**Terraform Assignment – 1**

You have been asked to:

* **Create an EC2 service in the default subnet in the Ohio region.**

First created EC2 instace in Mumbai region and installed terraform, connected through putty and then using terraform script created another instance in Ohio region as below.

Installation steps:

curl -O https://releases.hashicorp.com/terraform/1.0.11/terraform\_1.0.11\_linux\_amd64.zip

sudo apt install unzip -y

unzip terraform\_1.0.11\_linux\_amd64.zip

sudo mv terraform /usr/local/bin

Now create user with admin policy in Ohio region and use access key and secret key

ubuntu@ip-172-31-40-222:~$ cat variable.tf

variable "access\_key" {

type = string

}

variable "secret\_key" {

type = string

}

ubuntu@ip-172-31-40-222:~$ cat terraform.tfvars

access\_key = "Ohio\_accessKey"

secret\_key = "Ohio\_secretKey"

ubuntu@ip-172-31-40-222:~$ cat provid.tf

provider "aws" {

version = "~> 3.0"

secret\_key = var.secret\_key

access\_key = var.access\_key

region = "us-east-2"

}

ubuntu@ip-172-31-40-222:~$ cat instance.tf

Using ssh-keygen generated connect key

resource "aws\_key\_pair" "genconnectkey" {

key\_name = "connectkey"

public\_key = "ssh-rsa /Zh5ZNK2wA5iDfkBCfZXUUPBJl0tlY/qH36Q48YXV5vp4ppOWWUw5xNtijh7oG3sLJTopD2LdvX1WaOuKi8SaXUssqc7FjNSmxNRIwAgB/+geZQ2HE= ubuntu@ip-172-31-40-222"

}

data "aws\_ami" "ubuntu" {

most\_recent = true

owners = ["099720109477"] # Canonical

filter {

name = "name"

values = ["ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server\*"]

}

filter {

name = "virtualization-type"

values = ["hvm"]

}

filter {

name = "root-device-type"

values = ["ebs"]

}

}

resource "aws\_instance" "web" {

ami = data.aws\_ami.ubuntu.id

instance\_type = "t2.micro"

key\_name = "connectkey"

tags = {

Name = "HelloWorld"

