Ishita Singhal
R171217025
500060649
DevOps Sem 7

ASSIGNMENT 1

Write Terraform script to do perform following tasks on AWS cloud Platform

First Terraform is intalled and configured in our machine.

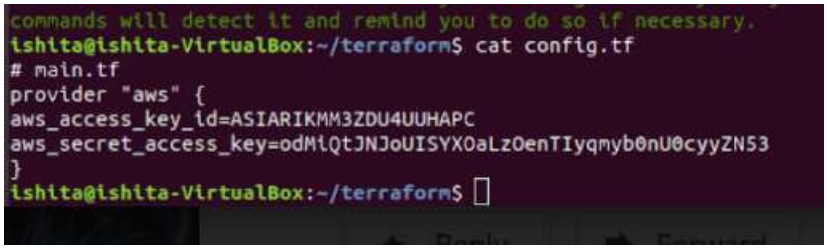
The successful installation of terraform is checked using the `terraform version` command.

```
ishita@ishita-VirtualBox:~/project-terraform$ sudo mv terraform_0.13.5_linux_amd64/ /usr/local/bin
[sudo] password for ishita:
ishita@ishita-VirtualBox:~/project-terraform$ terraform version
Terraform v0.13.5
ishita@ishita-VirtualBox:~/project-terraform$
```

```
ishita@ishita-VirtualBox:~/terraform$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v3.16.0...
- Installed hashicorp/aws v3.16.0 (signed by HashiCorp)

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, we recommend adding version constraints in a required_providers block
in your configuration, with the constraint strings suggested below.

* hashicorp/aws: version = "~> 3.16.0"

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
ishita@ishita-VirtualBox:~/terraform$
```

Ishita Singhal
R171217025
500060649
DevOps Sem 7

Then terraform is configured to aws by adding the necessary files and access key id and password in the terraform configuration file, `config.tf`.



Ishita Singhal
R171217025
500060649
DevOps Sem 7

Step 1: Create two T2 Micro EC2 Instances.

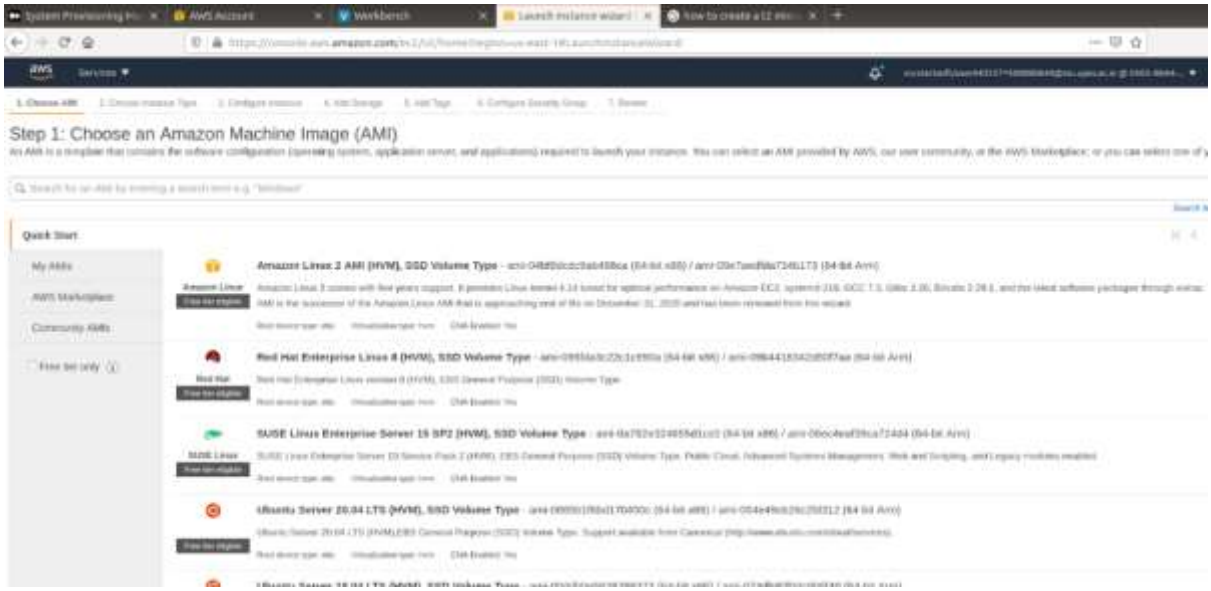
Terraform script for creating two EC2 instances:

Open ▾ Instances.tf
~/terraform

```
provider "aws" {
  region = "ap-south-1"
}

provider "aws" {
  region = "us-eadt-1"
  alias = "usa"

}

resource "aws_instance" "first-instance-india" {
  ami = "ami-0e306788ff2473ccb"
  instance_type = "t2.micro"
}

resource "aws_instance" "second-instance-USA" {
  ami = "ami-0947d2ba12ee1ff75"
  instance_type = "t2.micro"
  privider = aws.usa
}
```

Ishita Singhal
R171217025
500060649
DevOps Sem 7

An AMI is chosen from the AWS management console.



