**Assignment-1**

**Create Two Virtual machines and add the Load balancer to them using terraform code.**

step1: Create the main.tf file.

#provider Block

provider "azurerm" {

  features {}

  client\_id       = "1f79e427-2ac4-4eb6-9ca0-f4dd4b3f31ee"

  client\_secret   = "Vb18Q~Zj4DMUeKgAiQ-Lpm~pj92y.7s7SLBaAcyG"

  tenant\_id       = "4a623a04-9917-4ee2-8f59-02586964c992"

  subscription\_id = "51c6d184-6756-4a9a-ade4-cd0f3d57cded"

}

Step2: Create the Resource group using terraform code.

#resource group creation

resource "azurerm\_resource\_group" "rg" {

    name = var.rg-name

    location = var.loc-name

}

#variable blocks

variable "rg-name" {

    type = string

    default = "RG-01"

    description = "resource group name"

}

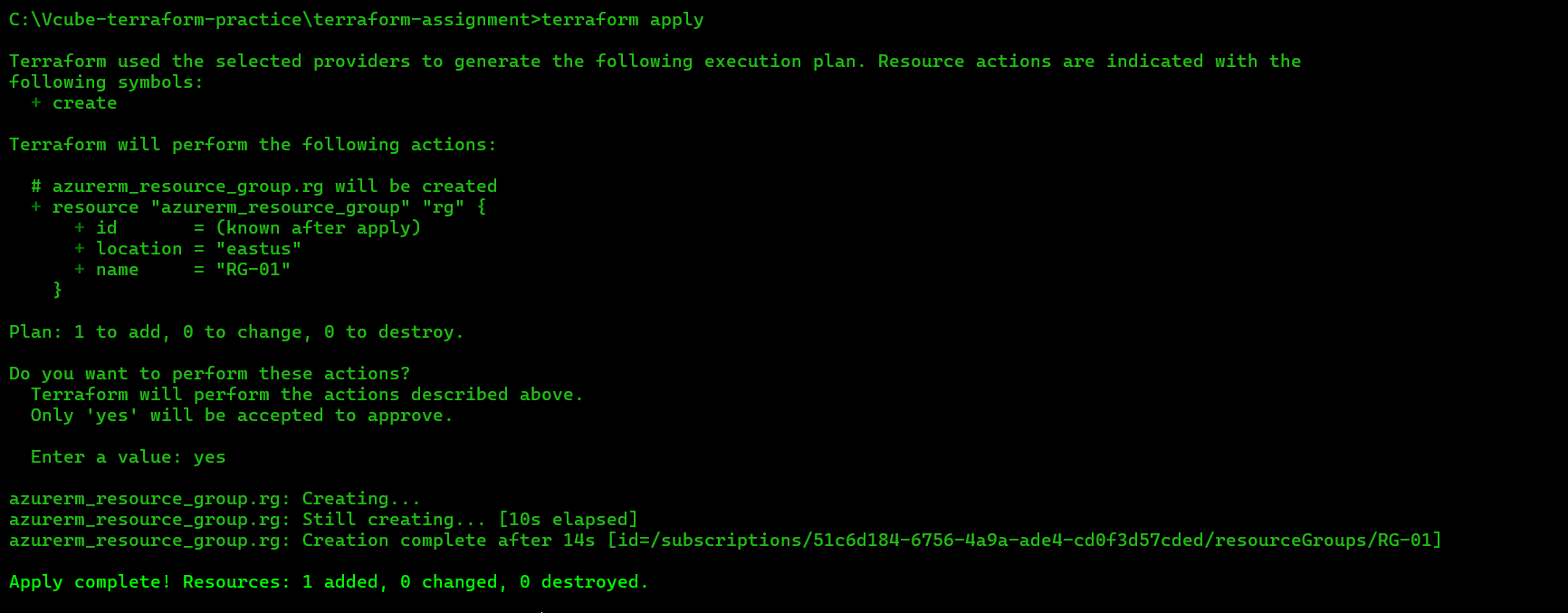
variable "loc-name" {

    type = string

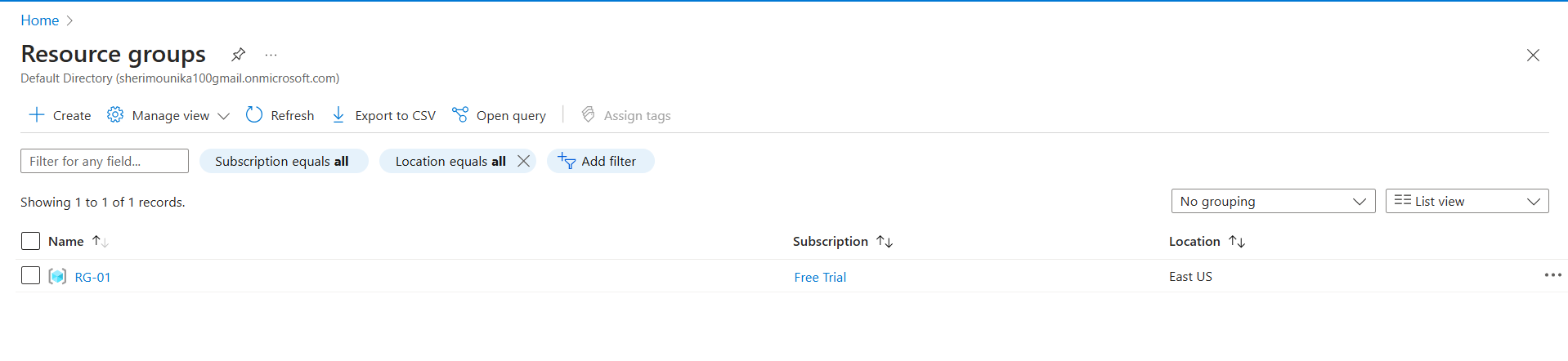
    default = "eastus"

    description = "location name"

}



**Fig:** Terraform apply.



**Fig:** resource group (RG-01).

**Step2:** Create the Virtual network in resource group (RG-01).

#virtual netvork creation

resource "virtual\_network" "Vnet" {

    name = var.Vnet-name

    resource\_group\_name = azurerm\_resource\_group.rg.name

    location = azurerm\_resource\_group.rg.location

    address\_space = var.vnet-ip-range

}

variable "Vnet-name" {

    type = string

    default = "Vnet-01"

    description = "virtual network name"

}

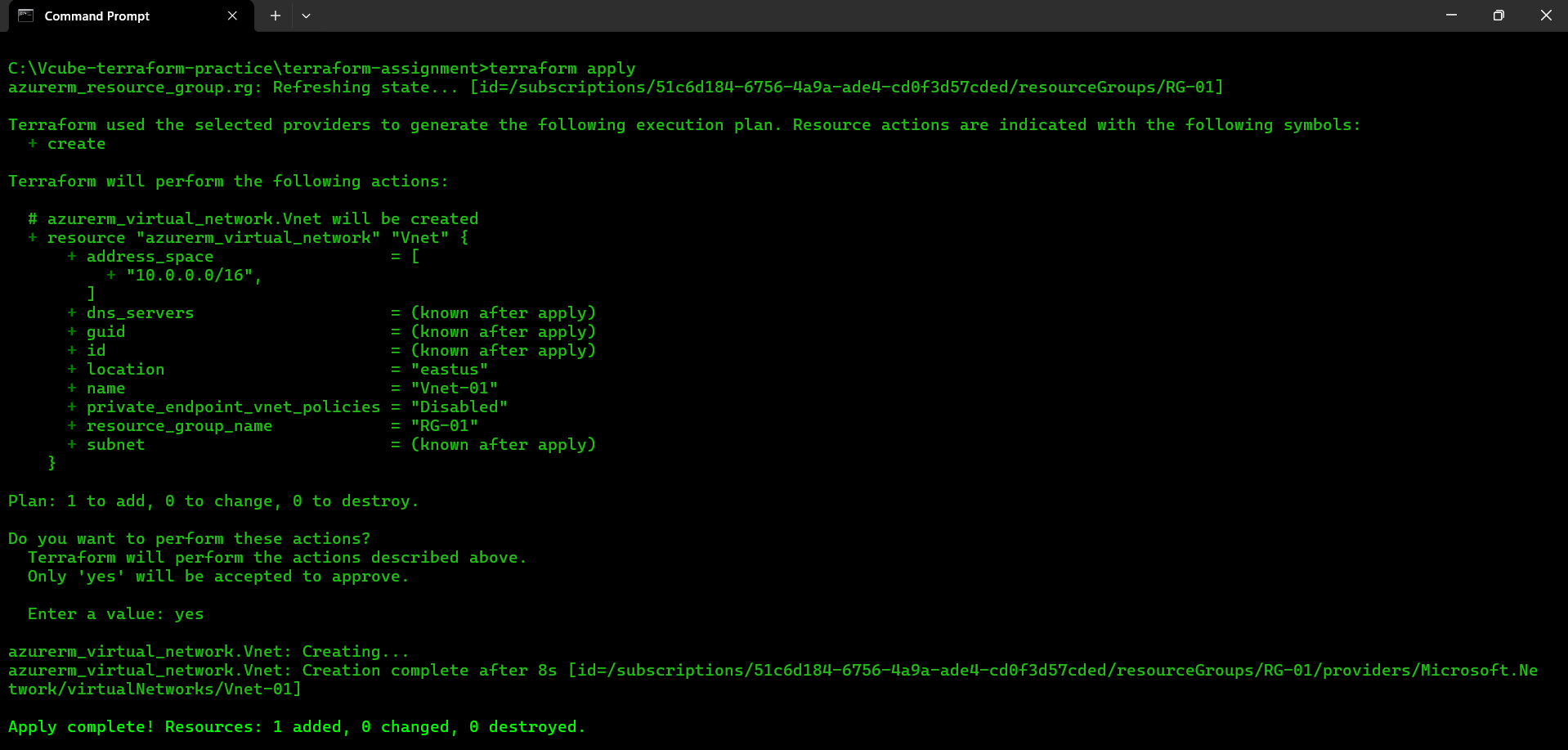
variable "vnet-ip-range" {

    type = list(string)

