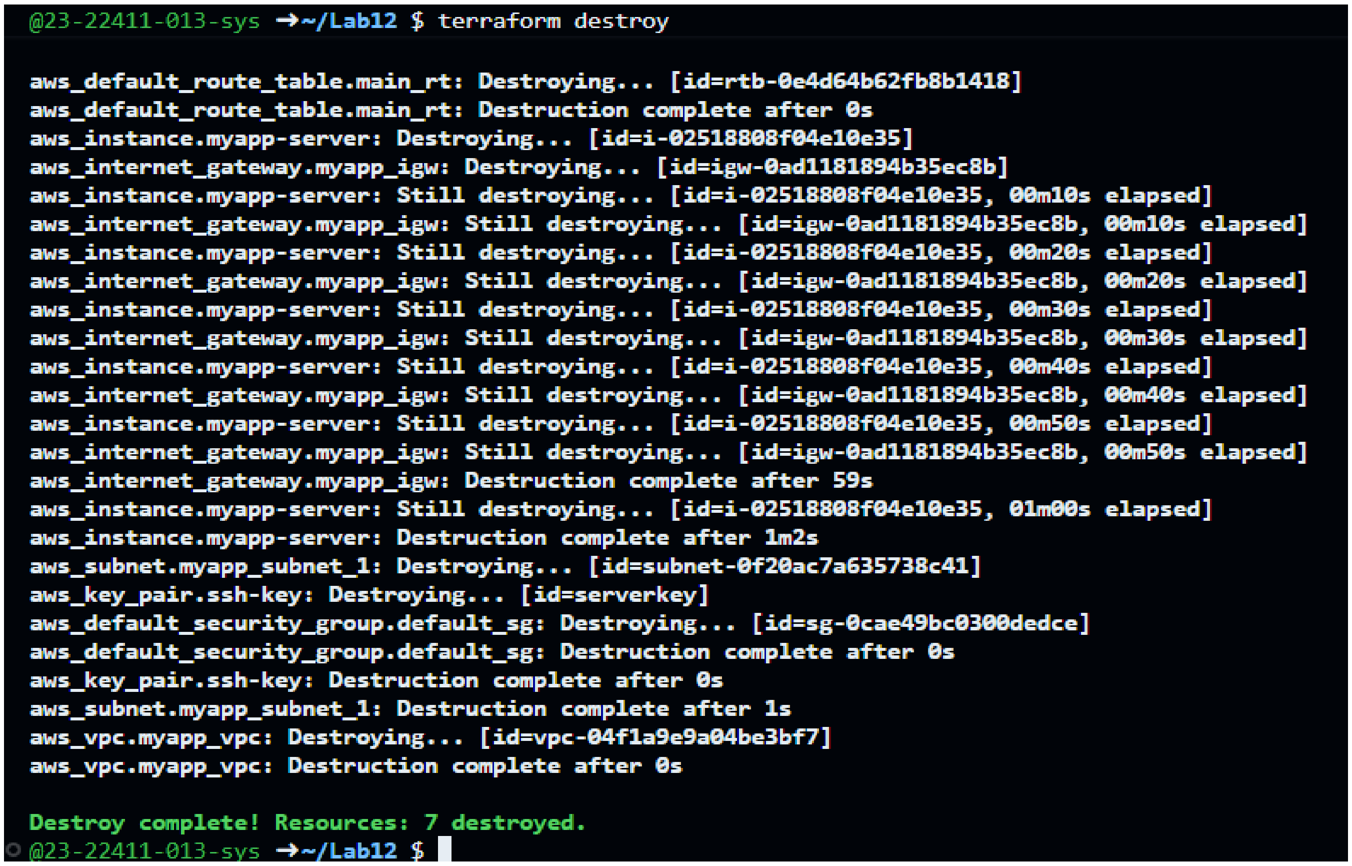
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

14. Destroy resources:

terraform destroy

- Type yes when prompted for confirmation.


```
@23-22411-013-sys ➜~/Lab12 $ terraform destroy

aws_default_route_table.main_rt: Destroying... [id=rtb-0e4d64b62fb8b1418]
aws_default_route_table.main_rt: Destruction complete after 0s
aws_instance.myapp-server: Destroying... [id=i-02518808f04e10e35]
aws_internet_gateway.myapp_igw: Destroying... [id=igw-0ad1181894b35ec8b]
aws_instance.myapp-server: Still destroying... [id=i-02518808f04e10e35, 00m10s elapsed]
aws_internet_gateway.myapp_igw: Still destroying... [id=igw-0ad1181894b35ec8b, 00m10s elapsed]
aws_instance.myapp-server: Still destroying... [id=i-02518808f04e10e35, 00m20s elapsed]
aws_internet_gateway.myapp_igw: Still destroying... [id=igw-0ad1181894b35ec8b, 00m20s elapsed]
aws_instance.myapp-server: Still destroying... [id=i-02518808f04e10e35, 00m30s elapsed]
aws_internet_gateway.myapp_igw: Still destroying... [id=igw-0ad1181894b35ec8b, 00m30s elapsed]
aws_instance.myapp-server: Still destroying... [id=i-02518808f04e10e35, 00m40s elapsed]
aws_internet_gateway.myapp_igw: Still destroying... [id=igw-0ad1181894b35ec8b, 00m40s elapsed]
aws_instance.myapp-server: Still destroying... [id=i-02518808f04e10e35, 00m50s elapsed]
aws_internet_gateway.myapp_igw: Still destroying... [id=igw-0ad1181894b35ec8b, 00m50s elapsed]
aws_internet_gateway.myapp_igw: Destruction complete after 59s
aws_instance.myapp-server: Still destroying... [id=i-02518808f04e10e35, 01m00s elapsed]
aws_instance.myapp-server: Destruction complete after 1m2s
aws_subnet.myapp_subnet_1: Destroying... [id=subnet-0f20ac7a635738c41]
aws_key_pair.ssh-key: Destroying... [id=serverkey]
aws_default_security_group.default_sg: Destroying... [id=sg-0cae49bc0300dedce]
aws_default_security_group.default_sg: Destruction complete after 0s
aws_key_pair.ssh-key: Destruction complete after 0s
aws_subnet.myapp_subnet_1: Destruction complete after 1s
aws_vpc.myapp_vpc: Destroying... [id=vpc-04f1a9e9a04be3bf7]
aws_vpc.myapp_vpc: Destruction complete after 0s

Destroy complete! Resources: 7 destroyed.
@23-22411-013-sys ➜~/Lab12 $
```

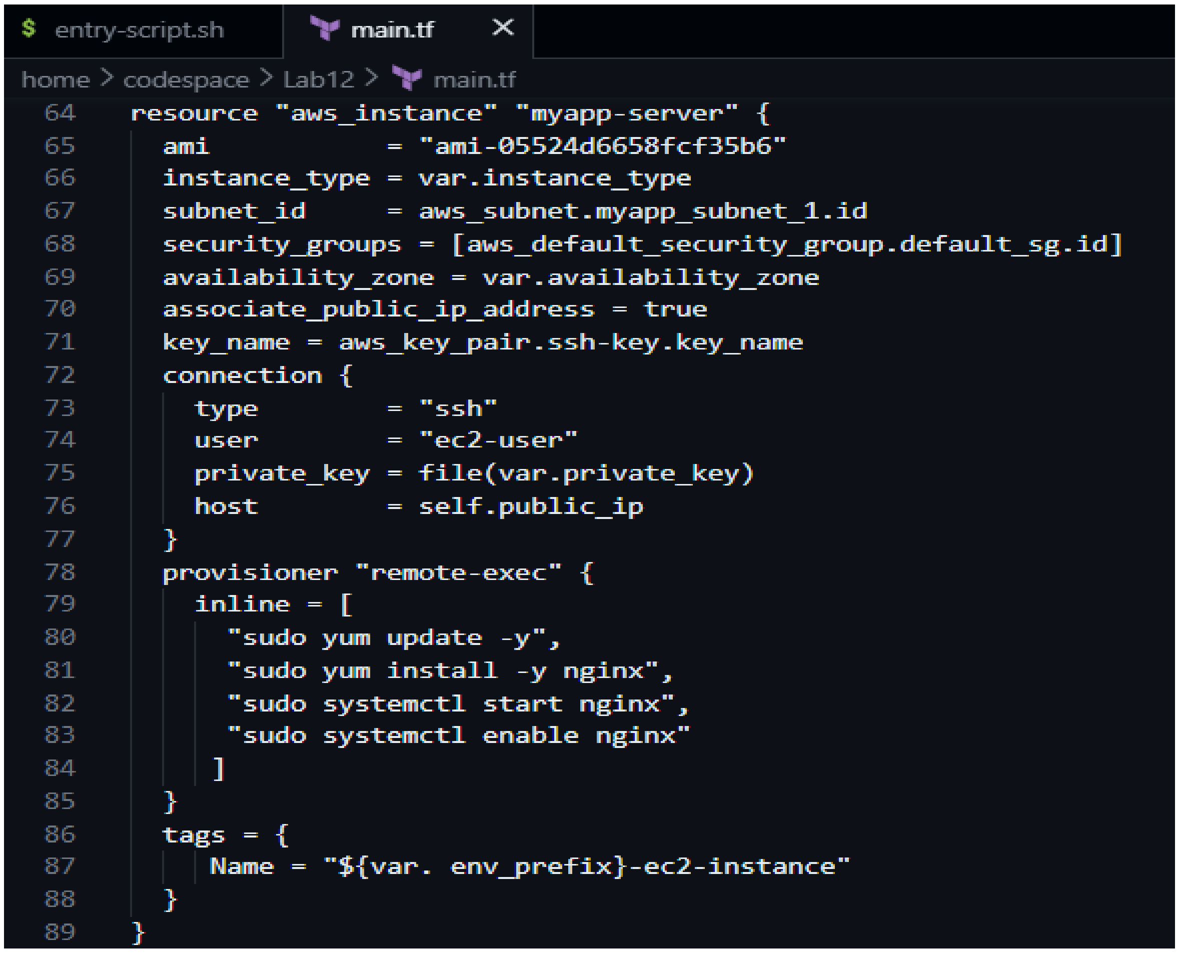
Task 2 — Use remote-exec provisioner

In this task, you will replace the user_data approach with the remote-exec provisioner to install and configure nginx.

1. Modify the aws_instance resource in main.tf to use remote-exec provisioner:

Replace the user_data line with the following provisioner block:

```
resource "aws_instance" "myapp-server" {
 ami           = "ami-05524d6658fcf35b6"
 instance_type = var.instance_type
 subnet_id     = aws_subnet.myapp_subnet_1.id
 security_groups = [aws_default_security_group.default_sg.id]
 availability_zone = var.availability_zone
 associate_public_ip_address = true
 key_name = aws_key_pair.ssh-key.key_name
 connection {
  type        = "ssh"
  user        = "ec2-user"
  private_key = file(var.private_key)
  host        = self.public_ip
 }
 provisioner "remote-exec" {
  inline = [
   "sudo yum update -y",
   "sudo yum install -y nginx",
   "sudo systemctl start nginx",
   "sudo systemctl enable nginx"
   ]
 }
 tags = {
   Name = "${var. env_prefix}-ec2-instance"
 }
}
```



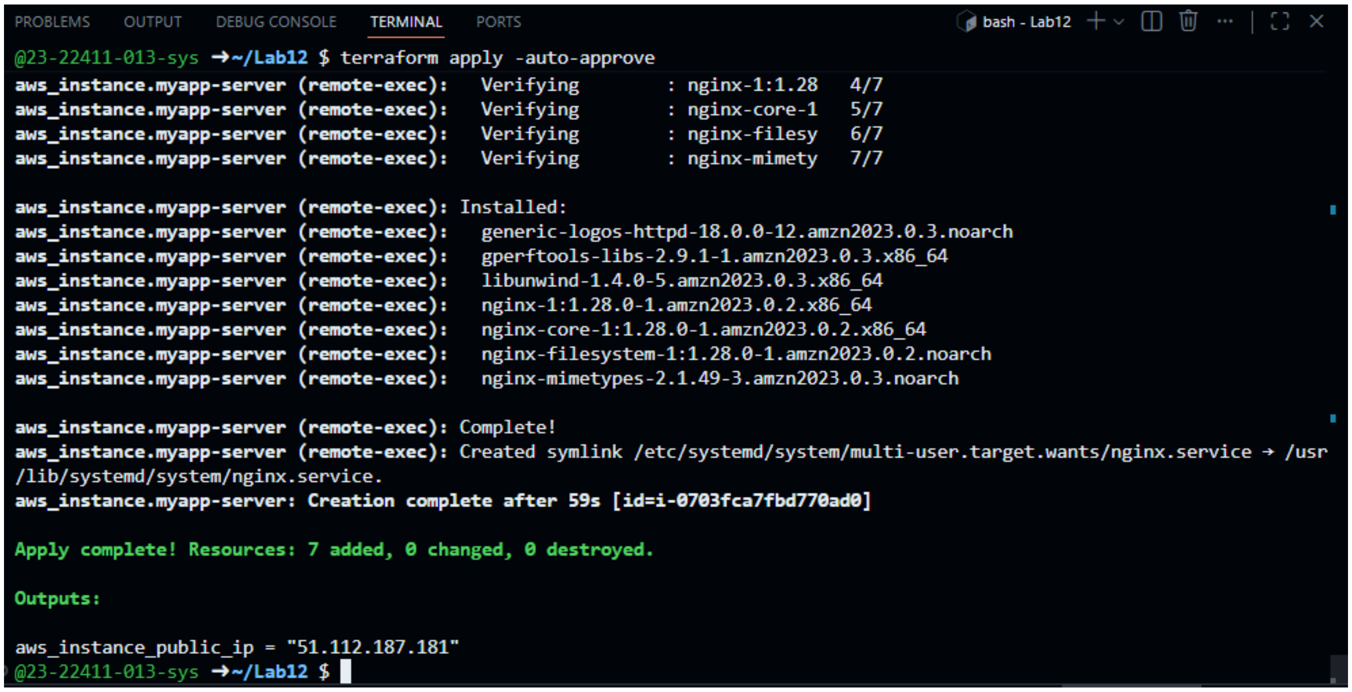
entry-script.sh   main.tf

home > codespace > Lab12 > main.tf

```
resource "aws_instance" "myapp-server" {
  ami            = "ami-05524d6658fcf35b6"
  instance_type = var.instance_type
  subnet_id      = aws_subnet.myapp_subnet_1.id
  security_groups = [aws_default_security_group.default_sg.id]
  availability_zone = var.availability_zone
  associate_public_ip_address = true
  key_name = aws_key_pair.ssh-key.key_name
  connection {
    type         = "ssh"
    user         = "ec2-user"
    private_key = file(var.private_key)
    host         = self.public_ip
  }
  provisioner "remote-exec" {
    inline = [
      "sudo yum update -y",
      "sudo yum install -y nginx",
      "sudo systemctl start nginx",
      "sudo systemctl enable nginx"
    ]
  }
  tags = {
    Name = "${var. env_prefix}-ec2-instance"
  }
}
```

2. Apply the configuration:

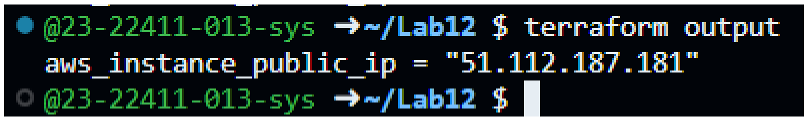
terraform apply -auto-approve



```
@23-22411-013-sys →~/Lab12 $ terraform apply -auto-approve
aws_instance.myapp-server (remote-exec):   Verifying        : nginx-1:1.28    4/7
aws_instance.myapp-server (remote-exec):   Verifying        : nginx-core-1    5/7
aws_instance.myapp-server (remote-exec):   Verifying        : nginx-filesy    6/7
aws_instance.myapp-server (remote-exec):   Verifying        : nginx-mimety    7/7

aws_instance.myapp-server (remote-exec): Installed:
aws_instance.myapp-server (remote-exec):   generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
aws_instance.myapp-server (remote-exec):   gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64
aws_instance.myapp-server (remote-exec):   libunwind-1.4.0-5.amzn2023.0.3.x86_64
aws_instance.myapp-server (remote-exec):   nginx-1:1.28.0-1.amzn2023.0.2.x86_64
aws_instance.myapp-server (remote-exec):   nginx-core-1:1.28.0-1.amzn2023.0.2.x86_64
aws_instance.myapp-server (remote-exec):   nginx-filesystem-1:1.28.0-1.amzn2023.0.2.noarch
aws_instance.myapp-server (remote-exec):   nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch

aws_instance.myapp-server (remote-exec): Complete!
aws_instance.myapp-server (remote-exec): Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr
/lib/systemd/system/nginx.service.
aws_instance.myapp-server: Creation complete after 59s [id=i-0703fca7fbd770ad0]

Apply complete! Resources: 7 added, 0 changed, 0 destroyed.

Outputs:

aws_instance_public_ip = "51.112.187.181"
@23-22411-013-sys →~/Lab12 $
```

3. Display the output:

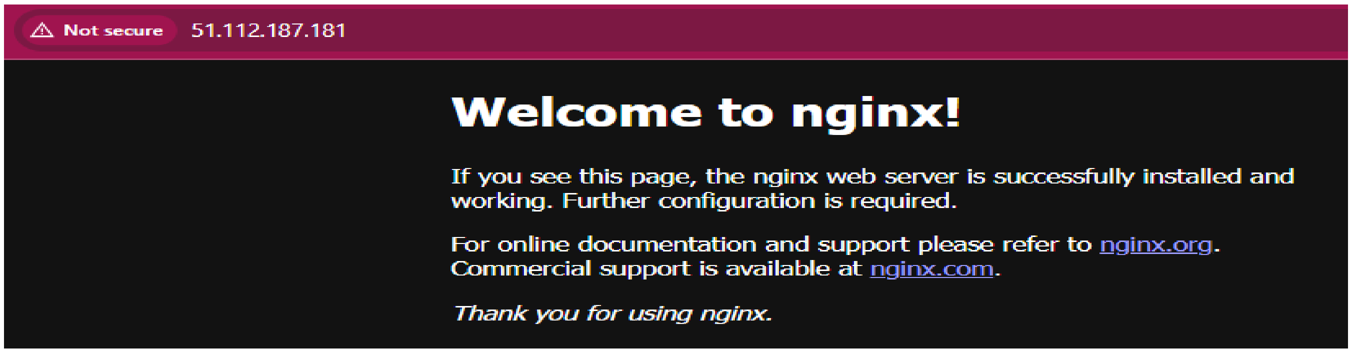
terraform output



```
@23-22411-013-sys ➜~/Lab12 $ terraform output
aws_instance_public_ip = "51.112.187.181"
@23-22411-013-sys ➜~/Lab12 $
```

4. Test nginx in browser:

- Open browser and navigate to http://<public-ip>



Not secure 51.112.187.181

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

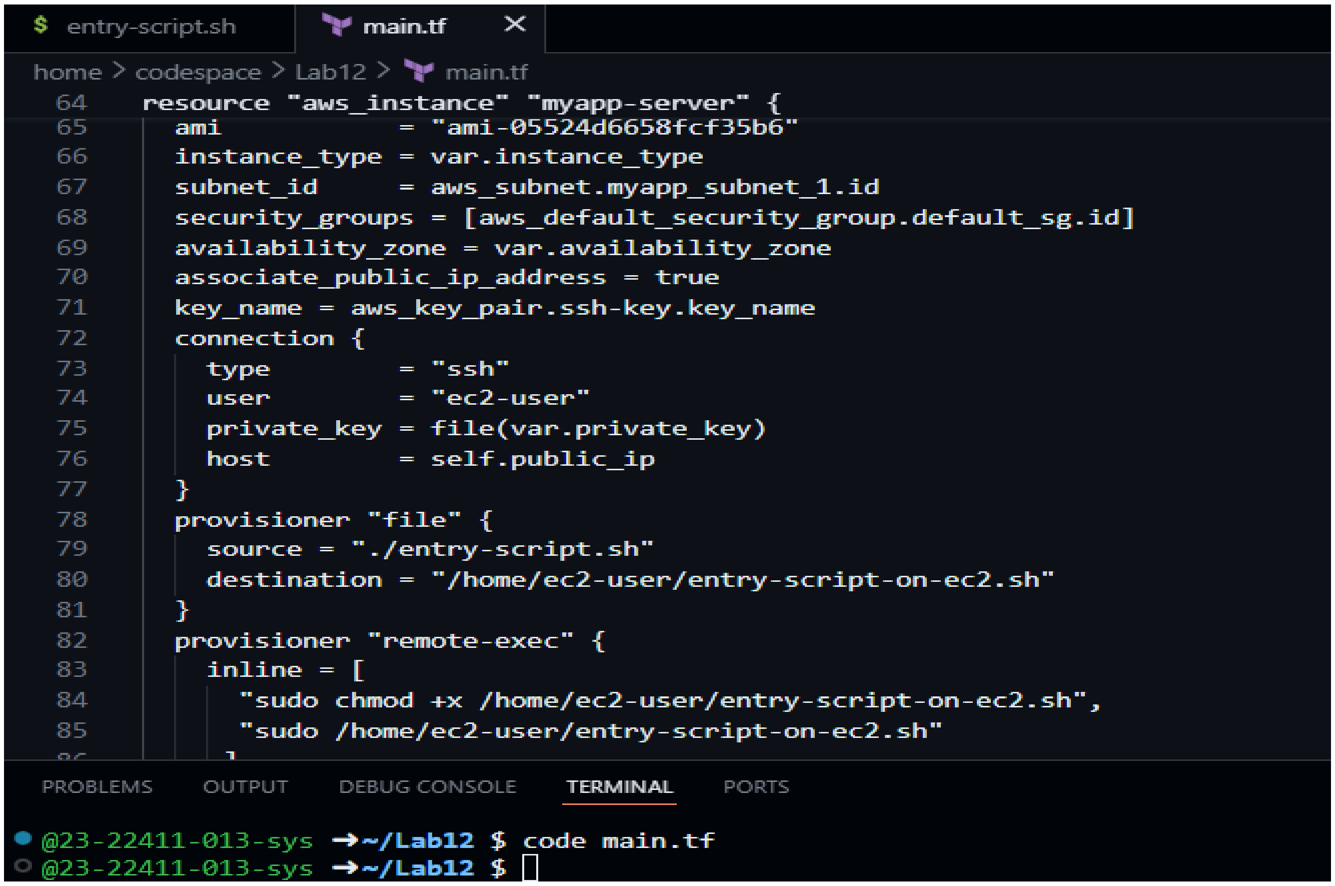
Thank you for using nginx.

Task 3 — Use file and local-exec provisioners

In this task, you will add the file provisioner to upload the script and the local-exec provisioner to log instance information locally.

1. Modify the aws_instance resource in main.tf to include all three provisioners:

```
resource "aws_instance" "myapp-server" {
  ami           = "ami-05524d6658fcf35b6"
  instance_type = var.instance_type
  subnet_id     = aws_subnet.myapp_subnet_1.id
  security_groups = [aws_default_security_group.default_sg.id]
  availability_zone = var.availability_zone
  associate_public_ip_address = true
  key_name = aws_key_pair.ssh-key.key_name
  connection {
    type        = "ssh"
    user        = "ec2-user"
    private_key = file(var.private_key)
    host        = self.public_ip
  }
  provisioner "file" {
    source = "./entry-script.sh"
    destination = "/home/ec2-user/entry-script-on-ec2.sh"
  }
  provisioner "remote-exec" {
    inline = [
      "sudo chmod +x /home/ec2-user/entry-script-on-ec2.sh",
      "sudo /home/ec2-user/entry-script-on-ec2.sh"
     ]  }
  provisioner "local-exec" {
    command = <<-EOF
      echo Instance ${self.id} with public IP ${self.public_ip} has been created
    EOF   }
  tags = {
     Name = "${var.env_prefix}-ec2-instance"
  }  }
```

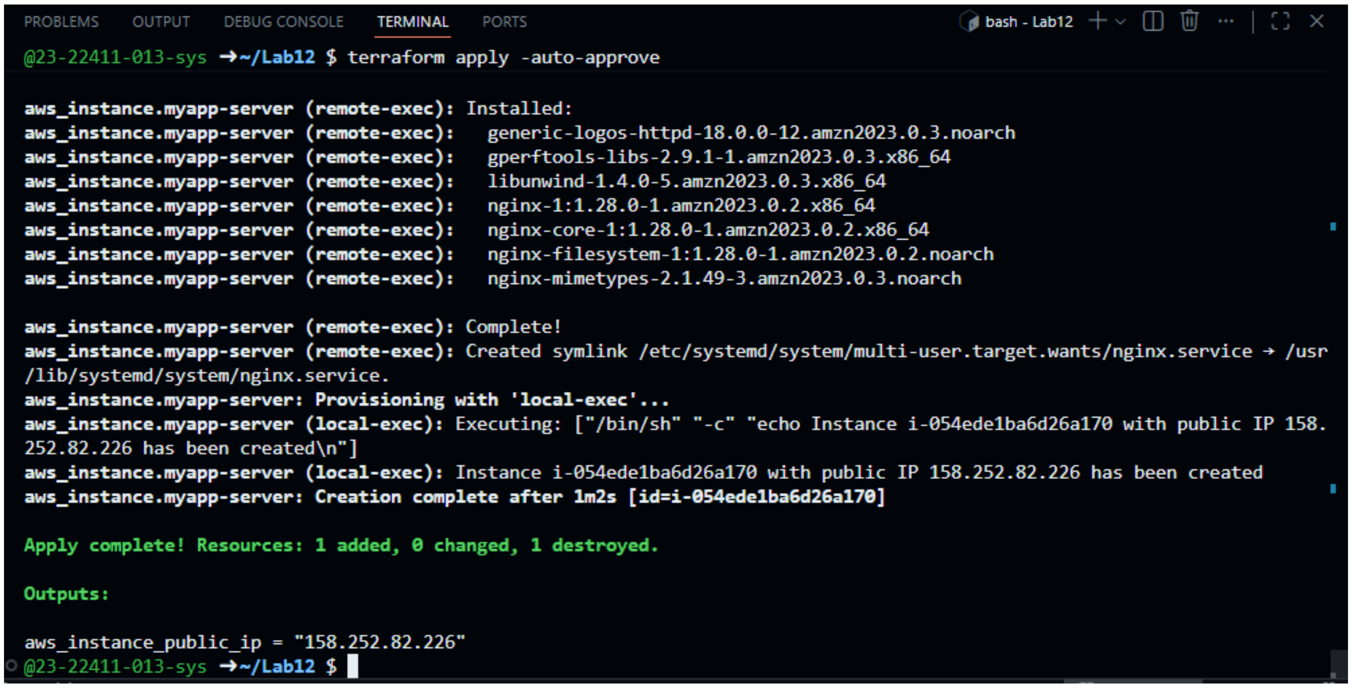


```
resource "aws_instance" "myapp-server" {
  ami           = "ami-05524d6658fcf35b6"
  instance_type = var.instance_type
  subnet_id     = aws_subnet.myapp_subnet_1.id
  security_groups = [aws_default_security_group.default_sg.id]
  availability_zone = var.availability_zone
  associate_public_ip_address = true
  key_name = aws_key_pair.ssh-key.key_name
  connection {
    type        = "ssh"
    user        = "ec2-user"
    private_key = file(var.private_key)
    host        = self.public_ip
  }
  provisioner "file" {
    source = "./entry-script.sh"
    destination = "/home/ec2-user/entry-script-on-ec2.sh"
  }
  provisioner "remote-exec" {
    inline = [
      "sudo chmod +x /home/ec2-user/entry-script-on-ec2.sh",
      "sudo /home/ec2-user/entry-script-on-ec2.sh"
```

```
@23-22411-013-sys ➜ ~/Lab12 $ code main.tf
@23-22411-013-sys ➜ ~/Lab12 $
```

2. Apply the configuration:

terraform apply -auto-approve

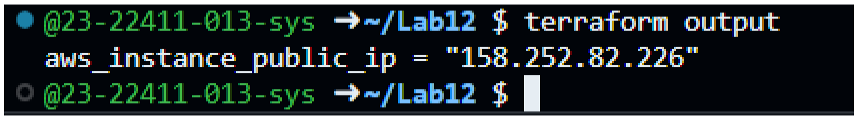


```
@23-22411-013-sys ➜~/Lab12 $ terraform apply -auto-approve

aws_instance.myapp-server (remote-exec): Installed:
aws_instance.myapp-server (remote-exec):   generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
aws_instance.myapp-server (remote-exec):   gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64
aws_instance.myapp-server (remote-exec):   libunwind-1.4.0-5.amzn2023.0.3.x86_64
aws_instance.myapp-server (remote-exec):   nginx-1:1.28.0-1.amzn2023.0.2.x86_64
aws_instance.myapp-server (remote-exec):   nginx-core-1:1.28.0-1.amzn2023.0.2.x86_64
aws_instance.myapp-server (remote-exec):   nginx-filesystem-1:1.28.0-1.amzn2023.0.2.noarch
aws_instance.myapp-server (remote-exec):   nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch

aws_instance.myapp-server (remote-exec): Complete!
aws_instance.myapp-server (remote-exec): Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr
/lib/systemd/system/nginx.service.
aws_instance.myapp-server: Provisioning with 'local-exec'...
aws_instance.myapp-server (local-exec): Executing: ["/bin/sh" "-c" "echo Instance i-054ede1ba6d26a170 with public IP 158.
252.82.226 has been created\n"]
aws_instance.myapp-server (local-exec): Instance i-054ede1ba6d26a170 with public IP 158.252.82.226 has been created
aws_instance.myapp-server: Creation complete after 1m2s [id=i-054ede1ba6d26a170]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.

Outputs:

aws_instance_public_ip = "158.252.82.226"
@23-22411-013-sys ➜~/Lab12 $
```

- **Save screenshot as:** task3_terraform_apply.png — terraform apply output showing all provisioners execution.

3. Display the output:

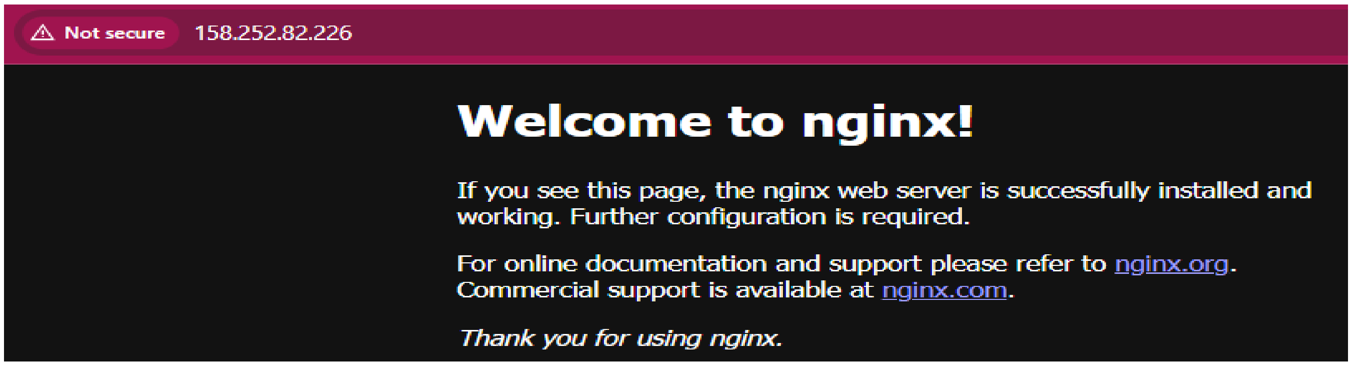
terraform output



```
@23-22411-013-sys ➜~/Lab12 $ terraform output
aws_instance_public_ip = "158.252.82.226"
@23-22411-013-sys ➜~/Lab12 $
```

4. Test nginx in browser:

- Open browser and navigate to http://<public-ip>



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

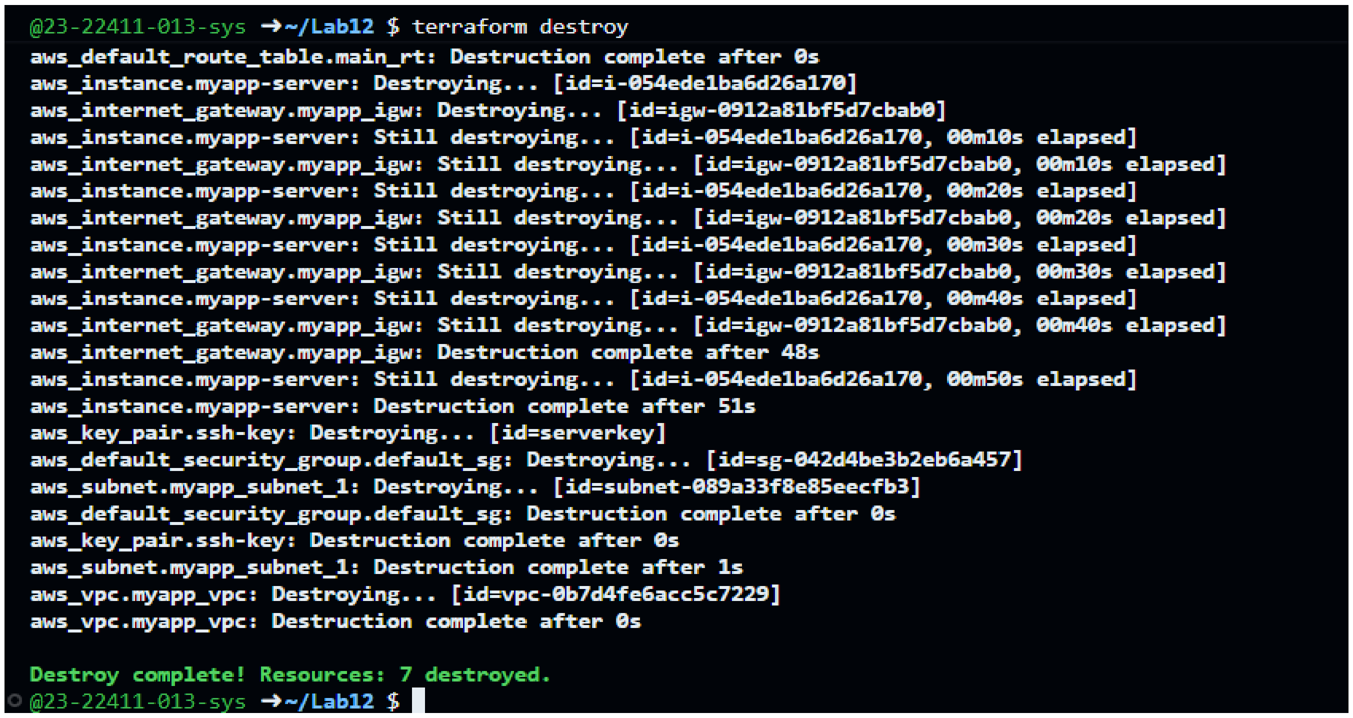
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

5. Destroy the resources:

terraform destroy

- Type yes when prompted.