}

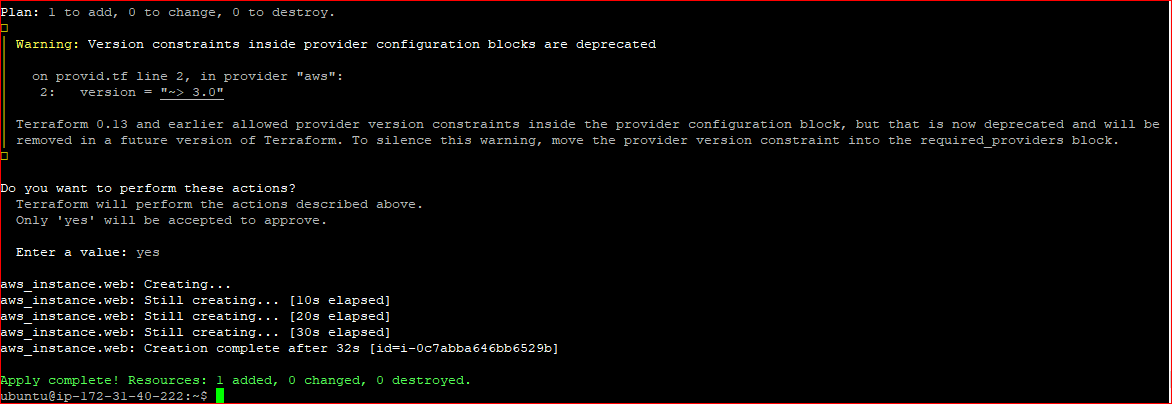
}

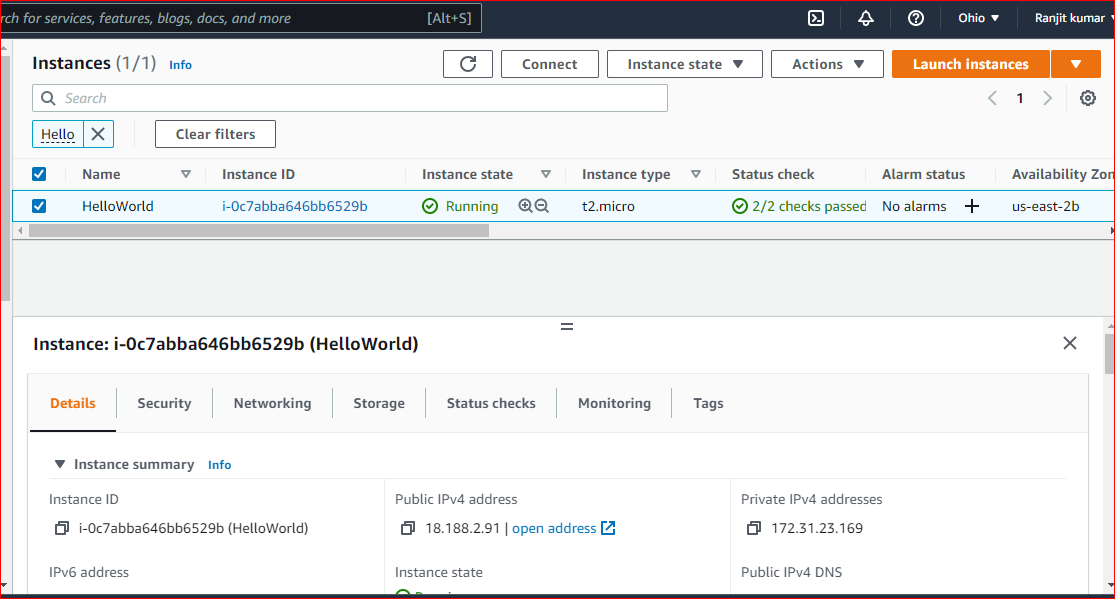
Terraform init

Terraform fmt

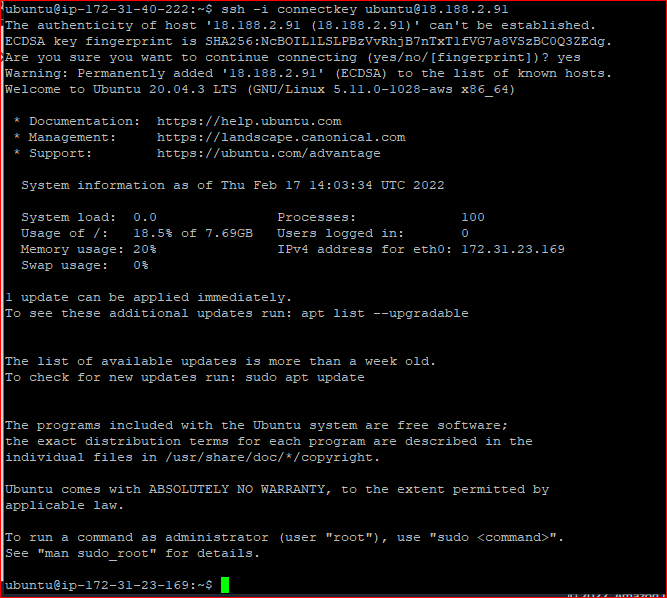
Terraform plan

Terraform apply





Now connected created instance using terraform as below



**Terraform Assignment – 2**

You have been asked to:

* **Destroy the preview deployment**
* **Create a new EC2 instance with an Elastic IP**

terraform destroy # destroyed previews deployment

for variable reference terraform file considered above.

resource "aws\_key\_pair" "genconnectkey" {

key\_name = "connectkey"

public\_key = "ssh-rsa /+geZQ2HE= ubuntu@ip-172-3 1-40-222"

}

resource "aws\_eip" "eip" {

instance = aws\_instance.web.id

vpc = true

}

data "aws\_ami" "ubuntu" {

most\_recent = true

owners = ["099720109477"] # Canonical

filter {

name = "name"

values = ["ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server\*"]

}

filter {

name = "virtualization-type"

values = ["hvm"]

}

filter {

name = "root-device-type"

values = ["ebs"]

}

}

resource "aws\_instance" "web" {

ami = data.aws\_ami.ubuntu.id

instance\_type = "t2.micro"

key\_name = "connectkey"

tags = {

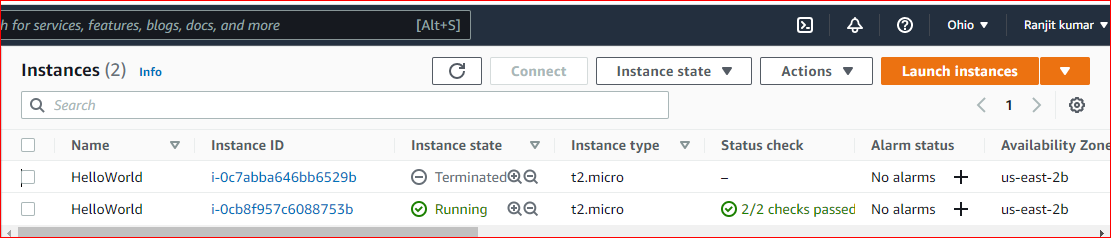
Name = "HelloWorld"

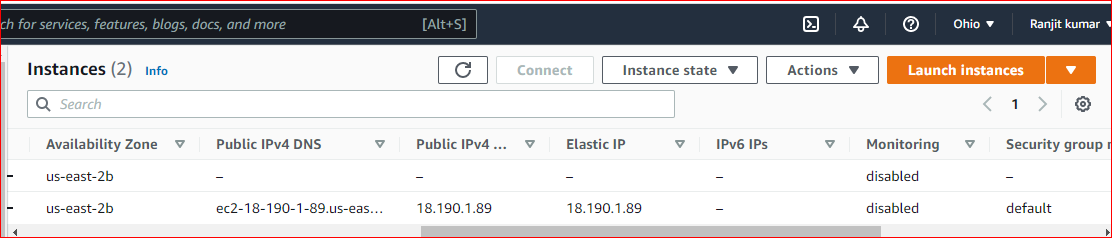
}

}

Terraform plan

Terraform apply





**Terraform Assignment – 3**

You have been asked to:

* **Destroy the previous deployment**
* **Create 2 EC2 instances in Ohio and N.Virginia respectively**
* **Rename Ohio’s instance to ‘hello-ohio’ and Virginia’s instance to ‘hello-virginia’**

Terraform destroy

**Just create new directory and execute below command to install plugin.**

**Terraform init**

There is already user created with admin policy in mubai region.

mumbai

Accesskey: AKIAVKTWTGWR6VP6SBNE

Secretkey: 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

ubuntu@ip-172-31-40-222:~$ cat variable.tf

variable "access\_key" {

type = string

}

variable "secret\_key" {

type = string

}

ubuntu@ip-172-31-40-222:~$ cat terraform.tfvars

access\_key = "AKIAVKTWTGWR6VP6SBNE"

secret\_key = "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"

ubuntu@ip-172-31-40-222:~$ cat provid.tf

provider "aws" {

version = "~> 3.0"

secret\_key = var.secret\_key

access\_key = var.access\_key

region = "us-east-2"