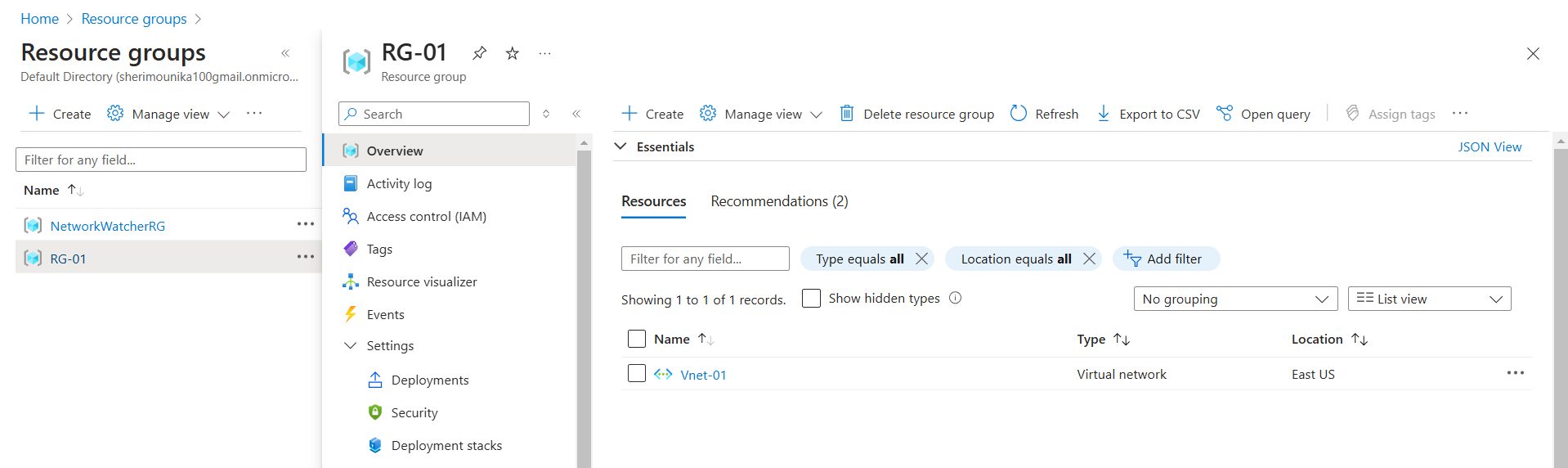
    default = [ "10.0.0.0/16" ]

    description = "virtual network IP range"

}



**Fig:** Terraform apply.



**Fig:** Virtual Network. (Vnet-01).

Step3: create two subnets (subnet-0 & subnet-1).

#subnet creation

resource "azurerm\_subnet" "sub" {

  count = 2

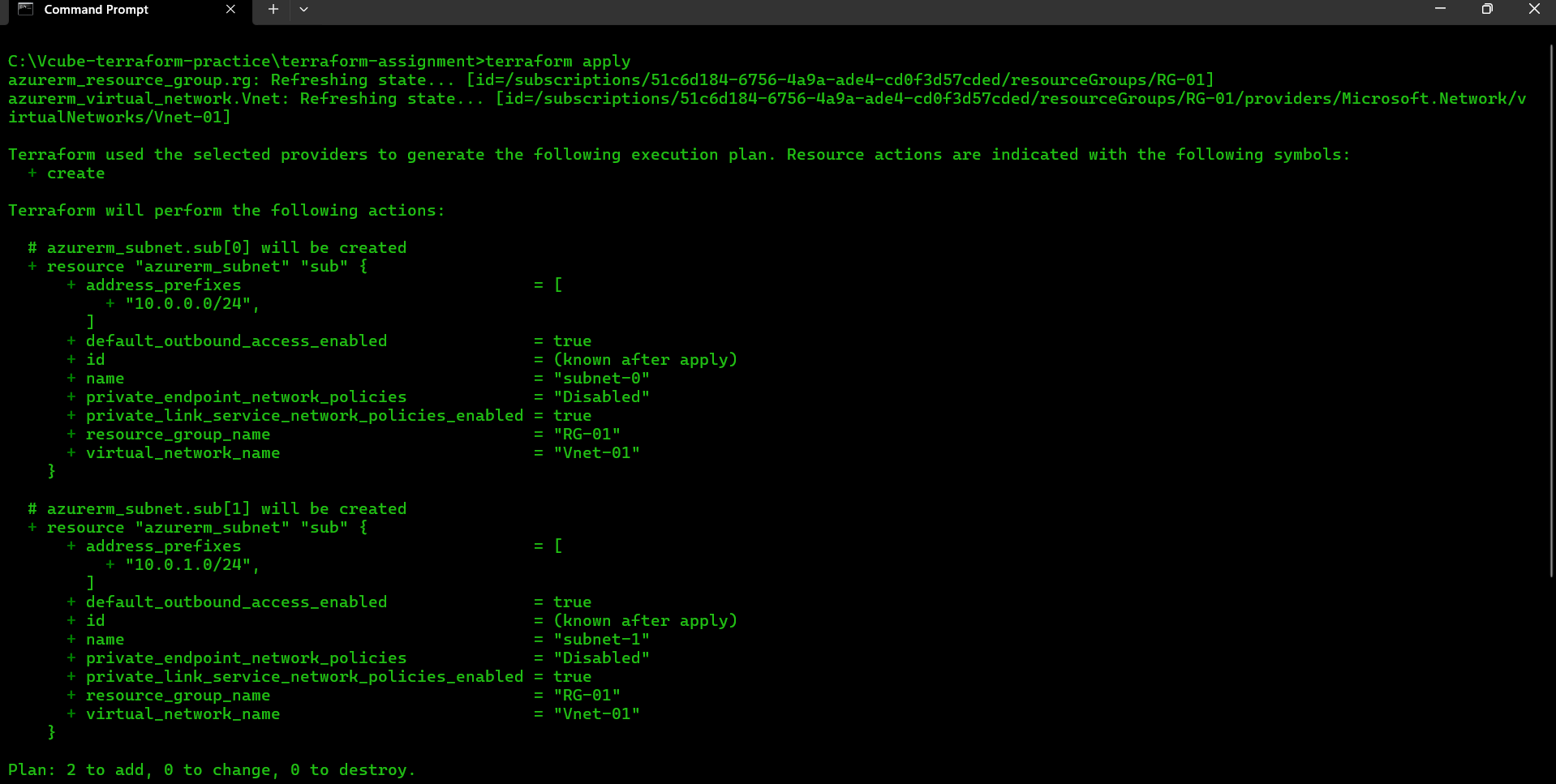
  name = "subnet-${count.index}"

  virtual\_network\_name = azurerm\_virtual\_network.Vnet.name

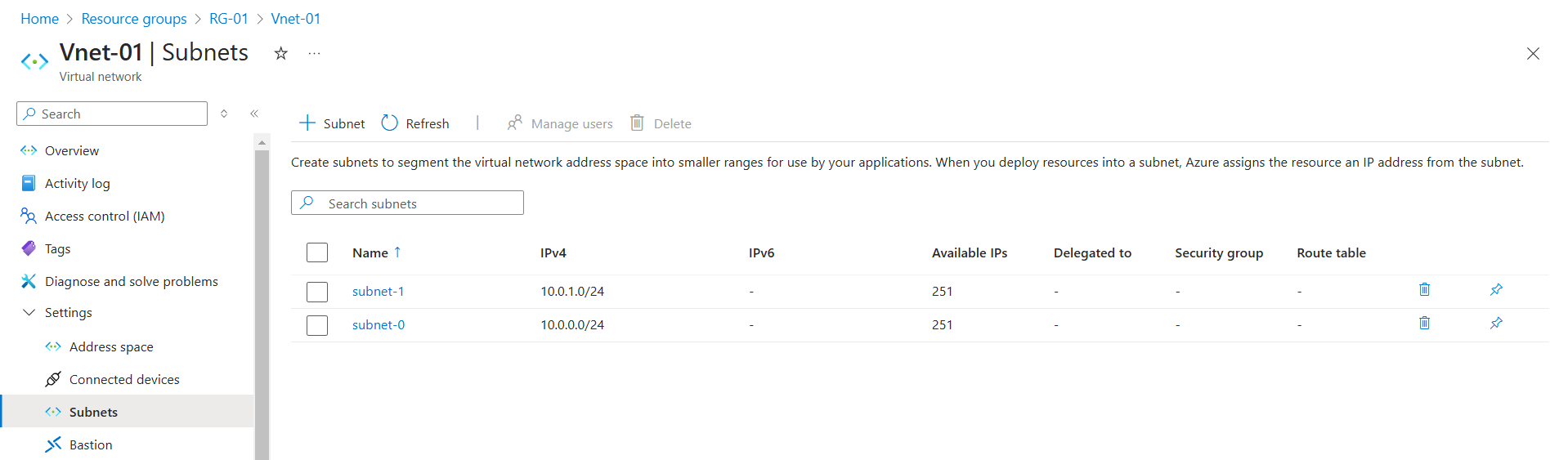
  resource\_group\_name = azurerm\_resource\_group.rg.name

  address\_prefixes = ["10.0.${count.index}.0/24"]

