Infrastructure Automation using Terraform

Create EC2 Instance

accenture

Table of Contents

This Activity demonstrates the creation of EC2 Instance in AWS using Terraform.

Prerequisite:

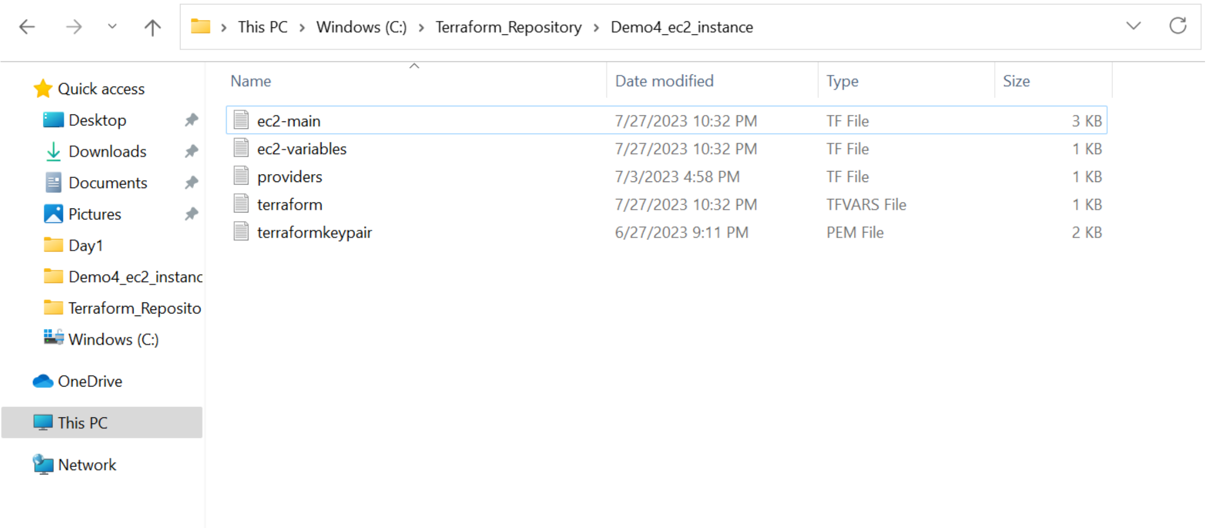
1) Download the zip file shared by the trainer and extract it.

Walkthrough:

1. Initializing Terraform Directory
2. Creating EC2 Instance
3. Destroying EC2 Instance

Part 1: Initializing Terraform Directory

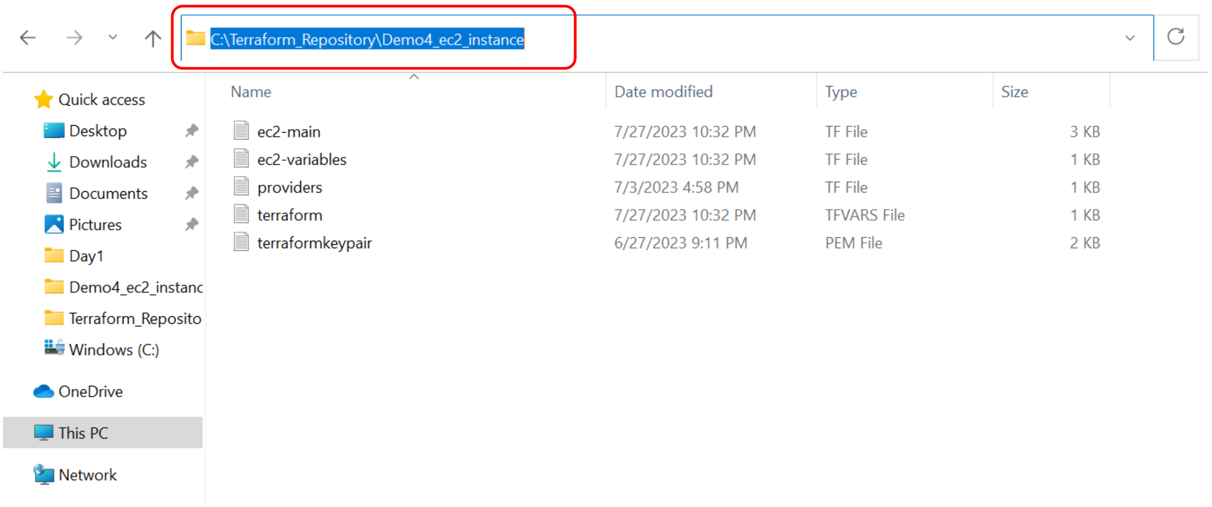
1 Open the extracted folder and navigate to ".tf" files.



