**TASK-3**

**Create two virtual machines in us-east (web servers)** **Configure load balancer for above servers**

**STEP-1: CREATE AN EC-2 INSTANCE**

**STEP-2: CONNECT TO THE INSTANCE**

* Connect the instance by the following commands

# sudo –i

# apt upadate –y

# apt install unzip –y

* Install amazon CLI

# curl "https://awscli.amazonaws.com/awscli-exe-linux-x86\_64.zip" -o "awscliv2.zip"

unzip awscliv2.zip

sudo ./aws/install

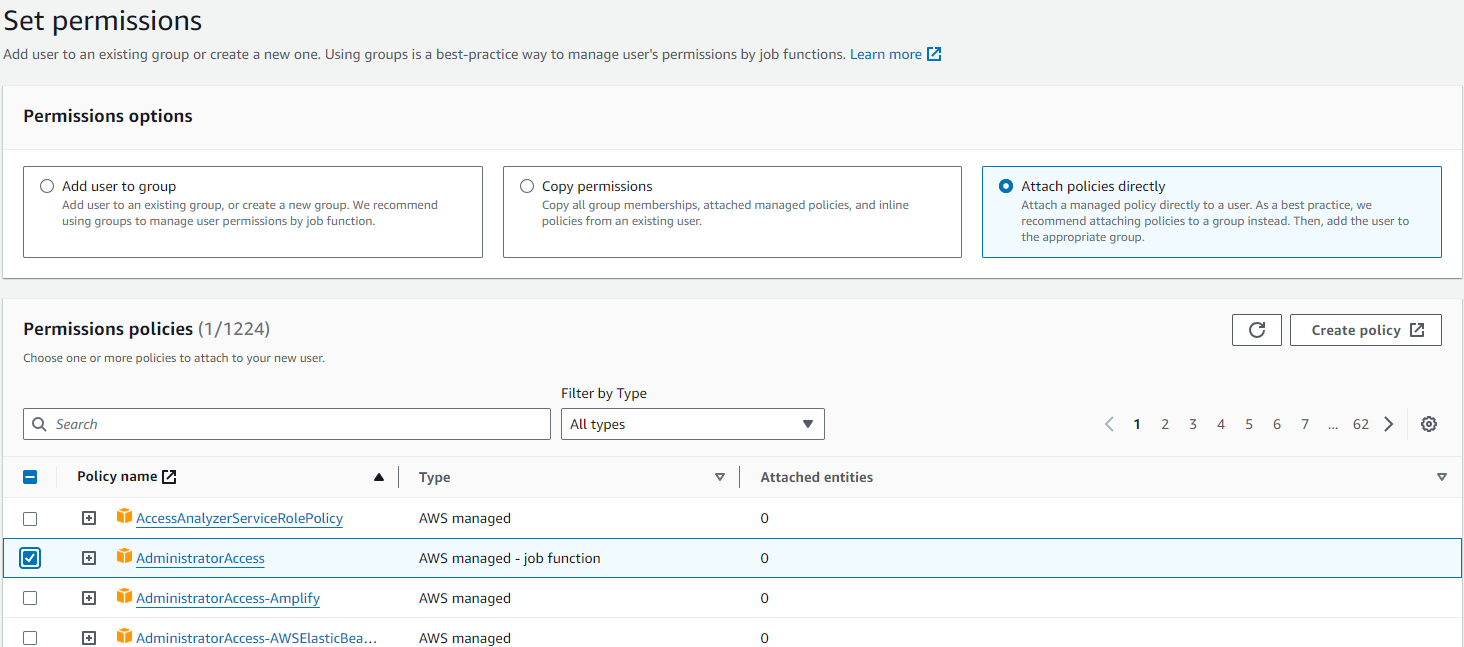
* Install terraform

# wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg

echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb\_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list

sudo apt update && sudo apt install terraform

**STEP-3: GO TO IAM**

* Go to users
* Click on create user
* Give user name and Click on next
* Select “attach policies directly” select “administrator access”
* Click on next
* Click on Create user
* Select the user
* Click on Create Access key
* Select the “Command Line Interface (CLI)”
* Click on Next and Click on Create Access key
* Copy the Access key and Secret Access key

