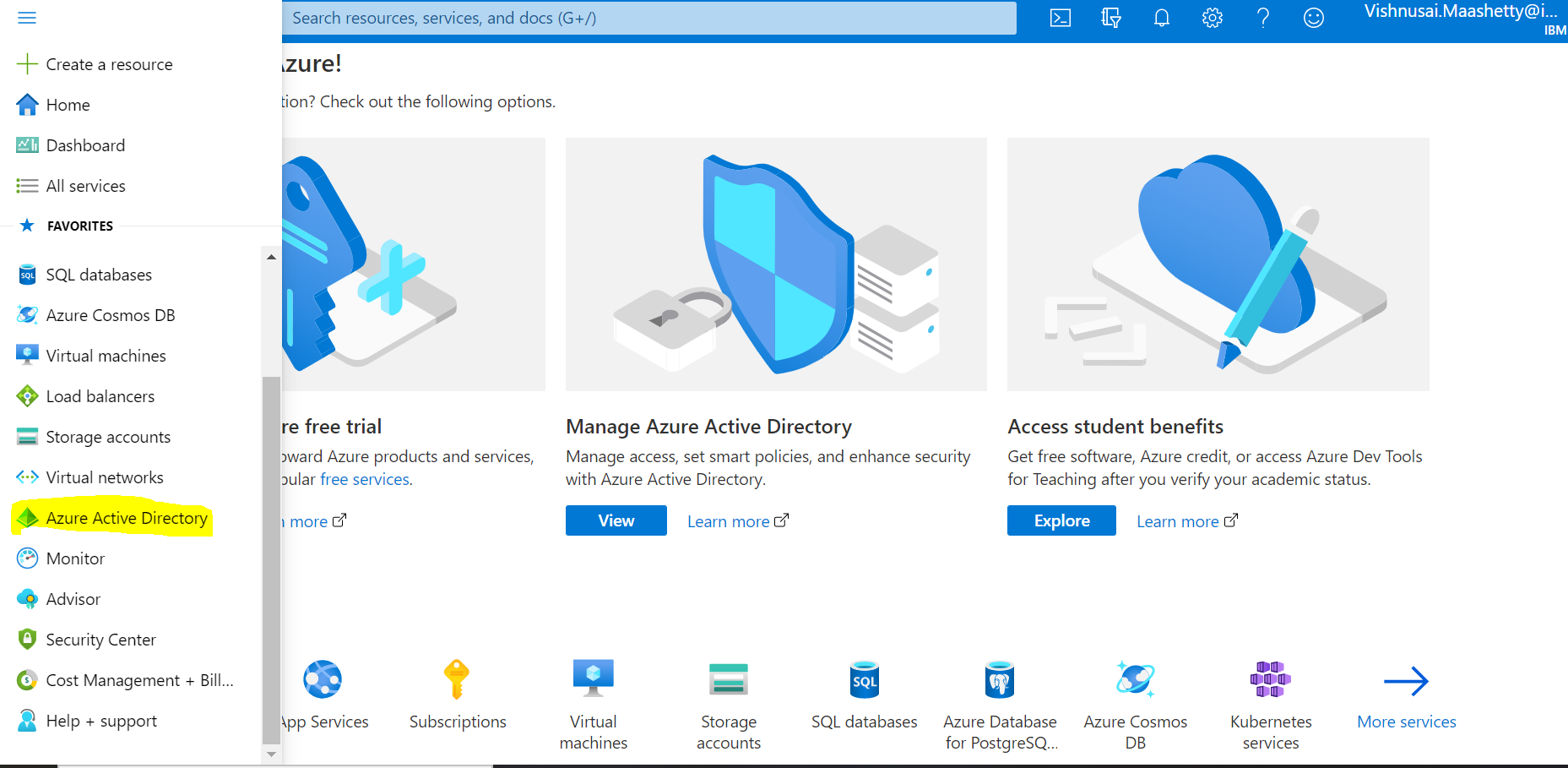
**TERRAFORM:**

To use terraform code in Azure, You have to create service principle

Terraform link: [HashiCorp Terraform - Provision & Manage any Infrastructure](https://www.hashicorp.com/products/terraform)