This PC › Windows (C:) › Terraform_Repository › Demo4_ec2_instance

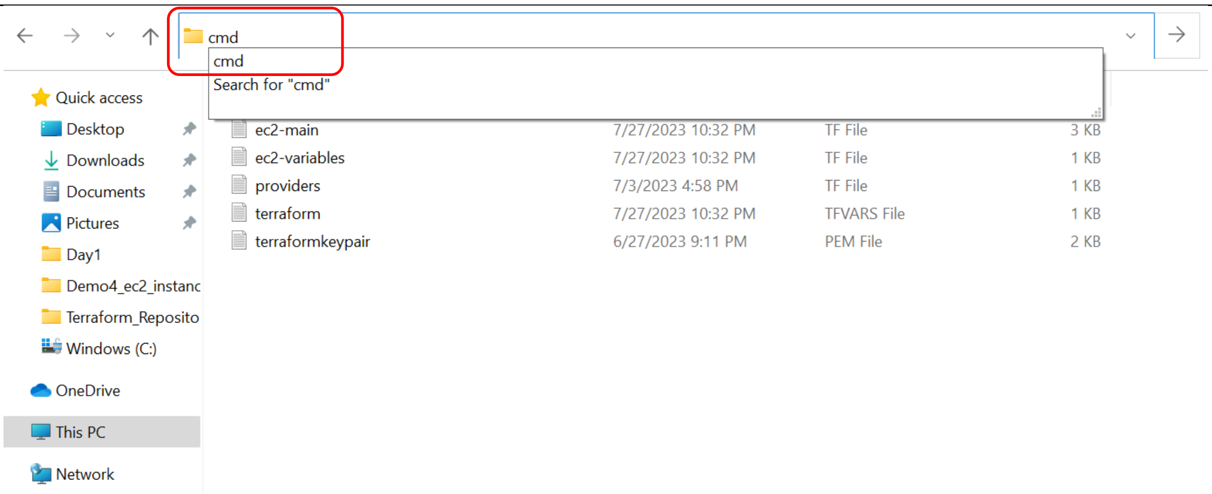
Name	Date modified	Type	Size
ec2-main	7/27/2023 10:32 PM	TF File	3 KB
ec2-variables	7/27/2023 10:32 PM	TF File	1 KB
providers	7/3/2023 4:58 PM	TF File	1 KB
terraform	7/27/2023 10:32 PM	TFVARS File	1 KB
terraformkeypair	6/27/2023 9:11 PM	PEM File	2 KB