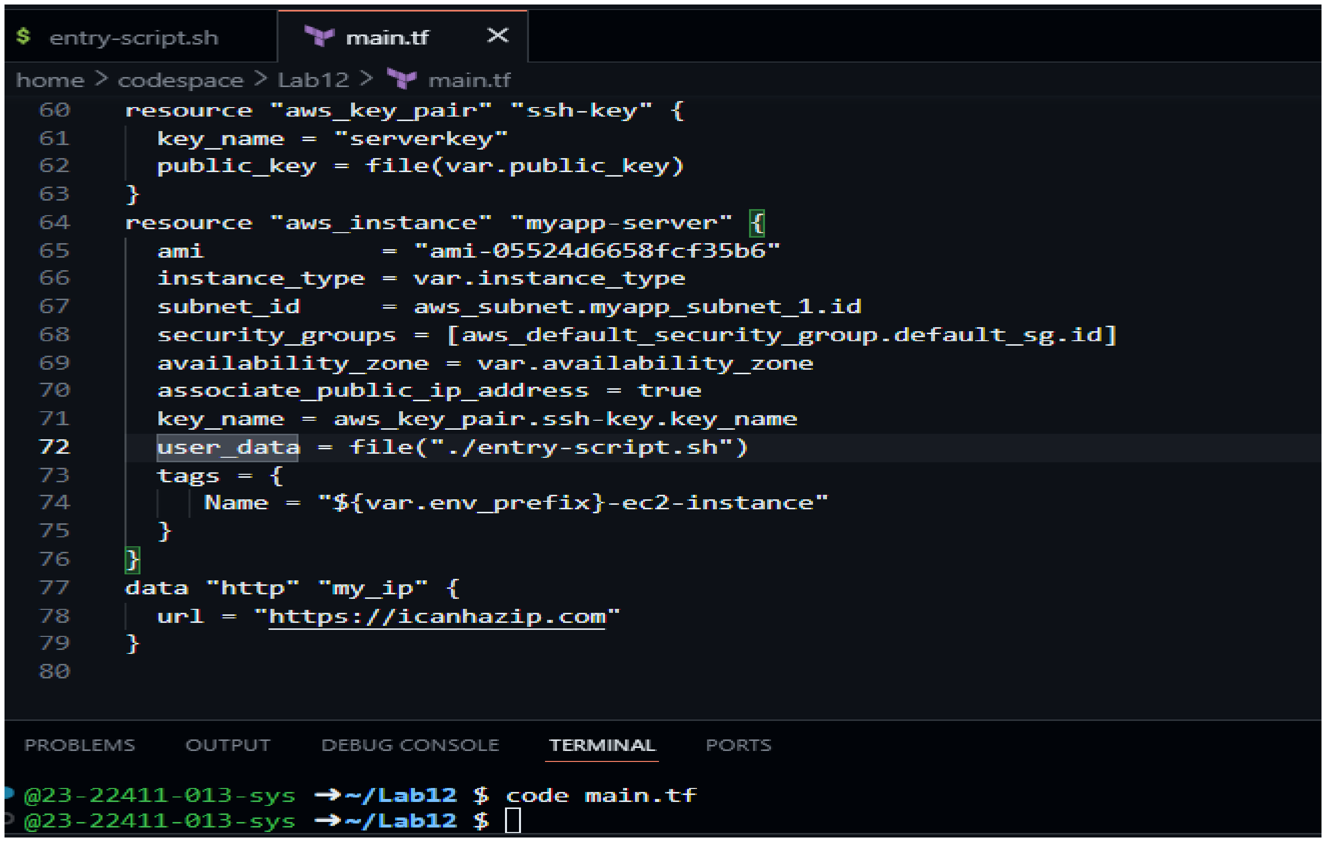


```
@23-22411-013-sys ➜~/Lab12 $ terraform destroy
aws_default_route_table.main_rt: Destruction complete after 0s
aws_instance.myapp-server: Destroying... [id=i-054ede1ba6d26a170]
aws_internet_gateway.myapp_igw: Destroying... [id=igw-0912a81bf5d7cbab0]
aws_instance.myapp-server: Still destroying... [id=i-054ede1ba6d26a170, 00m10s elapsed]
aws_internet_gateway.myapp_igw: Still destroying... [id=igw-0912a81bf5d7cbab0, 00m10s elapsed]
aws_instance.myapp-server: Still destroying... [id=i-054ede1ba6d26a170, 00m20s elapsed]
aws_internet_gateway.myapp_igw: Still destroying... [id=igw-0912a81bf5d7cbab0, 00m20s elapsed]
aws_instance.myapp-server: Still destroying... [id=i-054ede1ba6d26a170, 00m30s elapsed]
aws_internet_gateway.myapp_igw: Still destroying... [id=igw-0912a81bf5d7cbab0, 00m30s elapsed]
aws_instance.myapp-server: Still destroying... [id=i-054ede1ba6d26a170, 00m40s elapsed]
aws_internet_gateway.myapp_igw: Still destroying... [id=igw-0912a81bf5d7cbab0, 00m40s elapsed]
aws_internet_gateway.myapp_igw: Destruction complete after 48s
aws_instance.myapp-server: Still destroying... [id=i-054ede1ba6d26a170, 00m50s elapsed]
aws_instance.myapp-server: Destruction complete after 51s
aws_key_pair.ssh-key: Destroying... [id=serverkey]
aws_default_security_group.default_sg: Destroying... [id=sg-042d4be3b2eb6a457]
aws_subnet.myapp_subnet_1: Destroying... [id=subnet-089a33f8e85eecfb3]
aws_default_security_group.default_sg: Destruction complete after 0s
aws_key_pair.ssh-key: Destruction complete after 0s
aws_subnet.myapp_subnet_1: Destruction complete after 1s
aws_vpc.myapp_vpc: Destroying... [id=vpc-0b7d4fe6acc5c7229]
aws_vpc.myapp_vpc: Destruction complete after 0s

Destroy complete! Resources: 7 destroyed.
@23-22411-013-sys ➜~/Lab12 $
```

6. Remove the provisioners and restore user_data:

Replace the connection and provisioner blocks with:

user_data = file("./entry-script.sh")



```
resource "aws_key_pair" "ssh-key" {
  key_name = "serverkey"
  public_key = file(var.public_key)
}
resource "aws_instance" "myapp-server" {
  ami           = "ami-05524d6658fcf35b6"
  instance_type = var.instance_type
  subnet_id     = aws_subnet.myapp_subnet_1.id
  security_groups = [aws_default_security_group.default_sg.id]
  availability_zone = var.availability_zone
  associate_public_ip_address = true
  key_name = aws_key_pair.ssh-key.key_name
  user_data = file("./entry-script.sh")
  tags = {
    Name = "${var.env_prefix}-ec2-instance"
  }
}
data "http" "my_ip" {
  url = "https://icanhazip.com"
}
```

```
@23-22411-013-sys ➜~/Lab12 $ code main.tf
@23-22411-013-sys ➜~/Lab12 $
```

Task 4 — Create Terraform modules (subnet module)

In this task, you will create a reusable subnet module to organize your infrastructure code better.

1. Create the module directory structure:

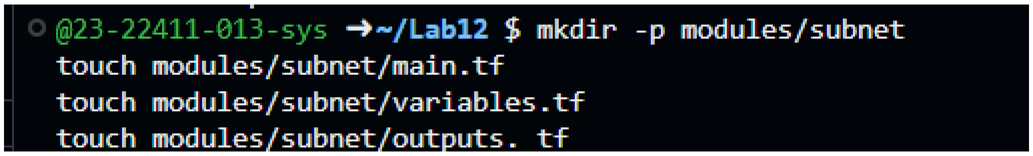
mkdir -p modules/subnet

touch modules/subnet/main.tf

touch modules/subnet/variables.tf

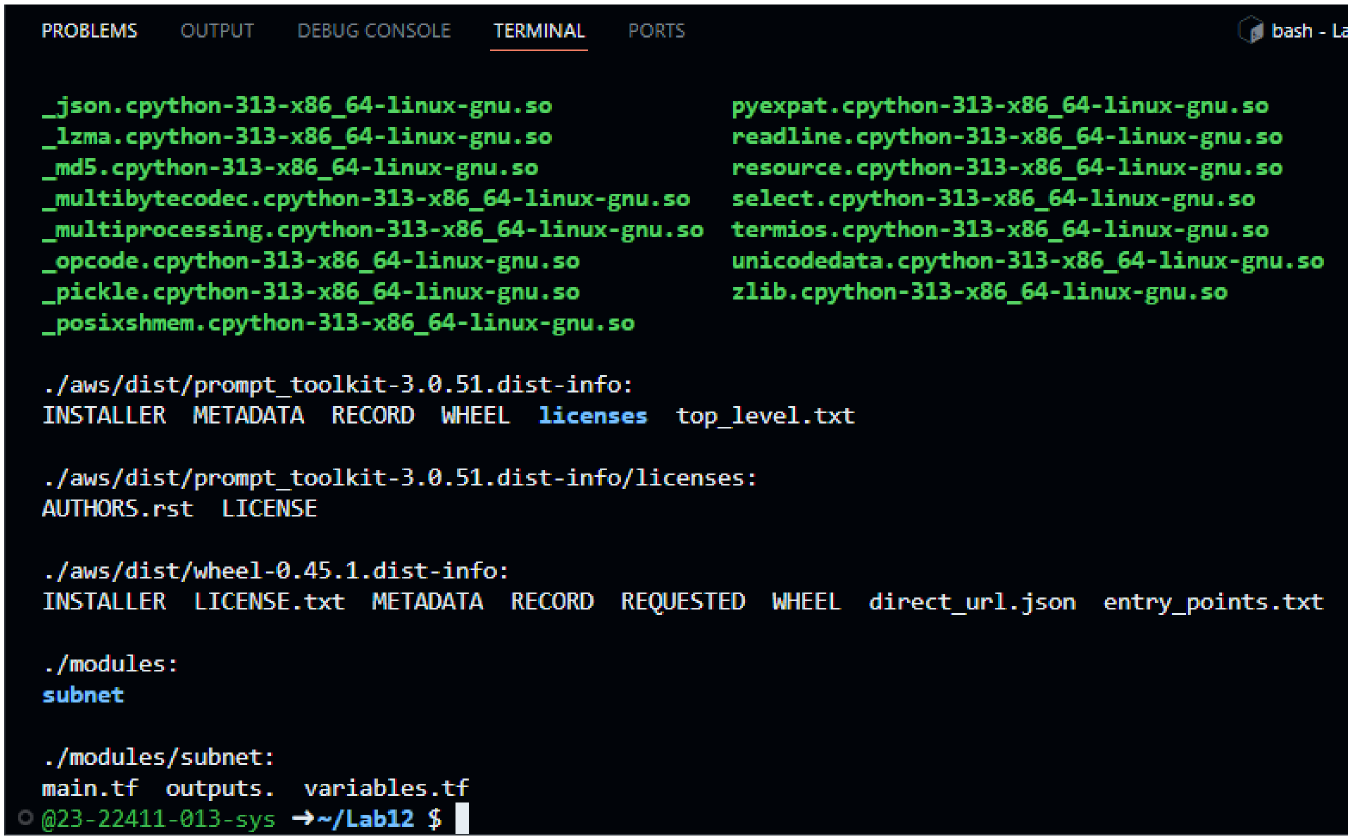
touch modules/subnet/outputs. tf

- **Save screenshot as:** task4_module_structure.png — terminal showing module directory structure (use tree or ls -R).



```
@23-22411-013-sys ➜~/Lab12 $ mkdir -p modules/subnet
touch modules/subnet/main.tf
touch modules/subnet/variables.tf
touch modules/subnet/outputs. tf
```

ls -R



```
_json.cpython-313-x86_64-linux-gnu.so            pyexpat.cpython-313-x86_64-linux-gnu.so
_lzma.cpython-313-x86_64-linux-gnu.so            readline.cpython-313-x86_64-linux-gnu.so
_md5.cpython-313-x86_64-linux-gnu.so             resource.cpython-313-x86_64-linux-gnu.so
_multibytecodec.cpython-313-x86_64-linux-gnu.so  select.cpython-313-x86_64-linux-gnu.so
_multiprocessing.cpython-313-x86_64-linux-gnu.so termios.cpython-313-x86_64-linux-gnu.so
_opcode.cpython-313-x86_64-linux-gnu.so          unicodedata.cpython-313-x86_64-linux-gnu.so
_pickle.cpython-313-x86_64-linux-gnu.so          zlib.cpython-313-x86_64-linux-gnu.so
_posixshmem.cpython-313-x86_64-linux-gnu.so

./aws/dist/prompt_toolkit-3.0.51.dist-info:
INSTALLER  METADATA  RECORD  WHEEL  licenses  top_level.txt

./aws/dist/prompt_toolkit-3.0.51.dist-info/licenses:
AUTHORS.rst  LICENSE

./aws/dist/wheel-0.45.1.dist-info:
INSTALLER  LICENSE.txt  METADATA  RECORD  REQUESTED  WHEEL  direct_url.json  entry_points.txt

./modules:
subnet

./modules/subnet:
main.tf  outputs.  variables.tf
@23-22411-013-sys ➜~/Lab12 $
```

2. Create modules/subnet/variables.tf:

variable "vpc_id" {}

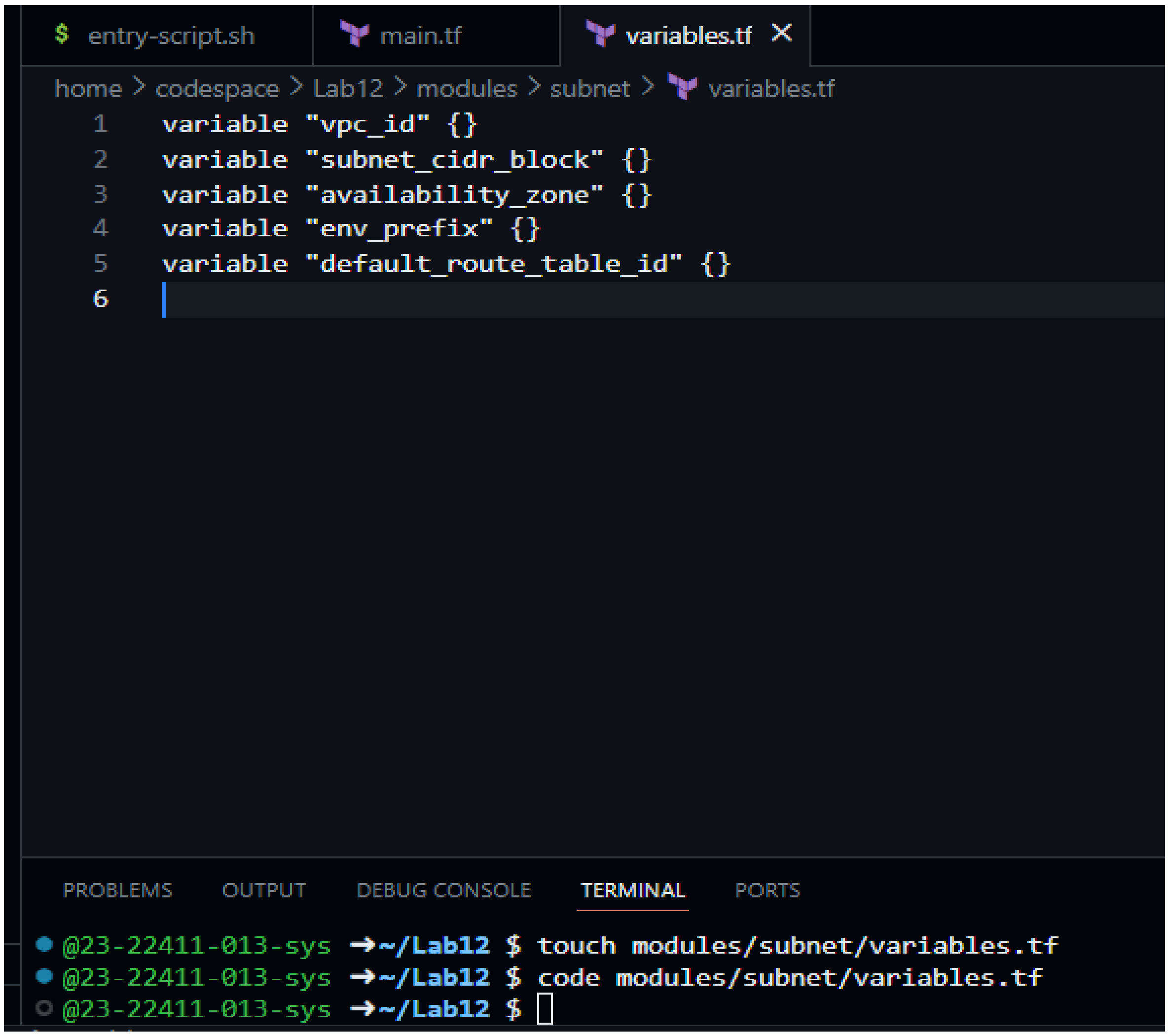
variable "subnet_cidr_block" {}

variable "availability_zone" {}

variable "env_prefix" {}

variable "default_route_table_id" {}

- **Save screenshot as:** task4_subnet_variables.png — content of modules/subnet/variables.tf.



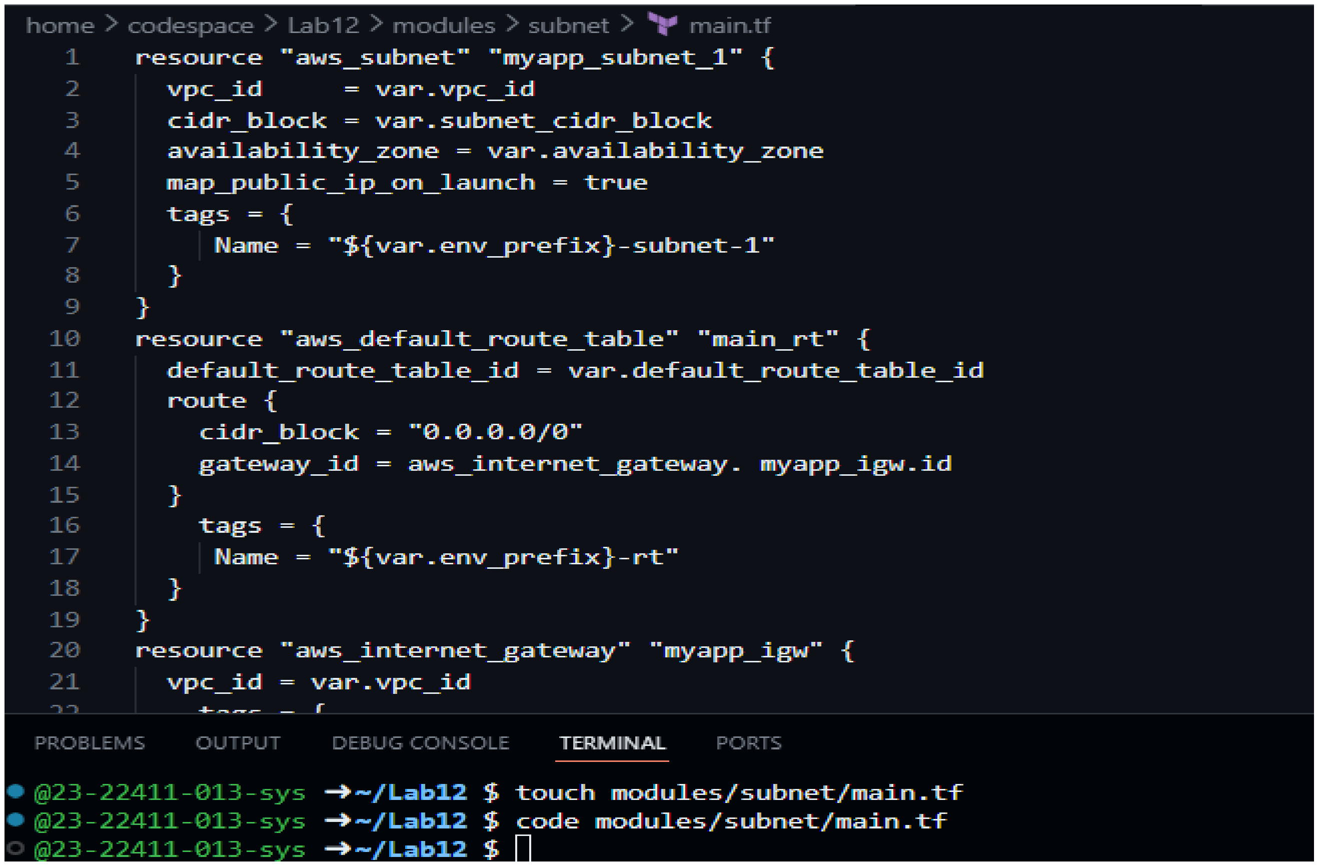
```
variable "vpc_id" {}
variable "subnet_cidr_block" {}
variable "availability_zone" {}
variable "env_prefix" {}
variable "default_route_table_id" {}
```

```
@23-22411-013-sys →~/Lab12 $ touch modules/subnet/variables.tf
@23-22411-013-sys →~/Lab12 $ code modules/subnet/variables.tf
@23-22411-013-sys →~/Lab12 $ 
```

3. Create modules/subnet/main.tf:

```
resource "aws_subnet" "myapp_subnet_1" {
  vpc_id     = var.vpc_id
  cidr_block = var.subnet_cidr_block
  availability_zone = var.availability_zone
  map_public_ip_on_launch = true
  tags = {
    Name = "${var.env_prefix}-subnet-1"
  }
}
```

```
resource "aws_default_route_table" "main_rt" {
  default_route_table_id = var.default_route_table_id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway. myapp_igw.id
  }
    tags = {
     Name = "${var.env_prefix}-rt"
  }
}
resource "aws_internet_gateway" "myapp_igw" {
  vpc_id = var.vpc_id
    tags = {
     Name = "${var.env_prefix}-igw"
  }
}
```

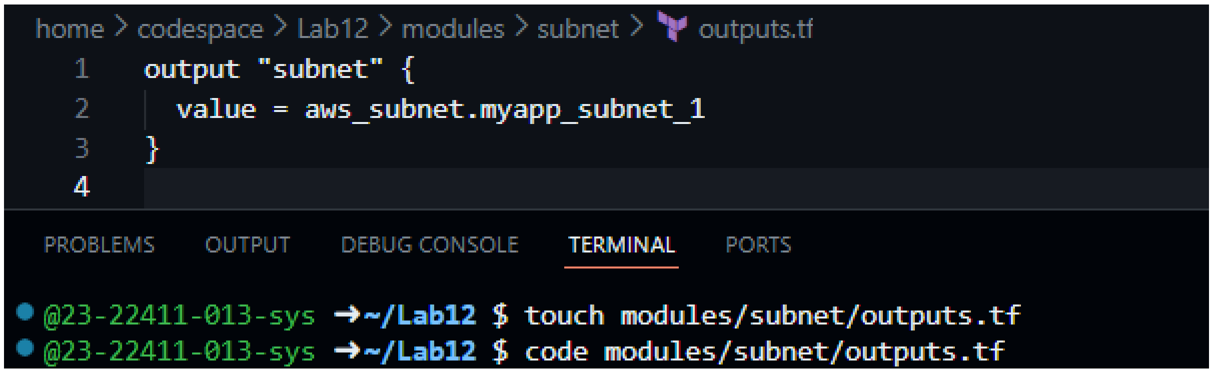


```
resource "aws_subnet" "myapp_subnet_1" {
  vpc_id      = var.vpc_id
  cidr_block = var.subnet_cidr_block
  availability_zone = var.availability_zone
  map_public_ip_on_launch = true
  tags = {
    Name = "${var.env_prefix}-subnet-1"
  }
}
resource "aws_default_route_table" "main_rt" {
  default_route_table_id = var.default_route_table_id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway. myapp_igw.id
  }
    tags = {
    Name = "${var.env_prefix}-rt"
  }
}
resource "aws_internet_gateway" "myapp_igw" {
  vpc_id = var.vpc_id
```

```
@23-22411-013-sys →~/Lab12 $ touch modules/subnet/main.tf
@23-22411-013-sys →~/Lab12 $ code modules/subnet/main.tf
@23-22411-013-sys →~/Lab12 $
```

4. Create modules/subnet/outputs.tf:

```
output "subnet" {
  value = aws_subnet.myapp_subnet_1
}
```



```
output "subnet" {
  value = aws_subnet.myapp_subnet_1
}
```

```
@23-22411-013-sys ➜~/Lab12 $ touch modules/subnet/outputs.tf
@23-22411-013-sys ➜~/Lab12 $ code modules/subnet/outputs.tf
```

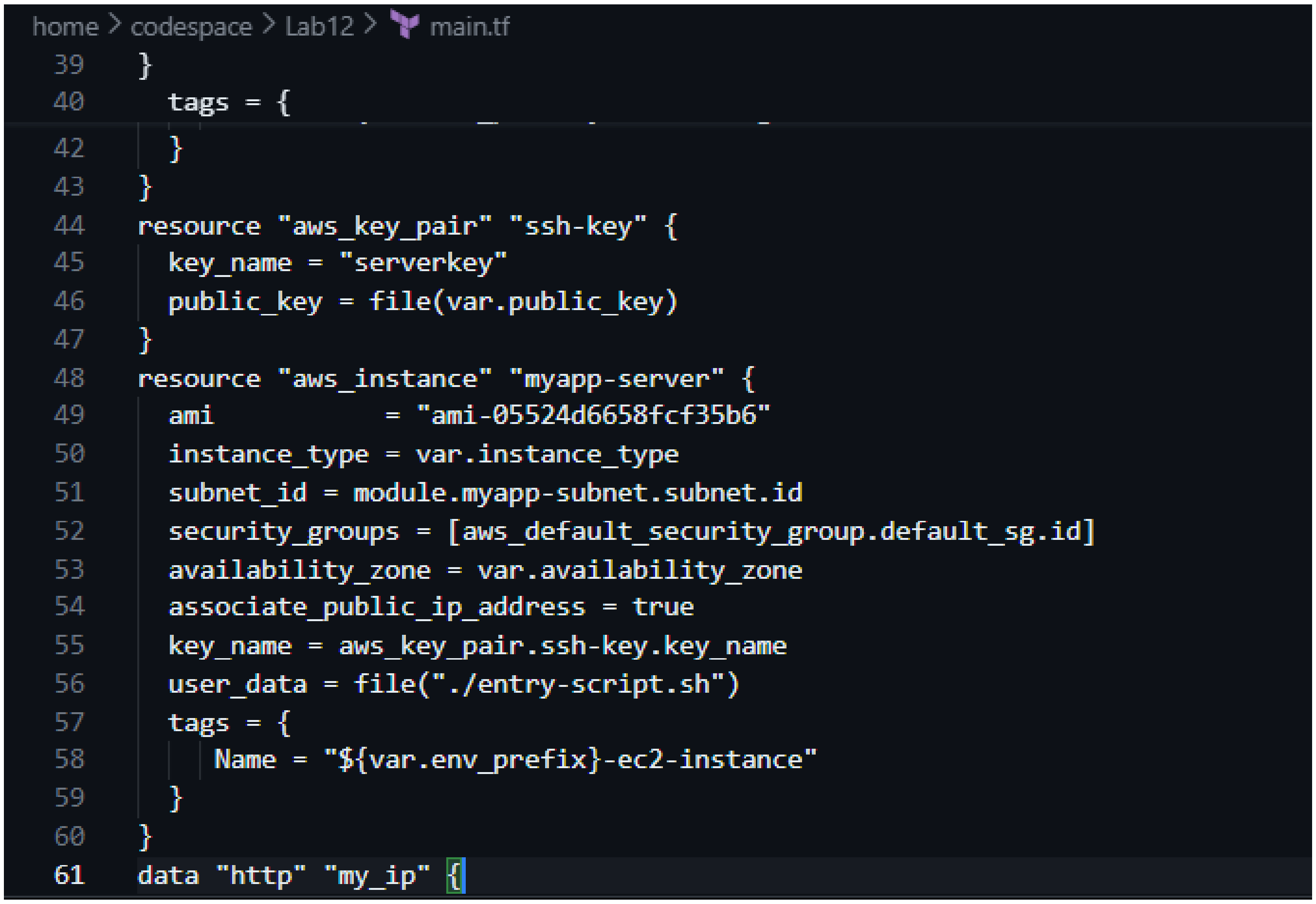
5. Modify the root main.tf to use the subnet module:

Remove the subnet, route table, and internet gateway resources and replace them with:

```
module "myapp-subnet" {
  source = "./modules/subnet"
  vpc_id = aws_vpc.myapp_vpc. id
  subnet_cidr_block = var.subnet_cidr_block
  availability_zone = var.availability_zone
  env_prefix = var.env_prefix
  default_route_table_id = aws_vpc.myapp_vpc.default_route_table_id
}
```

And update the instance resource to reference the module output:

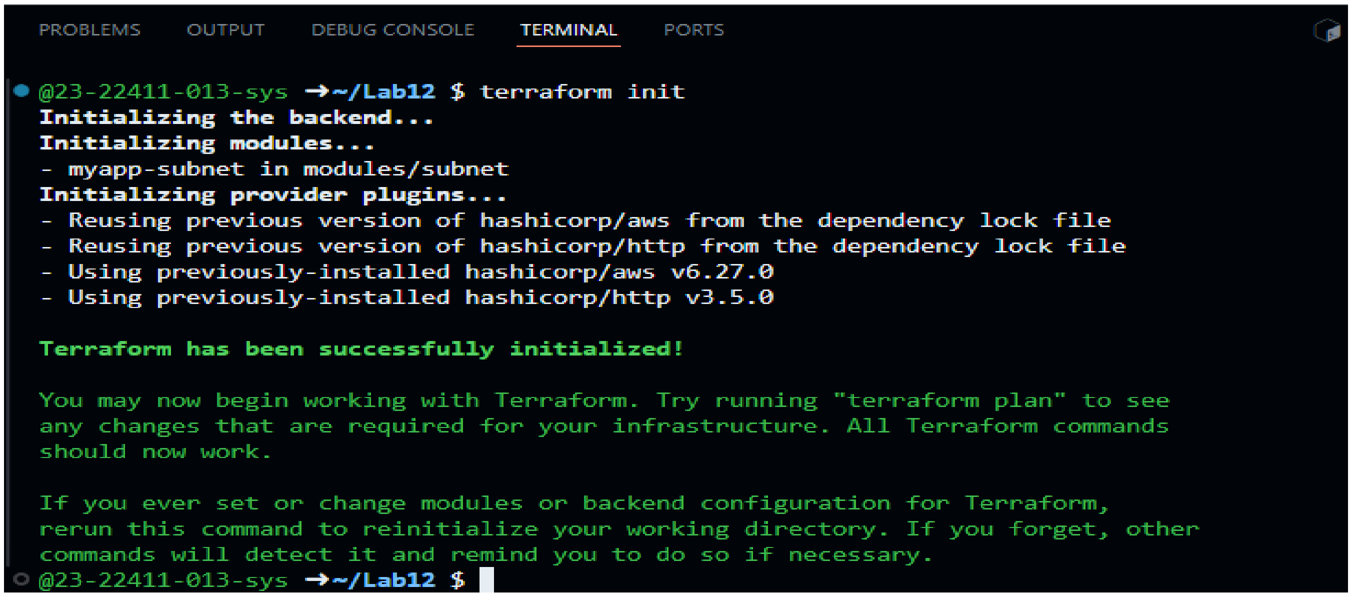
```
resource "aws_instance" "myapp-server" {
  # ... other settings ...
  subnet_id     = module.myapp-subnet. subnet.id
  # ... rest of configuration ...
}
```



```
home > codespace > Lab12 > main.tf
}
  tags = {
  }
}
resource "aws_key_pair" "ssh-key" {
  key_name = "serverkey"
  public_key = file(var.public_key)
}
resource "aws_instance" "myapp-server" {
  ami           = "ami-05524d6658fcf35b6"
  instance_type = var.instance_type
  subnet_id = module.myapp-subnet.subnet.id
  security_groups = [aws_default_security_group.default_sg.id]
  availability_zone = var.availability_zone
  associate_public_ip_address = true
  key_name = aws_key_pair.ssh-key.key_name
  user_data = file("./entry-script.sh")
  tags = {
    Name = "${var.env_prefix}-ec2-instance"
  }
}
data "http" "my_ip" {
```