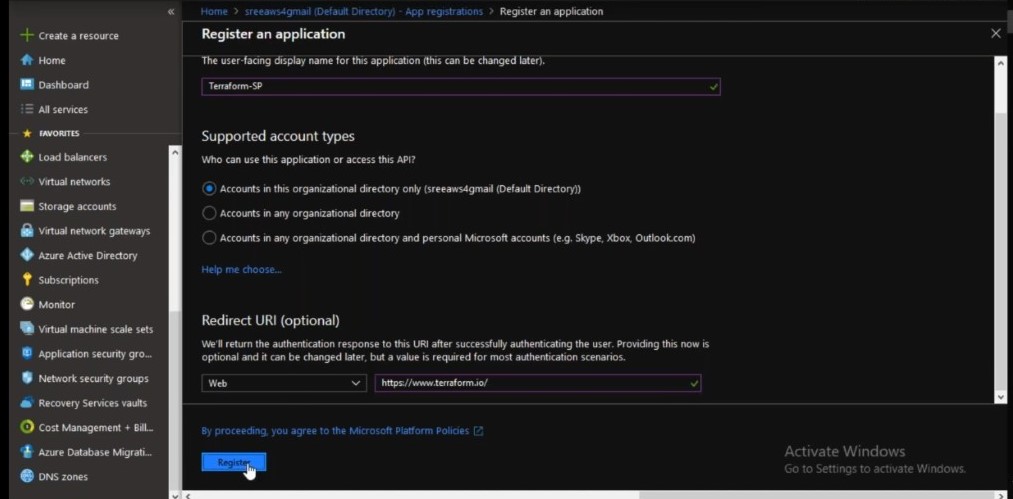
To create Service principle.