2 Click on the Address bar and type cmd. Press Enter (It will open a command prompt from that location).

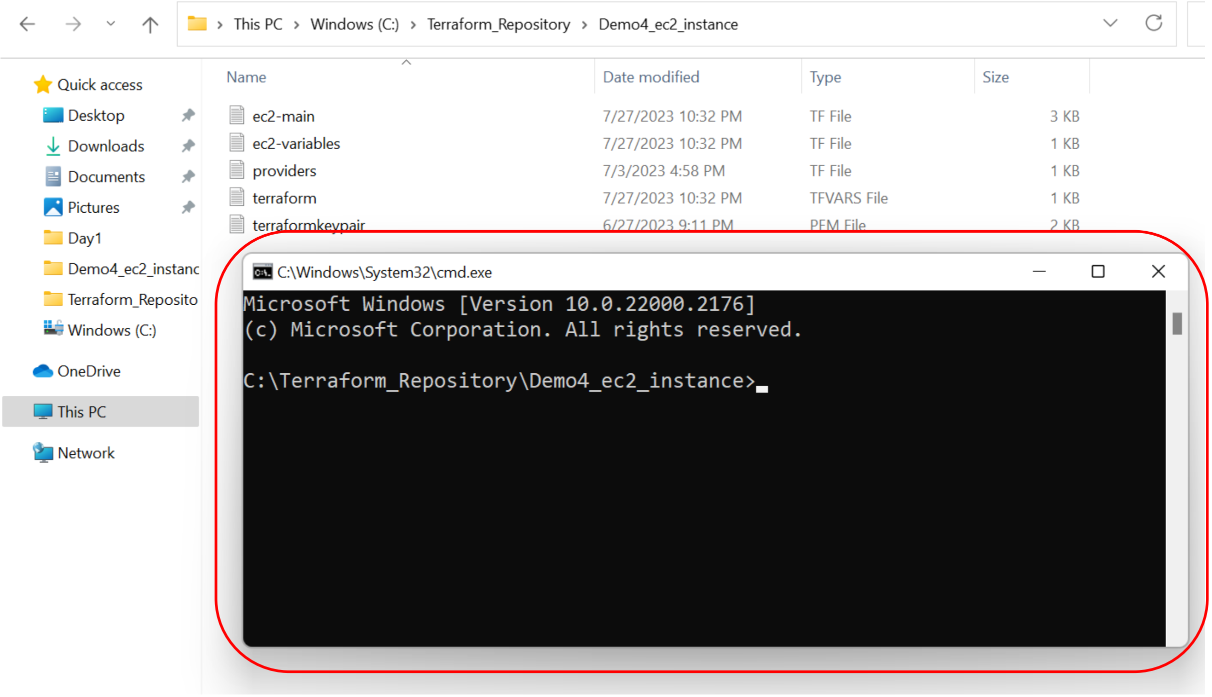


C:\Terraform_Repository\Demo4_ec2_instance

Name	Date modified	Type	Size
ec2-main	7/27/2023 10:32 PM	TF File	3 KB
ec2-variables	7/27/2023 10:32 PM	TF File	1 KB
providers	7/3/2023 4:58 PM	TF File	1 KB
terraform	7/27/2023 10:32 PM	TFVARS File	1 KB
terraformkeypair	6/27/2023 9:11 PM	PEM File	2 KB



Name	Date modified	Type	Size
ec2-main	7/27/2023 10:32 PM	TF File	3 KB
ec2-variables	7/27/2023 10:32 PM	TF File	1 KB
providers	7/3/2023 4:58 PM	TF File	1 KB
terraform	7/27/2023 10:32 PM	TFVARS File	1 KB
terraformkeypair	6/27/2023 9:11 PM	PEM File	2 KB

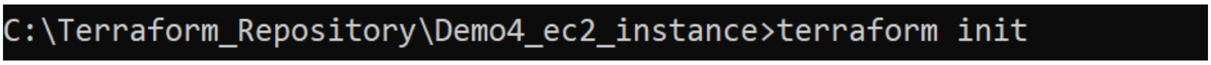


Name	Date modified	Type	Size
ec2-main	7/27/2023 10:32 PM	TF File	3 KB
ec2-variables	7/27/2023 10:32 PM	TF File	1 KB
providers	7/3/2023 4:58 PM	TF File	1 KB
terraform	7/27/2023 10:32 PM	TFVARS File	1 KB
terraformkeypair	6/27/2023 9:11 PM	PEM File	2 KB

```
Microsoft Windows [Version 10.0.22000.2176]
(c) Microsoft Corporation. All rights reserved.

C:\Terraform_Repository\Demo4_ec2_instance>
```

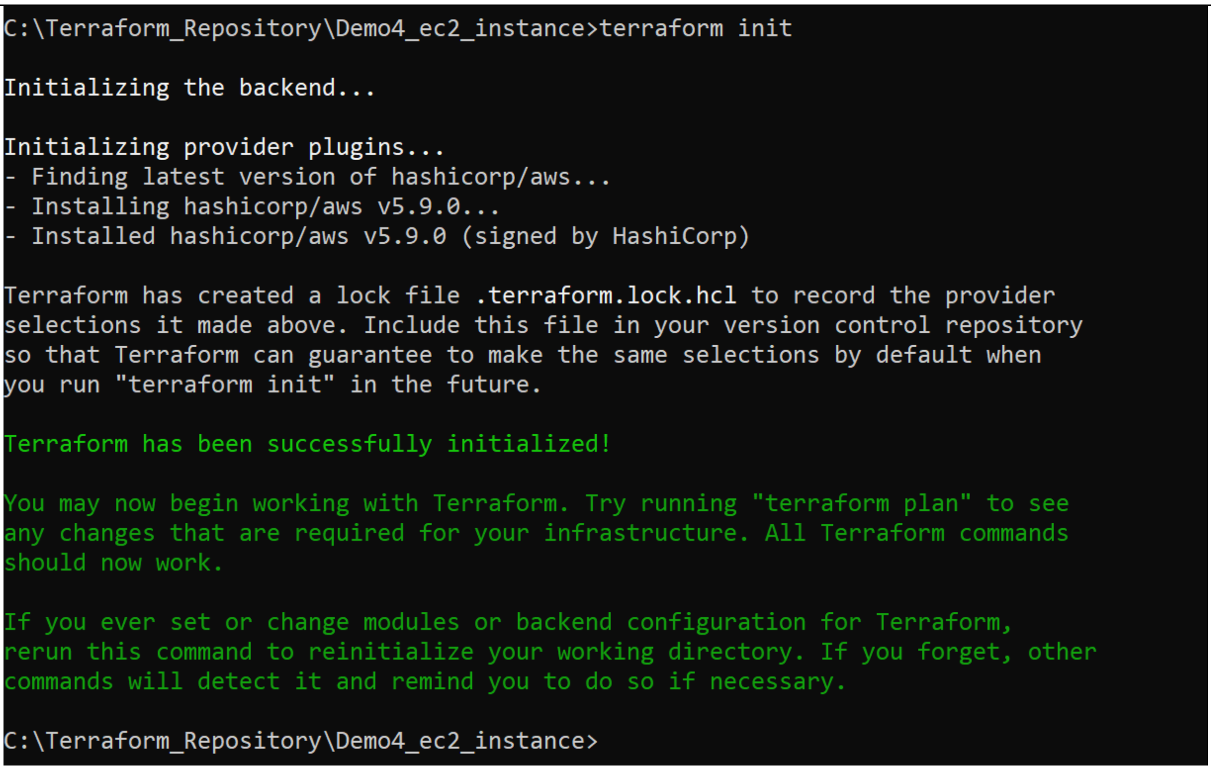
3 Execute below command to initialize the current directory as Terraform directory which enables us to run terraform commands to manage Infrastructure.

