CLOUD COMPUTING

LAB 12

Submitted To:

Engr. Shoaib

Submitted By:

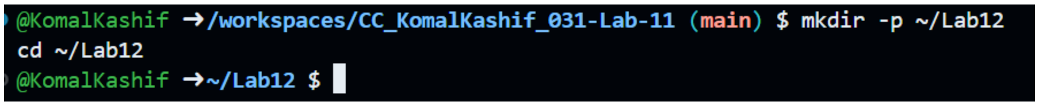
Komal Kashif

BSE V-A

2023-BSE-031

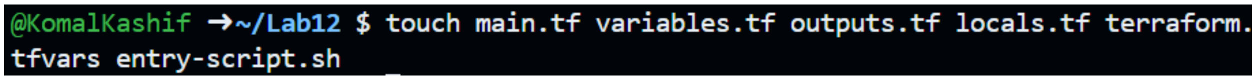
Task 1 — Organize Terraform code into separate files

task1_project_directory.png



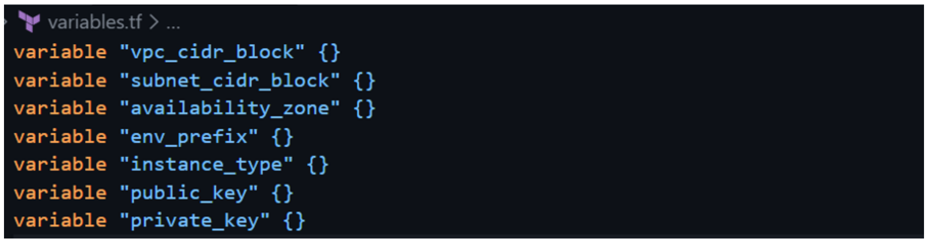
```
@KomalKashif ➜/workspaces/CC_KomalKashif_031-Lab-11 (main) $ mkdir -p ~/Lab12
cd ~/Lab12
@KomalKashif ➜~/Lab12 $
```

task1_files_created.png



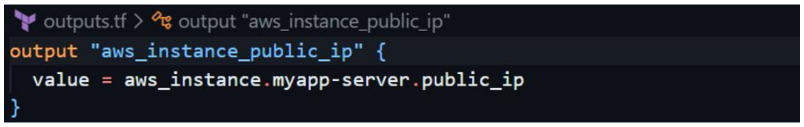
```
@KomalKashif ➜~/Lab12 $ touch main.tf variables.tf outputs.tf locals.tf terraform.
tfvars entry-script.sh
```

task1_variables_tf.png



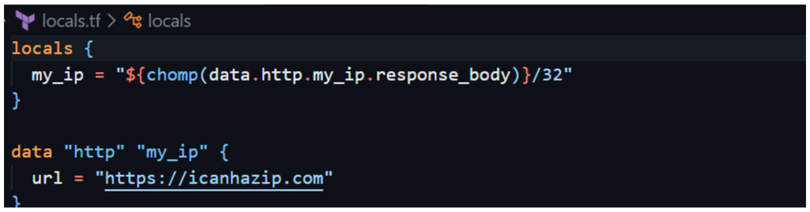
```
variable "vpc_cidr_block" {}
variable "subnet_cidr_block" {}
variable "availability_zone" {}
variable "env_prefix" {}
variable "instance_type" {}
variable "public_key" {}
variable "private_key" {}
```

task1_outputs_tf.png



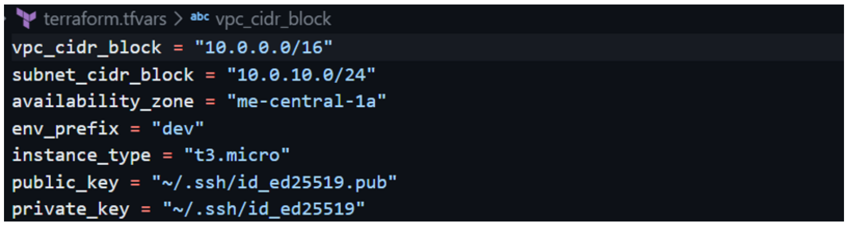
```
output "aws_instance_public_ip" {
  value = aws_instance.myapp-server.public_ip
}
```

task1_locals_tf.png



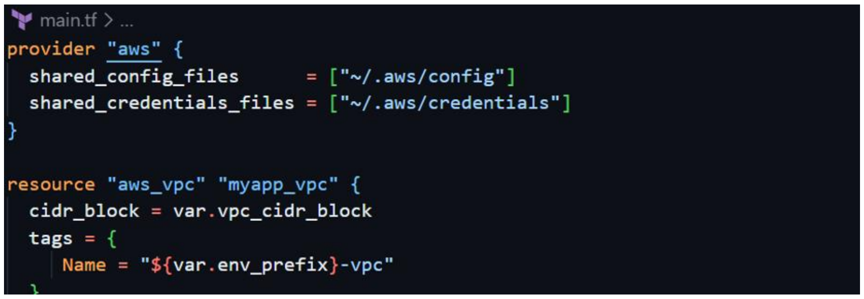
```
locals {
  my_ip = "${chomp(data.http.my_ip.response_body)}/32"
}

data "http" "my_ip" {
  url = "https://icanhazip.com"
}
```

task1_terraform_tfvars.png



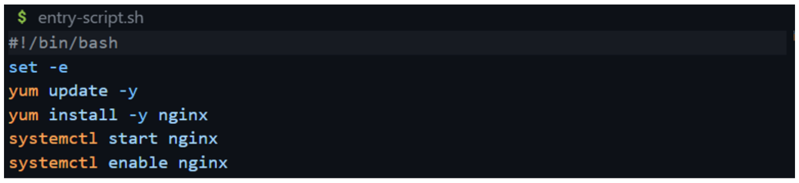
```
vpc_cidr_block = "10.0.0.0/16"
subnet_cidr_block = "10.0.10.0/24"
availability_zone = "me-central-1a"
env_prefix = "dev"
instance_type = "t3.micro"
public_key = "~/.ssh/id_ed25519.pub"
private_key = "~/.ssh/id_ed25519"
```

task1_main_tf.png



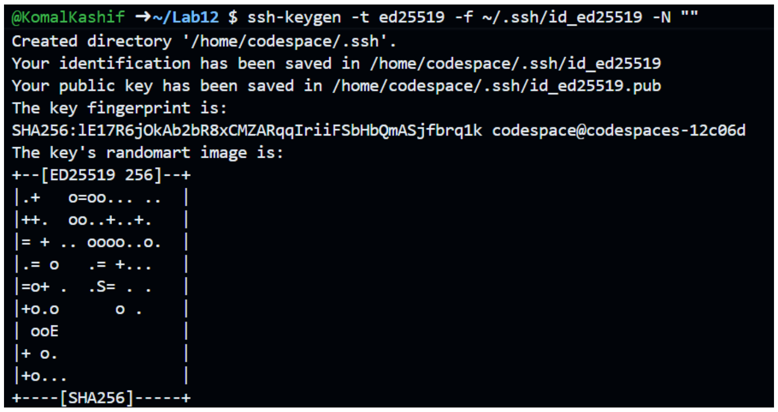
```
main.tf > ...
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}

resource "aws_vpc" "myapp_vpc" {
  cidr_block = var.vpc_cidr_block
  tags = {
    Name = "${var.env_prefix}-vpc"
```

