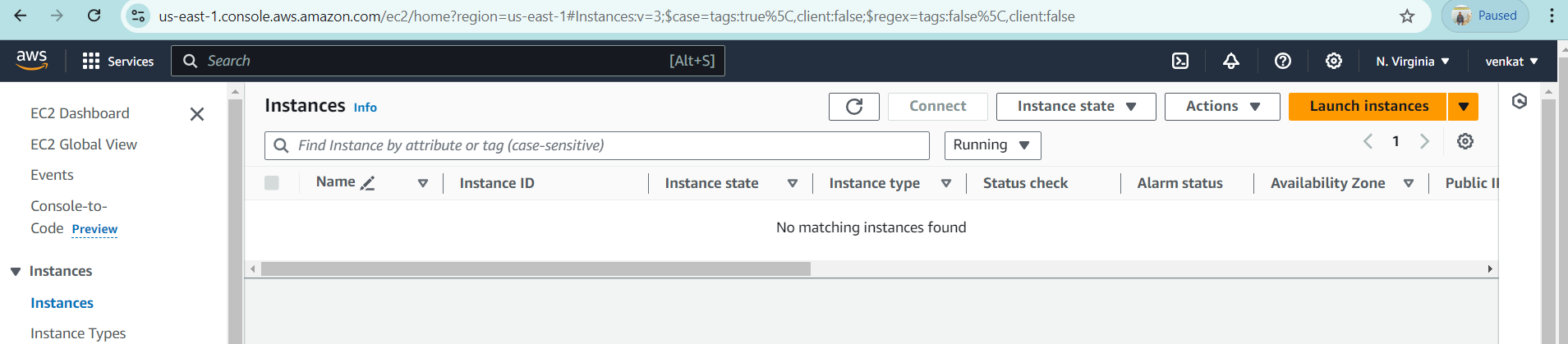
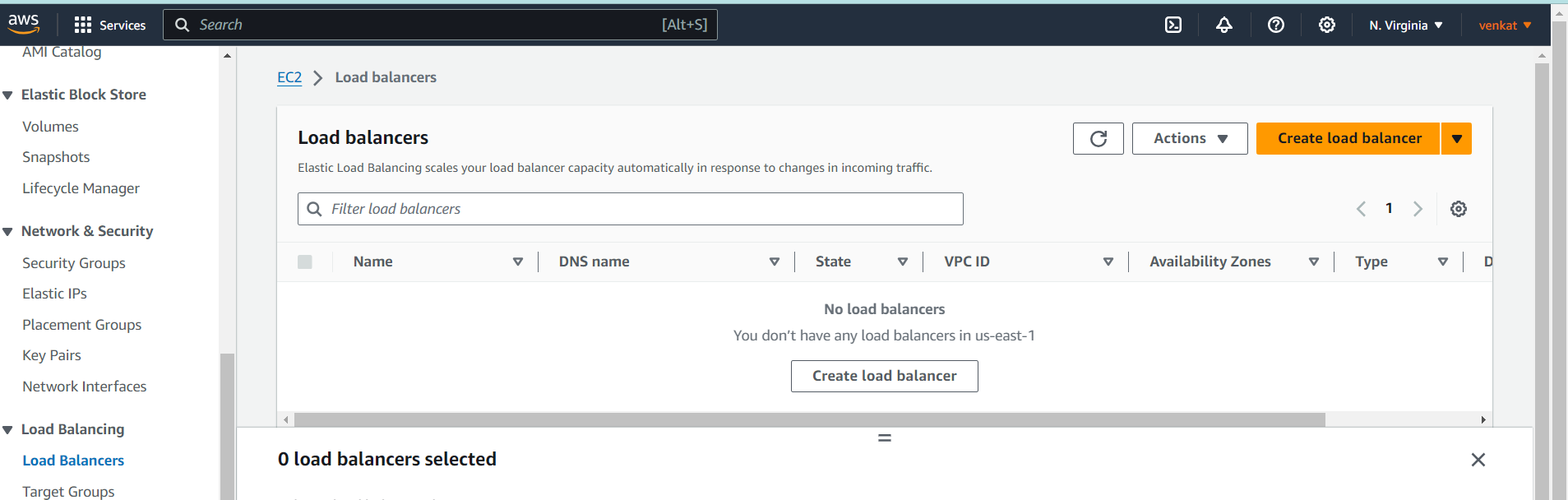
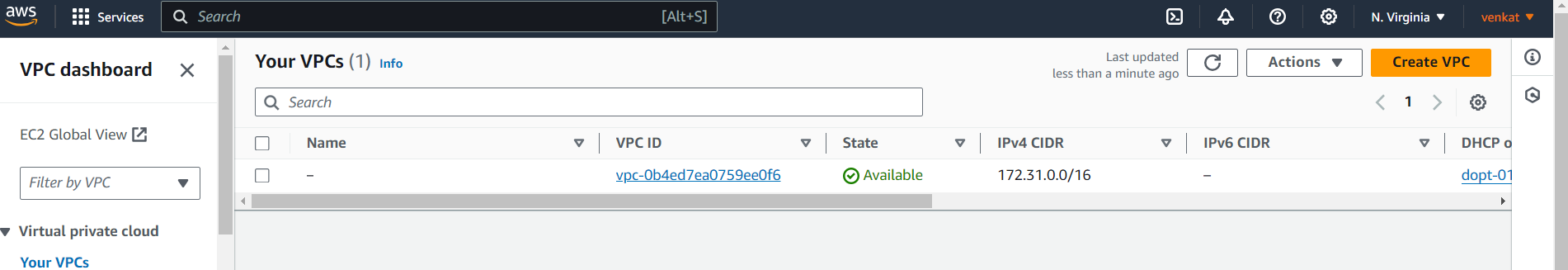
CREATE LOAD BALANCER USING TERRAFORM (us-east-1)

1. Login aws account





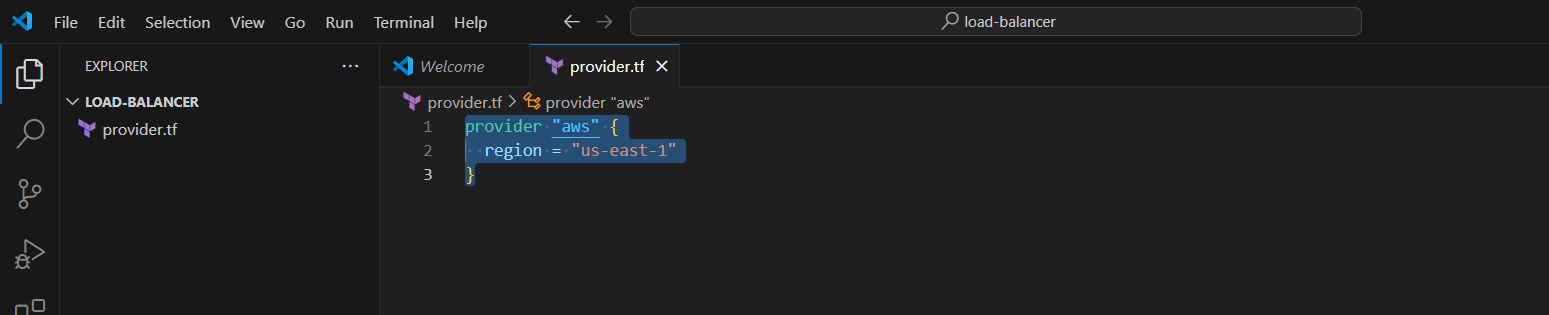


1. Now open visual studio code
2. Select folder
3. Crete file provider.tf

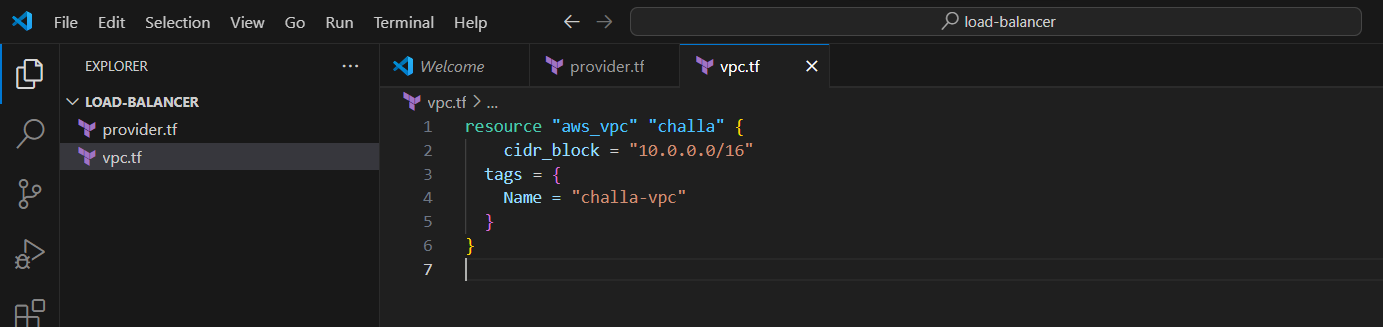
provider "aws" {

region = "us-east-1"

}



1. Now create vpc.tf



resource "aws\_vpc" "challa" {

cidr\_block = "10.0.0.0/16"

tags = {

Name = "challa-vpc"

}

}

1. Now create subnet.tf

resource "aws\_subnet" "challa-public-subnet1" {

vpc\_id = aws\_vpc.challa.id

cidr\_block = "10.0.0.0/24"

map\_public\_ip\_on\_launch = "true"

availability\_zone = "us-east-1a"

tags = {

Name = "challa-public-subnet1"