6. Initialize Terraform to download the module:

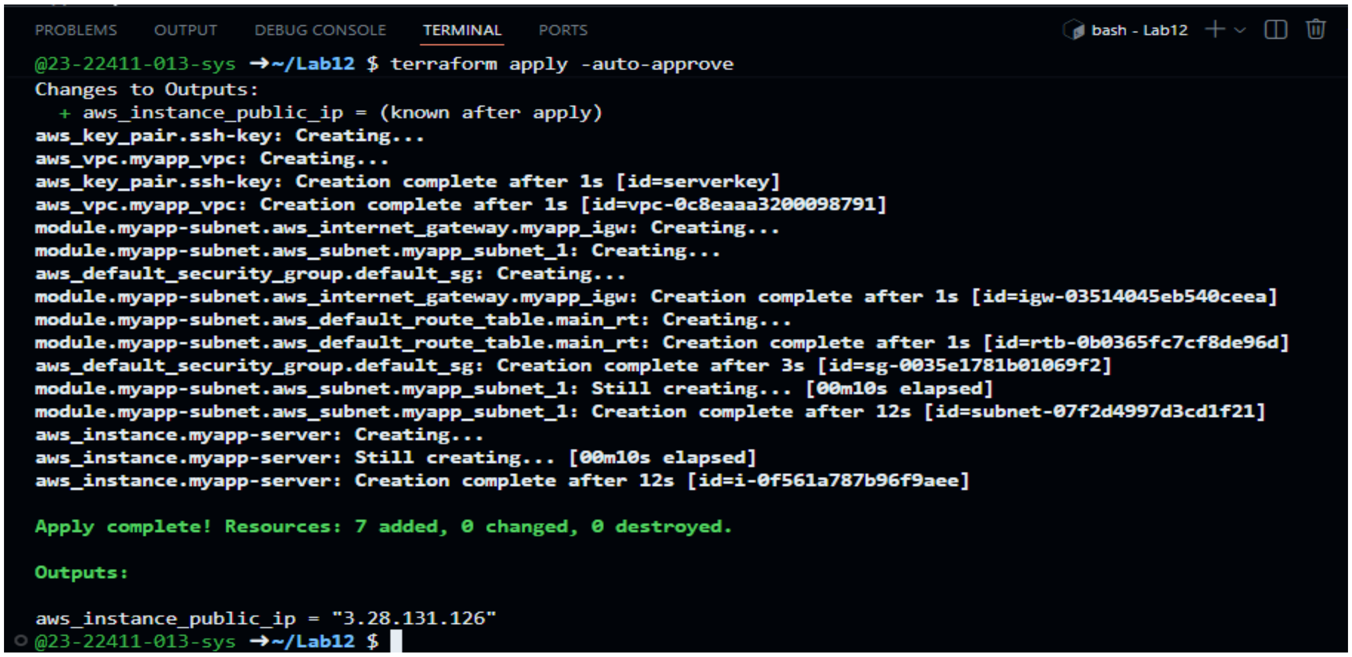
terraform init



```
@23-22411-013-sys →~/Lab12 $ terraform init
Initializing the backend...
Initializing modules...
- myapp-subnet in modules/subnet
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Reusing previous version of hashicorp/http from the dependency lock file
- Using previously-installed hashicorp/aws v6.27.0
- Using previously-installed hashicorp/http v3.5.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
@23-22411-013-sys →~/Lab12 $
```

7. Apply the configuration:

terraform apply -auto-approve

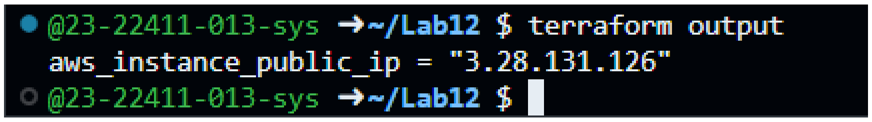


```
@23-22411-013-sys ➜~/Lab12 $ terraform apply -auto-approve
Changes to Outputs:
  + aws_instance_public_ip = (known after apply)
aws_key_pair.ssh-key: Creating...
aws_vpc.myapp_vpc: Creating...
aws_key_pair.ssh-key: Creation complete after 1s [id=serverkey]
aws_vpc.myapp_vpc: Creation complete after 1s [id=vpc-0c8eaaa3200098791]
module.myapp-subnet.aws_internet_gateway.myapp_igw: Creating...
module.myapp-subnet.aws_subnet.myapp_subnet_1: Creating...
aws_default_security_group.default_sg: Creating...
module.myapp-subnet.aws_internet_gateway.myapp_igw: Creation complete after 1s [id=igw-03514045eb540ceea]
module.myapp-subnet.aws_default_route_table.main_rt: Creating...
module.myapp-subnet.aws_default_route_table.main_rt: Creation complete after 1s [id=rtb-0b0365fc7cf8de96d]
aws_default_security_group.default_sg: Creation complete after 3s [id=sg-0035e1781b01069f2]
module.myapp-subnet.aws_subnet.myapp_subnet_1: Still creating... [00m10s elapsed]
module.myapp-subnet.aws_subnet.myapp_subnet_1: Creation complete after 12s [id=subnet-07f2d4997d3cd1f21]
aws_instance.myapp-server: Creating...
aws_instance.myapp-server: Still creating... [00m10s elapsed]
aws_instance.myapp-server: Creation complete after 12s [id=i-0f561a787b96f9aee]

Apply complete! Resources: 7 added, 0 changed, 0 destroyed.

Outputs:

aws_instance_public_ip = "3.28.131.126"
@23-22411-013-sys ➜~/Lab12 $
```

8. Display the output:

terraform output

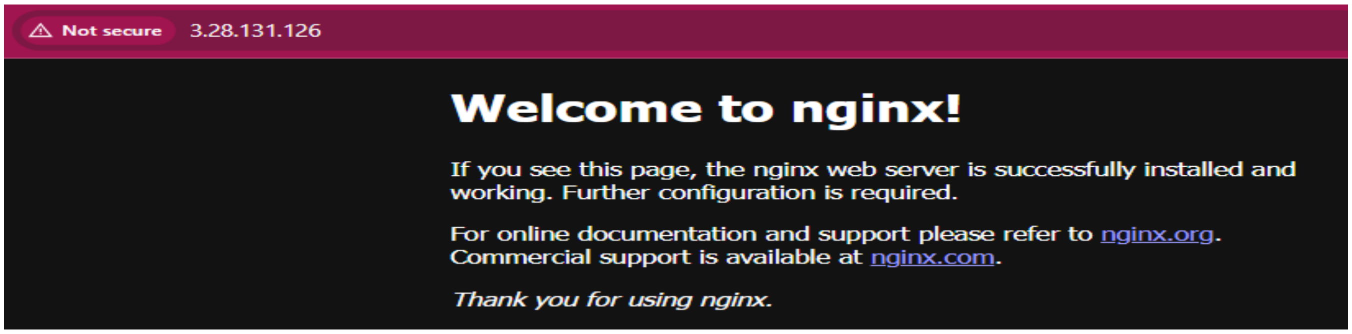
- **Save screenshot as:** task4_terraform_output.png — terraform output showing public IP.



```
@23-22411-013-sys ➜~/Lab12 $ terraform output
aws_instance_public_ip = "3.28.131.126"
@23-22411-013-sys ➜~/Lab12 $
```

9. Test nginx in browser:

- Open browser and navigate to http://<public-ip>



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Task 5 — Create webserver module

In this task, you will create a reusable webserver module for EC2 instances.

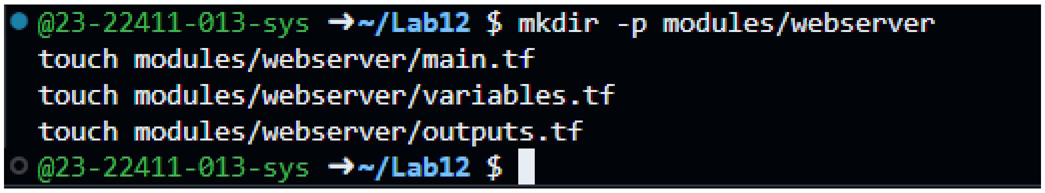
1. Create the webserver module directory structure:

mkdir -p modules/webserver

touch modules/webserver/main.tf

touch modules/webserver/variables.tf

touch modules/webserver/outputs.tf



```
@23-22411-013-sys ➜~/Lab12 $ mkdir -p modules/webserver
touch modules/webserver/main.tf
touch modules/webserver/variables.tf
touch modules/webserver/outputs.tf
@23-22411-013-sys ➜~/Lab12 $ 
```

2. Create modules/webserver/variables.tf:

variable "env_prefix" {}

variable "instance_type" {}

variable "availability_zone" {}

variable "public_key" {}

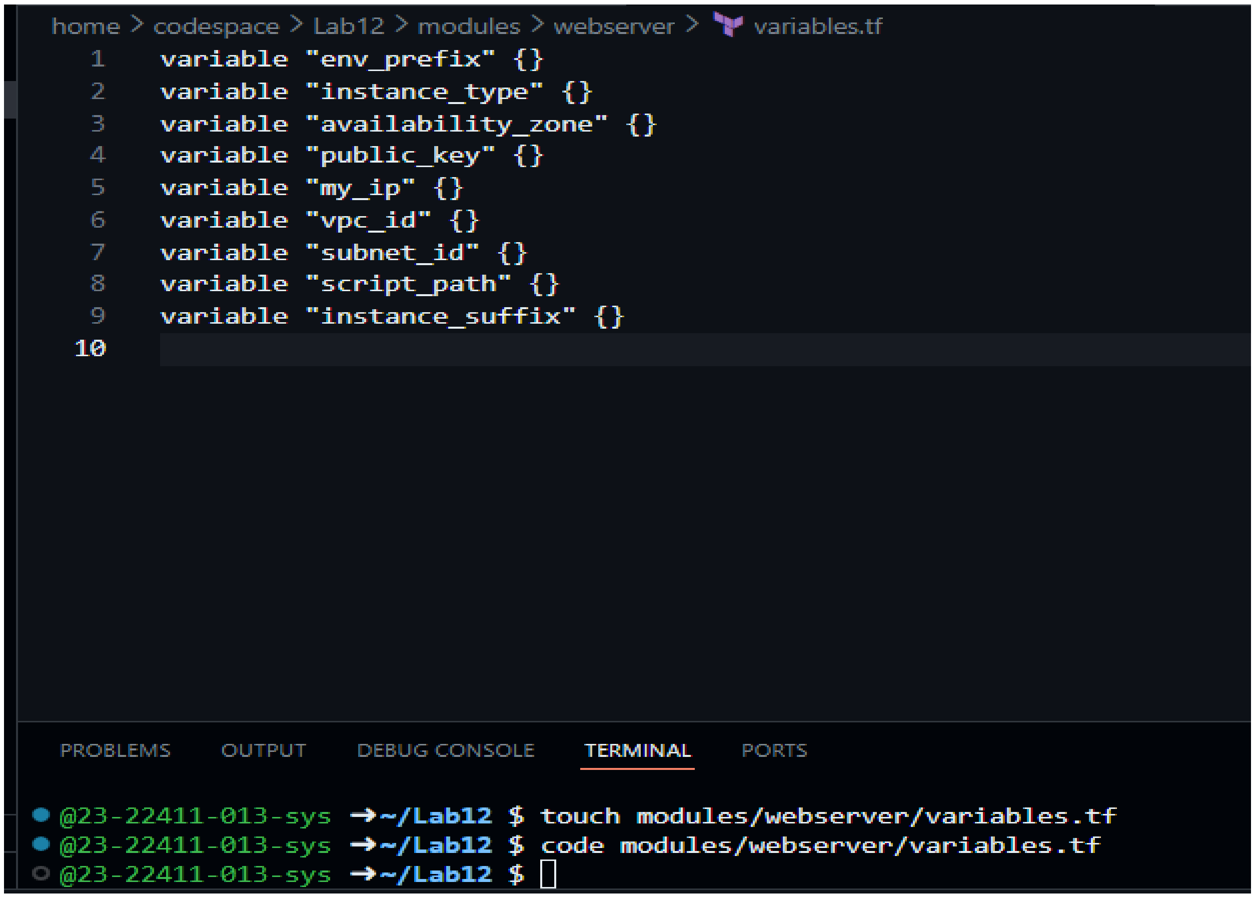
variable "my_ip" {}

variable "vpc_id" {}

variable "subnet_id" {}

variable "script_path" {}

variable "instance_suffix" {}



home > codespace > Lab12 > modules > webserver > variables.tf

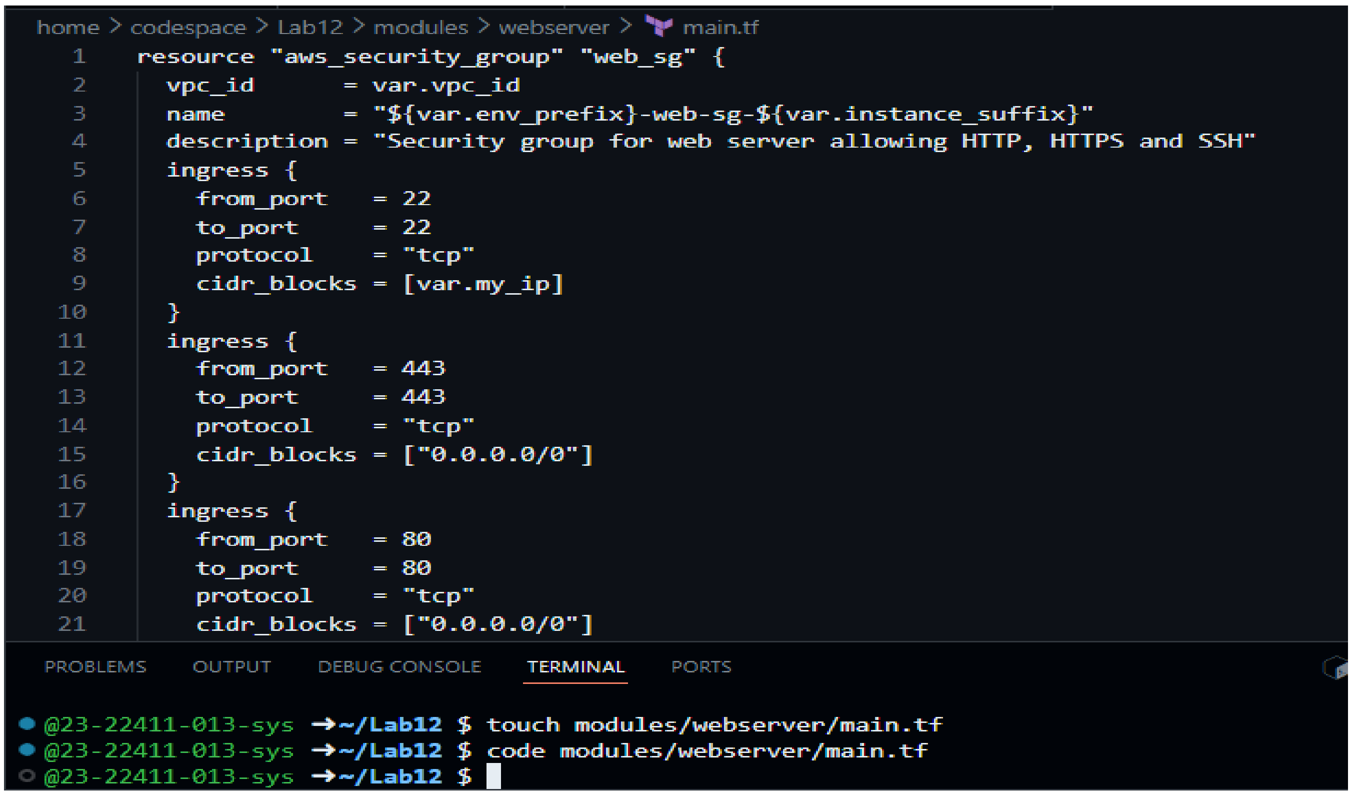
```
variable "env_prefix" {}
variable "instance_type" {}
variable "availability_zone" {}
variable "public_key" {}
variable "my_ip" {}
variable "vpc_id" {}
variable "subnet_id" {}
variable "script_path" {}
variable "instance_suffix" {}
```

```
@23-22411-013-sys ➜~/Lab12 $ touch modules/webserver/variables.tf
@23-22411-013-sys ➜~/Lab12 $ code modules/webserver/variables.tf
@23-22411-013-sys ➜~/Lab12 $ 
```

3. Create modules/webserver/main.tf:

```
resource "aws_security_group" "web_sg" {
  vpc_id      = var.vpc_id
  name        = "${var.env_prefix}-web-sg-${var.instance_suffix}"
  description = "Security group for web server allowing HTTP, HTTPS and SSH"
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = [var.my_ip]
  }
  ingress {
    from_port   = 443
    to_port     = 443
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
    prefix_list_ids = []
}
  tags = {
     Name = "${var.env_prefix}-default-sg"
```

```
 } }
resource "aws_key_pair" "ssh-key" {
  key_name = "${var.env_prefix}-serverkey-${var.instance_suffix}"
  public_key = file(var.public_key)
}
resource "aws_instance" "myapp-server" {
  ami           = "ami-05524d6658fcf35b6" # Amazon Linux 2023 Kernel 6.1 AMI
  instance_type = var. instance_type
  subnet_id     = var.subnet_id
  security_groups = [aws_security_group.web_sg.id]
  availability_zone = var. availability_zone
  associate_public_ip_address = true
  key_name = aws_key_pair.ssh-key.key_name
  user_data = file(var.script_path)
  tags = {
    Name = "${var.env_prefix}-ec2-instance-${var.instance_suffix}"
  }
}
```

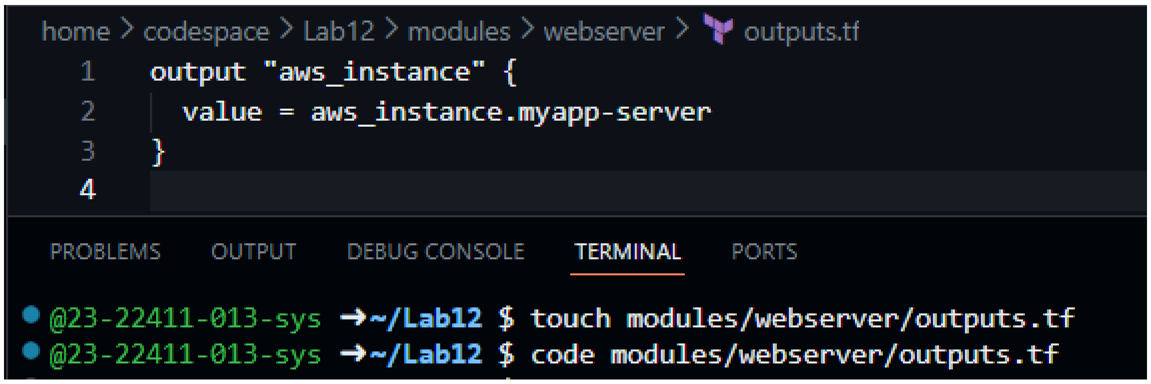


```
resource "aws_security_group" "web_sg" {
  vpc_id      = var.vpc_id
  name        = "${var.env_prefix}-web-sg-${var.instance_suffix}"
  description = "Security group for web server allowing HTTP, HTTPS and SSH"
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = [var.my_ip]
  }
  ingress {
    from_port   = 443
    to_port     = 443
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
```

```
@23-22411-013-sys ➜~/Lab12 $ touch modules/webserver/main.tf
@23-22411-013-sys ➜~/Lab12 $ code modules/webserver/main.tf
@23-22411-013-sys ➜~/Lab12 $ 
```

4. Create modules/webserver/outputs.tf:

```
output "aws_instance" {
  value = aws_instance.myapp-server
}
```



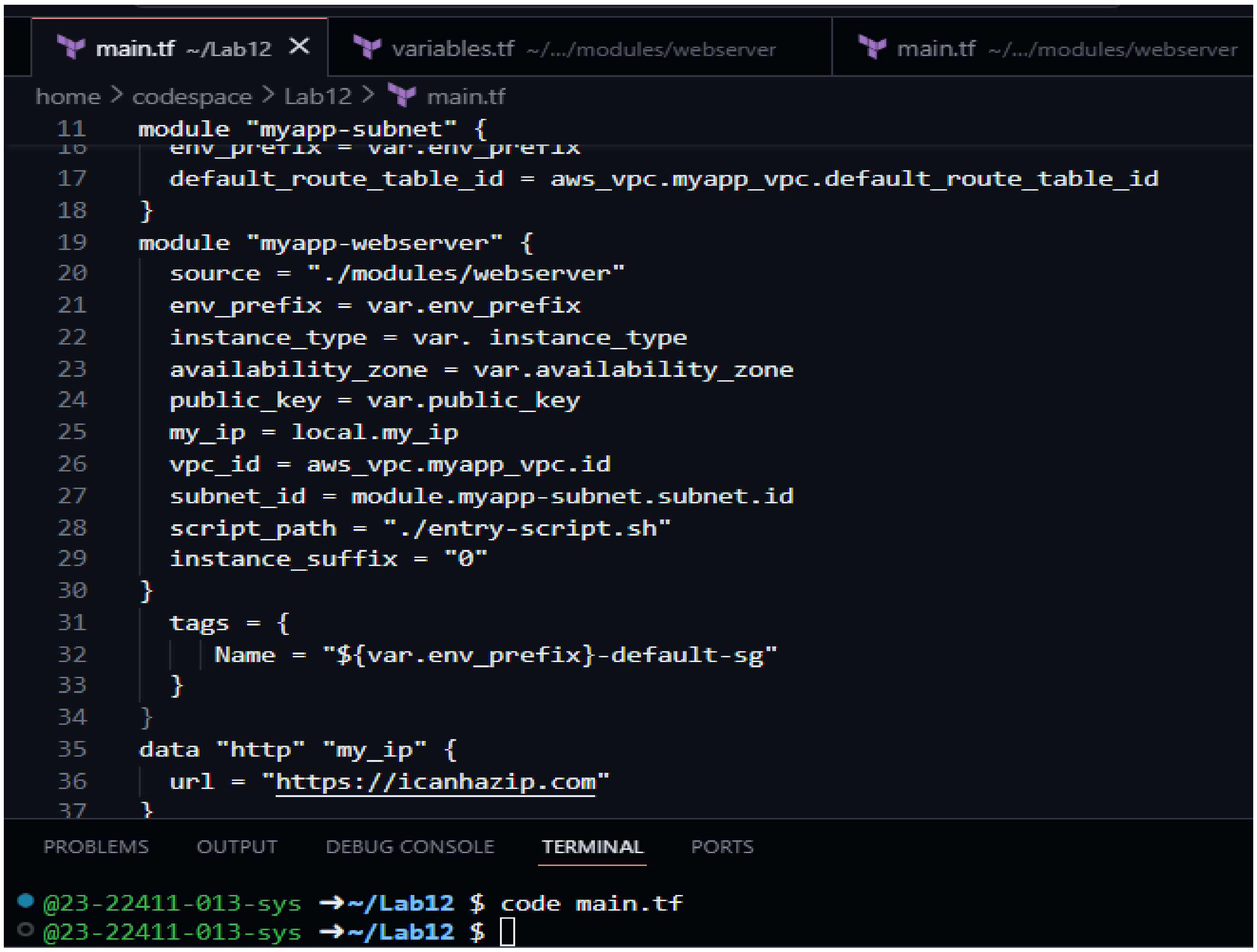
```
output "aws_instance" {
  value = aws_instance.myapp-server
}
```

```
@23-22411-013-sys ➜~/Lab12 $ touch modules/webserver/outputs.tf
@23-22411-013-sys ➜~/Lab12 $ code modules/webserver/outputs.tf
```

5. Modify the root main.tf:

Remove the security group, key pair, and instance resources. Replace them with:

```
module "myapp-webserver" {
  source = "./modules/webserver"
  env_prefix = var.env_prefix
  instance_type = var. instance_type
  availability_zone = var.availability_zone
  public_key = var.public_key
  my_ip = local.my_ip
  vpc_id = aws_vpc.myapp_vpc.id
  subnet_id = module.myapp-subnet.subnet.id
  script_path = "./entry-script.sh"
  instance_suffix = "0"
}
```


main.tf ~/Lab12
variables.tf ~/.../modules/webserver
main.tf ~/.../modules/webserver
home > codespace > Lab12 > main.tf
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS

```
module "myapp-subnet" {
  env_prefix = var.env_prefix
  default_route_table_id = aws_vpc.myapp_vpc.default_route_table_id
}
module "myapp-webserver" {
  source = "./modules/webserver"
  env_prefix = var.env_prefix
  instance_type = var. instance_type
  availability_zone = var.availability_zone
  public_key = var.public_key
  my_ip = local.my_ip
  vpc_id = aws_vpc.myapp_vpc.id
  subnet_id = module.myapp-subnet.subnet.id
  script_path = "./entry-script.sh"
  instance_suffix = "0"
}
  tags = {
     Name = "${var.env_prefix}-default-sg"
  }
}
data "http" "my_ip" {
  url = "https://icanhazip.com"
}
```

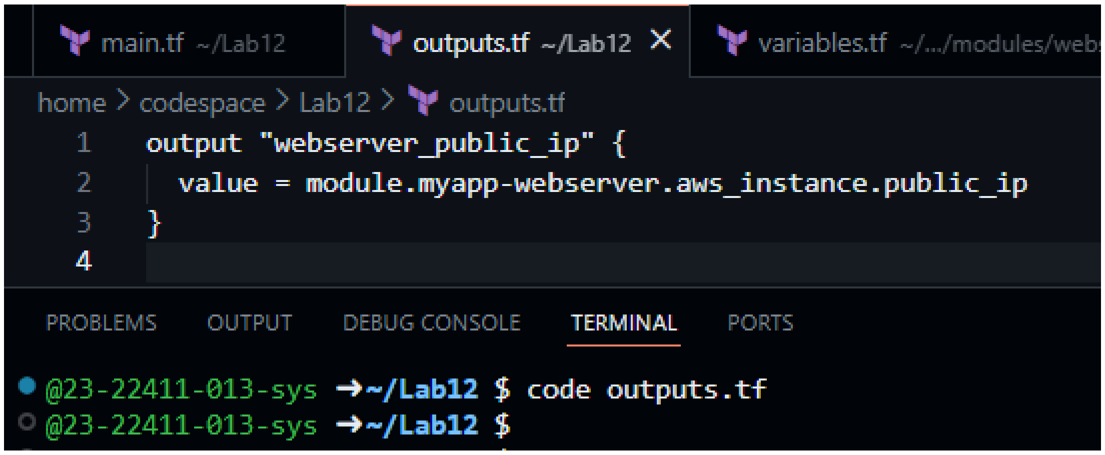
```
@23-22411-013-sys ➜~/Lab12 $ code main.tf
@23-22411-013-sys ➜~/Lab12 $ 
```

6. Update outputs.tf:

output "webserver_public_ip" {

 value = module.myapp-webserver.aws_instance.public_ip

}

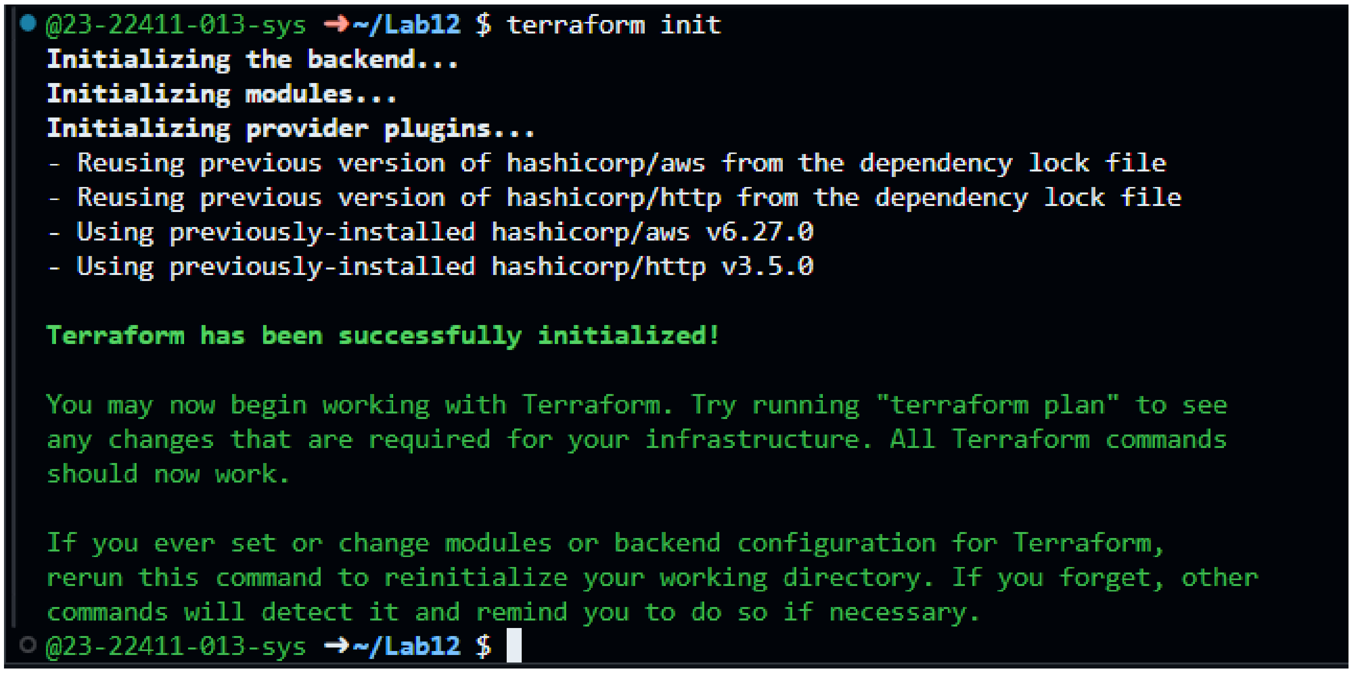


```
output "webserver_public_ip" {
  value = module.myapp-webserver.aws_instance.public_ip
}

```

```
@23-22411-013-sys ➜~/Lab12 $ code outputs.tf
@23-22411-013-sys ➜~/Lab12 $
```

7. Initialize Terraform:

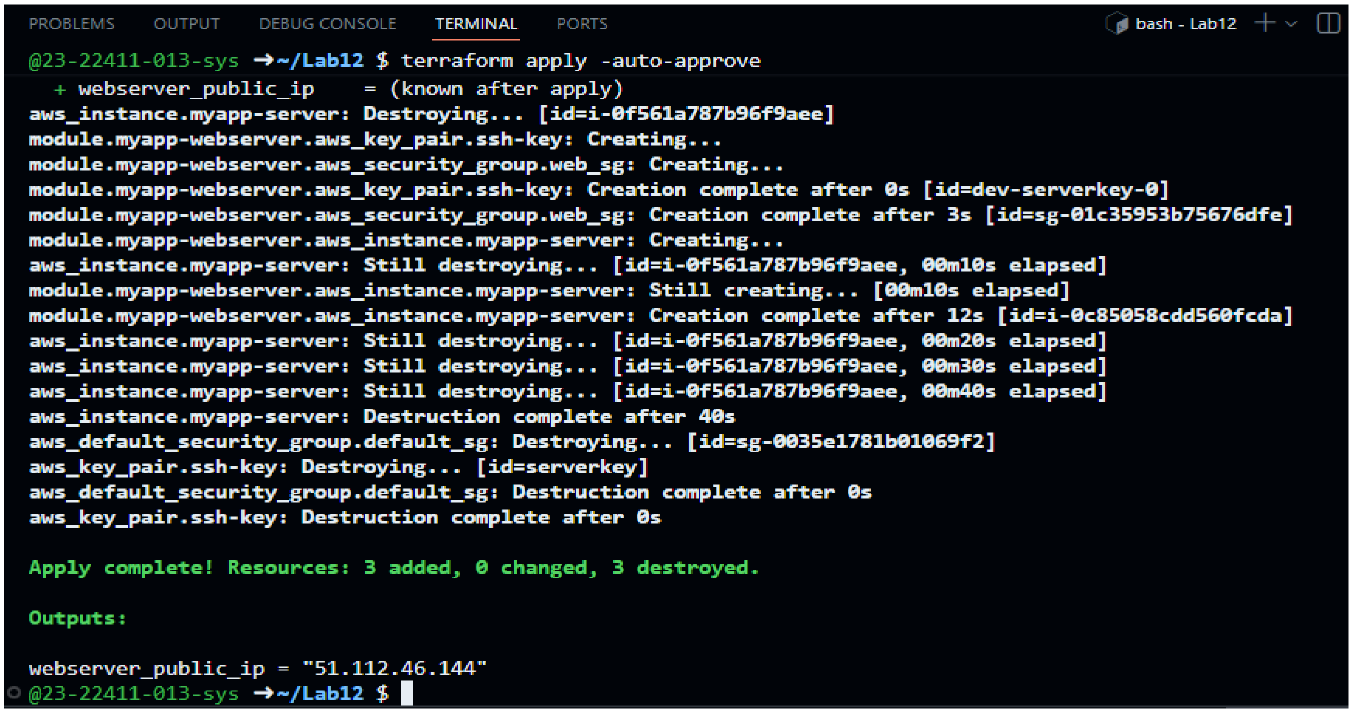
terraform init



```
@23-22411-013-sys ➜~/Lab12 $ terraform init
Initializing the backend...
Initializing modules...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Reusing previous version of hashicorp/http from the dependency lock file
- Using previously-installed hashicorp/aws v6.27.0
- Using previously-installed hashicorp/http v3.5.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
@23-22411-013-sys ➜~/Lab12 $
```

