Garishma Virk
R171217017
Btech CSE-DevOps
500060134

Assignment -1

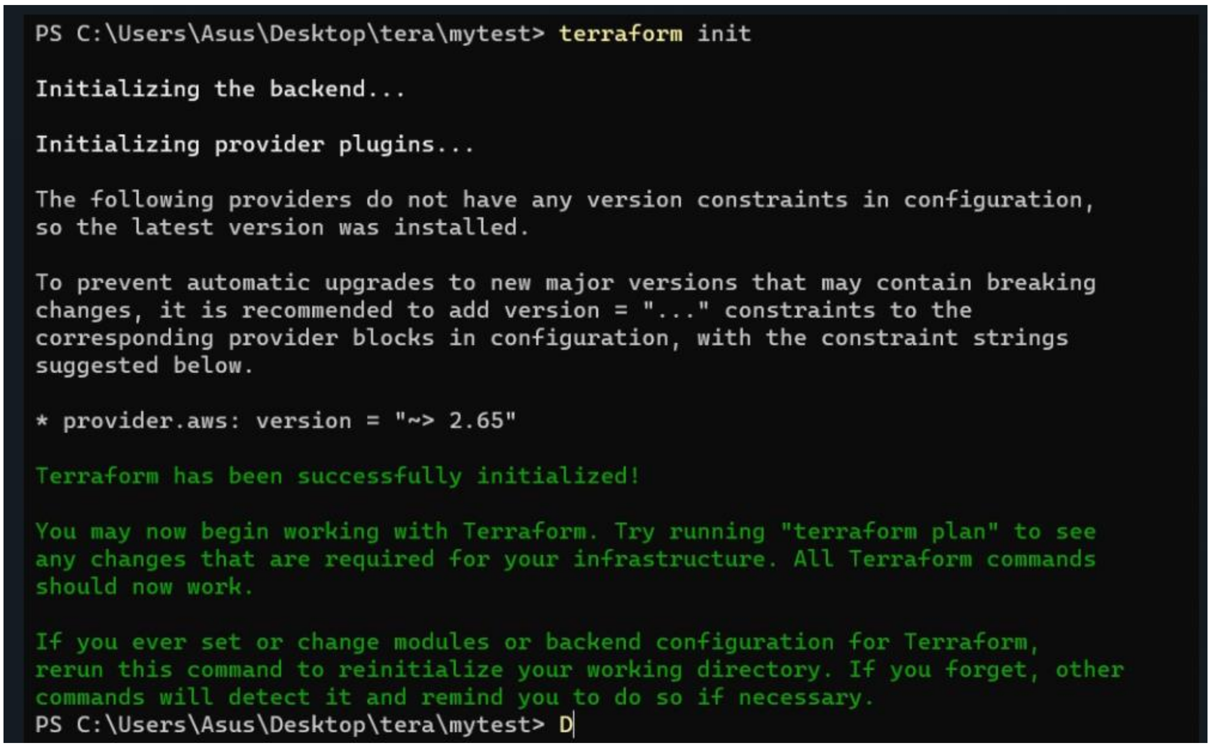
Write Terraform script to do perform following tasks on AWS cloud Platform
Create two T2 Micro EC2 Instances. Create a VPN on AWS. Create a S3 Bucket

Prerequisite

1. Install latest version of Terraform
2. Read about Amazon EC2
3. Access to AWS account (console)
4. AWS Secret Key and AWS access key Id

Let's get started with creating EC2 instance using Terraform.

Step1: Installing Terraform:



```
PS C:\Users\Asus\Desktop\tera\mytest> terraform init

Initializing the backend...

Initializing provider plugins...

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, it is recommended to add version = "..." constraints to the
corresponding provider blocks in configuration, with the constraint strings
suggested below.

* provider.aws: version = "~> 2.65"

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.
PS C:\Users\Asus\Desktop\tera\mytest> D
```

Create a directory(example-spcm) and go to your directory using the following commands

mkdir spcm

cd spcm

Configure AWS credentials

--aws configure: complete the steps of inserting aws credentials in cmd after creating IAM

User with permissions on aws console.

Step2: Creating Terraform Scripts:

Garishma Virk
R171217017
Btech CSE-DevOps

```
provider "aws"  {
 region= "ap-south-1"
 profile= "nandinisood"
}
resource "aws_instance" "myFirstInstance" {
  ami          = "ami-0db0b3ab7df22e366"
  count=2
  key_name = "keypair"
  instance_type = "t2.micro"
  security_groups= [ "security_jenkins_port"]
  tags= {
    Name = "jenkins_instance"
  }
}
resource  "aws_s3_bucket" "tf_course" {
  bucket = "ilovedevops987"
  acl    = "private"
}
resource "aws_vpc" "vpc" {
  cidr_block = "10.0.0.0/16"
}
resource "aws_vpn_gateway" "vpn_gateway" {
  vpc_id = aws_vpc.vpc.id
}
resource "aws_customer_gateway" "customer_gateway" {
  bgp_asn    = 65000
  ip_address = "172.0.0.1"
  type       = "ipsec.1"
}
resource "aws_vpn_connection" "main" {
  vpn_gateway_id      = aws_vpn_gateway.vpn_gateway.id
  customer_gateway_id = aws_customer_gateway.customer_gateway.id
  type              = "ipsec.1"
  static_routes_only  = true
}
resource "aws_security_group" "security_jenkins_port" {
  name        = "security_jenkins_port"
  description = "security group for jenkins"

  ingress {
    from_port   = 8080
    to_port     = 8080
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
 ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
# outbound from jenkis server
  egress {
    from_port   = 0
    to_port     = 65535
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
 tags= {
    Name = "security_jenkins_port"
  }
}
```

Step 3: Now run the following commands:

Garishma Virk
R171217017
Btech CSE-DevOps
500060134

- **terraform init**

It initializes working directory containing terraform configuration files(.tf files) and it is safe to run this command multiple times.

- **terraform validate**

It checks if terraform scripts have no syntax errors and are internally consistent.

- **terraform plan**

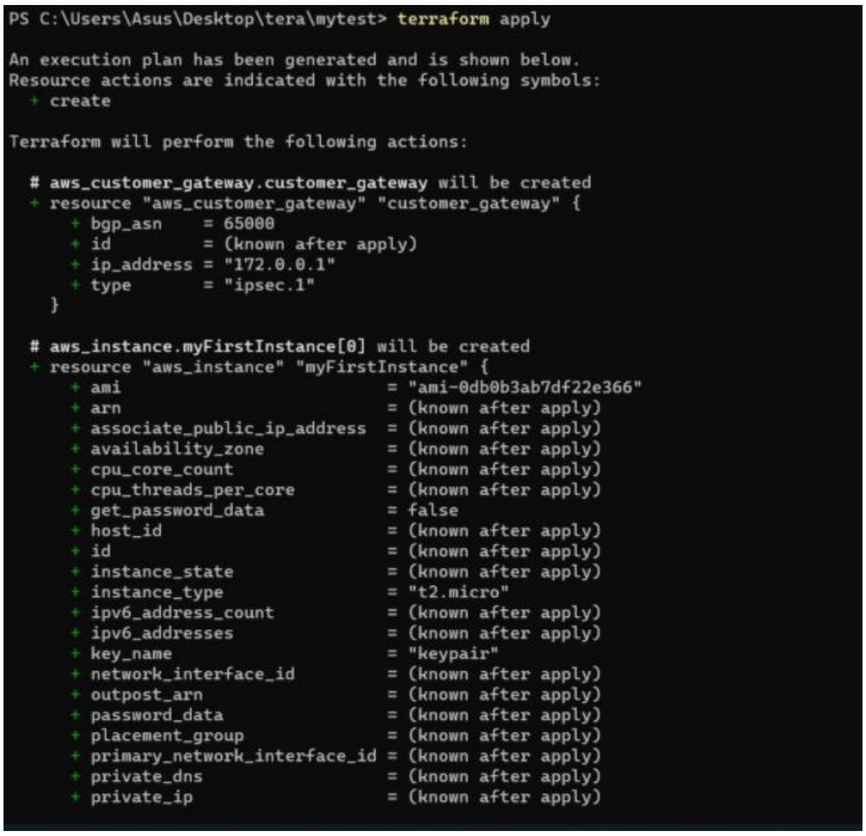
It create execution plan that helps you check whether execution plan matches your expectations.

- **terraform apply**

It applies changes to reach the desired state of the configuration.