}



**Fig:** Terraform apply.



**Fig:** Subnets.

**Step4:** Create two Public IP’s (publicIP-0 & publicIP-1).

# public ip's creation

resource "azurerm\_public\_ip" "pips" {

  count = 2

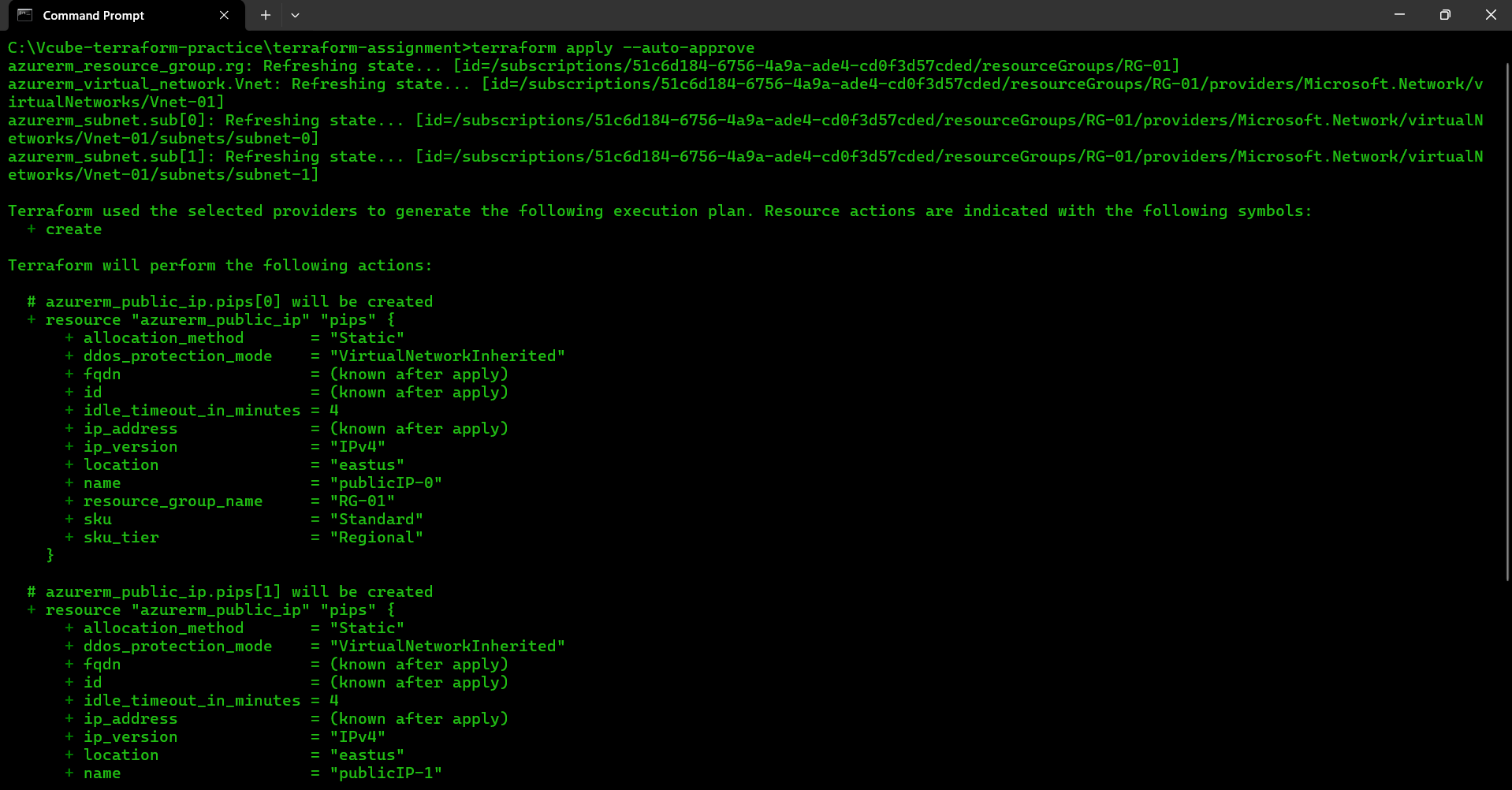
  name = "publicIP-${count.index}"

  resource\_group\_name = azurerm\_resource\_group.rg.name

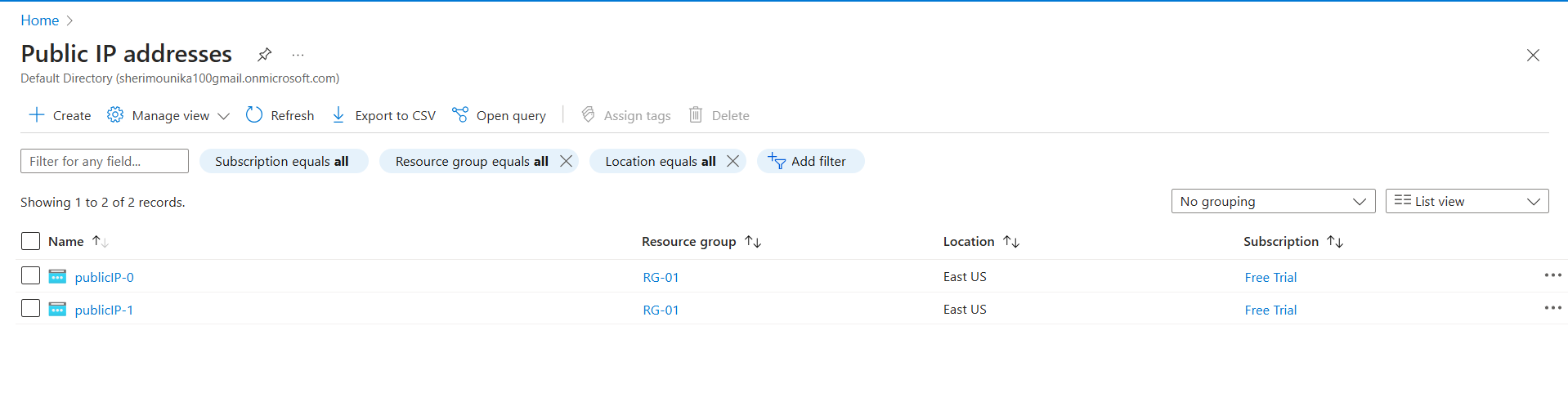
  location = azurerm\_resource\_group.rg.location

  allocation\_method = "Static"

}



**Fig:** Terraform apply.



**Fig:** public IP addresses.

**Step5:** Create two NIC cards(nic-0 & nic-1).

# NIC cards creation

resource "azurerm\_network\_interface" "NICards" {

  count               = 2

  name                = "nic-${count.index}"

  location            = azurerm\_resource\_group.rg.location

  resource\_group\_name = azurerm\_resource\_group.rg.name

  ip\_configuration {

    name                          = "internal"

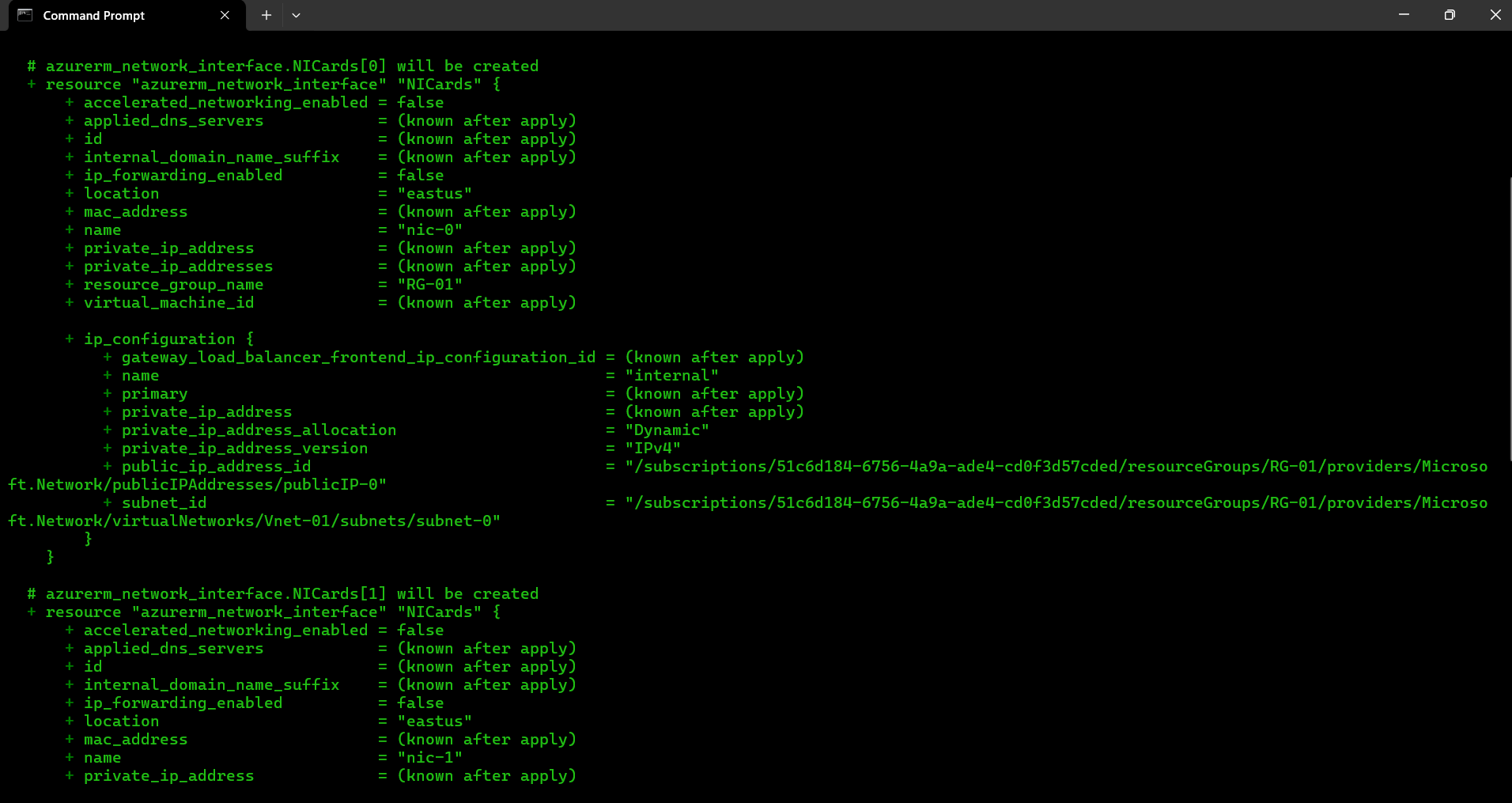
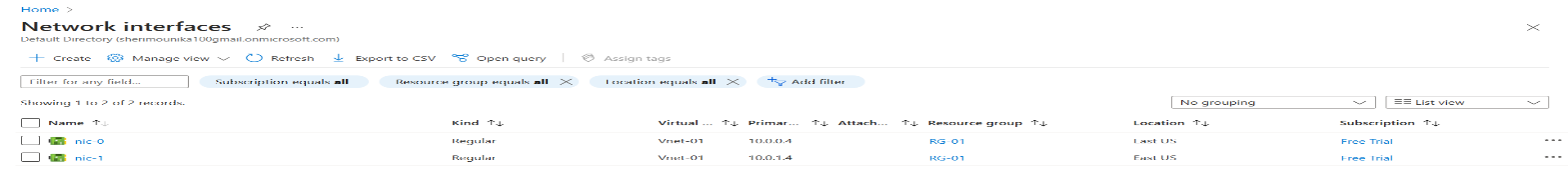
    subnet\_id                     = azurerm\_subnet.sub[count.index].id

    private\_ip\_address\_allocation = "Dynamic"

    public\_ip\_address\_id          = azurerm\_public\_ip.pips[count.index].id

  }

}

****

**Fig:** NIC cards.

**Step6:** Create the network security group (NSG-0) and write its rule.

#Network security group (NSG-0)

  resource "azurerm\_network\_security\_group" "TFNSG" {

  name                = "NSG-0"

  location            = azurerm\_resource\_group.rg.location

  resource\_group\_name = azurerm\_resource\_group.rg.name

}

#network security rules-1

resource "azurerm\_network\_security\_rule" "allow\_ssh" {

  name                         = "AllowSSH"

  priority                     = 100

  direction                    = "Inbound"

  access                       = "Allow"

  protocol                     = "Tcp"

  source\_port\_range            = "\*"

  destination\_port\_range       = "22"

  source\_address\_prefix        = "\*"

  destination\_address\_prefix = "\*"

  resource\_group\_name          = azurerm\_resource\_group.rg.name

  network\_security\_group\_name  = azurerm\_network\_security\_group.TFNSG.name

}

#network security rules-2

resource "azurerm\_network\_security\_rule" "allow\_http" {

  name                         = "AllowHTTP"

  priority                     = 120

  direction                    = "Inbound"

  access                       = "Allow"

  protocol                     = "Tcp"

  source\_port\_range            = "\*"

  destination\_port\_range       = "80"