task1_entry_script.png



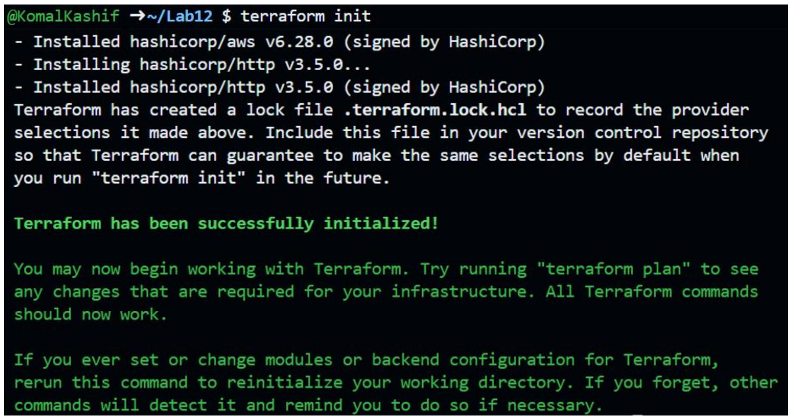
```
entry-script.sh
#!/bin/bash
set -e
yum update -y
yum install -y nginx
systemctl start nginx
systemctl enable nginx
```

task1_ssh_keygen.png



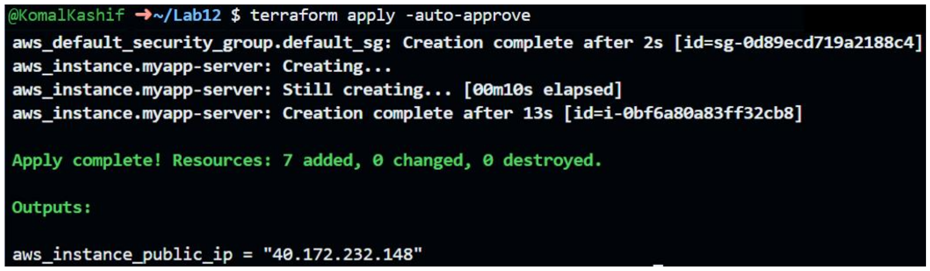
```
@KomalKashif ➜~/Lab12 $ ssh-keygen -t ed25519 -f ~/.ssh/id_ed25519 -N ""
Created directory '/home/codespace/.ssh'.
Your identification has been saved in /home/codespace/.ssh/id_ed25519
Your public key has been saved in /home/codespace/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:lE17R6jOkAb2bR8xCMZARqqIriiFSbHbQmASjfbrq1k codespace@codespaces-12c06d
The key's randomart image is:
+--[ED25519 256]--+
|.+   o=oo... ..  |
|++.  oo..+..+.   |
|= + .. oooo..o.  |
|.= o   .= +...   |
|=o+ .  .S= . .   |
|+o.o      o .    |
| ooE             |
|+ o.             |
|+o...            |
+----[SHA256]-----+
```

task1_terraform_init.png



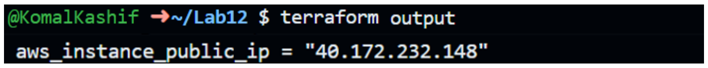
```
@KomalKashif ➜~/Lab12 $ terraform init
- Installed hashicorp/aws v6.28.0 (signed by HashiCorp)
- Installing hashicorp/http v3.5.0...
- Installed hashicorp/http v3.5.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include 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.
```

task1_terraform_apply.png



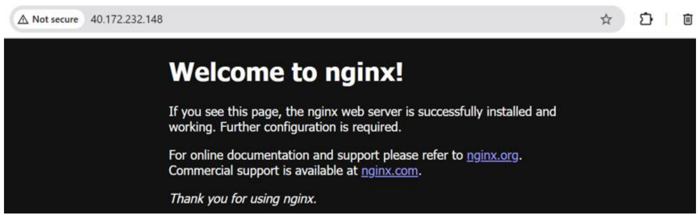
```
@KomalKashif ➜~/Lab12 $ terraform apply -auto-approve
aws_default_security_group.default_sg: Creation complete after 2s [id=sg-0d89ecd719a2188c4]
aws_instance.myapp-server: Creating...
aws_instance.myapp-server: Still creating... [00m10s elapsed]
aws_instance.myapp-server: Creation complete after 13s [id=i-0bf6a80a83ff32cb8]

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

Outputs:

aws_instance_public_ip = "40.172.232.148"
```

task1_terraform_output.png



```
@KomalKashif ➜~/Lab12 $ terraform output
aws_instance_public_ip = "40.172.232.148"
```

task1_nginx_browser.png



Not secure 40.172.232.148

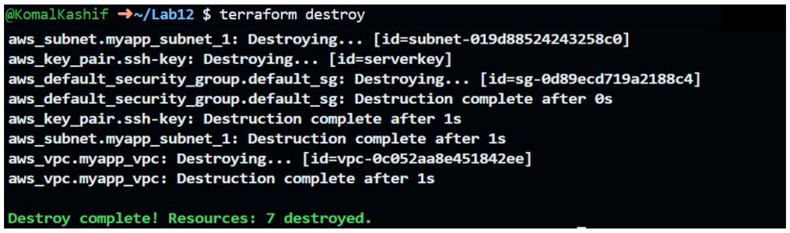
Welcome to nginx!

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.

