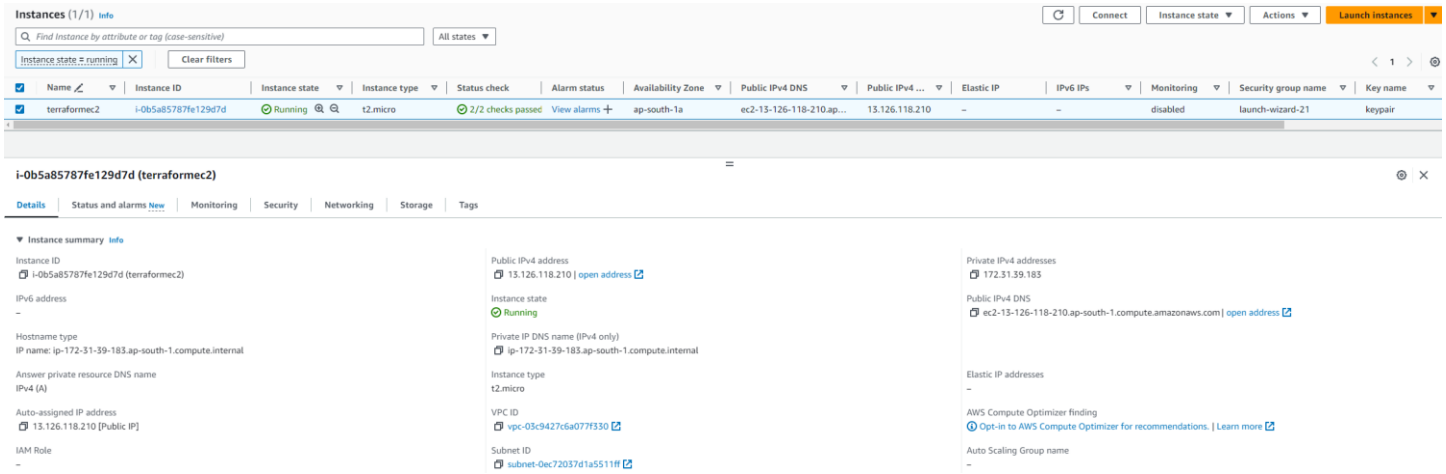
1.Created EC2 instance for Terraform creation



2.EC2 Launched terraform & AWS CLI installed and configured in the EC2 machine

```
ubuntu@ip-172-31-39-183:~/ec2$ terraform version
Terraform v1.8.2
on linux_amd64
+ provider registry.terraform.io/hashicorp/aws v5.48.0
ubuntu@ip-172-31-39-183:~/ec2$ aws --version
aws-cli/2.15.45 Python/3.11.8 Linux/6.5.0-1018-aws exe/x86_64.ubuntu.22 prompt/off
ubuntu@ip-172-31-39-183:~/ec2$
```

3.Creating terraform script file

```
ubuntu@ip-172-31-39-183:~$ cd ec2
ubuntu@ip-172-31-39-183:~/ec2$ nano ec2.tf
ubuntu@ip-172-31-39-183:~/ec2$
```

4.Terraform Script File for Launching 2 EC2 in 2 Regions with Nginx App.

```
  GNU nano 6.2
#Define AWS provider for the first region (ap-south-1) with an alias
provider "aws" {
 alias  = "ap_south_1"
 region = "ap-south-1"
}

provider "aws" {
 alias  = "us_east_1"
 region = "us-east-1"
}

resource "aws_instance" "aws1" {
  provider      = aws.ap_south_1
  ami           = "ami-05e00961530ae1b55"
  instance_type = "t2.micro"
  tags = {
        Name = "Nginx1"
  }
  user_data     = <<-EOF
              #!/bin/bash
              sudo apt-get update
              sudo apt-get install nginx -y
              sudo service nginx start
              echo "Hello..! This is direct nginx server creation using terraform" >> /usr/share/nginx/html/index.html
              sudo service nginx restart
              EOF

}

resource "aws_instance" "aws2" {
  provider      = aws.us_east_1
  ami           = "ami-0e001c9271cf7f3b9"
  instance_type = "t2.micro"
  tags = {
        Name = "Nginx2"
  }
  user_data     = <<-EOF
              #!/bin/bash
              sudo apt-get update
              sudo apt-get install nginx -y
              sudo service nginx start
              echo "Hello..! This is derict nginx server creation using Terraform" >> /usr/share/nginx/html/index.html
              sudo service nginx restart
              EOF

}
```

5.Terraform file created and Terraform initialized in the directory

```
ubuntu@ip-172-31-39-183:~/ec2$ terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.48.0

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.
ubuntu@ip-172-31-39-183:~/ec2$
```

