1.1. Tools and Software’s

* Install Azure CLI
* Install Visual Studio Code
* Terraform Extension
* Terraform Installation and Configuration

1.2. Terraform Workflow

* Learn the terraform [workflow](https://www.ciraltos.com/terraform-workflow-with-azure-write-plan-and-apply/).
* Validate if terraform init command is downloading all the providers

1.3. Terraform Configuration Files

* Learn how to [validate](https://www.terraform.io/cli/commands/validate) the Terraform Configuration file Content in Visual Studio Code
* Learn how to validate the Terraform Configuration file using Terraform Validate and ADO Pipeline
* Learn how to [Preview](https://www.terraform.io/cli/commands/plan) the changes before submitting to Azure
* How to create [Resource Group](https://learn.microsoft.com/en-us/azure/developer/terraform/create-resource-group?tabs=azure-cli)

1.4. Create the below Terraform Modules

* 0.Resourcegroup
* 1.VirtualNetwork
* 2.IPAddress
* 3.NSG
* 4.NIC
* 5.DiagnosticsService
* 6.VirtualMachine

1.4.1. Create a module named **0. Resourcegroup**

**Please see the below attached example for creation of resource group and storage account**



Module folder, specify

1.4.1.1. Configuration

This module should create the Resource Group

1.4.1.2. Inputs – This module should take the following inputs

1. Location = "westeurope"

2. ResourceGroupName = "myTFResourceGroup"

1.4.1.3. Outputs – This module should return the following Outputs

1. None

1.4.2. Create a module named **1. VirtualNetwork**

1.4.2.1. Configuration

This module should create the VirtualNetwork and Subnet

1.4.2.2. Inputs – This module should take the following inputs

1. ResourceGroupName="myTFResourceGroup"

2. Location="eastus"

3. VNet = "tf\_vnet"

4. SubNet="tf\_subnet"

5. AdressFix=””

6.SubnetPrefix=””

1.4.2.2.1. Outputs – This module should return the following Outputs

* Return Subnet Id as Output. Below is an example

output "SubNetId" {

value = azurerm\_subnet.subnet.id

}

1.5. Create a module named **2.IPAddress**

1.5.1. Configuration

This module should create the IP Address

1.5.1.1. Inputs – This module should take the following inputs

ResourceGroupName="myTFResourceGroup"

Location="eastus"

IpAddressName = "myPublicIP"

1.5.1.2. Outputs – This module should return the following Outputs

Return the resource ID of the Public IP Address

output "IPAddressId" {

value = azurerm\_public\_ip.myterraformpublicip.id

}

1.6. Create a module named **3.NSG**

1.6.1. Configuration

This module should create the Network Security Group

1.6.1.1. Inputs – This module should take the following inputs

ResourceGroupName="myTFResourceGroup"

Location="eastus"

NSGName="myNetworkSecurityGroup"

1.6.1.2. Outputs – This module should return the following Outputs

Return the Resource Id of the NSG created

output "NSGId" {

value = azurerm\_network\_security\_group.myterraformnsg.id

}

1.7. Create a module named **4.NIC**

1.7.1. Configuration

This module should create the Network Interface Card

1.7.1.1. Inputs – This module should take the following inputs

ResourceGroupName="myTFResourceGroup"

Location="eastus"

NICName="myNIC"

1.7.1.2. Outputs – This module should return the following Outputs

Return the Resource Id of the NIC created

output "NICId" {

value = azurerm\_network\_interface.myterraformnic.id

}

1.8. Create a module named **5.DiagnosticsService**

1.8.1. Configuration

This module should create the Storage Account for Storing the Diagnostics Information.

1.8.1.1. Inputs – This module should take the following inputs

ResourceGroupName="myTFResourceGroup"

Location="eastus"

1.8.1.2. Outputs – This module should return the following Outputs

Return the primary end point of the Blob Storage created

output "StorageURI" {

value = azurerm\_storage\_account.mystorageaccount.primary\_blob\_endpoint

}

1.9. Create a module named **6.VirtualMachine**

1.9.0.3. Configuration

This module should create the Virtual Machine

1.9.0.4. Inputs – This module should take the following inputs

ResourceGroupName="myTFResourceGroup"

Location="eastus"

Vnet=”VnetName”

Subnet=”SubnetName”

1.9.0.5. Outputs – This module should return the following Outputs

None

1.9.0.5. Outputs – This module should return the following Outputs

None

Create the above in the root folder as well.

1.9.1. Root Module

1.9.1.1. Configuration

Invoke ALL the modules.

1.9.1.2. Inputs

Location = "eastus"

ResourceGroupName="myTFResourceGroup"

VNet = "tf\_vnet"

SubNet="tf\_subnet"

IpAddressName = "myPublicIP"

NSGName="myNetworkSecurityGroup"

NICName="myNIC"

VirtualMachineName="vm-Terraform"

1.9.2. Configuration of backend for Remote State

* Create a new Storage Account for storing the State
* Create a container named terraformstate
* Use the Access Keys and other configurations of this storage account in the backend configuration of the Terraform file in the root module. Below is an example

terraform {

backend "azurerm" {

resource\_group\_name = "StorageAccount-ResourceGroup"

storage\_account\_name = "abcd1234"

container\_name = "tfstate"

key = "prod.terraform.tfstate"

}

1.10. Key Vault – Integrate Key-Vault with IAC Templates

* Remove the hard-coded secret in the 6.Virtualmachine module
* Create the secret in the Key-Vault
* Create a Service Principle
* Provide appropriate permissions to the Service Priciple on Key-Vault using Access Control (IaM)
* Execute the template