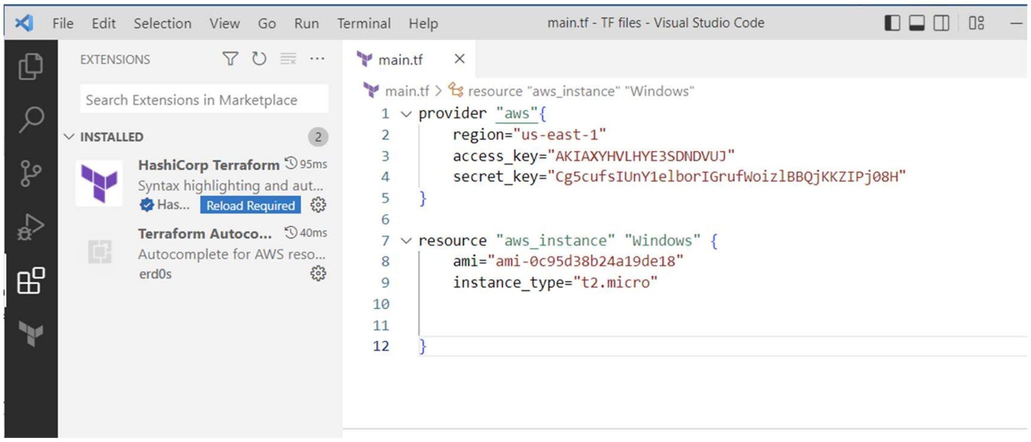
Procedure:

Follow the following steps to create an EC2 instance using Terraform :

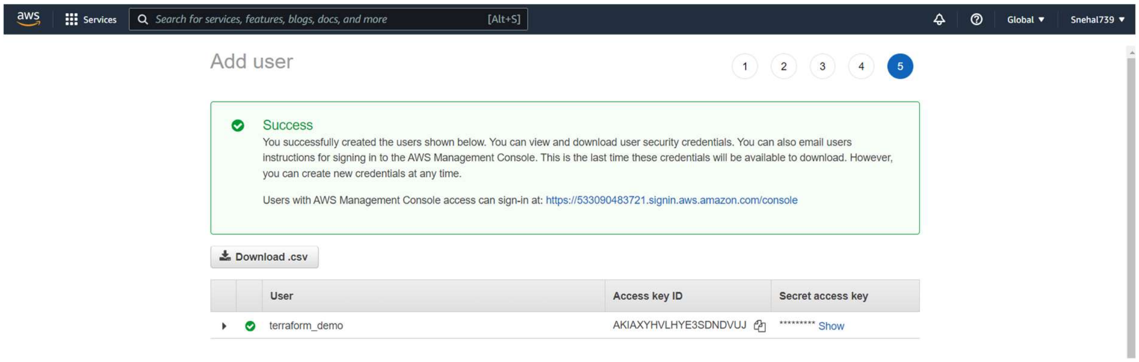
1.Install VC Extension to support terraform coding.


File Edit Selection View Go Run Terminal Help
main.tf - TF files - Visual Studio Code
EXTENSIONS
Search Extensions in Marketplace
INSTALLED 2
HashiCorp Terraform 95ms
Syntax highlighting and aut...
Has... Reload Required
Terraform Autoco... 40ms
Autocomplete for AWS reso...
erd0s
main.tf
main.tf > resource "aws_instance" "Windows"

```
provider "aws"{
    region="us-east-1"
    access_key="AKIAXYHVLHYE3SDNDVUJ"
    secret_key="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"
}

resource "aws_instance" "Windows" {
    ami="ami-0c95d38b24a19de18"
    instance_type="t2.micro"


}
```

2. We need to connect AWS account (for that you will require AWS Account)

3. We need to Create Users in IAM giving the EC2 access policy or the Admin Access


aws
Services
Search for services, features, blogs, docs, and more [Alt+S]
Global
Snehal739
Add user
1 2 3 4 5
Download .csv