}

ubuntu@ip-172-31-40-222:~$ cat ohioinstance.tf

resource "aws\_key\_pair" "genconnectkey" {

key\_name = "connectkey"

public\_key = "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDBoGLDuPnp4R6XNGNI8dJdYO0FrgVKZKd8E0LGhHb5mARFfsmcr2lNlc5zzoGED5h9wkEDXDtXKmyw hsfy/bAMJgFxBq6vVYFBfR8zS3KEPx833IR/JCgf+KmqbKcVmoFr7tauHfOUTs9e6guIJ3ceN+yxmgEIyglVRTS36Z7sph28DAVmzoR5HhzyykXxwr+TZOX7djivlPw/9xBT rT4fO4BHQXlIEeUB1dlg1TvHB6w2UGcVjZtNDv96apx8hcqvxnUU9CEU1wdfOpA0SOLy5umvB3s7mtSC1ThjexZv6eUylfxvlszrWga/YPqXr+dy/aWMr5wNT6Jf7pS5TajU BoFU5qBYFFwVmuuhJexn1ggM3psJBS8ezJcgqtk7HupCfjv9V3fHEP0a0/Zh5ZNK2wA5iDfkBCfZXUUPBJl0tlY/qH36Q48YXV5vp4ppOWWUw5xNtijh7oG3sLJTopD2LdvX 1WaOuKi8SaXUssqc7FjNSmxNRIwAgB/+geZQ2HE= ubuntu@ip-172-31-40-222"

}

data "aws\_ami" "ubuntu" {

most\_recent = true

owners = ["099720109477"] # Canonical

filter {

name = "name"

values = ["ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server\*"]

}

filter {

name = "virtualization-type"

values = ["hvm"]

}

filter {

name = "root-device-type"

values = ["ebs"]

}

}

resource "aws\_instance" "web" {

ami = data.aws\_ami.ubuntu.id

instance\_type = "t2.micro"

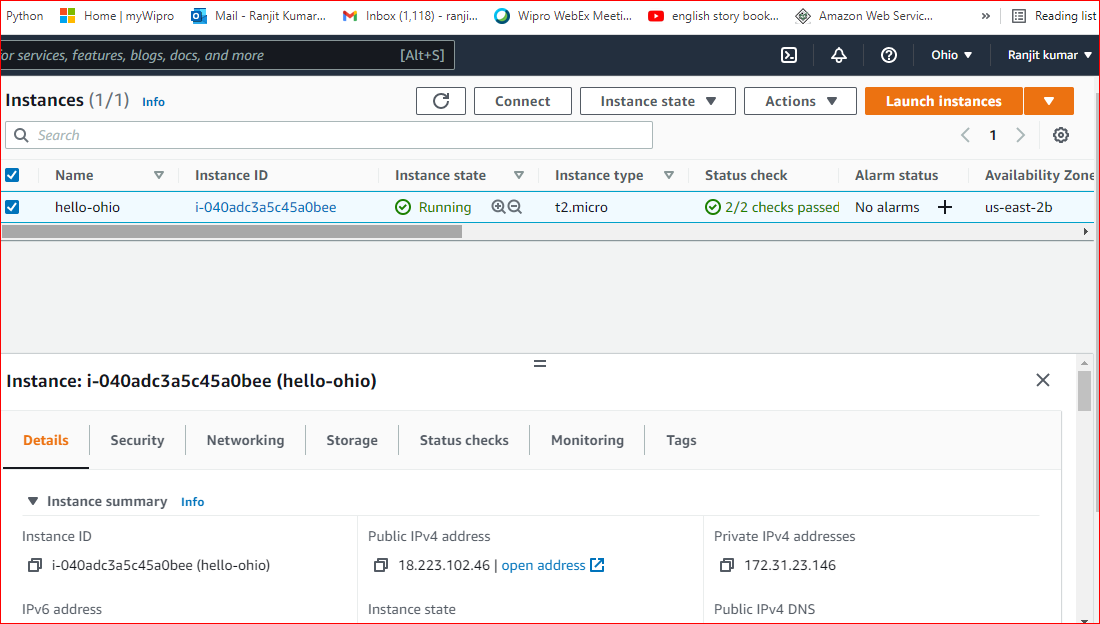
key\_name = "connectkey"

tags = {

Name = "hello-ohio"

}

}



Using public ip connect putty through generated key.

Ssh –I connectkey ubuntu@publicIP

**Now Just create new directory and execute below command to install plugin.**

**Terraform init**

ubuntu@ip-172-31-40-222:~/virginia$ vi virgeninstance.tf

resource "aws\_key\_pair" "genconnectkey" {

key\_name = "connectkey"

public\_key = "ssh-rsa  ubuntu@ip-172-31-40-222"

}

resource "aws\_instance" "web" {

ami = "ami-04505e74c0741db8d"

instance\_type = "t2.micro"

key\_name = "connectkey"

tags = {

Name = "hello-virginia"

}

}

ubuntu@ip-172-31-40-222:~/virginia$ cat provid.tf

provider "aws" {

version = "~> 3.0"

secret\_key = var.secret\_key

access\_key = var.access\_key

region = "us-east-1"