}

}

#public-subnet2 creation

resource "aws\_subnet" "challa-public-subnet2" {

vpc\_id = aws\_vpc.challa.id

cidr\_block = "10.0.1.0/24"

map\_public\_ip\_on\_launch = "true"

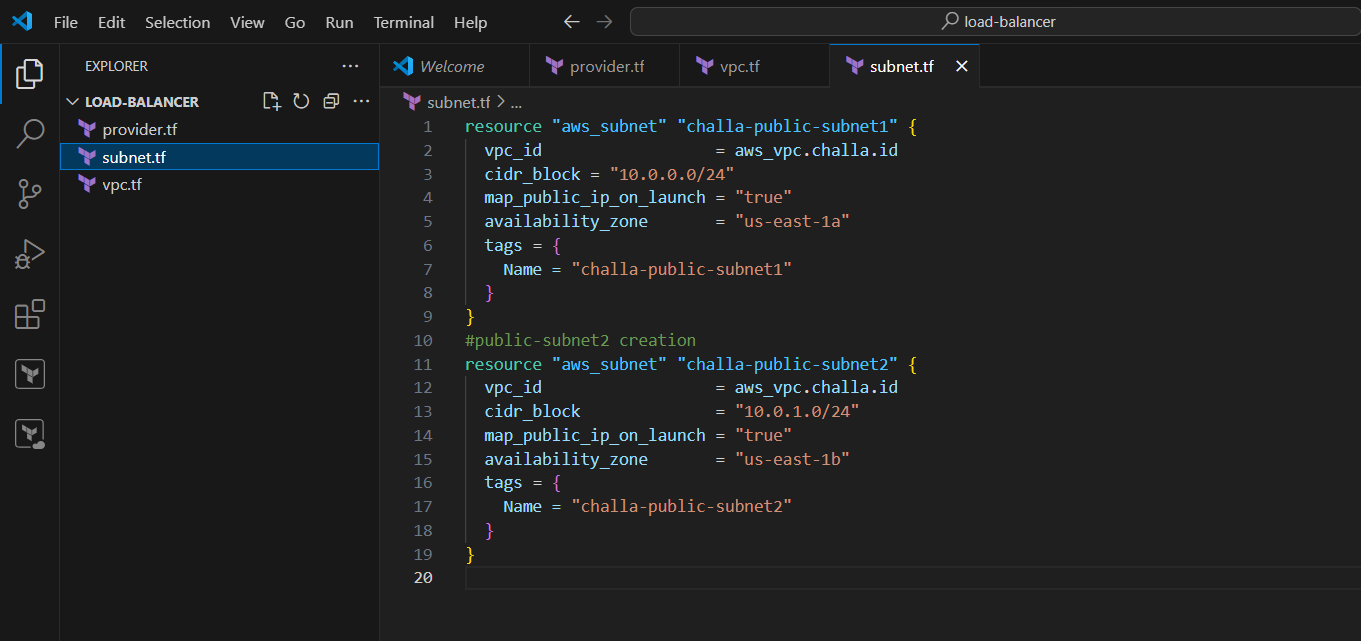
availability\_zone = "us-east-1b"

tags = {

Name = "challa-public-subnet2"

}

}

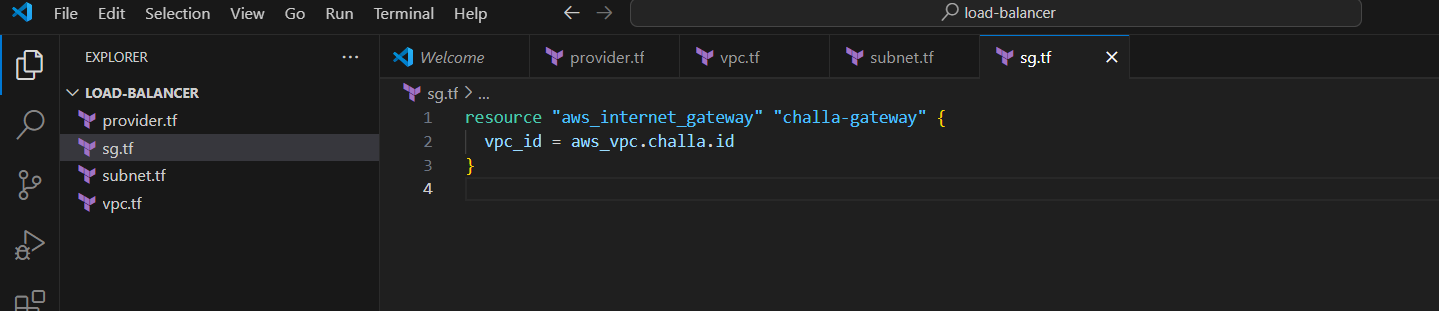


1. Now create igw.tf

resource "aws\_internet\_gateway" "challa-gateway" {

vpc\_id = aws\_vpc.challa.id

}



1. Now create route table add route and subnet association
2. route.tf

resource "aws\_route\_table" "challa-route" {

vpc\_id = aws\_vpc.challa.id

route {

cidr\_block = "0.0.0.0/0"

gateway\_id = aws\_internet\_gateway.challa-gateway.id

}

tags = {

Name = "route to internet"

}

}

#route 1

resource "aws\_route\_table\_association" "challa-route1" {

subnet\_id = aws\_subnet.challa-public-subnet1.id

route\_table\_id = aws\_route\_table.challa-route.id

}

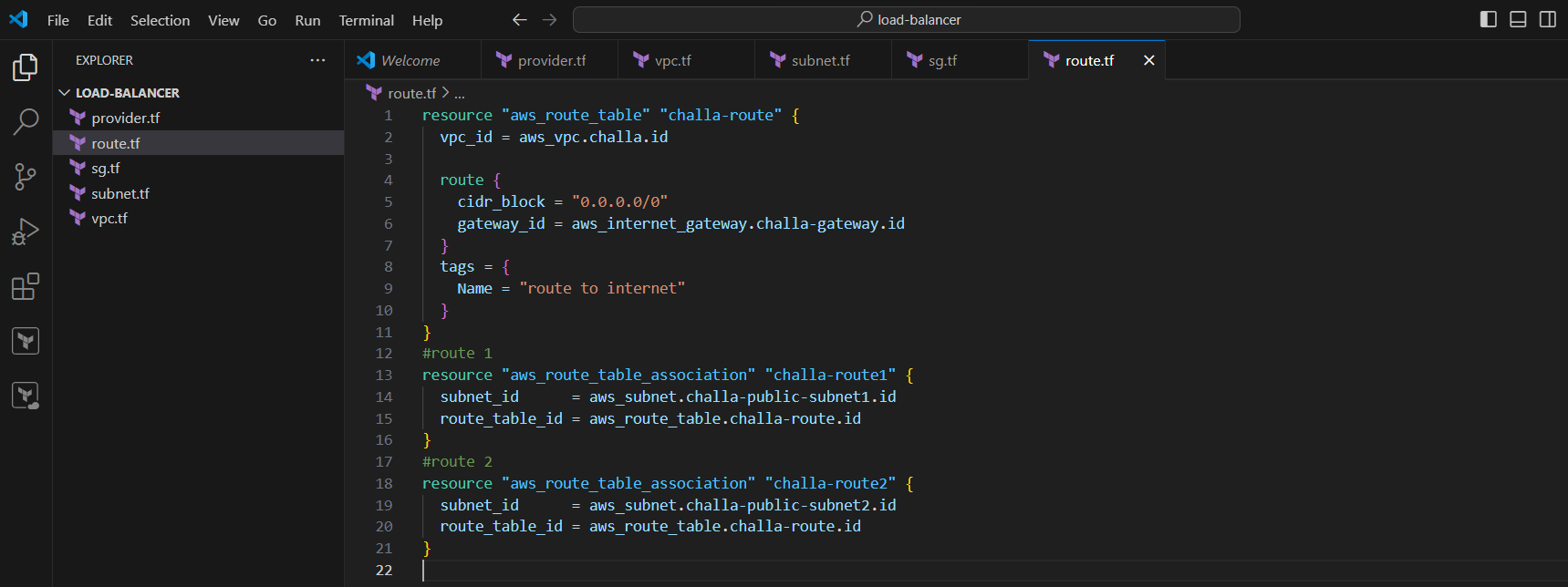
#route 2

resource "aws\_route\_table\_association" "challa-route2" {

subnet\_id = aws\_subnet.challa-public-subnet2.id

route\_table\_id = aws\_route\_table.challa-route.id

}



1. Now create security group sg.tf

resource "aws\_security\_group" "challa-sg" {

vpc\_id = aws\_vpc.challa.id

ingress {

from\_port = 80

to\_port = 80

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

ingress {

from\_port = 443

to\_port = 443

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

ingress {

from\_port = 22

to\_port = 22

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"]