Success

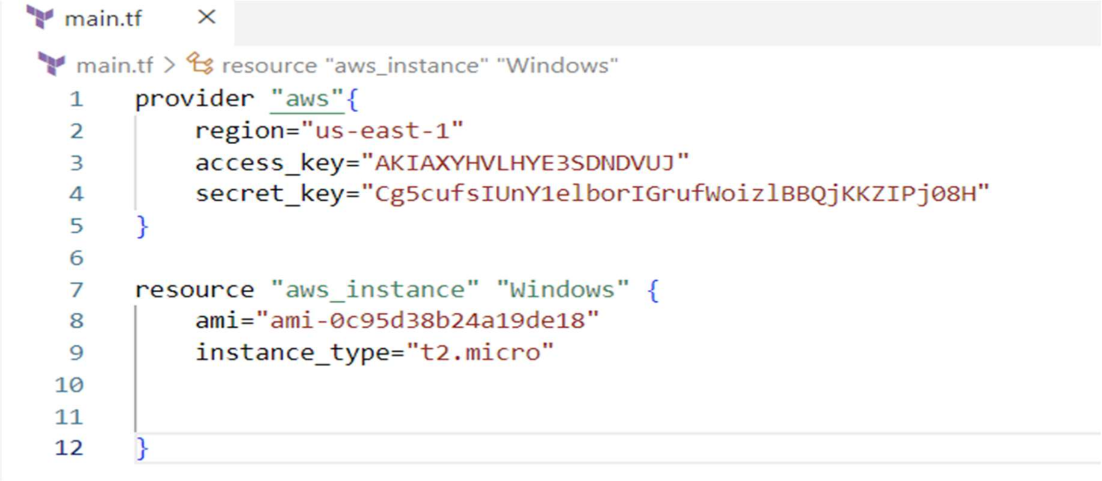
You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: https://533090483721.signin.aws.amazon.com/console

	User	Access key ID	Secret access key
✓	terraform_demo	AKIAXYHVLHYE3SDNDVUJ	******** Show

4. Provide your access key and Secret key and AMI ID (Amazon Machine Image ID)

5.Ready with code to launch EC2 Instance



```
provider "aws"{
    region="us-east-1"
    access_key="AKIAXYHVLHYE3SDNDVUJ"
    secret_key="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"
}

resource "aws_instance" "Windows" {
    ami="ami-0c95d38b24a19de18"
    instance_type="t2.micro"


}
```

Give the following Commands in the Terminal

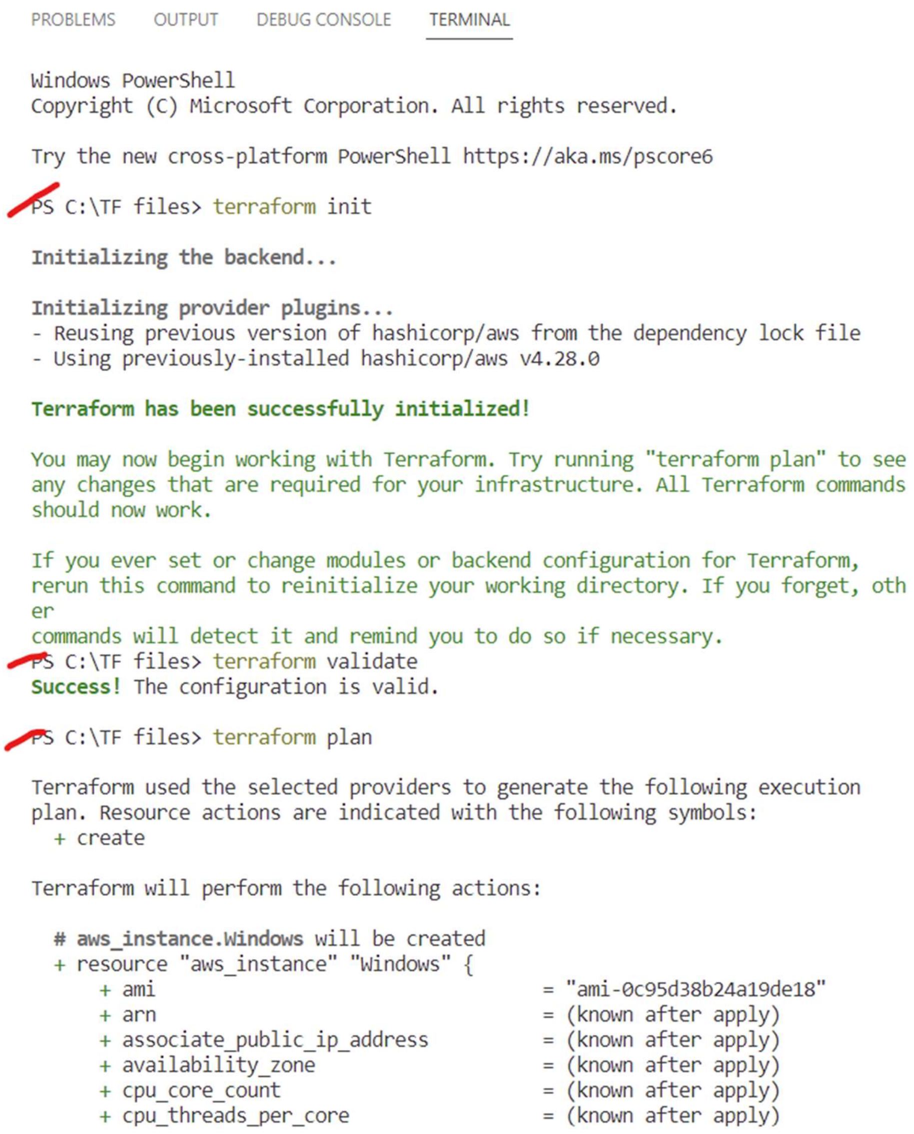
>> terraform init

>> terraform validate

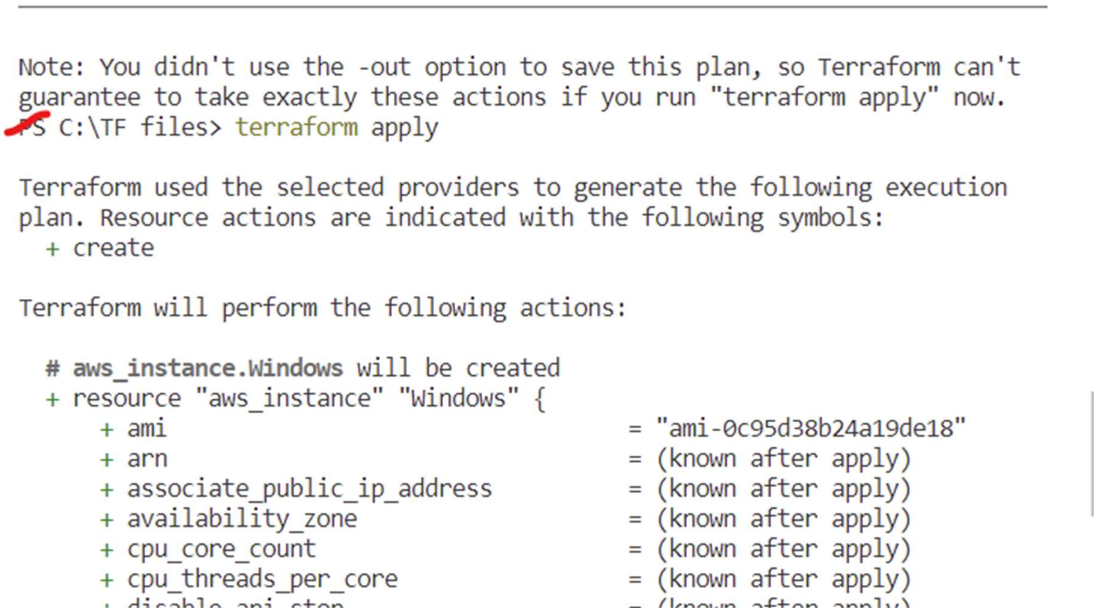
>> terraform plan

>> terraform apply

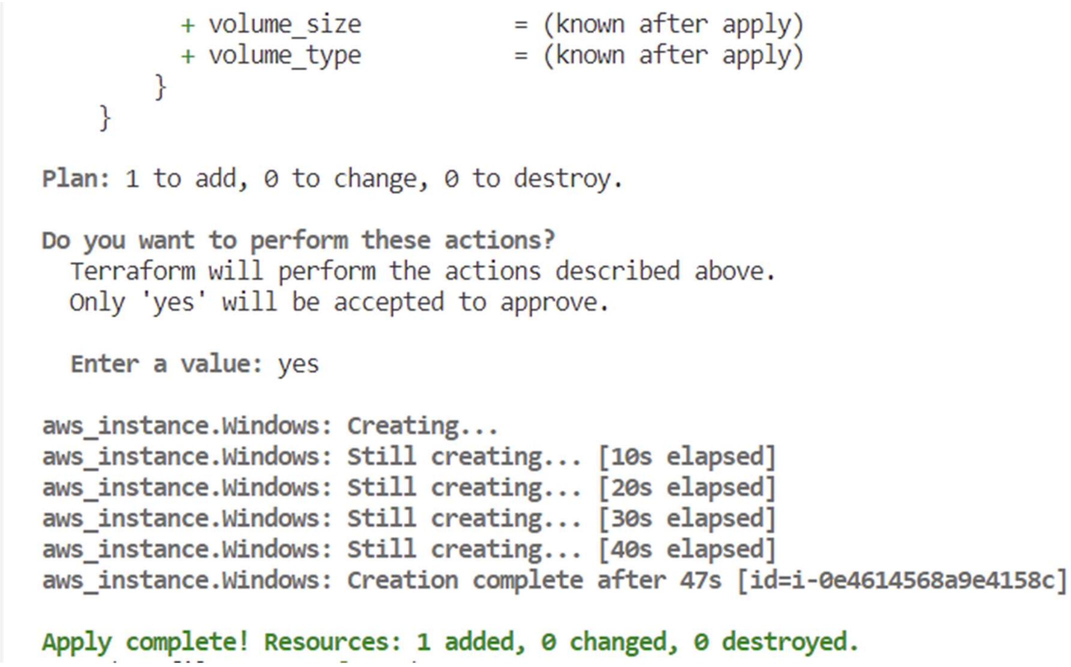
>> terraform destroy



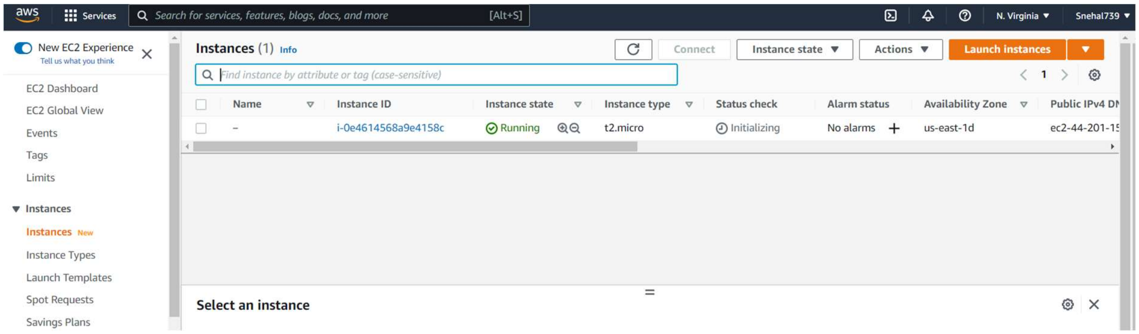
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\TF files> terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v4.28.0

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, oth
er
commands will detect it and remind you to do so if necessary.
PS C:\TF files> terraform validate
Success! The configuration is valid.

PS C:\TF files> 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.Windows will be created
  + resource "aws_instance" "Windows" {
      + ami                                  = "ami-0c95d38b24a19de18"
      + 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)
```



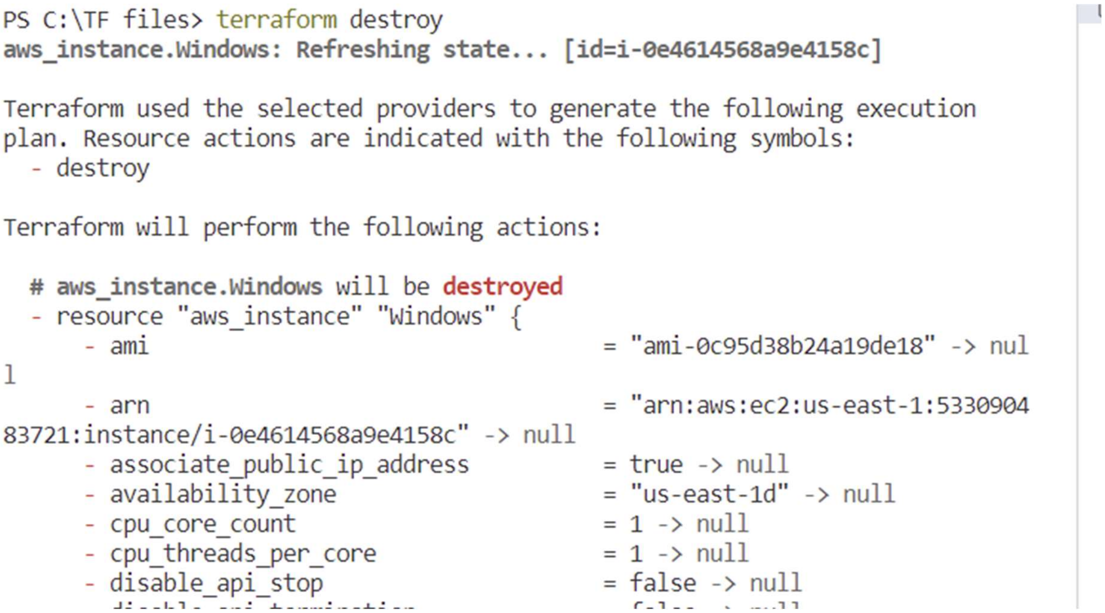
```
Note: You didn't use the -out option to save this plan, so Terraform can't
guarantee to take exactly these actions if you run "terraform apply" now.
PS C:\TF files> terraform apply

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.Windows will be created
  + resource "aws_instance" "Windows" {
      + ami                                  = "ami-0c95d38b24a19de18"
      + 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)
```



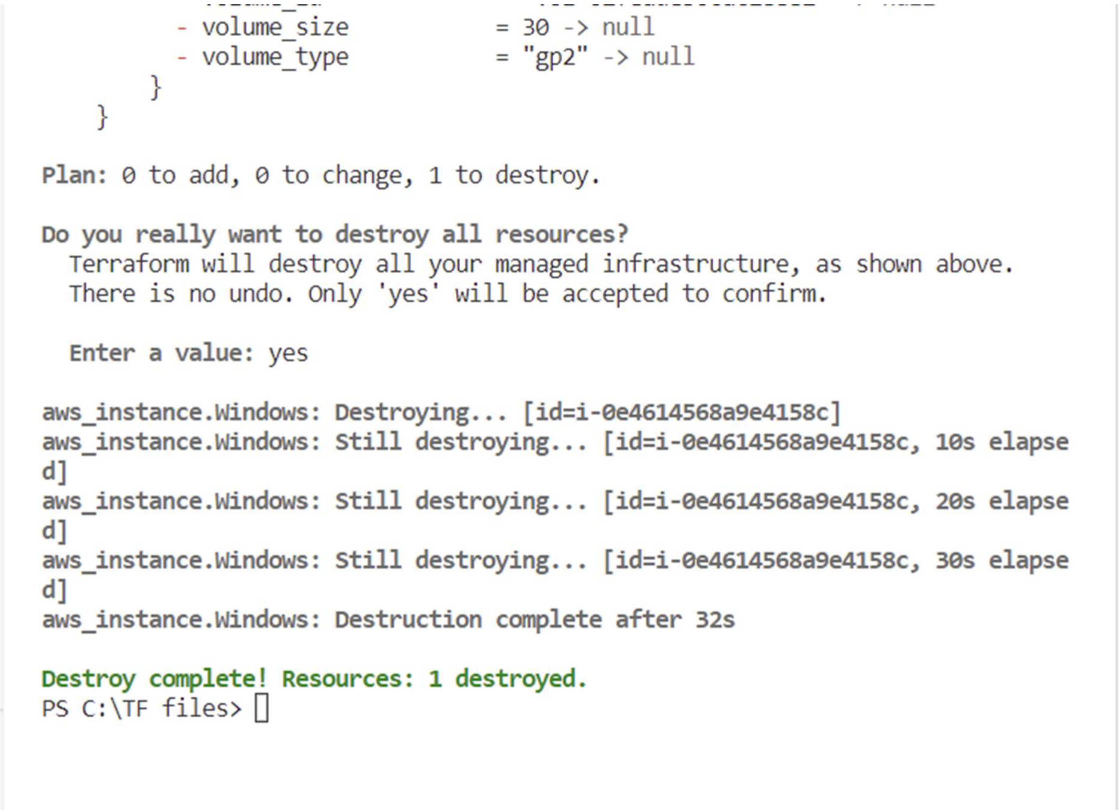
```
          + volume_size           = (known after apply)
          + volume_type           = (known after apply)
        }
    }

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.Windows: Creating...
aws_instance.Windows: Still creating... [10s elapsed]
aws_instance.Windows: Still creating... [20s elapsed]
aws_instance.Windows: Still creating... [30s elapsed]
aws_instance.Windows: Still creating... [40s elapsed]
aws_instance.Windows: Creation complete after 47s [id=i-0e4614568a9e4158c]

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



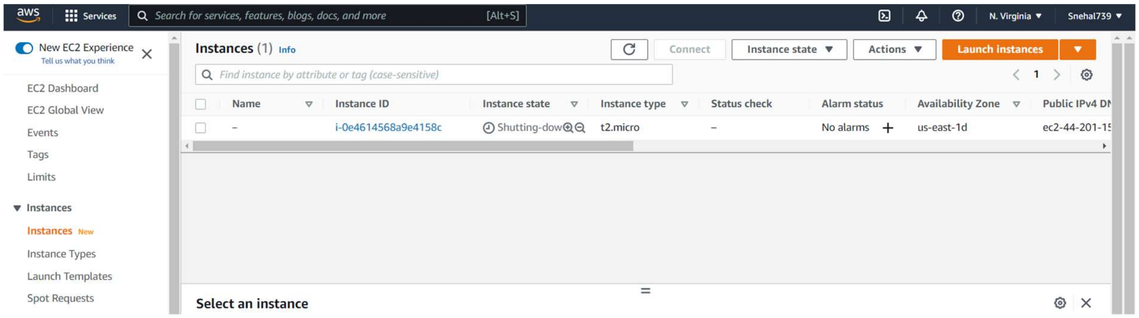
□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DN
□	–	i-0e4614568a9e4158c	✓ Running	t2.micro	Initializing	No alarms +	us-east-1d	ec2-44-201-15



```
PS C:\TF files> terraform destroy
aws_instance.Windows: Refreshing state... [id=i-0e4614568a9e4158c]

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.Windows will be destroyed
  - resource "aws_instance" "Windows" {
      - ami                                  = "ami-0c95d38b24a19de18" -> nul
l
      - arn                                  = "arn:aws:ec2:us-east-1:5330904
83721:instance/i-0e4614568a9e4158c" -> null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "us-east-1d" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 1 -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
```



```
        - volume_size           = 30 -> null
        - volume_type           = "gp2" -> null
      }
  }

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.Windows: Destroying... [id=i-0e4614568a9e4158c]
aws_instance.Windows: Still destroying... [id=i-0e4614568a9e4158c, 10s elapse
d]
aws_instance.Windows: Still destroying... [id=i-0e4614568a9e4158c, 20s elapse
d]
aws_instance.Windows: Still destroying... [id=i-0e4614568a9e4158c, 30s elapse
d]
aws_instance.Windows: Destruction complete after 32s

Destroy complete! Resources: 1 destroyed.
PS C:\TF files> 
```



Instances (1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DN
–	i-0e4614568a9e4158c	Shutting-dow	t2.micro	–	No alarms +	us-east-1d	ec2-44-201-15