System Provisioning & Configuration Management

ASSIGNMENT-1

SUBMITTED BY:

NAME: Anurag Negi

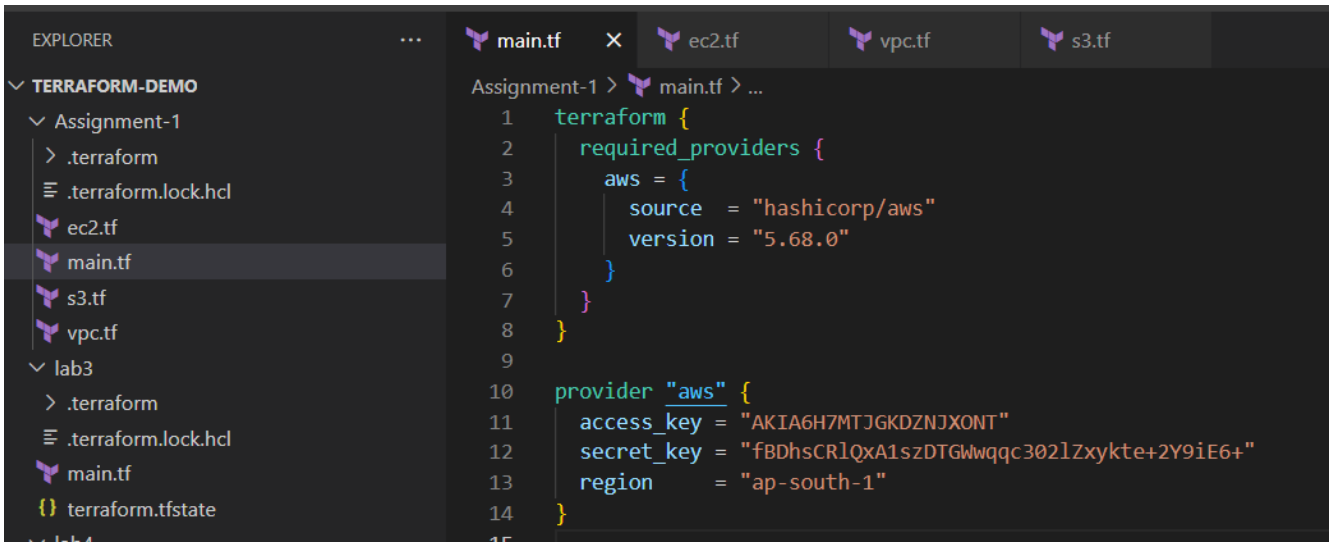
SAP ID: 500110161

BATCH: 2 (DevOps)