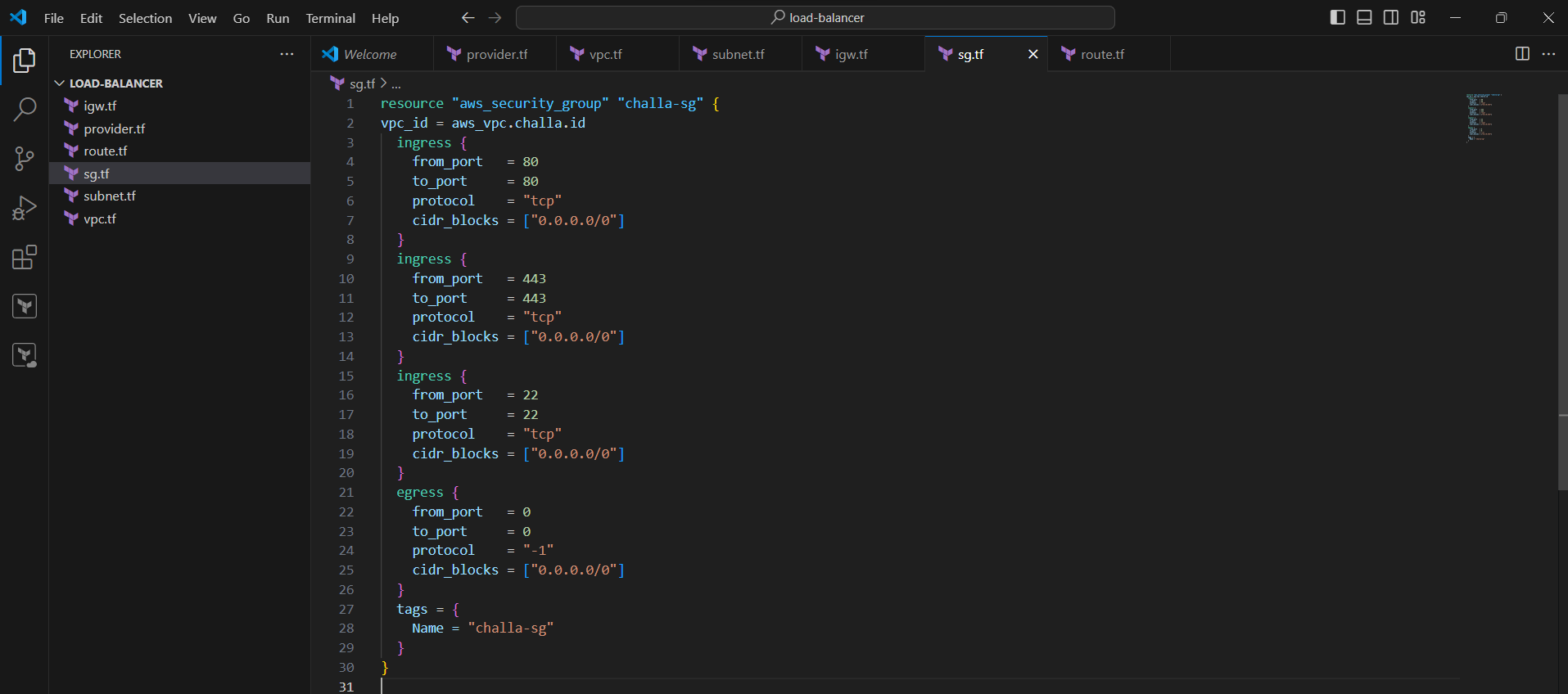
}

tags = {

Name = "challa-sg"

}

}



1. Now create 2 ec2 instance ec2.tf

resource "aws\_instance" "challa" {

ami = "ami-0e86e20dae9224db8" #use your ubuntu ami because user data write ubuntu script

instance\_type = "t2.micro"

key\_name = "challa"

vpc\_security\_group\_ids = [aws\_security\_group.challa-sg.id]

subnet\_id = aws\_subnet.challa-public-subnet1.id

associate\_public\_ip\_address = true

user\_data = file("${path.module}/appache1.sh")

tags = {

Name = "challa"

}

}

resource "aws\_instance" "rao" {

ami = "ami-0e86e20dae9224db8" #use your ubuntu ami because user data write ubuntu script

instance\_type = "t2.micro"

key\_name = "rao"

vpc\_security\_group\_ids = [aws\_security\_group.challa-sg.id]

subnet\_id = aws\_subnet.challa-public-subnet2.id

associate\_public\_ip\_address = true

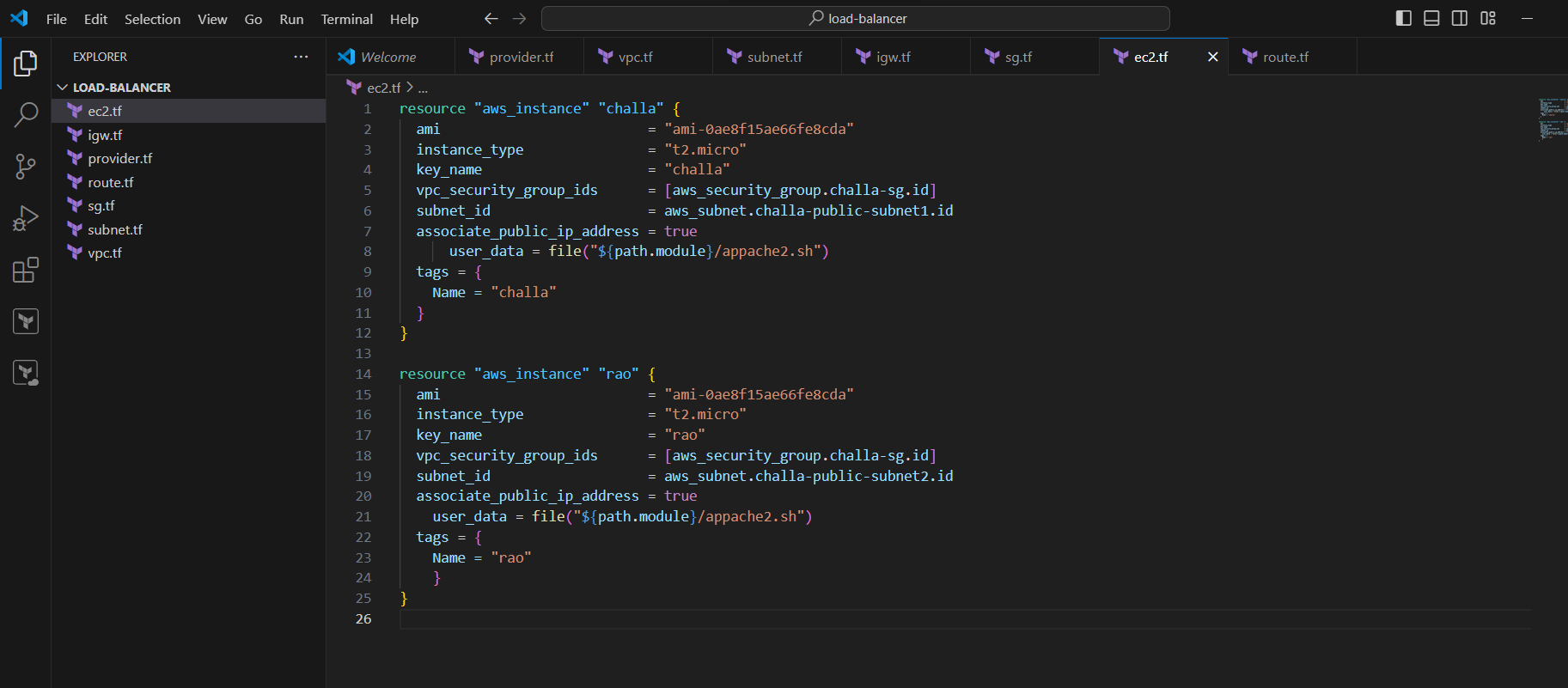
user\_data = file("${path.module}/appache2.sh")

tags = {

Name = "rao"

}

}



1. Now create appache1.sh to install nginx for challa instance

#appache1.sh

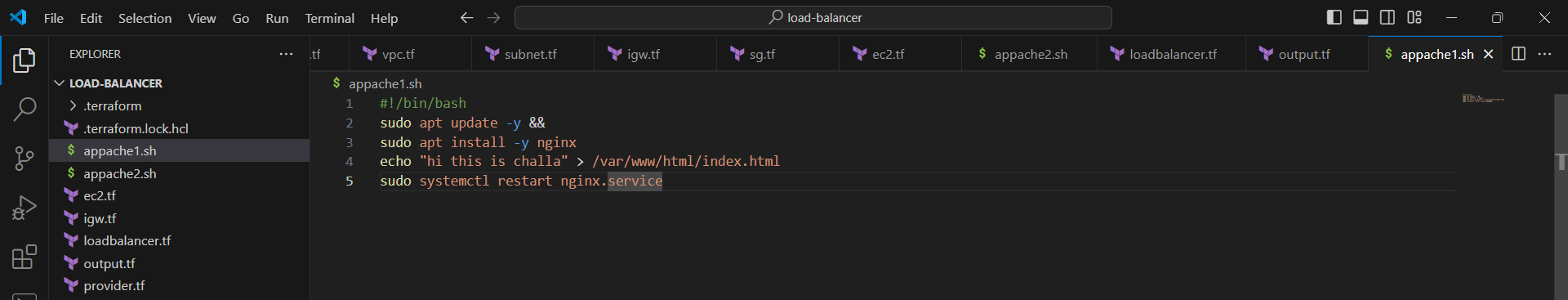
#!/bin/bash

sudo apt update -y &&

sudo apt install -y nginx

echo "hi this is challa" > /var/www/html/index.html

sudo systemctl restart nginx.service



1. Now create appache2.sh to install nginx for rao instance

#appache2.sh

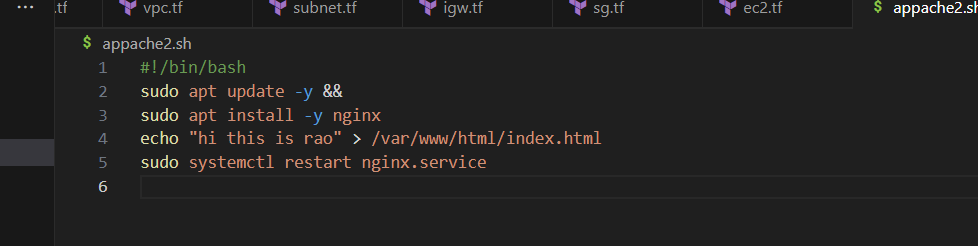
#!/bin/bash

sudo apt update -y &&

sudo apt install -y nginx

echo "hi this is rao" > /var/www/html/index.html

sudo systemctl restart nginx.service



1. Now create load balancer and target group

resource "aws\_lb" "challa-alb" {

name = "challa-LB"

internal = false

load\_balancer\_type = "application"

security\_groups = [aws\_security\_group.challa-sg.id]

subnets = [aws\_subnet.challa-public-subnet1.id, aws\_subnet.challa-public-subnet2.id]