Command :



```
C:\Terraform_Repository\Demo4_ec2_instance>terraform init
```

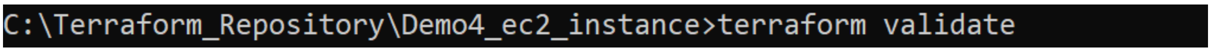
Result :



```
C:\Terraform_Repository\Demo4_ec2_instance>terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.9.0...
- Installed hashicorp/aws v5.9.0 (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.

C:\Terraform_Repository\Demo4_ec2_instance>
```

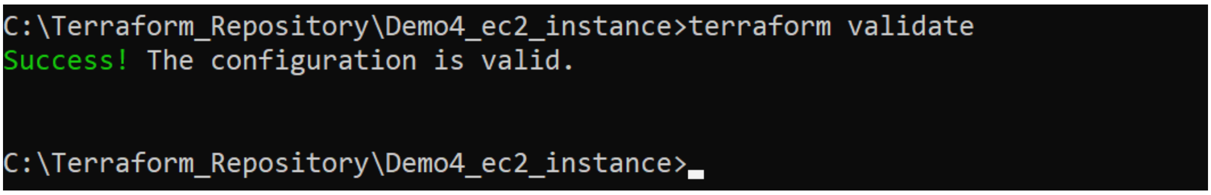
4 Next execute below command to validate syntax and configuration of terraform configuration files. If everything is proper, it will return a success message otherwise it will display the errors.

Command :



```
C:\Terraform_Repository\Demo4_ec2_instance>terraform validate
```

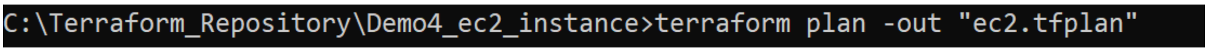
Result :



```
C:\Terraform_Repository\Demo4_ec2_instance>terraform validate
Success! The configuration is valid.


C:\Terraform_Repository\Demo4_ec2_instance>
```

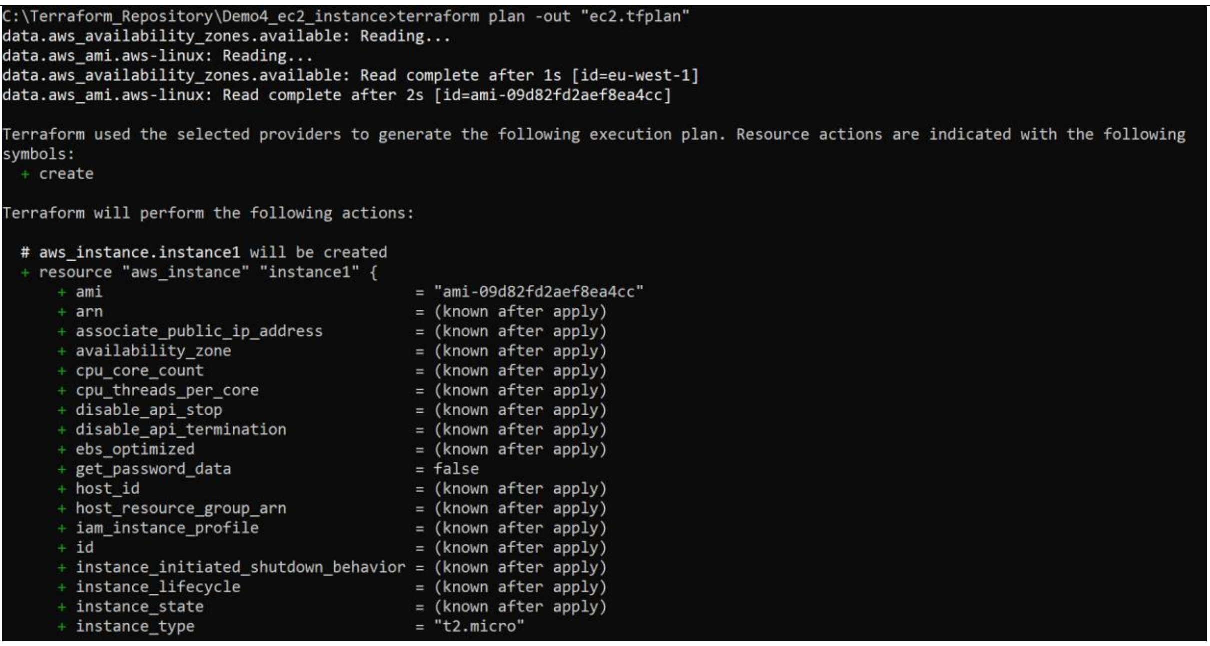
5 Next run below command and observe the output. The output contains information depicting all the changes which will happen in the AWS cloud. It is like dry-run to ensure whatever we are trying to do using terraform commands is what we want.

Command :



```
C:\Terraform_Repository\Demo4_ec2_instance>terraform plan -out "ec2.tfplan"
```

Result :

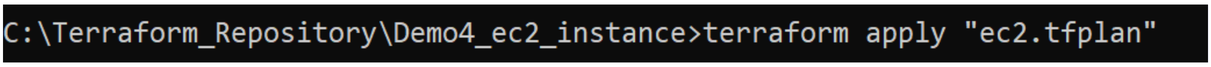


```
C:\Terraform_Repository\Demo4_ec2_instance>terraform plan -out "ec2.tfplan"
data.aws_availability_zones.available: Reading...
data.aws_ami.aws-linux: Reading...
data.aws_availability_zones.available: Read complete after 1s [id=eu-west-1]
data.aws_ami.aws-linux: Read complete after 2s [id=ami-09d82fd2aef8ea4cc]

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.instance1 will be created
  + resource "aws_instance" "instance1" {
      + ami                                  = "ami-09d82fd2aef8ea4cc"
      + 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"
```

Part 2: Creating EC2 Instance

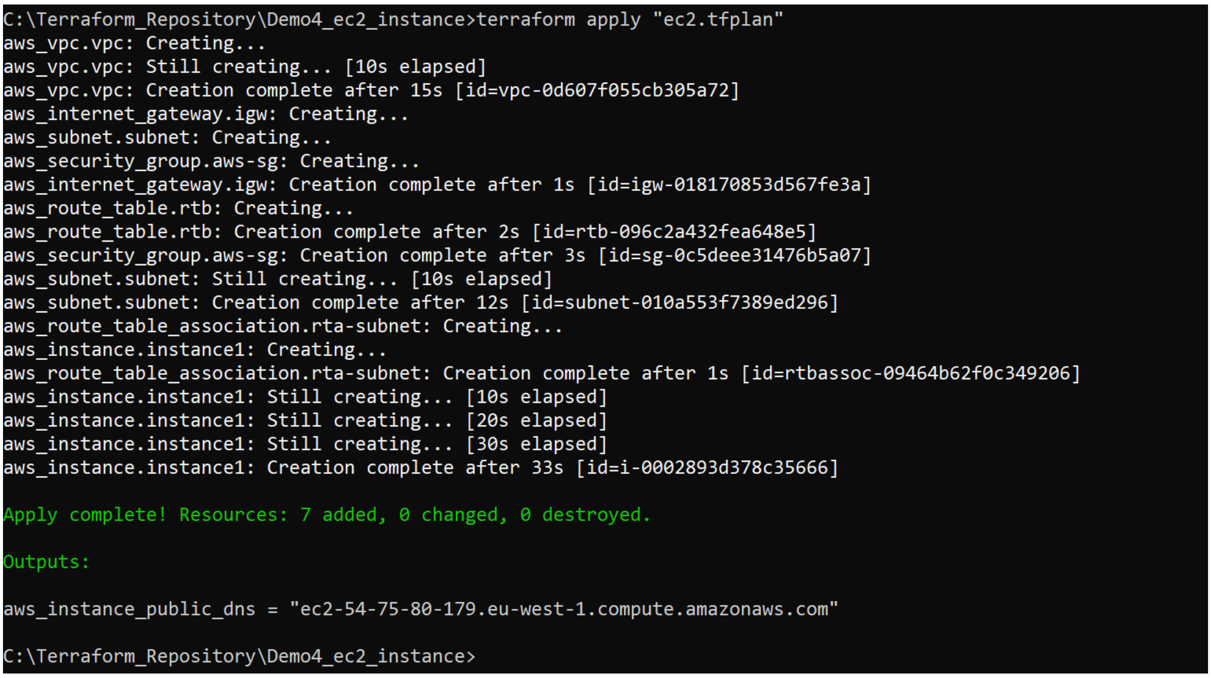
1 For creating EC2 Instance , execute below command and observe the actions performed by the command.

Command :



```
C:\Terraform_Repository\Demo4_ec2_instance>terraform apply "ec2.tfplan"
```

Result :

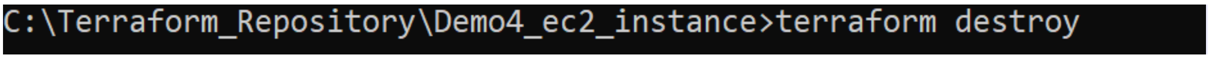


```
C:\Terraform_Repository\Demo4_ec2_instance>terraform apply "ec2.tfplan"
aws_vpc.vpc: Creating...
aws_vpc.vpc: Still creating... [10s elapsed]
aws_vpc.vpc: Creation complete after 15s [id=vpc-0d607f055cb305a72]
aws_internet_gateway.igw: Creating...
aws_subnet.subnet: Creating...
aws_security_group.aws-sg: Creating...
aws_internet_gateway.igw: Creation complete after 1s [id=igw-018170853d567fe3a]
aws_route_table.rtb: Creating...
aws_route_table.rtb: Creation complete after 2s [id=rtb-096c2a432fea648e5]
aws_security_group.aws-sg: Creation complete after 3s [id=sg-0c5deee31476b5a07]
aws_subnet.subnet: Still creating... [10s elapsed]
aws_subnet.subnet: Creation complete after 12s [id=subnet-010a553f7389ed296]
aws_route_table_association.rta-subnet: Creating...
aws_instance.instance1: Creating...
aws_route_table_association.rta-subnet: Creation complete after 1s [id=rtbassoc-09464b62f0c349206]
aws_instance.instance1: Still creating... [10s elapsed]
aws_instance.instance1: Still creating... [20s elapsed]
aws_instance.instance1: Still creating... [30s elapsed]
aws_instance.instance1: Creation complete after 33s [id=i-0002893d378c35666]

Apply complete! Resources: 7 added, 0 changed, 0 destroyed.

Outputs:

aws_instance_public_dns = "ec2-54-75-80-179.eu-west-1.compute.amazonaws.com"

C:\Terraform_Repository\Demo4_ec2_instance>
```

Part 3: Destroying EC2 Instance

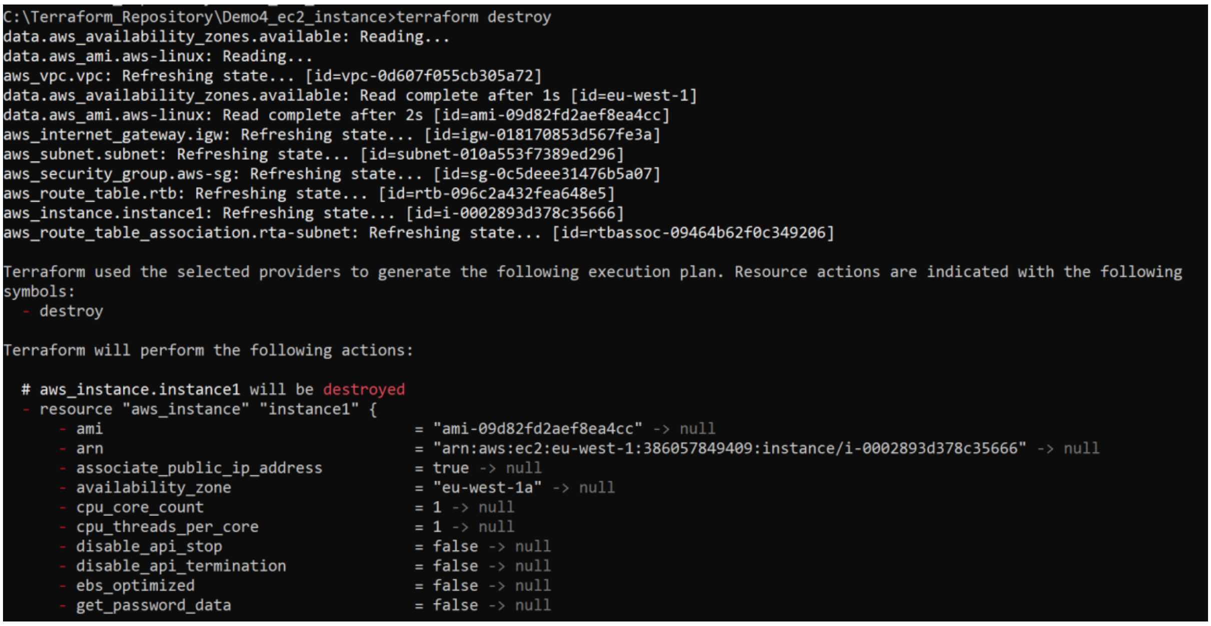
1 Execute below command to destroy the EC2 Instance which we have created in previous step. After you execute below command, it will show you what changes will be done and before doing those changes it will ask for your approval. So, if you want to proceed with destroying EC2 Instance, provide “yes”.

