Karthick J (DO14)

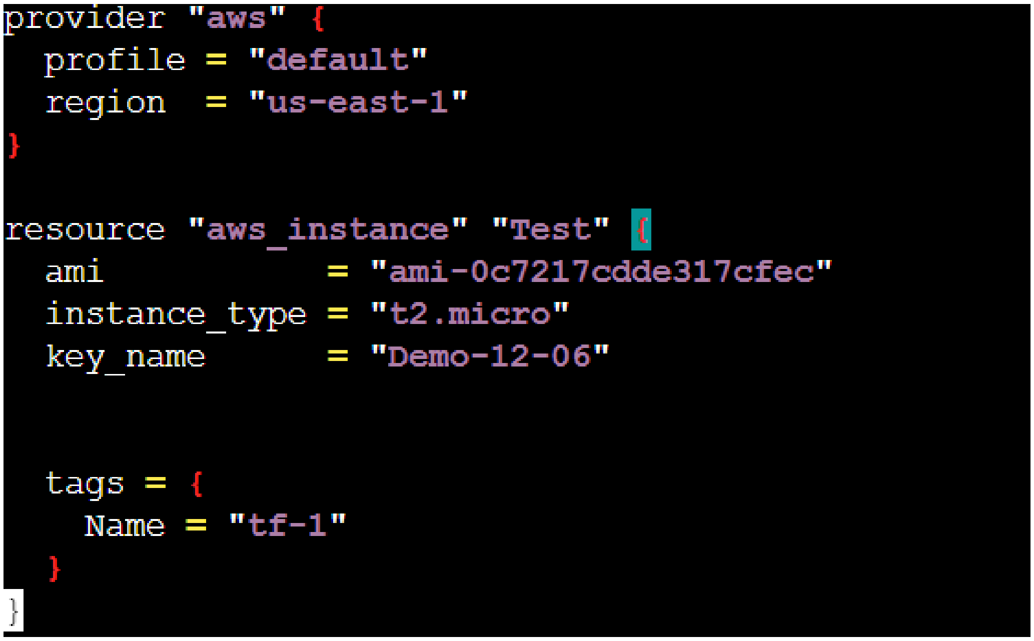
Question :

Launch an ec2 instance under a default subnet and VPC using terraform template.

Solution:

Step :1

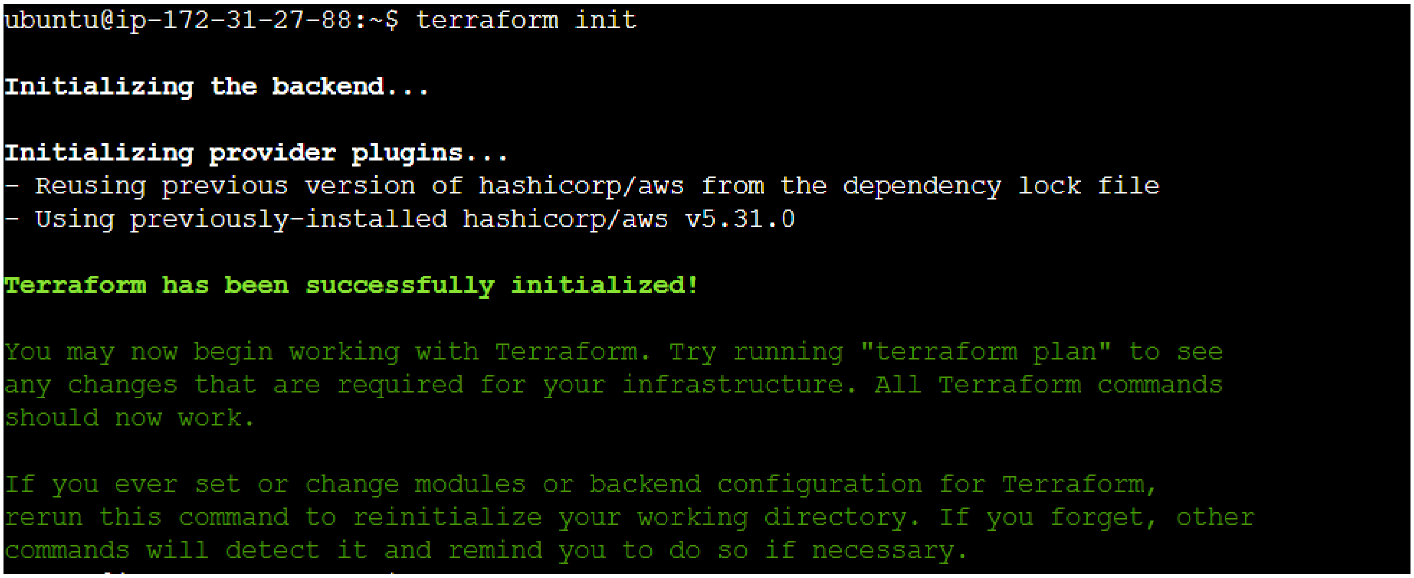
Code for default subnet and VPC in terraform.



```
provider "aws" {
  profile = "default"
  region  = "us-east-1"
}

resource "aws_instance" "Test" {
  ami            = "ami-0c7217cdde317cfec"
  instance_type = "t2.micro"
  key_name       = "Demo-12-06"


  tags = {
    Name = "tf-1"
  }
}
```

