**Internship program**

**Celebal technologies**

**Project🡪 Configure FTP server and apache server on virtual machine using terraform**

**Team name🡪 Cloud 4**

**Team members**

|  |  |
| --- | --- |
| NAME | Email |
| Ekta Jarwal | 2018cscloudekta6855@poornima.edu.in |
| Gopal Mali | 2018cscloudgopal6699@poornima.edu.in |
| Harneet Singh | 2018cscloudharneet6670@poornima.edu.in |
| Pallavi Jeengar | 2018cscloudpallavi6620@poornima.edu.in |

Project name

**Configure FTP server and apache server on virtual machine using terraform**

Terraform is an open-source infrastructure as the source software to that enables us to safely and predictably change and improve infrastructure.

Terraform can manage existing and popular service providers as well as custom in-house solutions.

Terraform generate and execution plan describing what it will do to reach the desired state and then execute it to build the described infrastructure.

As the configuration changes terraform is able to determine what changed and create incremental execution plans which can be applied.

Both low level components such as compute, instances, storage, and networking, as well as high level components purchase DNS entries, SaaS features, etc.

**# Setting up terraform**

1. Install terraform

We need to install terraform in a system from hashicorp official site as a binary package.

* Go to terraform official site
* Go to download Tab
* Click on download for Windows 64 bit
* Zip file of terraform will be downloaded

1. Unzip terraform file

To use terraform we need to unzip the folder we downloaded

for this we follow these steps

* Right click on the folder
* Click on unzip and select the Destination path
* Set the environment path of this destination folder in environmental variables for system
* Terraform is now ready to use

1. Verify the installation

To verify the installation reason are going to run following commands in the terminal/cmd of our window

**$ terraform**

Or

**$ terraform help**

Or

**$ terraform version**

**# Setting up Azure and Azure cli**

1. Create an Azure account using your email ID in subscription for student or free account.
2. Install the Azure cli tool

* Open your PowerShell prompt as an administrator and run the following command

**Invoke-WebRequest -Uri https://aka.ms/installazurecliwindows -OutFile .\AzureCLI.msi; Start-Process msiexec.exe -Wait -ArgumentList '/I AzureCLI.msi /quiet'; rm .\AzureCLI.msi**

1. Authenticate using Azure CLI

* In your terminal, use the Azure CLI tool to setup your account permissions locally by this command.

**$ az login**

Your browser window will open and you will be prompted to enter your login credentials

After successful authentication your terminal will display your subscription information

**# Write configuration**

Create folder called **azureTask**

1. **Providers**

The provided block configures specified provider like azurerm

A provider is a plugin that terraform uses to create and manage are resources

We can Define Multiple provider block in a terraform configuration to manage resources from different providers

File name🡪provider.tf

**provider “azurerm ”{**

**features{}**

**}**

1. **Credentials**

File name🡪c.tvars

|  |
| --- |
| **computername = "avi"** |
| **adminuser = "avi"** |  |
| **adminpasswd = "Vrockpokey@11"** |  |