Command:

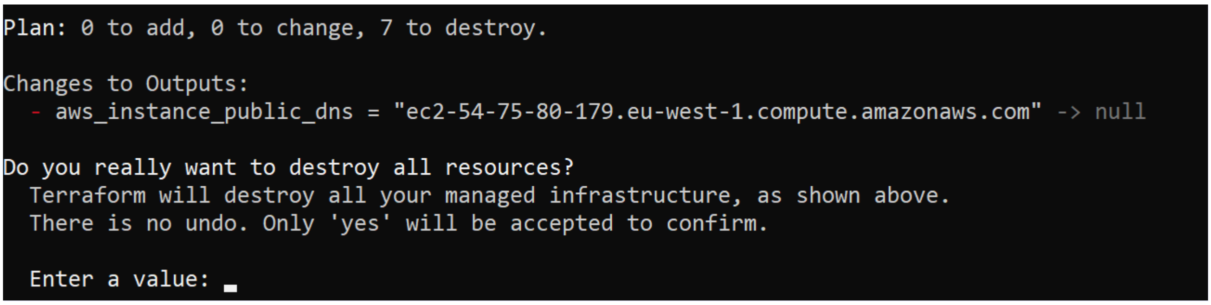


```
C:\Terraform_Repository\Demo4_ec2_instance>terraform destroy
```