8. Apply the configuration:

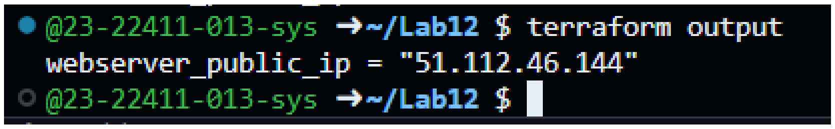
terraform apply -auto-approve



```
@23-22411-013-sys ➜~/Lab12 $ terraform apply -auto-approve
  + webserver_public_ip    = (known after apply)
aws_instance.myapp-server: Destroying... [id=i-0f561a787b96f9aee]
module.myapp-webserver.aws_key_pair.ssh-key: Creating...
module.myapp-webserver.aws_security_group.web_sg: Creating...
module.myapp-webserver.aws_key_pair.ssh-key: Creation complete after 0s [id=dev-serverkey-0]
module.myapp-webserver.aws_security_group.web_sg: Creation complete after 3s [id=sg-01c35953b75676dfe]
module.myapp-webserver.aws_instance.myapp-server: Creating...
aws_instance.myapp-server: Still destroying... [id=i-0f561a787b96f9aee, 00m10s elapsed]
module.myapp-webserver.aws_instance.myapp-server: Still creating... [00m10s elapsed]
module.myapp-webserver.aws_instance.myapp-server: Creation complete after 12s [id=i-0c85058cdd560fcda]
aws_instance.myapp-server: Still destroying... [id=i-0f561a787b96f9aee, 00m20s elapsed]
aws_instance.myapp-server: Still destroying... [id=i-0f561a787b96f9aee, 00m30s elapsed]
aws_instance.myapp-server: Still destroying... [id=i-0f561a787b96f9aee, 00m40s elapsed]
aws_instance.myapp-server: Destruction complete after 40s
aws_default_security_group.default_sg: Destroying... [id=sg-0035e1781b01069f2]
aws_key_pair.ssh-key: Destroying... [id=serverkey]
aws_default_security_group.default_sg: Destruction complete after 0s
aws_key_pair.ssh-key: Destruction complete after 0s

Apply complete! Resources: 3 added, 0 changed, 3 destroyed.

Outputs:

webserver_public_ip = "51.112.46.144"
@23-22411-013-sys ➜~/Lab12 $
```

9. Display the output:

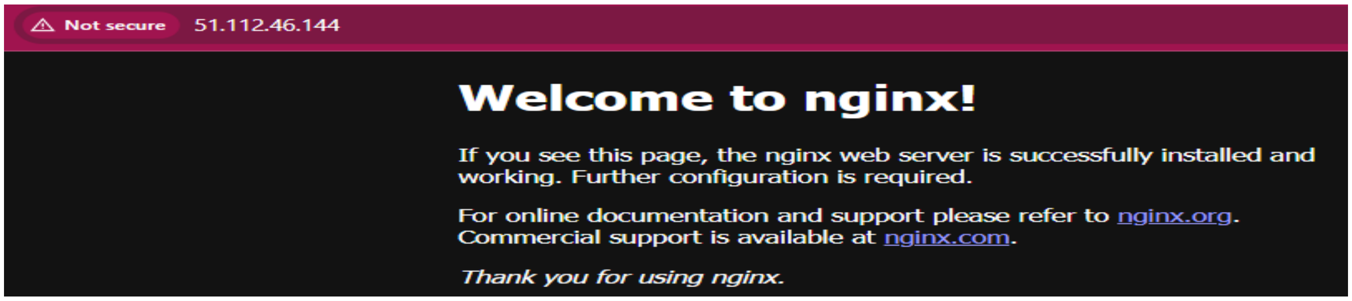
terraform output



```
@23-22411-013-sys ➜~/Lab12 $ terraform output
webserver_public_ip = "51.112.46.144"
@23-22411-013-sys ➜~/Lab12 $
```

10. Test nginx in browser:

- Open browser and navigate to http://<public-ip>



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

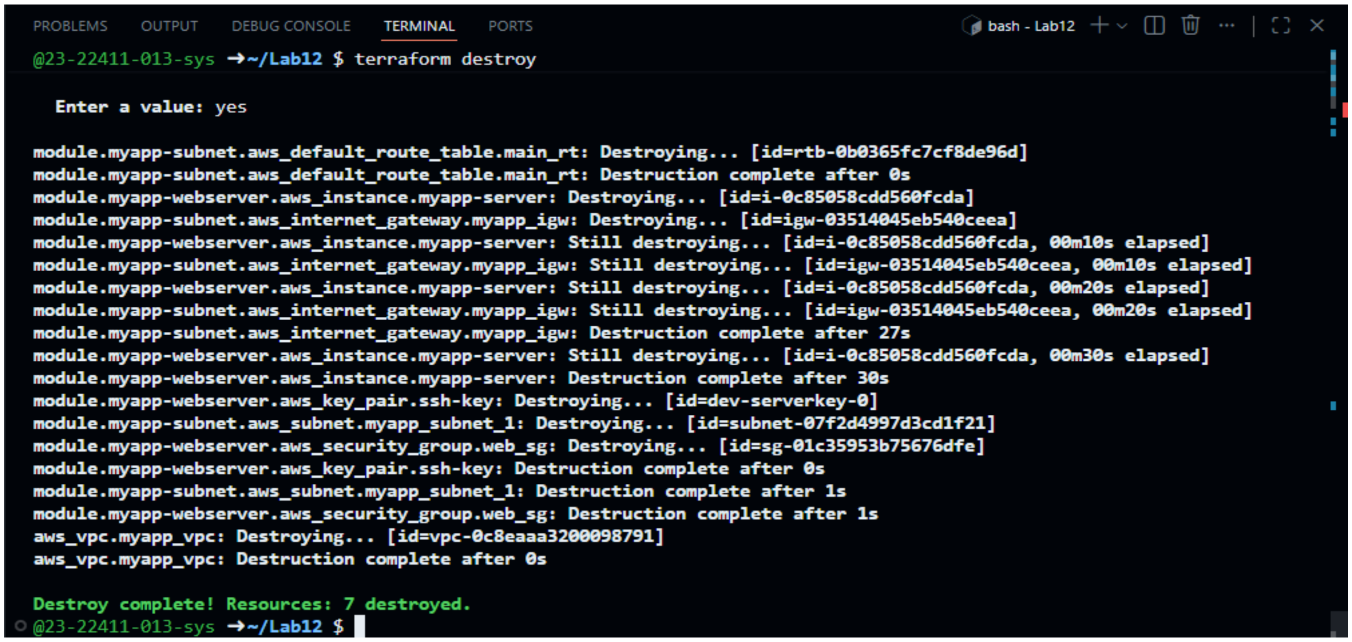
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

11. Destroy resources:

terraform destroy

- Type yes when prompted.



```
@23-22411-013-sys →~/Lab12 $ terraform destroy

  Enter a value: yes

module.myapp-subnet.aws_default_route_table.main_rt: Destroying... [id=rtb-0b0365fc7cf8de96d]
module.myapp-subnet.aws_default_route_table.main_rt: Destruction complete after 0s
module.myapp-webserver.aws_instance.myapp-server: Destroying... [id=i-0c85058cdd560fcda]
module.myapp-subnet.aws_internet_gateway.myapp_igw: Destroying... [id=igw-03514045eb540ceea]
module.myapp-webserver.aws_instance.myapp-server: Still destroying... [id=i-0c85058cdd560fcda, 00m10s elapsed]
module.myapp-subnet.aws_internet_gateway.myapp_igw: Still destroying... [id=igw-03514045eb540ceea, 00m10s elapsed]
module.myapp-webserver.aws_instance.myapp-server: Still destroying... [id=i-0c85058cdd560fcda, 00m20s elapsed]
module.myapp-subnet.aws_internet_gateway.myapp_igw: Still destroying... [id=igw-03514045eb540ceea, 00m20s elapsed]
module.myapp-subnet.aws_internet_gateway.myapp_igw: Destruction complete after 27s
module.myapp-webserver.aws_instance.myapp-server: Still destroying... [id=i-0c85058cdd560fcda, 00m30s elapsed]
module.myapp-webserver.aws_instance.myapp-server: Destruction complete after 30s
module.myapp-webserver.aws_key_pair.ssh-key: Destroying... [id=dev-serverkey-0]
module.myapp-subnet.aws_subnet.myapp_subnet_1: Destroying... [id=subnet-07f2d4997d3cd1f21]
module.myapp-webserver.aws_security_group.web_sg: Destroying... [id=sg-01c35953b75676dfe]
module.myapp-webserver.aws_key_pair.ssh-key: Destruction complete after 0s
module.myapp-subnet.aws_subnet.myapp_subnet_1: Destruction complete after 1s
module.myapp-webserver.aws_security_group.web_sg: Destruction complete after 1s
aws_vpc.myapp_vpc: Destroying... [id=vpc-0c8eaaa3200098791]
aws_vpc.myapp_vpc: Destruction complete after 0s

Destroy complete! Resources: 7 destroyed.
@23-22411-013-sys →~/Lab12 $
```

Task 6 — Configure HTTPS with self-signed certificates

In this task, you will configure Nginx to serve traffic over HTTPS using self-signed certificates.

1. Update entry-script.sh with SSL configuration:

#!/bin/bash

set -e

yum update -y

yum install -y nginx

systemctl start nginx

systemctl enable nginx

```
# Create directories for SSL certificates if they don't exist
mkdir -p /etc/ssl/private
mkdir -p /etc/ssl/certs
# Get IMDSv2 token
TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/token" \
  -H "X-aws-ec2-metadata-token-ttl-seconds: 21600")
# Get current public IP
PUBLIC_IP=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \
  http://169.254.169.254/latest/meta-data/public-ipv4)
PUBLIC_HOSTNAME=$(curl -s -H "X-aws-ec2-metadata-token:  $TOKEN" \
  http://169.254.169.254/latest/meta-data/public-hostname)
# Generate self-signed certificate with dynamic IP
openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
  -keyout /etc/ssl/private/selfsigned.key \
  -out /etc/ssl/certs/selfsigned.crt \
  -subj "/CN=$PUBLIC_IP" \
  -addext "subjectAltName=IP:$PUBLIC_IP" \
  -addext "basicConstraints=CA:FALSE" \
  -addext "keyUsage=digitalSignature,keyEncipherment" \
  -addext "extendedKeyUsage=serverAuth"
echo "Self-signed certificate created for IP: $PUBLIC_IP"
# Backup existing nginx. conf
cp /etc/nginx/nginx.conf /etc/nginx/nginx.conf.bak
# Overwrite nginx.conf with the desired content
cat <<EOF > /etc/nginx/nginx.conf
user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log notice;
pid /run/nginx. pid;
events {
    worker_connections 1024;
}
```

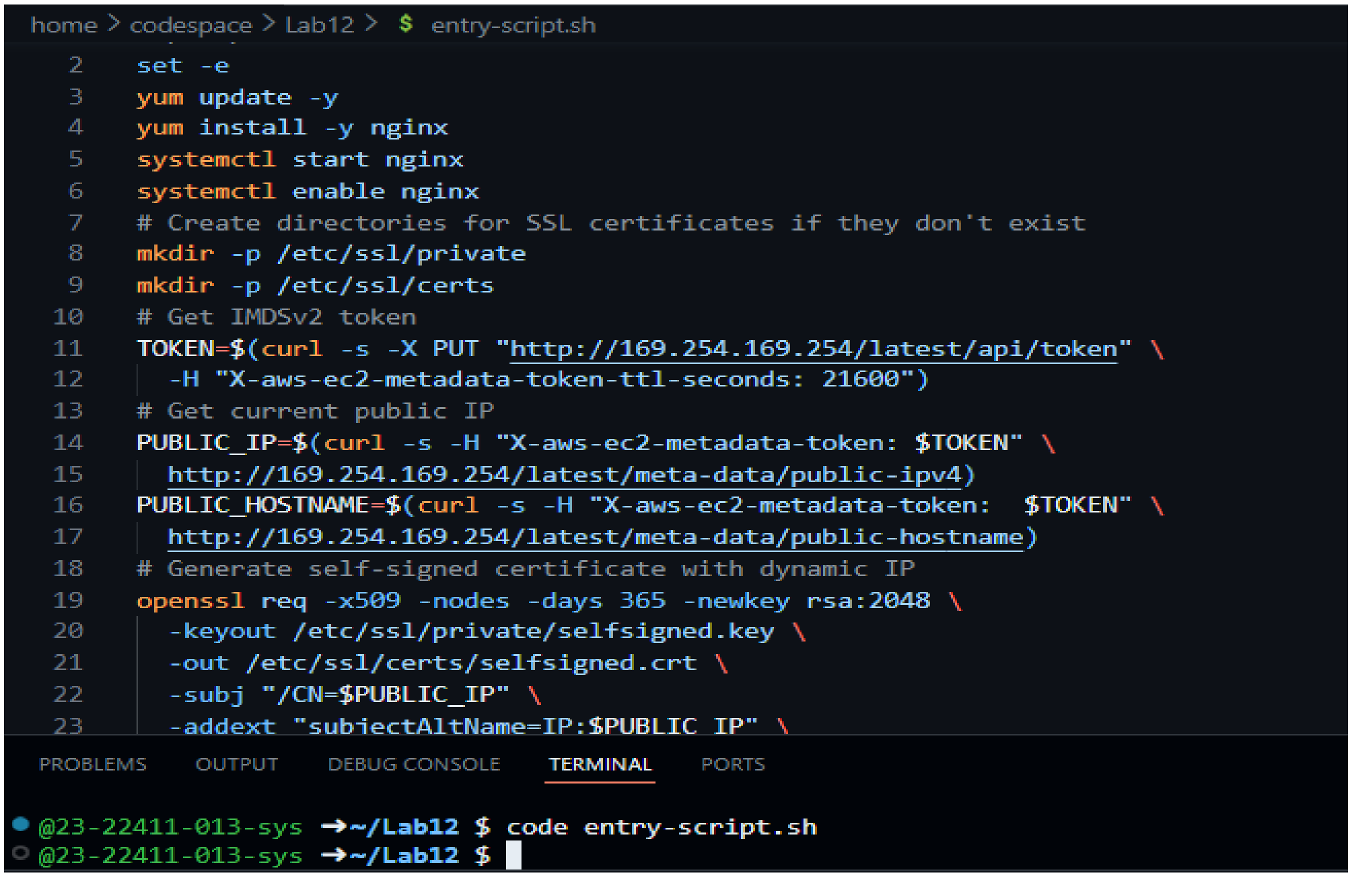
```
http {
    log_format  main  '\$remote_addr - \$remote_user [\$time_local] "\$request" '
                      '\$status \$body_bytes_sent "\$http_referer" '
                      '"\$http_user_agent" "\$http_x_forwarded_for"';
    access_log  /var/log/nginx/access.log  main;
    sendfile            on;
    tcp_nopush          on;
    keepalive_timeout   65;
    types_hash_max_size 4096;
    include             /etc/nginx/mime.types;
    default_type        application/octet-stream;
    upstream backend_servers {
        server 158.252.94.241:80;
        server 158.252.94.242:80 backup;
    }
    server {
        listen 443 ssl;
        server_name $PUBLIC_IP;
        ssl_certificate /etc/ssl/certs/selfsigned. crt;
        ssl_certificate_key /etc/ssl/private/selfsigned.key;
        location / {
            root /usr/share/nginx/html;
            index index.html;
    #       proxy_pass http://158.252.94.241:80;
    #       proxy_pass http://backend_servers;
        }
    }
    server {
        listen 80;
        server_name _;
        return 301 https://\$host\$request_uri;
    }
```

}

EOF

# Test and restart Nginx

systemctl restart nginx



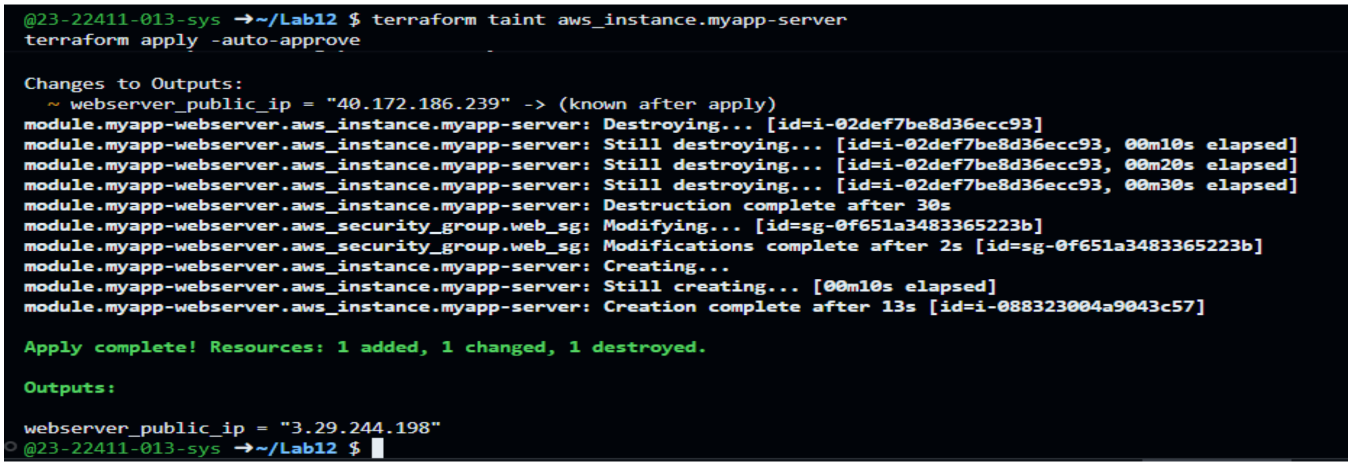
```
home > codespace > Lab12 > $ entry-script.sh
set -e
yum update -y
yum install -y nginx
systemctl start nginx
systemctl enable nginx
# Create directories for SSL certificates if they don't exist
mkdir -p /etc/ssl/private
mkdir -p /etc/ssl/certs
# Get IMDSv2 token
TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/token" \
  -H "X-aws-ec2-metadata-token-ttl-seconds: 21600")
# Get current public IP
PUBLIC_IP=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \
  http://169.254.169.254/latest/meta-data/public-ipv4)
PUBLIC_HOSTNAME=$(curl -s -H "X-aws-ec2-metadata-token:  $TOKEN" \
  http://169.254.169.254/latest/meta-data/public-hostname)
# Generate self-signed certificate with dynamic IP
openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
  -keyout /etc/ssl/private/selfsigned.key \
  -out /etc/ssl/certs/selfsigned.crt \
  -subj "/CN=$PUBLIC_IP" \
  -addext "subjectAltName=IP:$PUBLIC_IP" \
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
@23-22411-013-sys →~/Lab12 $ code entry-script.sh
@23-22411-013-sys →~/Lab12 $
```

2. Apply the configuration:

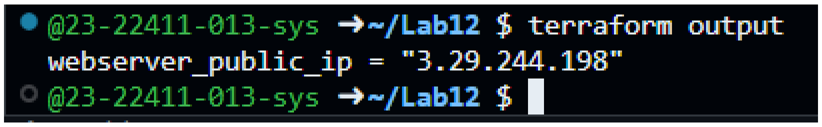
terraform apply -auto-approve



```
@23-22411-013-sys →~/Lab12 $ terraform taint aws_instance.myapp-server
terraform apply -auto-approve

Changes to Outputs:
  ~ webserver_public_ip = "40.172.186.239" -> (known after apply)
module.myapp-webserver.aws_instance.myapp-server: Destroying... [id=i-02def7be8d36ecc93]
module.myapp-webserver.aws_instance.myapp-server: Still destroying... [id=i-02def7be8d36ecc93, 00m10s elapsed]
module.myapp-webserver.aws_instance.myapp-server: Still destroying... [id=i-02def7be8d36ecc93, 00m20s elapsed]
module.myapp-webserver.aws_instance.myapp-server: Still destroying... [id=i-02def7be8d36ecc93, 00m30s elapsed]
module.myapp-webserver.aws_instance.myapp-server: Destruction complete after 30s
module.myapp-webserver.aws_security_group.web_sg: Modifying... [id=sg-0f651a3483365223b]
module.myapp-webserver.aws_security_group.web_sg: Modifications complete after 2s [id=sg-0f651a3483365223b]
module.myapp-webserver.aws_instance.myapp-server: Creating...
module.myapp-webserver.aws_instance.myapp-server: Still creating... [00m10s elapsed]
module.myapp-webserver.aws_instance.myapp-server: Creation complete after 13s [id=i-088323004a9043c57]

Apply complete! Resources: 1 added, 1 changed, 1 destroyed.

Outputs:

webserver_public_ip = "3.29.244.198"
@23-22411-013-sys →~/Lab12 $
```

3. Display the output:

terraform output

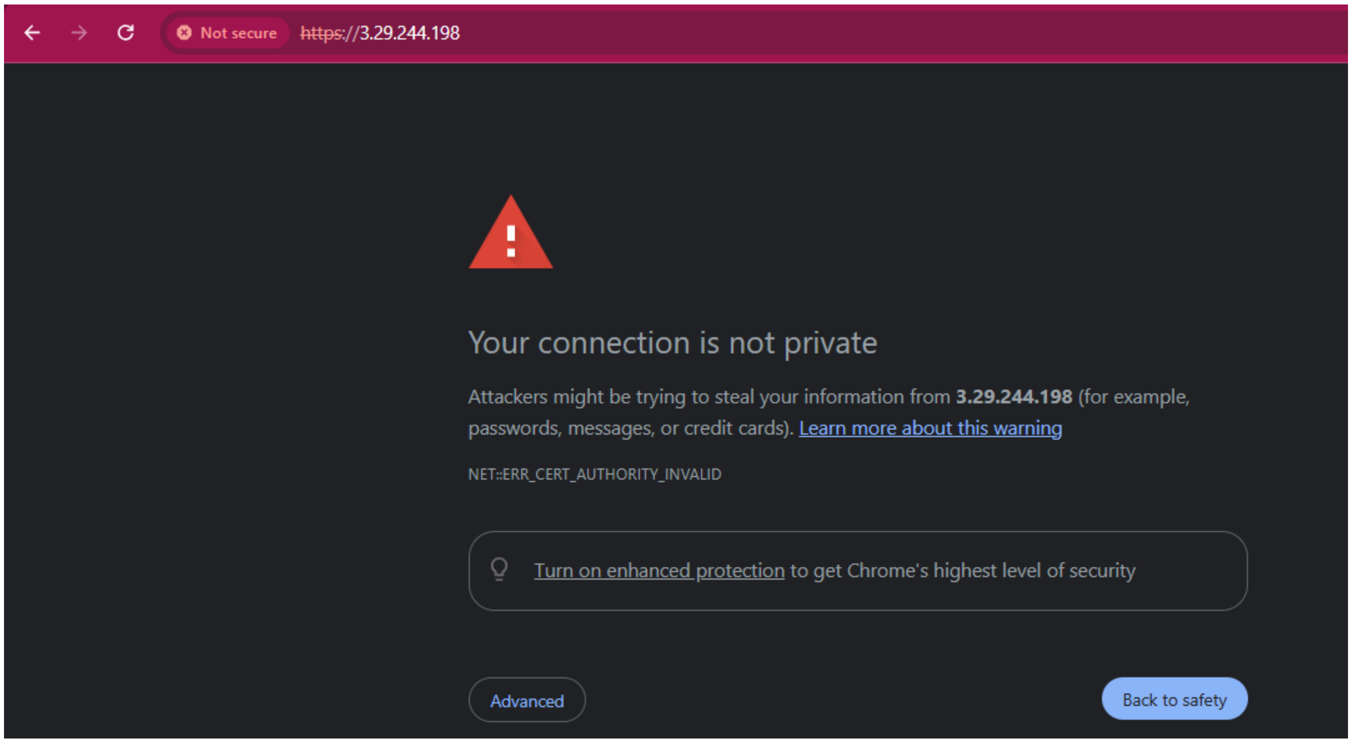


```
@23-22411-013-sys ➜~/Lab12 $ terraform output
webserver_public_ip = "3.29.244.198"
@23-22411-013-sys ➜~/Lab12 $
```

4. Test HTTPS in browser:

Open browser and navigate to https://<public-ip>

You will see a warning: "Warning: Potential Security Risk Ahead"

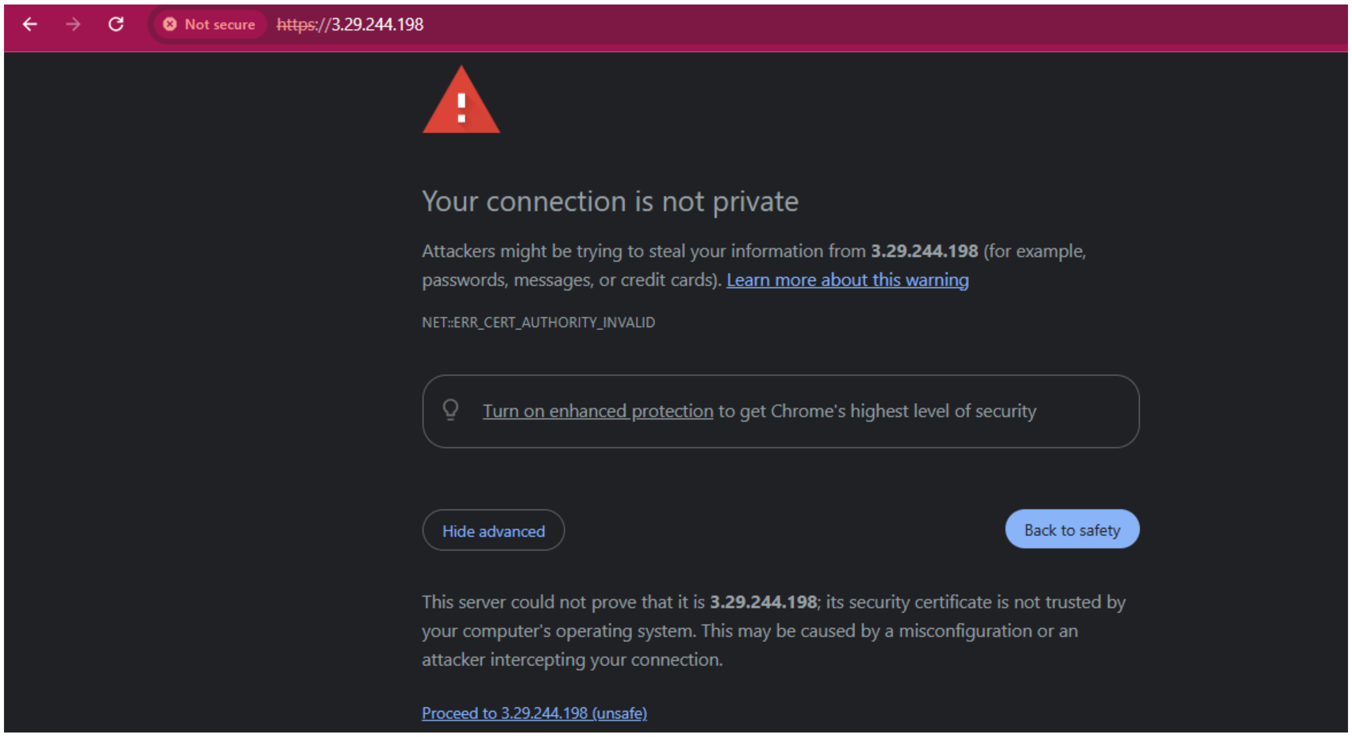


Your connection is not private

Attackers might be trying to steal your information from **3.29.244.198** (for example, passwords, messages, or credit cards). Learn more about this warning

NET::ERR_CERT_AUTHORITY_INVALID

Turn on enhanced protection to get Chrome's highest level of security



Your connection is not private

Attackers might be trying to steal your information from **3.29.244.198** (for example, passwords, messages, or credit cards). Learn more about this warning

NET::ERR_CERT_AUTHORITY_INVALID

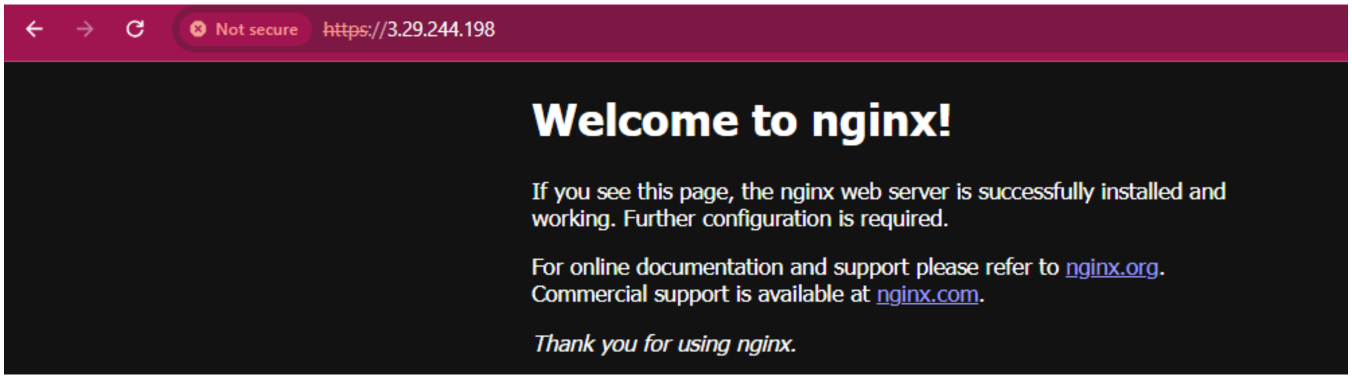
Turn on enhanced protection to get Chrome's highest level of security

This server could not prove that it is **3.29.244.198**; its security certificate is not trusted by your computer's operating system. This may be caused by a misconfiguration or an attacker intercepting your connection.

Proceed to 3.29.244.198 (unsafe)

Click "Advanced" button

Click "Accept the Risk and Continue"



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

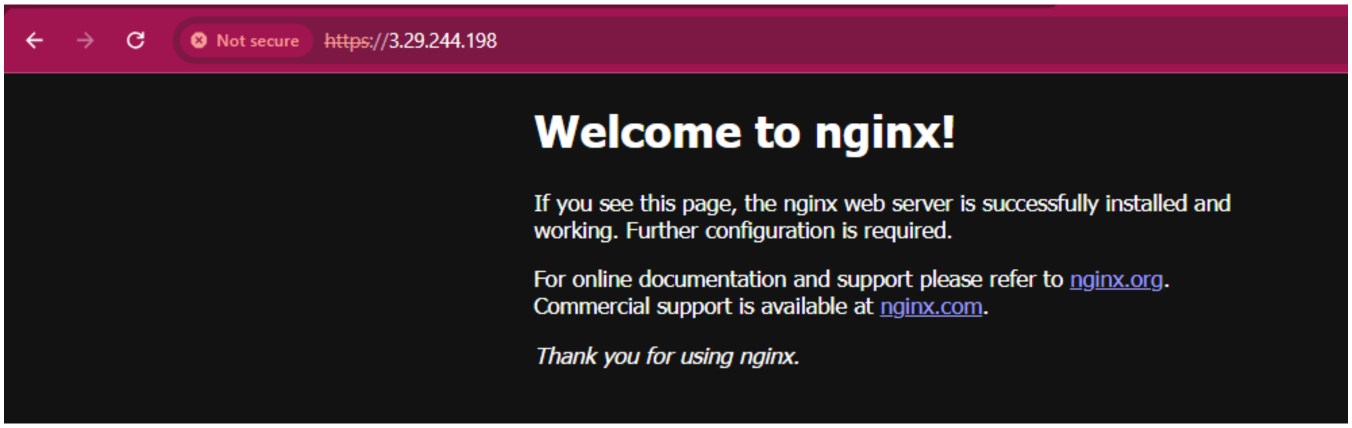
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

5. Verify HTTP to HTTPS redirect:

Open browser and navigate to http://<public-ip>

Verify it redirects to https://<public-ip>



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

When accessing the server using HTTP, the request is automatically redirected to HTTPS, confirming successful HTTP to HTTPS redirection

---

Task 7 — Configure Nginx as reverse proxy

In this task, you will create a backend web server and configure Nginx to act as a reverse proxy.

