Comprobación de la instalación de Terraform

```
+ provider registry.terraform.io/hashicorp/aws v4.67.0
PS C:\Users\pc\Documents\2023-I\CLOUD_COMPUTING\Tercer_Corte\learn-terraform-aws-instance> terraform --version
Terraform v1.4.6
on windows_amd64
+ provider registry.terraform.io/hashicorp/aws v4.67.0
```

Configuración de las variables de entorno

Archivo **main.tf**

```terraform
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 4.16"
    }
  }

  required_version = ">= 1.2.0"
}

provider "aws" {
  region  = "us-west-2"
}

resource "aws_instance" "app_server1" {
  ami           = "ami-830c94e3"
  instance_type = "t2.micro"

  tags = {
    Name = "ExampleAppServerInstance-${var.username}"
  }
}

resource "aws_instance" "app_server2" {
  ami           = "ami-830c94e3"
  instance_type = "t2.micro"

  tags = {
    Name = "ExampleAppServerInstance-${var.username}"
  }
}

variable "username" {
  description = "Usuario de kevin Gómez Cantillo en AWS"
  default     = "kevin-sys"
}
```

Comandos necesarios para crear las dos instancias de EC2 con Terraform

```
Windows PowerShell
PS C:\Users\pc\Documents\2023-I\CLOUD_COMPUTING\Tercer_Corte\learn-terraform-aws-instance> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 4.16"...
- Installing hashicorp/aws v4.67.0...
- Installed hashicorp/aws v4.67.0 (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.
PS C:\Users\pc\Documents\2023-I\CLOUD_COMPUTING\Tercer_Corte\learn-terraform-aws-instance> terraform fmt
main.tf
PS C:\Users\pc\Documents\2023-I\CLOUD_COMPUTING\Tercer_Corte\learn-terraform-aws-instance> terraform validate
Success! The configuration is valid.
```
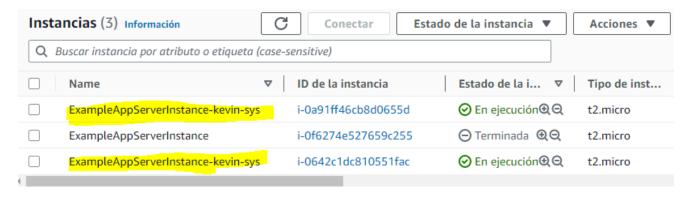}

Visualización de las dos instancias de EC2 en AWS

Por ultimo usamos el comando **terraform destroy** que como su nombre lo indica destruirá las dos instancias de EC2 que hicimos en el paso anterior.

```
Plan: 0 to add, 0 to change, 2 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.app_server2: Destroying... [id=i-0a91ff46cb8d0655d]
aws_instance.app_server1: Destroying... [id=i-0642c1dc810551fac]
aws_instance.app_server1: Still destroying... [id=i-0642c1dc810551fac, 10s elapsed]
aws_instance.app_server2: Still destroying... [id=i-0a91ff46cb8d0655d, 10s elapsed]
aws_instance.app_server2: Still destroying... [id=i-0a91ff46cb8d0655d, 21s elapsed]
aws_instance.app_server1: Still destroying... [id=i-0642c1dc810551fac, 21s elapsed]
aws_instance.app_server1: Destruction complete after 22s
aws_instance.app_server2: Still destroying... [id=i-0a91ff46cb8d0655d, 31s elapsed]
aws_instance.app_server2: Destruction complete after 33s

Destroy complete! Resources: 2 destroyed.
```

Y en AWS vemos que las instancias fueron detenidas