6.First EC2 machine Plan

```
ubuntu@ip-172-31-39-183:~/ec2$ terraform plan

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_instance.aws1 will be created
  + resource "aws_instance" "aws1" {
      + ami                                  = "ami-05e00961530ae1b55"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_partition_number           = (known after apply)
      + primary_network_interface_id         = (known after apply)
      + private_dns                          = (known after apply)
      + private_ip                           = (known after apply)
      + public_dns                           = (known after apply)
      + public_ip                            = (known after apply)
      + secondary_private_ips                = (known after apply)
      + security_groups                      = (known after apply)
      + source_dest_check                    = true
      + spot_instance_request_id             = (known after apply)
      + subnet_id                            = (known after apply)
      + tags                                 = {
          + "Name" = "Nginx1"
        }
      + tags_all                             = {
          + "Name" = "Nginx1"
        }
      + tenancy                              = (known after apply)
      + user_data                            = "757ff0aae12ff9c94b9e82eb0400868a47f63b0e"
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)
    }
```

7.Second EC2 machine Plan and Both Plan Added

```
  # aws_instance.aws2 will be created
  + resource "aws_instance" "aws2" {
      + ami                                  = "ami-0e001c9271cf7f3b9"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_partition_number           = (known after apply)
      + primary_network_interface_id         = (known after apply)
      + private_dns                          = (known after apply)
      + private_ip                           = (known after apply)
      + public_dns                           = (known after apply)
      + public_ip                            = (known after apply)
      + secondary_private_ips                = (known after apply)
      + security_groups                      = (known after apply)
      + source_dest_check                    = true
      + spot_instance_request_id             = (known after apply)
      + subnet_id                            = (known after apply)
      + tags                                 = {
          + "Name" = "Nginx2"
        }
      + tags_all                             = {
          + "Name" = "Nginx2"
        }
      + tenancy                              = (known after apply)
      + user_data                            = "0a064c76c421a04ef34b83156f5aa417ceb65644"
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)
    }

Plan: 2 to add, 0 to change, 0 to destroy.


Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
ubuntu@ip-172-31-39-183:~/ec2$
```

8.Terraform Applied with the Script File

First EC2 is creating:

```
ubuntu@ip-172-31-39-183:~/ec2$ terraform apply

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

Terraform will perform the following actions:

  # aws_instance.aws1 will be created
  + resource "aws_instance" "aws1" {
      + ami                                  = "ami-05e00961530ae1b55"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
```

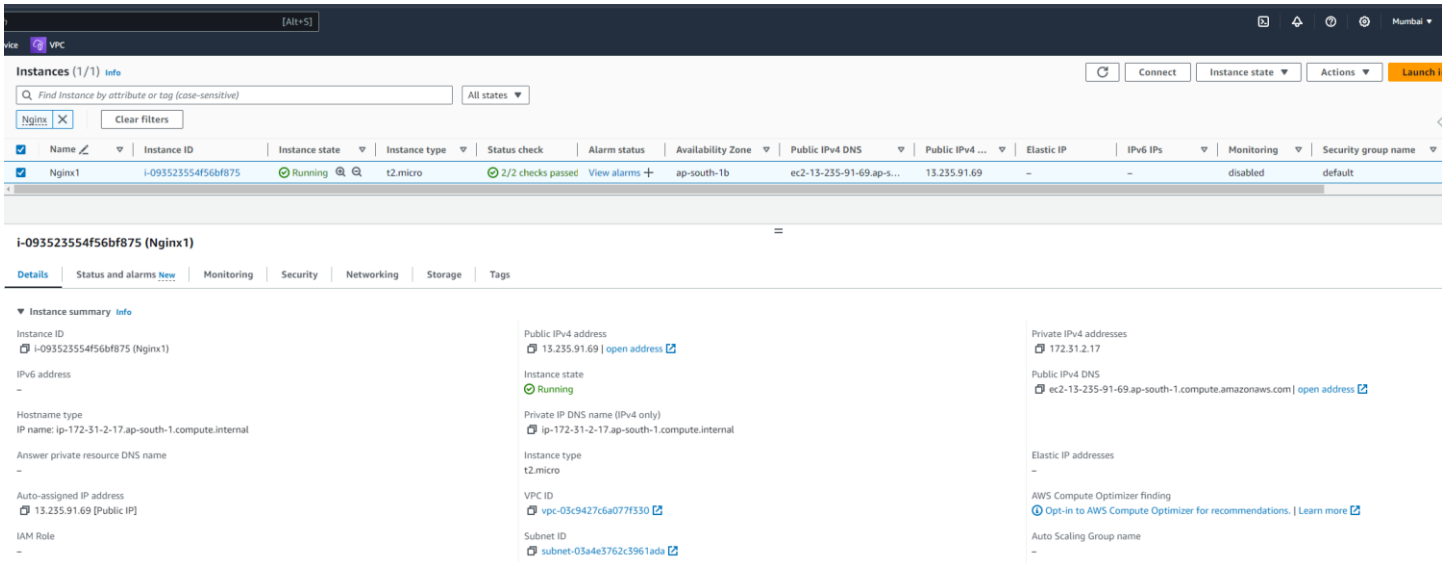
Second EC2 is creating:

```
  # aws_instance.aws2 will be created
  + resource "aws_instance" "aws2" {
      + ami                                  = "ami-0e001c9271cf7f3b9"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
```