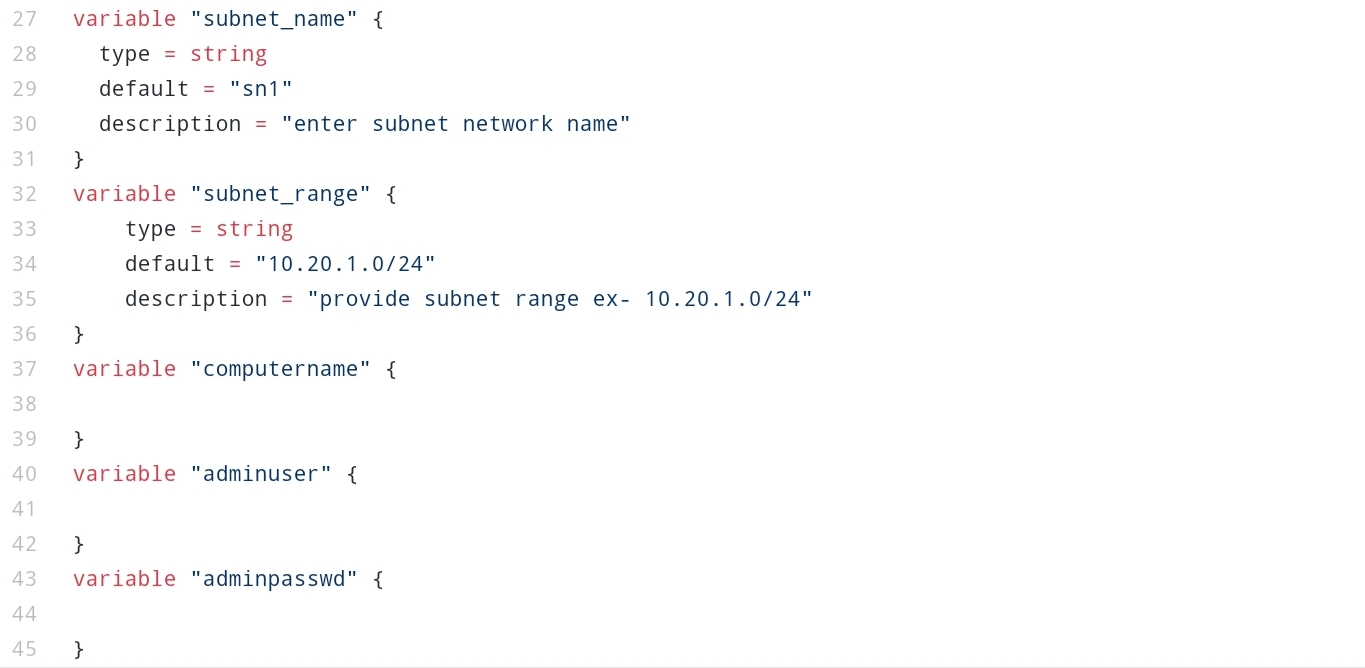
1. **Variables**

Variables servers as parameters for terraform module, allowing aspects of the module to be customised without altering the module’s own source code, and allowing modules to be shared between two different configurations.

File name🡪variable.tf

|  |
| --- |
| **variable "rg1" {** |
|  | **type = string** |
|  | **default = "avi"** |
|  | **description = "enter resource group name"** |
|  | **}** |
|  |  |
|  | **variable "location" {** |
|  | **type = string** |
|  | **default = "East Asia"** |
|  | **description = "enter region from list Southeast Asia,East US, East Asia, South India, West India"** |
|  | **}** |
|  | **variable "vm\_name" {** |
|  | **type = string** |
|  | **default = "vm1"** |
|  | **description = "enter virtual machine name"** |
|  | **}** |
|  | **variable "vn\_name" {** |
|  | **type = string** |
|  | **default = "vn1"** |
|  | **description = "enter virtual network name"** |
|  | **}** |
|  | **variable "range" {** |
|  | **type = string** |
|  | **default = "10.20.0.0/16"** |
|  | **description = "provide cidr range ex- 10.20.0.0/16"** |
|  | **}** |
|  | **variable "subnet\_name" {** |
|  | **type = string** |
|  | **default = "sn1"** |
|  | **description = "enter subnet network name"** |
|  | **}** |
|  | **variable "subnet\_range" {** |
|  | **type = string** |
|  | **default = "10.20.1.0/24"** |
|  | **description = "provide subnet range ex- 10.20.1.0/24"** |
|  | **}** |
|  | **variable "computername" {** |
|  |  |
|  | **}** |
|  | **variable "adminuser" {** |
|  |  |
|  | **}** |
|  | **variable "adminpasswd" {** |
|  |  |
|  | **}** |





1. **terraform.tfstate**

Terraform tfstate is used by terraform to map real-world resources to our configuration, keep track of metadata, and to improve performance for large infrastructure.

This is state is stored by default in a local file name terraform.tfstate.

This file can also be stored remotely, which works better in a team environment.

1. **terraform.tfstate.backup**

|  |
| --- |
| **{** |
|  | **"version": 4,** |
|  | **"terraform\_version": "0.15.3",** |
|  | **"serial": 506,** |
|  | **"lineage": "8087627f-287f-d42e-1ab4-138eaed1a78a",** |
|  | **"outputs": {},** |
|  | **"resources": []** |
|  | **}** |

By default a backup of our state file is written to terraform.tfstate.backup in case the state file is lost or corrupted to simplify recovery.

The state file is used by terraform to keep track of resources and metadata information about our infrastructure.