Instances created.

Ishita Singhal
R171217025
500060649
DevOps Sem 7

Step2: Create a VPN on AWS

Terraform script for creating VPN on AWS.

```
shita@ishita-VirtualBox:~/terraform$ cat vpn.tf
esource "aws_vpc" ""vpc" {
  cidr_block = "10.0.0.0/16"
}

esource "aws_vpn_gateway" "vpn_gateway" {
  vpc_id = aws_vpc.vpc.id
}

esource "aws_customer_gateway" "customer_gateway" {
  bgp_asn = 65000
  ip_address = "172.0.0.1"
  type = "ipsec.1"
}

esource "aws_vpn_connection" "main" {
  vpn_gateway_id = aws_vpn_gateway.vpn_gateway.id
  customer_gateway_id = aws_cutomer_gateway.customer_gateway
  type = "ipsec.1"
  static_routes_only = true
}
```

VPN by the name ishita-vpc created.

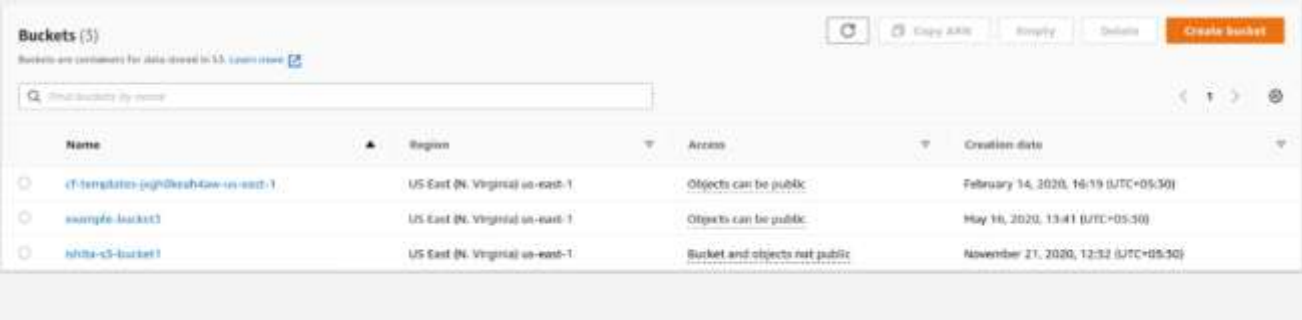
Ishita Singhal
R171217025
500060649
DevOps Sem 7

Step 3: Create a S3 Bucket

Terraform script for creating an S3 bucket.

```
}
ishita@ishita-VirtualBox:~/terraform$ cat s3.tf
resource "aws_s3_bucket" "ishita-s3-bucket" {
  bucket = "ishita-s3-bucket1234"
  ac1 = "public-read"
  tags = {
    Name = "ishita-s3-bucket1"
  }
  versioning {
    enabled = true
  }
}
```

S3 bucket by the name ishita-s3-bucket1 created.



Ishita Singhal
R171217025
500060649
DevOps Sem 7

Step 4: Terraform commands to run the terraform files:

- Terraform init: This command initailizes the working directory which contains all the terraform configuration files.

```
ishita@ishita-VirtualBox:~/terraform$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v3.16.0...
- Installed hashicorp/aws v3.16.0 (signed by HashiCorp)

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, we recommend adding version constraints in a required_providers block
in your configuration, with the constraint strings suggested below.

* hashicorp/aws: version = "~> 3.16.0"

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

- Terraform plan: This command is used to create an execution plan.
- Terraform apply: This command applies the changes which were required to perform the task assigned.

```
onfiguration and real physical resources that exist. As a re
ctions need to be performed.
shita@ishita-VirtualBox:~/terraform$ terraform apply

pply complete! Resources: 0 added, 0 changed, 0 destroyed.
shita@ishita-VirtualBox:~/terraform$ gedit config.tf
```

- Terraform validate: This validates the configuration file in a directory.