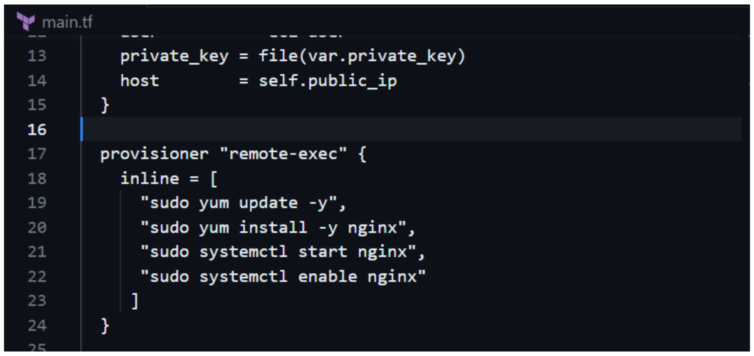
task1_terraform_destroy.png



```
@KomalKashif ➜~/Lab12 $ terraform destroy
aws_subnet.myapp_subnet_1: Destroying... [id=subnet-019d88524243258c0]
aws_key_pair.ssh-key: Destroying... [id=serverkey]
aws_default_security_group.default_sg: Destroying... [id=sg-0d89ecd719a2188c4]
aws_default_security_group.default_sg: Destruction complete after 0s
aws_key_pair.ssh-key: Destruction complete after 1s
aws_subnet.myapp_subnet_1: Destruction complete after 1s
aws_vpc.myapp_vpc: Destroying... [id=vpc-0c052aa8e451842ee]
aws_vpc.myapp_vpc: Destruction complete after 1s

Destroy complete! Resources: 7 destroyed.
```

Task 2 — Use remote-exec provisioner

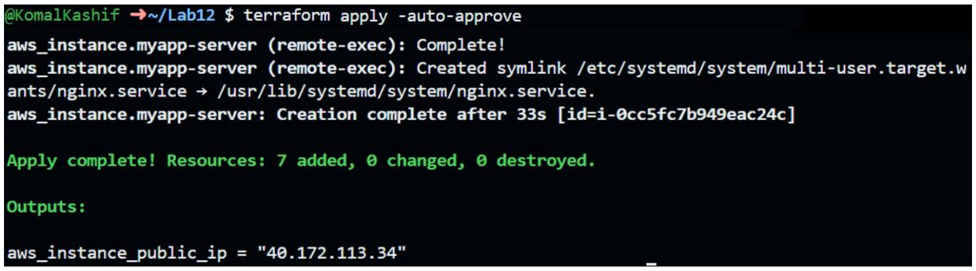
terraform apply -auto-approve



main.tf

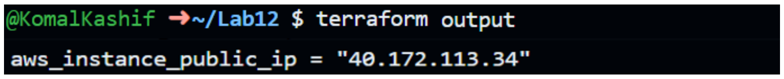
```
    private_key = file(var.private_key)
    host        = self.public_ip
  }

  provisioner "remote-exec" {
    inline = [
      "sudo yum update -y",
      "sudo yum install -y nginx",
      "sudo systemctl start nginx",
      "sudo systemctl enable nginx"
     ]
  }
```

task2_terraform_apply.png



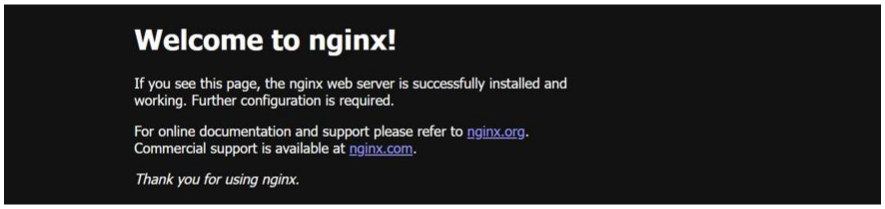
```
@KomalKashif ➜~/Lab12 $ terraform apply -auto-approve
aws_instance.myapp-server (remote-exec): Complete!
aws_instance.myapp-server (remote-exec): Created symlink /etc/systemd/system/multi-user.target.w
ants/nginx.service → /usr/lib/systemd/system/nginx.service.
aws_instance.myapp-server: Creation complete after 33s [id=i-0cc5fc7b949eac24c]

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

Outputs:

aws_instance_public_ip = "40.172.113.34"
```

task2_terraform_output.png



```
@KomalKashif ➜~/Lab12 $ terraform output
aws_instance_public_ip = "40.172.113.34"
```

task2_nginx_browser.png



Welcome to nginx!

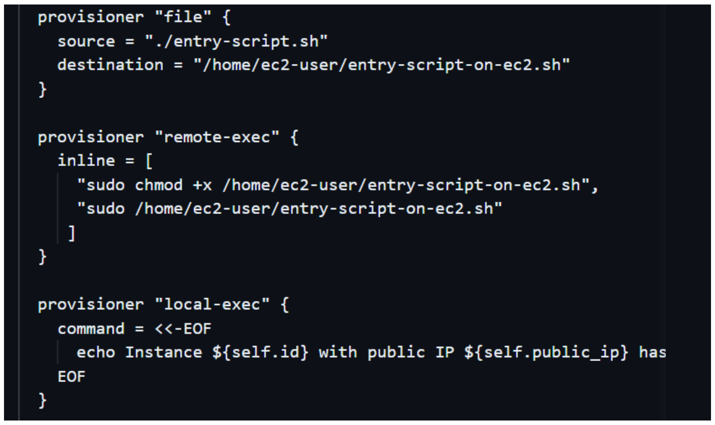
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

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

Thank you for using nginx.

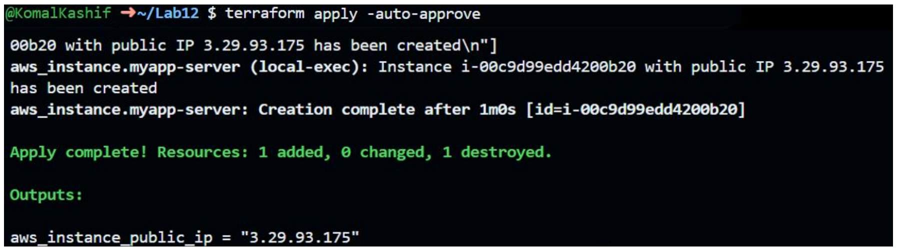
Task 3 — Use file and local-exec provisioners

task3_main_tf_all_provisioners.png



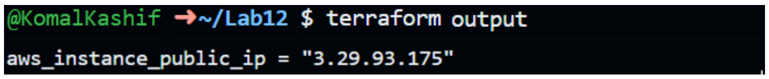
```
provisioner "file" {
  source = "./entry-script.sh"
  destination = "/home/ec2-user/entry-script-on-ec2.sh"
}

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
  EOF
}
```

task3_terraform_apply.png



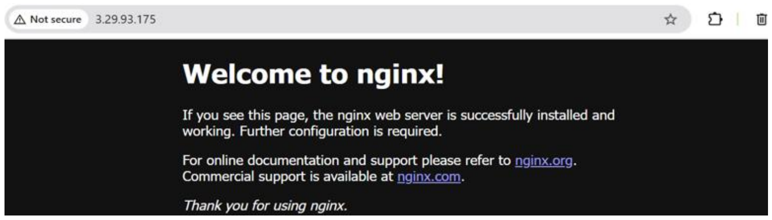
```
@KomalKashif ➜~/Lab12 $ terraform apply -auto-approve
00b20 with public IP 3.29.93.175 has been created\n"]
aws_instance.myapp-server (local-exec): Instance i-00c9d99edd4200b20 with public IP 3.29.93.175
has been created
aws_instance.myapp-server: Creation complete after 1m0s [id=i-00c9d99edd4200b20]

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

Outputs:

aws_instance_public_ip = "3.29.93.175"
```

task3_terraform_output.png



```
@KomalKashif ➜~/Lab12 $ terraform output
aws_instance_public_ip = "3.29.93.175"
```

task3_nginx_browser.png



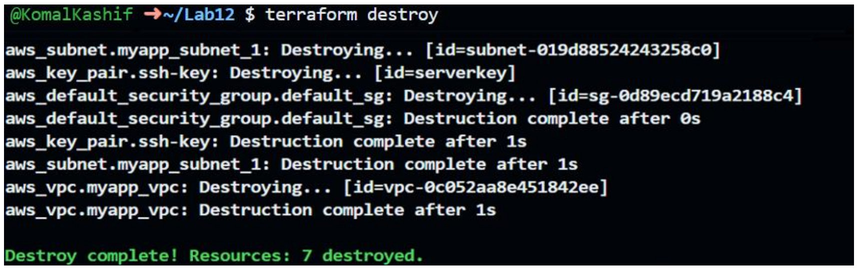
Welcome to nginx!

