# Configure the Microsoft Azure Provider

provider "azurerm" {

  tenant\_id       = "415a8c7e-8647-4b46-b291-00000000000"

  subscription\_id = "c355ece9-9b1a-46ce-9c84-00000000000"

  client\_id       = "09a66f62-4a78-4dd8-b60d-00000000000"

  client\_secret   = "xlG7Q~9BM-yu1xPtrQAd\_00000000000"

  features {}

}

# Specify the version of the AzureRM Provider to use

terraform {

  required\_providers {

    azurerm = {

      source  = "hashicorp/azurerm"

      version = "=3.0.0"

    }

  }

}

# Create a resource group

resource "azurerm\_resource\_group" "test" {

   name     = "acctestrg"

   location = "EastUS2"

 }

resource "azurerm\_virtual\_network" "test" {

   name                = "acctvn"

   address\_space       = ["10.0.0.0/16"]

   location            = azurerm\_resource\_group.test.location

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

 }

 resource "azurerm\_subnet" "test" {

   name                 = "acctsub"

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

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

   address\_prefixes     = ["10.0.2.0/24"]

 }

 resource "azurerm\_public\_ip" "test" {

   name                         = "publicIPForLB"

   location                     = azurerm\_resource\_group.test.location

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

   allocation\_method            = "Static"

 }

 resource "azurerm\_lb" "test" {

   name                = "loadBalancer"

   location            = azurerm\_resource\_group.test.location

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

   frontend\_ip\_configuration {

     name                 = "publicIPAddress"

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

   }

 }

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

   loadbalancer\_id     = azurerm\_lb.test.id

   name                = "BackEndAddressPool"

 }

 resource "azurerm\_network\_interface" "test" {

   count               = 2

   name                = "acctni${count.index}"

   location            = azurerm\_resource\_group.test.location

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

   ip\_configuration {

     name                          = "testConfiguration"

     subnet\_id                     = azurerm\_subnet.test.id

     private\_ip\_address\_allocation = "Dynamic"

   }

 }

 resource "azurerm\_managed\_disk" "test" {

   count                = 2

   name                 = "datadisk\_existing\_${count.index}"

   location             = azurerm\_resource\_group.test.location

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

   storage\_account\_type = "Standard\_LRS"

   create\_option        = "Empty"

   disk\_size\_gb         = "1023"

 }

 resource "azurerm\_availability\_set" "avset" {

   name                         = "avset"

   location                     = azurerm\_resource\_group.test.location

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

   platform\_fault\_domain\_count  = 2

   platform\_update\_domain\_count = 2

   managed                      = true

 }

 resource "azurerm\_virtual\_machine" "test" {

   count                 = 2

   name                  = "acctvm${count.index}"

   location              = azurerm\_resource\_group.test.location

   availability\_set\_id   = azurerm\_availability\_set.avset.id

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

   network\_interface\_ids = [element(azurerm\_network\_interface.test.\*.id, count.index)]

   vm\_size               = "Standard\_DS1\_v2"

   # Uncomment this line to delete the OS disk automatically when deleting the VM

   # delete\_os\_disk\_on\_termination = true

   # Uncomment this line to delete the data disks automatically when deleting the VM

   # delete\_data\_disks\_on\_termination = true

   storage\_image\_reference {

     publisher = "Canonical"

     offer     = "UbuntuServer"

     sku       = "16.04-LTS"

     version   = "latest"

   }

   storage\_os\_disk {

     name              = "myosdisk${count.index}"

     caching           = "ReadWrite"

     create\_option     = "FromImage"

     managed\_disk\_type = "Standard\_LRS"

   }

   # Optional data disks

   storage\_data\_disk {

     name              = "datadisk\_new\_${count.index}"

     managed\_disk\_type = "Standard\_LRS"

     create\_option     = "Empty"

     lun               = 0

     disk\_size\_gb      = "1023"

   }

   storage\_data\_disk {

     name            = element(azurerm\_managed\_disk.test.\*.name, count.index)

     managed\_disk\_id = element(azurerm\_managed\_disk.test.\*.id, count.index)

     create\_option   = "Attach"

     lun             = 1

     disk\_size\_gb    = element(azurerm\_managed\_disk.test.\*.disk\_size\_gb, count.index)

   }

   os\_profile {

     computer\_name  = "hostname"

     admin\_username = "testadmin"

     admin\_password = "Password1234!"

