System Provisioning and Configuration Module Lab

Assignment 1

Under the Guidance of: Dr. Hitesh Kumar Sharma

Submitted by – Vaibhav Singh

SAP ID - 500105558

Roll No - R2142220995

Batch - DevOps B2(N.H.)

1. **Create two T2 Micro EC2 Instances. This instance.tf file contains the Iac code to create two instances of type t2.micro and ami of ubuntu.**

instance.tf

```
# Public Subnet for VPN Endpoint (ins1)
resource "aws_subnet" "public" {
  vpc_id                  = aws_vpc.main.id
  cidr_block              = "10.0.1.0/24"
  availability_zone       = "ap-south-1a"
  map_public_ip_on_launch = true
  tags = {
    Name = "Sid_Public_Subnet"
  }
}

# Security Group for VPN Endpoint
resource "aws_security_group" "vpn_sg" {
  name        = "vpn_endpoint_sg"
  description = "Allow IPSec VPN traffic"
  vpc_id      = aws_vpc.main.id

  ingress {
    description = "IKE (UDP 500)"
    from_port   = 500
    to_port     = 500
    protocol    = "udp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  ingress {
    description = "NAT-T (UDP 4500)"
    from_port   = 4500
    to_port     = 4500
    protocol    = "udp"
    cidr_blocks = ["0.0.0.0/0"]
```

instance.tf

```
13  resource "aws_security_group" "vpn_sg" {
33
34    ingress {
35      description = "SSH"
36      from_port   = 22
37      to_port     = 22
38      protocol    = "tcp"
39      cidr_blocks = ["0.0.0.0/0"]
40    }
41
42    egress {
43      from_port   = 0
44      to_port     = 0
45      protocol    = "-1"
46      cidr_blocks = ["0.0.0.0/0"]
47    }
48
49    tags = {
50      Name = "Sid_VPN_SecurityGroup"
51    }
52  }
53
54  # VPN Endpoint Instance
55  resource "aws_instance" "ins1" {
56    ami                    = "ami-0e35ddab05955cf57" # Ubuntu 24.04
57    instance_type          = "t2.micro"
58    subnet_id              = aws_subnet.public.id
59    vpc_security_group_ids = [aws_security_group.vpn_sg.id]
60    tags = {
61      Name = "Sid_VPN_Endpoint"
62    }
```

```
65  # Regular Instance (Optional)
66  resource "aws_instance" "ins2" {
67    ami           = "ami-0e35ddab05955cf57"
68    instance_type = "t2.micro"
69    subnet_id     = aws_subnet.public.id # Can change to private subnet if needed
70    tags = {
71      Name = "Sid_Instance2"
72    }
73  }
74
75  # Elastic IP for VPN Endpoint
76  resource "aws_eip" "vpn_eip" {
77    instance = aws_instance.ins1.id
78    tags = {
79      Name = "Sid_VPN_EIP"
80    }
81  }
```

2. **Create a VPN on AWS This resource.tf file contains the complete code to make a VPN. It consists of resources like vpc, customer gateway and vpn connection.**

resource.tf

```
# Main VPC
resource "aws_vpc" "main" {
  cidr_block           = "10.0.0.0/16"
  enable_dns_support   = true
  enable_dns_hostnames = true
  tags = {
    Name = "Sid_VPC"
  }
}

# Internet Gateway
resource "aws_internet_gateway" "igw" {
  vpc_id = aws_vpc.main.id
  tags = {
    Name = "Sid_IGW"
  }
}

# Route Table for Public Subnet
resource "aws_route_table" "public" {
  vpc_id = aws_vpc.main.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.igw.id
  }

  tags = {
    Name = "Sid_Public_RT"
  }
}
```

resource.tf

```
# Route Table Association
resource "aws_route_table_association" "public" {
  subnet_id      = aws_subnet.public.id
  route_table_id = aws_route_table.public.id
}

# VPN Gateway
resource "aws_vpn_gateway" "vpn_gw" {
  vpc_id = aws_vpc.main.id
  tags = {
    Name = "Sid_VPN_Gateway"
  }
}

# Customer Gateway (Using EC2's EIP)
resource "aws_customer_gateway" "cgw" {
  bgp_asn    = 65000
  ip_address = aws_eip.vpn_eip.public_ip
  type       = "ipsec.1"
  tags = {
    Name = "Sid_Customer_Gateway"
  }
}
```

```
# VPN Connection
resource "aws_vpn_connection" "main" {
  vpn_gateway_id      = aws_vpn_gateway.vpn_gw.id
  customer_gateway_id = aws_customer_gateway.cgw.id
  type                = "ipsec.1"
  static_routes_only  = true
  tags = {
    Name = "Sid_VPN_Connection"
  }
}
```

3. **Create a S3 Bucket Code to create a s3 bucket.**

s3.tf

```
resource "aws_s3_bucket" "assignment_bucket" {
  bucket = "r2142220666"
  tags = {
    Name        = "Sid_Assignment_Bucket"
    Environment = "Assignment"
  }
}
```

4. **Main.tf file to perform the above-mentioned tasks**

main.tf

```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.68.0"
    }
  }
}

provider "aws" {
  access_key = "AKIA6GSNHCFAR2SWI2NN"
  secret_key = "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"
  region = "ap-south-1"
}
```

5. **Terraform init to initialize the terraform folder which will have the aws provider.**

```
PS G:\New Volume E\6th sem\System Provisioning Lab\Assignment-1> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.68.0"...
- Installing hashicorp/aws v5.68.0...
- Installed hashicorp/aws v5.68.0 (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.
```

6. **Terraform plan to see the resources that will be created.**

```
PS G:\New Volume E\6th sem\System Provisioning Lab\Assignment-1> terraform plan
aws_vpc.main: Refreshing state... [id=vpc-04ff9b57c16aadb1f]
aws_s3_bucket.assignment_bucket: Refreshing state... [id=r2142220666]
aws_internet_gateway.igw: Refreshing state... [id=igw-0e2e8738c0ed9ca43]
aws_vpn_gateway.vpn_gw: Refreshing state... [id=vgw-04e7657a0495cd1b1]
aws_subnet.public: Refreshing state... [id=subnet-0e26a936205a4d00d]
aws_security_group.vpn_sg: Refreshing state... [id=sg-039725890fd2770ce]
aws_route_table.public: Refreshing state... [id=rtb-0d67b64b84661baff]
aws_instance.ins2: Refreshing state... [id=i-0436d3c5e1c951f53]
aws_instance.ins1: Refreshing state... [id=i-0c7ffb55b094abc44]
aws_route_table_association.public: Refreshing state... [id=rtbassoc-0d7c070f171a7bf8f]
aws_eip.vpn_eip: Refreshing state... [id=eipalloc-0b7533407a3bdfeed]
aws_customer_gateway.cgw: Refreshing state... [id=cgw-0c9a8b7812ecf9660]
aws_vpn_connection.main: Refreshing state... [id=vpn-08f2ede948ccd26ac]

No changes. Your infrastructure matches the configuration.
```

P.s. - I performed terraform plan after terraform apply so the resources were actually created. In practice, it is advised to perform terraform plan before terraform apply to see what resources will be created.

7. **Terraform apply to create the mentioned resources.**

```
PS G:\New Volume E\6th sem\System Provisioning Lab\Assignment-1> terraform apply

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_customer_gateway.cgw will be created
  + resource "aws_customer_gateway" "cgw" {
      + arn        = (known after apply)
      + bgp_asn    = "65000"
      + id         = (known after apply)
      + ip_address = (known after apply)
      + tags       = {
          + "Name" = "Sid_Customer_Gateway"
        }
      + tags_all   = {
          + "Name" = "Sid_Customer_Gateway"
        }
      + type       = "ipsec.1"
    }

  # aws_eip.vpn_eip will be created
  + resource "aws_eip" "vpn_eip" {
      + allocation_id        = (known after apply)
      + arn                  = (known after apply)
      + association_id       = (known after apply)
      + carrier_ip           = (known after apply)
      + customer_owned_ip    = (known after apply)
      + domain               = (known after apply)
      + id                   = (known after apply)
      + instance             = (known after apply)
      + network_border_group = (known after apply)
```

```
    + network_interface    = (known after apply)
    + private_dns          = (known after apply)
    + private_ip           = (known after apply)
    + ptr_record           = (known after apply)
    + public_dns           = (known after apply)
    + public_ip            = (known after apply)
    + public_ipv4_pool     = (known after apply)
    + tags                 = {
        + "Name" = "Sid_VPN_EIP"
      }
    + tags_all             = {
        + "Name" = "Sid_VPN_EIP"
      }
    + vpc                  = (known after apply)
  }

# aws_instance.ins1 will be created
+ resource "aws_instance" "ins1" {
    + ami                                  = "ami-0e35ddab05955cf57"
    + 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)
    + public_ip                            = (known after apply)
    + secondary_private_ips                = (known after apply)
    + security_groups                      = (known after apply)
    + source_dest_check                    = true
    + spot_instance_request_id             = (known after apply)
    + subnet_id                            = (known after apply)
    + tags                                 = {
        + "Name" = "Sid_VPN_Endpoint"
      }
    + tags_all                             = {
        + "Name" = "Sid_VPN_Endpoint"
      }
    + tenancy                              = (known after apply)
    + user_data                            = (known after apply)
    + user_data_base64                     = (known after apply)
    + user_data_replace_on_change          = false
    + vpc_security_group_ids               = (known after apply)

    + capacity_reservation_specification (known after apply)
```

```
    + cpu_options (known after apply)

    + ebs_block_device (known after apply)

    + enclave_options (known after apply)

    + ephemeral_block_device (known after apply)

    + instance_market_options (known after apply)

    + maintenance_options (known after apply)

    + metadata_options (known after apply)

    + network_interface (known after apply)

    + private_dns_name_options (known after apply)

    + root_block_device (known after apply)
  }

# aws_instance.ins2 will be created
+ resource "aws_instance" "ins2" {
    + ami                                  = "ami-0e35ddab05955cf57"
    + 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)
    + public_ip                            = (known after apply)
    + secondary_private_ips                = (known after apply)
    + security_groups                      = (known after apply)
    + source_dest_check                    = true
    + spot_instance_request_id             = (known after apply)
    + subnet_id                            = (known after apply)
    + tags                                 = {
        + "Name" = "Sid_Instance2"
      }
    + tags_all                             = {
        + "Name" = "Sid_Instance2"
      }
    + tenancy                              = (known after apply)
    + user_data                            = (known after apply)
    + user_data_base64                     = (known after apply)
```

```
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)

      + capacity_reservation_specification (known after apply)

      + cpu_options (known after apply)

      + ebs_block_device (known after apply)

      + enclave_options (known after apply)

      + ephemeral_block_device (known after apply)

      + instance_market_options (known after apply)

      + maintenance_options (known after apply)

      + metadata_options (known after apply)

      + network_interface (known after apply)

      + private_dns_name_options (known after apply)

      + root_block_device (known after apply)
    }

  # aws_internet_gateway.igw will be created
  + resource "aws_internet_gateway" "igw" {
      + arn      = (known after apply)
      + id       = (known after apply)
      + owner_id = (known after apply)
      + tags     = {
          + "Name" = "Sid_IGW"
        }
```

```
      + tags_all = {
          + "Name" = "Sid_IGW"
        }
      + vpc_id   = (known after apply)
    }

  # aws_route_table.public will be created
  + resource "aws_route_table" "public" {
      + arn              = (known after apply)
      + id               = (known after apply)
      + owner_id         = (known after apply)
      + propagating_vgws = (known after apply)
      + route            = [
          + {
              + cidr_block                = "0.0.0.0/0"
              + gateway_id                = (known after apply)
              # (11 unchanged attributes hidden)
            },
        ]
      + tags             = {
          + "Name" = "Sid_Public_RT"
        }
      + tags_all         = {
          + "Name" = "Sid_Public_RT"
        }
      + vpc_id           = (known after apply)
    }

  # aws_route_table_association.public will be created
  + resource "aws_route_table_association" "public" {
      + id             = (known after apply)
      + route_table_id = (known after apply)
      + subnet_id      = (known after apply)
    }
```

```
# aws_s3_bucket.assignment_bucket will be created
+ resource "aws_s3_bucket" "assignment_bucket" {
    + acceleration_status         = (known after apply)
    + acl                         = (known after apply)
    + arn                         = (known after apply)
    + bucket                      = "R2142220666"
    + bucket_domain_name          = (known after apply)
    + bucket_prefix               = (known after apply)
    + bucket_regional_domain_name = (known after apply)
    + force_destroy               = false
    + hosted_zone_id              = (known after apply)
    + id                          = (known after apply)
    + object_lock_enabled         = (known after apply)
    + policy                      = (known after apply)
    + region                      = (known after apply)
    + request_payer               = (known after apply)
    + tags                        = {
        + "Environment" = "Assignment"
        + "Name"        = "Sid_Assignment_Bucket"
      }
    + tags_all                    = {
        + "Environment" = "Assignment"
        + "Name"        = "Sid_Assignment_Bucket"
      }
    + website_domain              = (known after apply)
    + website_endpoint            = (known after apply)

    + cors_rule (known after apply)

    + grant (known after apply)

    + lifecycle_rule (known after apply)

    + logging (known after apply)
```

```
# aws_subnet.public will be created
+ resource "aws_subnet" "public" {
    + arn                                            = (known after apply)
    + assign_ipv6_address_on_creation                = false
    + availability_zone                              = "ap-south-1a"
    + availability_zone_id                           = (known after apply)
    + cidr_block                                     = "10.0.1.0/24"
    + enable_dns64                                   = false
    + enable_resource_name_dns_a_record_on_launch    = false
    + enable_resource_name_dns_aaaa_record_on_launch = false
    + id                                             = (known after apply)
    + ipv6_cidr_block_association_id                 = (known after apply)
    + ipv6_native                                    = false
    + map_public_ip_on_launch                        = true
    + owner_id                                       = (known after apply)
    + private_dns_hostname_type_on_launch            = (known after apply)
    + tags                                           = {
        + "Name" = "Sid_Public_Subnet"
      }
    + tags_all                                       = {
        + "Name" = "Sid_Public_Subnet"
      }
    + vpc_id                                         = (known after apply)
  }

# aws_vpc.main will be created
+ resource "aws_vpc" "main" {
    + 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               = true
    + 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)
```

```
  # aws_vpn_connection.main will be created
  + resource "aws_vpn_connection" "main" {
      + arn                              = (known after apply)
      + core_network_arn                 = (known after apply)
      + core_network_attachment_arn      = (known after apply)
      + customer_gateway_configuration   = (sensitive value)
      + customer_gateway_id              = (known after apply)
      + enable_acceleration              = (known after apply)
      + id                               = (known after apply)
      + local_ipv4_network_cidr          = (known after apply)
      + local_ipv6_network_cidr          = (known after apply)
      + outside_ip_address_type          = (known after apply)
      + remote_ipv4_network_cidr         = (known after apply)
      + remote_ipv6_network_cidr         = (known after apply)
      + routes                           = (known after apply)
      + static_routes_only               = true
      + tags                             = {
          + "Name" = "Sid_VPN_Connection"
        }
      + tags_all                         = {
          + "Name" = "Sid_VPN_Connection"
        }
      + transit_gateway_attachment_id    = (known after apply)
      + tunnel1_address                  = (known after apply)
      + tunnel1_bgp_asn                  = (known after apply)
      + tunnel1_bgp_holdtime             = (known after apply)
      + tunnel1_cgw_inside_address       = (known after apply)
      + tunnel1_inside_cidr              = (known after apply)
      + tunnel1_inside_ipv6_cidr         = (known after apply)
      + tunnel1_preshared_key            = (sensitive value)
      + tunnel1_vgw_inside_address       = (known after apply)
      + tunnel2_address                  = (known after apply)
      + tunnel2_bgp_asn                  = (known after apply)
      + tunnel2_bgp_holdtime             = (known after apply)
      + tunnel2_cgw_inside_address       = (known after apply)
```

```
  # aws_vpn_gateway.vpn_gw will be created
  + resource "aws_vpn_gateway" "vpn_gw" {
      + amazon_side_asn = (known after apply)
      + arn             = (known after apply)
      + id              = (known after apply)
      + tags            = {
          + "Name" = "Sid_VPN_Gateway"
        }
      + tags_all        = {
          + "Name" = "Sid_VPN_Gateway"
        }
      + vpc_id          = (known after apply)
    }

Plan: 13 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + s3_bucket_name     = "R2142220666"
  + vpn_connection_id  = (known after apply)
  + vpn_endpoint_ip    = (known after apply)
  + vpn_tunnel_details = (known after apply)

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_vpc.main: Creating...
aws_s3_bucket.assignment_bucket: Creating...
aws_vpc.main: Still creating... [10s elapsed]
aws_vpc.main: Creation complete after 12s [id=vpc-04ff9b57c16aadb1f]
aws_internet_gateway.igw: Creating...
aws_vpn_gateway.vpn_gw: Creating...
aws_subnet.public: Creating...
aws_security_group.vpn_sg: Creating...
aws_internet_gateway.igw: Creation complete after 0s [id=igw-0e2e8738c0ed9ca43]
aws_route_table.public: Creating...
aws_route_table.public: Creation complete after 1s [id=rtb-0d67b64b84661baff]
aws_security_group.vpn_sg: Creation complete after 2s [id=sg-039725890fd2770ce]
aws_vpn_gateway.vpn_gw: Still creating... [10s elapsed]
aws_subnet.public: Still creating... [10s elapsed]
```

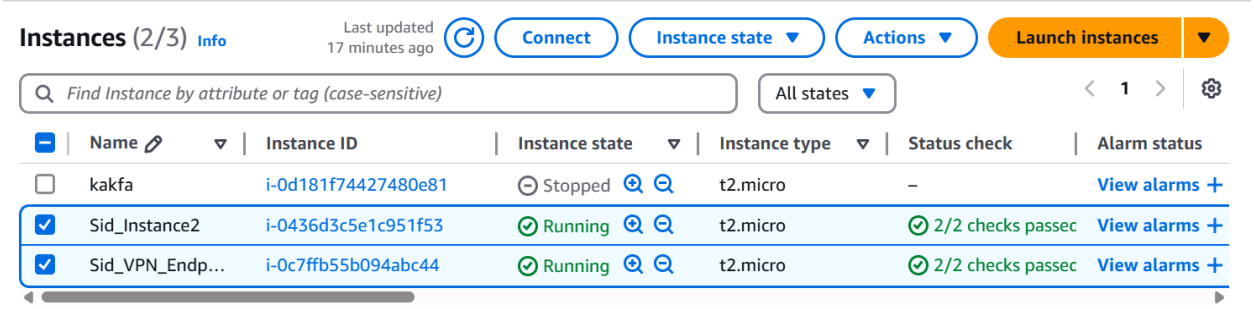
```
aws_route_table.public: Creation complete after 1s [id=rtb-0d67b64b84661baff]
aws_security_group.vpn_sg: Creation complete after 2s [id=sg-039725890fd2770ce]
aws_vpn_gateway.vpn_gw: Still creating... [10s elapsed]
aws_subnet.public: Still creating... [10s elapsed]
aws_subnet.public: Creation complete after 11s [id=subnet-0e26a936205a4d00d]
aws_route_table_association.public: Creating...
aws_instance.ins2: Creating...
aws_instance.ins1: Creating...
aws_route_table_association.public: Creation complete after 0s [id=rtbassoc-0d7c070f171a7bf8f]
aws_vpn_gateway.vpn_gw: Still creating... [20s elapsed]
aws_instance.ins2: Still creating... [10s elapsed]
aws_instance.ins1: Still creating... [10s elapsed]
aws_instance.ins1: Creation complete after 13s [id=i-0c7ffb55b094abc44]
aws_eip.vpn_eip: Creating...
aws_instance.ins2: Creation complete after 13s [id=i-0436d3c5e1c951f53]
aws_eip.vpn_eip: Creation complete after 1s [id=eipalloc-0b7533407a3bdfeed]
aws_customer_gateway.cgw: Creating...
aws_vpn_gateway.vpn_gw: Still creating... [30s elapsed]
aws_customer_gateway.cgw: Still creating... [10s elapsed]
aws_customer_gateway.cgw: Creation complete after 11s [id=cgw-0c9a8b7812ecf9660]
aws_vpn_gateway.vpn_gw: Still creating... [40s elapsed]
aws_vpn_gateway.vpn_gw: Creation complete after 44s [id=vgw-04e7657a0495cd1b1]
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... [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: Creation complete after 3m16s [id=vpn-08f2ede948ccd26ac]
```

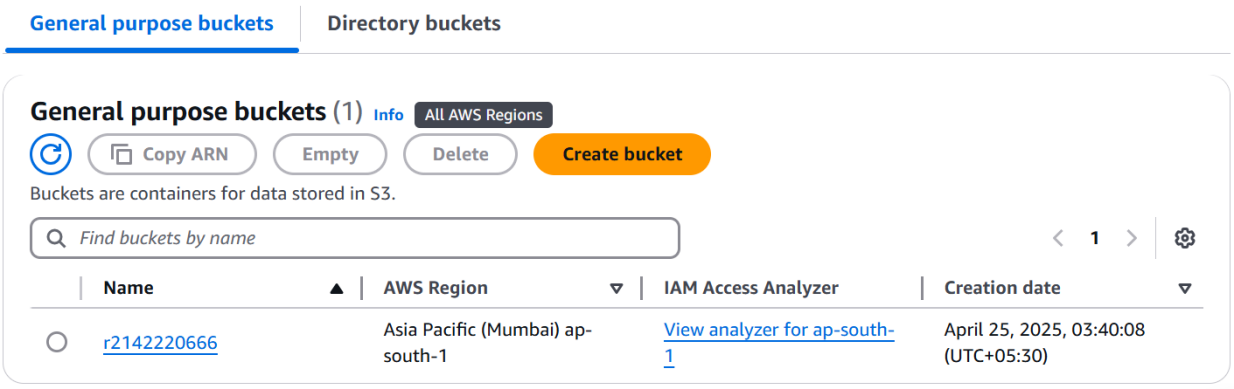
```
aws_s3_bucket.assignment_bucket: Creating...
aws_s3_bucket.assignment_bucket: Creation complete after 2s [id=r2142220666]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

Resources Created -

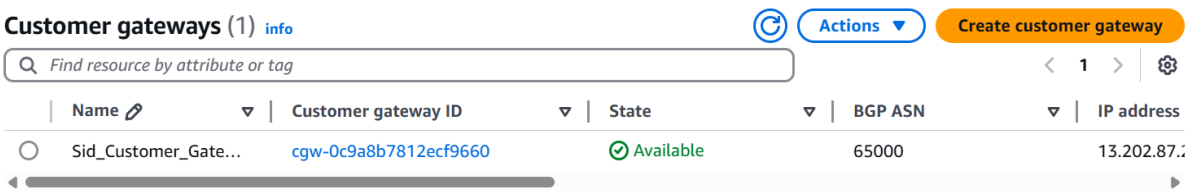
1. Instances



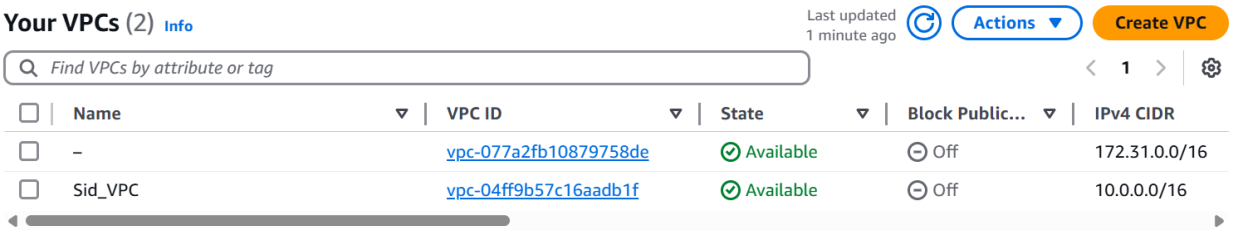
2. S3 Bucket



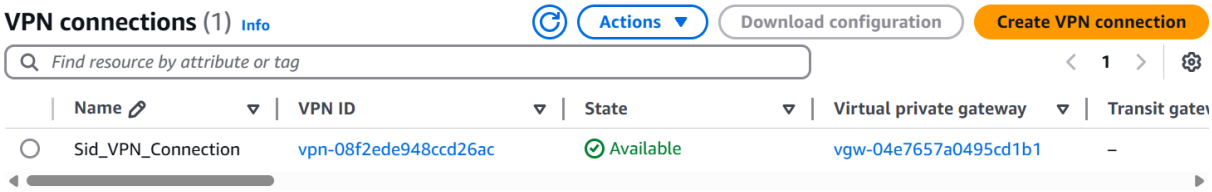
3. Customer Gateway



4. VPC



5. VPN Connections



Then use terraform destroy to clean up all the resources.

```
PS G:\New Volume E\6th sem\System Provisioning Lab\Assignment-1> terraform destroy
aws_vpc.main: Refreshing state... [id=vpc-04ff9b57c16aadb1f]
aws_s3_bucket.assignment_bucket: Refreshing state... [id=r2142220666]
aws_vpn_gateway.vpn_gw: Refreshing state... [id=vgw-04e7657a0495cd1b1]
aws_internet_gateway.igw: Refreshing state... [id=igw-0e2e8738c0ed9ca43]
aws_subnet.public: Refreshing state... [id=subnet-0e26a936205a4d00d]
aws_security_group.vpn_sg: Refreshing state... [id=sg-039725890fd2770ce]
aws_route_table.public: Refreshing state... [id=rtb-0d67b64b84661baff]
aws_route_table_association.public: Refreshing state... [id=rtbassoc-0d7c070f171a7bf8f]
aws_instance.ins2: Refreshing state... [id=i-0436d3c5e1c951f53]
aws_instance.ins1: Refreshing state... [id=i-0c7ffb55b094abc44]
aws_eip.vpn_eip: Refreshing state... [id=eipalloc-0b7533407a3bdfeed]
aws_customer_gateway.cgw: Refreshing state... [id=cgw-0c9a8b7812ecf9660]
aws_vpn_connection.main: Refreshing state... [id=vpn-08f2ede948ccd26ac]

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

Terraform will perform the following actions:

  # aws_customer_gateway.cgw will be destroyed
  - resource "aws_customer_gateway" "cgw" {
      - arn              = "arn:aws:ec2:ap-south-1:976193261889:customer-gateway/cgw-0c9a8b7812ecf9660" -> null
      - bgp_asn          = "65000" -> null
      - id               = "cgw-0c9a8b7812ecf9660" -> null
      - ip_address       = "13.202.87.220" -> null
      - tags             = {
          - "Name" = "Sid_Customer_Gateway"
        } -> null
      - tags_all         = {
          - "Name" = "Sid_Customer_Gateway"
        } -> null
```

```
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

aws_route_table_association.public: Destroying... [id=rtbassoc-0d7c070f171a7bf8f]
aws_s3_bucket.assignment_bucket: Destroying... [id=r2142220666]
aws_instance.ins2: Destroying... [id=i-0436d3c5e1c951f53]
aws_vpn_connection.main: Destroying... [id=vpn-08f2ede948ccd26ac]
aws_route_table_association.public: Destruction complete after 1s
aws_route_table.public: Destroying... [id=rtb-0d67b64b84661baff]
aws_route_table.public: Destruction complete after 0s
aws_s3_bucket.assignment_bucket: Destruction complete after 1s
aws_internet_gateway.igw: Destroying... [id=igw-0e2e8738c0ed9ca43]
aws_vpn_connection.main: Still destroying... [id=vpn-08f2ede948ccd26ac, 10s elapsed]
aws_instance.ins2: Still destroying... [id=i-0436d3c5e1c951f53, 10s elapsed]
aws_vpn_connection.main: Destruction complete after 11s
aws_subnet.public: Destroying... [id=subnet-0e26a936205a4d00d]
aws_security_group.vpn_sg: Destroying... [id=sg-039725890fd2770ce]
aws_subnet.public: Destruction complete after 1s
aws_security_group.vpn_sg: Destruction complete after 1s
aws_vpc.main: Destroying... [id=vpc-04ff9b57c16aadb1f]
aws_vpc.main: Destruction complete after 0s

Destroy complete! Resources: 13 destroyed.
PS G:\New Volume E\6th sem\System Provisioning Lab\Assignment-1>
```