Se former autrement

HONORIS UNITED UNIVERSITIES

Terraform –MS Azure

Initiation à l'automatisation

4ArcTIC

2024-2025

Introduction :

Dans ce TP, nous allons apprendre à utiliser Terraform pour automatiser la création et la gestion de ressources sur Microsoft Azure.

Prérequis :

- Un compte **Azure for Students** actif.
- Une machine sous **Windows** avec une connexion Internet stable

Objectif :

L'objectif de cet atelier est de :

- ✓ Installer Terraform sur une machine Windows
- ✓ Installer et configurer la CLI Azure sur une machines Windows.
- ✓ Automatiser la création d'un ressource group
- ✓ Automatiser le déploiement d'une instance.

Partie 1 : Installation des outils

I- Installation de Terraform

Pour installer Terraform sur votre machine Windows vous devez :

1. Télécharger l'outils qui correspond à votre architecture matérielle à partir de ce lien.
2. Extraire l'exécutable et l'enregistrer dans un emplacement donné (par exemple c:\Apps\terraform.exe)
3. Rajouter l'emplacement de l'exécutable terraform.exe à votre variable d'environnement path comme suit :
    - Sur votre panneau de configuration, allez vers system ➔ system settings ➔ Environment Variables.
    - Dans la section variables systèmes, cherchez la variable Path.
    - Cliquez sur le bouton modifier
    - Rajouter le chemin d'accès vers le fichier terraform.exe (par exemple c:\Apps\terraform.exe)

Vous pouvez suivre cet vidéo.

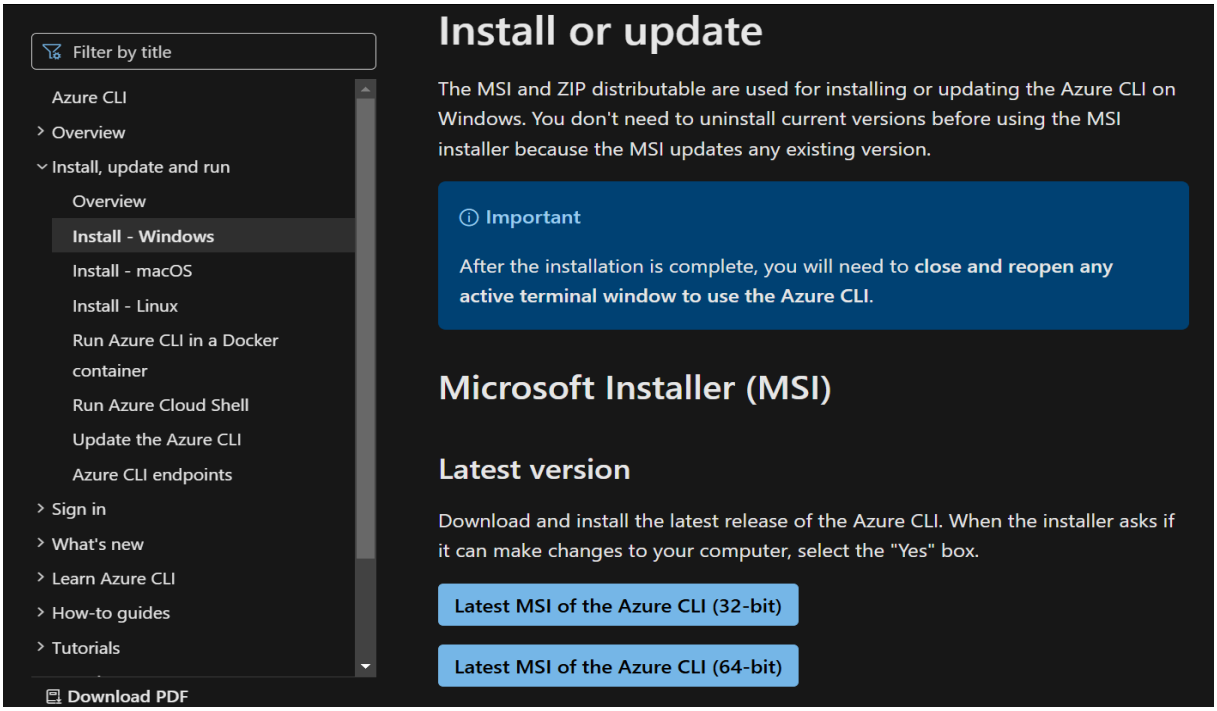
4. Ouvrez **PowerShell** et vérifiez l'installation avec :

```
terraform -version
```

Remarque : Si Terraform n'est pas reconnu, redémarrez votre machine.