1. Go to [Microsoft Azure](https://portal.azure.com/)
2. Click on Azure Active Directory

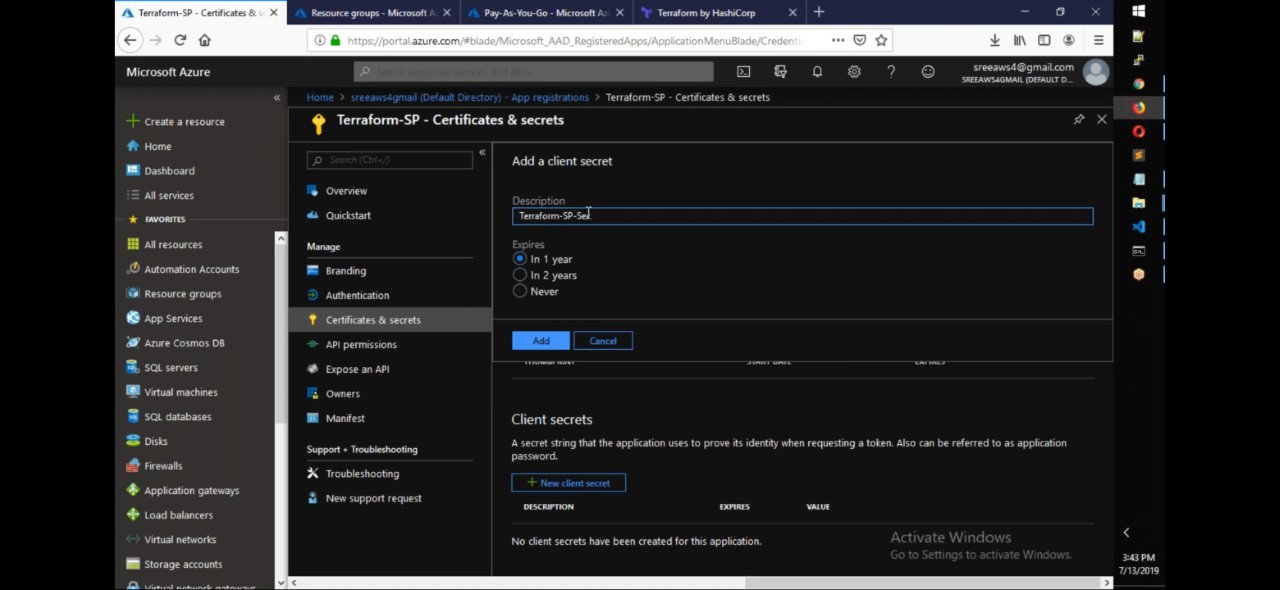


3.Click on app registrations and then select new registration

4. Give the details below and click on register.



5. Select the created Service principle and go to certificates and secrets