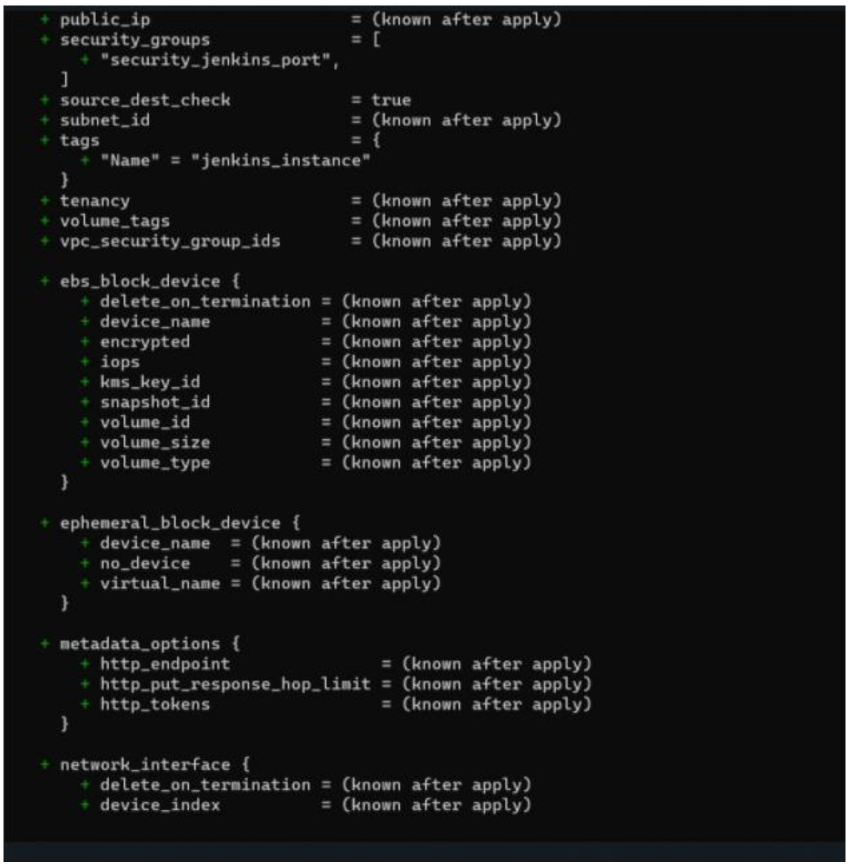
- **terraform destroy**

It terminates resources defined in your Terraform configuration.

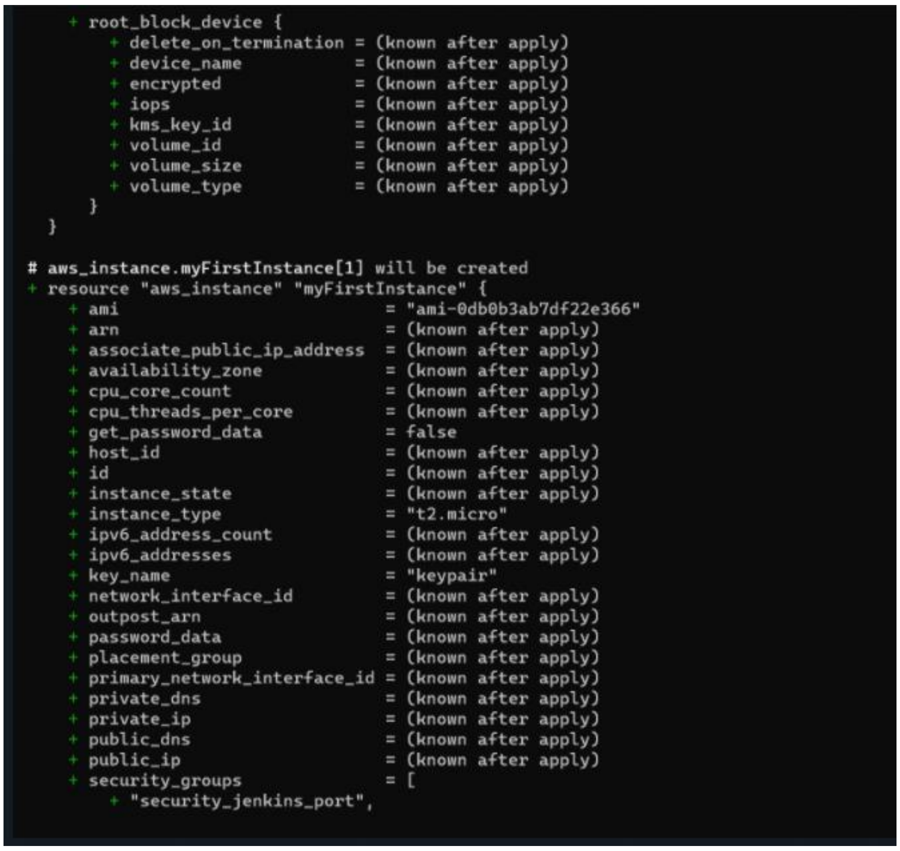


```
PS C:\Users\Asus\Desktop\tera\mytest> terraform apply

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_customer_gateway.customer_gateway will be created
  + resource "aws_customer_gateway" "customer_gateway" {
      + bgp_asn    = 65000
      + id         = (known after apply)
      + ip_address = "172.0.0.1"
      + type       = "ipsec.1"
    }

  # aws_instance.myFirstInstance[0] will be created
  + resource "aws_instance" "myFirstInstance" {
      + ami                          = "ami-0db0b3ab7df22e366"
      + arn                          = (known after apply)
      + associate_public_ip_address  = (known after apply)
      + availability_zone            = (known after apply)
      + cpu_core_count               = (known after apply)
      + cpu_threads_per_core         = (known after apply)
      + get_password_data            = false
      + host_id                      = (known after apply)
      + id                           = (known after apply)
      + instance_state               = (known after apply)
      + instance_type                = "t2.micro"
      + ipv6_address_count           = (known after apply)
      + ipv6_addresses               = (known after apply)
      + key_name                     = "keypair"
      + network_interface_id         = (known after apply)
      + outpost_arn                  = (known after apply)
      + password_data                = (known after apply)
      + placement_group              = (known after apply)
      + primary_network_interface_id = (known after apply)
      + private_dns                  = (known after apply)
      + private_ip                   = (known after apply)
```