   }

   os\_profile\_linux\_config {

     disable\_password\_authentication = false

   }

   tags = {

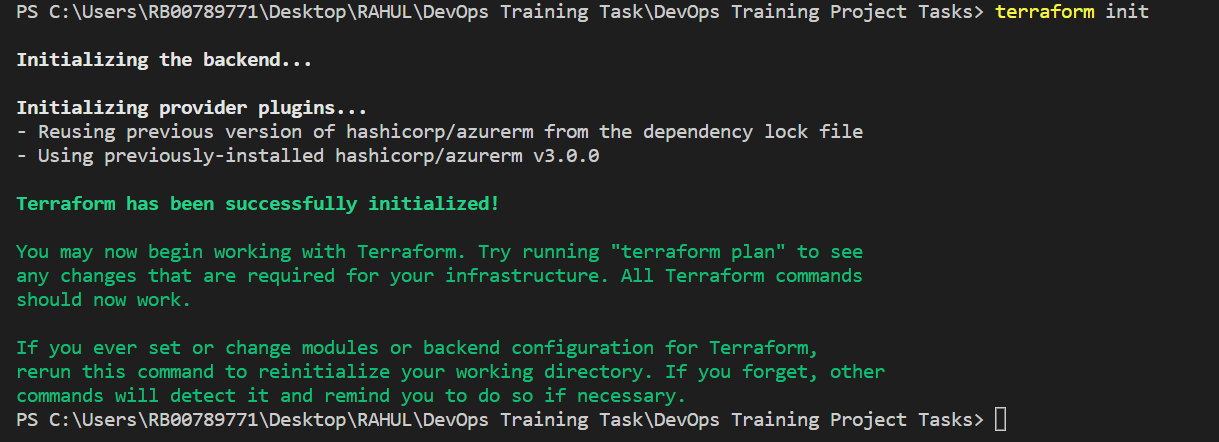
     environment = "staging"