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.

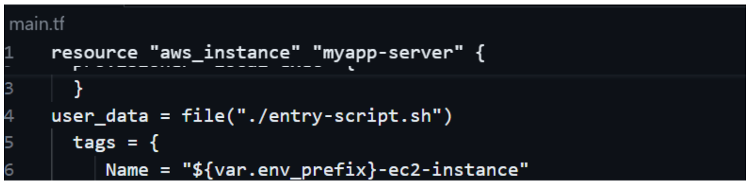
Thank you for using nginx.

task3_terraform_destroy.png



```
@KomalKashif ➜~/Lab12 $ terraform destroy
aws_subnet.myapp_subnet_1: Destroying... [id=subnet-019d88524243258c0]
aws_key_pair.ssh-key: Destroying... [id=serverkey]
aws_default_security_group.default_sg: Destroying... [id=sg-0d89ecd719a2188c4]
aws_default_security_group.default_sg: Destruction complete after 0s
aws_key_pair.ssh-key: Destruction complete after 1s
aws_subnet.myapp_subnet_1: Destruction complete after 1s
aws_vpc.myapp_vpc: Destroying... [id=vpc-0c052aa8e451842ee]
aws_vpc.myapp_vpc: Destruction complete after 1s

Destroy complete! Resources: 7 destroyed.
```

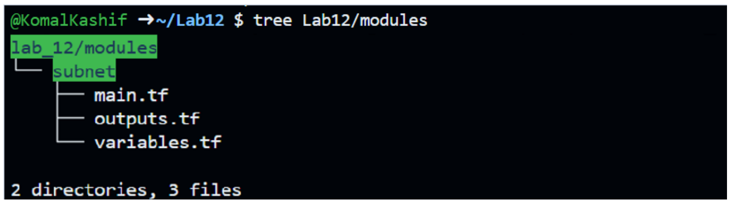
task3_main_tf_restored.png



```
main.tf
resource "aws_instance" "myapp-server" {
  }
user_data = file("./entry-script.sh")
  tags = {
     Name = "${var.env_prefix}-ec2-instance"
```

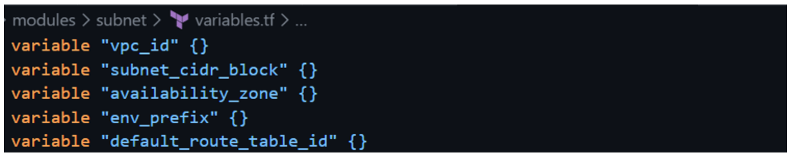
Task 4 — Create Terraform modules

task4_module_structure.png



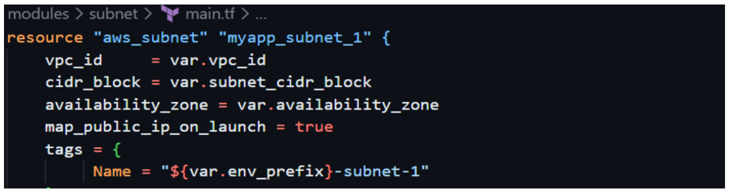
```
@KomalKashif ➜~/Lab12 $ tree Lab12/modules
lab_12/modules
└── subnet
    ├── main.tf
    ├── outputs.tf
    └── variables.tf

2 directories, 3 files
```

task4_subnet_variables.png



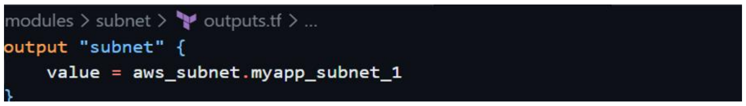
```
modules > subnet > variables.tf > ...
variable "vpc_id" {}
variable "subnet_cidr_block" {}
variable "availability_zone" {}
variable "env_prefix" {}
variable "default_route_table_id" {}
```

task4_subnet_main.png



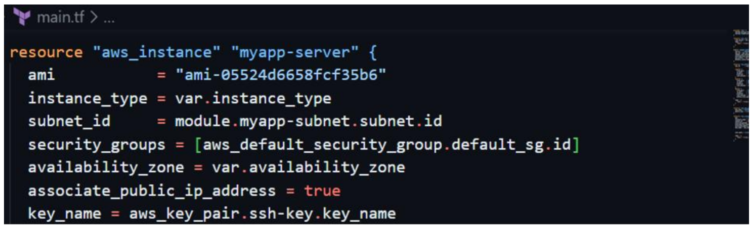
```
modules > subnet > main.tf > ...
resource "aws_subnet" "myapp_subnet_1" {
    vpc_id     = var.vpc_id
    cidr_block = var.subnet_cidr_block
    availability_zone = var.availability_zone
    map_public_ip_on_launch = true
    tags = {
        Name = "${var.env_prefix}-subnet-1"
```

task4_subnet_outputs.png



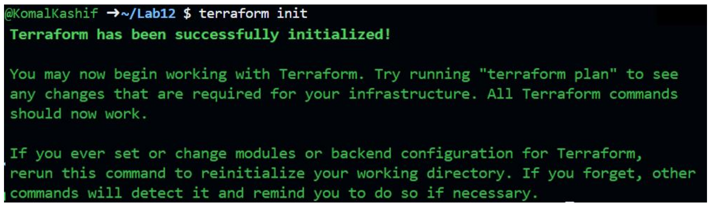
```
modules > subnet > outputs.tf > ...
output "subnet" {
    value = aws_subnet.myapp_subnet_1
}
```

task4_main_tf_with_module.png



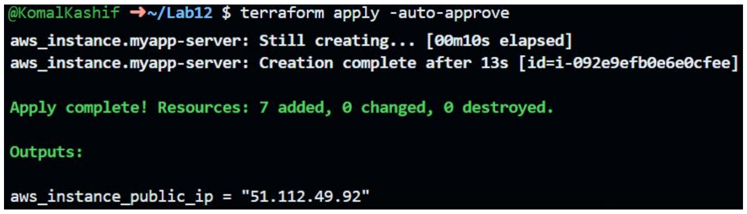
```
main.tf > ...

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
```

