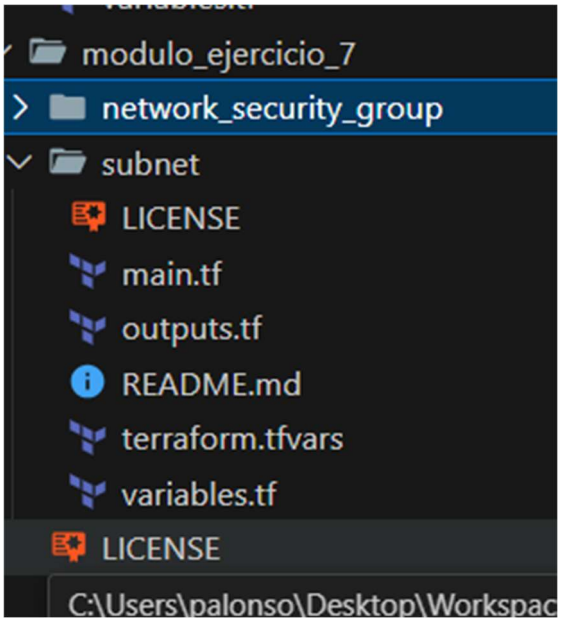
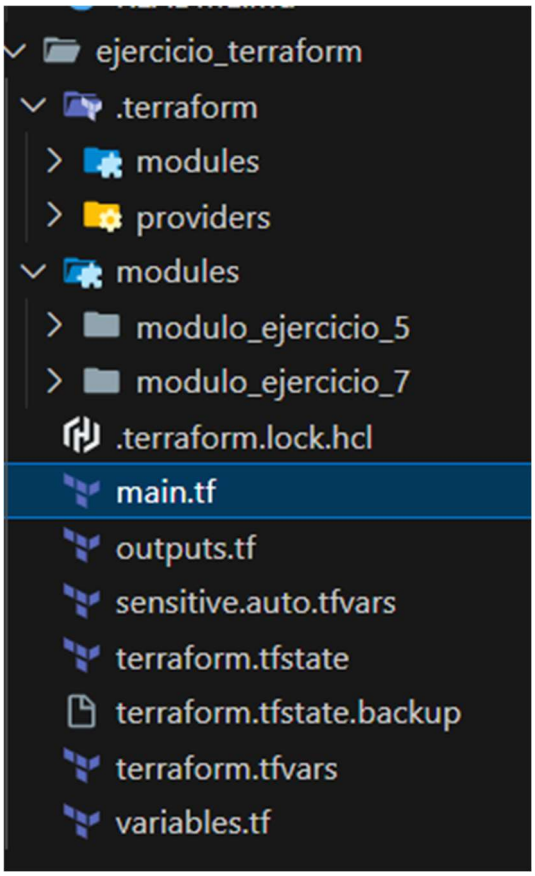
ENUNCIADO 7

Primero la estructura:



Yo tengo una estructura donde tengo un main.tf que viene de la raíz del proyecto desde donde llamo a todos y hago pruebas.



Desde aquí llamaré a los módulos de esta manera:

```
#PRUEBA DEL MODULO EJERCICIO 7

module "modulo_ejercicio_7" {
  source                = "./modules/modulo_ejercicio_7"
  resource_group_name   = var.existent_resource_group_name
  location              = var.location
  vnet_name             = var.vnet_name
  vnet_address_space    = var.vnet_address_space

}
```

He querido dejar que el modulo maneje sus propias variables y sus asignaciones para que sea modular, pensando en el módulo como una funcionalidad que puede ser implementada en diferentes contextos por lo tanto él mismo es responsable de sus datos.

Entonces desde aquí solamente paso las variables más generales.

```
# Creación de subnets
module "subnet" {
  source         = "./subnet"
  vnet_name      = azurerm_virtual_network.example.name
  resource_group = azurerm_resource_group.rg.name
  subnets        = var.subnets
}

module "network_security_group" {
  source              = "./network_security_group"
  resource_group = azurerm_resource_group.rg.name
  network_security_groups = var.network_security_groups
  location = var.location
}
```

Dentro de módulo ejercicio 7 llamo a los submodulos que se encargarán de crear las subnets y el que se encargará de asociarla a un grupo de seguridad.