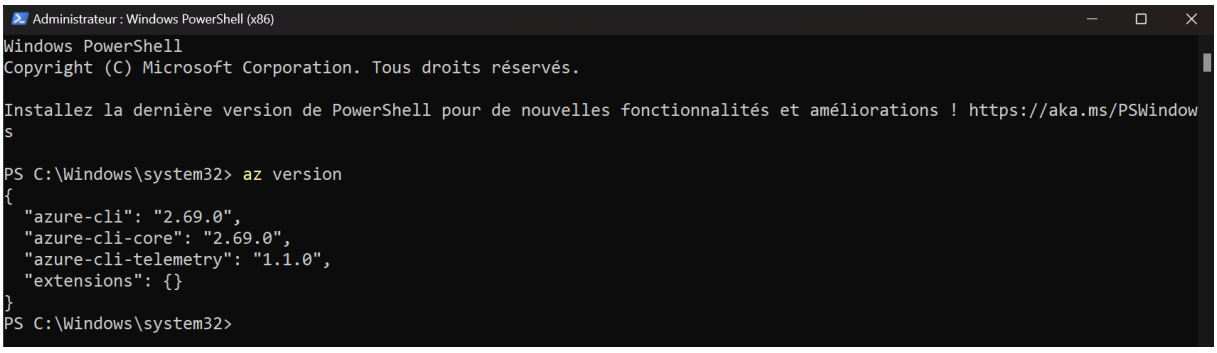
II- Installation et configuration de Azure CLI