}

resource "aws\_lb\_target\_group" "challa-tg" {

name = "challa-TG"

port = 80

protocol = "HTTP"

vpc\_id = aws\_vpc.challa.id

health\_check {

path = "/health"

port = 80

protocol = "HTTP"

}

}

resource "aws\_lb\_target\_group\_attachment" "challas" {

target\_group\_arn = aws\_lb\_target\_group.challa-tg.arn

target\_id = aws\_instance.challa.id

port = 80

depends\_on = [

aws\_lb\_target\_group.challa-tg,

aws\_instance.challa,

]

}

resource "aws\_lb\_target\_group\_attachment" "raos" {

target\_group\_arn = aws\_lb\_target\_group.challa-tg.arn

target\_id = aws\_instance.rao.id

port = 80

depends\_on = [

aws\_lb\_target\_group.challa-tg,

aws\_instance.rao,

]

}

resource "aws\_lb\_listener" "listener\_elb" {

load\_balancer\_arn = aws\_lb.challa-alb.arn

port = 80

protocol = "HTTP"

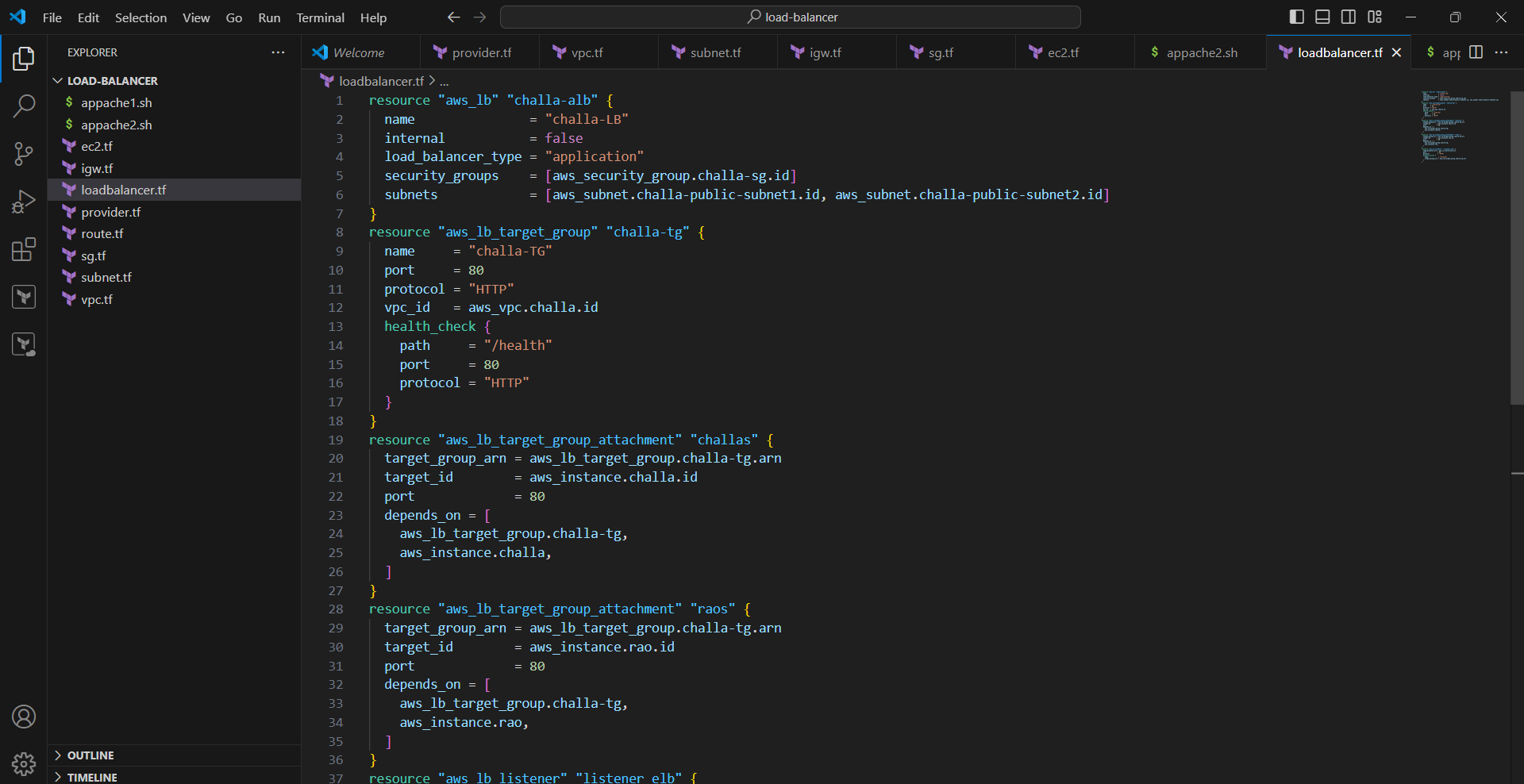
default\_action {

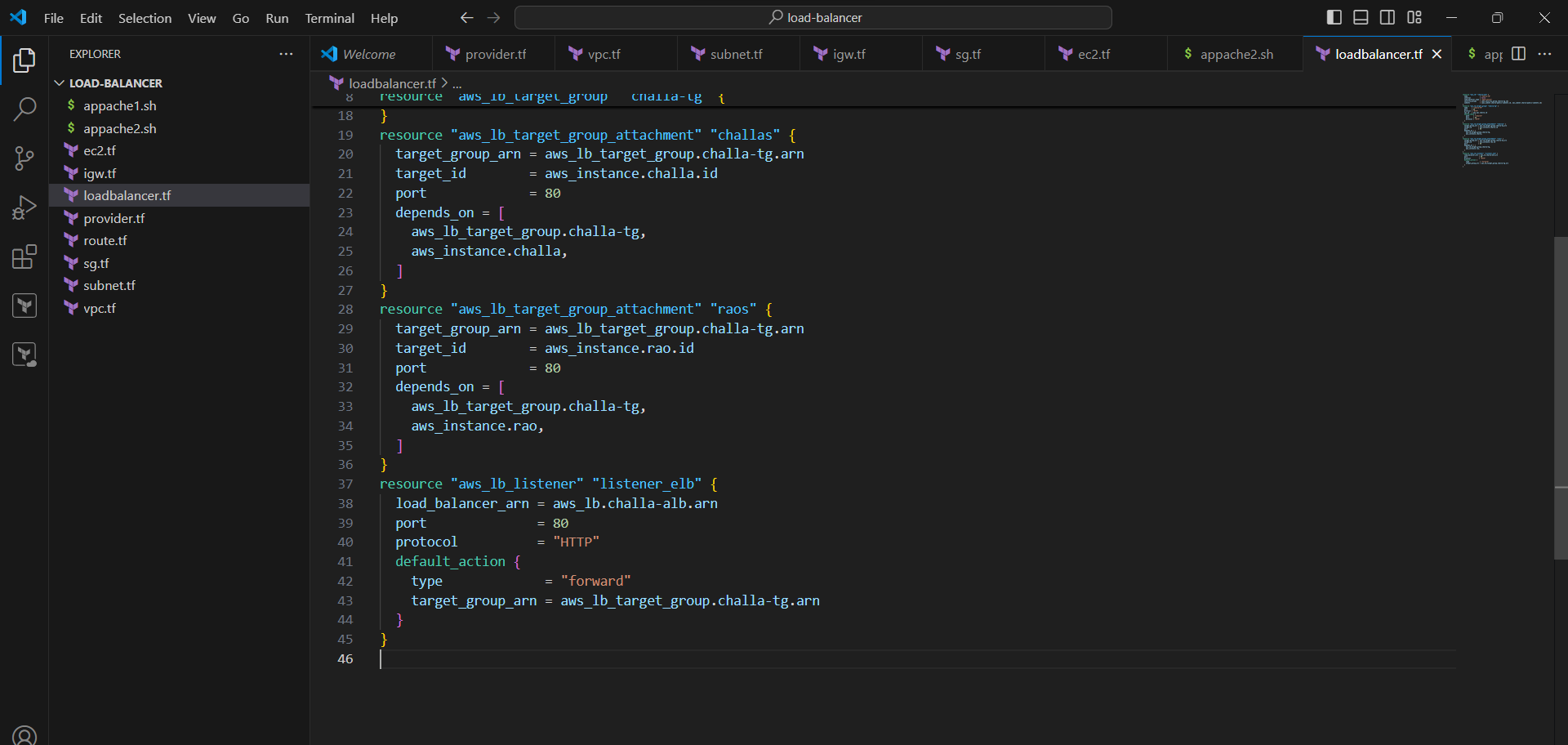
type = "forward"

target\_group\_arn = aws\_lb\_target\_group.challa-tg.arn

}

}





1. Now create output.tf

#output.tf

#DNS of LoadBalancer

output "lb\_dns\_name" {

description = "DNS of Load balancer"