Garishma Virk
R171217017
Btech CSE-DevOps
500060134

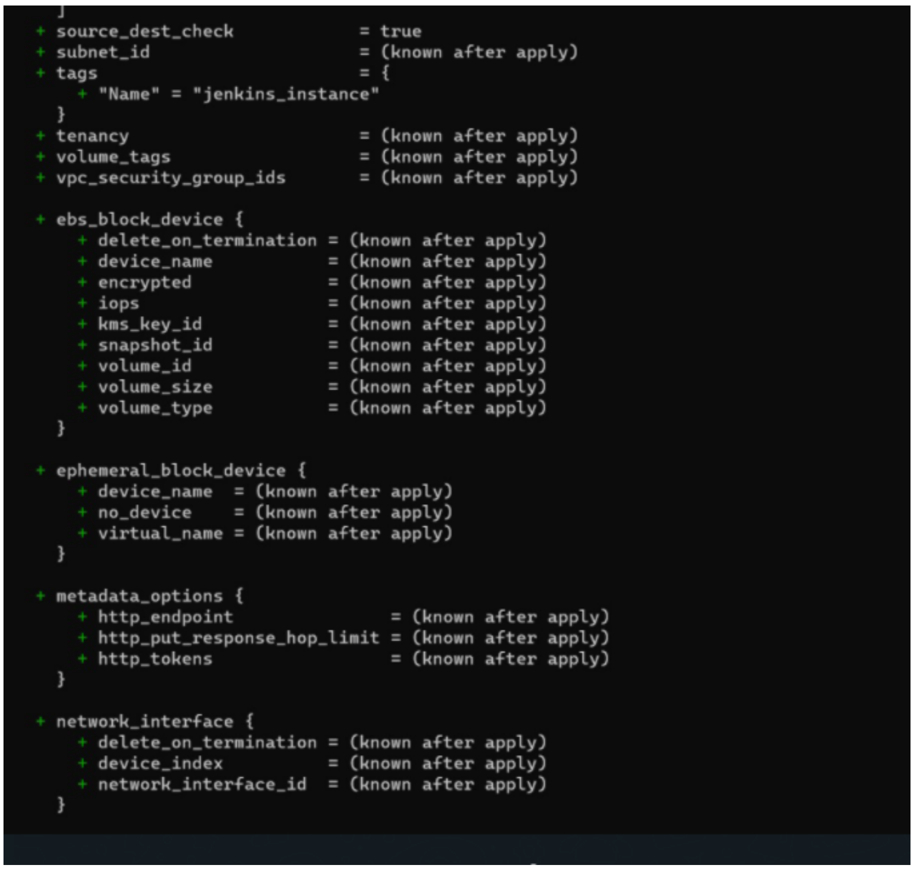


```
      + public_ip                    = (known after apply)
      + security_groups              = [
          + "security_jenkins_port",
        ]
      + source_dest_check            = true
      + subnet_id                    = (known after apply)
      + tags                         = {
          + "Name" = "jenkins_instance"
        }
      + tenancy                      = (known after apply)
      + volume_tags                  = (known after apply)
      + vpc_security_group_ids       = (known after apply)

      + ebs_block_device {
          + delete_on_termination = (known after apply)
          + device_name           = (known after apply)
          + encrypted             = (known after apply)
          + iops                  = (known after apply)
          + kms_key_id            = (known after apply)
          + snapshot_id           = (known after apply)
          + volume_id             = (known after apply)
          + volume_size           = (known after apply)
          + volume_type           = (known after apply)
        }

      + ephemeral_block_device {
          + device_name  = (known after apply)
          + no_device    = (known after apply)
          + virtual_name = (known after apply)
        }

      + metadata_options {
          + http_endpoint               = (known after apply)
          + http_put_response_hop_limit = (known after apply)
          + http_tokens                 = (known after apply)
        }

      + network_interface {
          + delete_on_termination = (known after apply)
          + device_index          = (known after apply)
```