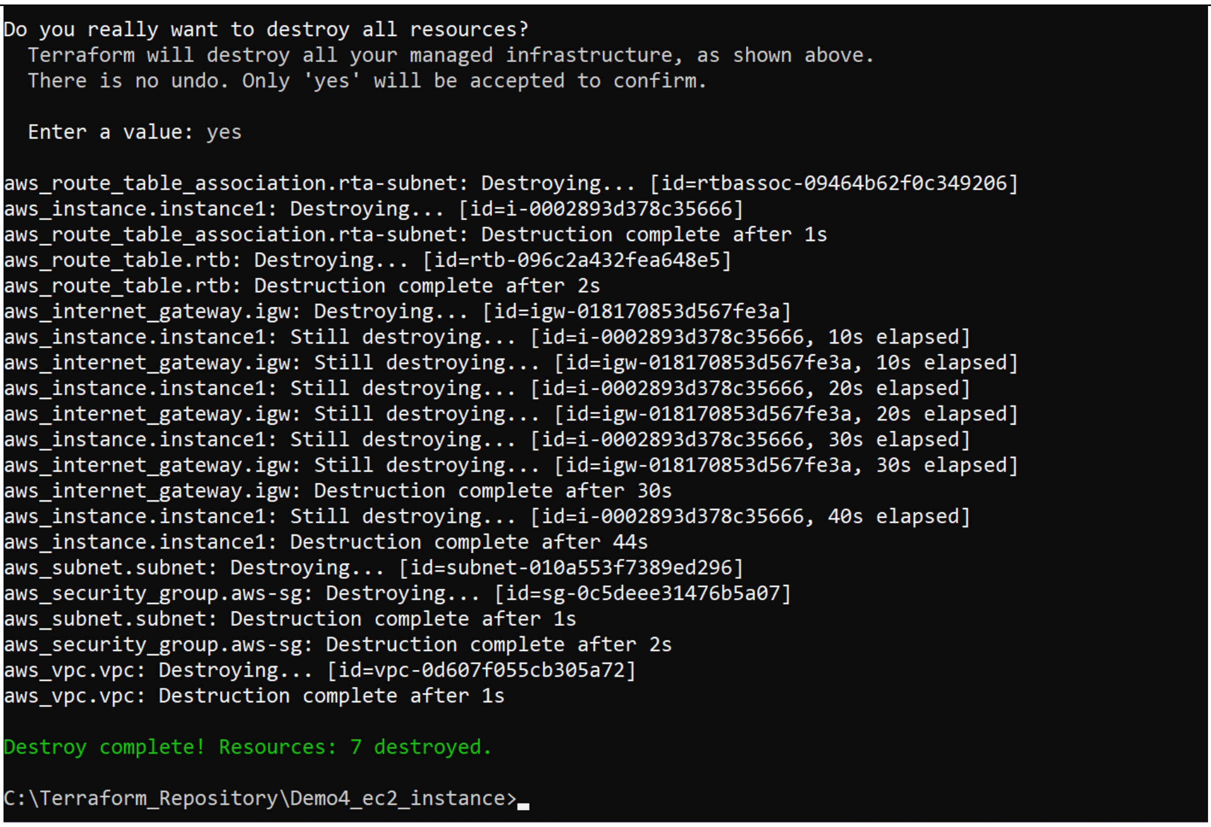
Result:



```
C:\Terraform_Repository\Demo4_ec2_instance>terraform destroy
data.aws_availability_zones.available: Reading...
data.aws_ami.aws-linux: Reading...
aws_vpc.vpc: Refreshing state... [id=vpc-0d607f055cb305a72]
data.aws_availability_zones.available: Read complete after 1s [id=eu-west-1]
data.aws_ami.aws-linux: Read complete after 2s [id=ami-09d82fd2aef8ea4cc]
aws_internet_gateway.igw: Refreshing state... [id=igw-018170853d567fe3a]
aws_subnet.subnet: Refreshing state... [id=subnet-010a553f7389ed296]
aws_security_group.aws-sg: Refreshing state... [id=sg-0c5deee31476b5a07]
aws_route_table.rtb: Refreshing state... [id=rtb-096c2a432fea648e5]
aws_instance.instance1: Refreshing state... [id=i-0002893d378c35666]
aws_route_table_association.rta-subnet: Refreshing state... [id=rtbassoc-09464b62f0c349206]

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.instance1 will be destroyed
  - resource "aws_instance" "instance1" {
      - ami                                  = "ami-09d82fd2aef8ea4cc" -> null
      - arn                                  = "arn:aws:ec2:eu-west-1:386057849409:instance/i-0002893d378c35666" -> null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "eu-west-1a" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 1 -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - get_password_data                    = false -> null
```



```
Plan: 0 to add, 0 to change, 7 to destroy.

Changes to Outputs:
  - aws_instance_public_dns = "ec2-54-75-80-179.eu-west-1.compute.amazonaws.com" -> 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:
```



```
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.rta-subnet: Destroying... [id=rtbassoc-09464b62f0c349206]
aws_instance.instance1: Destroying... [id=i-0002893d378c35666]
aws_route_table_association.rta-subnet: Destruction complete after 1s
aws_route_table.rtb: Destroying... [id=rtb-096c2a432fea648e5]
aws_route_table.rtb: Destruction complete after 2s
aws_internet_gateway.igw: Destroying... [id=igw-018170853d567fe3a]
aws_instance.instance1: Still destroying... [id=i-0002893d378c35666, 10s elapsed]
aws_internet_gateway.igw: Still destroying... [id=igw-018170853d567fe3a, 10s elapsed]
aws_instance.instance1: Still destroying... [id=i-0002893d378c35666, 20s elapsed]
aws_internet_gateway.igw: Still destroying... [id=igw-018170853d567fe3a, 20s elapsed]
aws_instance.instance1: Still destroying... [id=i-0002893d378c35666, 30s elapsed]
aws_internet_gateway.igw: Still destroying... [id=igw-018170853d567fe3a, 30s elapsed]
aws_internet_gateway.igw: Destruction complete after 30s
aws_instance.instance1: Still destroying... [id=i-0002893d378c35666, 40s elapsed]
aws_instance.instance1: Destruction complete after 44s
aws_subnet.subnet: Destroying... [id=subnet-010a553f7389ed296]
aws_security_group.aws-sg: Destroying... [id=sg-0c5deee31476b5a07]
aws_subnet.subnet: Destruction complete after 1s
aws_security_group.aws-sg: Destruction complete after 2s
aws_vpc.vpc: Destroying... [id=vpc-0d607f055cb305a72]
aws_vpc.vpc: Destruction complete after 1s

Destroy complete! Resources: 7 destroyed.

C:\Terraform_Repository\Demo4_ec2_instance>
```