value = aws\_lb.challa-alb.dns\_name

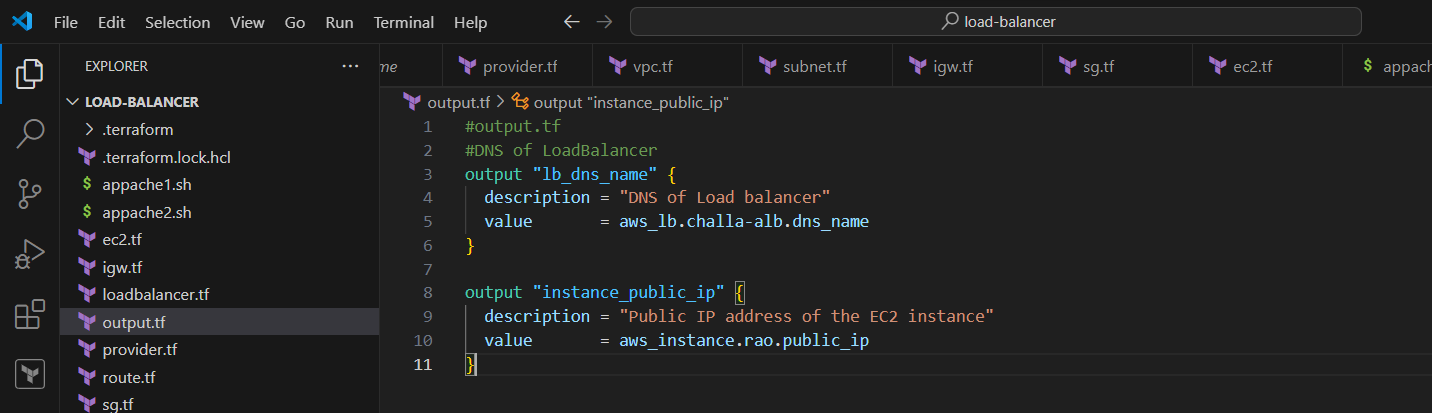
}

output "instance\_public\_ip" {

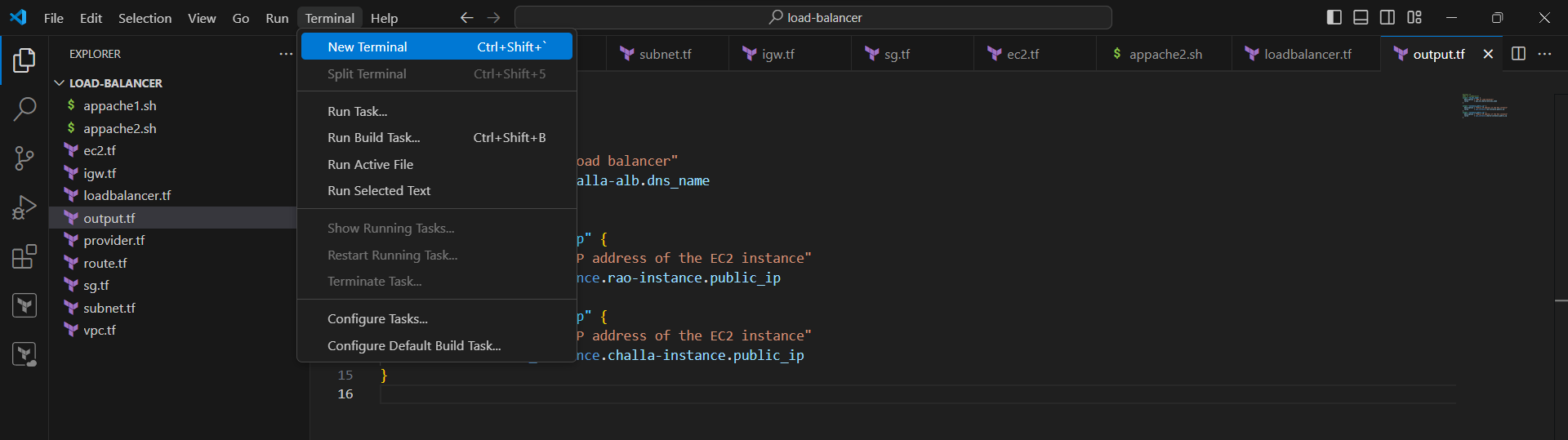
description = "Public IP address of the EC2 instance"