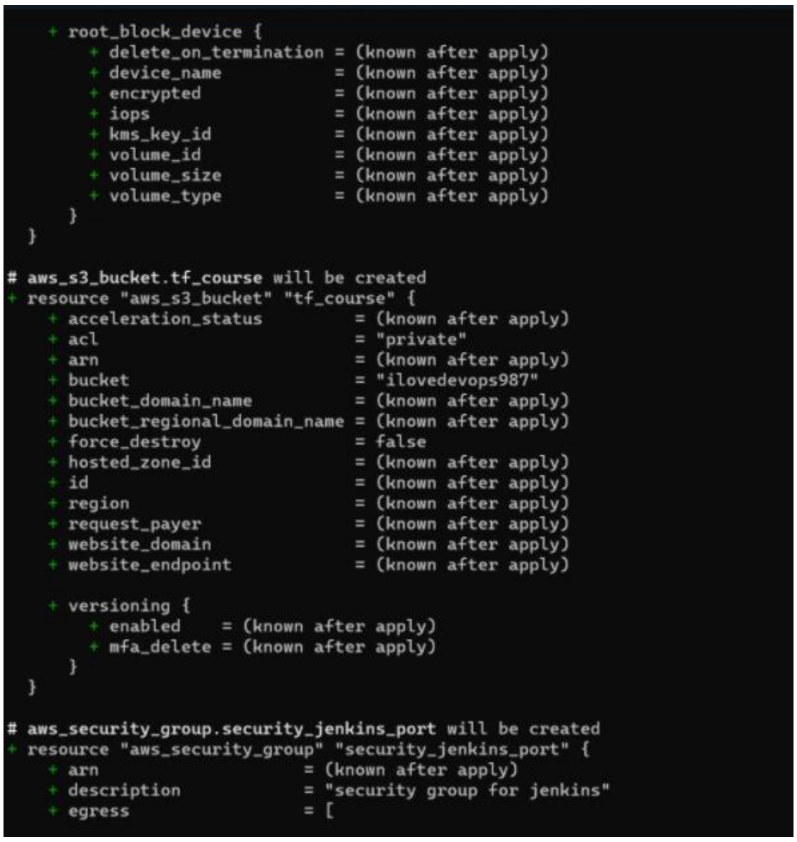


```
      + root_block_device {
          + delete_on_termination = (known after apply)
          + device_name           = (known after apply)
          + encrypted             = (known after apply)
          + iops                  = (known after apply)
          + kms_key_id            = (known after apply)
          + volume_id             = (known after apply)
          + volume_size           = (known after apply)
          + volume_type           = (known after apply)
        }
    }

  # aws_instance.myFirstInstance[1] will be created
  + resource "aws_instance" "myFirstInstance" {
      + ami                          = "ami-0db0b3ab7df22e366"
      + arn                          = (known after apply)
      + associate_public_ip_address  = (known after apply)
      + availability_zone            = (known after apply)
      + cpu_core_count               = (known after apply)
      + cpu_threads_per_core         = (known after apply)
      + get_password_data            = false
      + host_id                      = (known after apply)
      + id                           = (known after apply)
      + instance_state               = (known after apply)
      + instance_type                = "t2.micro"
      + ipv6_address_count           = (known after apply)
      + ipv6_addresses               = (known after apply)
      + key_name                     = "keypair"
      + network_interface_id         = (known after apply)
      + outpost_arn                  = (known after apply)
      + password_data                = (known after apply)
      + placement_group              = (known after apply)
      + primary_network_interface_id = (known after apply)
      + private_dns                  = (known after apply)
      + private_ip                   = (known after apply)
      + public_dns                   = (known after apply)
      + public_ip                    = (known after apply)
      + security_groups              = [
          + "security_jenkins_port",
```