1. Create apache.sh script for backend web server:

```
#!/bin/bash

yum update -y

yum install httpd -y

systemctl start httpd

systemctl enable httpd

echo "<h1>Welcome to My Web Server</h1>" > /var/www/html/index.html
```

hostnamectl set-hostname myapp-webserver

echo "<h2>Hostname: $(hostname)</h2>" >> /var/www/html/index.html

TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/token" \

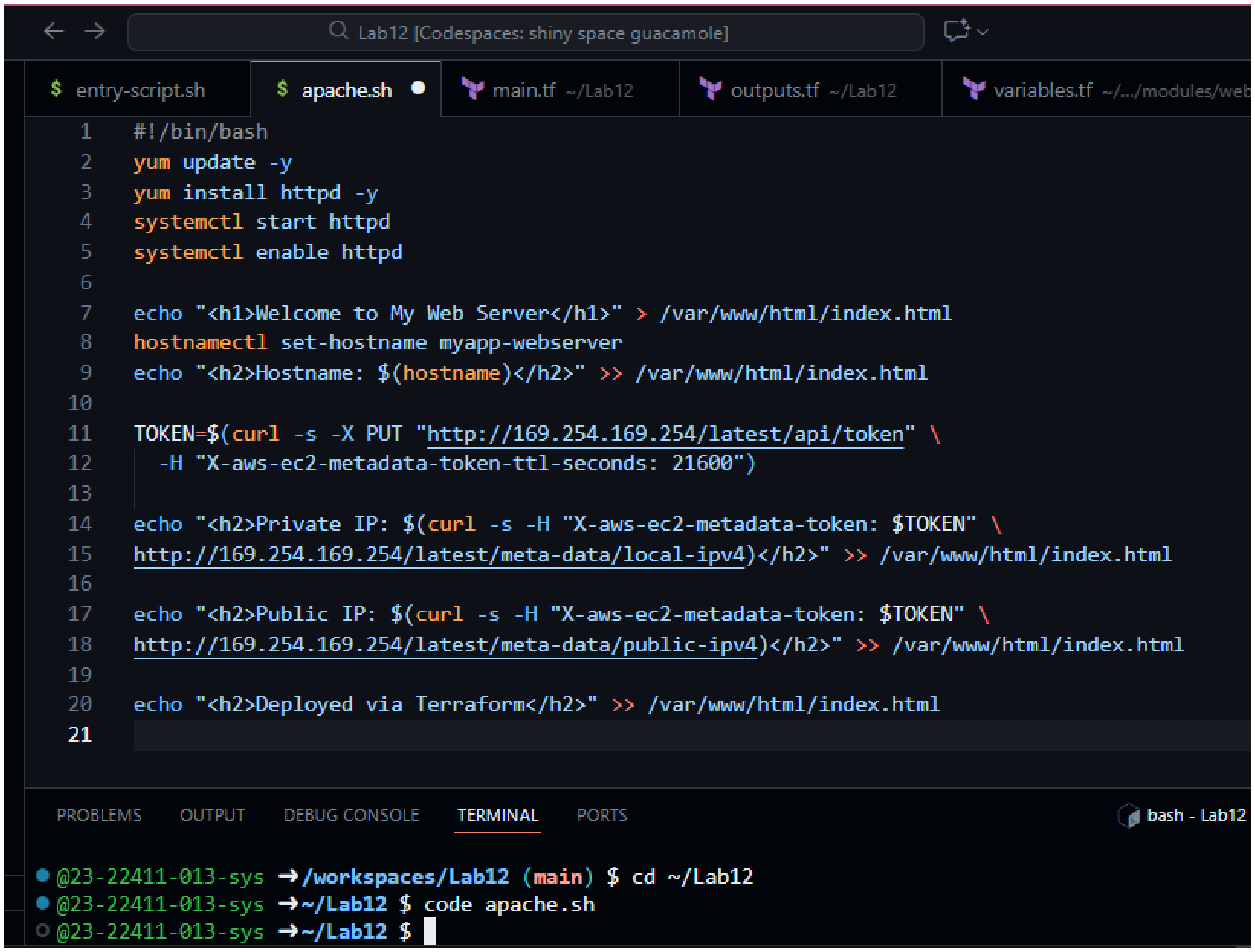
  -H "X-aws-ec2-metadata-token-ttl-seconds: 21600")

echo "<h2>Private IP: $(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/local-ipv4)</h2>" >> /var/www/html/index.html

echo "<h2>Public IP: $(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/public-ipv4)</h2>" >> /var/www/html/index.html

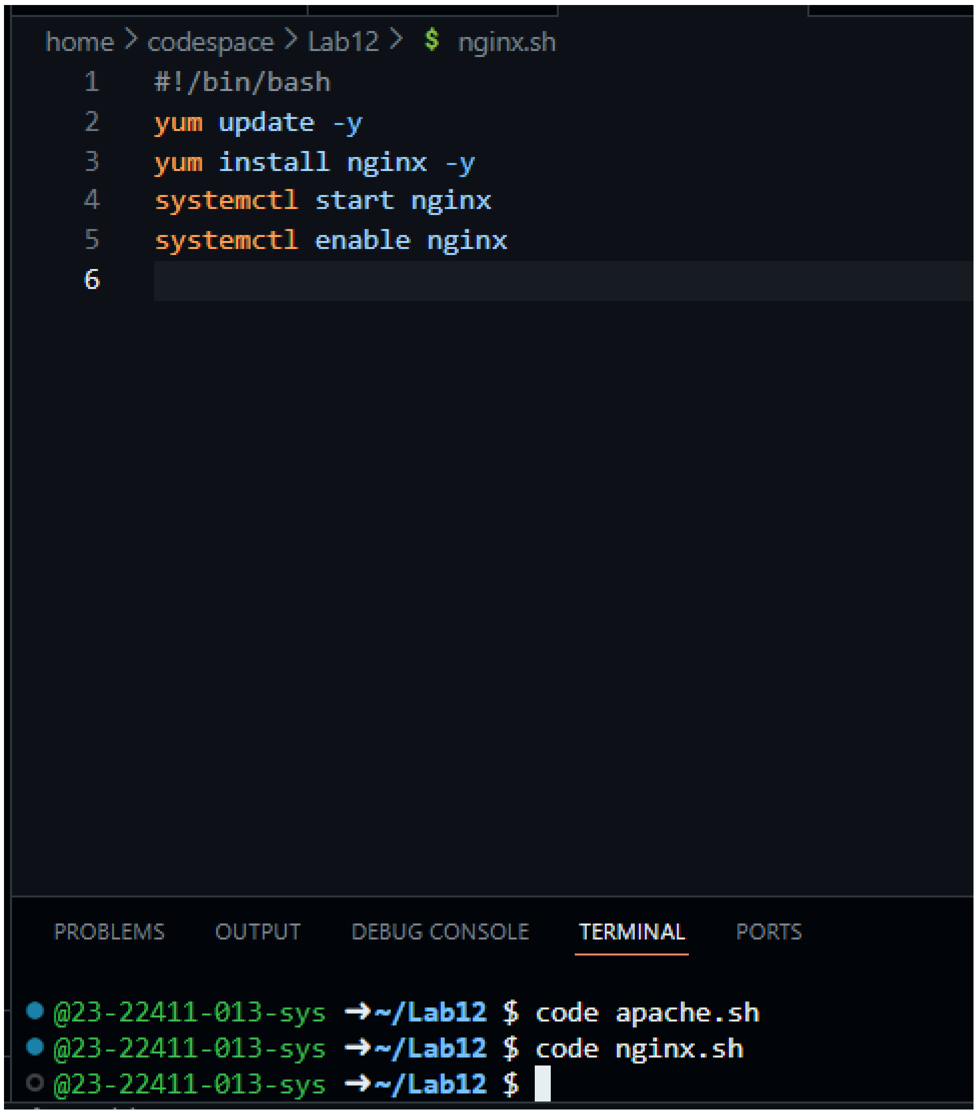
echo "<h2>Public DNS: $(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/public-hostname)</h2>" >> /var/www/html/index.html

echo "<h2>Deployed via Terraform</h2>" >> /var/www/html/index. html



```
#!/bin/bash
yum update -y
yum install httpd -y
systemctl start httpd
systemctl enable httpd

echo "<h1>Welcome to My Web Server</h1>" > /var/www/html/index.html
hostnamectl set-hostname myapp-webserver
echo "<h2>Hostname: $(hostname)</h2>" >> /var/www/html/index.html

TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/token" \
  -H "X-aws-ec2-metadata-token-ttl-seconds: 21600")

echo "<h2>Private IP: $(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \
http://169.254.169.254/latest/meta-data/local-ipv4)</h2>" >> /var/www/html/index.html

echo "<h2>Public IP: $(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \
http://169.254.169.254/latest/meta-data/public-ipv4)</h2>" >> /var/www/html/index.html

echo "<h2>Deployed via Terraform</h2>" >> /var/www/html/index.html

```

```
@23-22411-013-sys ➜ /workspaces/Lab12 (main) $ cd ~/Lab12
@23-22411-013-sys ➜ ~/Lab12 $ code apache.sh
@23-22411-013-sys ➜ ~/Lab12 $ 
```



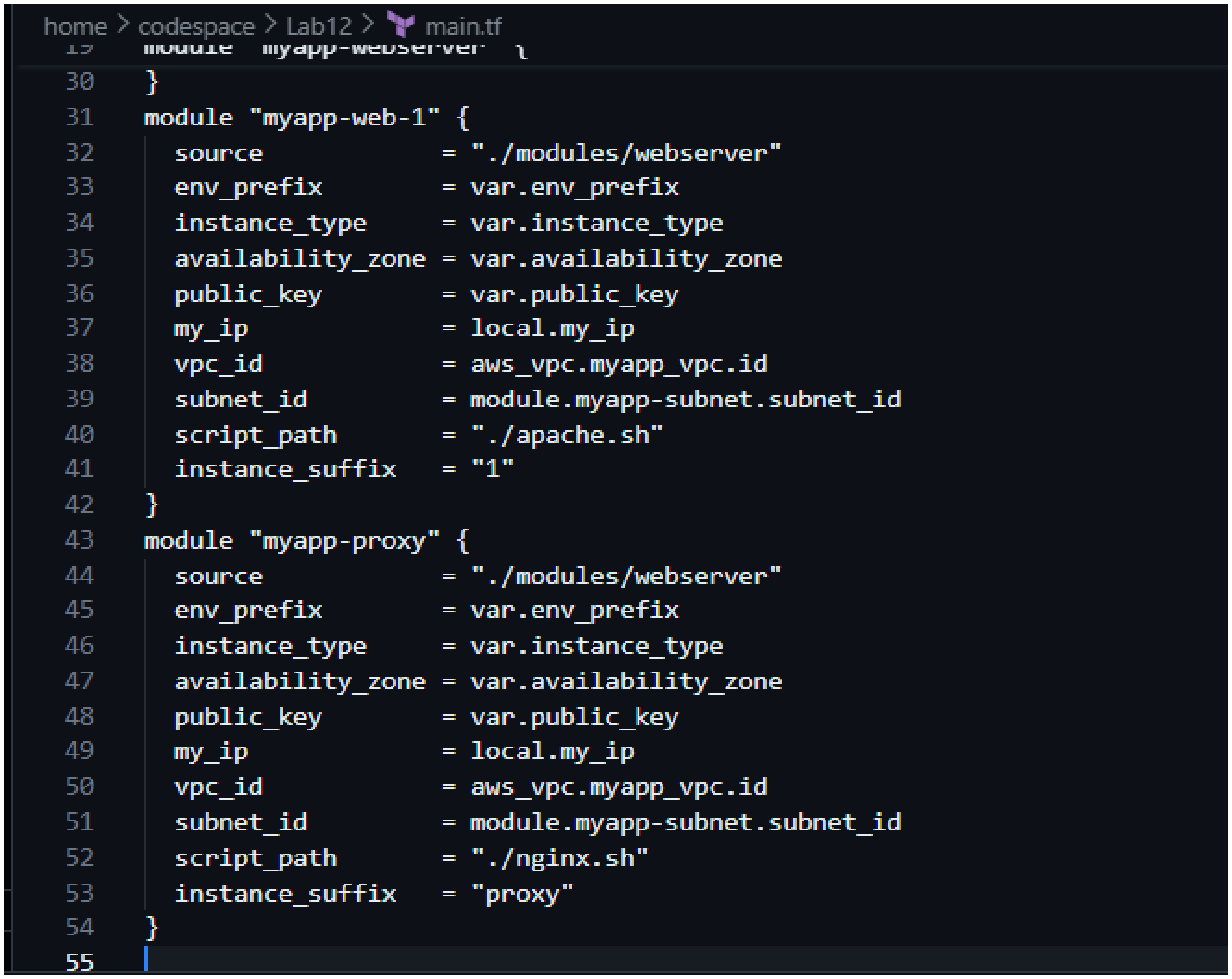
```
#!/bin/bash
yum update -y
yum install nginx -y
systemctl start nginx
systemctl enable nginx

```

```
@23-22411-013-sys →~/Lab12 $ code apache.sh
@23-22411-013-sys →~/Lab12 $ code nginx.sh
@23-22411-013-sys →~/Lab12 $
```

2. Add the backend web server module to main.tf:

```
module "myapp-web-1" {
  source = "./modules/webserver"
  env_prefix = var.env_prefix
  instance_type = var.instance_type
  availability_zone = var.availability_zone
  public_key = var. public_key
  my_ip = local.my_ip
  vpc_id = aws_vpc. myapp_vpc.id
  subnet_id = module.myapp-subnet.subnet. id
  script_path = "./apache.sh"
  instance_suffix = "1"
```

}

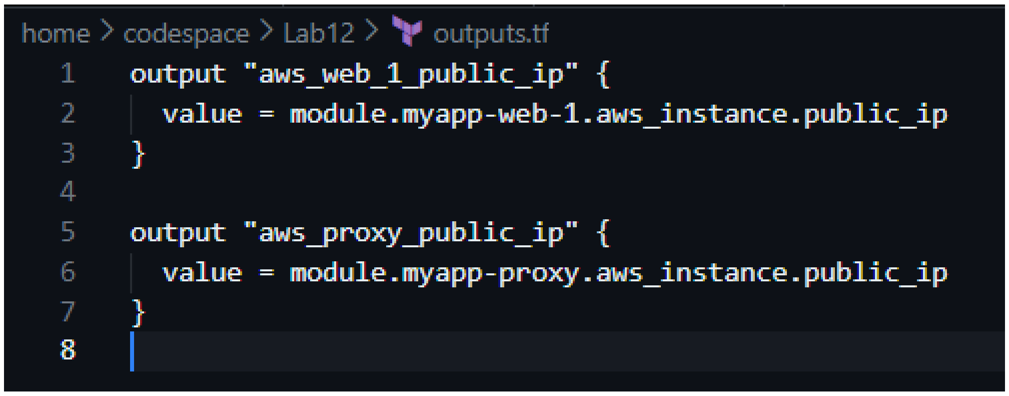


home > codespace > Lab12 > main.tf

```
module "myapp-webserver" {
}
module "myapp-web-1" {
  source            = "./modules/webserver"
  env_prefix        = var.env_prefix
  instance_type     = var.instance_type
  availability_zone = var.availability_zone
  public_key        = var.public_key
  my_ip             = local.my_ip
  vpc_id            = aws_vpc.myapp_vpc.id
  subnet_id         = module.myapp-subnet.subnet_id
  script_path       = "./apache.sh"
  instance_suffix   = "1"
}
module "myapp-proxy" {
  source            = "./modules/webserver"
  env_prefix        = var.env_prefix
  instance_type     = var.instance_type
  availability_zone = var.availability_zone
  public_key        = var.public_key
  my_ip             = local.my_ip
  vpc_id            = aws_vpc.myapp_vpc.id
  subnet_id         = module.myapp-subnet.subnet_id
  script_path       = "./nginx.sh"
  instance_suffix   = "proxy"
}
```

3. Update outputs.tf:

```
output "aws_web-1_public_ip" {
  value = module.myapp-web-1.aws_instance.public_ip
}
```

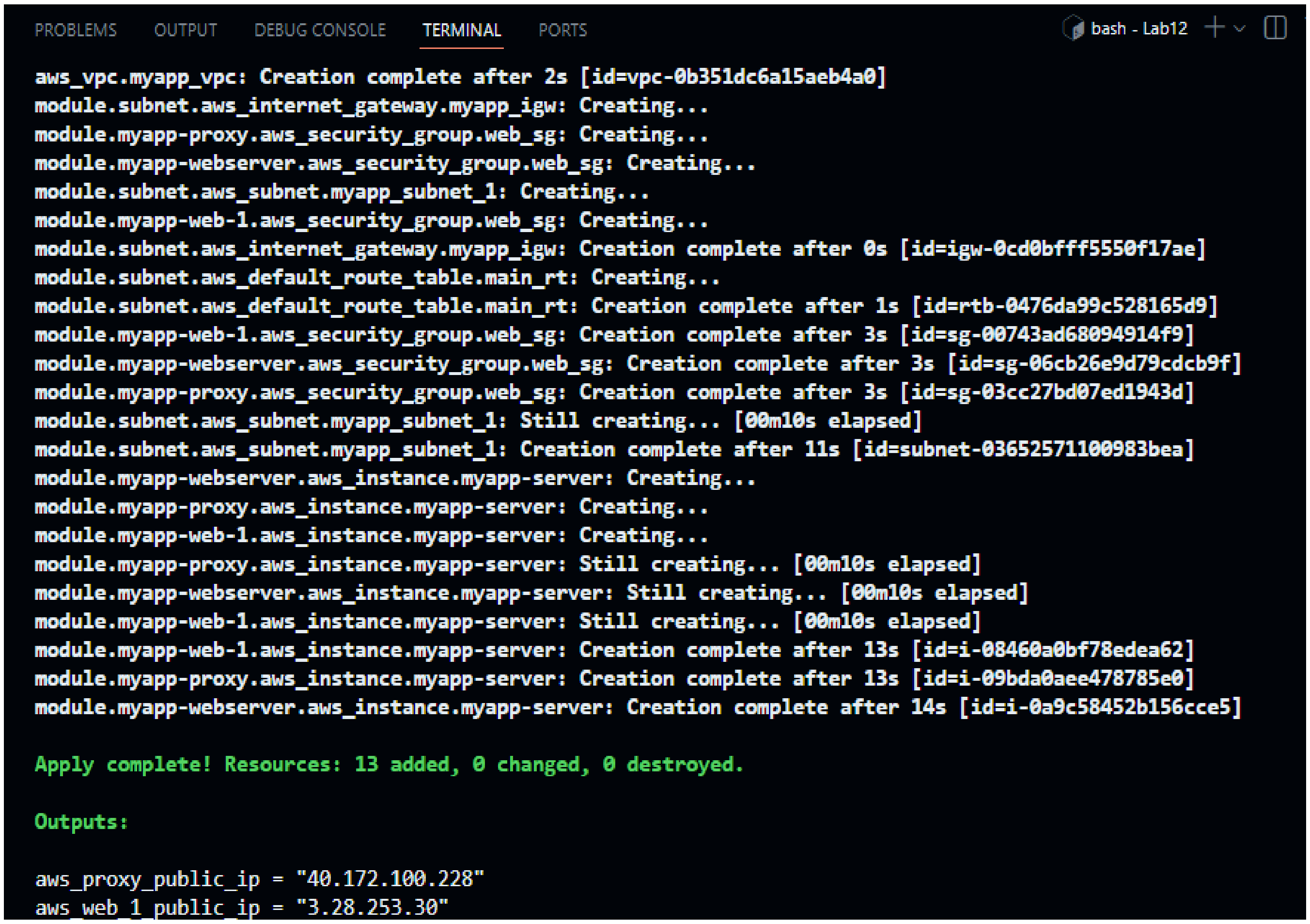


home > codespace > Lab12 > outputs.tf

```
output "aws_web_1_public_ip" {
  value = module.myapp-web-1.aws_instance.public_ip
}

output "aws_proxy_public_ip" {
  value = module.myapp-proxy.aws_instance.public_ip
}
```

4. Apply the configuration:

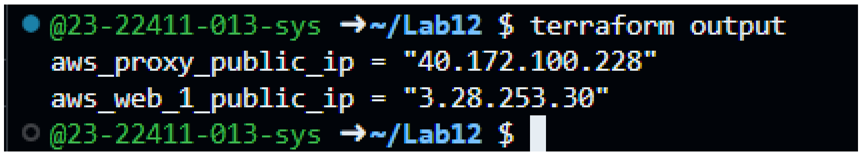
terraform apply -auto-approve



```
aws_vpc.myapp_vpc: Creation complete after 2s [id=vpc-0b351dc6a15aeb4a0]
module.subnet.aws_internet_gateway.myapp_igw: Creating...
module.myapp-proxy.aws_security_group.web_sg: Creating...
module.myapp-webserver.aws_security_group.web_sg: Creating...
module.subnet.aws_subnet.myapp_subnet_1: Creating...
module.myapp-web-1.aws_security_group.web_sg: Creating...
module.subnet.aws_internet_gateway.myapp_igw: Creation complete after 0s [id=igw-0cd0bfff5550f17ae]
module.subnet.aws_default_route_table.main_rt: Creating...
module.subnet.aws_default_route_table.main_rt: Creation complete after 1s [id=rtb-0476da99c528165d9]
module.myapp-web-1.aws_security_group.web_sg: Creation complete after 3s [id=sg-00743ad68094914f9]
module.myapp-webserver.aws_security_group.web_sg: Creation complete after 3s [id=sg-06cb26e9d79cdcb9f]
module.myapp-proxy.aws_security_group.web_sg: Creation complete after 3s [id=sg-03cc27bd07ed1943d]
module.subnet.aws_subnet.myapp_subnet_1: Still creating... [00m10s elapsed]
module.subnet.aws_subnet.myapp_subnet_1: Creation complete after 11s [id=subnet-03652571100983bea]
module.myapp-webserver.aws_instance.myapp-server: Creating...
module.myapp-proxy.aws_instance.myapp-server: Creating...
module.myapp-web-1.aws_instance.myapp-server: Creating...
module.myapp-proxy.aws_instance.myapp-server: Still creating... [00m10s elapsed]
module.myapp-webserver.aws_instance.myapp-server: Still creating... [00m10s elapsed]
module.myapp-web-1.aws_instance.myapp-server: Still creating... [00m10s elapsed]
module.myapp-web-1.aws_instance.myapp-server: Creation complete after 13s [id=i-08460a0bf78edea62]
module.myapp-proxy.aws_instance.myapp-server: Creation complete after 13s [id=i-09bda0aee478785e0]
module.myapp-webserver.aws_instance.myapp-server: Creation complete after 14s [id=i-0a9c58452b156cce5]

Apply complete! Resources: 13 added, 0 changed, 0 destroyed.

Outputs:

aws_proxy_public_ip = "40.172.100.228"
aws_web_1_public_ip = "3.28.253.30"
```

5. Get the outputs:

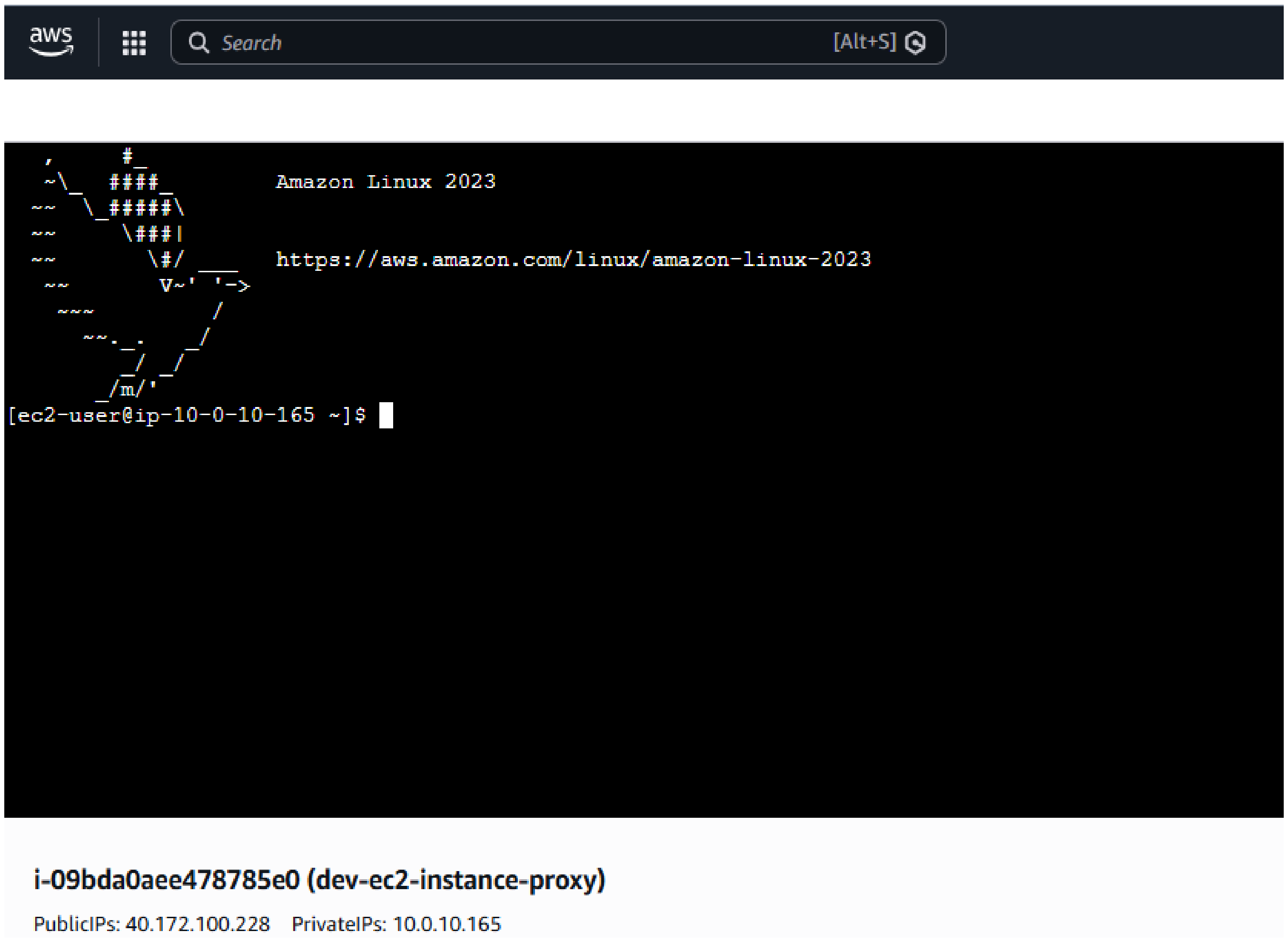
terraform output



```
@23-22411-013-sys ➜~/Lab12 $ terraform output
aws_proxy_public_ip = "40.172.100.228"
aws_web_1_public_ip = "3.28.253.30"
@23-22411-013-sys ➜~/Lab12 $
```

6. SSH into the webserver (Nginx proxy server):

ssh ec2-user@<webserver-public-ip>



```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-10-0-10-165 ~]$
```

i-09bda0aee478785e0 (dev-ec2-instance-proxy)

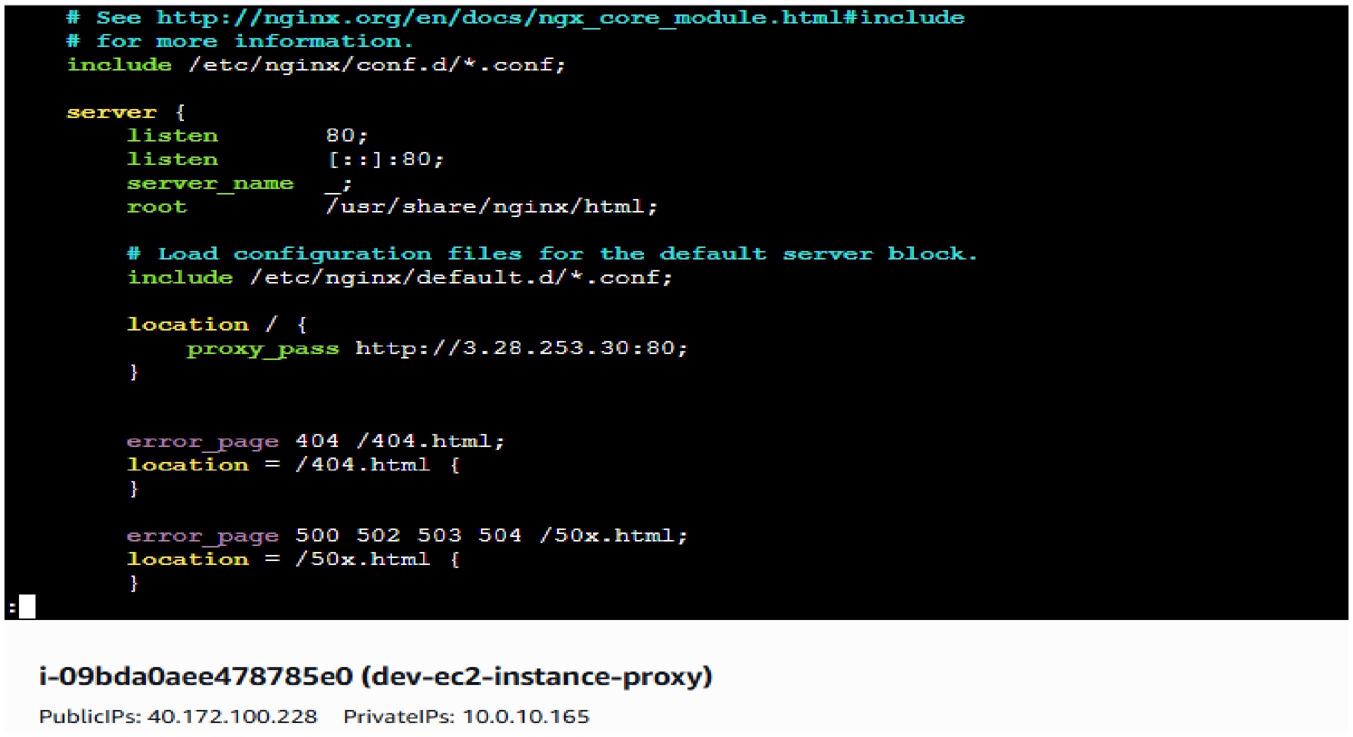
PublicIPs: 40.172.100.228   PrivateIPs: 10.0.10.165

7. Edit the Nginx configuration:

sudo vim /etc/nginx/nginx.conf

Modify the location block to proxy to web-1:

```
    location / {
  #      root /usr/share/nginx/html;
  #      index index. html;
        proxy_pass http://<web-1-public-ip>:80;
  #     proxy_pass http://backend_servers;
    }
```



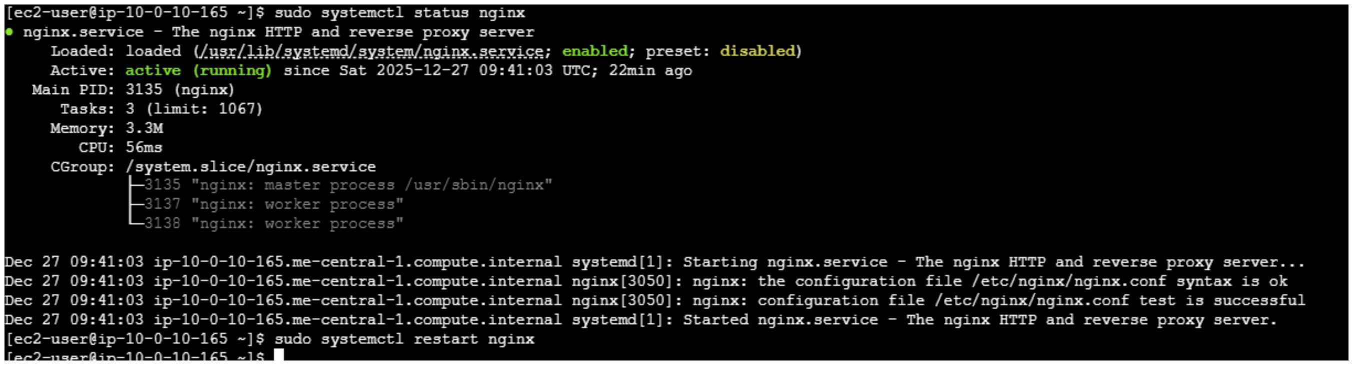
```
    # See http://nginx.org/en/docs/ngx_core_module.html#include
    # for more information.
    include /etc/nginx/conf.d/*.conf;

    server {
        listen       80;
        listen       [::]:80;
        server_name  _;
        root         /usr/share/nginx/html;

        # Load configuration files for the default server block.
        include /etc/nginx/default.d/*.conf;

        location / {
            proxy_pass http://3.28.253.30:80;
        }


        error_page 404 /404.html;
        location = /404.html {
        }

        error_page 500 502 503 504 /50x.html;
        location = /50x.html {
        }
:
```

i-09bda0aee478785e0 (dev-ec2-instance-proxy)

PublicIPs: 40.172.100.228   PrivateIPs: 10.0.10.165

8. Restart Nginx:

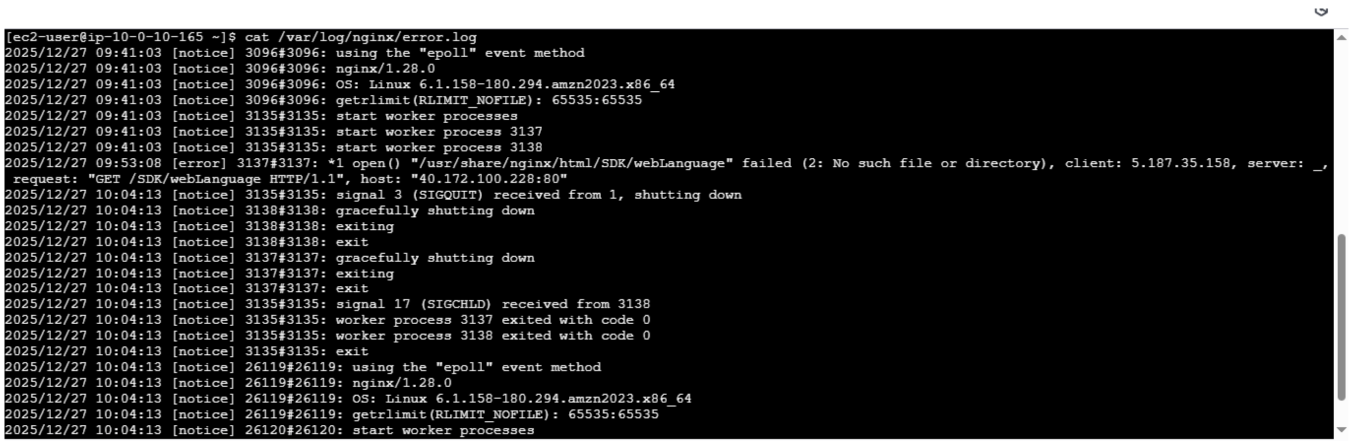
sudo systemctl restart nginx



```
[ec2-user@ip-10-0-10-165 ~]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
     Active: active (running) since Sat 2025-12-27 09:41:03 UTC; 22min ago
   Main PID: 3135 (nginx)
      Tasks: 3 (limit: 1067)
     Memory: 3.3M
        CPU: 56ms
     CGroup: /system.slice/nginx.service
             ├─3135 "nginx: master process /usr/sbin/nginx"
             ├─3137 "nginx: worker process"
             └─3138 "nginx: worker process"

Dec 27 09:41:03 ip-10-0-10-165.me-central-1.compute.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Dec 27 09:41:03 ip-10-0-10-165.me-central-1.compute.internal nginx[3050]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Dec 27 09:41:03 ip-10-0-10-165.me-central-1.compute.internal nginx[3050]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Dec 27 09:41:03 ip-10-0-10-165.me-central-1.compute.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-10-0-10-165 ~]$ sudo systemctl restart nginx
[ec2-user@ip-10-0-10-165 ~]$
```