task4_terraform_init.png



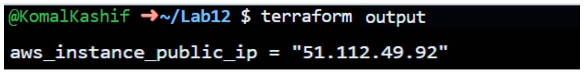
```
@KomalKashif ➜~/Lab12 $ terraform init
Terraform has been successfully initialized!

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.
```

task4_terraform_apply.png



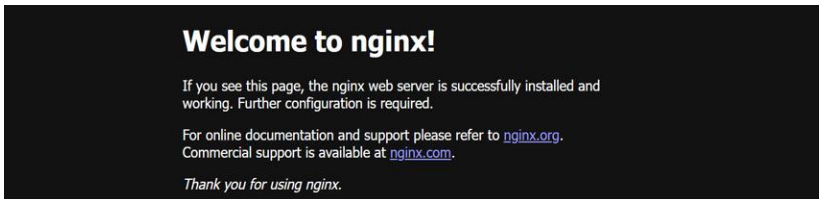
```
@KomalKashif ➜~/Lab12 $ terraform apply -auto-approve
aws_instance.myapp-server: Still creating... [00m10s elapsed]
aws_instance.myapp-server: Creation complete after 13s [id=i-092e9efb0e6e0cfee]

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

Outputs:

aws_instance_public_ip = "51.112.49.92"
```

task4_terraform_output.png



```
@KomalKashif ➜~/Lab12 $ terraform output
aws_instance_public_ip = "51.112.49.92"
```

task4_nginx_browser.png



Welcome to nginx!

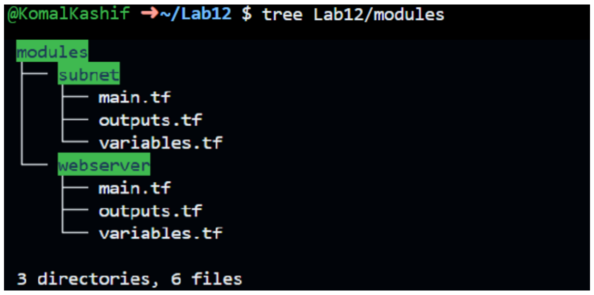
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

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

Thank you for using nginx.

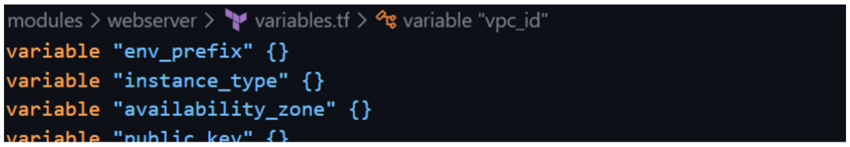
Task 5 — Create webserver module

task5_webserver_module_structure.png



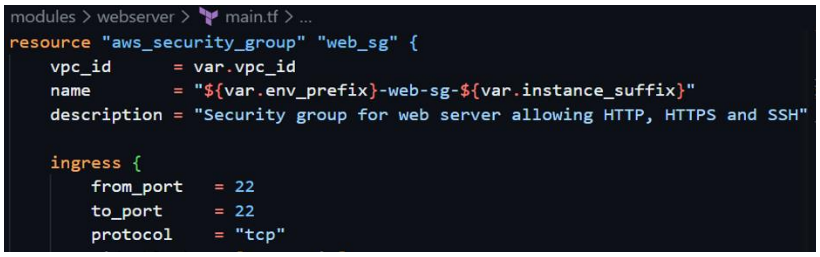
```
@KomalKashif ➜~/Lab12 $ tree Lab12/modules
modules
├── subnet
│   ├── main.tf
│   ├── outputs.tf
│   └── variables.tf
└── webserver
    ├── main.tf
    ├── outputs.tf
    └── variables.tf

3 directories, 6 files
```

task5_webserver_variables.png



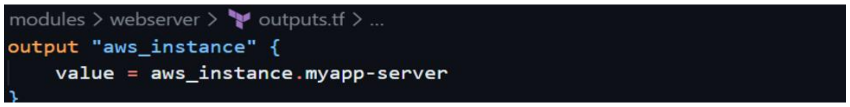
```
modules > webserver > variables.tf > variable "vpc_id"
variable "env_prefix" {}
variable "instance_type" {}
variable "availability_zone" {}
variable "public_key" {}
```

task5_webserver_main.png



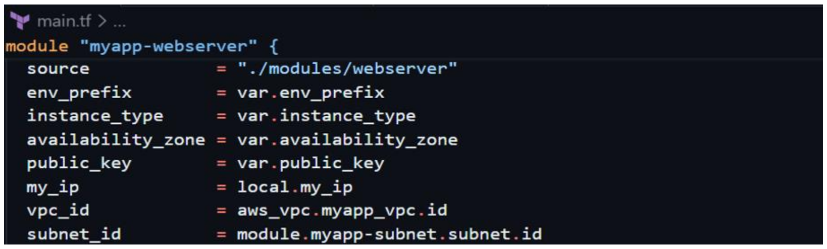
```
modules > webserver > main.tf > ...
resource "aws_security_group" "web_sg" {
    vpc_id      = var.vpc_id
    name        = "${var.env_prefix}-web-sg-${var.instance_suffix}"
    description = "Security group for web server allowing HTTP, HTTPS and SSH"

    ingress {
        from_port    = 22
        to_port      = 22
        protocol     = "tcp"
```

task5_webserver_outputs.png



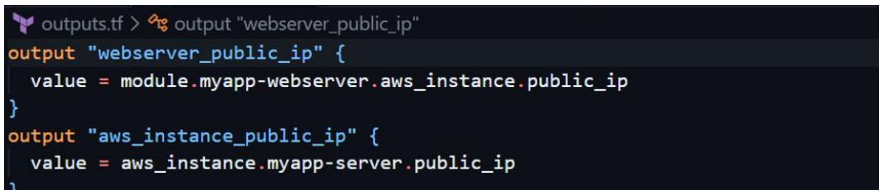
```
modules > webserver > outputs.tf > ...
output "aws_instance" {
    value = aws_instance.myapp-server
}
```

task5_main_tf_webserver_module.png



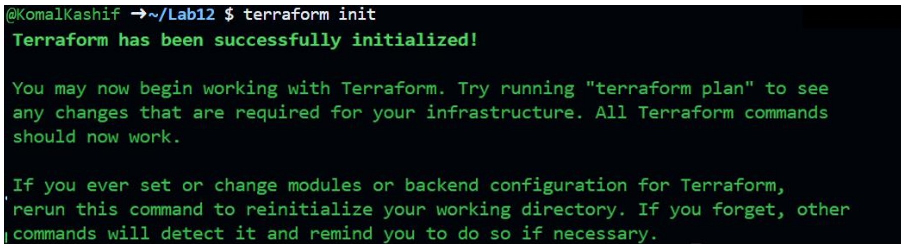
```
main.tf > ...
module "myapp-webserver" {
  source            = "./modules/webserver"
  env_prefix        = var.env_prefix
  instance_type     = var.instance_type
  availability_zone = var.availability_zone
  public_key        = var.public_key
  my_ip             = local.my_ip
  vpc_id            = aws_vpc.myapp_vpc.id
  subnet_id         = module.myapp-subnet.subnet.id
```