Garishma Virk
R171217017
Btech CSE-DevOps
500060134

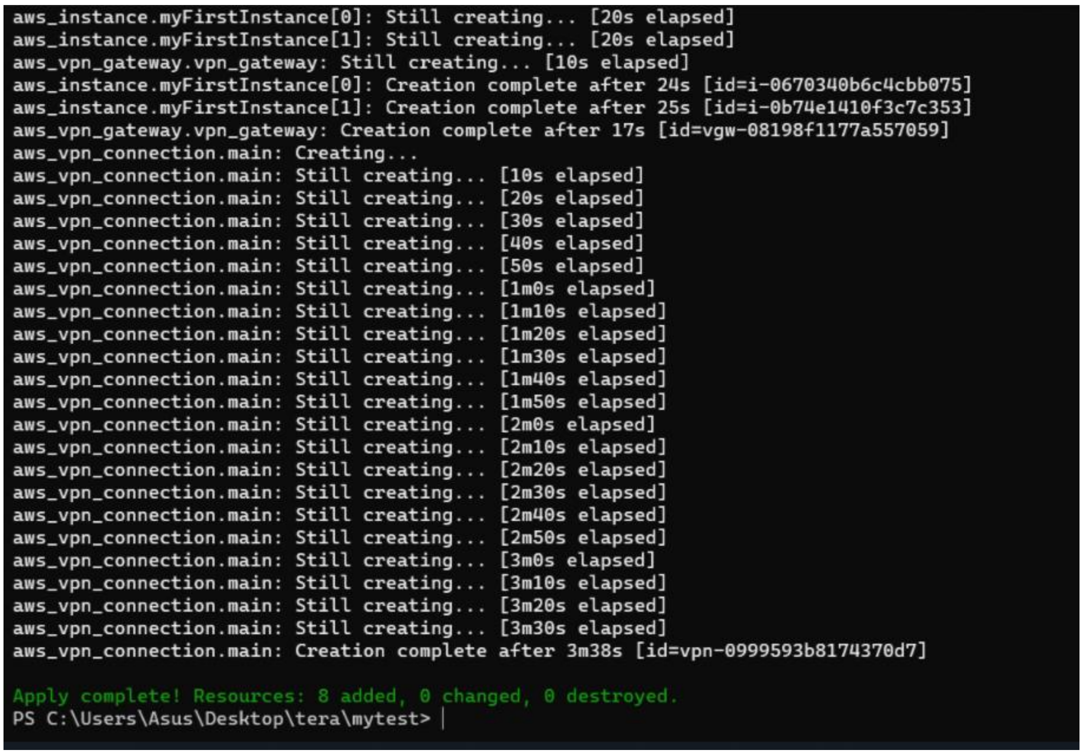


```
      ]
    + source_dest_check            = true
    + subnet_id                    = (known after apply)
    + tags                         = {
        + "Name" = "jenkins_instance"
      }
    + tenancy                      = (known after apply)
    + volume_tags                  = (known after apply)
    + vpc_security_group_ids       = (known after apply)

    + ebs_block_device {
        + delete_on_termination = (known after apply)
        + device_name           = (known after apply)
        + encrypted             = (known after apply)
        + iops                  = (known after apply)
        + kms_key_id            = (known after apply)
        + snapshot_id           = (known after apply)
        + volume_id             = (known after apply)
        + volume_size           = (known after apply)
        + volume_type           = (known after apply)
      }

    + ephemeral_block_device {
        + device_name  = (known after apply)
        + no_device    = (known after apply)
        + virtual_name = (known after apply)
      }

    + metadata_options {
        + http_endpoint               = (known after apply)
        + http_put_response_hop_limit = (known after apply)
        + http_tokens                 = (known after apply)
      }

    + network_interface {
        + delete_on_termination = (known after apply)
        + device_index          = (known after apply)
        + network_interface_id  = (known after apply)
      }
```



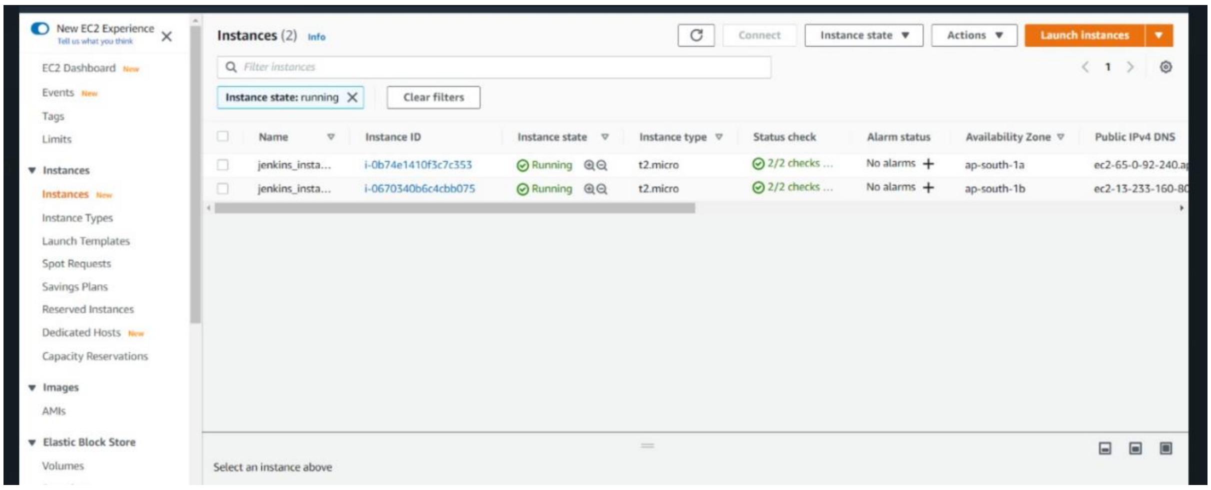
```
    + root_block_device {
        + delete_on_termination = (known after apply)
        + device_name           = (known after apply)
        + encrypted             = (known after apply)
        + iops                  = (known after apply)
        + kms_key_id            = (known after apply)
        + volume_id             = (known after apply)
        + volume_size           = (known after apply)
        + volume_type           = (known after apply)
      }
  }

# aws_s3_bucket.tf_course will be created
+ resource "aws_s3_bucket" "tf_course" {
    + acceleration_status         = (known after apply)
    + acl                         = "private"
    + arn                         = (known after apply)
    + bucket                      = "ilovedevops987"
    + bucket_domain_name          = (known after apply)
    + bucket_regional_domain_name = (known after apply)
    + force_destroy               = false
    + hosted_zone_id              = (known after apply)
    + id                          = (known after apply)
    + region                      = (known after apply)
    + request_payer               = (known after apply)
    + website_domain              = (known after apply)
    + website_endpoint            = (known after apply)

    + versioning {
        + enabled    = (known after apply)
        + mfa_delete = (known after apply)
      }
  }

# aws_security_group.security_jenkins_port will be created
+ resource "aws_security_group" "security_jenkins_port" {
    + arn                    = (known after apply)
    + description            = "security group for jenkins"
    + egress                 = [
```