1. **main.tf**

File name🡪main.tf

* Resource group

|  |
| --- |
| **resource "azurerm\_resource\_group" "rg1" {** |
| **name = "${var.rg1}"** |  |
| **location = "${var.location}"** |  |
| **}** |  |

* Virtual machine

|  |
| --- |
|  |
| resource "azurerm\_ \_machine" "os1"{ |
| resource\_group\_name = azurerm\_resource\_group.rg1.name |  |
| name = "${var.vm\_name}" |  |
| location = azurerm\_resource\_group.rg1.location |  |
| network\_interface\_ids = [azurerm\_network\_interface.net-interface.id] |  |
| os\_profile\_linux\_config { |  |
| disable\_password\_authentication = false |  |
| } |  |
| storage\_image\_reference { |  |
| publisher = "RedHat" |  |
| offer = "RHEL" |  |
| sku = "8.1" |  |
| version = "latest" |  |
| } |  |
| vm\_size = "Standard\_DS1\_v2" |  |
| os\_profile { |  |
| computer\_name = var.computername |  |
| admin\_username = var.adminuser |  |
| admin\_password = var.adminpasswd |  |
| } |  |
| storage\_os\_disk { |  |
| name = "d1" |  |
| caching = "ReadWrite" |  |
| create\_option = "FromImage" |  |
| managed\_disk\_type = "Standard\_LRS" |  |
| disk\_size\_gb = "70" |  |
| } |  |
| tags = { |  |
| "Env" = "conf" |  |
| } |  |

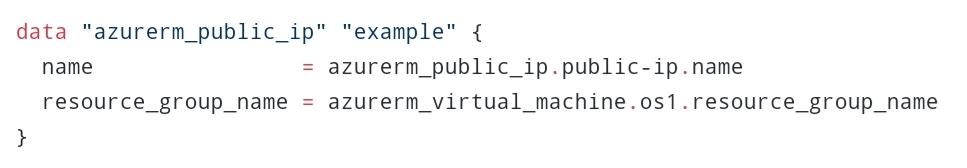
* Virtual network

|  |
| --- |
| **resource "azurerm\_virtual\_network" "vn" {** |

|  |  |
| --- | --- |
|  | **name = "${var.vn\_name}"** |
|  | **address\_space = ["${var.range}"]** |
|  | **location = azurerm\_resource\_group.rg1.location** |
|  | **resource\_group\_name = azurerm\_resource\_group.rg1.name** |
|  | **}** |

* Subnet

|  |
| --- |
| **resource "azurerm\_subnet" "sn1" {** |
| |  | | --- | | **name = "${var.subnet\_name}"** | | **resource\_group\_name = azurerm\_resource\_group.rg1.name** | | **virtual\_network\_name = azurerm\_virtual\_network.vn.name** | | **address\_prefixes = ["${var.subnet\_range}"]** | | **}** | |  |
|  |  |
|  |  |
|  |  |
|  |  |

* Public ip

|  |
| --- |
| **resource "azurerm\_public\_ip" "public-ip" {** |
| |  | | --- | | **name = "public-ip"** | | **location = azurerm\_resource\_group.rg1.location** | | **resource\_group\_name = azurerm\_resource\_group.rg1.name** | | **allocation\_method = "Dynamic"** | | **tags = {** | | **use = "auth"** | | **}** | | **}** | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

* Network interface

|  |
| --- |
| **resource "azurerm\_network\_interface" "net-interface" {** |
| |  | | --- | | **name = "nic-1"** | | **resource\_group\_name = azurerm\_resource\_group.rg1.name** | | **location = azurerm\_resource\_group.rg1.location** | | **ip\_configuration {** | | **name = "nic-1"** | | **subnet\_id = azurerm\_subnet.sn1.id** | | **private\_ip\_address\_allocation = "Dynamic"** | | **private\_ip\_address = "10.0.2.5"** | | **public\_ip\_address\_id = azurerm\_public\_ip.public-ip.id** | | **}** | | **}** | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

* Security groups

|  |
| --- |
| **resource "azurerm\_network\_interface\_security\_group\_association" "ansg" {** |
| |  | | --- | | **network\_interface\_id = azurerm\_network\_interface.net-interface.id** | | **network\_security\_group\_id = azurerm\_network\_security\_group.nsg.id** | | **}** | |  | | **resource "azurerm\_subnet\_network\_security\_group\_association" "asnsga" {** | | **subnet\_id = azurerm\_subnet.sn1.id** | | **network\_security\_group\_id = azurerm\_network\_security\_group.nsg.id** | | **}** | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

* Inbound and outbound rules

|  |
| --- |
| resource "azurerm\_network\_security\_group" "nsg" { |
| |  | | --- | | name = "nsg-1" | | location = azurerm\_resource\_group.rg1.location | | resource\_group\_name = azurerm\_resource\_group.rg1.name | | security\_rule { | | name = "req" | | priority = 100 | | direction = "Inbound" | | access = "Allow" | | protocol = "Tcp" | | destination\_port\_range = [“80”,”21",”22",”49152-50000] | | source\_address\_prefix = "\*" | | destination\_address\_prefix = "\*" | | source\_port\_range = "\*" | | } | | } | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