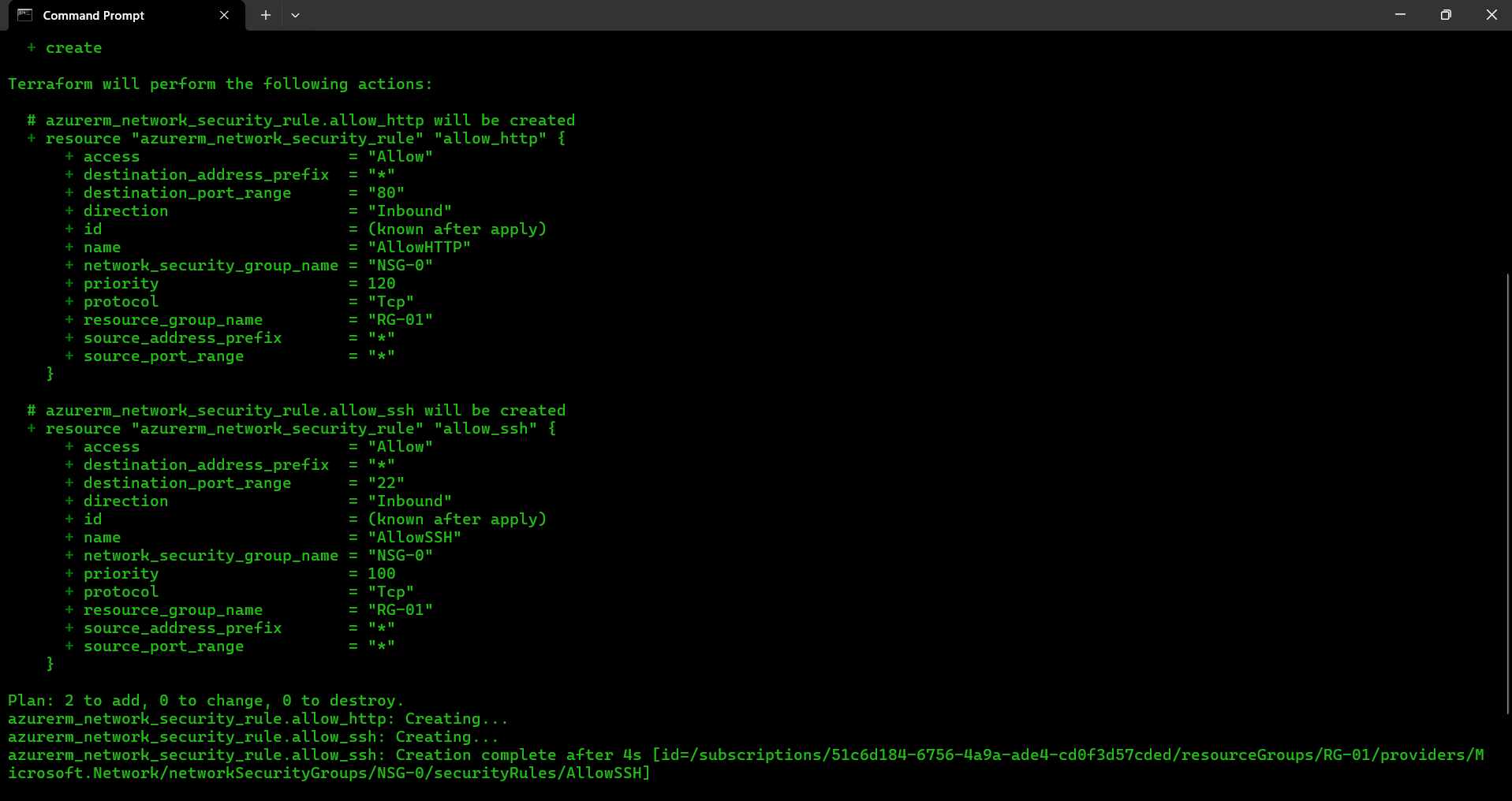
  source\_address\_prefix        = "\*"

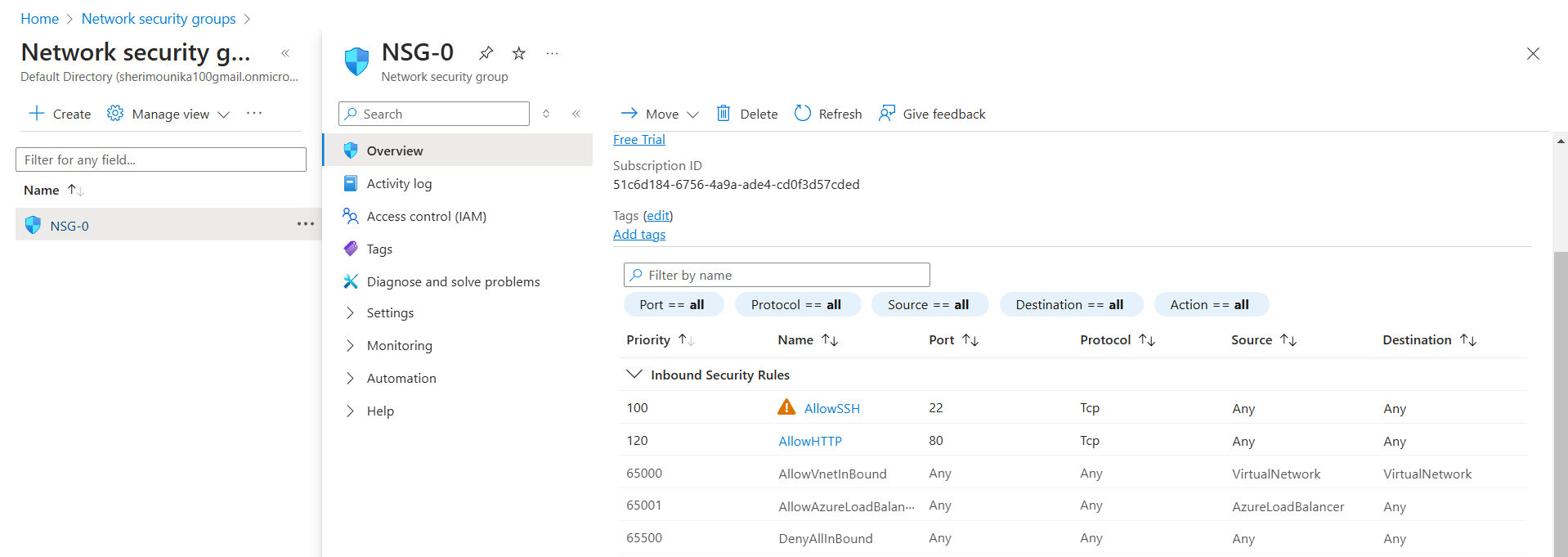
  destination\_address\_prefix = "\*"