}

ubuntu@ip-172-31-40-222:~/virginia$ cat variable.tf

variable "access\_key" {

type = string

}

variable "secret\_key" {

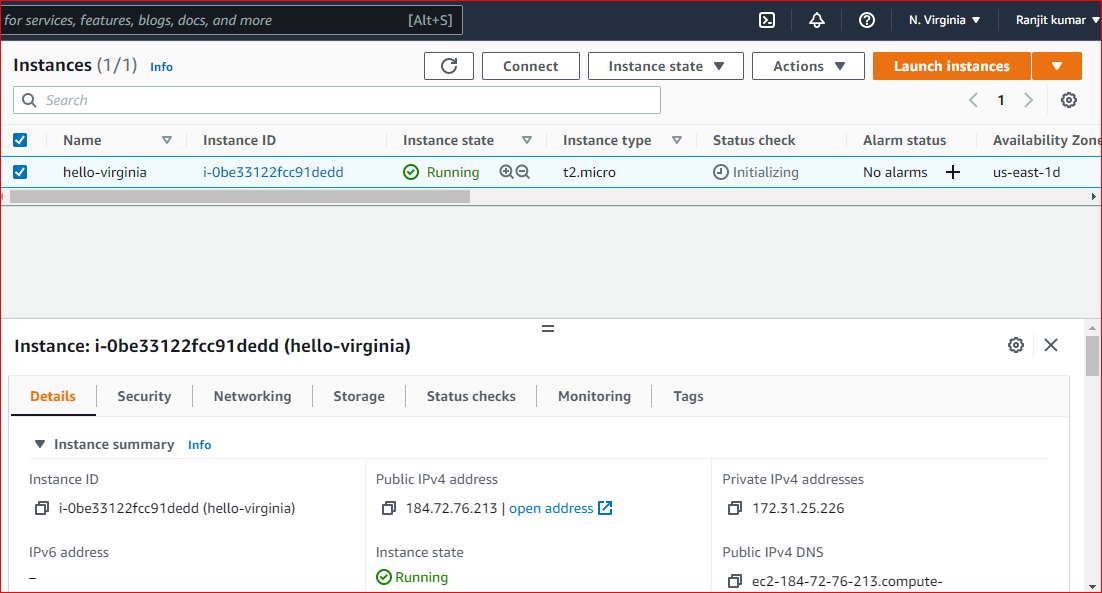
type = string

}

ubuntu@ip-172-31-40-222:~/virginia$ cat terraform.tfvars

access\_key = "AKIAVKTWTG"

secret\_key = "mH3Rnp2/q7MIhyExCBZ"



Using public ip connect putty through generated key.

Ssh –I connectkey ubuntu@publicIP

**Terraform Assignment – 4**

You have been asked to:

* Destroy the previous deployments
* Create a VPC with the required components using Terraform
* Deploy an EC2 instance inside the VPC

Terraform destroy

Create seprate folder and execute below command.

ubuntu@ip-172-31-40-222:~/folder1$ cat vpc.tf

resource "aws\_vpc" "main" {

cidr\_block = "10.0.0.0/16"

tags={

Name = "demo\_vpc\_terraform"

}

}

resource "aws\_subnet" "public\_subnet" {

vpc\_id = aws\_vpc.main.id

cidr\_block = "10.0.1.0/24"

map\_public\_ip\_on\_launch = "1"

availability\_zone\_id = "aps1-az1"

tags = {

Name = "public\_tf\_subnet"

}

}

resource "aws\_security\_group" "securtiy\_group" {

vpc\_id = aws\_vpc.main.id

ingress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

tags = {

Name = "tf-security\_grp"

}

}

resource "aws\_internet\_gateway" "gw" {

vpc\_id = aws\_vpc.main.id

tags = {

Name = "tf-IGW"

}

}

#route table for public subnet with IGW

resource "aws\_route\_table" "table\_public" {

vpc\_id = "${aws\_vpc.main.id}"

route {

cidr\_block = "0.0.0.0/0"

gateway\_id = "${aws\_internet\_gateway.gw.id}"

}

tags = {

Name = "rt\_public"

}

}

# route table association public subnet

resource "aws\_route\_table\_association" "association\_rt\_public" {

subnet\_id = aws\_subnet.public\_subnet.id

route\_table\_id = aws\_route\_table.table\_public.id

}

# launch an instance

provider "aws" {

access\_key = "AKIAVKTWVP6SBNE"

secret\_key = "mH3Rnp2/q7MIhyExCB"

region = "ap-south-1"

}

# launch an instance

resource "aws\_instance" "web" {

ami = "ami-08ee6644906ff4d6c"

instance\_type = "t2.micro"

vpc\_security\_group\_ids= ["${aws\_security\_group.securtiy\_group.id}"]

subnet\_id= aws\_subnet.public\_subnet.id

key\_name= "latest\_key"

tags = {

Name = "master"