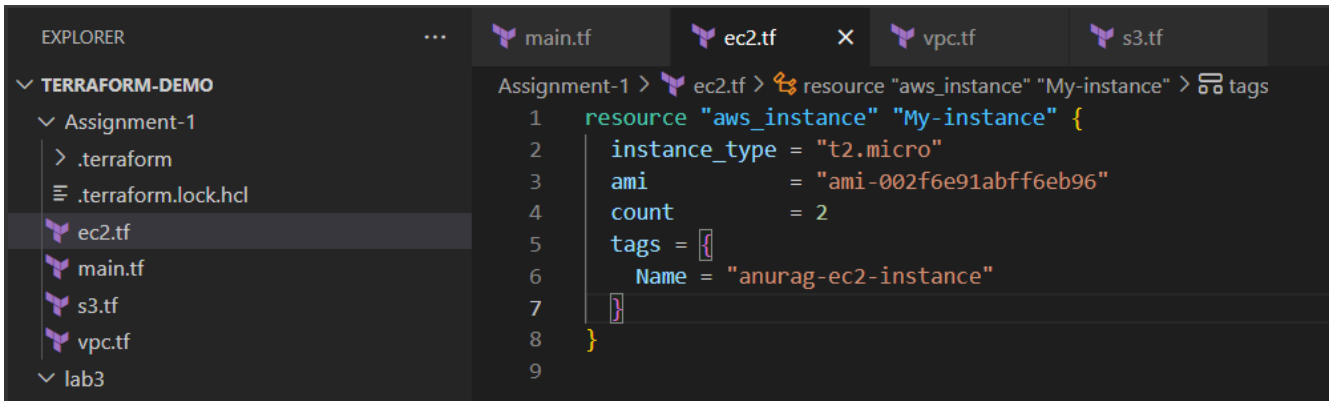
ROLL NO: R2142221383

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

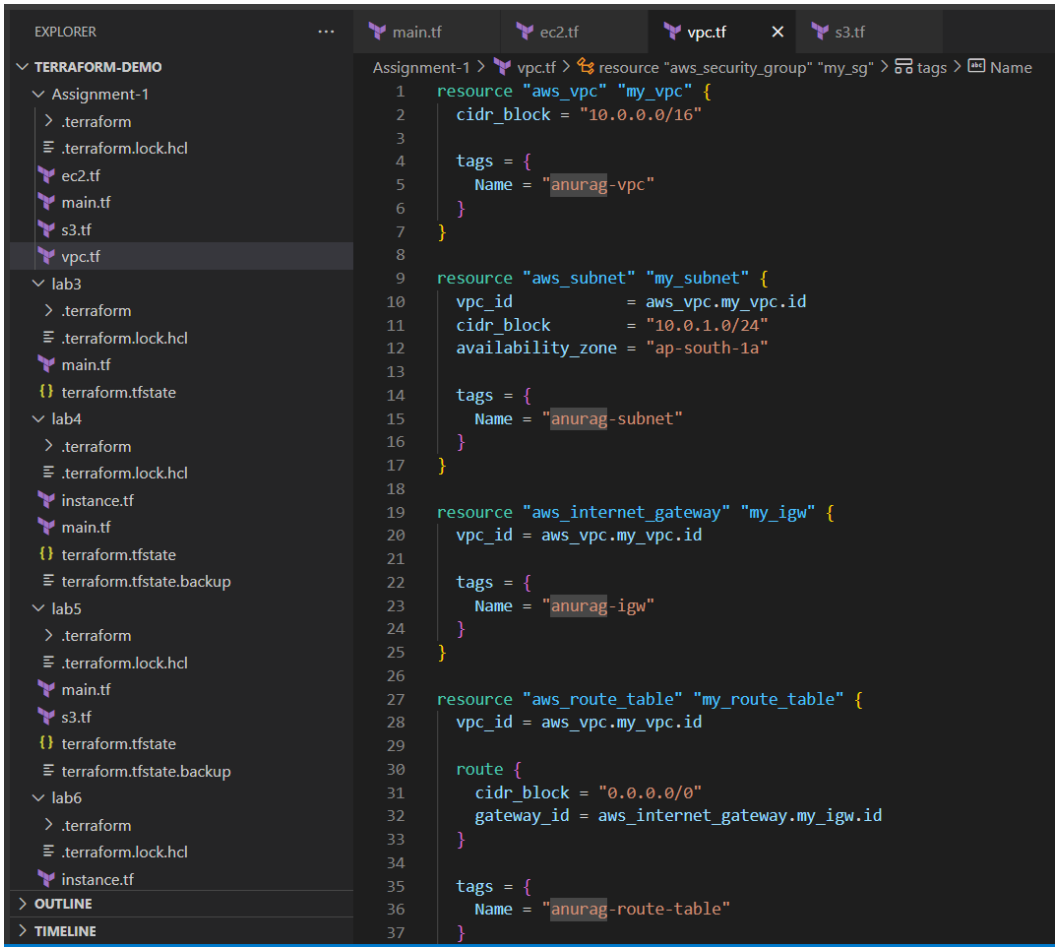
Step 1: Create Terraform Configuration File (main.tf):