  resource\_group\_name          = azurerm\_resource\_group.rg.name

  network\_security\_group\_name  = azurerm\_network\_security\_group.TFNSG.name

}





**Fig:** NSG-0 inbound rules.

**Step7:** Assign the NSG-0 to the subnet-0.

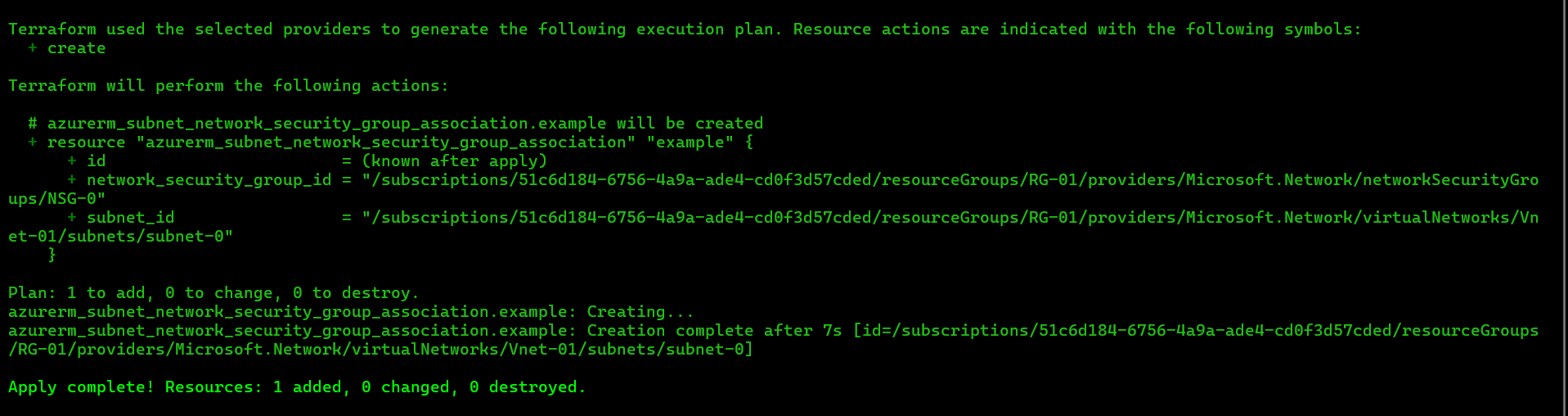
 #assigning of NSG to subnet-0

 resource "azurerm\_subnet\_network\_security\_group\_association" "example" {

  subnet\_id                 = azurerm\_subnet.sub[0].id

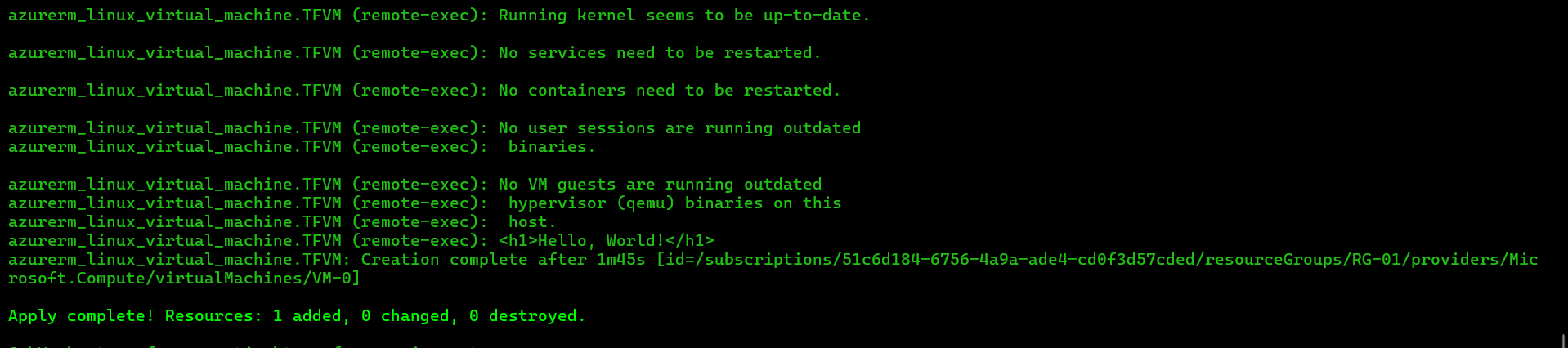
  network\_security\_group\_id = azurerm\_network\_security\_group.TFNSG.id

 }



**Fig:** Terraform apply.

**Step8:** Create the Linux VM (VM-0) with NSG-0, publicIP-0, NIC-0 in subnet-0 and install Nginx in it.



 #Linux Virtual machine creation

 resource "azurerm\_linux\_virtual\_machine" "TFVM" {

  name                            = "VM-0"

  resource\_group\_name             = azurerm\_resource\_group.rg.name

  location                        = azurerm\_resource\_group.rg.location

  size                            = "Standard\_B1s"

  admin\_username                  = "harish"

  admin\_password                  = "Harish@123456789"

  network\_interface\_ids = [

    azurerm\_network\_interface.NICards[0].id,

  ]

  disable\_password\_authentication = false # Important: Allow password auth

  source\_image\_reference {

    publisher = "Canonical"

    offer     = "0001-com-ubuntu-server-jammy" # Ubuntu 22.04 LTS

    sku       = "22\_04-lts-gen2" # or "22\_04-lts"

    version   = "latest"

  }

  os\_disk {

    storage\_account\_type = "Standard\_LRS"

    caching              = "ReadWrite"

  }

  #provisioning block

  provisioner "remote-exec" {

    inline = [

      "sudo apt-get update",

      "sudo apt-get install -y nginx",

      "echo '<h1>Hello, World!</h1>' | sudo tee /var/www/html/index.html",

    ]

    connection {

        type = "ssh"

      user     = "harish"

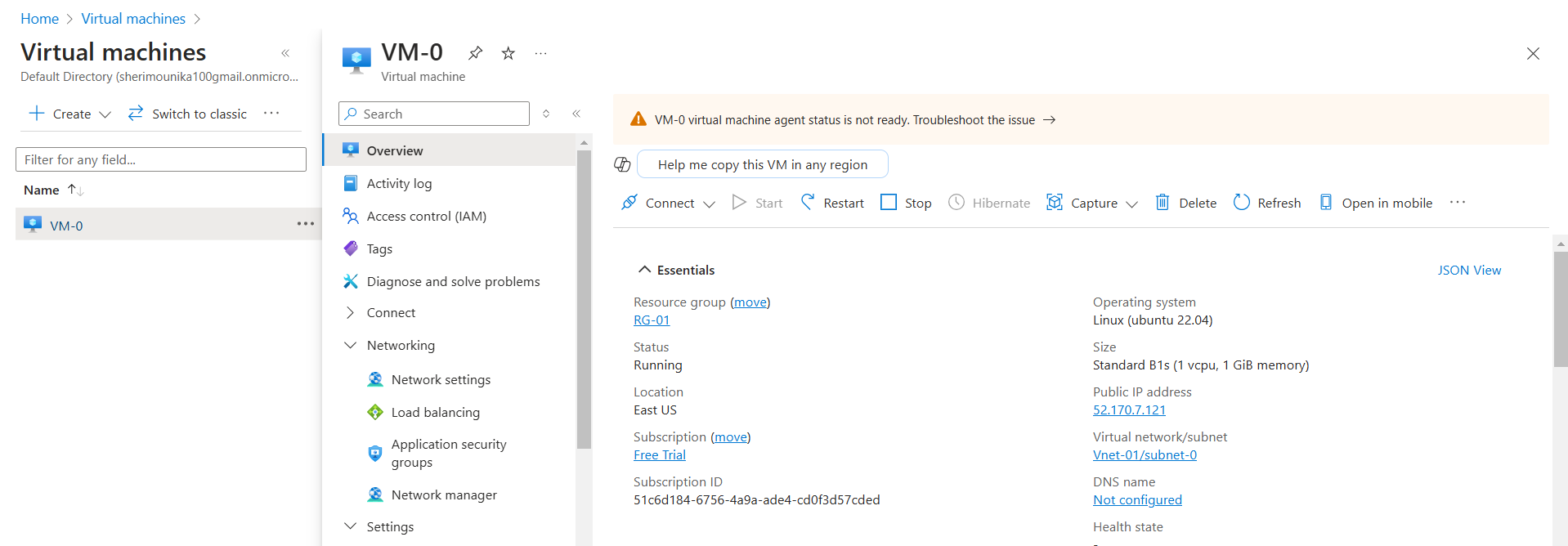
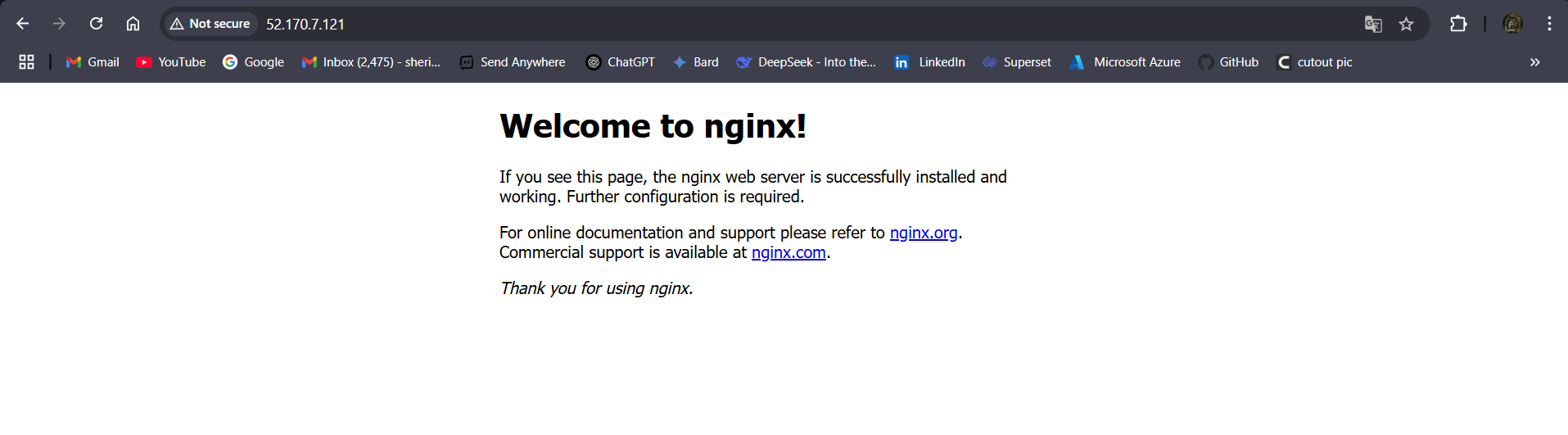
      password = "Harish@123456789"