**STEP-4: GO TO SERVER**

* Type the commands

# aws configure

# paste the access key

# paste secret access key

# choose the region

# choose the output format

* Create terraform directory

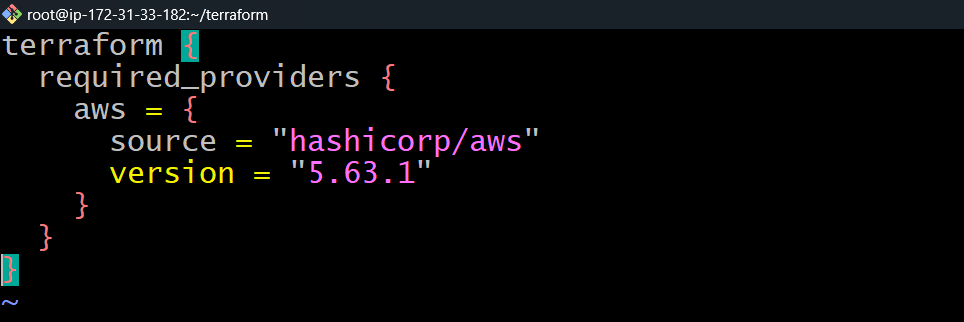
# mkdir terraform

* Go to the directory

# cd terraform

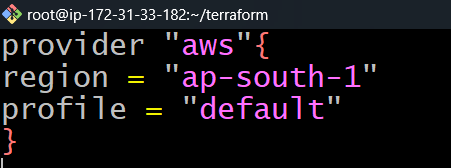
* Create terraformblock

# vi terraformblock.tf



* Create providerblock

# vi providerblock.tf



* Create resourseblock

# vi resourseblock.tf

* Paste the code in resourseblock.tf
* # Create a VPC
* resource "aws\_vpc" "web\_vpc" {
* cidr\_block           = "10.0.0.0/16"
* enable\_dns\_hostnames = true
* tags = {
* Name = "web-vpc"
* }
* }
* # Create an Internet Gateway
* resource "aws\_internet\_gateway" "web\_igw" {
* vpc\_id = aws\_vpc.web\_vpc.id
* tags = {
* Name = "web-igw"
* }
* }
* # Create a subnet in ap-south-1a
* resource "aws\_subnet" "web\_subnet\_1" {
* vpc\_id            = aws\_vpc.web\_vpc.id
* cidr\_block        = "10.0.1.0/24"
* availability\_zone = "ap-south-1a"
* tags = {
* Name = "web-subnet-1"
* }
* }
* # Create a subnet in ap-south-1b
* resource "aws\_subnet" "web\_subnet\_2" {
* vpc\_id            = aws\_vpc.web\_vpc.id
* cidr\_block        = "10.0.2.0/24"
* availability\_zone = "ap-south-1b"
* tags = {
* Name = "web-subnet-2"
* }
* }
* # Create a route table
* resource "aws\_route\_table" "web\_route\_table" {
* vpc\_id = aws\_vpc.web\_vpc.id
* route {
* cidr\_block = "0.0.0.0/0"
* gateway\_id = aws\_internet\_gateway.web\_igw.id
* }
* tags = {
* Name = "web-route-table"
* }
* }
* # Associate the route table with subnet 1
* resource "aws\_route\_table\_association" "web\_route\_assoc\_1" {
* subnet\_id      = aws\_subnet.web\_subnet\_1.id
* route\_table\_id = aws\_route\_table.web\_route\_table.id
* }
* # Associate the route table with subnet 2
* resource "aws\_route\_table\_association" "web\_route\_assoc\_2" {
* subnet\_id      = aws\_subnet.web\_subnet\_2.id
* route\_table\_id = aws\_route\_table.web\_route\_table.id
* }
* # Create a security group for the web servers
* resource "aws\_security\_group" "web\_sg" {
* name        = "web-sg"
* description = "Security group for web servers"
* vpc\_id      = aws\_vpc.web\_vpc.id
* ingress {
* from\_port   = 80
* to\_port     = 80
* 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"]  # Note: In production, restrict this to your IP
* }
* egress {
* from\_port   = 0
* to\_port     = 0
* protocol    = "-1"
* cidr\_blocks = ["0.0.0.0/0"]
* }
* tags = {
* Name = "web-sg"
* }
* }
