System Provisioning and Configuration Management

Submitted by:

NAME: Tanishqua Tellakula

SAP ID: 500084269

Roll number: R120220007

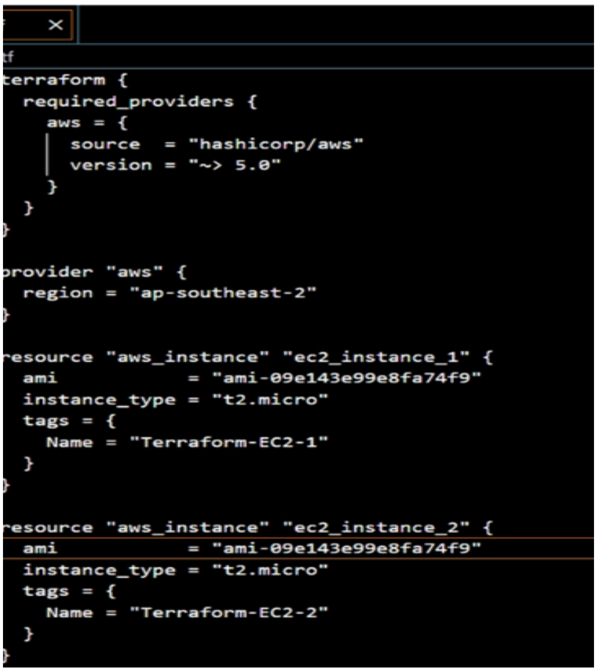
Batch: B1(Hons)

Assignment - 1

Write Terraform script to do perform following tasks on AWS cloud

Platform

Step 1: Create two T2 Micro EC2 Instances.



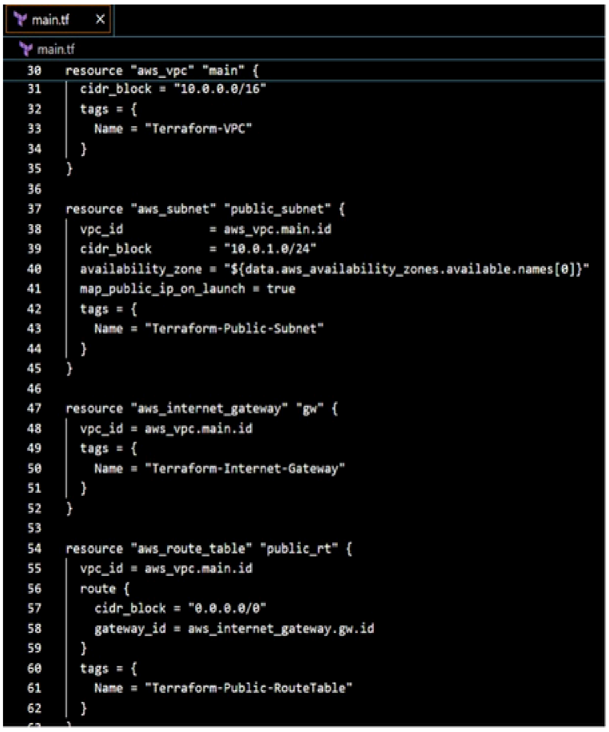
```
tf
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}

provider "aws" {
  region = "ap-southeast-2"
}

resource "aws_instance" "ec2_instance_1" {
  ami           = "ami-09e143e99e8fa74f9"
  instance_type = "t2.micro"
  tags = {
    Name = "Terraform-EC2-1"
  }
}

resource "aws_instance" "ec2_instance_2" {
  ami           = "ami-09e143e99e8fa74f9"
  instance_type = "t2.micro"
  tags = {
    Name = "Terraform-EC2-2"
  }
}
```

Step2: Create a VPN on AWS



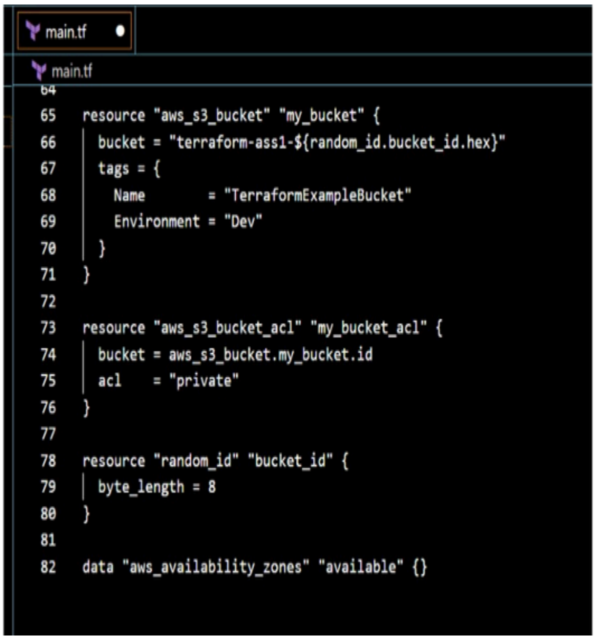
```
main.tf
resource "aws_vpc" "main" {
  cidr_block = "10.0.0.0/16"
  tags = {
    Name = "Terraform-VPC"
  }
}

resource "aws_subnet" "public_subnet" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.1.0/24"
  availability_zone = "${data.aws_availability_zones.available.names[0]}"
  map_public_ip_on_launch = true
  tags = {
    Name = "Terraform-Public-Subnet"
  }
}

resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.main.id
  tags = {
    Name = "Terraform-Internet-Gateway"
  }
}

resource "aws_route_table" "public_rt" {
  vpc_id = aws_vpc.main.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.gw.id
  }
  tags = {
    Name = "Terraform-Public-RouteTable"
  }
```