task5_outputs_updated.png



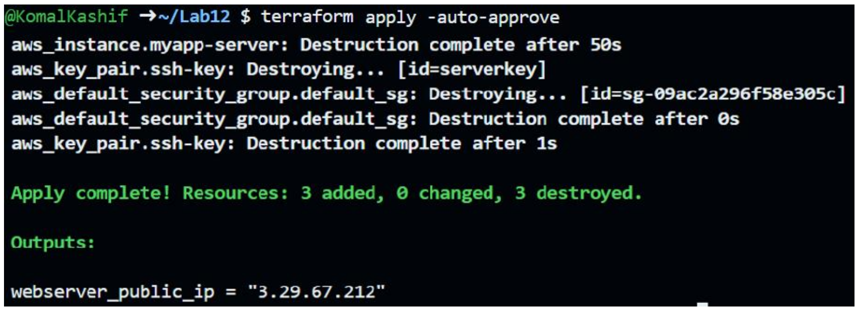
```
outputs.tf > output "webserver_public_ip"
output "webserver_public_ip" {
  value = module.myapp-webserver.aws_instance.public_ip
}
output "aws_instance_public_ip" {
  value = aws_instance.myapp-server.public_ip
}
```

task5_terraform_init.png



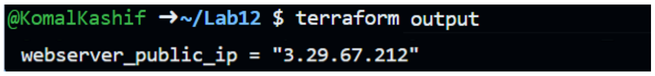
```
@KomalKashif ➜~/Lab12 $ terraform init
Terraform has been successfully initialized!

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.
```

task5_terraform_apply.png



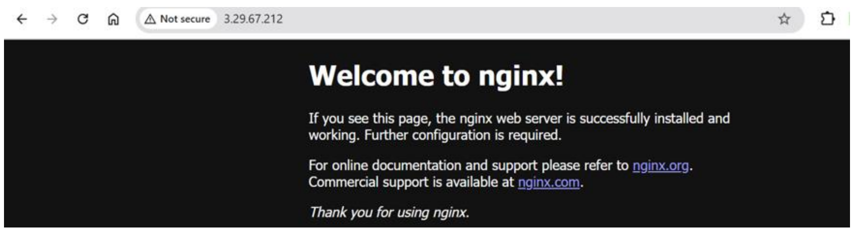
```
@KomalKashif ➜~/Lab12 $ terraform apply -auto-approve
aws_instance.myapp-server: Destruction complete after 50s
aws_key_pair.ssh-key: Destroying... [id=serverkey]
aws_default_security_group.default_sg: Destroying... [id=sg-09ac2a296f58e305c]
aws_default_security_group.default_sg: Destruction complete after 0s
aws_key_pair.ssh-key: Destruction complete after 1s

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

Outputs:

webserver_public_ip = "3.29.67.212"
```

task5_terraform_output.png



```
@KomalKashif ➜~/Lab12 $ terraform output
 webserver_public_ip = "3.29.67.212"
```

task5_nginx_browser.png



Not secure 3.29.67.212

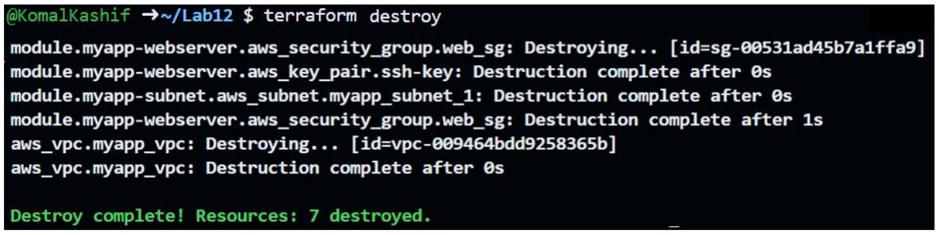
Welcome to nginx!

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.

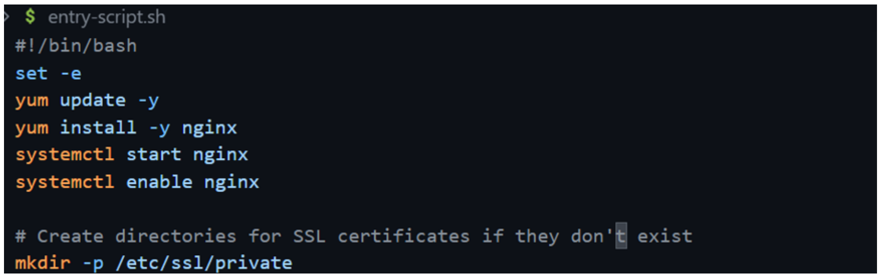
task5_terraform_destroy.png



```
@KomalKashif ➜~/Lab12 $ terraform destroy
module.myapp-webserver.aws_security_group.web_sg: Destroying... [id=sg-00531ad45b7a1ffa9]
module.myapp-webserver.aws_key_pair.ssh-key: Destruction complete after 0s
module.myapp-subnet.aws_subnet.myapp_subnet_1: Destruction complete after 0s
module.myapp-webserver.aws_security_group.web_sg: Destruction complete after 1s
aws_vpc.myapp_vpc: Destroying... [id=vpc-009464bdd9258365b]
aws_vpc.myapp_vpc: Destruction complete after 0s

Destroy complete! Resources: 7 destroyed.
```

