# Configure the Microsoft Azure Provider

provider "azurerm" {

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

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

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

client\_secret = "xlG7Q~9BM-yu1xPtrQAd\_\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

features {}

}

resource "azurerm\_resource\_group" "myterraformgroup" {

name = "myResourceGroup"

location = "Central US"

tags = {

environment = "Test"

}

}

# Create virtual network

resource "azurerm\_virtual\_network" "myterraformnetwork" {

name = "myVnet"

address\_space = ["10.0.0.0/16"]

location = "Central US"

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

tags = {

environment = "Test"

}

}

# Create subnet

resource "azurerm\_subnet" "myterraformsubnet" {

name = "mySubnet"

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

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

address\_prefixes = ["10.0.1.0/24"]

}

# Create public IPs

resource "azurerm\_public\_ip" "myterraformpublicip" {

name = "myPublicIP"

location = "Central US"

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

allocation\_method = "Dynamic"

tags = {

environment = "Test"

}

}

# Create Network Security Group and rule

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

name = "myNetworkSecurityGroup"

location = "Central US"

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

security\_rule {

name = "SSH"

priority = 1001

direction = "Inbound"

access = "Allow"

protocol = "Tcp"

source\_port\_range = "\*"

destination\_port\_range = "22"

source\_address\_prefix = "\*"

destination\_address\_prefix = "\*"

}

tags = {

environment = "Test"

}

}

# Create network interface

resource "azurerm\_network\_interface" "myterraformnic" {

name = "myNIC"

location = "Central US"

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

ip\_configuration {

name = "myNicConfiguration"

subnet\_id = azurerm\_subnet.myterraformsubnet.id

private\_ip\_address\_allocation = "Dynamic"

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

}

tags = {

environment = "Test"

}

}

# Connect the security group to the network interface

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

network\_interface\_id = azurerm\_network\_interface.myterraformnic.id

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

}

# Generate random text for a unique storage account name

resource "random\_id" "randomId" {

keepers = {

# Generate a new ID only when a new resource group is defined

resource\_group = azurerm\_resource\_group.myterraformgroup.name

}

byte\_length = 8

}

# Create storage account for boot diagnostics

resource "azurerm\_storage\_account" "mystorageaccount" {

name = "diag${random\_id.randomId.hex}"

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

location = "Central US"

account\_tier = "Standard"

account\_replication\_type = "LRS"

tags = {

environment = "Test"

}

}

# Create (and display) an SSH key

resource "tls\_private\_key" "example\_ssh" {

algorithm = "RSA"

rsa\_bits = 4096

}

output "tls\_private\_key" {

value = tls\_private\_key.example\_ssh.private\_key\_pem

sensitive = true

}

# Create virtual machine

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

name = "myVM"

location = "Central US"

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

network\_interface\_ids = [azurerm\_network\_interface.myterraformnic.id]

size = "Standard\_DS1\_v2"

os\_disk {

name = "myOsDisk"

caching = "ReadWrite"

storage\_account\_type = "Premium\_LRS"

}

source\_image\_reference {

publisher = "Canonical"

offer = "UbuntuServer"

sku = "18.04-LTS"

version = "latest"

}

computer\_name = "myvm"

admin\_username = "azureuser"

admin\_password = "Password1234!"

disable\_password\_authentication = false

admin\_ssh\_key {

username = "azureuser"

public\_key = tls\_private\_key.example\_ssh.public\_key\_openssh

}

provisioner "file" {

source = "example\_file.txt"

destination = "/tmp/example\_file.sh"

}

provisioner "remote-exec" {

inline = [

"sudo yum install -y jenkins java-11-openjdk-devel",

"sudo yum -y install wget",

"git clone https://github.com/devopsschool-training-notes/terraform-ey-june-2021",

"sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo",

"sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key",

"sudo yum upgrade -y",

"sudo yum install jenkins -y",

"sudo systemctl start jenkins",

"export PATH=$PATH:/usr/bin",

“sudo yum install -y maven”,

“ sudo yum install -y git “,

“sudo yum update –y”,

“sudo yum install docker –y”,

“sudo sudo chkconfig docker on”,

“sudo service docker start”,

“sudo service jenkins start”,

"sudo yum update -y",

"sudo yum install python-setuptools python-pip -y",

"sudo pip install httplib2"

]

}

provisioner "local-exec" {

command = "deploy.bat"

}

}

Create **example\_file.sh** file

#!/bin/bash

sudo yum -y update

echo "Install Java JDK 8"

sudo yum remove -y java

sudo yum install -y java-1.8.0-openjdk

echo "Install Maven"

sudo yum install -y maven

echo "Install git"

sudo yum install -y git

echo "Install Docker engine"

sudo yum update -y

sudo yum install docker -y

sudo sudo chkconfig docker on

echo "Install Jenkins"

sudo wget -O /etc/yum.repos.d/jenkins.repo http://pkg.jenkins-ci.org/redhat-stable/jenkins.repo

sudo rpm --import https://jenkins-ci.org/redhat/jenkins-ci.org.key

sudo yum install -y jenkins

sudo usermod -a -G docker jenkins

sudo chkconfig jenkins on

echo "Start Docker & Jenkins services"

sudo service docker start

sudo service jenkins start