1. **Configuring server**

* Index.html file

<html>

<head>

<title>welcome to terraform and azure project</title>

</head>

<body>

<h1>welcome to apache and ftp server configuration on virtual machine using terraform

</body>

</html>

* Vsftpd.conf file
* Copying web pages and config file from our desktop to Azure

|  |
| --- |
| resource "null\_resource" "copy\_config\_file" { |
| |  | | --- | | **provisioner "file" {** | | **source = "C:/Users/user/Desktop/vsftpd.conf"** | | **destination = "/home/avi/vsftpd.conf"** | | **}** | | **connection {** | | **type = "ssh"** | | **user = "avi"** | | **password = "Vrockpokey@11"** | | **host = data.azurerm\_public\_ip.example.ip\_address** | | **}** | | **}** | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |
| --- |
| resource "null\_resource" "copy\_web\_pages" { |
| |  | | --- | | provisioner "file" { | | source = "C:/Users/user/Desktop/index.html" | | destination = "/home/avi/index.html" | | } | | connection { | | type = "ssh" | | user = "avi" | | password = "Vrockpokey@11" | | host = data.azurerm\_public\_ip.example.ip\_address | | } | | } | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

* Configuring apache server

|  |
| --- |
| resource "null\_resource" "conf\_http\_server" { |
| |  | | --- | | provisioner "remote-exec" { | | inline = [ | | "sudo yum install httpd -y", | | "sudo echo hello world > /var/www/html/index.html", | | "sudo systemctl enable httpd --now", | | "sudo firewall-cmd --zone=public --add-port=80/tcp --permanent", | | "sudo firewall-cmd --reload", | | "sudo cp /home/avi/index.html /var/www/html"] | | } | | connection { | | type = "ssh" | | user = "avi" | | password = "Vrockpokey@11" | | host = data.azurerm\_public\_ip.example.ip\_address | | } | | } | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

* Configuring ftp server

|  |
| --- |
| **resource "null\_resource" "conf\_ftp\_server" {** |
| |  | | --- | | **provisioner "remote-exec" {** | | **inline = [** | | **"sudo dnf install vsftpd -y",** | | **"sudo systemctl start vsftpd",** | | **"sudo systemctl enable vsftpd --now",** | | **"sudo yum install ftp -y",** | | **"sudo adduser ftpuser",** | | **"echo avi6886 | sudo passwd ftpuser --stdin",** | | **"sudo mkdir -p /home/ftpuser/ftp\_dir",** | | **"sudo chmod -R 750 /home/ftpuser/ftp\_dir",** | | **"sudo chown -R ftpuser: /home/ftpuser/ftp\_dir",** | | **"sudo cp /home/avi/vsftpd.conf /etc/vsftpd/vsftpd.conf",** | | **"sudo systemctl restart vsftpd",** | | **"sudo firewall-cmd --permanent --add-port=20-21/tcp",** | | **"sudo firewall-cmd --permanent --add-port=49152-65535/tcp",** | | **"sudo firewall-cmd --reload"** | | **]** | | **}** | | **connection {** | | **type = "ssh"** | | **user = "avi"** | | **password = "Vrockpokey@11"** | | **host = data.azurerm\_public\_ip.example.ip\_address** | | **}** | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**# Initialize terraform configuration**

Initialise your directory containing all the configuration files

**$ terraform init**

**# Format and validate the configuration**

Terraform fmt command automatically updates configuration in the current directory for readability and consistency

Terraform will print out names of files it modified, if any

**$ terraform fmt**

We use terraform validate command to make sure that are configuration is syntactically valid and internally consistent

**$ terraform validate**

**# Apply terraform configuration**

We run terraform apply command to apply our configuration

This shows the execution plan and prompt for approval before proceeding

Type yes for confirmation

If anything seems incorrect or dangerous, it is safe to abort here with no changes made to our infrastructure

**$ terraform apply**

As we apply our configuration, terraform writes data into a file called terraform. tfstate, as week discussed before.

**# Finally testing and result**

Now we try to access our apache and FTP server from web browser or our system by-

* Web browser

For ftp server

**ftp://ftp<ip of server>**

For apache server

**http://<ip of server>**

* From system

For ftp server

**$ ftp**

**ftp> open**

**enter IP of server**

For apache server

**$ whether http://<ip of server>**

**# Conclusions**

**We give knowledge about FTP server and Apache server but most importantly we learnt about a new technology terraform.**

**We learnt how we can set a complete infrastructure on any cloud platform by just using terraform**

**Hence we successfully configured Apache and FTP server on virtual machine using terraform.**