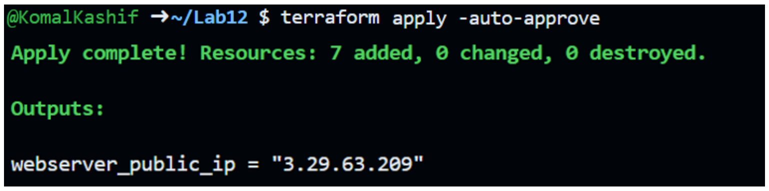
Task 6 — Configure HTTPS with self-signed certificates

task6_entry_script_https.png



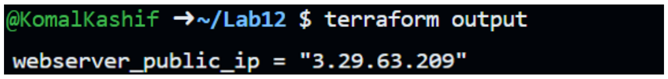
```
$ entry-script.sh
#!/bin/bash
set -e
yum update -y
yum install -y nginx
systemctl start nginx
systemctl enable nginx

# Create directories for SSL certificates if they don't exist
mkdir -p /etc/ssl/private
```

task6_terraform_apply.png



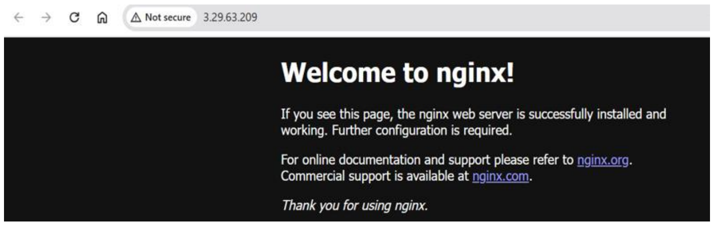
```
@KomalKashif ➜~/Lab12 $ terraform apply -auto-approve
Apply complete! Resources: 7 added, 0 changed, 0 destroyed.

Outputs:

webserver_public_ip = "3.29.63.209"
```

task6_terraform_output.png



```
@KomalKashif ➜~/Lab12 $ terraform output
webserver_public_ip = "3.29.63.209"
```

task6_http_redirect.png



Welcome to nginx!

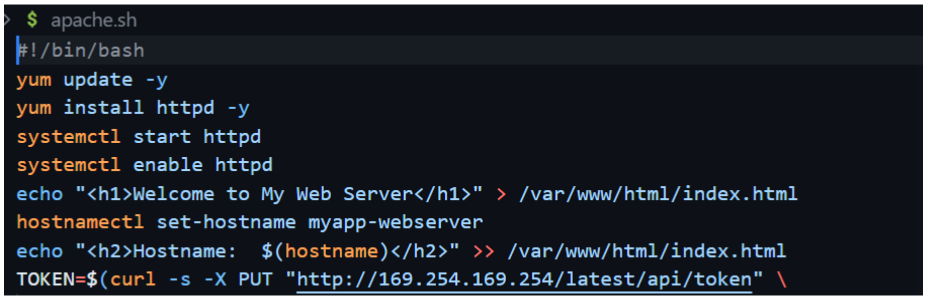
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

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

Thank you for using nginx.

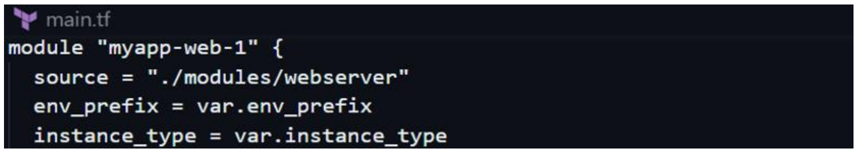
Task 7 — Configure Nginx as reverse proxy

task7_apache_script.png



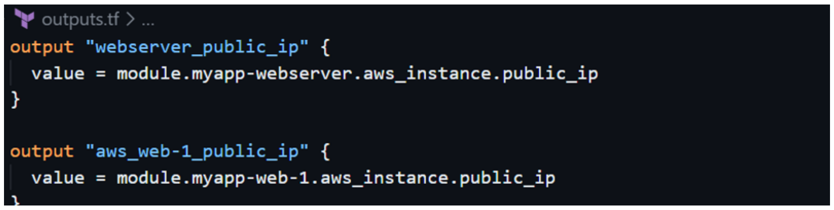
```
$ apache.sh
#!/bin/bash
yum update -y
yum install httpd -y
systemctl start httpd
systemctl enable httpd
echo "<h1>Welcome to My Web Server</h1>" > /var/www/html/index.html
hostnamectl set-hostname myapp-webserver
echo "<h2>Hostname:  $(hostname)</h2>" >> /var/www/html/index.html
TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/token" \
```

task7_main_tf_web1.png



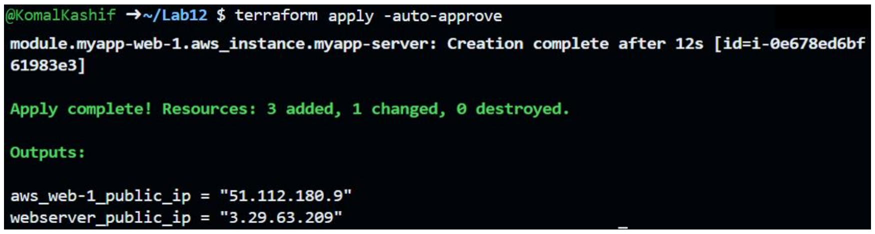
```
main.tf
module "myapp-web-1" {
  source = "./modules/webserver"
  env_prefix = var.env_prefix
  instance_type = var.instance_type
```

task7_outputs_web1.png



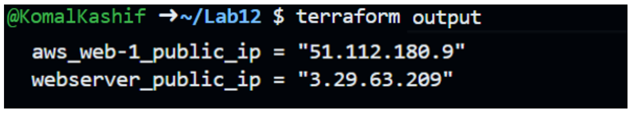
```
outputs.tf > ...
output "webserver_public_ip" {
  value = module.myapp-webserver.aws_instance.public_ip
}

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

task7_terraform_apply.png



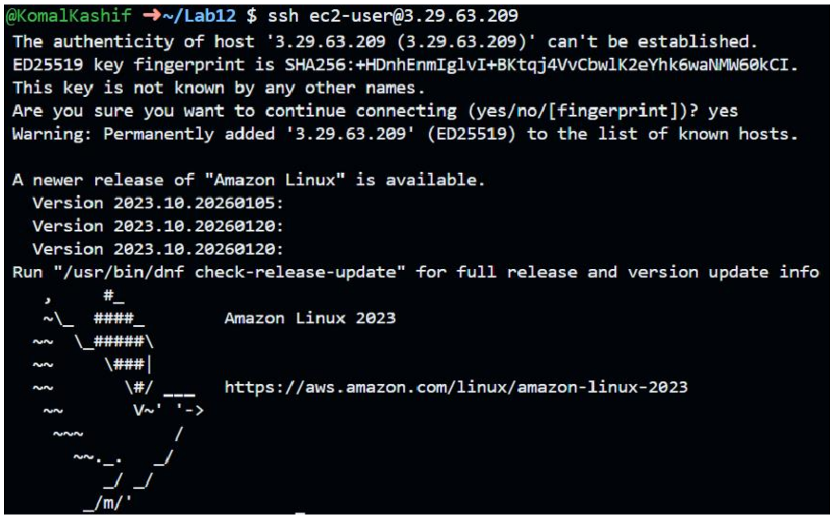
```
@KomalKashif ➜~/Lab12 $ terraform apply -auto-approve
module.myapp-web-1.aws_instance.myapp-server: Creation complete after 12s [id=i-0e678ed6bf
61983e3]

Apply complete! Resources: 3 added, 1 changed, 0 destroyed.