}

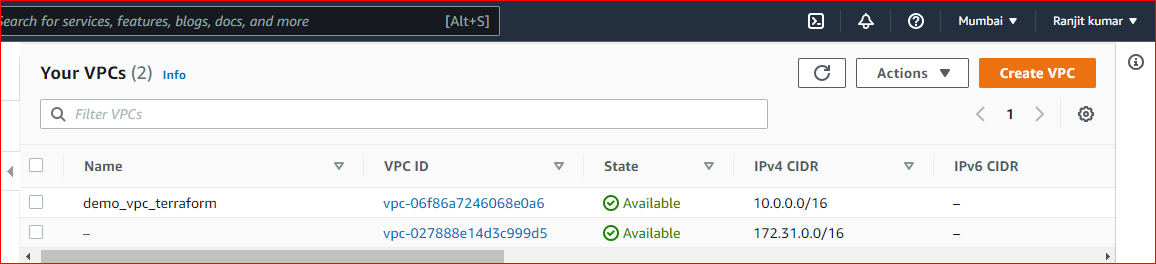
}

Terraform init

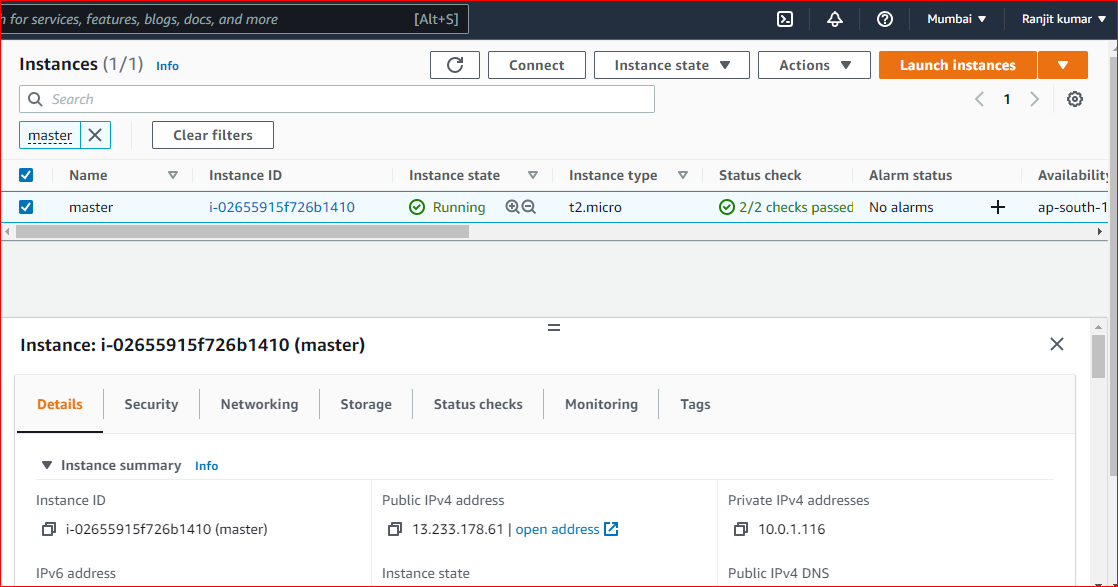
Terraform plan

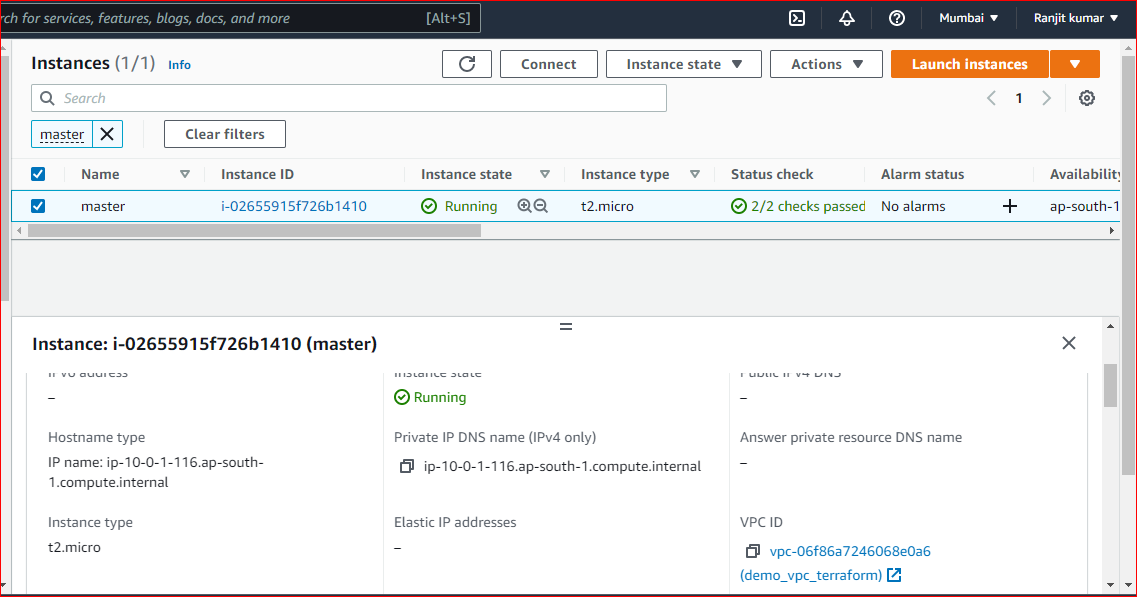
Terraform apply

Vpc is created.



Below is instance created with created vpc.





**Terraform Assignment – 5**

You have been asked to:

* **Destroy the previous deployment**
* **Create a script to install apache2**
* **Run this script on a newly created EC2 instance**
* **Print the IP address of the instance in a file on the local, once deployed**.

Terraform destroy.

Vi script.sh

#! /bin/bash

sudo apt-get update

sudo apt-get install apache2 -y

sudo echo "Welcome to apache"> /var/www/html/index.html

ubuntu@ip-172-31-40-222:~/folder2$ cat instance.tf

resource "aws\_instance" "web" {

ami = "ami-0851b76e8b1bce90b"

instance\_type = "t2.micro"

key\_name= "latest\_key"

tags = {

Name = "master"

}

user\_data = file("${path.module}/script.sh")

provisioner "local-exec" {

working\_dir = "/tmp/"

command = "echo ${self.public\_ip} > mypublicIP.txt"

}

}

ubuntu@ip-172-31-40-222:~/folder2$ cat terraform.tfvars

access\_key = "AKIAVKTWTGWSBNE"

secret\_key = "mH3Rnp2/q7MIhyExCBZMIgkwj"

ubuntu@ip-172-31-40-222:~/folder2$ cat variable.tf

variable "access\_key" {

type = string

}

variable "secret\_key" {

type = string

}

ubuntu@ip-172-31-40-222:~/folder2$ cat provid.tf

provider "aws" {

version = "~> 3.0"