value = aws\_instance.rao.public\_ip

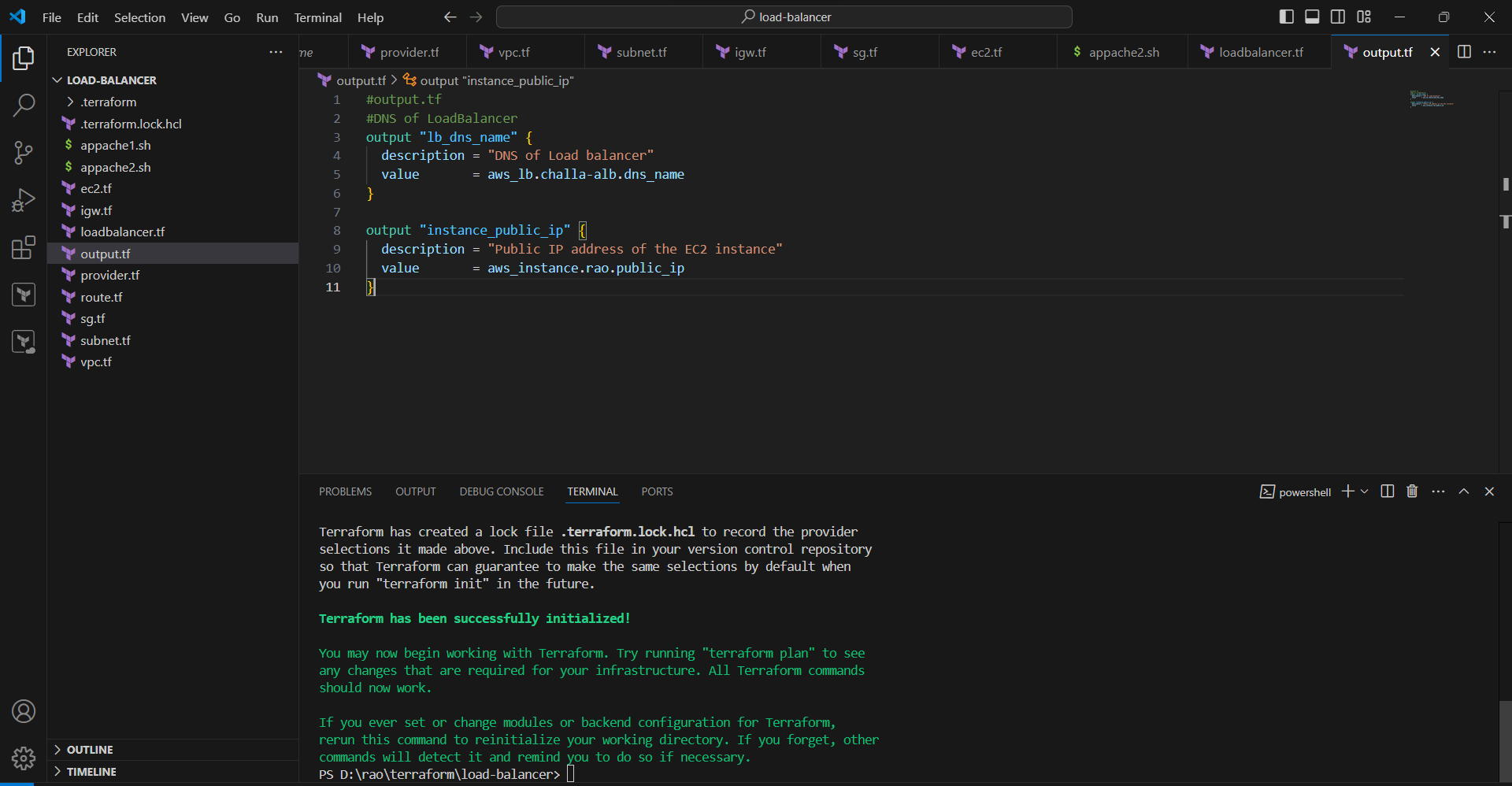
}



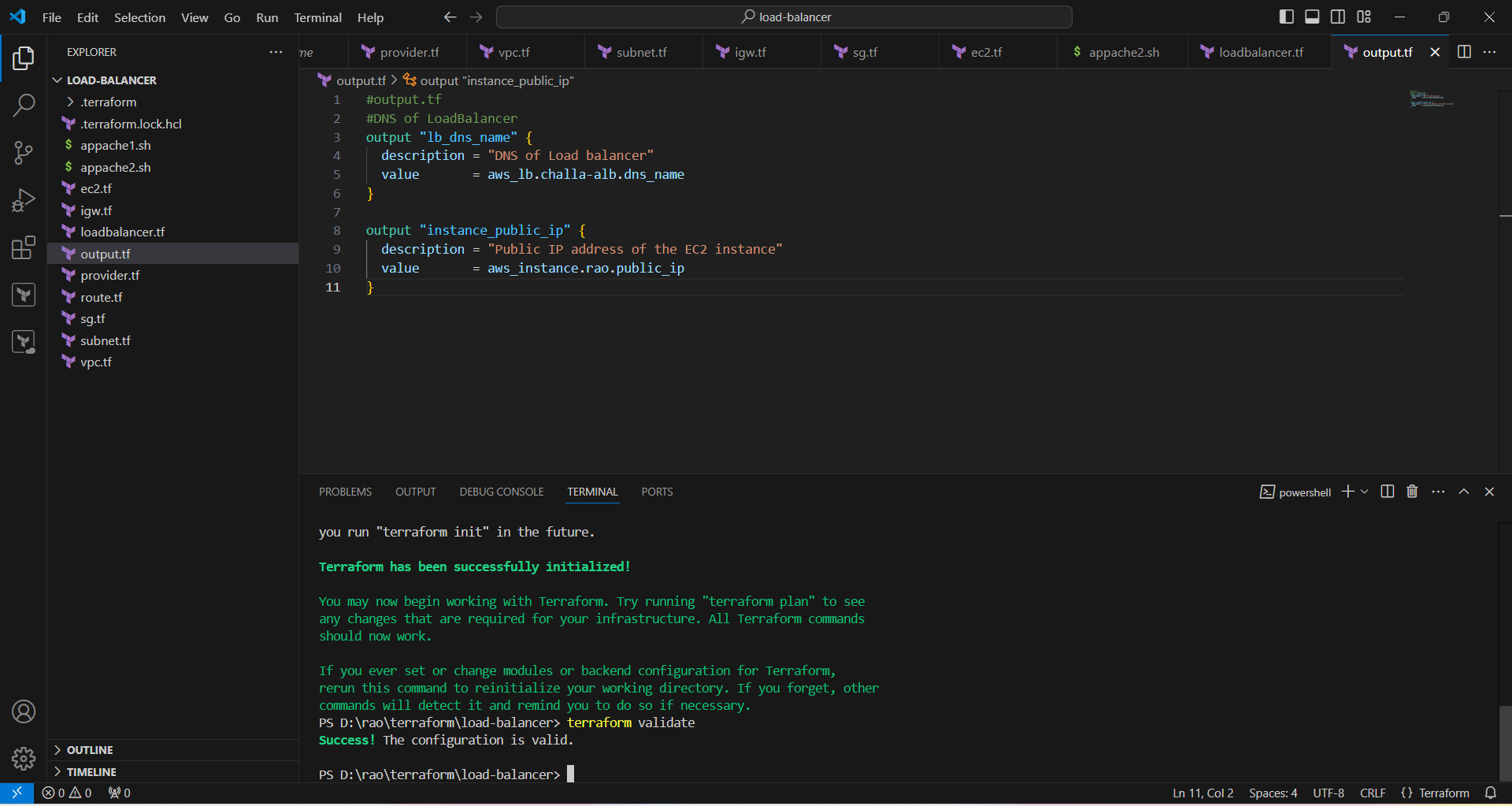
1. Now save all
2. Now click on terminal select new terminal



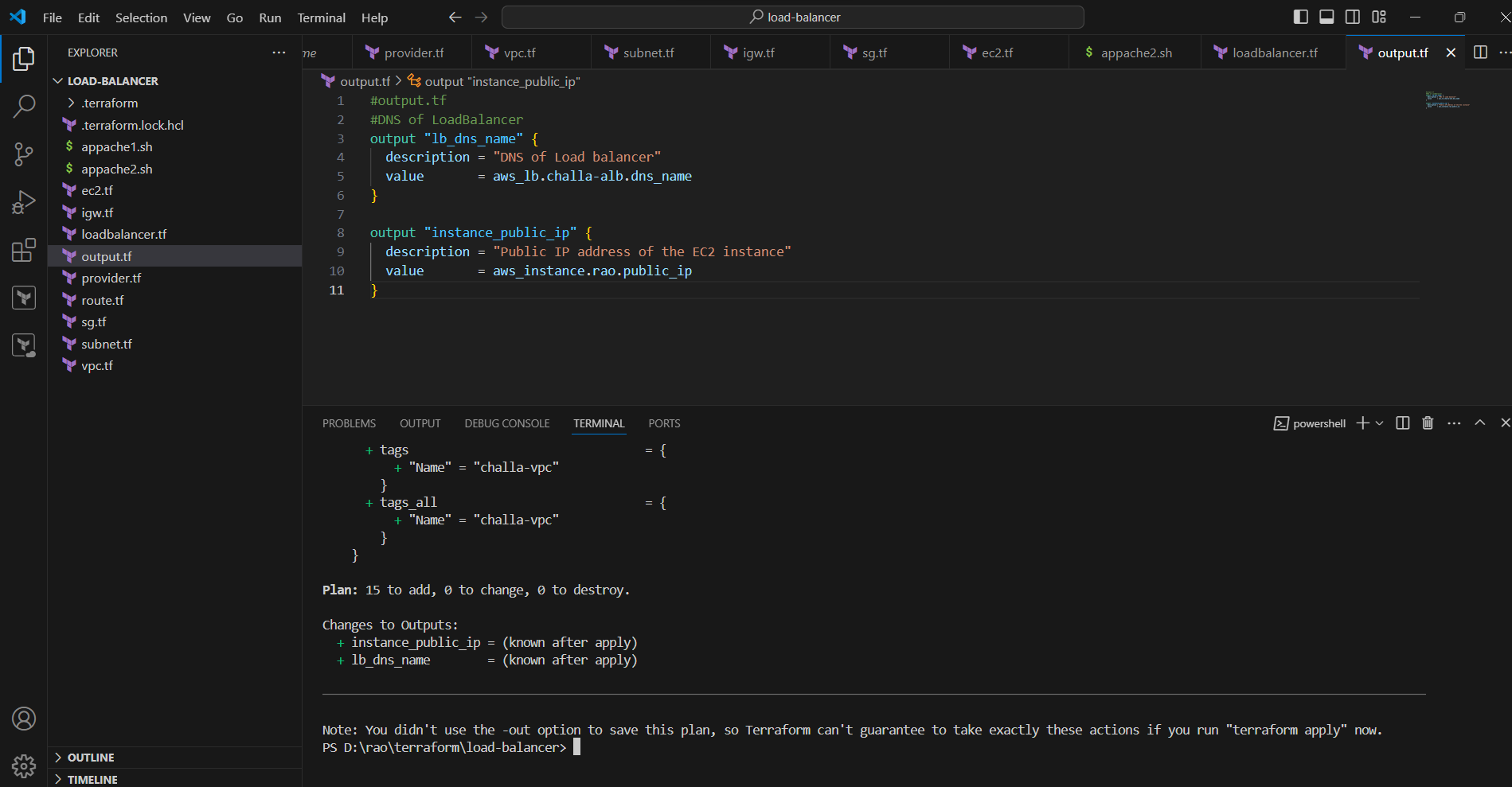
1. Now click on terminal and use #terraform init