   }

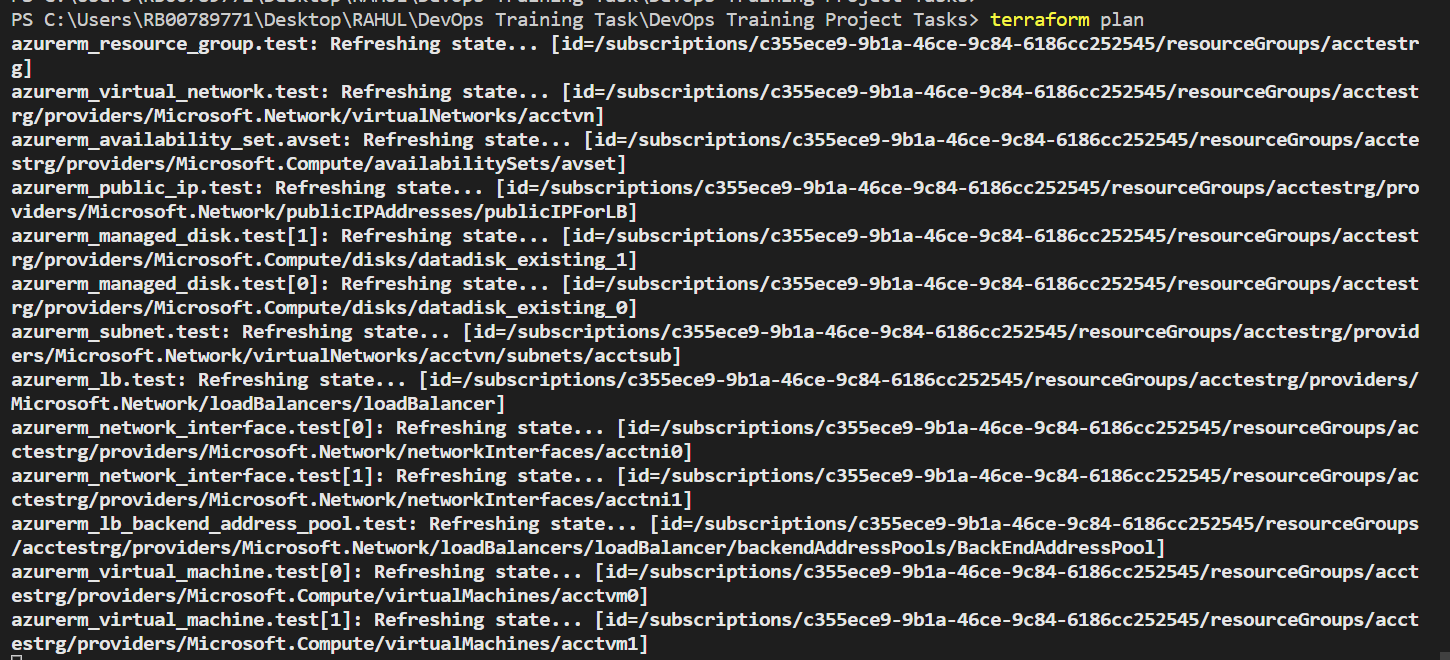
 }

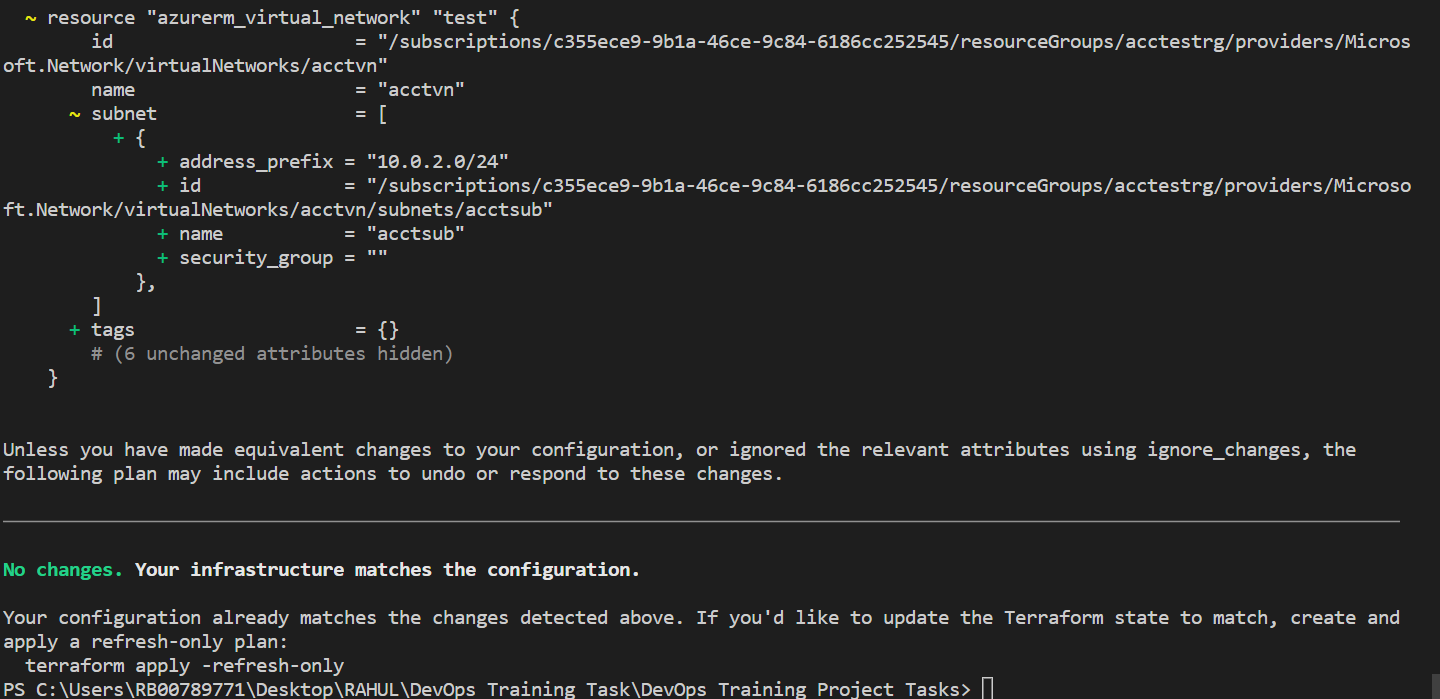
---------------------------------------------------------------------------------------------------------------------------------------------------

**Terraform Init :-**



**Terraform plan :-**





**Terraform apply :-**