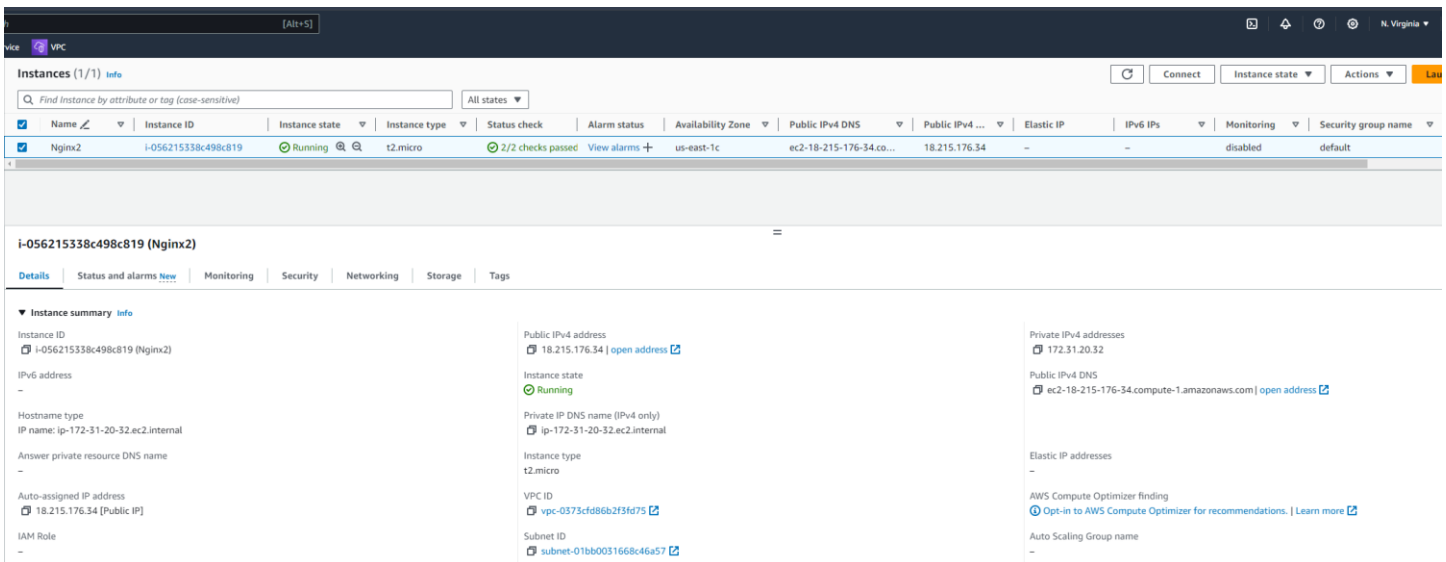
9.Two EC2 machines are created in Two different Regions with Nginx server

```
Plan: 2 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.aws1: Creating...
aws_instance.aws2: Creating...
aws_instance.aws1: Still creating... [10s elapsed]
aws_instance.aws2: Still creating... [10s elapsed]
aws_instance.aws1: Still creating... [20s elapsed]
aws_instance.aws2: Still creating... [20s elapsed]
aws_instance.aws1: Still creating... [30s elapsed]
aws_instance.aws2: Still creating... [30s elapsed]
aws_instance.aws1: Creation complete after 32s [id=i-093523554f56bf875]
aws_instance.aws2: Creation complete after 36s [id=i-056215338c498c819]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
ubuntu@ip-172-31-39-183:~/ec2$
```

10.First EC2 is Running in ap-south-1 Region with Nginx



11.Second EC2 is Running in us-east-1 Region with Nginx



12.First Nginx running in ap-south-1 region IP address

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.

13.Second Nginx running in us-east-1 region IP address

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.