* # Create two EC2 instances (web servers)
* resource "aws\_instance" "web\_server" {
* count                  = 2
* ami                    = "ami-0c55b159cbfafe1f0"  # Amazon Linux 2 AMI (HVM), SSD Volume Type
* instance\_type          = "t2.micro"
* key\_name               = "your-key-pair-name"  # Replace with your key pair name
* vpc\_security\_group\_ids = [aws\_security\_group.web\_sg.id]
* subnet\_id              = count.index == 0 ? aws\_subnet.web\_subnet\_1.id : aws\_subnet.web\_subnet\_2.id
* user\_data = <<-EOF
* #!/bin/bash
* yum update -y
* yum install -y httpd
* systemctl start httpd
* systemctl enable httpd
* echo "<h1>Hello from $(hostname -f)</h1>" > /var/www/html/index.html
* EOF
* tags = {
* Name = "web-server-${count.index + 1}"
* }
* }
* # Create an Application Load Balancer
* resource "aws\_lb" "web\_alb" {
* name               = "web-alb"
* internal           = false
* load\_balancer\_type = "application"
* security\_groups    = [aws\_security\_group.web\_sg.id]
* subnets            = [aws\_subnet.web\_subnet\_1.id, aws\_subnet.web\_subnet\_2.id]
* tags = {
* Name = "web-alb"
* }
* }
* # Create a target group for the ALB
* resource "aws\_lb\_target\_group" "web\_tg" {
* name     = "web-tg"
* port     = 80
* protocol = "HTTP"
* vpc\_id   = aws\_vpc.web\_vpc.id
* health\_check {
* path                = "/"
* healthy\_threshold   = 2
* unhealthy\_threshold = 10
* }
* }
* # Attach the EC2 instances to the target group
* resource "aws\_lb\_target\_group\_attachment" "web\_tg\_attachment" {
* count            = 2
* target\_group\_arn = aws\_lb\_target\_group.web\_tg.arn
* target\_id        = aws\_instance.web\_server[count.index].id
* port             = 80
* }
* # Create a listener for the ALB
* resource "aws\_lb\_listener" "web\_listener" {
* load\_balancer\_arn = aws\_lb.web\_alb.arn
* port              = "80"
* protocol          = "HTTP"
* default\_action {
* type             = "forward"
* target\_group\_arn = aws\_lb\_target\_group.web\_tg.arn
* }
* }
* # Output the public IPs of the EC2 instances
* output "web\_server\_public\_ips" {
* value = aws\_instance.web\_server[\*].public\_ip
* }
* # Output the DNS name of the load balancer
* output "alb\_dns\_name" {
* value       = aws\_lb.web\_alb.dns\_name
* description = "The DNS name of the Application Load Balancer"
* }

## STEP-5: Running Terraform commands to Launch multiple AWS EC2 instances.

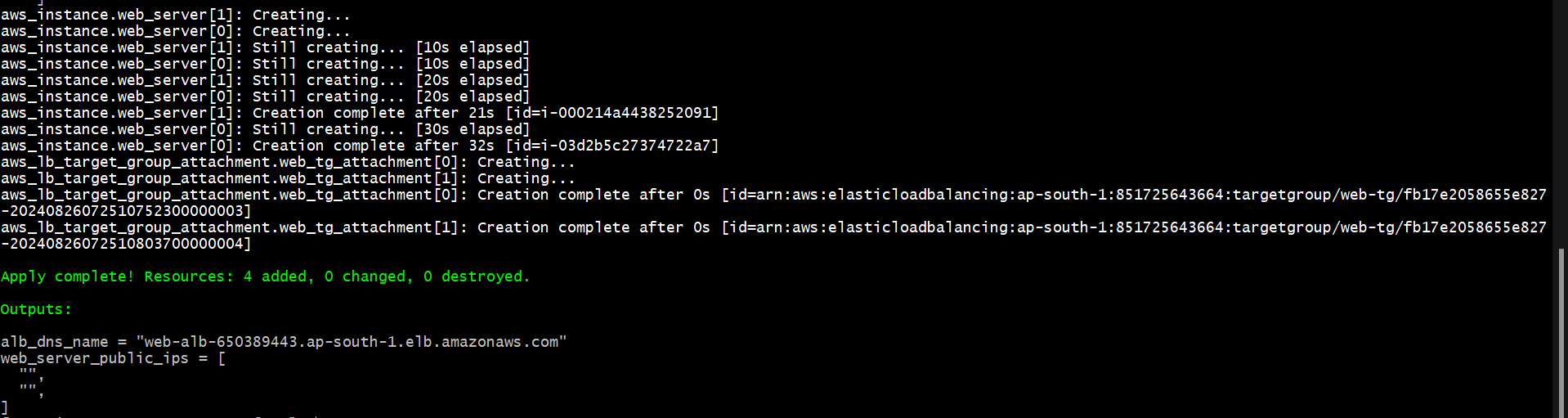
After creating the block perform the terraform commands to launch the instances.