Step :2

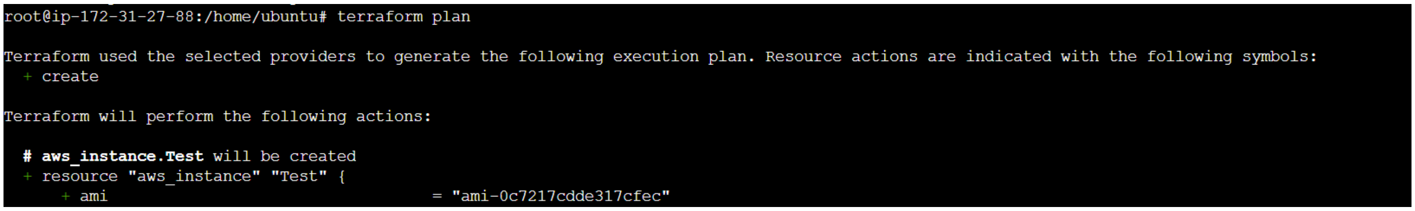
Terraform Init:



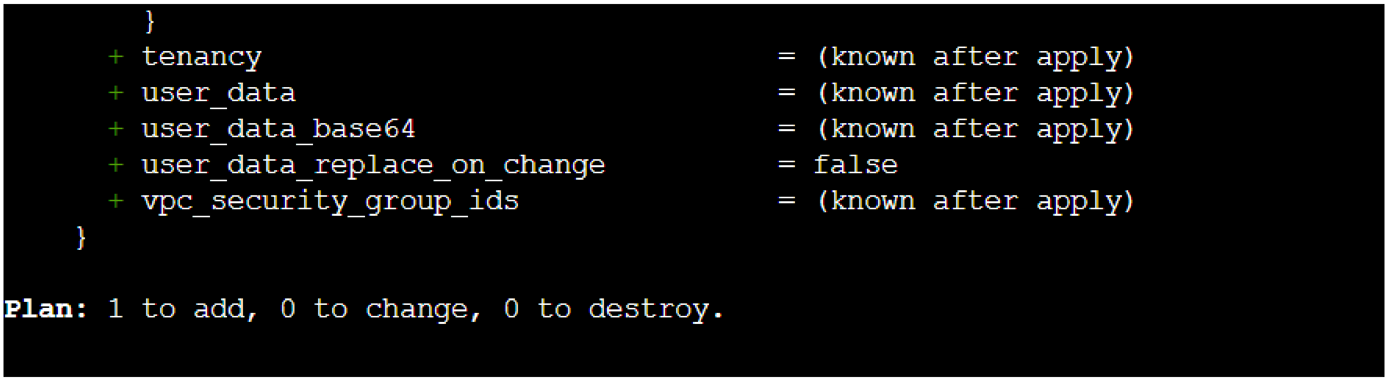
```
ubuntu@ip-172-31-27-88:~$ terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.31.0

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.
```

Install and configure awscli, if it is not configured.