Step 3: Create a S3 Bucket



```
resource "aws_s3_bucket" "my_bucket" {
  bucket = "terraform-ass1-${random_id.bucket_id.hex}"
  tags = {
    Name        = "TerraformExampleBucket"
    Environment = "Dev"
  }
}

resource "aws_s3_bucket_acl" "my_bucket_acl" {
  bucket = aws_s3_bucket.my_bucket.id
  acl    = "private"
}

resource "random_id" "bucket_id" {
  byte_length = 8
}

data "aws_availability_zones" "available" {}
```

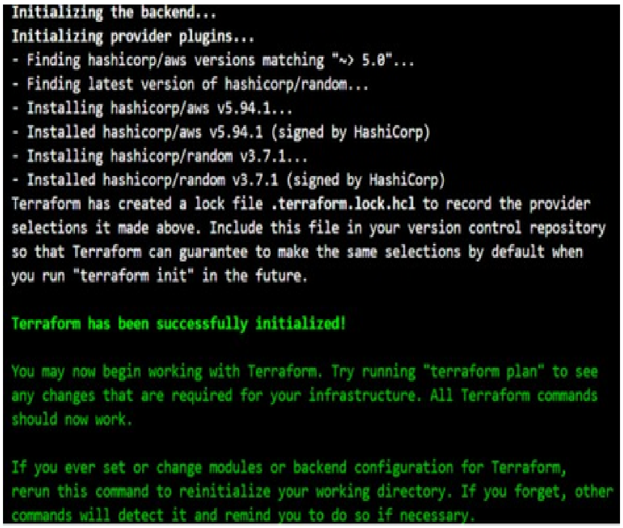
Step 4: Write the code for step 1,2 and 3 in a IaC terraform file and

run terraform commands to execute these steps.

- *Codes for step 1, step 2 and step 3 : These codes will create all*

the three AWS entities in one go......

Terraform init:



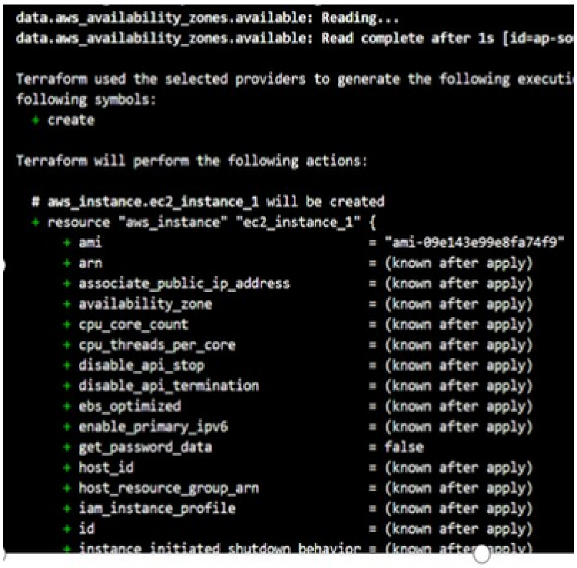
```
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Finding latest version of hashicorp/random...
- Installing hashicorp/aws v5.94.1...
- Installed hashicorp/aws v5.94.1 (signed by HashiCorp)
- Installing hashicorp/random v3.7.1...
- Installed hashicorp/random v3.7.1 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

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:

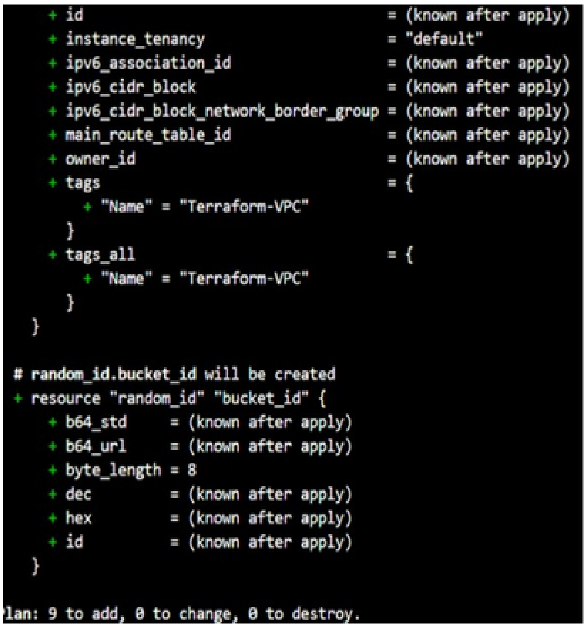