secret\_key = var.secret\_key

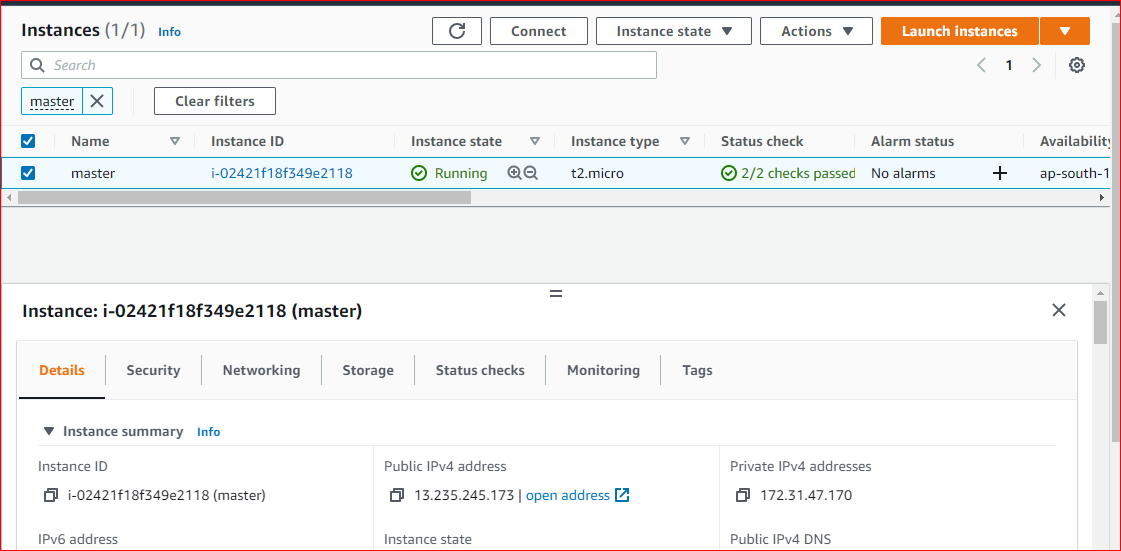
access\_key = var.access\_key

region = "ap-south-1"

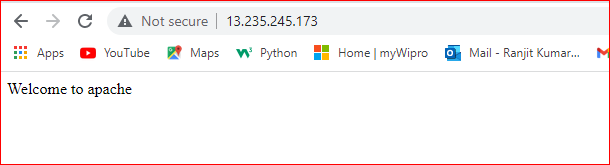
}

Terraform plan

Terraform apply



URL: ec2publicIP



Public ip of the instances is saved in local system as below.



**CASE STUDY –CREATING AN ARCHITECTURE USING TERRAFORM ON AWS**

**You work as a DevOps Engineer in leading Software Company. You have been asked to build an infrastructure safely and efficiently.**

The company Requirements:

1. **Use AWS cloud Provider and the software to be installed is Apache2**
2. **Use Ubuntu AMI**

The company wants the Architecture to have the following services:

1. **Create a template with a VPC, 2 subnets and 1 instance in each subnet**
2. **Attach Security groups, internet gateway and network interface to the instance.**

ubuntu@ip-172-31-40-222:~/casestudy$ cat vpc.tf

resource "aws\_vpc" "myvpc" {

cidr\_block = "10.0.0.0/16"

instance\_tenancy = "default"

tags = {

Name = "myvpc"

}

}

resource "aws\_subnet" "pub\_subnet" {

vpc\_id = aws\_vpc.myvpc.id

cidr\_block = "10.0.1.0/24"

map\_public\_ip\_on\_launch = "1"

availability\_zone\_id = "aps1-az1"

tags = {

Name = "pub\_subnet"

}

}

resource "aws\_subnet" "pub\_subnet1" {

vpc\_id = aws\_vpc.myvpc.id

cidr\_block = "10.0.2.0/24"

map\_public\_ip\_on\_launch = "1"

availability\_zone\_id = "aps1-az3"

tags = {

Name = "pub\_subnet1"

}

}

resource "aws\_internet\_gateway" "internet\_gw" {

vpc\_id = aws\_vpc.myvpc.id

tags = {

Name = "internet\_gw"

}

}

resource "aws\_route\_table" "rout\_table" {

vpc\_id = aws\_vpc.myvpc.id

route {

cidr\_block = "0.0.0.0/0"

gateway\_id = aws\_internet\_gateway.internet\_gw.id

}

tags = {

Name = "rout\_table"

}

}

resource "aws\_route\_table\_association" "rout\_table\_associate" {

subnet\_id = aws\_subnet.pub\_subnet.id

route\_table\_id = aws\_route\_table.rout\_table.id

}

resource "aws\_route\_table\_association" "rout\_table\_associate1" {

subnet\_id = aws\_subnet.pub\_subnet1.id

route\_table\_id = aws\_route\_table.rout\_table.id

}

resource "aws\_security\_group" "securtiy\_group" {

vpc\_id = aws\_vpc.myvpc.id

ingress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

tags = {

Name = "tf-security\_grp"

}

}

ubuntu@ip-172-31-40-222:~/casestudy$ cat variable.tf

variable "access\_key" {

type = string

}

variable "secret\_key" {

type = string

}

ubuntu@ip-172-31-40-222:~/casestudy$ cat terraform.tfvars

access\_key = "AKIAVKTWTGW6SBNE"

secret\_key = "mH3Rnp2/q7MIhyExCBZMClvNQsMIgkwj"

ubuntu@ip-172-31-40-222:~/casestudy$ cat instance.tf

resource "aws\_instance" "web" {

ami = "ami-08ee6644906ff4d6c"

instance\_type = "t2.micro"

vpc\_security\_group\_ids= ["${aws\_security\_group.securtiy\_group.id}"]

subnet\_id= aws\_subnet.pub\_subnet.id

key\_name= "latest\_key"

tags = {

Name = "subnet\_instance"

}

user\_data = file("${path.module}/script.sh")

}

resource "aws\_instance" "web1" {

ami = "ami-08ee6644906ff4d6c"

instance\_type = "t2.micro"

vpc\_security\_group\_ids= ["${aws\_security\_group.securtiy\_group.id}"]

subnet\_id= aws\_subnet.pub\_subnet1.id

key\_name= "latest\_key"

tags = {

Name = "subnet1\_instance"

}

user\_data = file("${path.module}/script.sh")

}

resource "aws\_network\_interface" "network\_int" {

subnet\_id = aws\_subnet.pub\_subnet.id

private\_ips = ["10.0.1.10"] # ip range should be as per vpc CIDR block.

security\_groups = [aws\_security\_group.securtiy\_group.id]

attachment {

instance = aws\_instance.web.id

device\_index = 1

}

tags = {

Name = "example"