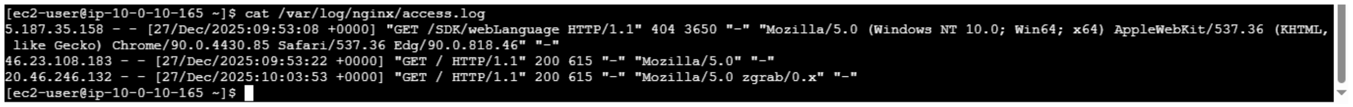
9. View Nginx logs and configuration files:

cat /var/log/nginx/error.log



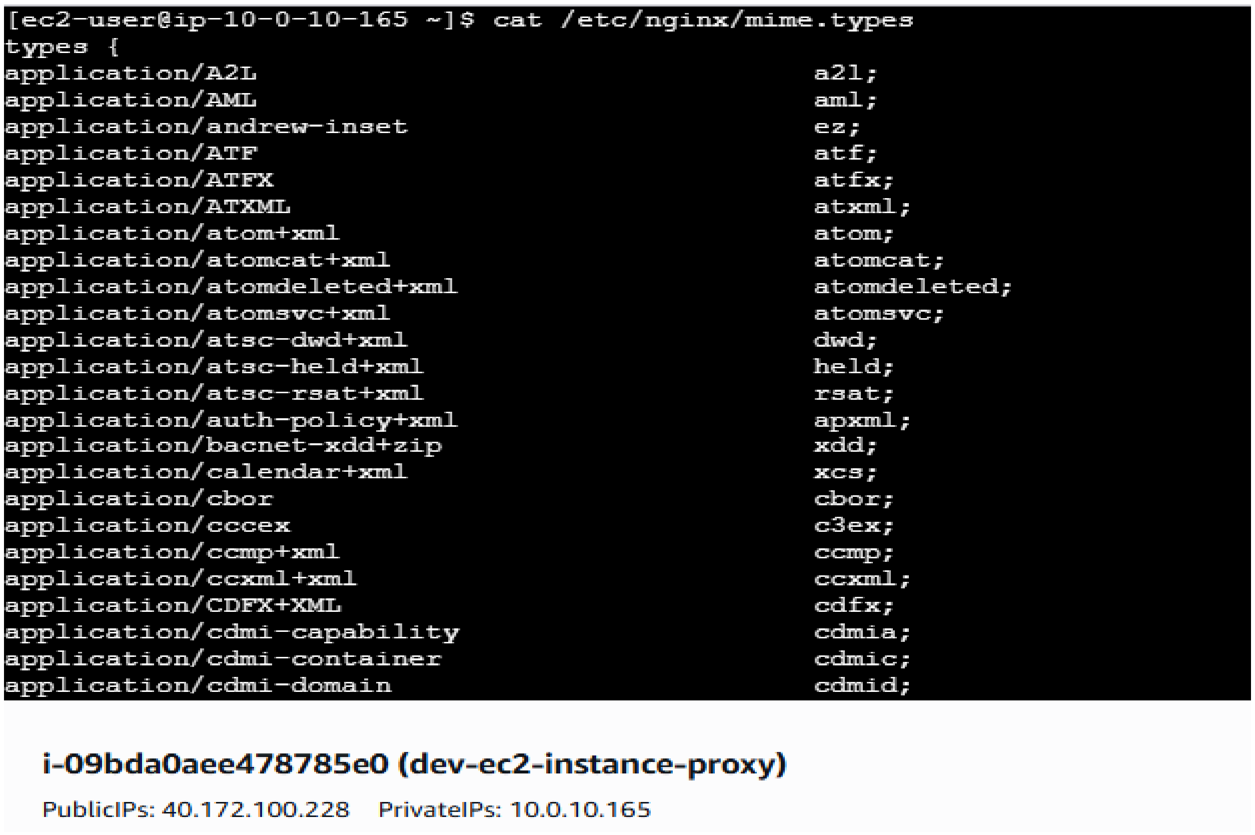
```
[ec2-user@ip-10-0-10-165 ~]$ cat /var/log/nginx/error.log
2025/12/27 09:41:03 [notice] 3096#3096: using the "epoll" event method
2025/12/27 09:41:03 [notice] 3096#3096: nginx/1.28.0
2025/12/27 09:41:03 [notice] 3096#3096: OS: Linux 6.1.158-180.294.amzn2023.x86_64
2025/12/27 09:41:03 [notice] 3096#3096: getrlimit(RLIMIT_NOFILE): 65535:65535
2025/12/27 09:41:03 [notice] 3135#3135: start worker processes
2025/12/27 09:41:03 [notice] 3135#3135: start worker process 3137
2025/12/27 09:41:03 [notice] 3135#3135: start worker process 3138
2025/12/27 09:53:08 [error] 3137#3137: *1 open() "/usr/share/nginx/html/SDK/webLanguage" failed (2: No such file or directory), client: 5.187.35.158, server: _,
 request: "GET /SDK/webLanguage HTTP/1.1", host: "40.172.100.228:80"
2025/12/27 10:04:13 [notice] 3135#3135: signal 3 (SIGQUIT) received from 1, shutting down
2025/12/27 10:04:13 [notice] 3138#3138: gracefully shutting down
2025/12/27 10:04:13 [notice] 3138#3138: exiting
2025/12/27 10:04:13 [notice] 3138#3138: exit
2025/12/27 10:04:13 [notice] 3137#3137: gracefully shutting down
2025/12/27 10:04:13 [notice] 3137#3137: exiting
2025/12/27 10:04:13 [notice] 3137#3137: exit
2025/12/27 10:04:13 [notice] 3135#3135: signal 17 (SIGCHLD) received from 3138
2025/12/27 10:04:13 [notice] 3135#3135: worker process 3137 exited with code 0
2025/12/27 10:04:13 [notice] 3135#3135: worker process 3138 exited with code 0
2025/12/27 10:04:13 [notice] 3135#3135: exit
2025/12/27 10:04:13 [notice] 26119#26119: using the "epoll" event method
2025/12/27 10:04:13 [notice] 26119#26119: nginx/1.28.0
2025/12/27 10:04:13 [notice] 26119#26119: OS: Linux 6.1.158-180.294.amzn2023.x86_64
2025/12/27 10:04:13 [notice] 26119#26119: getrlimit(RLIMIT_NOFILE): 65535:65535
2025/12/27 10:04:13 [notice] 26120#26120: start worker processes
```

cat /var/log/nginx/access.log



```
[ec2-user@ip-10-0-10-165 ~]$ cat /var/log/nginx/access.log
5.187.35.158 - - [27/Dec/2025:09:53:08 +0000] "GET /SDK/webLanguage HTTP/1.1" 404 3650 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/90.0.4430.85 Safari/537.36 Edg/90.0.818.46" "-"
46.23.108.183 - - [27/Dec/2025:09:53:22 +0000] "GET / HTTP/1.1" 200 615 "-" "Mozilla/5.0" "-"
20.46.246.132 - - [27/Dec/2025:10:03:53 +0000] "GET / HTTP/1.1" 200 615 "-" "Mozilla/5.0 zgrab/0.x" "-"
[ec2-user@ip-10-0-10-165 ~]$
```

cat /etc/nginx/mime.types

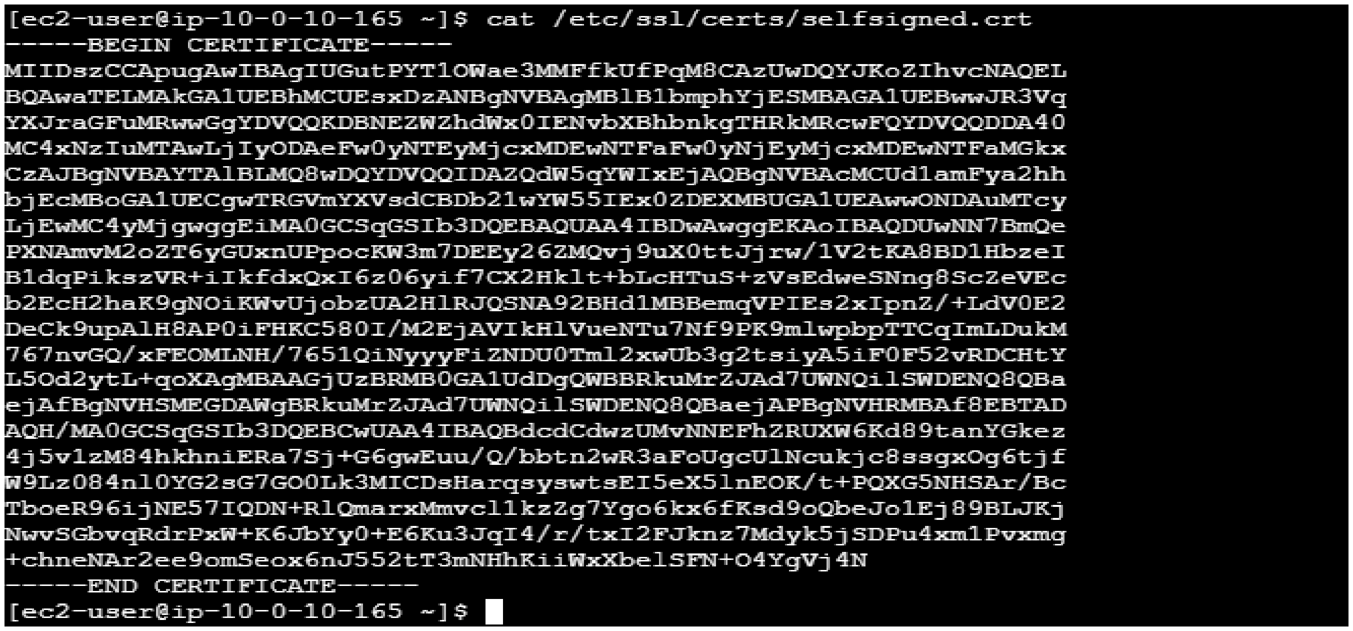


```
[ec2-user@ip-10-0-10-165 ~]$ cat /etc/nginx/mime.types
types {
application/A2L                                 a2l;
application/AML                                 aml;
application/andrew-inset                        ez;
application/ATF                                 atf;
application/ATFX                                atfx;
application/ATXML                               atxml;
application/atom+xml                            atom;
application/atomcat+xml                         atomcat;
application/atomdeleted+xml                     atomdeleted;
application/atomsvc+xml                         atomsvc;
application/atsc-dwd+xml                        dwd;
application/atsc-held+xml                       held;
application/atsc-rsat+xml                       rsat;
application/auth-policy+xml                     apxml;
application/bacnet-xdd+zip                      xdd;
application/calendar+xml                        xcs;
application/cbor                                cbor;
application/cccex                               c3ex;
application/ccmp+xml                            ccmp;
application/ccxml+xml                           ccxml;
application/CDFX+XML                            cdfx;
application/cdmi-capability                     cdmia;
application/cdmi-container                      cdmic;
application/cdmi-domain                         cdmid;
```

i-09bda0aee478785e0 (dev-ec2-instance-proxy)

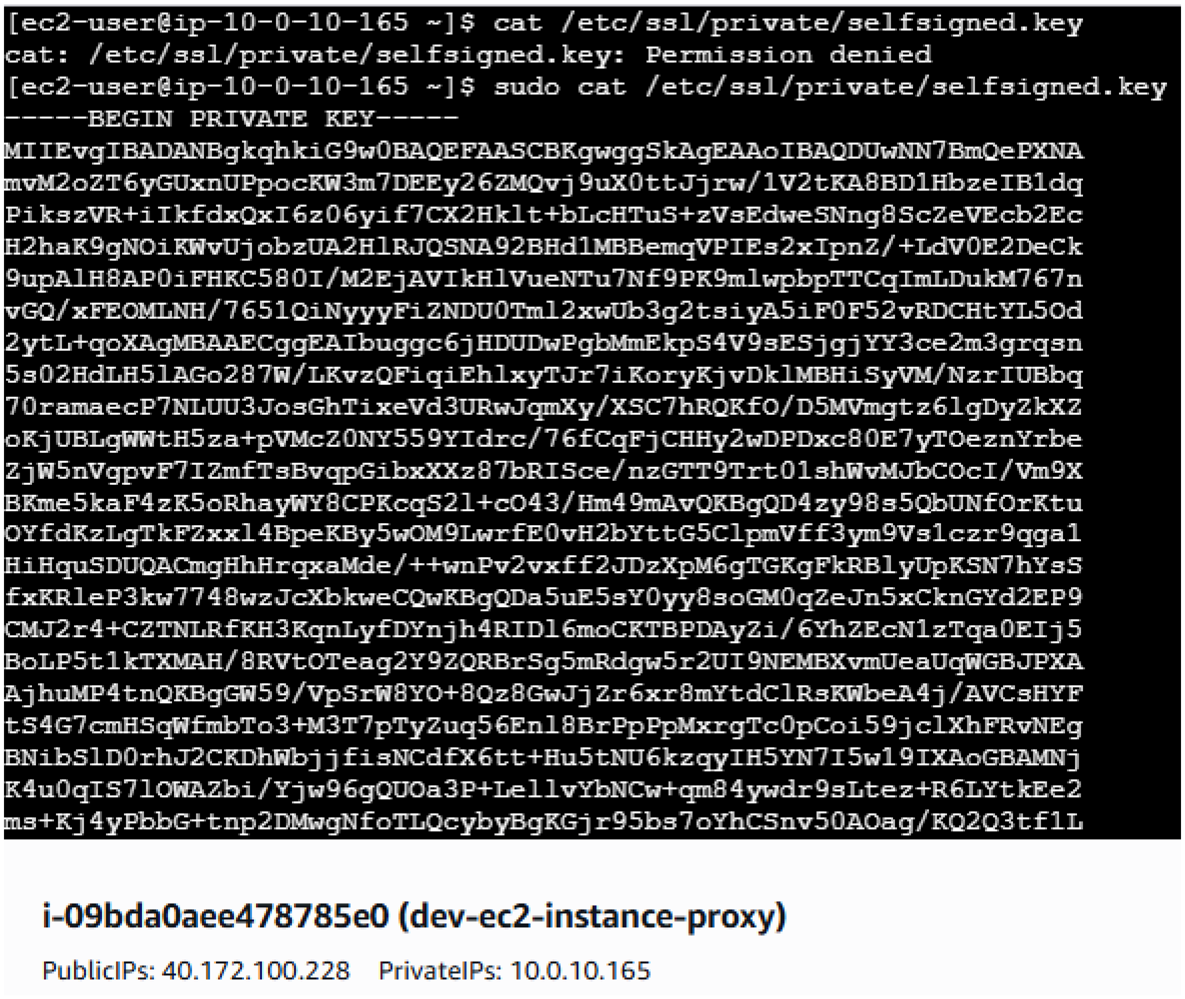
PublicIPs: 40.172.100.228   PrivateIPs: 10.0.10.165

cat /etc/ssl/certs/selfsigned. Crt



```
[ec2-user@ip-10-0-10-165 ~]$ cat /etc/ssl/certs/selfsigned.crt
-----BEGIN CERTIFICATE-----
MIIDszCCApugAwIBAgIUGutPYT1OWae3MMFfkUfPqM8CAzUwDQYJKoZIhvcNAQEL
BQAwaTELMAkGA1UEBhMCUEsxDzANBgNVBAgMBlB1bmphYjESMBAGA1UEBwwJR3Vq
YXJraGFuMRwwGgYDVQQKDBNEZWZhdWx0IENvbXBhbnkgTHRkMRcwFQYDVQQDDA40
MC4xNzIuMTAwLjIyODAeFw0yNTEyMjcxMDEwNTFaFw0yNjEyMjcxMDEwNTFaMGkx
CzAJBgNVBAYTAlBLMQ8wDQYDVQQIDAZQdW5qYWIxEjAQBgNVBAcMCUd1amFya2hh
bjEcMBoGA1UECgwTRGVmYXVsdCBDb21wYW55IEx0ZDEXMBUGA1UEAwwONDAuMTcy
LjEwMC4yMjgwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDUwNN7BmQe
PXNAmvM2oZT6yGUxnUPpocKW3m7DEEy26ZMQvj9uX0ttJjrw/1V2tKA8BD1HbzeI
B1dqPikszVR+iIkfdxQxI6z06yif7CX2Hklt+bLcHTuS+zVsEdweSNng8ScZeVEc
b2EcH2haK9gNOiKWvUjobzUA2HlRJQSNA92BHd1MBBemqVPIEs2xIpnZ/+LdV0E2
DeCk9upAlH8AP0iFHKC580I/M2EjAVIkHlVueNTu7Nf9PK9mlwpbpTTCqImLDukM
767nvGQ/xFEOMLNH/7651QiNyyyFiZNDU0Tml2xwUb3g2tsiyA5iF0F52vRDCHtY
L5Od2ytL+qoXAgMBAAGjUzBRMB0GA1UdDgQWBBRkuMrZJAd7UWNQilSWDENQ8QBa
ejAfBgNVHSMEGDAWgBRkuMrZJAd7UWNQilSWDENQ8QBaejAPBgNVHRMBAf8EBTAD
AQH/MA0GCSqGSIb3DQEBCwUAA4IBAQBdcdCdwzUMvNNEFhZRUXW6Kd89tanYGkez
4j5v1zM84hkhniERa7Sj+G6gwEuu/Q/bbtn2wR3aFoUgcUlNcukjc8ssgxOg6tjf
W9Lz084nl0YG2sG7GO0Lk3MICDsHarqsyswtsEI5eX5lnEOK/t+PQXG5NHSAr/Bc
TboeR96ijNE57IQDN+RlQmarxMmvcl1kzZg7Ygo6kx6fKsd9oQbeJo1Ej89BLJKj
NwvSGbvqRdrPxW+K6JbYy0+E6Ku3JqI4/r/txI2FJknz7Mdyk5jSDPu4xm1Pvxmg
+chneNAr2ee9omSeox6nJ552tT3mNHhKiiWxXbelSFN+O4YgVj4N
-----END CERTIFICATE-----
[ec2-user@ip-10-0-10-165 ~]$
```

cat /etc/ssl/private/selfsigned.key



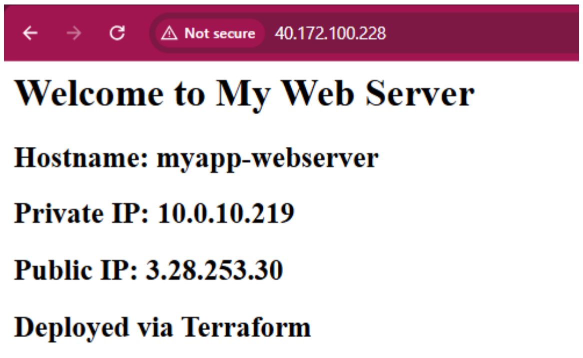
```
[ec2-user@ip-10-0-10-165 ~]$ cat /etc/ssl/private/selfsigned.key
cat: /etc/ssl/private/selfsigned.key: Permission denied
[ec2-user@ip-10-0-10-165 ~]$ sudo cat /etc/ssl/private/selfsigned.key
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQDUwNN7BmQePXNA
mvM2oZT6yGUxnUPpocKW3m7DEEy26ZMQvj9uX0ttJjrw/1V2tKA8BD1HbzeIB1dq
PikszVR+iIkfdxQxI6z06yif7CX2Hklt+bLcHTuS+zVsEdweSNng8ScZeVEcb2Ec
H2haK9gNOiKWvUjobzUA2HlRJQSNA92BHd1MBBemqVPIEs2xIpnZ/+LdV0E2DeCk
9upAlH8AP0iFHKC580I/M2EjAVIkHlVueNTu7Nf9PK9mlwpbpTTCqImLDukM767n
vGQ/xFEOMLNH/7651QiNyyyFiZNDU0Tml2xwUb3g2tsiyA5iF0F52vRDCHtYL5Od
2ytL+qoXAgMBAAECggEAIbuggc6jHDUDwPgbMmEkpS4V9sESjgjYY3ce2m3grqsn
5s02HdLH5lAGo287W/LKvzQFiqiEhlxyTJr7iKoryKjvDklMBHiSyVM/NzrIUBbq
70ramaecP7NLUU3JosGhTixeVd3URwJqmXy/XSC7hRQKfO/D5MVmgtz6lgDyZkXZ
oKjUBLgWWtH5za+pVMcZ0NY559YIdrc/76fCqFjCHHy2wDPDxc80E7yTOeznYrbe
ZjW5nVgpvF7IZmfTsBvqpGibxXXz87bRISce/nzGTT9Trt01shWvMJbCOcI/Vm9X
BKme5kaF4zK5oRhayWY8CPKcqS2l+cO43/Hm49mAvQKBgQD4zy98s5QbUNfOrKtu
OYfdKzLgTkFZxxl4BpeKBy5wOM9LwrfE0vH2bYttG5ClpmVff3ym9Vs1czr9qga1
HiHquSDUQACmgHhHrqxaMde/++wnPv2vxff2JDzXpM6gTGKgFkRBlyUpKSN7hYsS
fxKRleP3kw7748wzJcXbkweCQwKBgQDa5uE5sY0yy8soGM0qZeJn5xCknGYd2EP9
CMJ2r4+CZTNLRfKH3KqnLyfDYnjh4RIDl6moCKTBPDAyZi/6YhZEcN1zTqa0EIj5
BoLP5t1kTXMAH/8RVtOTeag2Y9ZQRBrSg5mRdgw5r2UI9NEMBXvmUeaUqWGBJPXA
AjhuMP4tnQKBgGW59/VpSrW8YO+8Qz8GwJjZr6xr8mYtdClRsKWbeA4j/AVCsHYF
tS4G7cmHSqWfmbTo3+M3T7pTyZuq56Enl8BrPpPpMxrgTc0pCoi59jclXhFRvNEg
BNibSlD0rhJ2CKDhWbjjfisNCdfX6tt+Hu5tNU6kzqyIH5YN7I5w19IXAoGBAMNj
K4u0qIS7lOWAZbi/Yjw96gQUOa3P+LellvYbNCw+qm84ywdr9sLtez+R6LYtkEe2
ms+Kj4yPbbG+tnp2DMwgNfoTLQcybyBgKGjr95bs7oYhCSnv50AOag/KQ2Q3tf1L
```

i-09bda0aee478785e0 (dev-ec2-instance-proxy)

PublicIPs: 40.172.100.228 PrivateIPs: 10.0.10.165

10. Test reverse proxy in browser:

Open browser and navigate to https://<webserver-public-ip>



Welcome to My Web Server

Hostname: myapp-webserver

Private IP: 10.0.10.219

Public IP: 3.28.253.30

Deployed via Terraform

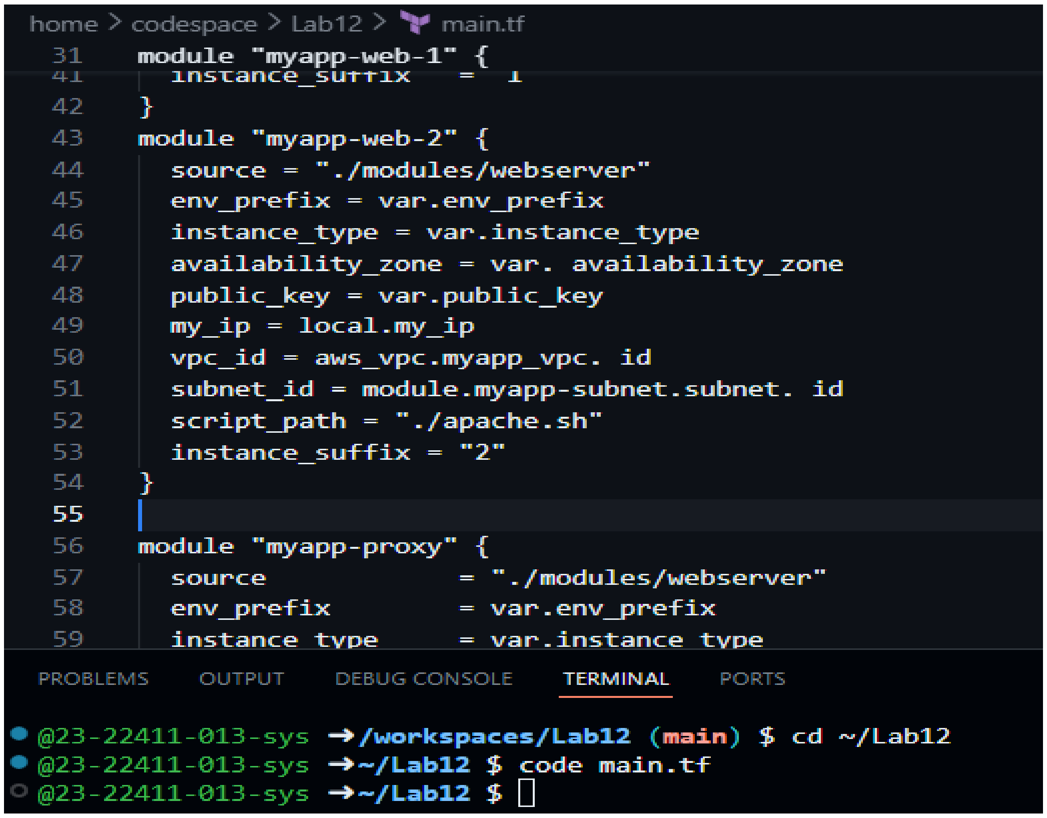
Browser output showing Apache backend content accessed through Nginx reverse proxy.

Task 8 — Configure Nginx as load balancer

In this task, you will add a second backend server and configure Nginx to load balance between them.

1. Add the second web server module to main.tf:

```
module "myapp-web-2" {
  source = "./modules/webserver"
  env_prefix = var.env_prefix
  instance_type = var.instance_type
  availability_zone = var. availability_zone
  public_key = var.public_key
  my_ip = local.my_ip
  vpc_id = aws_vpc.myapp_vpc. id
  subnet_id = module.myapp-subnet.subnet. id
  script_path = "./apache.sh"
  instance_suffix = "2"
}
```

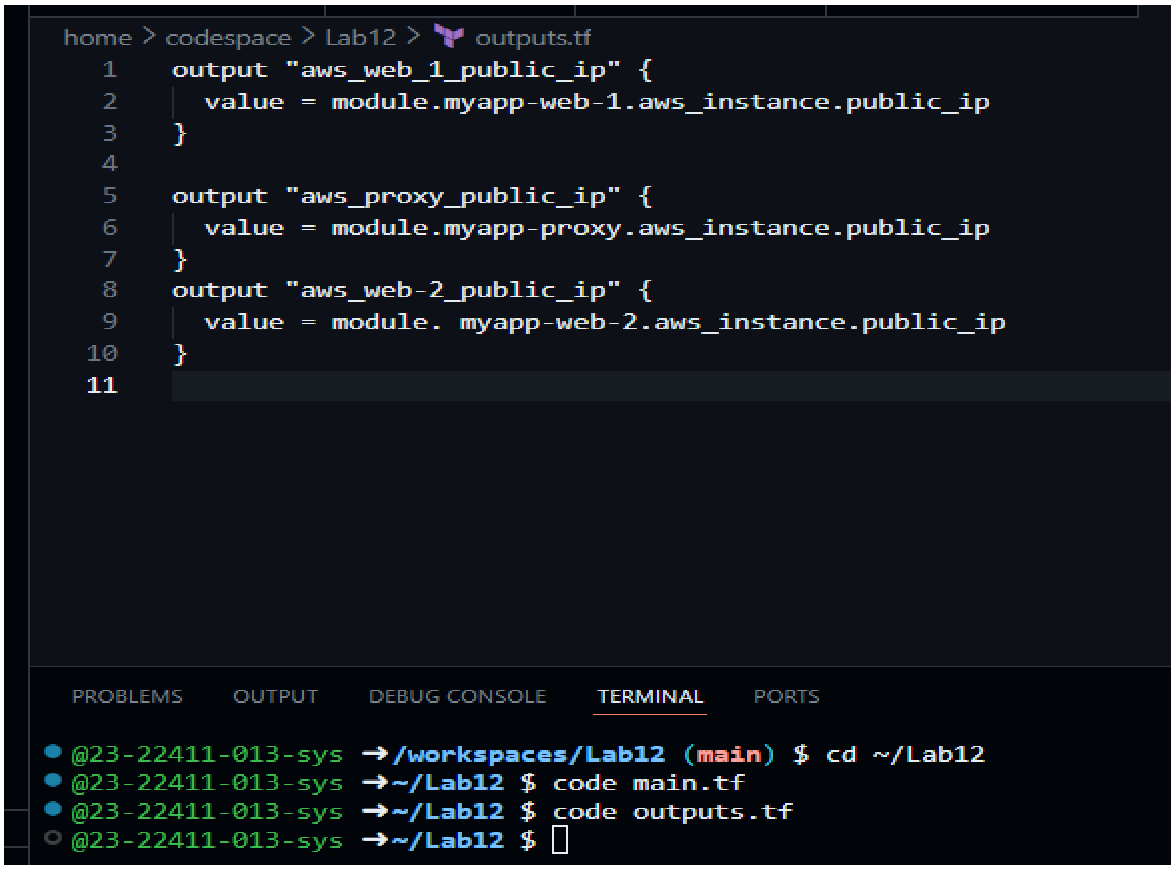


```
module "myapp-web-1" {
  instance_suffix    =  1
}
module "myapp-web-2" {
  source = "./modules/webserver"
  env_prefix = var.env_prefix
  instance_type = var.instance_type
  availability_zone = var. availability_zone
  public_key = var.public_key
  my_ip = local.my_ip
  vpc_id = aws_vpc.myapp_vpc. id
  subnet_id = module.myapp-subnet.subnet. id
  script_path = "./apache.sh"
  instance_suffix = "2"
}

module "myapp-proxy" {
  source              = "./modules/webserver"
  env_prefix          = var.env_prefix
  instance_type       = var.instance_type
```

```
@23-22411-013-sys →/workspaces/Lab12 (main) $ cd ~/Lab12
@23-22411-013-sys →~/Lab12 $ code main.tf
@23-22411-013-sys →~/Lab12 $
```

2. Update outputs.tf:

```
output "aws_web-2_public_ip" {
  value = module. myapp-web-2.aws_instance.public_ip
}
```

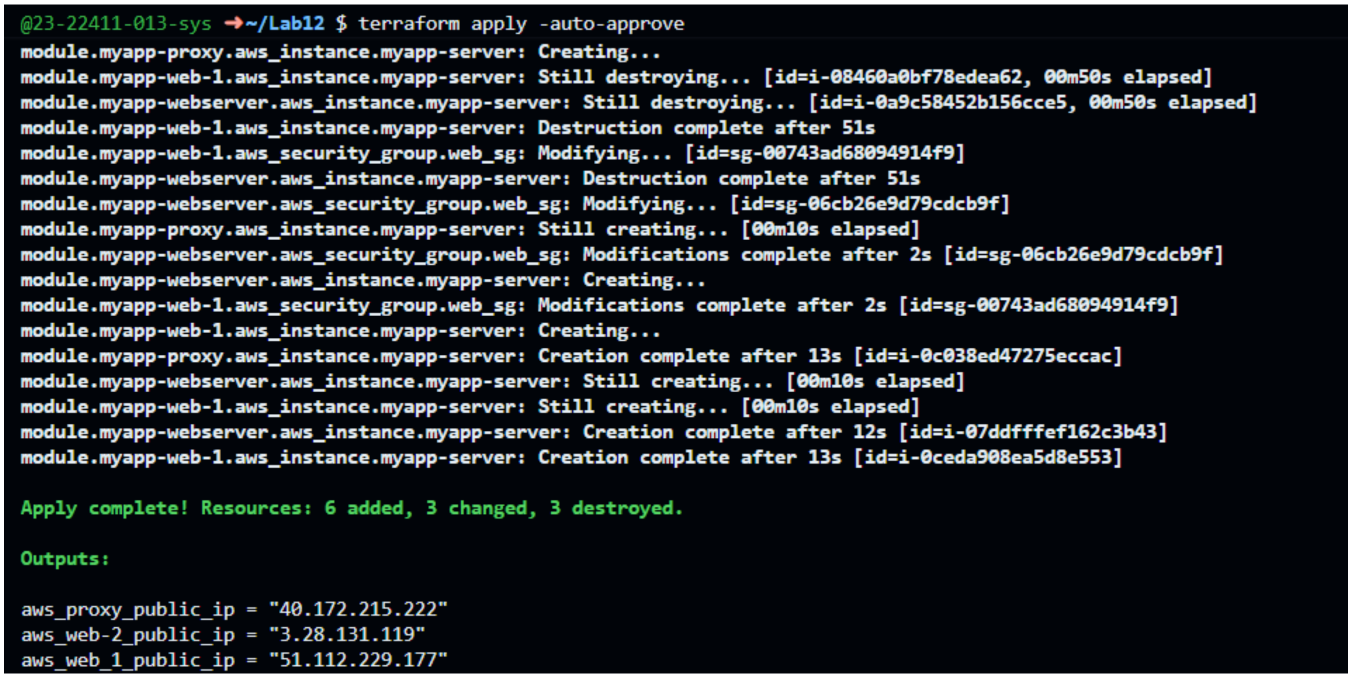


```
output "aws_web_1_public_ip" {
  value = module.myapp-web-1.aws_instance.public_ip
}

output "aws_proxy_public_ip" {
  value = module.myapp-proxy.aws_instance.public_ip
}
output "aws_web-2_public_ip" {
  value = module. myapp-web-2.aws_instance.public_ip
}

```

```
@23-22411-013-sys ➜/workspaces/Lab12 (main) $ cd ~/Lab12
@23-22411-013-sys ➜~/Lab12 $ code main.tf
@23-22411-013-sys ➜~/Lab12 $ code outputs.tf
@23-22411-013-sys ➜~/Lab12 $ 
```