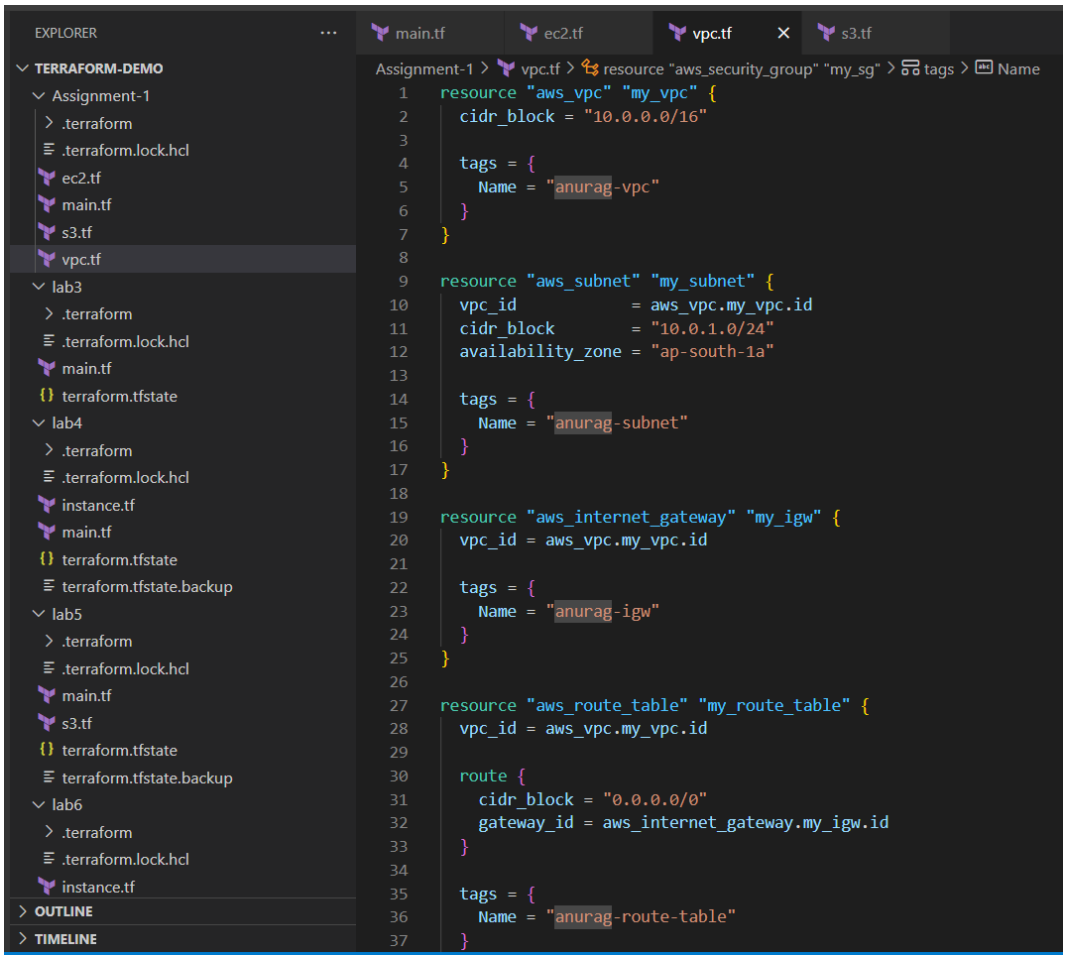


Step 2: To create two T2 Micro EC2 Instances create Ec2.tf :

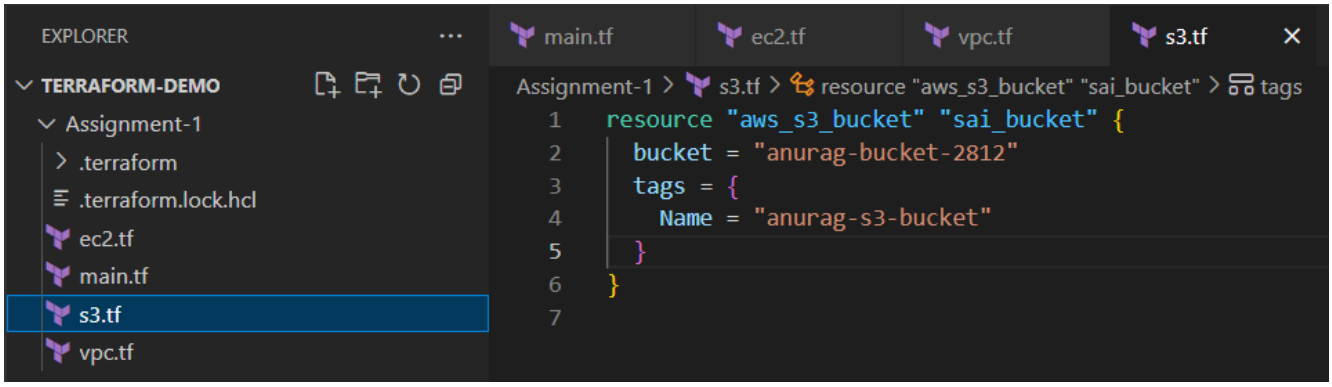


Step 3: To Create a VPC on AWS create VPC.tf :





Step 4: To Create a S3 Bucket create S3.tf :



Step 5: Initialize Terraform

Run the following command to initialize your Terraform working directory:

Terraform init

```
Terraform has been successfully initialized!

Terraform has been successfully initialized!
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
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.
```

Step 6: Review Plan

Run the following command to see what Terraform will do:

Terraform plan

```
PS C:\Github_Repositores\Terraform-Demo\Assignment-1> terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are
indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.My-instance[0] will be created
  + resource "aws_instance" "My-instance" {
      + ami                                  = "ami-002f6e91abff6eb96"
      + 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)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (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                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_partition_number           = (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)
```

```
  # aws_vpc.my_vpc will be created
  + resource "aws_vpc" "my_vpc" {
      + arn                                  = (known after apply)
      + cidr_block                           = "10.0.0.0/16"
      + default_network_acl_id               = (known after apply)
      + default_route_table_id               = (known after apply)
      + default_security_group_id            = (known after apply)
      + dhcp_options_id                      = (known after apply)
      + enable_dns_hostnames                 = (known after apply)
      + enable_dns_support                   = true
      + enable_network_address_usage_metrics = (known after apply)
      + id                                   = (known after apply)
      + instance_tenancy                     = "default"
      + ipv6_association_id                  = (known after apply)
      + ipv6_cidr_block                      = (known after apply)
      + ipv6_cidr_block_network_border_group = (known after apply)
      + main_route_table_id                  = (known after apply)
      + owner_id                             = (known after apply)
      + tags                                 = {
          + "Name" = "anurag-vpc"
        }
      + tags_all                             = {
          + "Name" = "anurag-vpc"
        }
    }

Plan: 9 to add, 0 to change, 0 to destroy.
```

Step 7: Apply Changes:

Apply the changes to create the AWS resources:

Terraform apply

```
PS C:\Github_Repositores\Terraform-Demo\Assignment-1> terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  + create
```

It will ask for approval before creating, enter “yes” to continue.

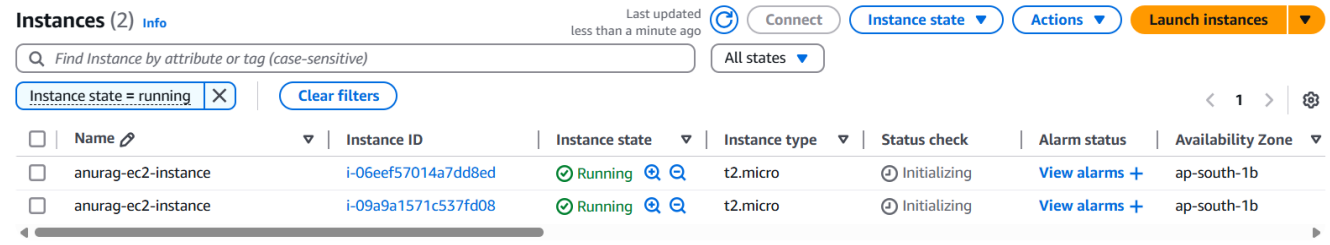
```
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes
```

After approval, it will start creating.

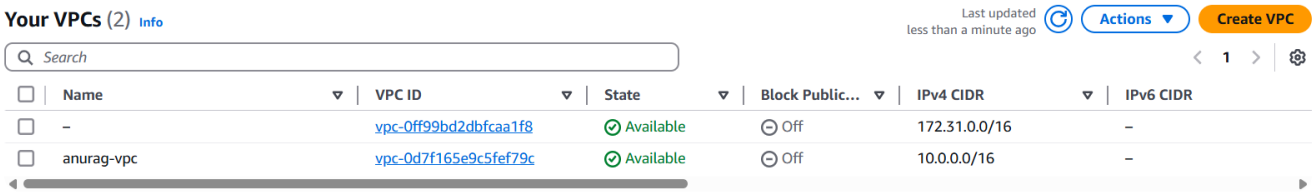
```
aws_vpc.my_vpc: Creating...
aws_instance.My-instance[1]: Creating...
aws_instance.My-instance[0]: Creating...
aws_s3_bucket.anurag_bucket: Creating...
aws_vpc.my_vpc: Creation complete after 2s [id=vpc-0d7f165e9c5fef79c]
aws_internet_gateway.my_igw: Creating...
aws_subnet.my_subnet: Creating...
aws_security_group.my_sg: Creating...
aws_s3_bucket.anurag_bucket: Creation complete after 2s [id=anurag-bucket-2812]
aws_internet_gateway.my_igw: Creation complete after 1s [id=igw-0d2ce55bd8e6249c6]
aws_route_table.my_route_table: Creating...
aws_subnet.my_subnet: Creation complete after 1s [id=subnet-03f097b216ba3db3b]
aws_route_table.my_route_table: Creation complete after 1s [id=rtb-0e6eabcf99b1c6c33]
aws_route_table_association.my_route_assoc: Creating...
aws_route_table_association.my_route_assoc: Creation complete after 0s [id=rtbassoc-08bb4914a78ce95c9]
aws_security_group.my_sg: Creation complete after 3s [id=sg-0a822a00307c64dab]
aws_instance.My-instance[1]: Still creating... [10s elapsed]
aws_instance.My-instance[0]: Still creating... [10s elapsed]
aws_instance.My-instance[1]: Creation complete after 13s [id=i-09a9a1571c537fd08]
aws_instance.My-instance[0]: Creation complete after 13s [id=i-06eef57014a7dd8ed]

Apply complete! Resources: 9 added, 0 changed, 0 destroyed.
PS C:\Github_Repositores\Terraform-Demo\Assignment-1>
```

