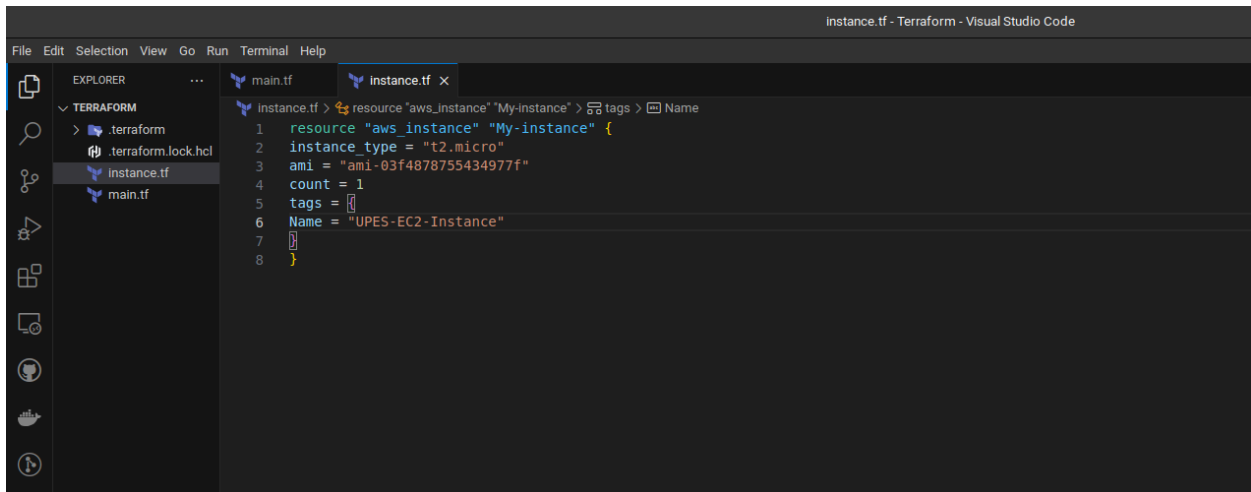


Lab Exercise 3—Provisioning an EC2 Instance on AWS

Step 1: Create a terraform configuration file for EC2 instance (instance.tf)

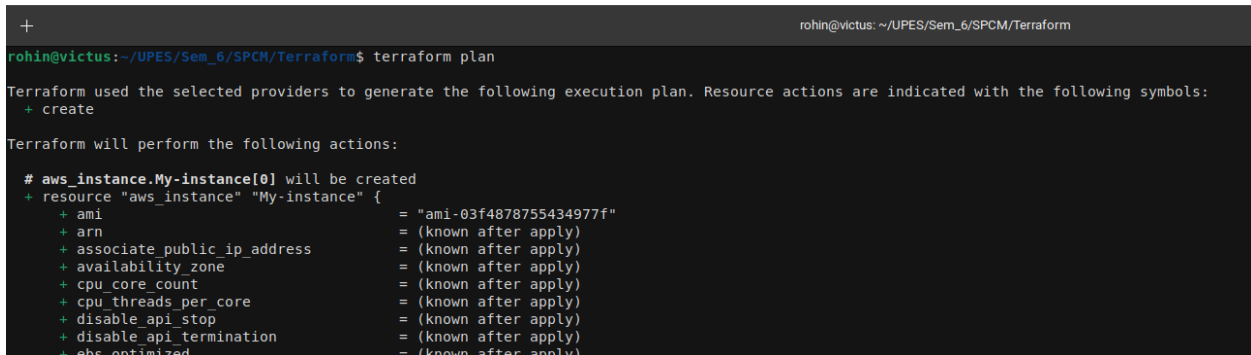
Create a file named instnace.tf with the following content:



```
Instance.tf - Terraform - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
  TERRAFORM
    .terraform
    .terraform.lock.hcl
    instance.tf
    main.tf
instance.tf x
instance.tf > resource "aws_instance" "My-instance" > tags > Name
1 resource "aws_instance" "My-instance" {
2   instance_type = "t2.micro"
3   ami = "ami-03f4878755434977f"
4   count = 1
5   tags = {}
6   Name = "UPES-EC2-Instance"
7
8 }
```

Step 2: Review Plan

Run the following command to see what Terraform will do:



```
+
rohin@victus: ~/UPES/Sem_6/SPCM/Terraform
rohin@victus:~/UPES/Sem_6/SPCM/Terraform$ 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-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)
```

Step 3: Apply changes

Apply the changes to create the AWS resources:

```
+ rohin@victus: ~/UPES/Sem_6/SPCM/Terraform
rohin@victus:~/UPES/Sem_6/SPCM/Terraform$ 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_instance.My-instance[0] 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)

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.

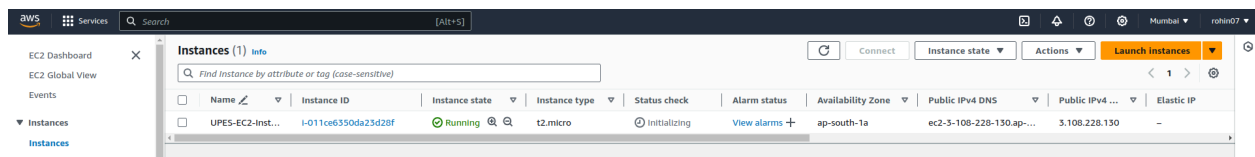
  Enter a value: yes

aws_instance.My-instance[0]: Creating...
aws_instance.My-instance[0]: Still creating... [10s elapsed]
aws_instance.My-instance[0]: Still creating... [20s elapsed]
aws_instance.My-instance[0]: Creation complete after 22s [id=i-011ce6350da23d28f]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
rohin@victus:~/UPES/Sem_6/SPCM/Terraform$
```

Step 4: 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 5 Cleanup Resources

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

```
+ rohin@victus: ~/UPES/Sem_6/SPCM/Terraform
rohin@victus:~/UPES/Sem_6/SPCM/Terraform$ terraform destroy
aws_instance.My-instance[0]: Refreshing state... [id=i-011ce6350da23d28f]

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_instance.My-instance[0] will be destroyed
- resource "aws_instance" "My-instance" {
  - ami                  = "ami-03f4878755434977f" -> null
  - arn                  = "arn:aws:ec2:ap-south-1:439533254892:instance/i-011ce6350da23d28f" -> null
  - associate_public_ip_address = true -> null
  - availability_zone     = "ap-south-1a" -> null
  - 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)
}
```

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[0]: Destroying... [id=i-011ce6350da23d28f]

aws_instance.My-instance[0]: Still destroying... [id=i-011ce6350da23d28f, 10s elapsed]

aws_instance.My-instance[0]: Still destroying... [id=i-011ce6350da23d28f, 20s elapsed]

aws_instance.My-instance[0]: Still destroying... [id=i-011ce6350da23d28f, 30s elapsed]

aws_instance.My-instance[0]: Destruction complete after 31s

Destroy complete! Resources: 1 destroyed.

rohin@victus:~/UPES/Sem_6/SPCM/Terraform\$