Garishma Virk
R171217017
Btech CSE-DevOps
500060134



```
aws_instance.myFirstInstance[0]: Still creating... [20s elapsed]
aws_instance.myFirstInstance[1]: Still creating... [20s elapsed]
aws_vpn_gateway.vpn_gateway: Still creating... [10s elapsed]
aws_instance.myFirstInstance[0]: Creation complete after 24s [id=i-0670340b6c4cbb075]
aws_instance.myFirstInstance[1]: Creation complete after 25s [id=i-0b74e1410f3c7c353]
aws_vpn_gateway.vpn_gateway: Creation complete after 17s [id=vgw-08198f1177a557059]
aws_vpn_connection.main: Creating...
aws_vpn_connection.main: Still creating... [10s elapsed]
aws_vpn_connection.main: Still creating... [20s elapsed]
aws_vpn_connection.main: Still creating... [30s elapsed]
aws_vpn_connection.main: Still creating... [40s elapsed]
aws_vpn_connection.main: Still creating... [50s elapsed]
aws_vpn_connection.main: Still creating... [1m0s elapsed]
aws_vpn_connection.main: Still creating... [1m10s elapsed]
aws_vpn_connection.main: Still creating... [1m20s elapsed]
aws_vpn_connection.main: Still creating... [1m30s elapsed]
aws_vpn_connection.main: Still creating... [1m40s elapsed]
aws_vpn_connection.main: Still creating... [1m50s elapsed]
aws_vpn_connection.main: Still creating... [2m0s elapsed]
aws_vpn_connection.main: Still creating... [2m10s elapsed]
aws_vpn_connection.main: Still creating... [2m20s elapsed]
aws_vpn_connection.main: Still creating... [2m30s elapsed]
aws_vpn_connection.main: Still creating... [2m40s elapsed]
aws_vpn_connection.main: Still creating... [2m50s elapsed]
aws_vpn_connection.main: Still creating... [3m0s elapsed]
aws_vpn_connection.main: Still creating... [3m10s elapsed]
aws_vpn_connection.main: Still creating... [3m20s elapsed]
aws_vpn_connection.main: Still creating... [3m30s elapsed]
aws_vpn_connection.main: Creation complete after 3m38s [id=vpn-0999593b8174370d7]

Apply complete! Resources: 8 added, 0 changed, 0 destroyed.
PS C:\Users\Asus\Desktop\tera\mytest> |
```

Now you can check the instances, VPN and S3 bucket have been created on your AWS cloud.