Outputs:

aws_web-1_public_ip = "51.112.180.9"
webserver_public_ip = "3.29.63.209"
```

task7_terraform_output.png



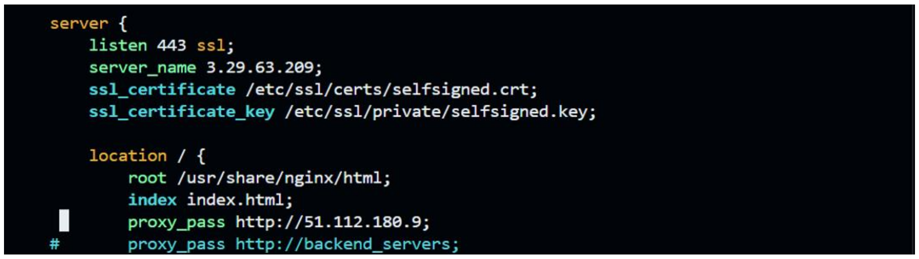
```
@KomalKashif ➜~/Lab12 $ terraform output
  aws_web-1_public_ip = "51.112.180.9"
  webserver_public_ip = "3.29.63.209"
```

task7_ssh_webserver.png



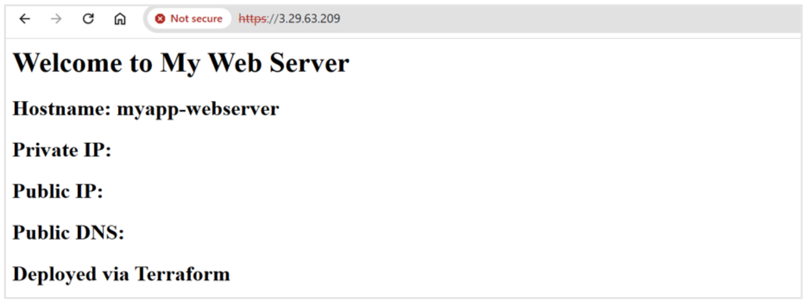
```
@KomalKashif ➜~/Lab12 $ ssh ec2-user@3.29.63.209
The authenticity of host '3.29.63.209 (3.29.63.209)' can't be established.
ED25519 key fingerprint is SHA256:+HDnhEnmIglvI+BKtqj4VvCbwlK2eYhk6waNMW60kCI.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.29.63.209' (ED25519) to the list of known hosts.

A newer release of "Amazon Linux" is available.
  Version 2023.10.20260105:
  Version 2023.10.20260120:
  Version 2023.10.20260120:
Run "/usr/bin/dnf check-release-update" for full release and version update info
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
```

task7_nginx_conf_reverse_proxy.png



```
server {
    listen 443 ssl;
    server_name 3.29.63.209;
    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://51.112.180.9;
#       proxy_pass http://backend_servers;
```

task7_reverse_proxy_browser.png



Welcome to My Web Server

Hostname: myapp-webserver

Private IP:

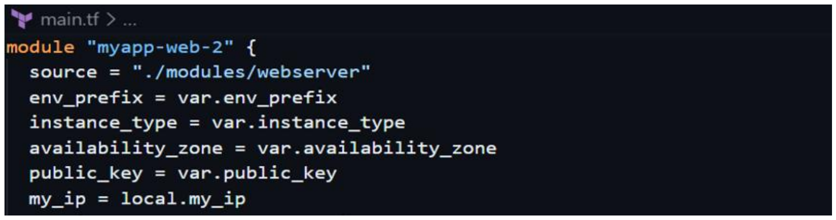
Public IP:

Public DNS:

Deployed via Terraform

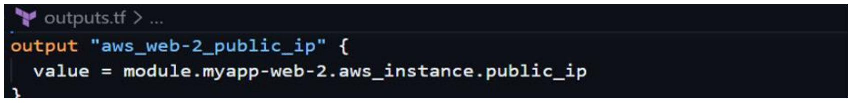
Task 8 — Configure Nginx as load balancer

task8_main_tf_web2.png



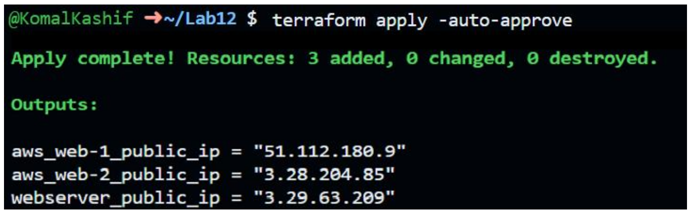
```
main.tf > ...
module "myapp-web-2" {
  source = "./modules/webserver"
  env_prefix = var.env_prefix
  instance_type = var.instance_type
  availability_zone = var.availability_zone
  public_key = var.public_key
  my_ip = local.my_ip
```

task8_outputs_web2.png



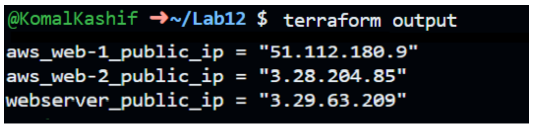
```
outputs.tf > ...
output "aws_web-2_public_ip" {
  value = module.myapp-web-2.aws_instance.public_ip
}
```

task8_terraform_apply.png



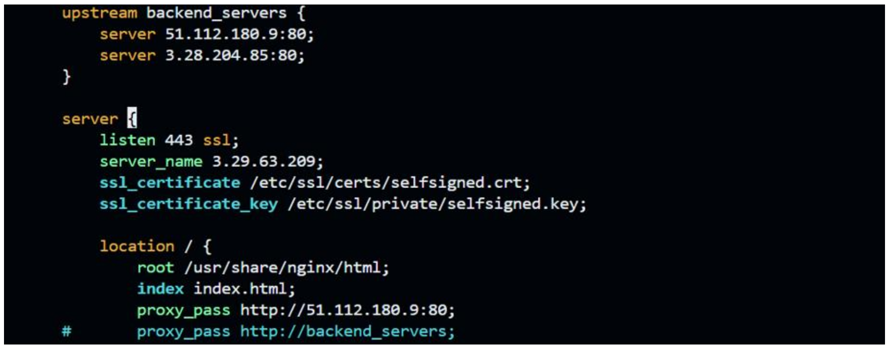
```
@KomalKashif ➜~/Lab12 $ terraform apply -auto-approve

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

Outputs:

aws_web-1_public_ip = "51.112.180.9"
aws_web-2_public_ip = "3.28.204.85"
webserver_public_ip = "3.29.63.209"
```

task8_terraform_output.png



```
@KomalKashif ➜~/Lab12 $ terraform output
aws_web-1_public_ip = "51.112.180.9"
aws_web-2_public_ip = "3.28.204.85"
webserver_public_ip = "3.29.63.209"
```

task8_nginx_conf_load_balancer.png



```
upstream backend_servers {
    server 51.112.180.9:80;
    server 3.28.204.85:80;
}

server {
    listen 443 ssl;
    server_name 3.29.63.209;
    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://51.112.180.9:80;
#       proxy_pass http://backend_servers;
```