# terraform init

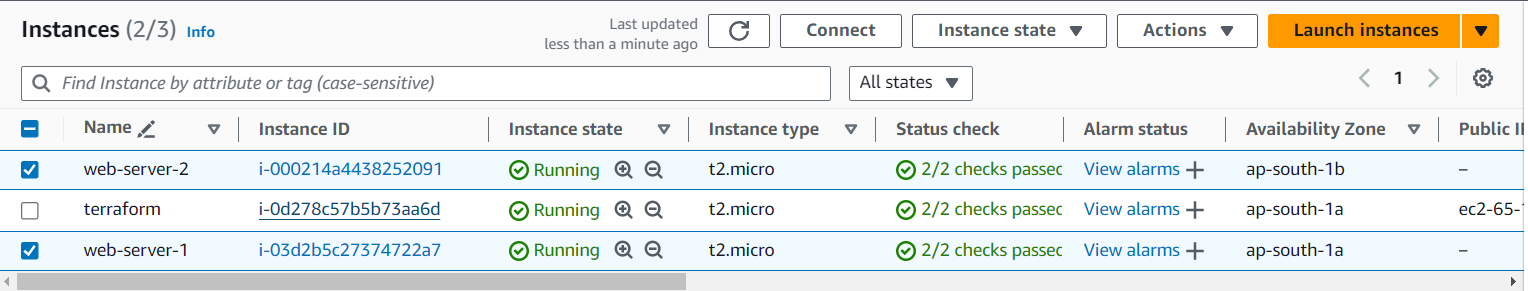
# terraform validate

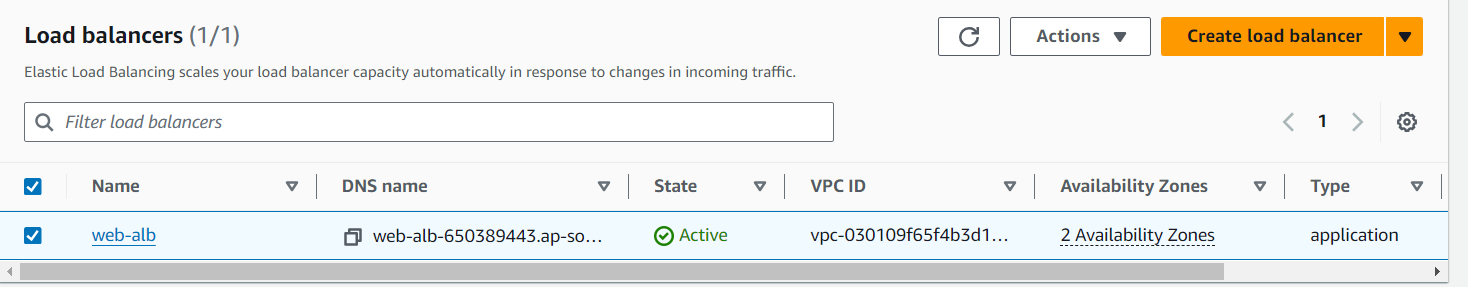
# terraform plan

# terraform apply



After performing the command, go to the AWS console check the instances are running. And check the load balancer is running.





After performing the operations destroy the applications that you created using terraform, by applying the command.

# terraform destroy