7. Give the details as shown in the above figure and click on the new client secret

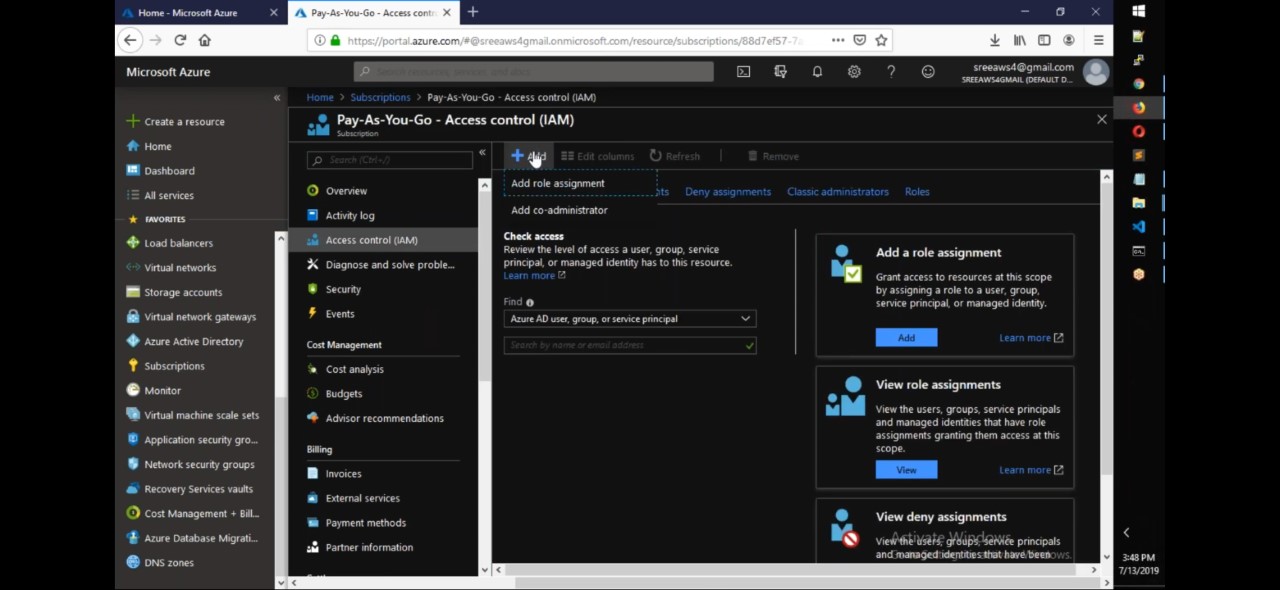
6.Save the secret key

8. Click on Add

To provide access to the terraform using service account.

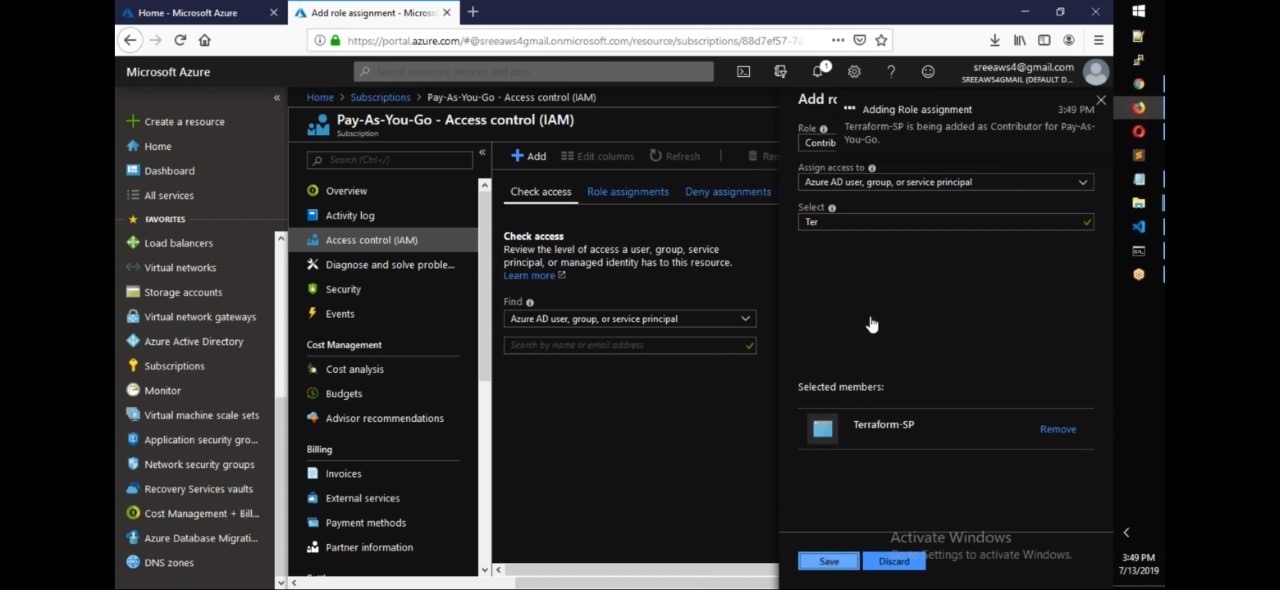
1.Go to subscription🡪Select the subscription🡪Access Control(IAM)