      host     = azurerm\_public\_ip.pips[0].ip\_address

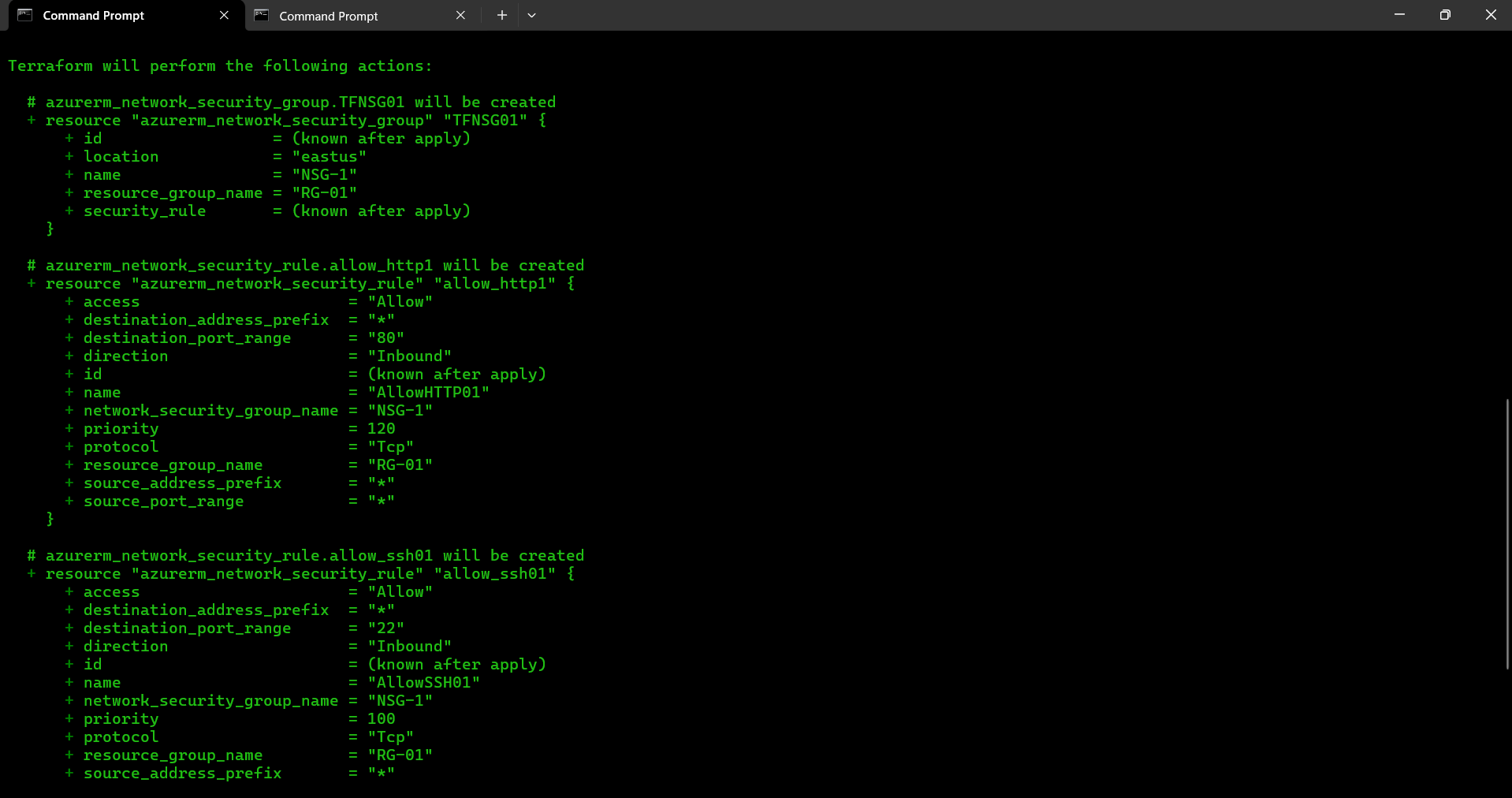
    }

  }

}

 **Fig:** Virtual machine (VM-0).

**Step9:** Similarly create the NSG (NSG-1) and write inbound rules for it.



**Fig:** terraform apply to create NSG-1 and write the inbound rules for HTTP &SSH.

**Fig:** terraform code to crate NSG -1 and its inbound rules.

#Network security group (NSG-1)

  resource "azurerm\_network\_security\_group" "TFNSG01" {

  name                = "NSG-1"

  location            = azurerm\_resource\_group.rg.location

  resource\_group\_name = azurerm\_resource\_group.rg.name

}

#network security rules-1

resource "azurerm\_network\_security\_rule" "allow\_ssh01" {

  name                         = "AllowSSH01"

  priority                     = 100

  direction                    = "Inbound"

  access                       = "Allow"

  protocol                     = "Tcp"

  source\_port\_range            = "\*"

  destination\_port\_range       = "22"

  source\_address\_prefix        = "\*"

  destination\_address\_prefix = "\*"

  resource\_group\_name          = azurerm\_resource\_group.rg.name

  network\_security\_group\_name  = azurerm\_network\_security\_group.TFNSG01.name

}

#network security rules-2

resource "azurerm\_network\_security\_rule" "allow\_http1" {

  name                         = "AllowHTTP01"

  priority                     = 120

  direction                    = "Inbound"

  access                       = "Allow"

  protocol                     = "Tcp"

  source\_port\_range            = "\*"

  destination\_port\_range       = "80"

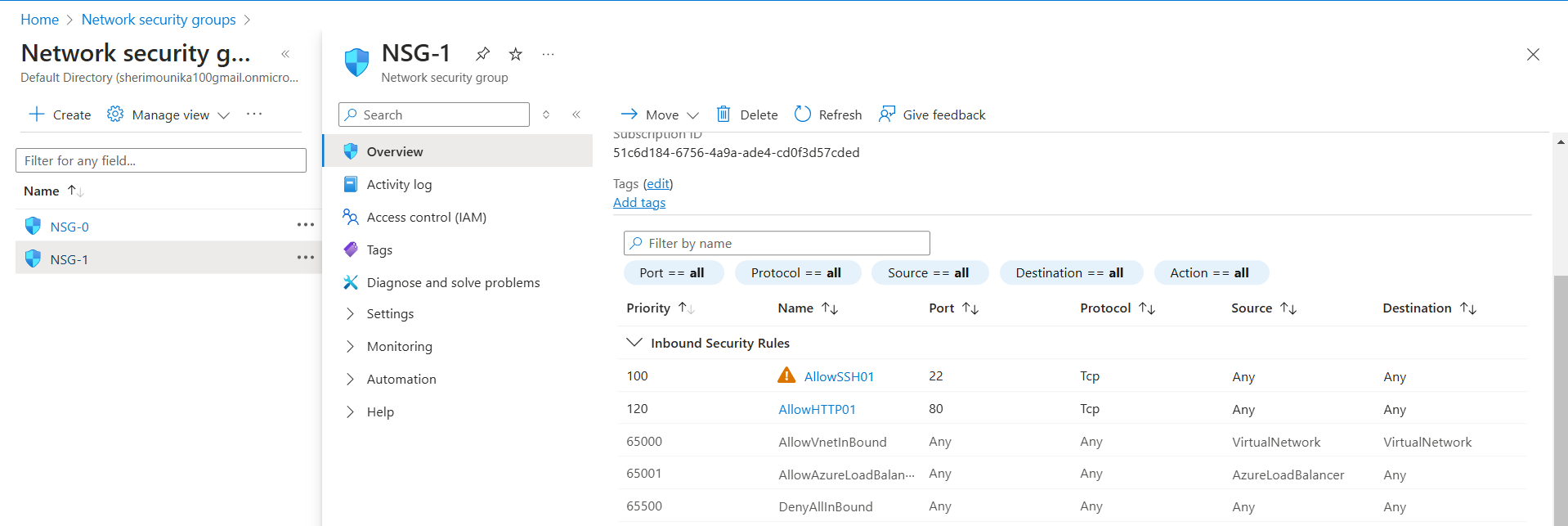
  source\_address\_prefix        = "\*"

  destination\_address\_prefix = "\*"

  resource\_group\_name          = azurerm\_resource\_group.rg.name

  network\_security\_group\_name  = azurerm\_network\_security\_group.TFNSG01.name

}



**Fig:** NSG-1 by allowing HTTP & SSH ports.

**Step10:** Assign the NSG-1 to the subnet-1.

#assigning of nsg to subnet-1

 resource "azurerm\_subnet\_network\_security\_group\_association" "assign" {

  subnet\_id                 = azurerm\_subnet.sub[1].id

  network\_security\_group\_id = azurerm\_network\_security\_group.TFNSG01.id

 }

**Fig:** terraform code to assign the NSG-1 to the subnet-1.



**Fig:** terraform apply.

**Step11:** Create the VM (VM-1) with NSG-1, NIC-1, publicIP-1 in subnet-1 and install Nginx in it.

#Linux Virtual machine creation

 resource "azurerm\_linux\_virtual\_machine" "TFVM01" {

  name                            = "VM-1"

  resource\_group\_name             = azurerm\_resource\_group.rg.name

  location                        = azurerm\_resource\_group.rg.location

  size                            = "Standard\_B1s"

  admin\_username                  = "harish"

  admin\_password                  = "Harish@123456789"

  network\_interface\_ids = [

    azurerm\_network\_interface.NICards[1].id,

  ]

  disable\_password\_authentication = false # Important: Allow password auth

  source\_image\_reference {

    publisher = "Canonical"

    offer     = "0001-com-ubuntu-server-jammy" # Ubuntu 22.04 LTS

    sku       = "22\_04-lts-gen2" # or "22\_04-lts"

    version   = "latest"

  }

  os\_disk {

    storage\_account\_type = "Standard\_LRS"

    caching              = "ReadWrite"

  }

  #provisioning block

  provisioner "remote-exec" {

    inline = [

      "sudo apt-get update",

      "sudo apt-get install -y nginx"

    ]

    connection {

        type = "ssh"

      user     = "harish"

      password = "Harish@123456789"

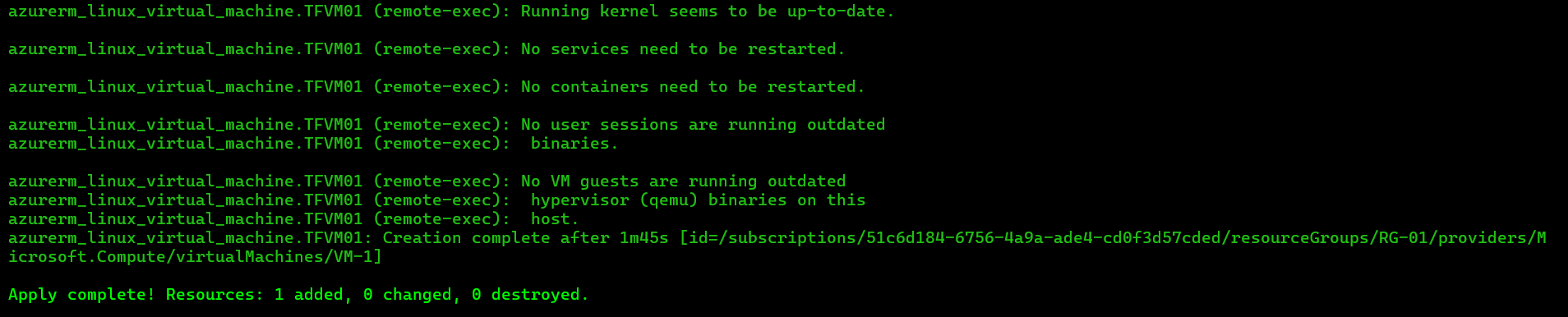
      host     = azurerm\_public\_ip.pips[1].ip\_address