3. Apply the configuration:

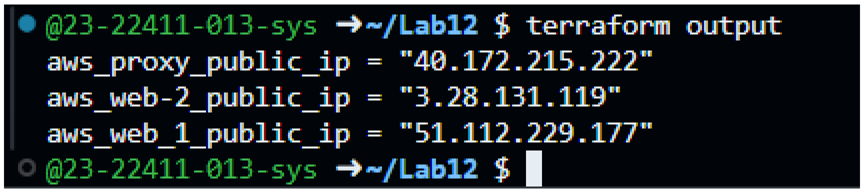
terraform apply -auto-approve



```
@23-22411-013-sys ➜~/Lab12 $ terraform apply -auto-approve
module.myapp-proxy.aws_instance.myapp-server: Creating...
module.myapp-web-1.aws_instance.myapp-server: Still destroying... [id=i-08460a0bf78edea62, 00m50s elapsed]
module.myapp-webserver.aws_instance.myapp-server: Still destroying... [id=i-0a9c58452b156cce5, 00m50s elapsed]
module.myapp-web-1.aws_instance.myapp-server: Destruction complete after 51s
module.myapp-web-1.aws_security_group.web_sg: Modifying... [id=sg-00743ad68094914f9]
module.myapp-webserver.aws_instance.myapp-server: Destruction complete after 51s
module.myapp-webserver.aws_security_group.web_sg: Modifying... [id=sg-06cb26e9d79cdcb9f]
module.myapp-proxy.aws_instance.myapp-server: Still creating... [00m10s elapsed]
module.myapp-webserver.aws_security_group.web_sg: Modifications complete after 2s [id=sg-06cb26e9d79cdcb9f]
module.myapp-webserver.aws_instance.myapp-server: Creating...
module.myapp-web-1.aws_security_group.web_sg: Modifications complete after 2s [id=sg-00743ad68094914f9]
module.myapp-web-1.aws_instance.myapp-server: Creating...
module.myapp-proxy.aws_instance.myapp-server: Creation complete after 13s [id=i-0c038ed47275eccac]
module.myapp-webserver.aws_instance.myapp-server: Still creating... [00m10s elapsed]
module.myapp-web-1.aws_instance.myapp-server: Still creating... [00m10s elapsed]
module.myapp-webserver.aws_instance.myapp-server: Creation complete after 12s [id=i-07ddfffef162c3b43]
module.myapp-web-1.aws_instance.myapp-server: Creation complete after 13s [id=i-0ceda908ea5d8e553]

Apply complete! Resources: 6 added, 3 changed, 3 destroyed.

Outputs:

aws_proxy_public_ip = "40.172.215.222"
aws_web-2_public_ip = "3.28.131.119"
aws_web_1_public_ip = "51.112.229.177"
```

4. Get all outputs:

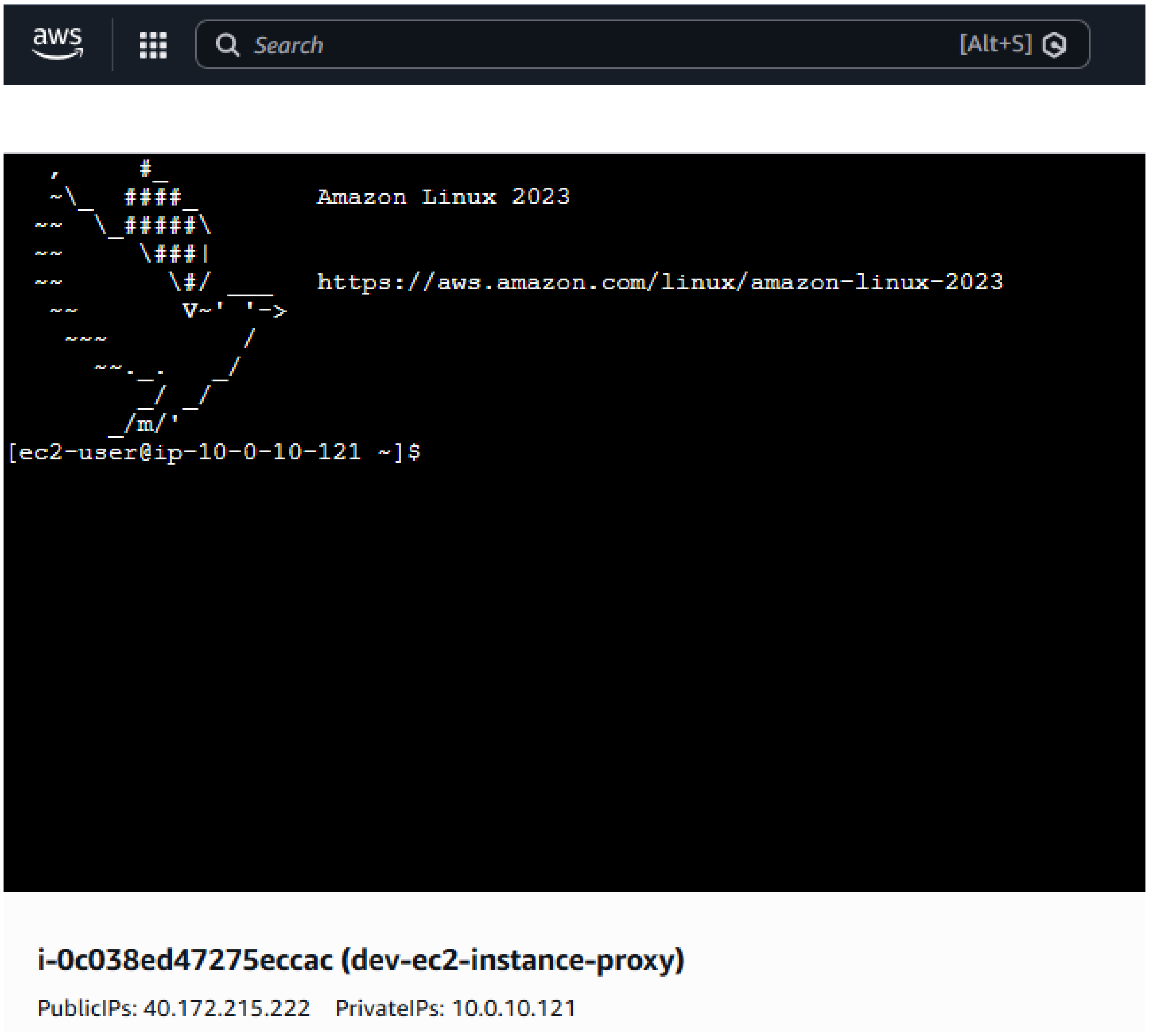
terraform output



```
@23-22411-013-sys ➜~/Lab12 $ terraform output
aws_proxy_public_ip = "40.172.215.222"
aws_web-2_public_ip = "3.28.131.119"
aws_web_1_public_ip = "51.112.229.177"
@23-22411-013-sys ➜~/Lab12 $ 
```

5. SSH into the webserver (Nginx proxy):

ssh ec2-user@<webserver-public-ip>



```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-10-0-10-121 ~]$
```

i-0c038ed47275eccac (dev-ec2-instance-proxy)

PublicIPs: 40.172.215.222   PrivateIPs: 10.0.10.121

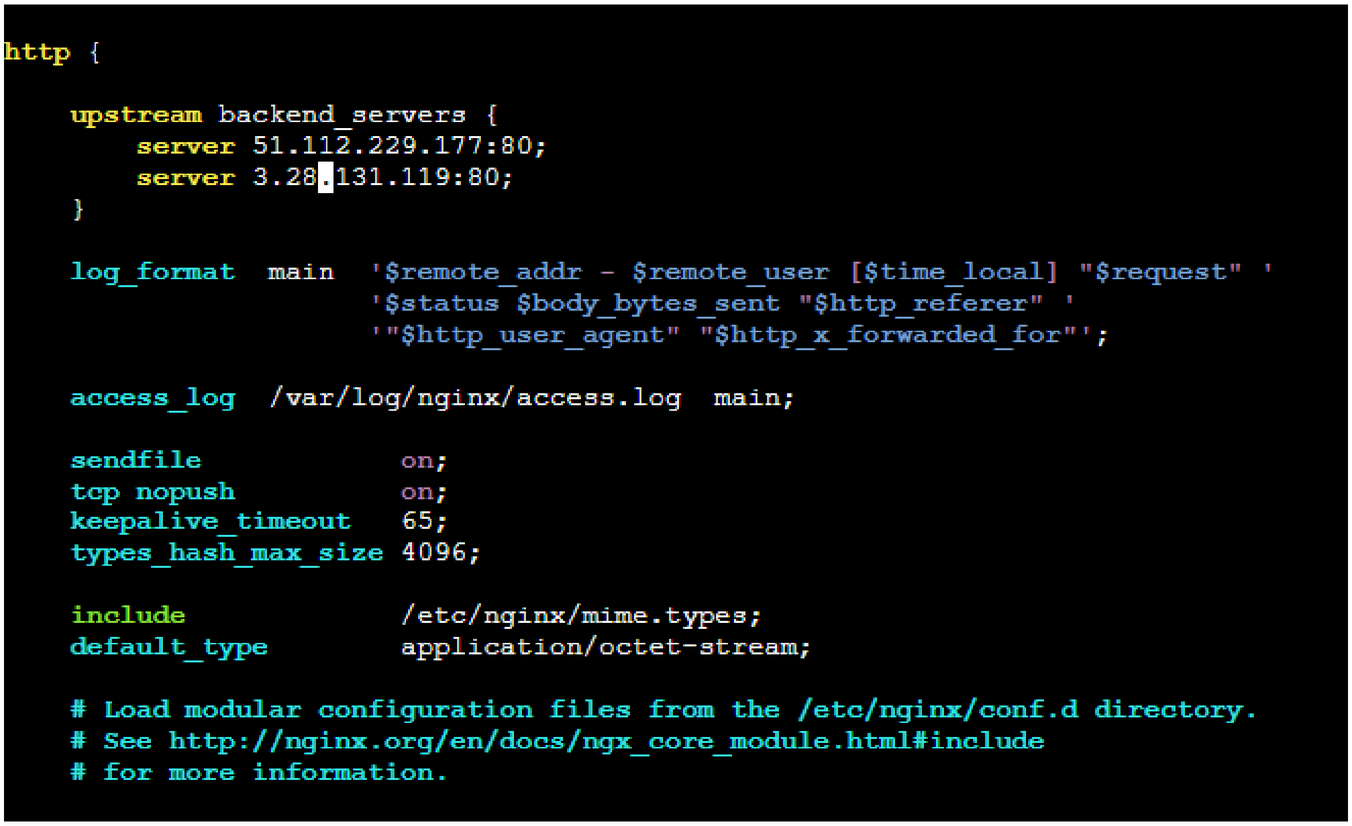
6. Edit Nginx configuration for load balancing:

sudo vim /etc/nginx/nginx.conf

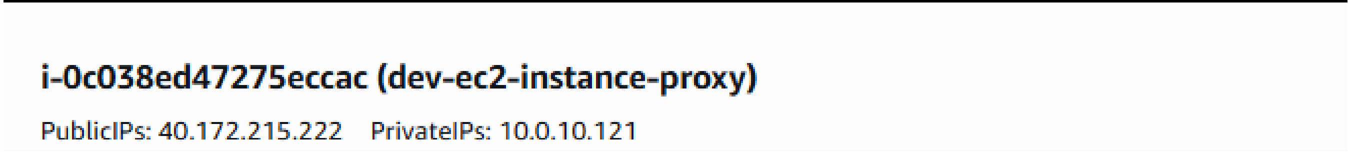
Update the upstream block and location:

```
upstream backend_servers {
    server <web-1-public-ip>:80;
```

```
    server <web-2-public-ip>: 80;
}
# ...  in server block:
    location / {
#       root /usr/share/nginx/html;
#       index index.html;
#       proxy_pass http://<web-1-public-ip>:80;
        proxy_pass http://backend_servers;
    }
```

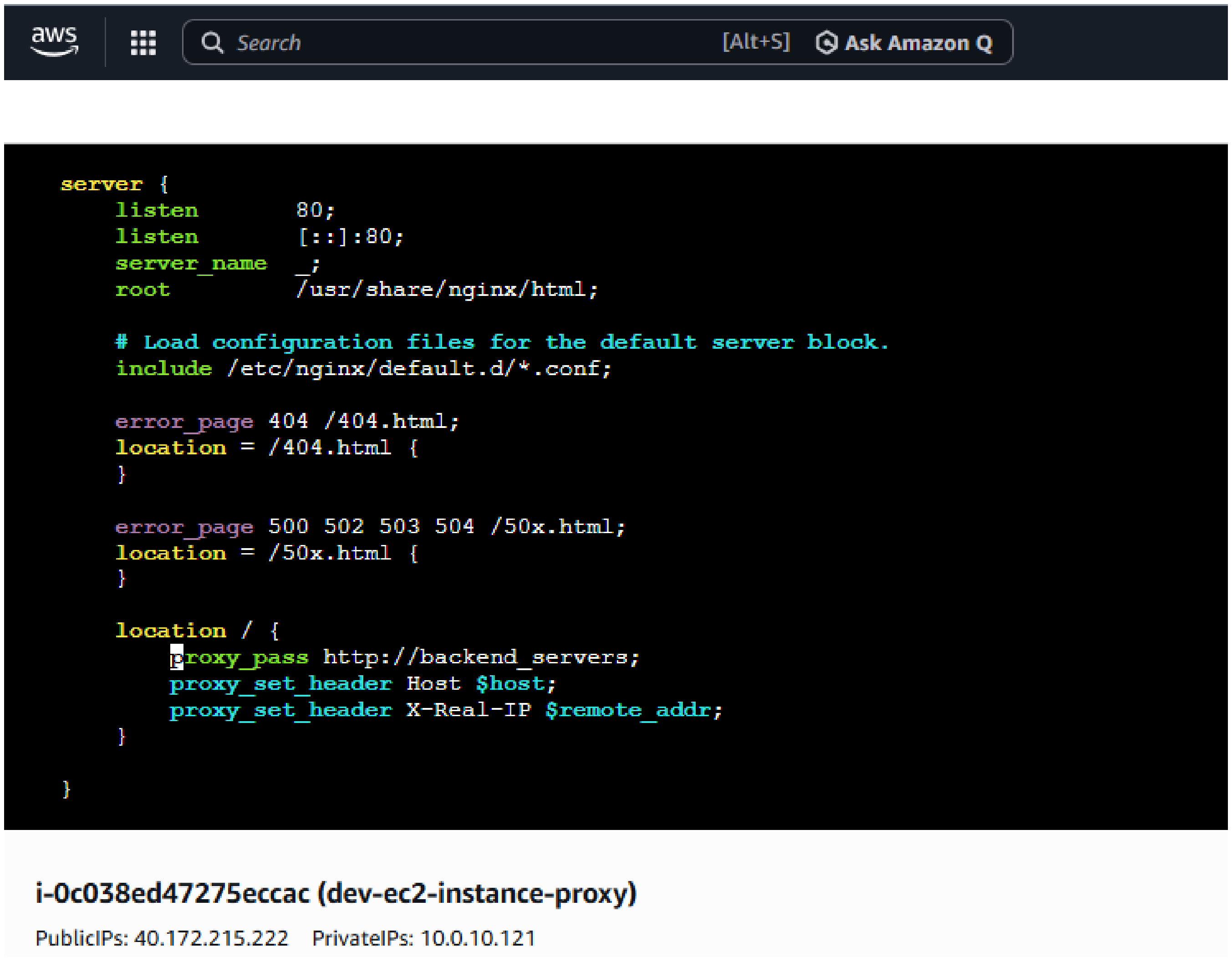


```
http {

    upstream backend_servers {
        server 51.112.229.177:80;
        server 3.28.131.119:80;
    }

    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;
    keepalive_timeout   65;
    types_hash_max_size 4096;

    include             /etc/nginx/mime.types;
    default_type        application/octet-stream;

    # Load modular configuration files from the /etc/nginx/conf.d directory.
    # See http://nginx.org/en/docs/ngx_core_module.html#include
    # for more information.
```



i-0c038ed47275eccac (dev-ec2-instance-proxy)

PublicIPs: 40.172.215.222   PrivateIPs: 10.0.10.121



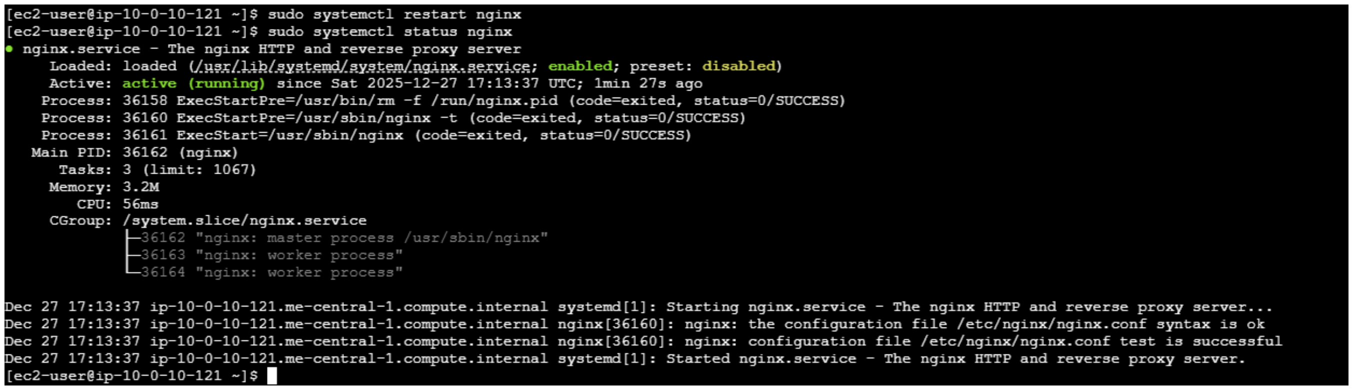
```
server {
    listen       80;
    listen       [::]:80;
    server_name  _;
    root         /usr/share/nginx/html;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    error_page 404 /404.html;
    location = /404.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }

    location / {
        proxy_pass http://backend_servers;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
    }

}
```

i-0c038ed47275eccac (dev-ec2-instance-proxy)

PublicIPs: 40.172.215.222   PrivateIPs: 10.0.10.121

7. Restart Nginx:

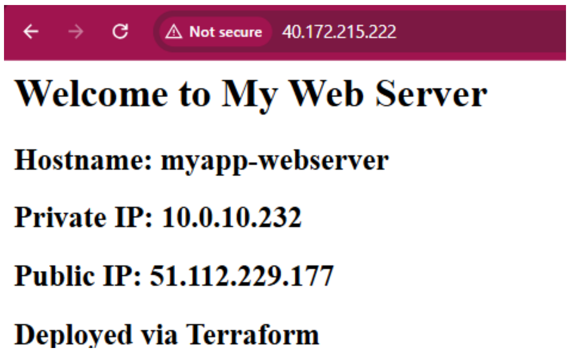
sudo systemctl restart nginx



```
[ec2-user@ip-10-0-10-121 ~]$ sudo systemctl restart nginx
[ec2-user@ip-10-0-10-121 ~]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
     Active: active (running) since Sat 2025-12-27 17:13:37 UTC; 1min 27s ago
    Process: 36158 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
    Process: 36160 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 36161 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 36162 (nginx)
      Tasks: 3 (limit: 1067)
     Memory: 3.2M
        CPU: 56ms
     CGroup: /system.slice/nginx.service
             ├─36162 "nginx: master process /usr/sbin/nginx"
             ├─36163 "nginx: worker process"
             └─36164 "nginx: worker process"

Dec 27 17:13:37 ip-10-0-10-121.me-central-1.compute.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Dec 27 17:13:37 ip-10-0-10-121.me-central-1.compute.internal nginx[36160]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Dec 27 17:13:37 ip-10-0-10-121.me-central-1.compute.internal nginx[36160]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Dec 27 17:13:37 ip-10-0-10-121.me-central-1.compute.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-10-0-10-121 ~]$
```

8. Test load balancing in browser:

Open browser and navigate to https://<webserver-public-ip>



Welcome to My Web Server

Hostname: myapp-webserver

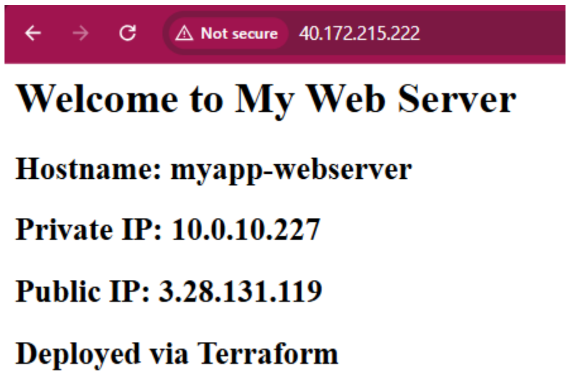
Private IP: 10.0.10.232

Public IP: 51.112.229.177

Deployed via Terraform

Reload the page multiple times

You should see the content alternating between web-1 and web-2 (check the hostname/IP in the page)



Welcome to My Web Server

Hostname: myapp-webserver

Private IP: 10.0.10.227

Public IP: 3.28.131.119

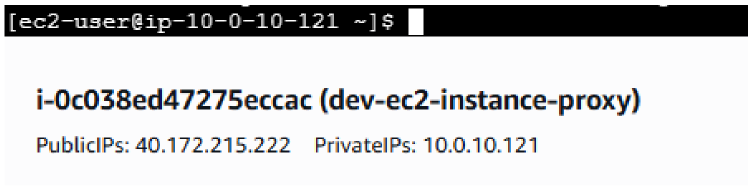
Deployed via Terraform

Task 9 — Configure high availability with backup servers

In this task, you will configure one server as primary and another as backup for high availability.

1. SSH into the webserver:

```
ssh ec2-user@<webserver-public-ip>
```



```
[ec2-user@ip-10-0-10-121 ~]$
```

i-0c038ed47275eccac (dev-ec2-instance-proxy)

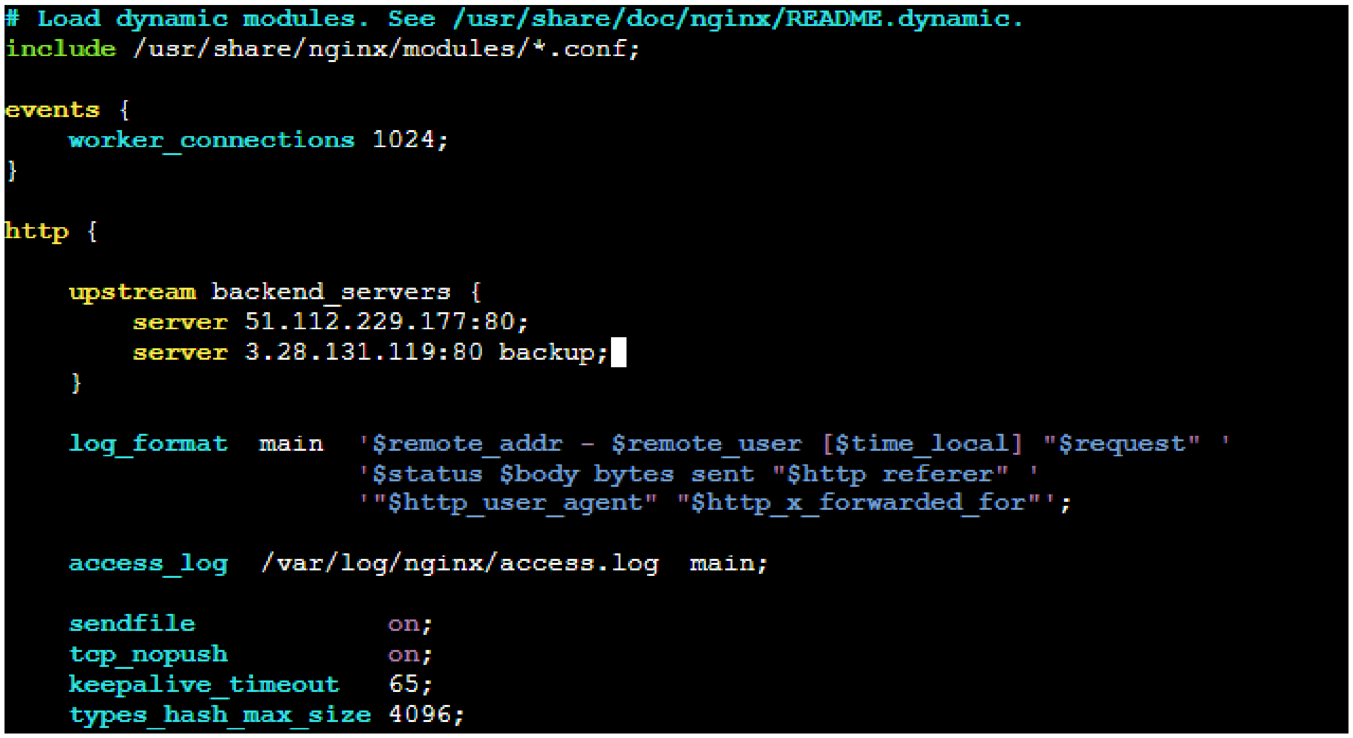
PublicIPs: 40.172.215.222 PrivateIPs: 10.0.10.121

2. Edit Nginx configuration for high availability:

```
sudo vim /etc/nginx/nginx.conf
```

Update the upstream block to make web-2 a backup:

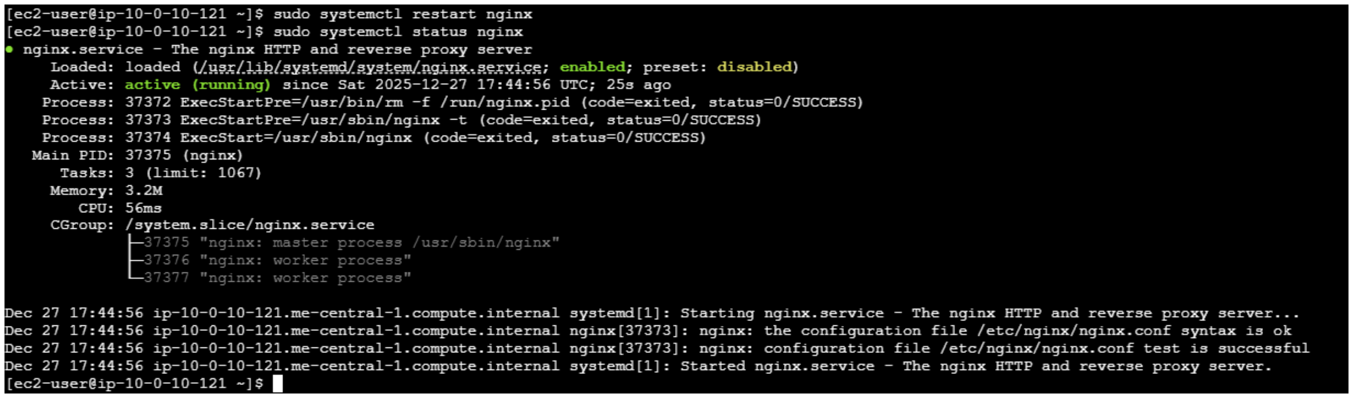
```
upstream backend_servers {
    server <web-1-public-ip>:80;
    server <web-2-public-ip>:80 backup;
}
```



```
# Load dynamic modules. See /usr/share/doc/nginx/README.dynamic.
include /usr/share/nginx/modules/*.conf;

events {
    worker_connections 1024;
}

http {

    upstream backend_servers {
        server 51.112.229.177:80;
        server 3.28.131.119:80 backup;
    }

    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;
    keepalive_timeout   65;
    types_hash_max_size 4096;
```

3. Restart Nginx:

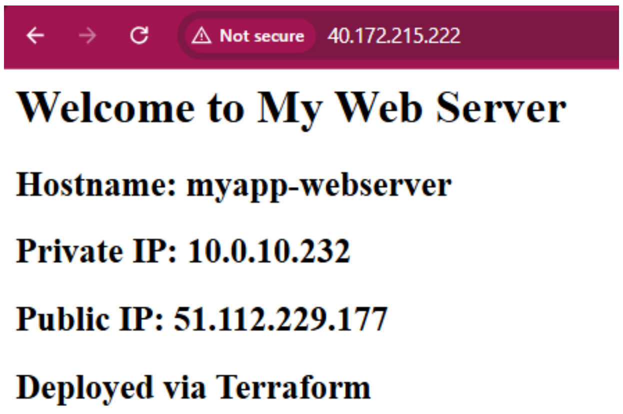
sudo systemctl restart nginx



```
[ec2-user@ip-10-0-10-121 ~]$ sudo systemctl restart nginx
[ec2-user@ip-10-0-10-121 ~]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
     Active: active (running) since Sat 2025-12-27 17:44:56 UTC; 25s ago
    Process: 37372 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
    Process: 37373 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 37374 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 37375 (nginx)
      Tasks: 3 (limit: 1067)
     Memory: 3.2M
        CPU: 56ms
     CGroup: /system.slice/nginx.service
             ├─37375 "nginx: master process /usr/sbin/nginx"
             ├─37376 "nginx: worker process"
             └─37377 "nginx: worker process"

Dec 27 17:44:56 ip-10-0-10-121.me-central-1.compute.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Dec 27 17:44:56 ip-10-0-10-121.me-central-1.compute.internal nginx[37373]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Dec 27 17:44:56 ip-10-0-10-121.me-central-1.compute.internal nginx[37373]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Dec 27 17:44:56 ip-10-0-10-121.me-central-1.compute.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-10-0-10-121 ~]$
```

4. Test in browser:

Open browser and navigate to https://<webserver-public-ip>



Welcome to My Web Server

Hostname: myapp-webserver

Private IP: 10.0.10.232

Public IP: 51.112.229.177

Deployed via Terraform

Reload multiple times

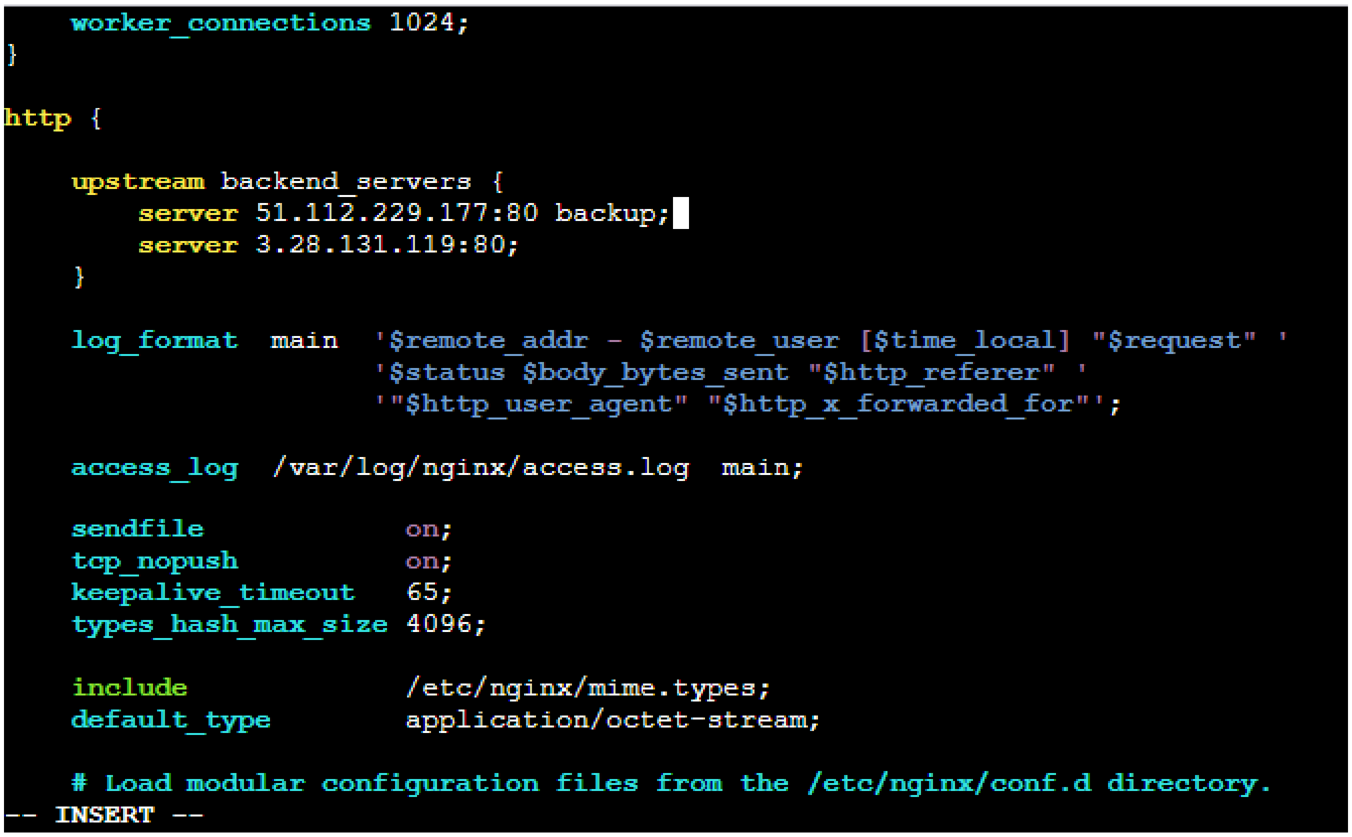
You should ONLY see web-1 (primary server)

5. Switch backup configuration:

sudo vim /etc/nginx/nginx.conf

Update to make web-1 backup:

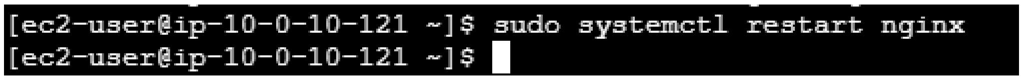
```
upstream backend_servers {
    server <web-1-public-ip>: 80 backup;
    server <web-2-public-ip>:80;
}
```



```
    worker_connections 1024;
}

http {

    upstream backend_servers {
        server 51.112.229.177:80 backup;
        server 3.28.131.119:80;
    }

    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;
    keepalive_timeout   65;
    types_hash_max_size 4096;

    include             /etc/nginx/mime.types;
    default_type        application/octet-stream;

    # Load modular configuration files from the /etc/nginx/conf.d directory.
-- INSERT --
```

6. Restart Nginx:

sudo systemctl restart nginx

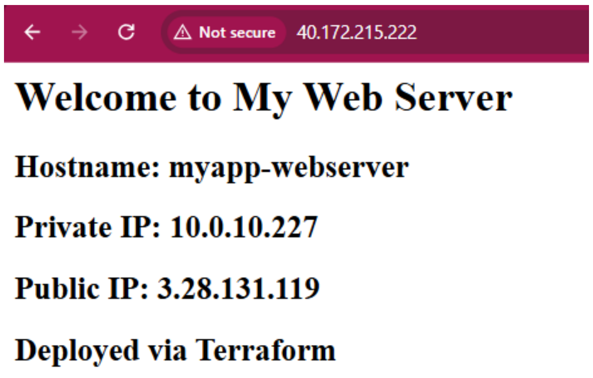


```
[ec2-user@ip-10-0-10-121 ~]$ sudo systemctl restart nginx
[ec2-user@ip-10-0-10-121 ~]$
```

7. Test in browser:

Reload multiple times

You should ONLY see web-2 (now the primary server)



Welcome to My Web Server

Hostname: myapp-webserver

Private IP: 10.0.10.227

Public IP: 3.28.131.119

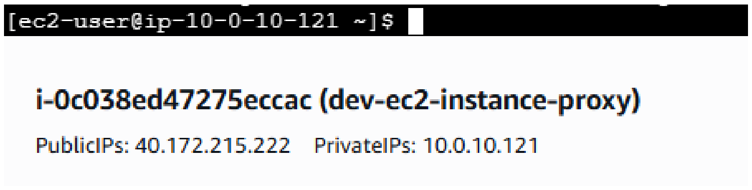
Deployed via Terraform

Task 10 — Enable Nginx caching

In this task, you will enable caching in Nginx to improve performance.