Terraform plan:

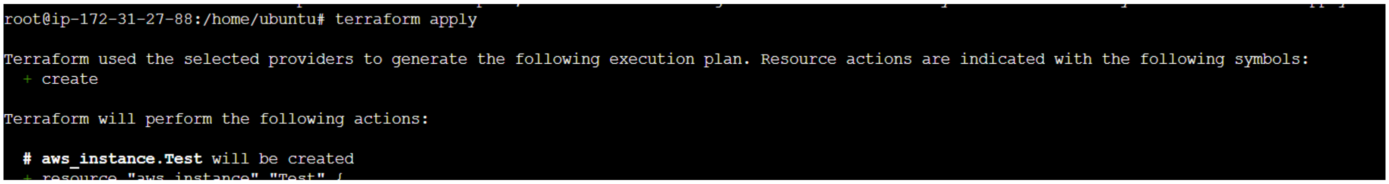


```
root@ip-172-31-27-88:/home/ubuntu# terraform plan

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.Test will be created
  + resource "aws_instance" "Test" {
      + ami                                  = "ami-0c7217cdde317cfec"
```

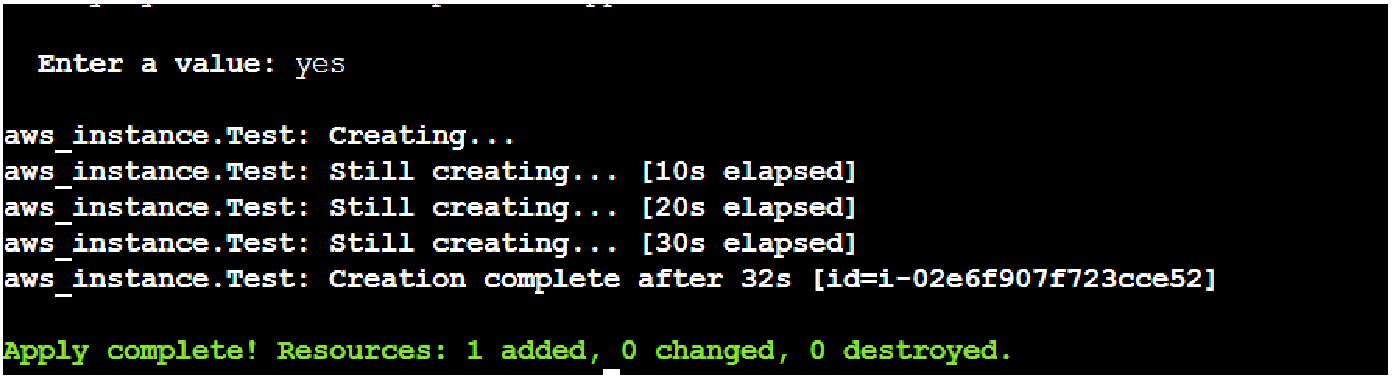


```
          }
      + tenancy                              = (known after apply)
      + user_data                            = (known after apply)
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids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

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

Terraform apply:



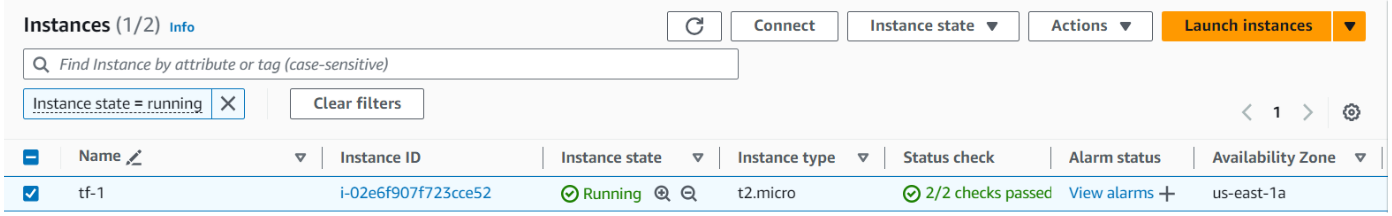
```
root@ip-172-31-27-88:/home/ubuntu# terraform apply

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.Test will be created
  + resource "aws_instance" "Test" {
```



```
  Enter a value: yes

aws_instance.Test: Creating...
aws_instance.Test: Still creating... [10s elapsed]
aws_instance.Test: Still creating... [20s elapsed]
aws_instance.Test: Still creating... [30s elapsed]
aws_instance.Test: Creation complete after 32s [id=i-02e6f907f723cce52]

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

Output:

Creation on EC2 instance.



☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
☑	tf-1	i-02e6f907f723cce52	✓ Running	t2.micro	✓ 2/2 checks passed	View alarms +	us-east-1a