En sus variables permito que puedan ser opcionales :

```
variable "subnets" {
  description = "Configuración de las subnets."
  type = map(object({
    name           = string
    address_prefix = string
  }))
  default = {}
#al ser un map permite que haya 0 o varias subnet
```

```
}


variable "network_security_groups" {
  description = "Configuración de los grupos de seguridad de red y sus
asociaciones con subnets."
  type        = map(object({
    name    = string
    rules   = list(object({
      name                       = string
      priority                   = number
      direction                  = string
      access                     = string
      protocol                   = string
      source_port_range          = string
      destination_port_range     = string
      source_address_prefix      = string
      destination_address_prefix = string
    }))
  }))
  default     = {}
}
```

Como indica el ejercicio

En el nested module subnet:

```
resource "azurerm_subnet" "subnet" {
  for_each          = var.subnets
  name              = each.value.name
  resource_group_name = var.resource_group
  virtual_network_name = var.vnet_name
  address_prefixes  = [each.value.address_prefix]
}
```

Creo las subnets

Y en el de security_group:

```
resource "azurerm_network_security_group" "nsg" {
  for_each         = var.network_security_groups
  name             = each.value.name
  location         = var.location
  resource_group_name = var.resource_group

  dynamic "security_rule" {
    for_each = each.value.rules
    content {
```

```
      name                       = security_rule.value.name
      priority                   = security_rule.value.priority
      direction                  = security_rule.value.direction
      access                     = security_rule.value.access
      protocol                   = security_rule.value.protocol
      source_port_range          = security_rule.value.source_port_range
      destination_port_range     =
security_rule.value.destination_port_range
      source_address_prefix      =
security_rule.value.source_address_prefix
      destination_address_prefix =
security_rule.value.destination_address_prefix
    }
  }
}
```

Creo los grupos.

Dentro del main root del modulo creo los recursos y asocio a una subnet un grupo de seguridad de red:

```
# Asociación de subnets con grupos de seguridad de red
resource "azurerm_subnet_network_security_group_association"
"subnet_nsg_association" {
  for_each                  = var.subnets

  subnet_id                 =
module.subnet.subnet_ids[each.value.subnet_ids]
  network_security_group_id =
module.network_security_group.network_security_group_ids[each.value.secur
ity_group]
}
```

Y al hacer 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:

```
# azurerm_resource_group.rg will be created
+ resource "azurerm_resource_group" "rg" {
+ id       = (known after apply)
+ location = "westus2"
+ name     = "myTFResourceGroup"
}

# azurerm_virtual_network.example will be created
+ resource "azurerm_virtual_network" "example" {
+ address_space      = [
+ "10.0.0.0/16",
]
+ dns_servers        = (known after apply)
+ guid               = (known after apply)
+ id                 = (known after apply)
+ location           = "westeurope"
+ name               = "vnetpalonsotfexercise02"
+ resource_group_name = "myTFResourceGroup"
+ subnet             = (known after apply)
+ tags               = {
+ "environment_tag" = "dev"
+ "owner_tag"       = "pilarAlonsoSTEMDO"
}
}

# module.modulo_ejercicio_5.azurerm_virtual_network.example will be created
+ resource "azurerm_virtual_network" "example" {
+ address_space      = [
+ "10.0.0.0/16",
]
```

```
      + dns_servers         = (known after apply)
      + guid                = (known after apply)
      + id                  = (known after apply)
      + location            = "westeurope"
      + name                = "vnetpalonsotfexercise02"
      + resource_group_name = "myTFResourceGroup"
      + subnet              = (known after apply)
      + tags                = {
          + "environment" = "dev"
          + "owner"       = "pilarAlonsoSTEMDO"
        }
    }

  # module.modulo_ejercicio_7.azurerm_resource_group.rg will be created
  + resource "azurerm_resource_group" "rg" {
      + id       = (known after apply)
      + location = "westeurope"
      + name     = "myTFResourceGroup"
    }

  # module.modulo_ejercicio_7.azurerm_virtual_network.example will be created
  + resource "azurerm_virtual_network" "example" {
      + address_space       = [
          + "10.0.0.0/16",
        ]
      + dns_servers         = (known after apply)
      + guid                = (known after apply)
      + id                  = (known after apply)
      + location            = "westeurope"
      + name                = "vnetpalonsotfexercise02"
      + resource_group_name = "myTFResourceGroup"
```

```
      + subnet              = (known after apply)
    }

  # module.modulo_ejercicio_5.module.mi_submodulo_ejercicio_5.azurerm_virtual_network.example will be created
  + resource "azurerm_virtual_network" "example" {
      + address_space       = [
          + "10.0.0.0/16",
        ]
      + dns_servers         = (known after apply)
      + guid                = (known after apply)
      + id                  = (known after apply)
      + location            = "westeurope"
      + name                = "vnetpalonsotfexercise02"
      + resource_group_name = "myTFResourceGroup"
      + subnet              = (known after apply)
      + tags                = {
          + "environment" = "dev"
          + "owner"       = "pilarAlonsoSTEMDO"
        }
    }
```