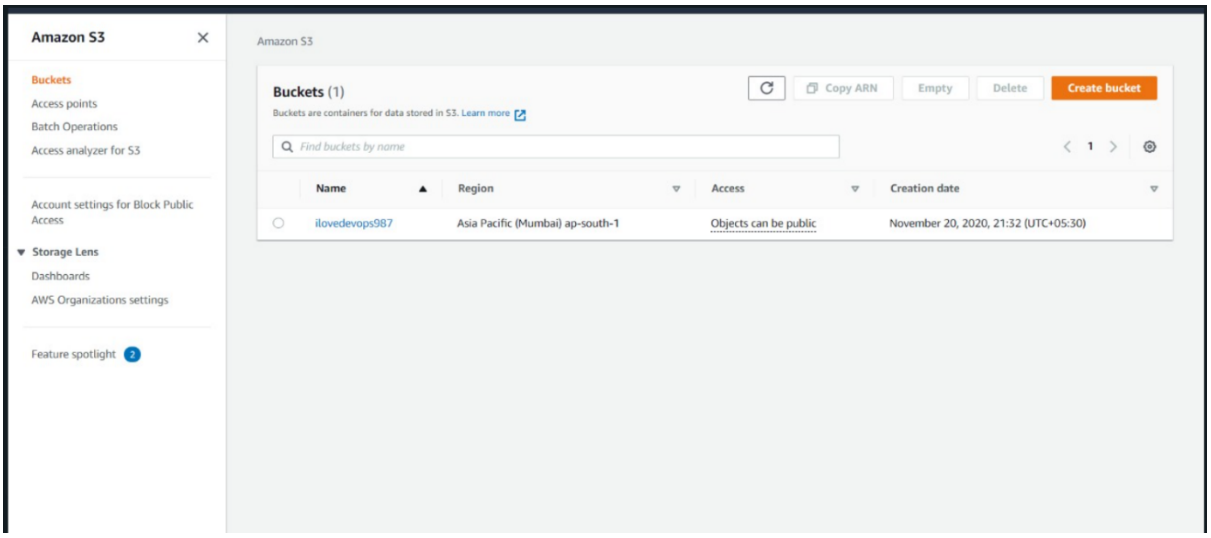


Instances (2) Info

Instance state: running

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
jenkins_insta...	i-0b74e1410f3c7c353	Running	t2.micro	2/2 checks ...	No alarms	ap-south-1a	ec2-65-0-92-240.a
jenkins_insta...	i-0670340b6c4cbb075	Running	t2.micro	2/2 checks ...	No alarms	ap-south-1b	ec2-13-233-160-8

Garishma Virk
R171217017
Btech CSE-DevOps
500060134

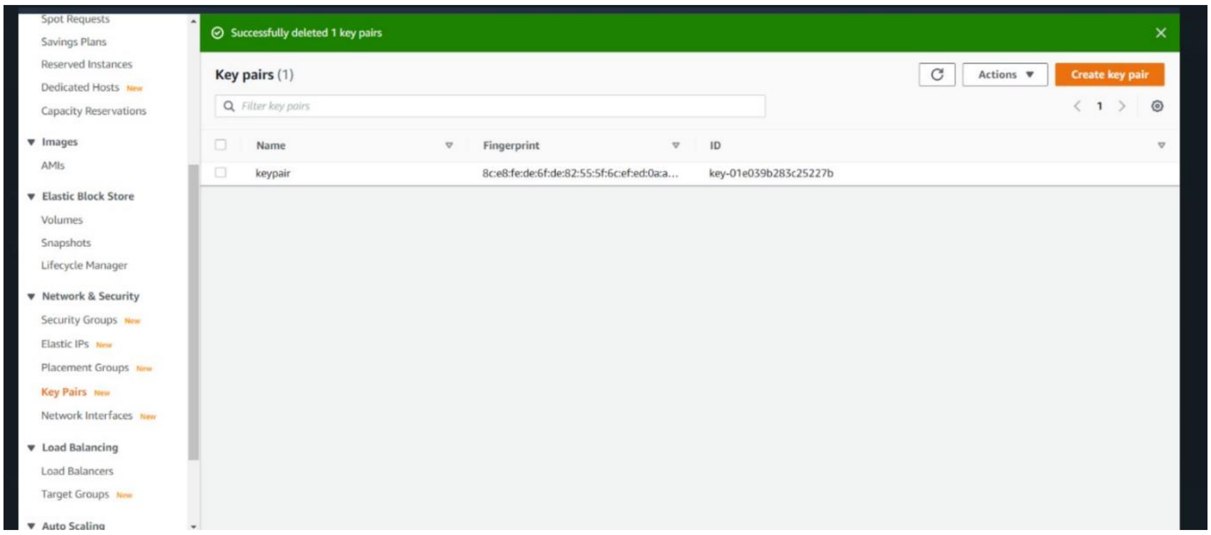


Amazon S3

Buckets (1)

Buckets are containers for data stored in S3. Learn more

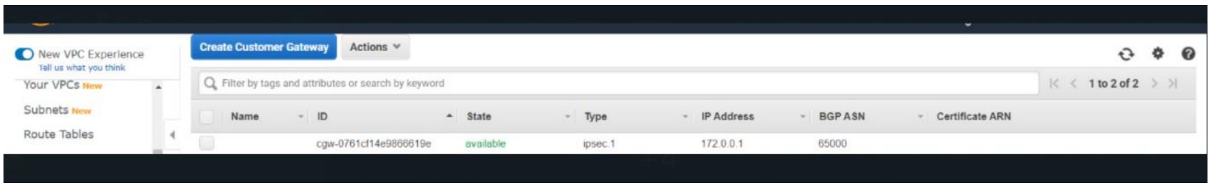
Name	Region	Access	Creation date
ilovedevops987	Asia Pacific (Mumbai) ap-south-1	Objects can be public	November 20, 2020, 21:32 (UTC+05:30)



Successfully deleted 1 key pairs

Key pairs (1)

Name	Fingerprint	ID
keypair	8c:e8:fe:de:6f:de:82:55:5f:6c:ef:ed:0a:a...	key-01e039b283c25227b



Name	ID	State	Type	IP Address	BGP ASN	Certificate ARN
	cgw-0761cf14e9866619e	available	ipsec.1	172.0.0.1	65000	

So, the services are running through scripts.

We can destroy all the resources which are no longer required with command : terraform destroy