}

}

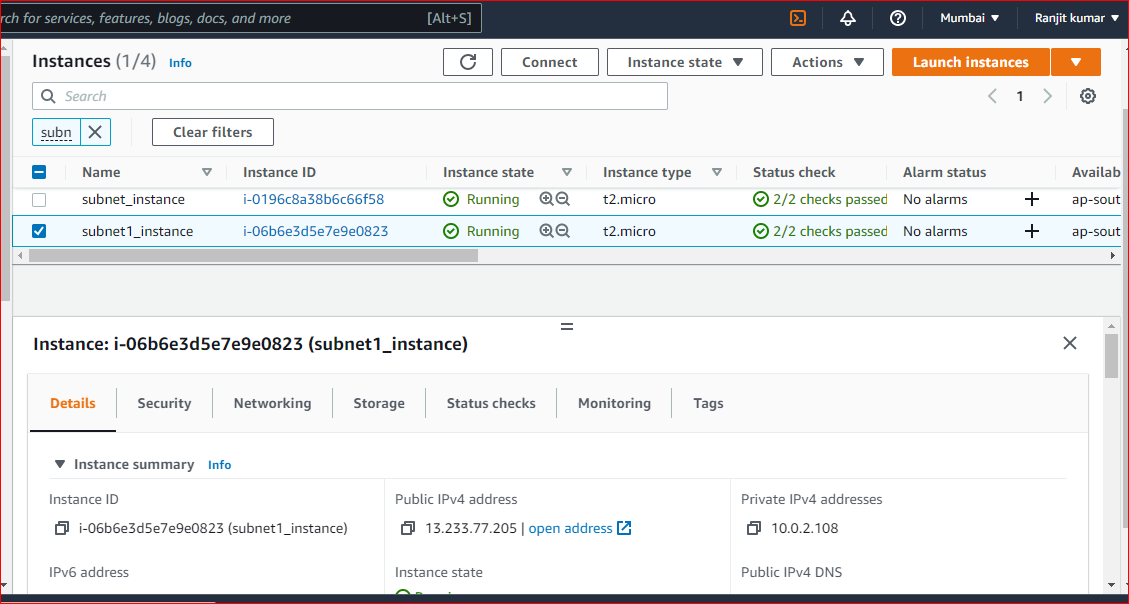
ubuntu@ip-172-31-40-222:~/casestudy$ cat script.sh