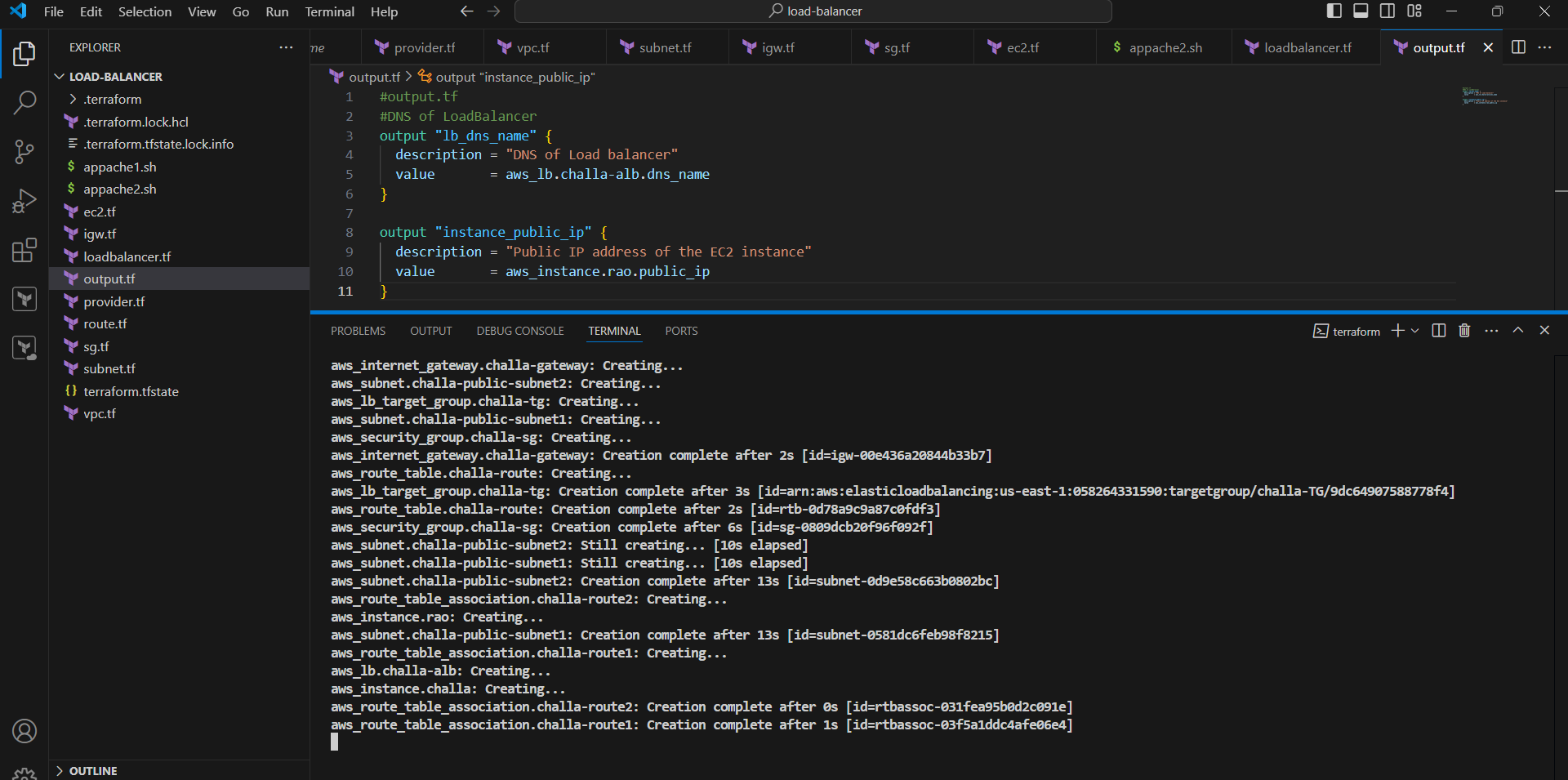
1. #terraform validate

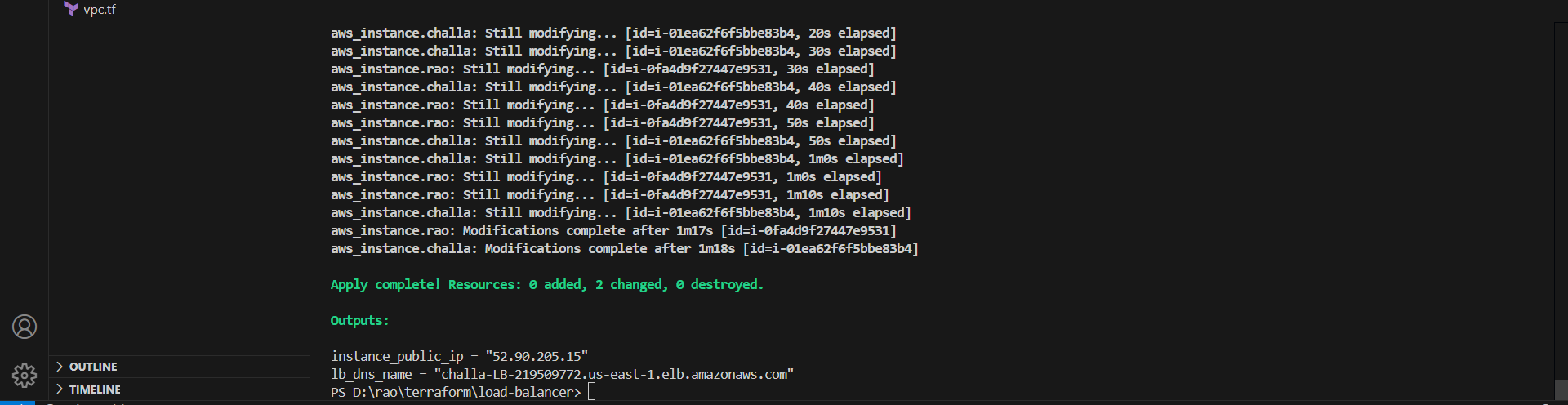


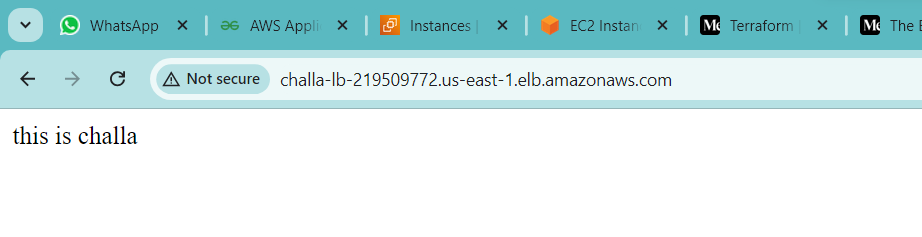
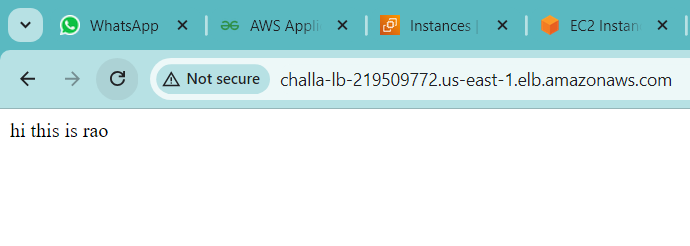
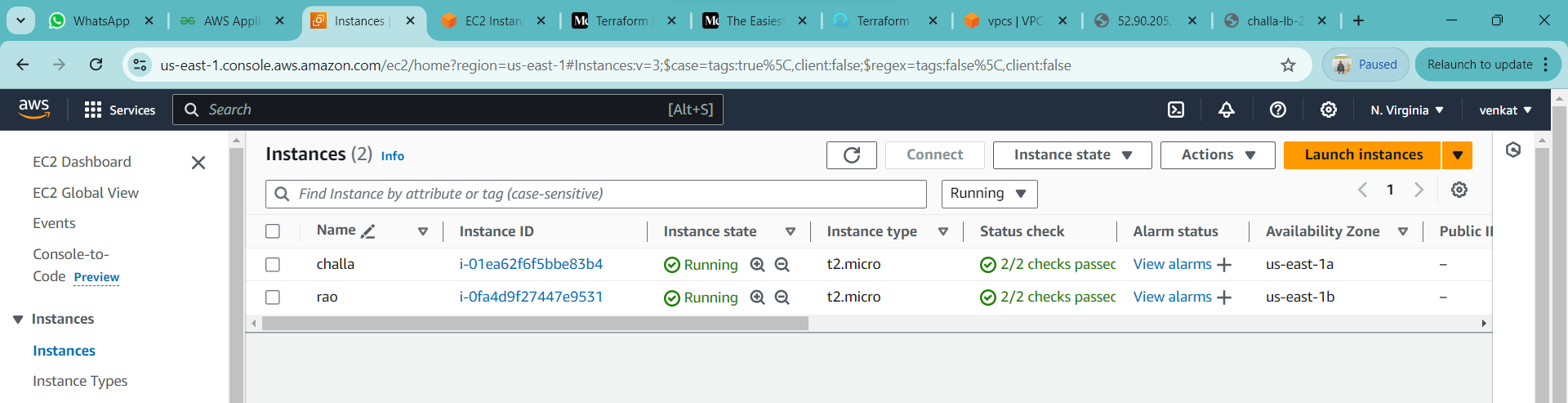
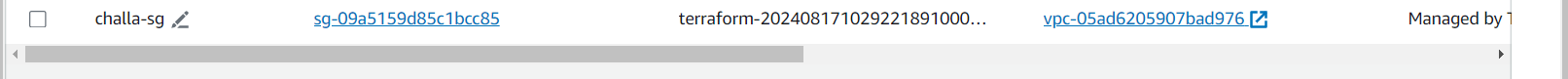
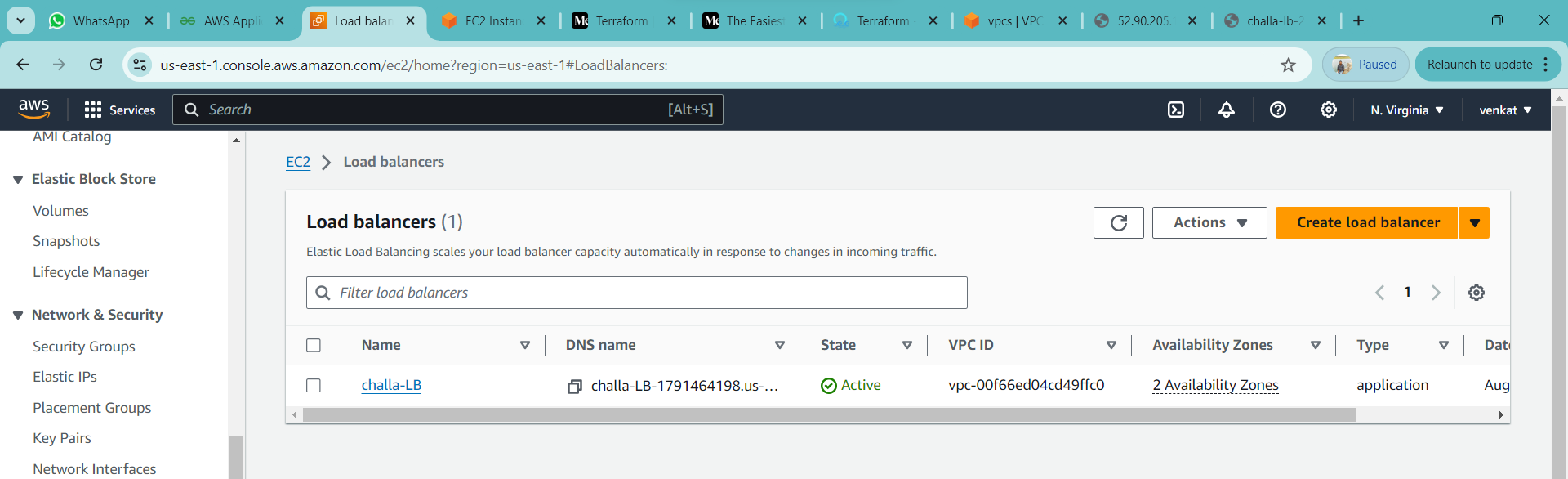
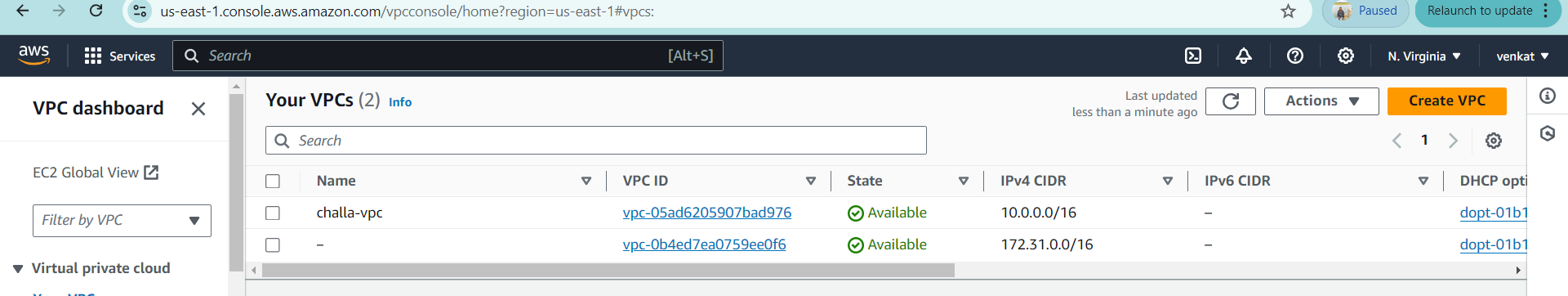
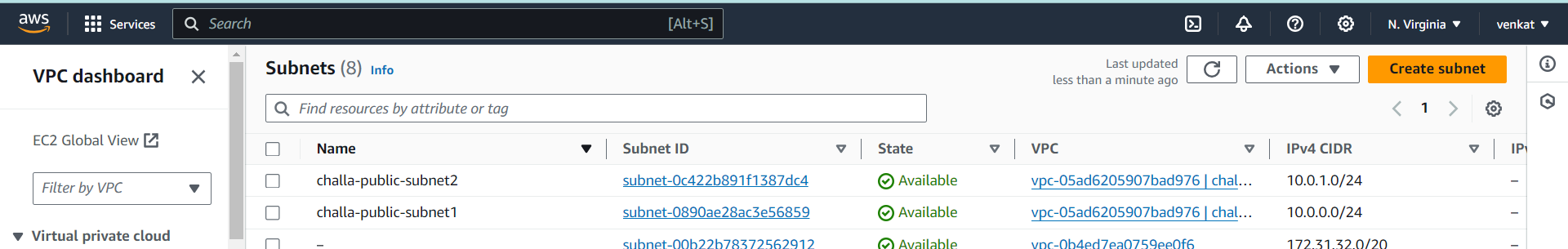
1. Now terraform plan #terraform plan