```
data.aws_availability_zones.available: Reading...
data.aws_availability_zones.available: Read complete after 1s [id=ap-so

Terraform used the selected providers to generate the following executi
following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.ec2_instance_1 will be created
  + resource "aws_instance" "ec2_instance_1" {
      + ami                                  = "ami-09e143e99e8fa74f9"
      + 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)
      + enable_primary_ipv6                  = (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)
```

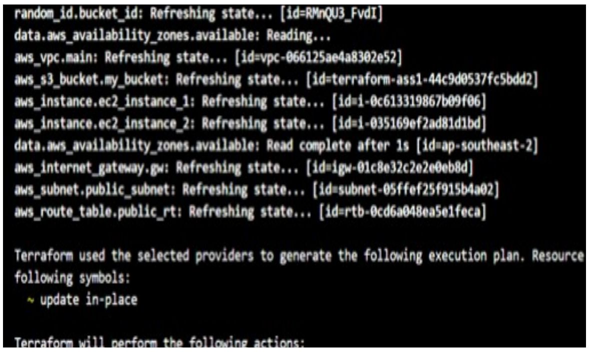


```
    + id                                   = (known after apply)
    + instance_tenancy                     = "default"
    + ipv6_association_id                  = (known after apply)
    + ipv6_cidr_block                      = (known after apply)
    + ipv6_cidr_block_network_border_group = (known after apply)
    + main_route_table_id                  = (known after apply)
    + owner_id                             = (known after apply)
    + tags                                 = {
        + "Name" = "Terraform-VPC"
      }
    + tags_all                             = {
        + "Name" = "Terraform-VPC"
      }
  }

# random_id.bucket_id will be created
+ resource "random_id" "bucket_id" {
    + b64_std     = (known after apply)
    + b64_url     = (known after apply)
    + byte_length = 8
    + dec         = (known after apply)
    + hex         = (known after apply)
    + id          = (known after apply)
  }

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

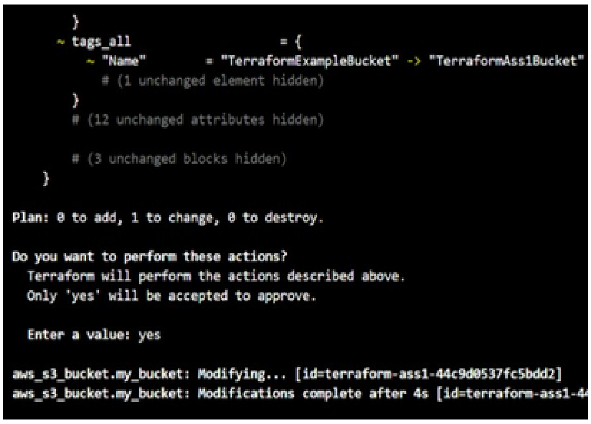
Terraform apply:



```
random_id.bucket_id: Refreshing state... [id=RMnQU3_FvdI]
data.aws_availability_zones.available: Reading...
aws_vpc.main: Refreshing state... [id=vpc-066125ae4a8302e52]
aws_s3_bucket.my_bucket: Refreshing state... [id=terraform-ass1-44c9d0537fc5bdd2]
aws_instance.ec2_instance_1: Refreshing state... [id=i-0c613319867b09f06]
aws_instance.ec2_instance_2: Refreshing state... [id=i-035169ef2ad81d1bd]
data.aws_availability_zones.available: Read complete after 1s [id=ap-southeast-2]
aws_internet_gateway.gw: Refreshing state... [id=igw-01c8e32c2e2e0eb8d]
aws_subnet.public_subnet: Refreshing state... [id=subnet-05ffef25f915b4a02]
aws_route_table.public_rt: Refreshing state... [id=rtb-0cd6a048ea5e1feca]

Terraform used the selected providers to generate the following execution plan. Resource
following symbols:
  ~ update in-place

Terraform will perform the following actions:
```



```
        }
      ~ tags_all                    = {
          ~ "Name"        = "TerraformExampleBucket" -> "TerraformAss1Bucket"
            # (1 unchanged element hidden)
        }
        # (12 unchanged attributes hidden)

        # (3 unchanged blocks hidden)
    }

Plan: 0 to add, 1 to change, 0 to destroy.

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

aws_s3_bucket.my_bucket: Modifying... [id=terraform-ass1-44c9d0537fc5bdd2]
aws_s3_bucket.my_bucket: Modifications complete after 4s [id=terraform-ass1-44
```