    }

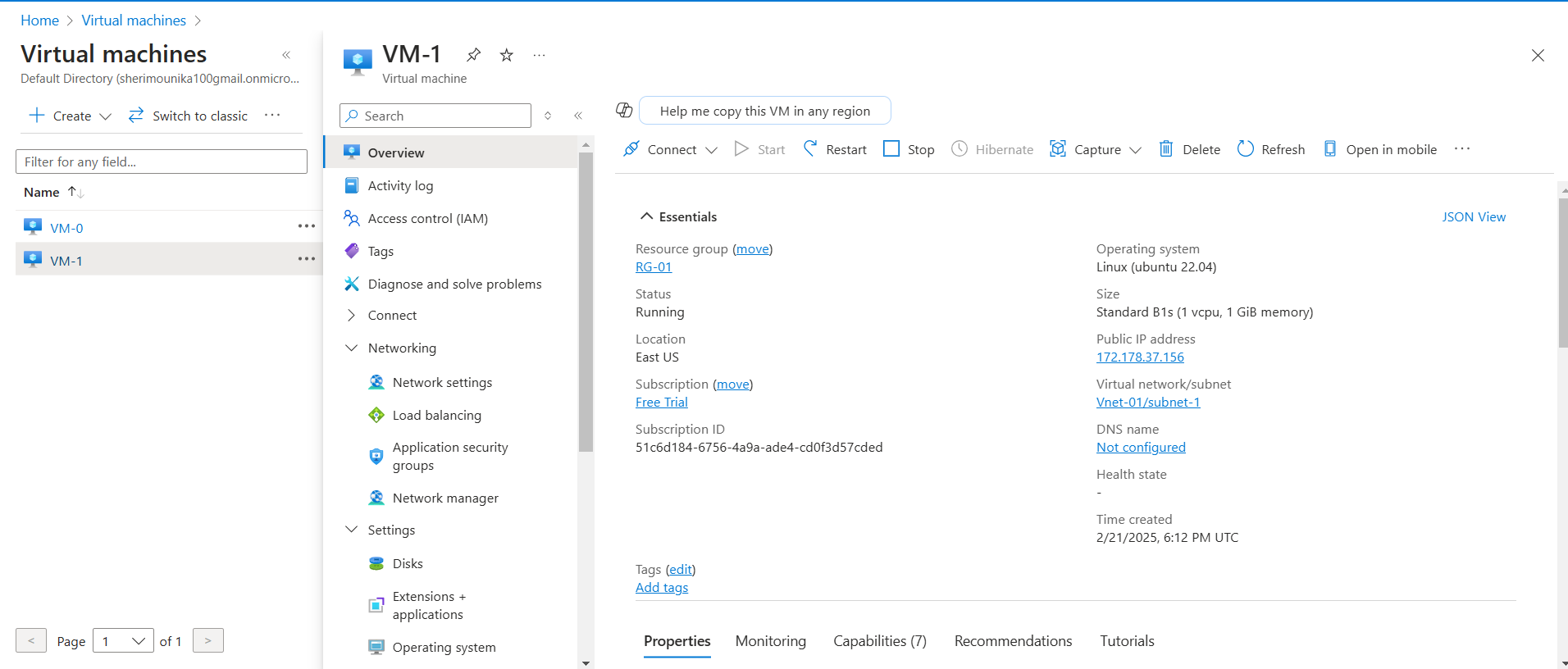
  }

}

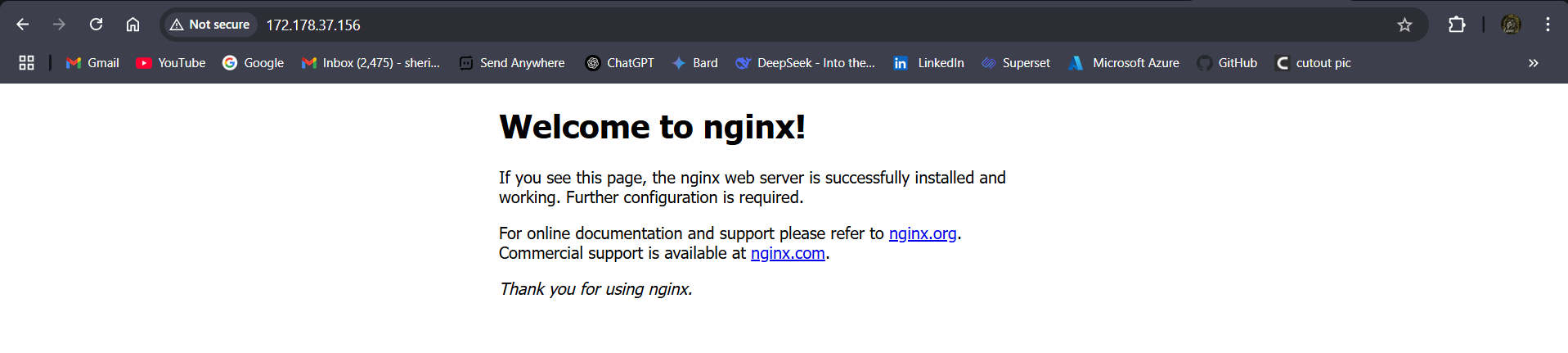
**Fig:** terraform code to create VM-1 and provisioning.



**Fig:** terraform apply for VM-1 creation.



**Fig:** virtual machine (VM-1).

****

**Fig:** web view of Nginx.

**Step12:** Creation of regional Azure load balancer with frontend IP, heath probe, load balancer rules and Backend pool with load balancer public IP.

# creation of public IP for load balancer

resource "azurerm\_public\_ip" "TFLBpip" {

  name                = "LB-pip"

  location            = azurerm\_resource\_group.rg.location

  resource\_group\_name = azurerm\_resource\_group.rg.name

  allocation\_method   = "Static"

  sku                 = "Standard" # Required for Standard Load Balancer

}

#creation of laod balancer

resource "azurerm\_lb" "TFLB" {

  name                = "LB"

  location            = azurerm\_resource\_group.rg.location

  resource\_group\_name = azurerm\_resource\_group.rg.name

  sku                 = "Standard" # Regional Load Balancer is Standard SKU

  frontend\_ip\_configuration {

    name                 = "FEPublicIPAddress"

    public\_ip\_address\_id = azurerm\_public\_ip.TFLBpip.id

  }

}

#creation of backend pool

resource "azurerm\_lb\_backend\_address\_pool" "BPLB" {

  loadbalancer\_id = azurerm\_lb.TFLB.id

  name            = "BackEndAddressPool"

}

#creation of Health probe

resource "azurerm\_lb\_probe" "LB-Health" {

  loadbalancer\_id     = azurerm\_lb.TFLB.id

  name                = "http-probe"

  port                = 80

  protocol            = "Tcp" # or "Http", or "Https"

  interval\_in\_seconds = 5

  number\_of\_probes    = 2

}

#writing of loadbalancer rules

resource "azurerm\_lb\_rule" "LB-rules" {

  loadbalancer\_id                = azurerm\_lb.TFLB.id

  name                           = "http-rule"

  protocol                       = "Tcp"

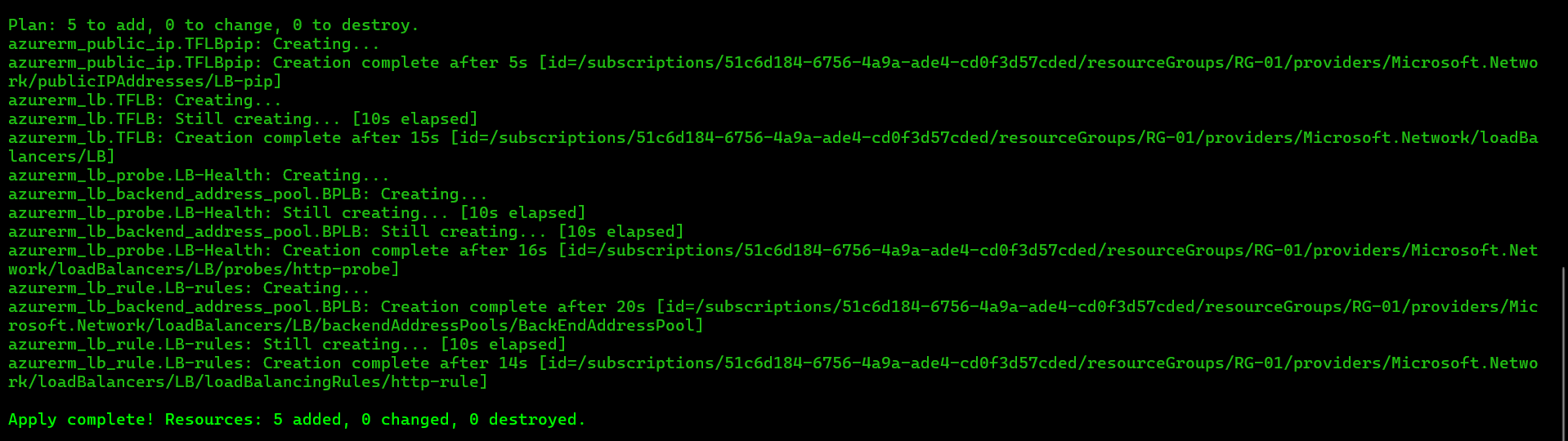
  frontend\_port                  = 80

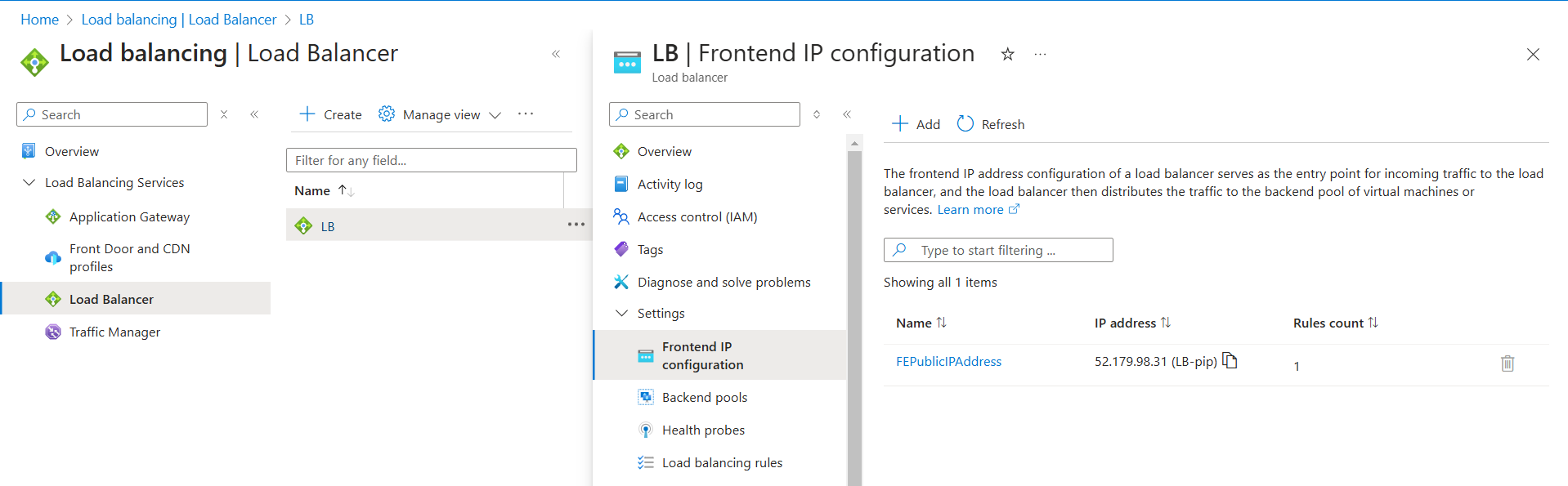
  backend\_port                   = 80

  frontend\_ip\_configuration\_name = "FEPublicIPAddress"