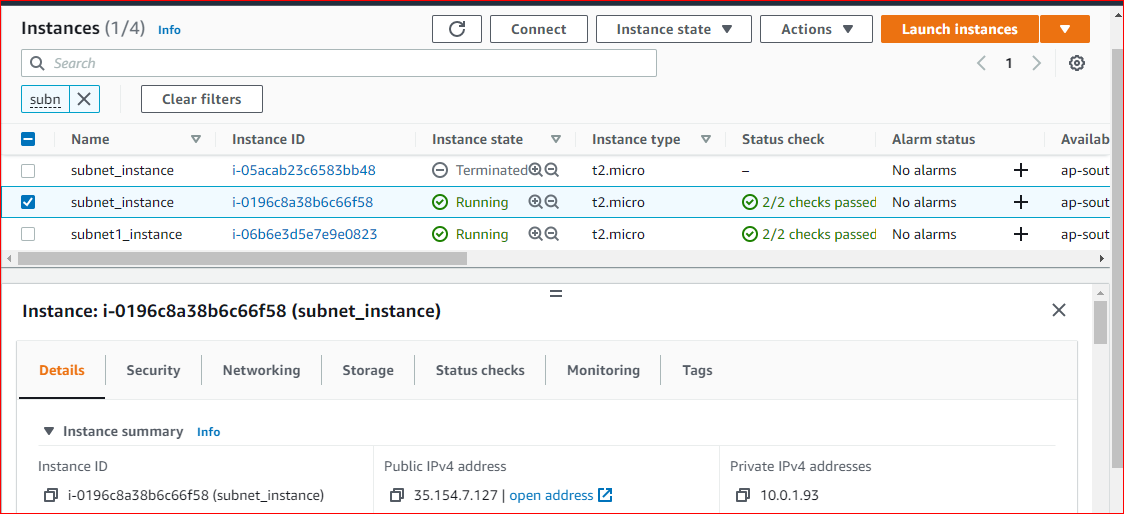
#! /bin/bash

sudo apt-get update

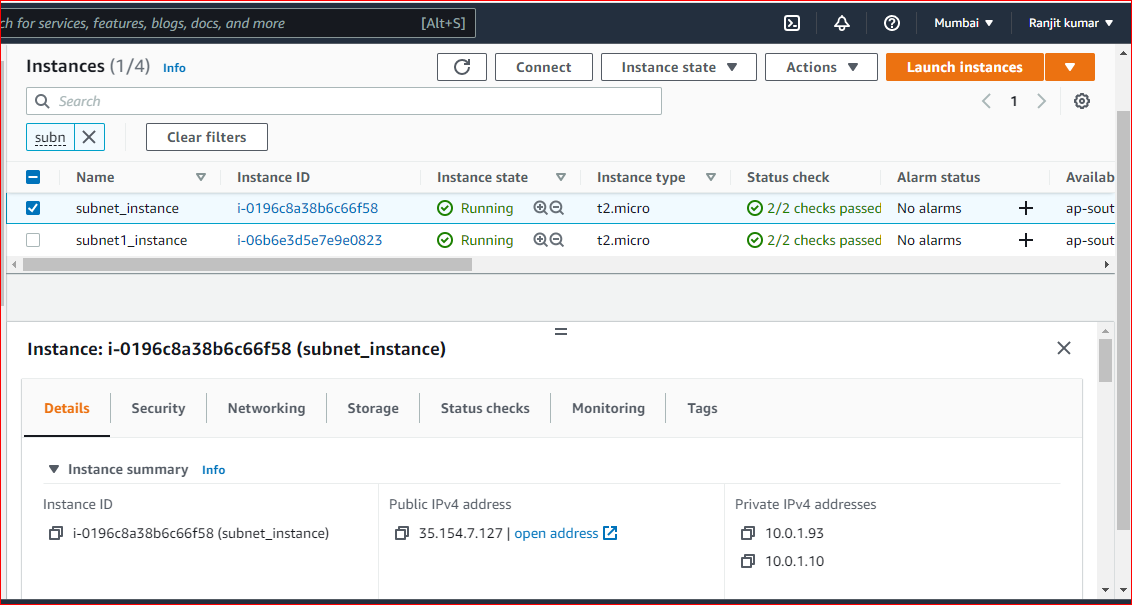
sudo apt-get install apache2 -y

sudo echo "Welcome to apache"> /var/www/html/index.html

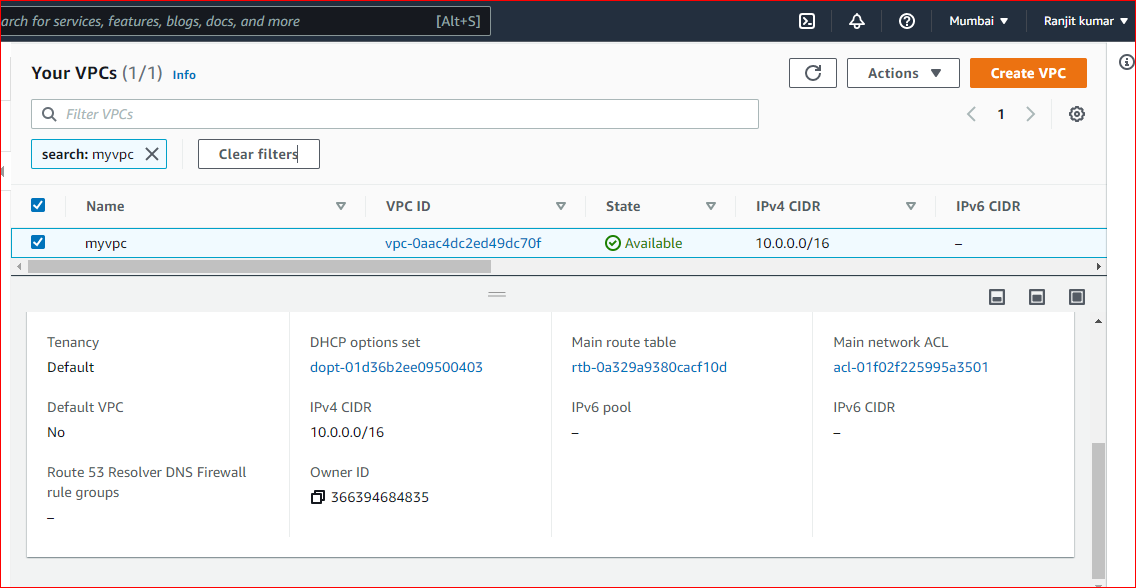


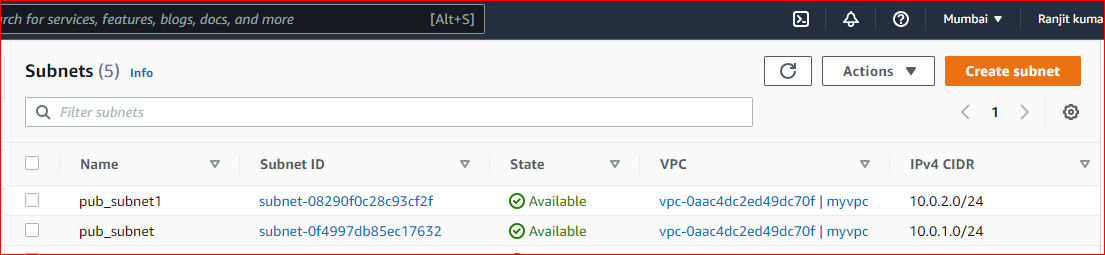


In blew instances network interface is attached 10.0.1.10

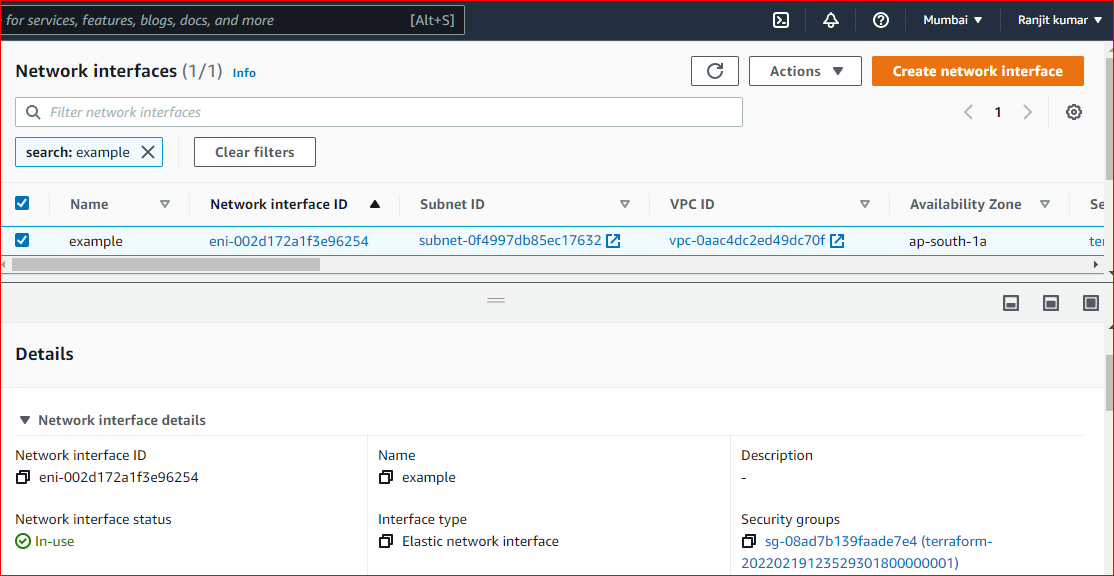


Below one is create vpc.





Below one is create network interface.



URL:

