Lab Exercise 4–Provisioning an EC2 Instance on AWS

Prerequisites: Terraform Installed: Make sure you have Terraform installed on your machine. Follow the official installation guide if needed.

AWS Credentials: Ensure you have AWS credentials (Access Key ID and Secret Access Key) configured. You can set them up using the AWS CLI or by setting environment variables.

Exercise Steps:

Step 1: Create a New Directory:

Create a new directory for your Terraform configuration:

"Terraform-Demo"

Step 2: Create Terraform Configuration File (main.tf):

Create a file named main.tf with the following content:

```
terraform {
 required_providers {
  aws = {
   source = "hashicorp/aws"
   version = "5.31.0"
  }
 }
}
```

```
provider "aws" {
  region     = "ap-south-1"
  access_key = "your IAM access key"
  secret_key = "your secret access key"
}
```

This script defines an AWS provider and provisions an EC2 instance.

Step 3: Initialize Terraform:

Run the following command to initialize your Terraform working directory:

```
terraform init
```

Step 4: Create Terraform Configuration File for EC2 instance (instance.tf):

Create a file named instnace.tf with the following content:

```
instance.tf
resource "aws_instance" "My-instance" {
        ami = "ami-03f4878755434977f"
        instance_type = "t2.micro"
    tags = {
      Name = "Akshit-EC2-Instance"
    }
}
```

Step 5: Review Plan:

Run the following command to see what Terraform will do:

```
terraform plan
```

```
PS D:\Coding 3rd Year\SPCM\Code> 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 will be created
  + resource "aws_instance" "My-instance" {
      + ami                                  = "ami-03f4878755434977f"
      + 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)
      + 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" = "Akshit-EC2-Instance"
        }
      + tags_all                             = {
          + "Name" = "Akshit-EC2-Instance"
        }
      + 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)
    }

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

─────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run
"terraform apply" now.
PS D:\Coding 3rd Year\SPCM\Code>
```

Review the plan to ensure it aligns with your expectations.

Step 6: Apply Changes:

Apply the changes to create the AWS resources:

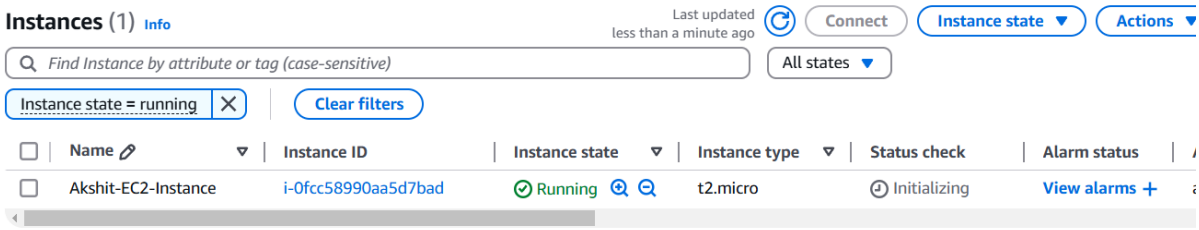
terraform apply

Type yes when prompted.

```
Plan: 1 to add, 0 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_instance.My-instance: Creating...
aws_instance.My-instance: Still creating... [10s elapsed]
aws_instance.My-instance: Creation complete after 14s [id=i-0fcc58990aa5d7bad]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS D:\Coding 3rd Year\SPCM\Code>
```

Step 7: Verify Resources:

After the terraform apply command completes, log in to your AWS Management Console and navigate to the EC2 dashboard. Verify that the EC2 instance has been created.



Step 8: Cleanup Resources:

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

```
terraform destroy
```

Type yes when prompted.

```
Plan: 0 to add, 0 to change, 1 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

aws_instance.My-instance: Destroying... [id=i-0fcc58990aa5d7bad]
aws_instance.My-instance: Still destroying... [id=i-0fcc58990aa5d7bad, 10s elapsed]
aws_instance.My-instance: Still destroying... [id=i-0fcc58990aa5d7bad, 20s elapsed]
aws_instance.My-instance: Still destroying... [id=i-0fcc58990aa5d7bad, 30s elapsed]
aws_instance.My-instance: Still destroying... [id=i-0fcc58990aa5d7bad, 40s elapsed]
aws_instance.My-instance: Still destroying... [id=i-0fcc58990aa5d7bad, 50s elapsed]
aws_instance.My-instance: Still destroying... [id=i-0fcc58990aa5d7bad, 1m0s elapsed]
aws_instance.My-instance: Destruction complete after 1m1s

Destroy complete! Resources: 1 destroyed.
PS D:\Coding 3rd Year\SPCM\Code>
```