2. Click Add and add role assignment.



3.Give the details as shown in the below figure

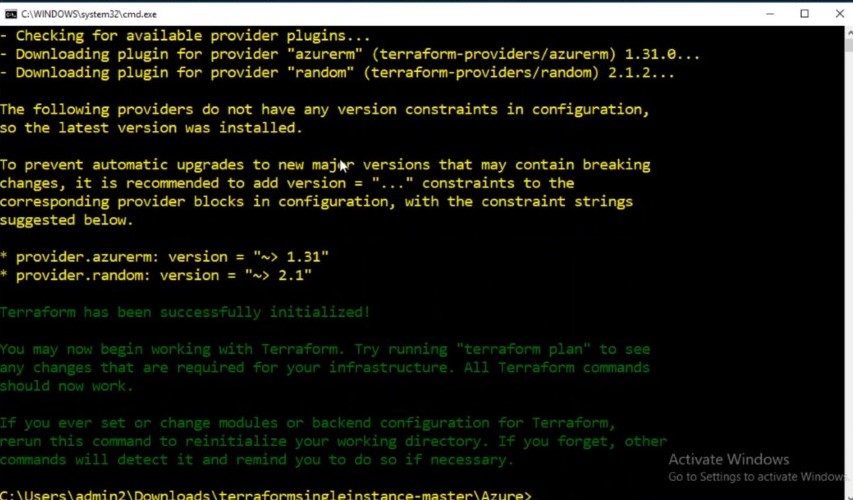
4.Click on save



**TERRAFORM COMMANDS:**

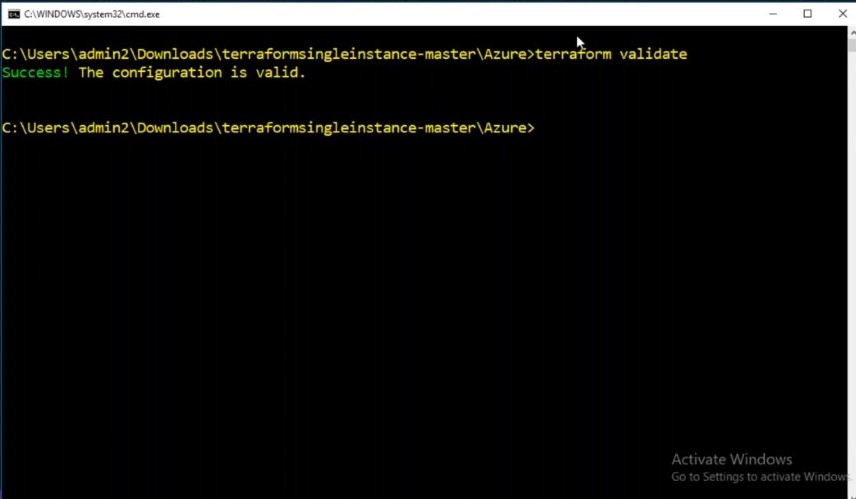
**Terrafom init:**

Downloads the plugins depending on the provider



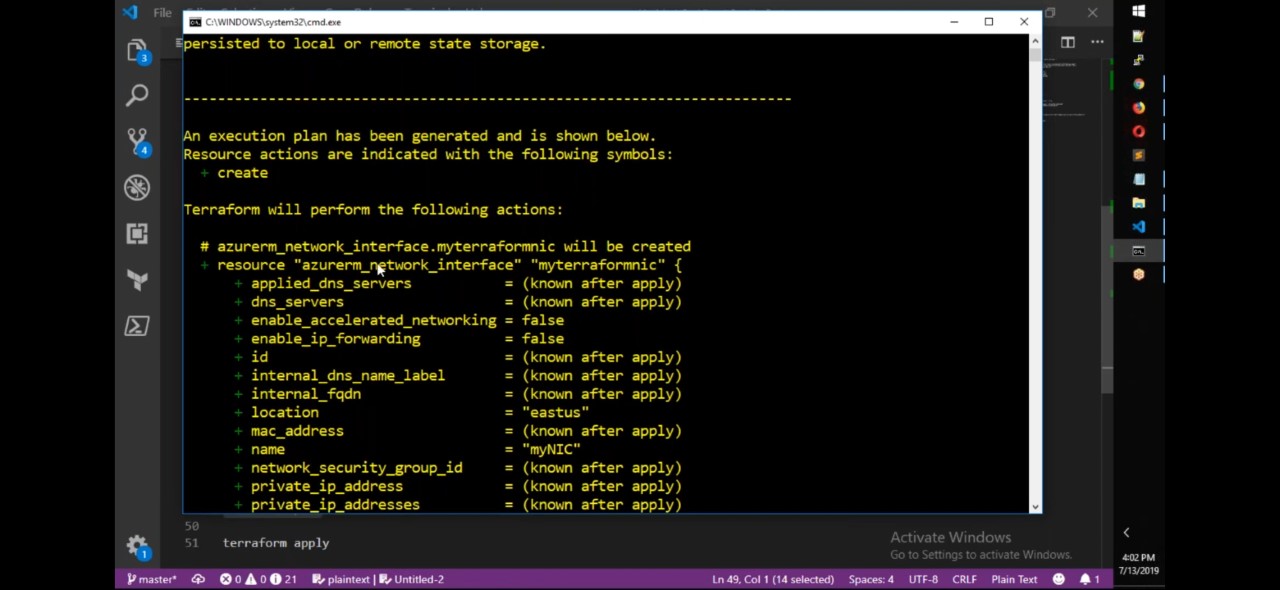
**terraform validate**

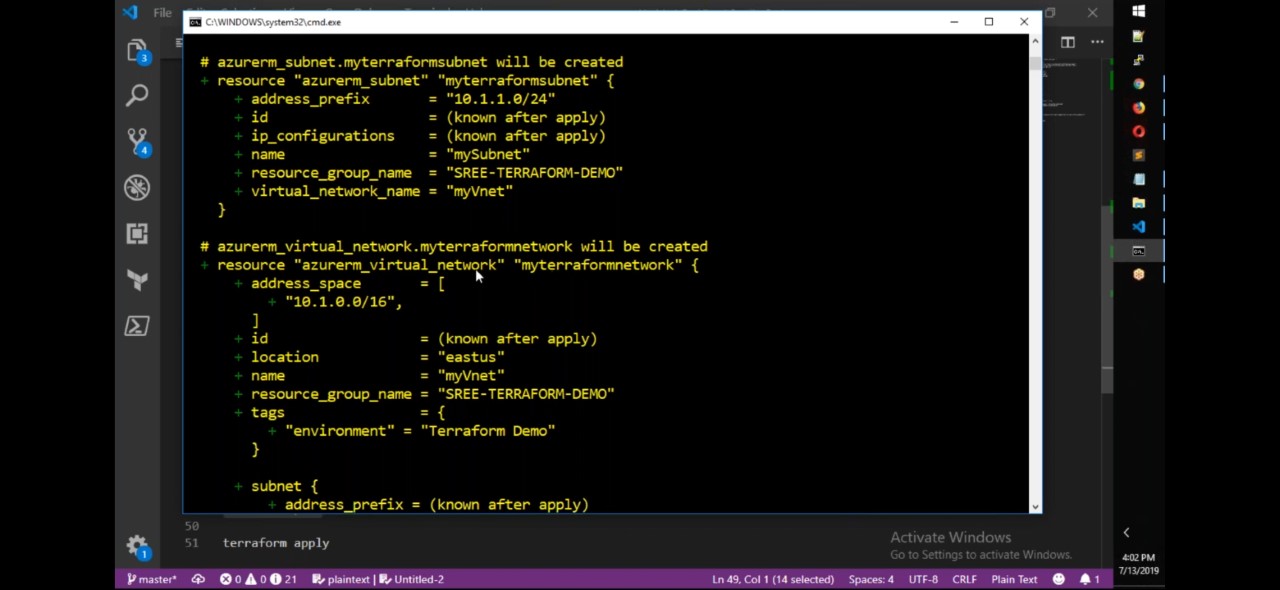
Check the terraform code

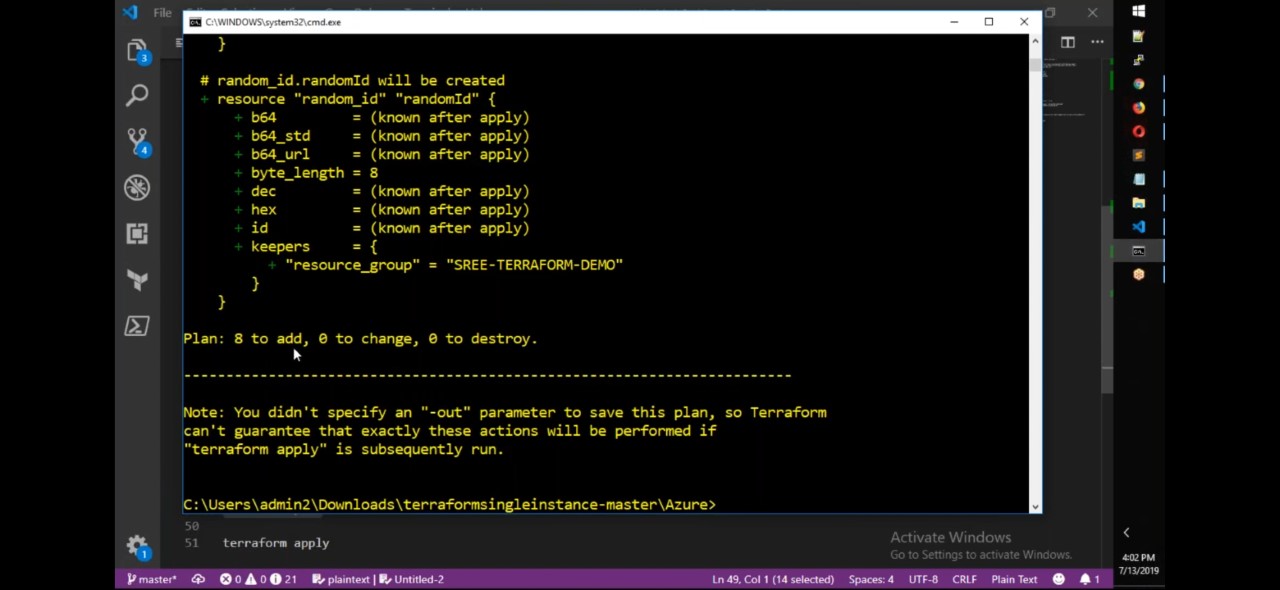


**terraform plan**

Will show the result of the dry run, what are the resources going to be created or modified







**Terraform apply:**

Executes the code and apply change in azure portal

Terraform apply –auto-approve

**Terraform destroy**

To destroy all the resources

**Terraform import**

Terraform is able to import existing infrastructure. This allows you take resources you've created by some other means and bring it under Terraform management.

provider "azurerm" {

version="1.38.0"

}

# create resource group

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

name = "rg-terraform"

location = "eastus"