1. Now terraform apply #terraform apply --auto-aprove

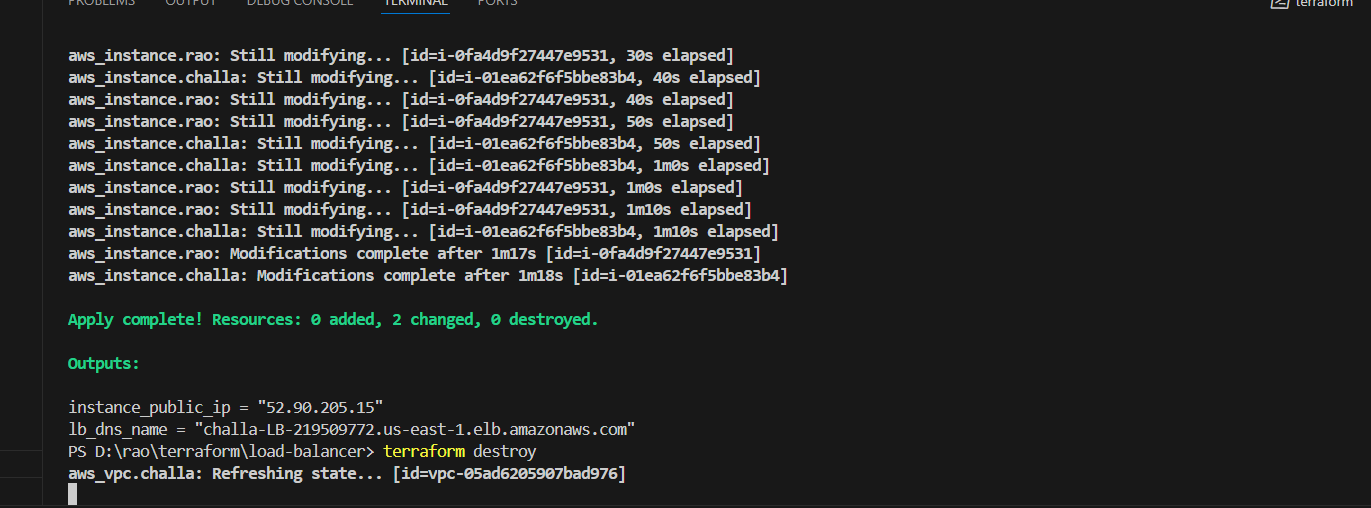


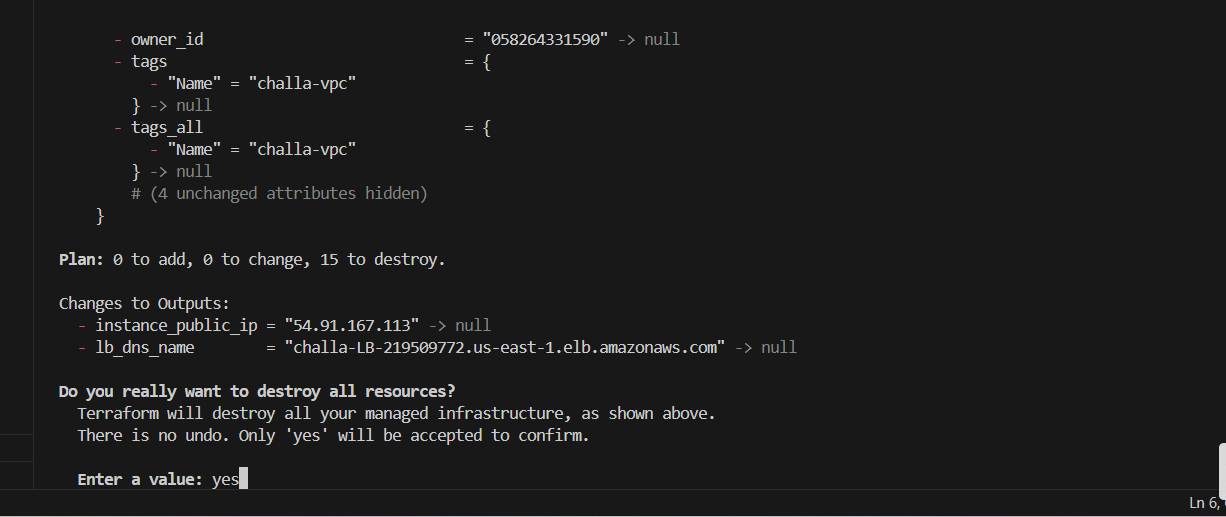


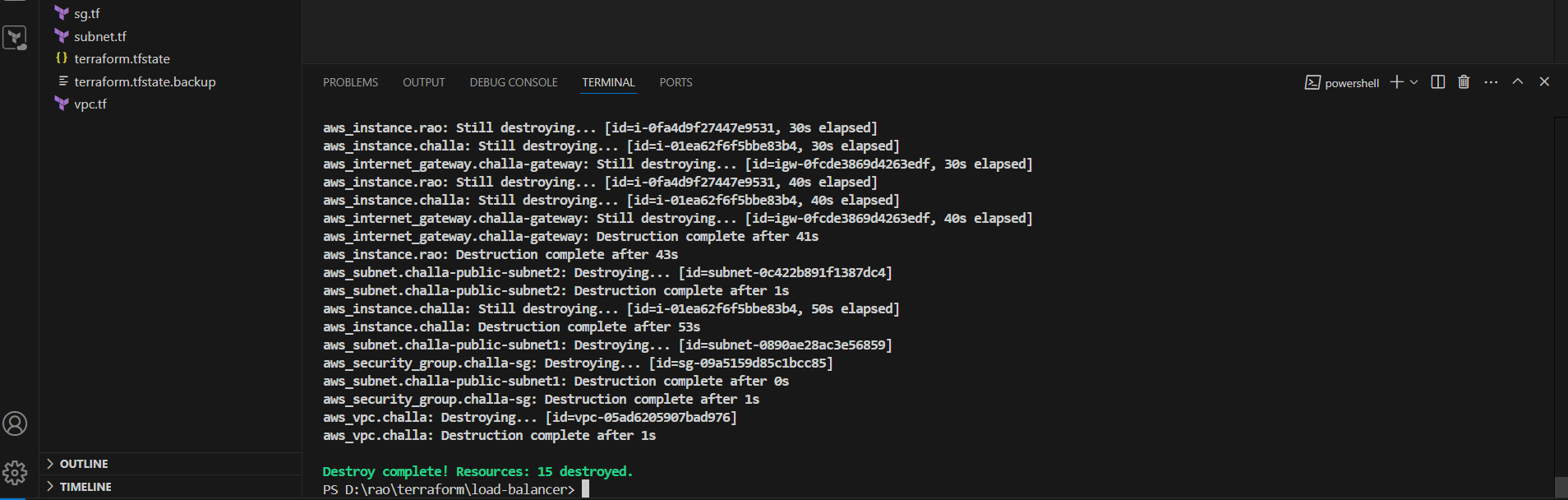
1. Copy dns and paste in google
2. 
3. Refresh
4. 
5. Once go and check instances and load balancers
6. 
7. 
8. 
9. 
10. 
11. 

========================================================================

If you want destroy use #terraform destroy







=================================================================