You can verify by logging into the AWS Console,

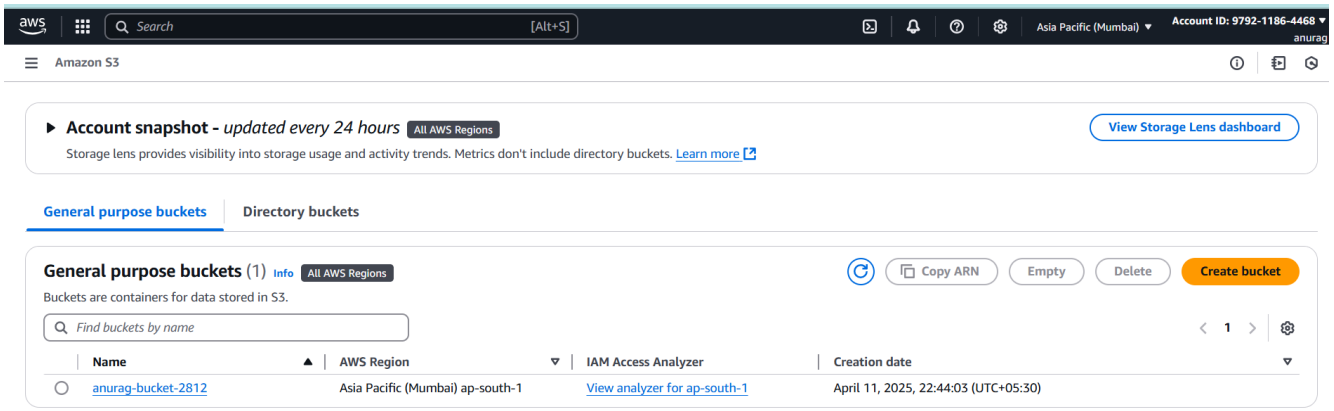
- Creation of ec2 instances



- Creation of a VPC



- Creation of S3 Bucket



Step 8: Cleanup Resources

When you are done experimenting, run the following command to destroy the created resources:

Terraform destroy

```
PS C:\Github_Repositores\Terraform-Demo\Assignment-1> terraform destroy
aws_vpc.my_vpc: Refreshing state... [id=vpc-0d7f165e9c5fef79c]
aws_s3_bucket.anurag_bucket: Refreshing state... [id=anurag-bucket-2812]
aws_instance.My-instance[1]: Refreshing state... [id=i-09a9a1571c537fd08]
aws_instance.My-instance[0]: Refreshing state... [id=i-06eef57014a7dd8ed]
aws_internet_gateway.my_igw: Refreshing state... [id=igw-0d2ce55bd8e6249c6]
aws_subnet.my_subnet: Refreshing state... [id=subnet-03f097b216ba3db3b]
aws_security_group.my_sg: Refreshing state... [id=sg-0a822a00307c64dab]
aws_route_table.my_route_table: Refreshing state... [id=rtb-0e6eabcf99b1c6c33]
aws_route_table_association.my_route_assoc: Refreshing state... [id=rtbassoc-08bb4914a78ce95c9]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  - destroy
```

It will ask for approval before destroying, enter “yes” to continue.

```
Plan: 0 to add, 0 to change, 9 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes
```

After approval, it will start destroying.