Terraform Assignment - 2

You have been asked to:
Destroy the previous deployment
Create a new EC2 instance with an Elastic IP

Answer

```
sudo terraform destroy
```

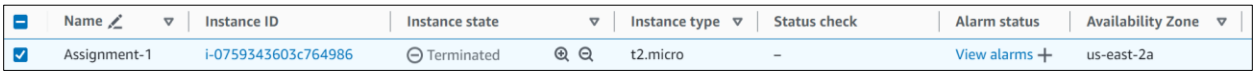
```
Plan: 0 to add, 0 to change, 1 to destroy.

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.Ins1: Destroying... [id=i-0759343603c764986]
aws_instance.Ins1: Still destroying... [id=i-0759343603c764986, 10s elapsed]
aws_instance.Ins1: Still destroying... [id=i-0759343603c764986, 20s elapsed]
aws_instance.Ins1: Destruction complete after 30s

Destroy complete! Resources: 1 destroyed.
ubuntu@ip-172-31-10-221:~/assignment1$
```



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
Assignment-1	i-0759343603c764986	Terminated	t2.micro	-	View alarms	us-east-2a

Destroyed the previous deployment successfully.

```
$ mkdir assignment2
$ cd assignment2
$ cp ../assignment1/var.tr .
$ vim main.tr

provider "aws" {
        region = "us-east-2"
        secret_key = var.secret
        access_key = var.access
}
resource "aws_instance" "Ins2" {
        ami = "ami-05fb0b8c1424f266b"
        instance_type = "t2.micro"
        key_name = "keypair_ravi"
        tags = {
                Name = "Assignment-2"
        }
}
```

```
resource "aws_eip" "Elastic-IP" {
        vpc = true
}
resource "aws_eip_association" "ElasticIP-Assocn" {
  instance_id   = aws_instance.Ins2.id
  allocation_id = aws_eip.Elastic-IP.id
}
```

```
ubuntu@ip-172-31-10-221:~/assignment2$ sudo vim main.tf
ubuntu@ip-172-31-10-221:~/assignment2$ sudo terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.33.0

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.
```

```
sudo terraform plan
```

```
Plan: 3 to add, 0 to change, 0 to destroy.

 Warning: Argument is deprecated

   with aws_eip.Elastic-IP,
   on main.tf line 15, in resource "aws_eip" "Elastic-IP":
   15:         vpc = true

 use domain attribute instead

 (and one more similar warning elsewhere)
```

```
sudo terraform apply
```

```
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.Ins2: Creating...
aws_eip.Elastic-IP: Creating...
aws_eip.Elastic-IP: Creation complete after 0s [id=eipalloc-0b1d9dc77f8a1f4cd]
aws_instance.Ins2: Still creating... [10s elapsed]
aws_instance.Ins2: Still creating... [20s elapsed]
aws_instance.Ins2: Still creating... [30s elapsed]
aws_instance.Ins2: Creation complete after 32s [id=i-050a19b2f8f409506]
aws_eip_association.ElasticIP-Assocn: Creating...
aws_eip_association.ElasticIP-Assocn: Creation complete after 1s [id=eipassoc-0b4a36c917684af5f]

  Warning: Argument is deprecated

    with aws_eip.Elastic-IP,
    on main.tf line 15, in resource "aws_eip" "Elastic-IP":
    15:         vpc = true

  use domain attribute instead


Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
```

Successfully Created the EC2 instance with EIP