Terraform Assignment - 3

You have been asked to:
Destroy the previous deployment
Create 2 EC2 instances in Ohio and N.Virginia respectively
Rename Ohio's instance to 'hello-ohio' and Virginia's instance to 'hello-virginia'

Answer

```
sudo terraform destroy
```

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


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_eip_association.ElasticIP-Assocn: Destroying... [id=eipassoc-0b4a36c917684af5f]
aws_eip_association.ElasticIP-Assocn: Destruction complete after 1s
aws_instance.Ins2: Destroying... [id=i-050a19b2f8f409506]
aws_eip.Elastic-IP: Destroying... [id=eipalloc-0b1d9dc77f8a1f4cd]
aws_eip.Elastic-IP: Destruction complete after 1s
aws_instance.Ins2: Still destroying... [id=i-050a19b2f8f409506, 10s elapsed]
aws_instance.Ins2: Still destroying... [id=i-050a19b2f8f409506, 20s elapsed]
aws_instance.Ins2: Destruction complete after 30s

Destroy complete! Resources: 3 destroyed.
```

Destroyed Successfully.

```
mkdir assignment3
cd assignment3
cp ../assignment2/var.tf .

vim main.tf

provider "aws" {
        alias = "Ohio"
        region = "us-east-2"
        secret_key = var.secret
        access_key = var.access
```

```
}
provider "aws" {
        alias = "NV"
        region = "us-east-1"
        secret_key = var.secret
        access_key = var.access
}
resource "aws_instance" "Ins3-1" {
        provider = aws.Ohio
        ami = "ami-05fb0b8c1424f266b"
        instance_type = "t2.micro"
        key_name = "keypair_ravi"
        tags = {
                Name = "Hello-Ohio"
        }
}
resource "aws_instance" "Ins3-2" {
        provider = aws.NV
        ami = "ami-0c7217cdde317cfec"
        instance_type = "t2.micro"
        key_name = "MyKeyPair-Ravi"
        tags = {
                Name = "Hello-Virginia"
        }
}
```

```
ubuntu@ip-172-31-10-221:~/assignment3$ sudo terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.33.0...
- Installed hashicorp/aws v5.33.0 (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.
```

```
sudo terraform plan
sudo terraform 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.Ins3-2: Creating...
aws_instance.Ins3-2: Still creating... [10s elapsed]
aws_instance.Ins3-2: Still creating... [20s elapsed]
aws_instance.Ins3-2: Still creating... [30s elapsed]
aws_instance.Ins3-2: Still creating... [40s elapsed]
aws_instance.Ins3-2: Creation complete after 42s [id=i-07499ea28647260b1]

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

Completed