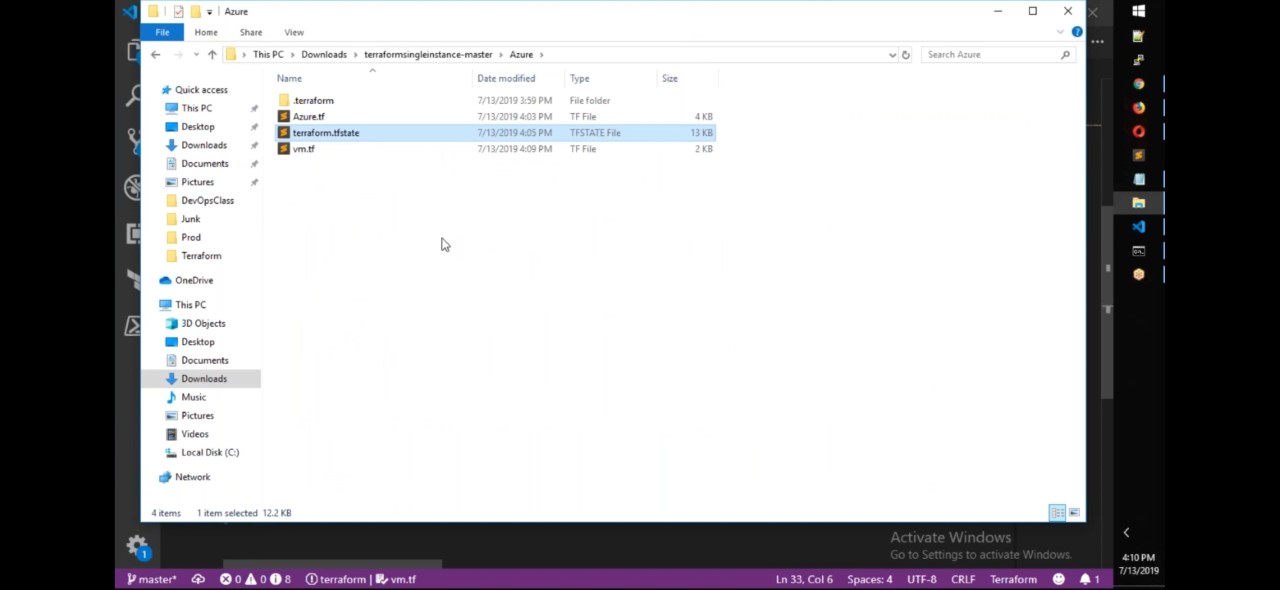
}

terraform import <Terraform Resource Name>.<Resource Label> <Azure Resource ID>

**Terraform State file:**

Terraform store state about your managed infrastructure and configuration. This state file is used by terraform to map real world resources to your configuration and keep track of metadata and to improve the performance.

This state is stored by default in a local file named terraform.tfstate but also it can be stored remotely



**terraform show**

command is used to provide human-readable output from a state or plan file.

**Terraform state list**

command is used to list resources within a state file

$ terraform state list

aws\_instance.foo

aws\_instance.bar[0]

aws\_instance.bar[1]

module.elb.aws\_elb.main

**Terraform state show**

command is used to show the attributes of a single resource in the [Terraform state](https://www.terraform.io/docs/language/state/index.html).

$ terraform state show 'packet\_device.worker'

# packet\_device.worker:

resource "packet\_device" "worker" {

billing\_cycle = "hourly"

created = "2015-12-17T00:06:56Z"

facility = "ewr1"

hostname = "prod-xyz01"

id = "6015bg2b-b8c4-4925-aad2-f0671d5d3b13"

locked = false

}

**Terraform Code:**

# Configure the Microsoft Azure Provider

provider "azurerm" {

subscription\_id =””(subscription id)

Client\_id=””(available in app registration overview)

Client\_secret=””(key generated after creating app registration)

Tenant\_id=””(available in app registration overview)

}

# Create a resource group

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

name = "myResourceGroup"

location = "eastus"

tags = {

environment = "Terraform Demo"

}

}

# Create virtual network

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

name = "myVnet"

address\_space = ["10.0.0.0/16"]

location = "eastus"

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

tags = {

environment = "Terraform Demo"

}

}

# 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 = "eastus"

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

allocation\_method = "Dynamic"

tags = {

environment = "Terraform Demo"

}

}

# Create Network Security Group and rule

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

name = "myNetworkSecurityGroup"

location = "eastus"

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 = "Terraform Demo"

}

}

# Create network interface

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

name = "myNIC"

location = "eastus"

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 = "Terraform Demo"

}

}

# 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 = "eastus"

account\_tier = "Standard"

account\_replication\_type = "LRS"

tags = {

environment = "Terraform Demo"

}

}

# 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 }

# Create virtual machine

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

name = "myVM"

location = "eastus"

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"

disable\_password\_authentication = true

admin\_ssh\_key {

username = "azureuser"

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

}

boot\_diagnostics {

storage\_account\_uri = azurerm\_storage\_account.mystorageaccount.primary\_blob\_endpoint

}

tags = {

environment = "Terraform Demo"

}

}