  probe\_id                       = azurerm\_lb\_probe.LB-Health.id

}





**Fig:** regional azure load balancer (LB).

**Step13:** Add Virtual machines (VM-0 & VM-1) to the backend pool of load balancer (LB).

#Adding of vm's NIC cards (private-ip) to the loadbalancer backend pool

resource "azurerm\_network\_interface\_backend\_address\_pool\_association" "vm1\_association" {

  network\_interface\_id    = azurerm\_network\_interface.NICards[0].id

  ip\_configuration\_name   = "internal"

  backend\_address\_pool\_id = azurerm\_lb\_backend\_address\_pool.BPLB.id

}

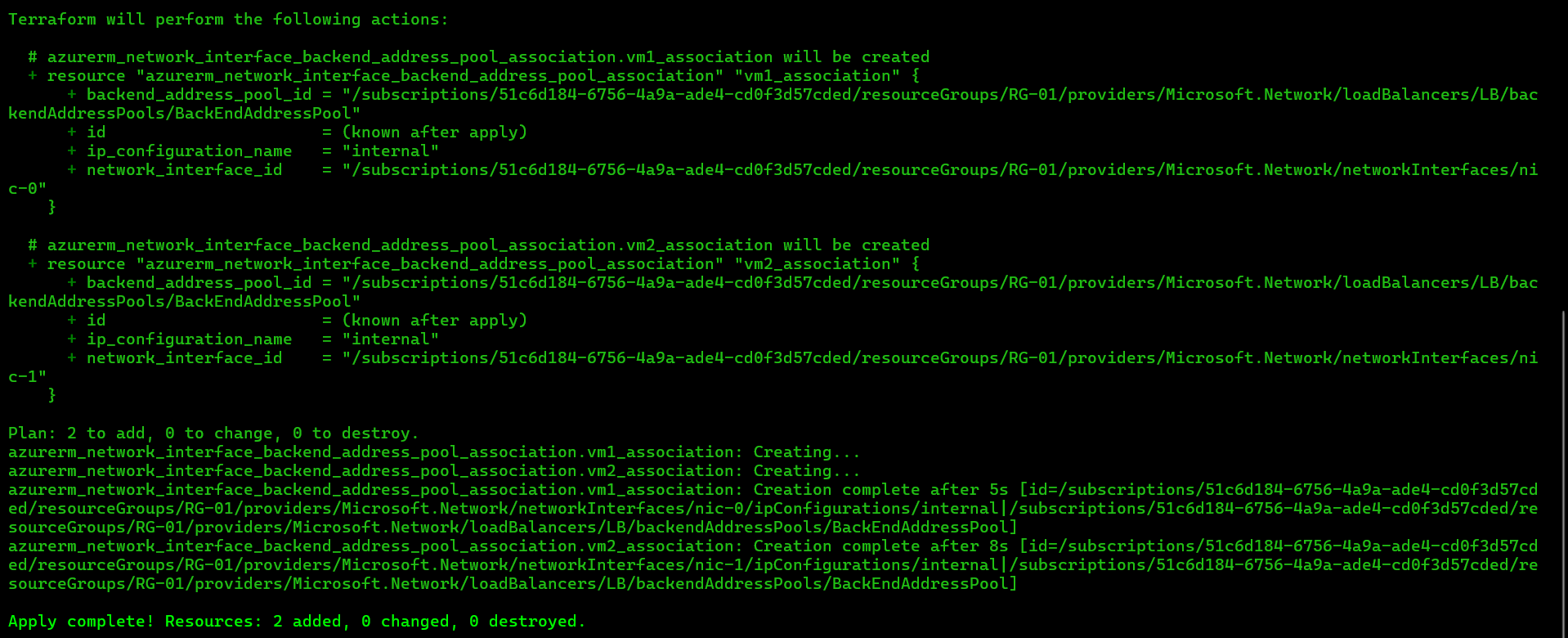
resource "azurerm\_network\_interface\_backend\_address\_pool\_association" "vm2\_association" {

  network\_interface\_id    = azurerm\_network\_interface.NICards[1].id

  ip\_configuration\_name   = "internal"

  backend\_address\_pool\_id = azurerm\_lb\_backend\_address\_pool.BPLB.id

}



**Fig:** terraform apply to add VM’s to the backend pool of load balancer.

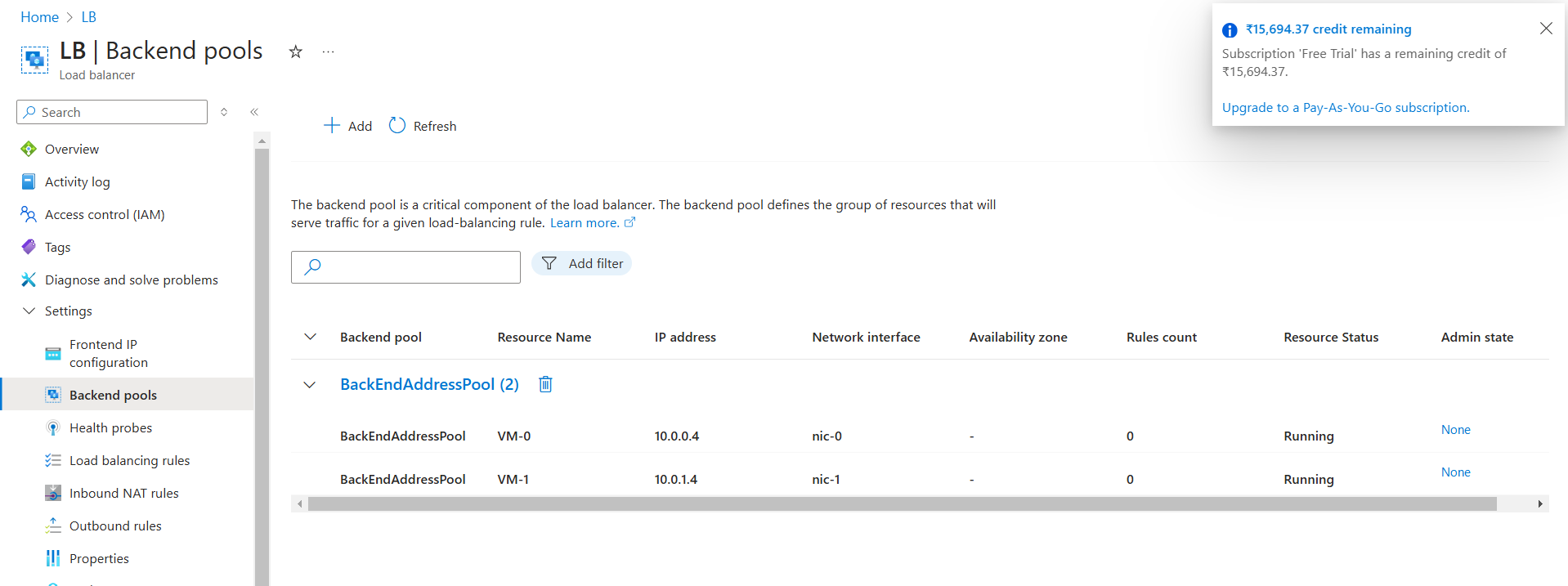
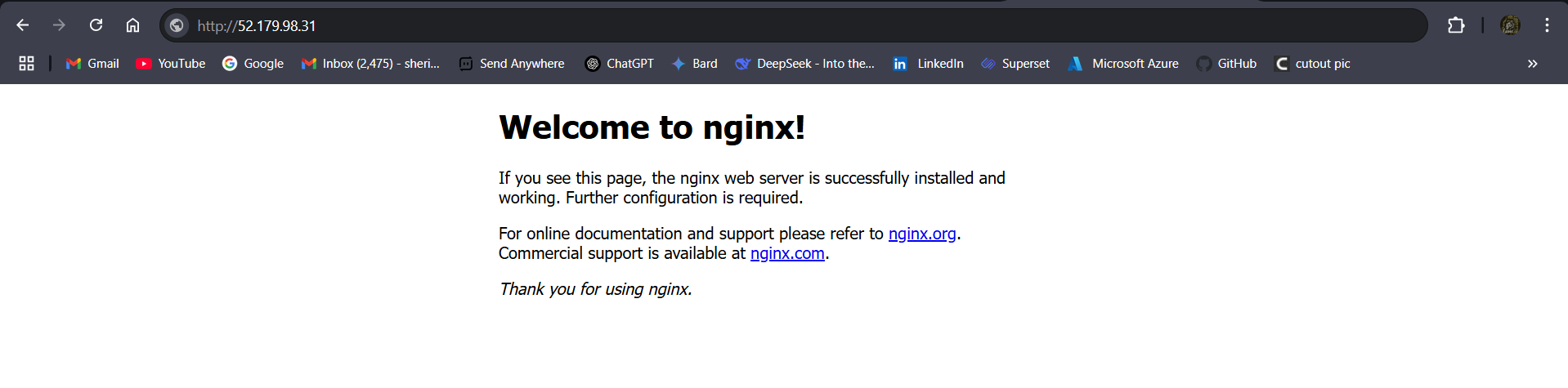


Fig: VM-0 & VM-1 are successfully added to the backend pool of load balancer (LB).

**Step14:** Now brows the in any browser using public IP of Load balancer. (52.179.98.31).



The block diagram of above performed task is given below.

LB

Vnet-01

Subnet-1

Subnet-0

VM-1

VM-0

Load Balancer

The creation of Load Balancer with two virtual machine using terraform code is completed successfully.