1. SSH into the webserver:

ssh ec2-user@<webserver-public-ip>



```
[ec2-user@ip-10-0-10-121 ~]$
```

i-0c038ed47275eccac (dev-ec2-instance-proxy)

PublicIPs: 40.172.215.222   PrivateIPs: 10.0.10.121

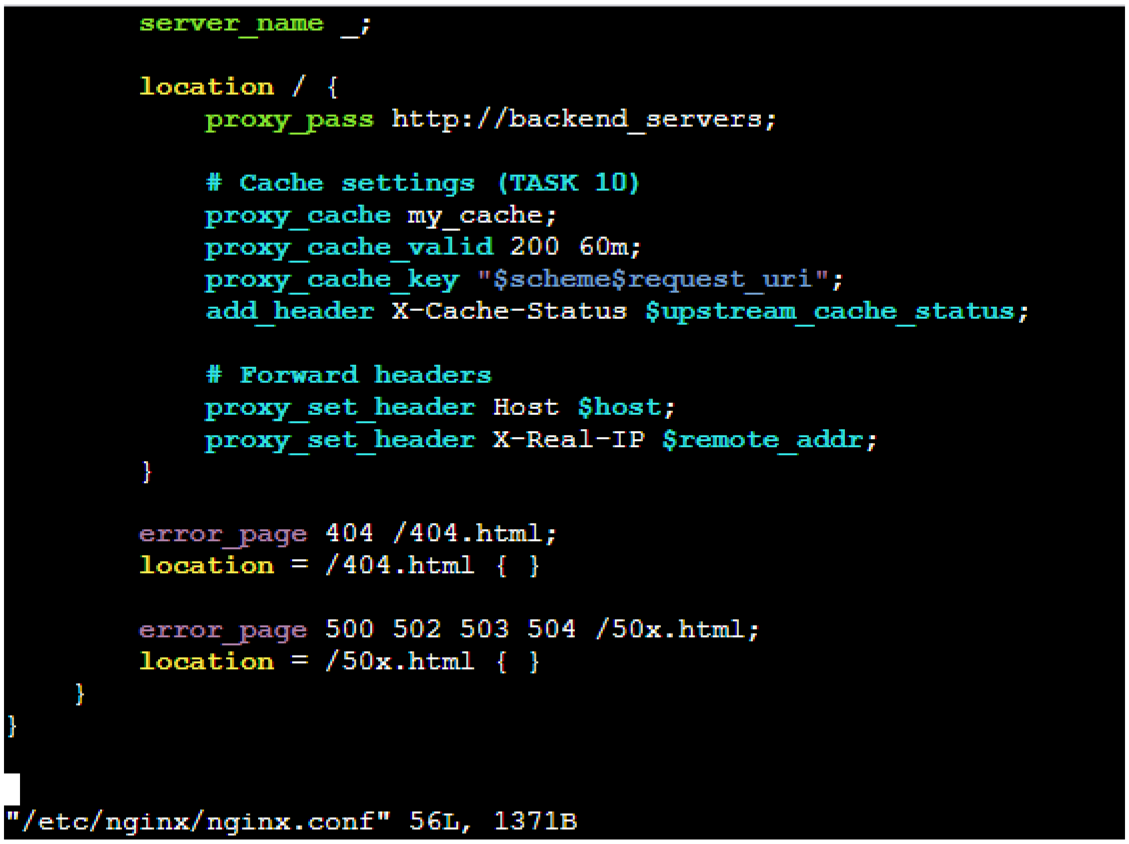
2. Edit Nginx configuration to enable caching:

sudo vim /etc/nginx/nginx.conf

Add proxy cache configuration in the http block and location block:

```
http {
   proxy_cache_path /var/cache/nginx levels=1:2 keys_zone=my_cache:10m inactive=60m max_size=1g;
   log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                     '$status $body_bytes_sent "$http_referer" '
                     '"$http_user_agent" "$http_x_forwarded_for"';
   # ... other settings ...
   upstream backend_servers {
      server <web-1-public-ip>:80;
      server <web-2-public-ip>: 80;
   }
   server {
      listen 443 ssl;
      server_name $PUBLIC_IP;
```

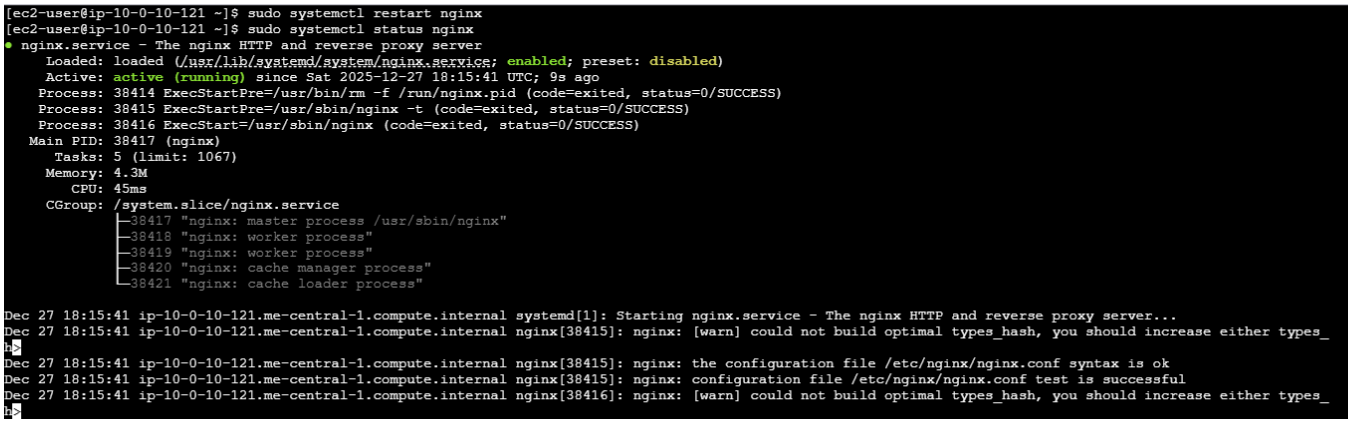
```
        ssl_certificate /etc/ssl/certs/selfsigned.crt;

        ssl_certificate_key /etc/ssl/private/selfsigned.key;

        location / {

    #       root /usr/share/nginx/html;

    #       index index.html;

    #       proxy_pass http://<web-1-public-ip>: 80;

          proxy_pass http://backend_servers;

          proxy_cache my_cache;

          proxy_cache_valid 200 60m;

          proxy_cache_key "$scheme$request_uri";

          add_header X-Cache-Status $upstream_cache_status;

        }

    }

    # ... rest of config ...

}
```



```
        server_name _;

        location / {
            proxy_pass http://backend_servers;

            # Cache settings (TASK 10)
            proxy_cache my_cache;
            proxy_cache_valid 200 60m;
            proxy_cache_key "$scheme$request_uri";
            add_header X-Cache-Status $upstream_cache_status;

            # Forward headers
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
        }

        error_page 404 /404.html;
        location = /404.html { }

        error_page 500 502 503 504 /50x.html;
        location = /50x.html { }
    }
}

"/etc/nginx/nginx.conf" 56L, 1371B
```

3. Restart Nginx:

```
sudo systemctl restart nginx
```



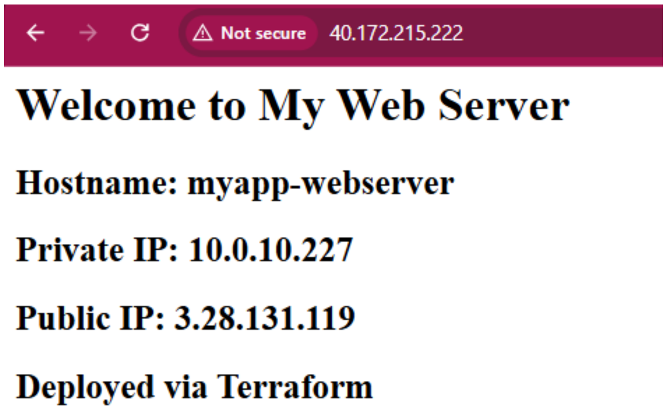
```
[ec2-user@ip-10-0-10-121 ~]$ sudo systemctl restart nginx
[ec2-user@ip-10-0-10-121 ~]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
     Active: active (running) since Sat 2025-12-27 18:15:41 UTC; 9s ago
    Process: 38414 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
    Process: 38415 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 38416 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 38417 (nginx)
      Tasks: 5 (limit: 1067)
     Memory: 4.3M
        CPU: 45ms
     CGroup: /system.slice/nginx.service
             ├─38417 "nginx: master process /usr/sbin/nginx"
             ├─38418 "nginx: worker process"
             ├─38419 "nginx: worker process"
             ├─38420 "nginx: cache manager process"
             └─38421 "nginx: cache loader process"

Dec 27 18:15:41 ip-10-0-10-121.me-central-1.compute.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Dec 27 18:15:41 ip-10-0-10-121.me-central-1.compute.internal nginx[38415]: nginx: [warn] could not build optimal types_hash, you should increase either types_
h>
Dec 27 18:15:41 ip-10-0-10-121.me-central-1.compute.internal nginx[38415]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Dec 27 18:15:41 ip-10-0-10-121.me-central-1.compute.internal nginx[38415]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Dec 27 18:15:41 ip-10-0-10-121.me-central-1.compute.internal nginx[38416]: nginx: [warn] could not build optimal types_hash, you should increase either types_
h>
```

4. Test caching in browser:

Open browser developer tools (F12)

Navigate to Network tab

Visit https://<webserver-public-ip>



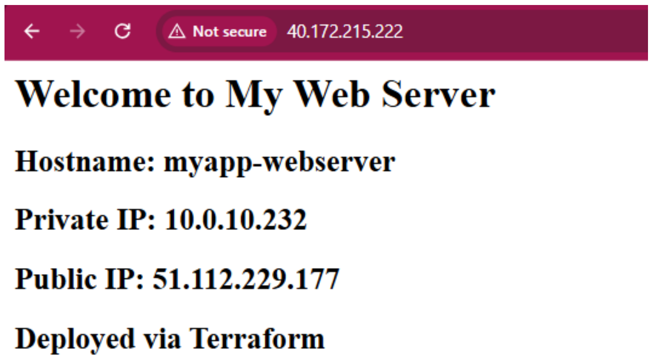
Welcome to My Web Server

Hostname: myapp-webserver

Private IP: 10.0.10.227

Public IP: 3.28.131.119

Deployed via Terraform



Welcome to My Web Server

Hostname: myapp-webserver

Private IP: 10.0.10.232

Public IP: 51.112.229.177

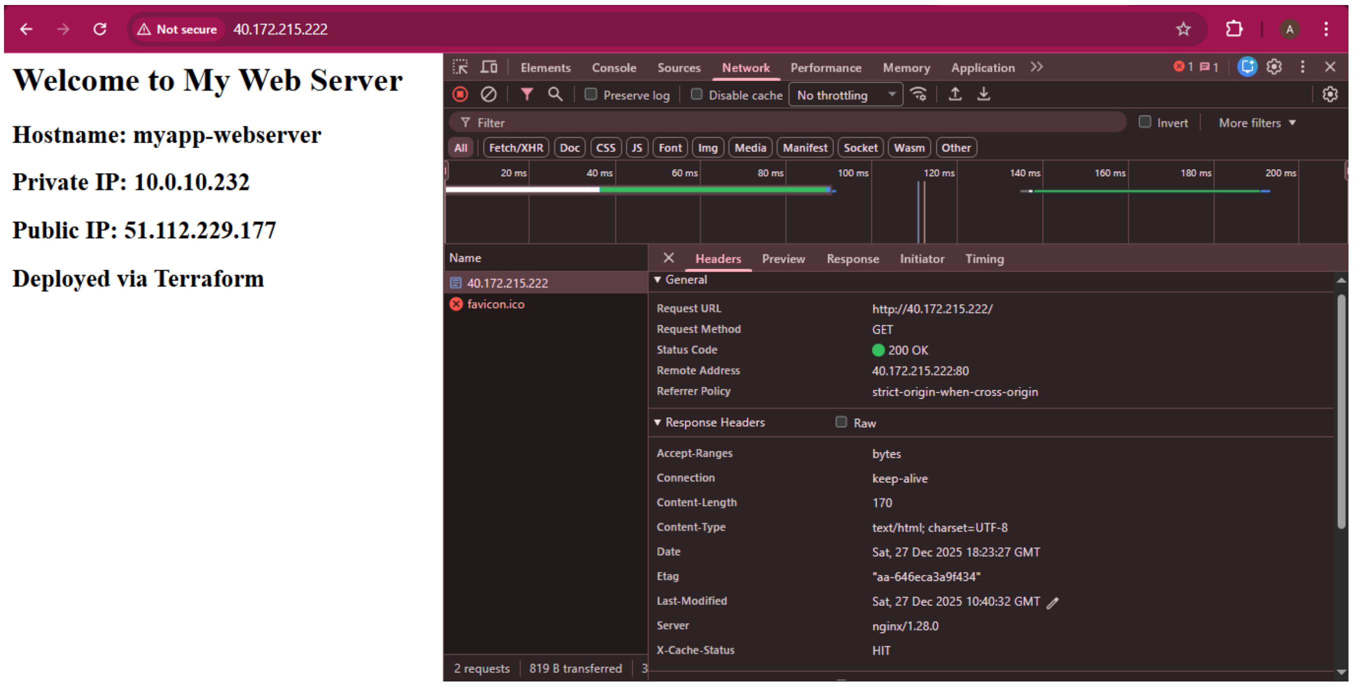
Deployed via Terraform

Check response headers for X-Cache-Status

First request should show MISS

Reload the page

Second request should show HIT



Not secure 40.172.215.222

Welcome to My Web Server

Hostname: myapp-webserver

Private IP: 10.0.10.232

Public IP: 51.112.229.177

Deployed via Terraform

Name: 40.172.215.222, favicon.ico

General

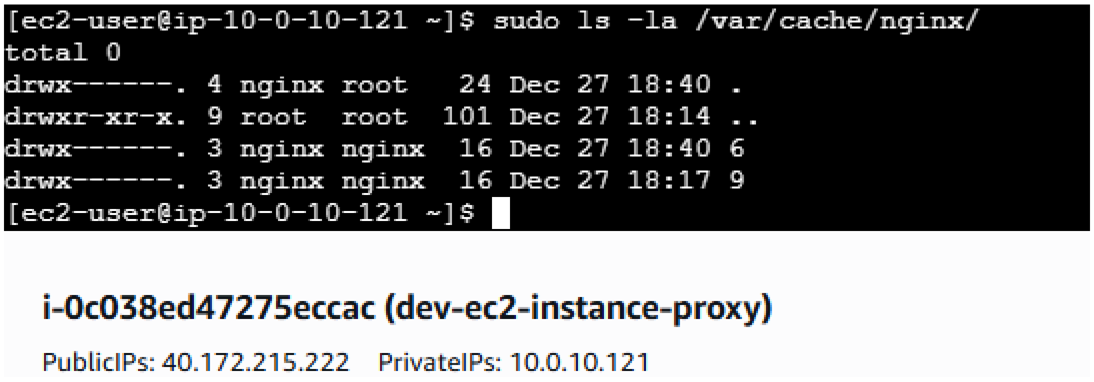
Request URL	http://40.172.215.222/
Request Method	GET
Status Code	200 OK
Remote Address	40.172.215.222:80
Referrer Policy	strict-origin-when-cross-origin

Response Headers

Accept-Ranges	bytes
Connection	keep-alive
Content-Length	170
Content-Type	text/html; charset=UTF-8
Date	Sat, 27 Dec 2025 18:23:27 GMT
Etag	"aa-646eca3a9f434"
Last-Modified	Sat, 27 Dec 2025 10:40:32 GMT
Server	nginx/1.28.0
X-Cache-Status	HIT

5. Verify cache directory:

ls -la /var/cache/nginx/



```
[ec2-user@ip-10-0-10-121 ~]$ sudo ls -la /var/cache/nginx/
total 0
drwx------. 4 nginx root   24 Dec 27 18:40 .
drwxr-xr-x. 9 root  root  101 Dec 27 18:14 ..
drwx------. 3 nginx nginx  16 Dec 27 18:40 6
drwx------. 3 nginx nginx  16 Dec 27 18:17 9
[ec2-user@ip-10-0-10-121 ~]$
```

i-0c038ed47275eccac (dev-ec2-instance-proxy)

PublicIPs: 40.172.215.222 PrivateIPs: 10.0.10.121

Cleanup

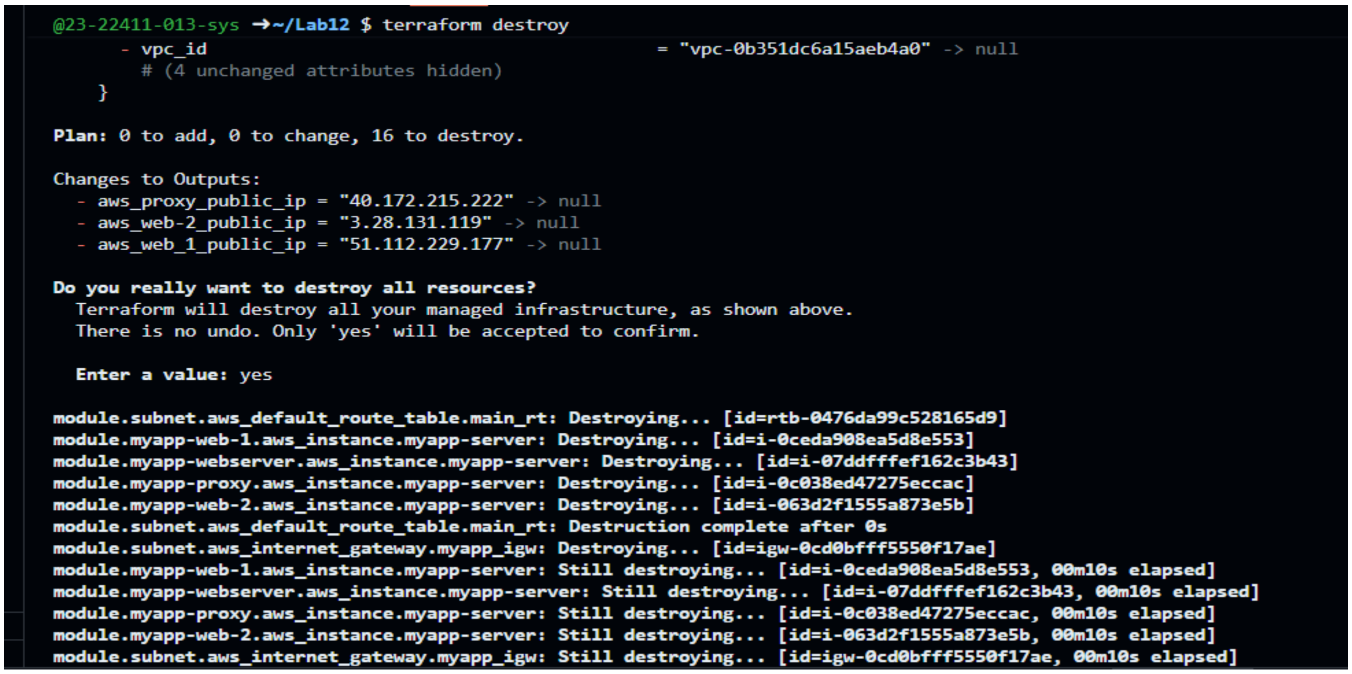
1. Exit SSH session:

exit

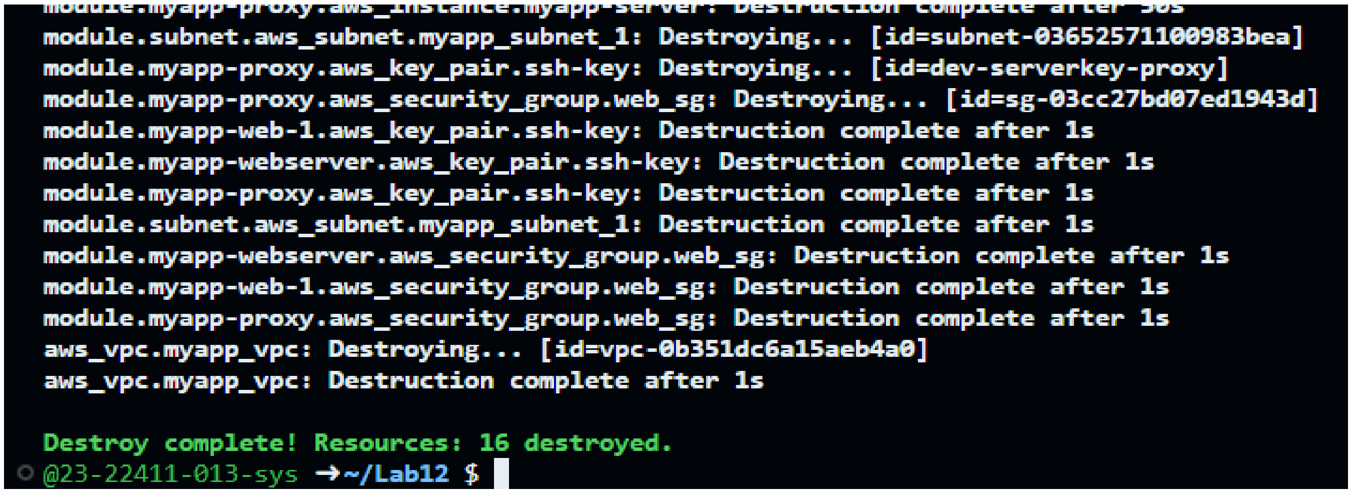
2. Destroy all resources:

terraform destroy

- Type yes when prompted for confirmation.



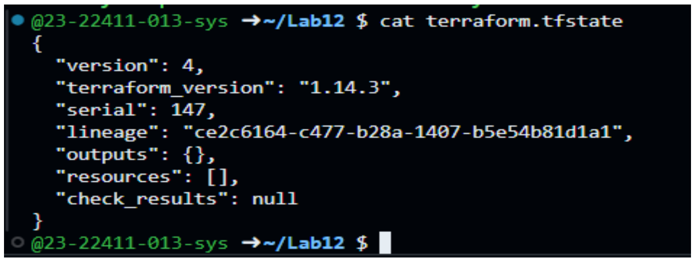
```
@23-22411-013-sys ➜~/Lab12 $ terraform destroy
      - vpc_id                                   = "vpc-0b351dc6a15aeb4a0" -> null
        # (4 unchanged attributes hidden)
    }

Plan: 0 to add, 0 to change, 16 to destroy.

Changes to Outputs:
  - aws_proxy_public_ip = "40.172.215.222" -> null
  - aws_web-2_public_ip = "3.28.131.119" -> null
  - aws_web_1_public_ip = "51.112.229.177" -> null

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

module.subnet.aws_default_route_table.main_rt: Destroying... [id=rtb-0476da99c528165d9]
module.myapp-web-1.aws_instance.myapp-server: Destroying... [id=i-0ceda908ea5d8e553]
module.myapp-webserver.aws_instance.myapp-server: Destroying... [id=i-07ddfffef162c3b43]
module.myapp-proxy.aws_instance.myapp-server: Destroying... [id=i-0c038ed47275eccac]
module.myapp-web-2.aws_instance.myapp-server: Destroying... [id=i-063d2f1555a873e5b]
module.subnet.aws_default_route_table.main_rt: Destruction complete after 0s
module.subnet.aws_internet_gateway.myapp_igw: Destroying... [id=igw-0cd0bfff5550f17ae]
module.myapp-web-1.aws_instance.myapp-server: Still destroying... [id=i-0ceda908ea5d8e553, 00m10s elapsed]
module.myapp-webserver.aws_instance.myapp-server: Still destroying... [id=i-07ddfffef162c3b43, 00m10s elapsed]
module.myapp-proxy.aws_instance.myapp-server: Still destroying... [id=i-0c038ed47275eccac, 00m10s elapsed]
module.myapp-web-2.aws_instance.myapp-server: Still destroying... [id=i-063d2f1555a873e5b, 00m10s elapsed]
module.subnet.aws_internet_gateway.myapp_igw: Still destroying... [id=igw-0cd0bfff5550f17ae, 00m10s elapsed]
```



```
module.myapp-proxy.aws_instance.myapp-server: Destruction complete after 30s
module.subnet.aws_subnet.myapp_subnet_1: Destroying... [id=subnet-03652571100983bea]
module.myapp-proxy.aws_key_pair.ssh-key: Destroying... [id=dev-serverkey-proxy]
module.myapp-proxy.aws_security_group.web_sg: Destroying... [id=sg-03cc27bd07ed1943d]
module.myapp-web-1.aws_key_pair.ssh-key: Destruction complete after 1s
module.myapp-webserver.aws_key_pair.ssh-key: Destruction complete after 1s
module.myapp-proxy.aws_key_pair.ssh-key: Destruction complete after 1s
module.subnet.aws_subnet.myapp_subnet_1: Destruction complete after 1s
module.myapp-webserver.aws_security_group.web_sg: Destruction complete after 1s
module.myapp-web-1.aws_security_group.web_sg: Destruction complete after 1s
module.myapp-proxy.aws_security_group.web_sg: Destruction complete after 1s
aws_vpc.myapp_vpc: Destroying... [id=vpc-0b351dc6a15aeb4a0]
aws_vpc.myapp_vpc: Destruction complete after 1s

Destroy complete! Resources: 16 destroyed.
@23-22411-013-sys ➜~/Lab12 $
```

3. Verify state files:

cat terraform.tfstate



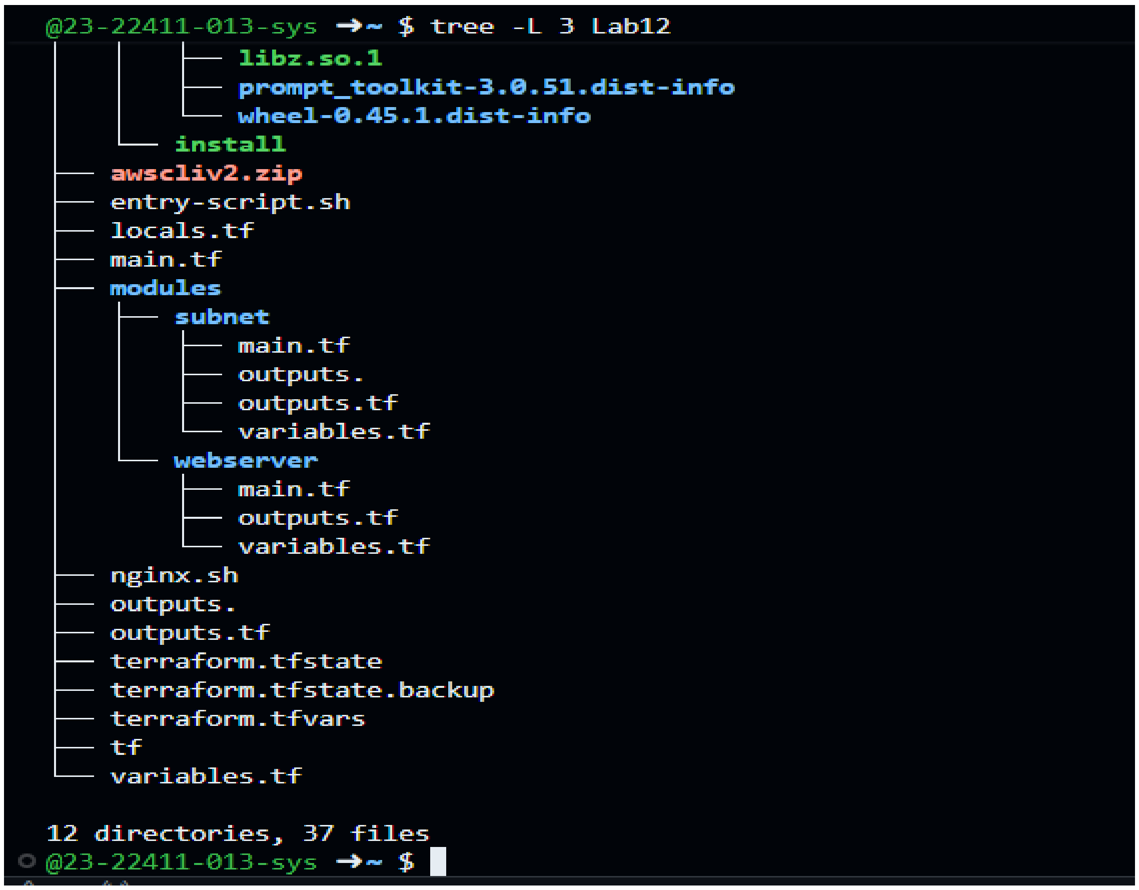
```
@23-22411-013-sys ➜~/Lab12 $ cat terraform.tfstate
{
  "version": 4,
  "terraform_version": "1.14.3",
  "serial": 147,
  "lineage": "ce2c6164-c477-b28a-1407-b5e54b81d1a1",
  "outputs": {},
  "resources": [],
  "check_results": null
}
@23-22411-013-sys ➜~/Lab12 $
```

4. List all project files:

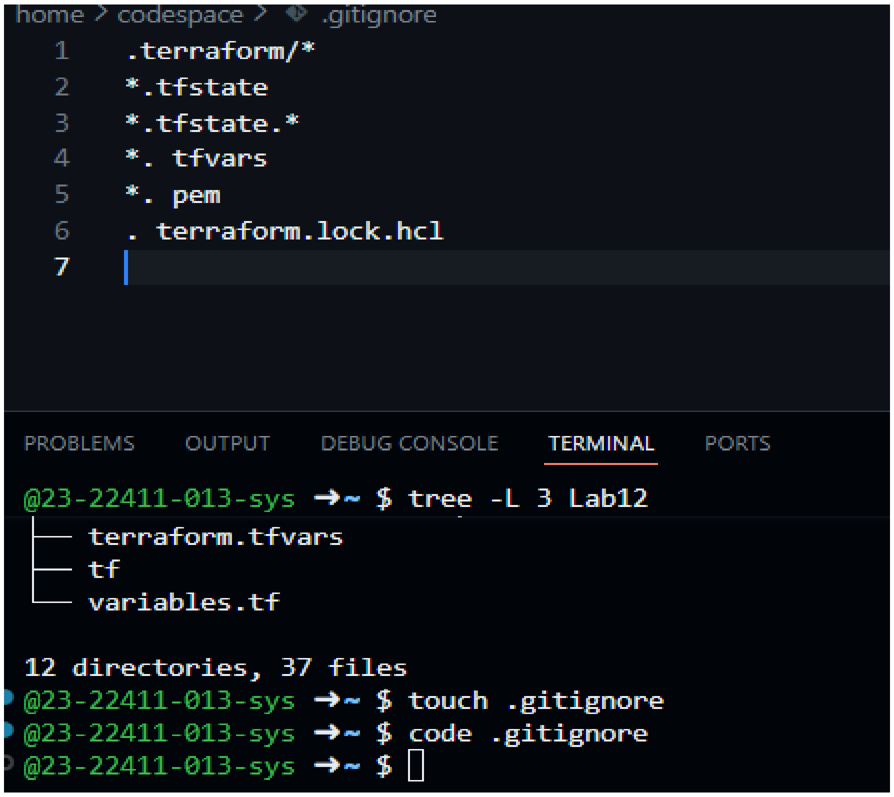
tree

or

ls -la



```
@23-22411-013-sys ➜~ $ tree -L 3 Lab12
│       │   ├── libz.so.1
│       │   ├── prompt_toolkit-3.0.51.dist-info
│       │   └── wheel-0.45.1.dist-info
│       └── install
├── awscliv2.zip
├── entry-script.sh
├── locals.tf
├── main.tf
├── modules
│   ├── subnet
│   │   ├── main.tf
│   │   ├── outputs.
│   │   ├── outputs.tf
│   │   └── variables.tf
│   └── webserver
│       ├── main.tf
│       ├── outputs.tf
│       └── variables.tf
├── nginx.sh
├── outputs.
├── outputs.tf
├── terraform.tfstate
├── terraform.tfstate.backup
├── terraform.tfvars
├── tf
└── variables.tf

12 directories, 37 files
@23-22411-013-sys ➜~ $
```



home > codespace > .gitignore

```
.terraform/*
*.tfstate
*.tfstate.*
*. tfvars
*. pem
. terraform.lock.hcl

```

```
@23-22411-013-sys ➜~ $ tree -L 3 Lab12
├── terraform.tfvars
├── tf
└── variables.tf

12 directories, 37 files
@23-22411-013-sys ➜~ $ touch .gitignore
@23-22411-013-sys ➜~ $ code .gitignore
@23-22411-013-sys ➜~ $
```

.