1. Téléchargez et installez **Azure CLI** depuis ce lien.



2. Vérifiez l'installation avec :

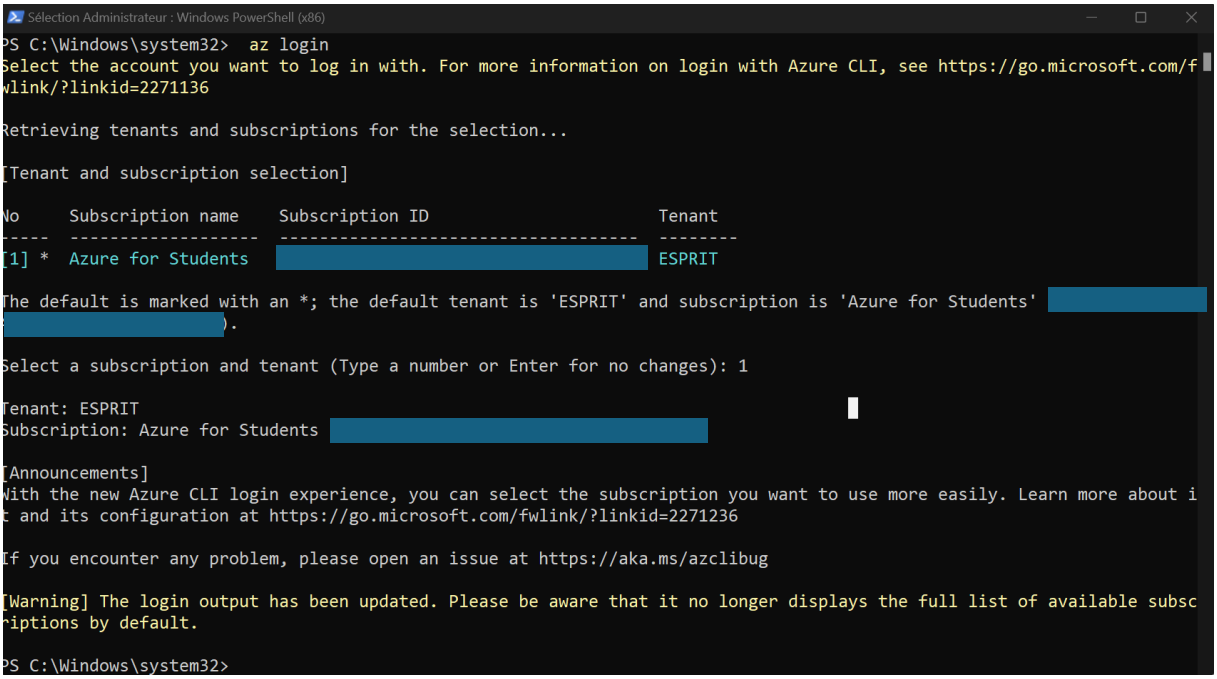
```
az version
```



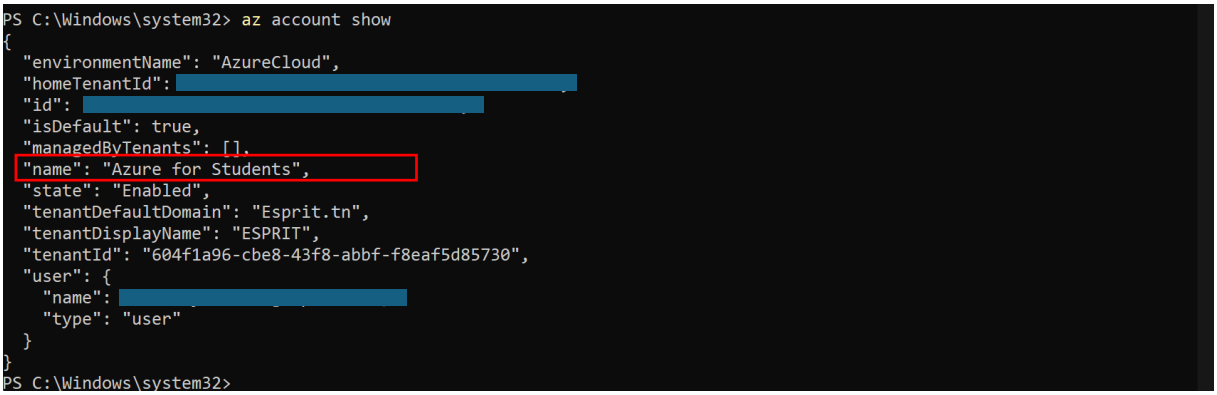
3. Connectez-vous à Azure :

az login

- Une page web s'ouvrira, connectez-vous avec votre **compte étudiant Azure**.
- Après connexion, les informations relatives à votre compte vont être affichées comme suit :



4. Vérifiez que l'abonnement actif est bien Azure for Students :



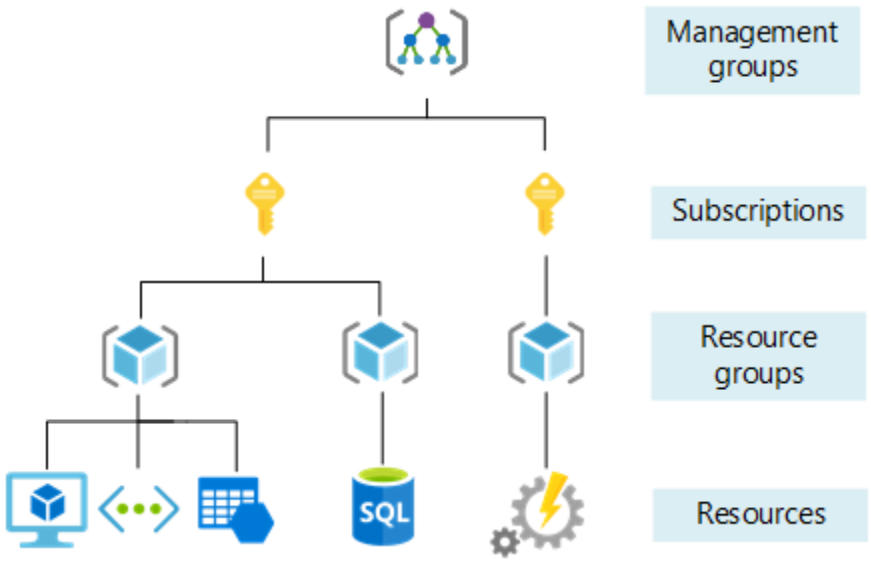
Si vous disposez de plusieurs souscriptions et vous voulez utiliser la souscription Azure for students, vous devez utiliser la commande suivante :

az account set --subscription "Azure for Students"

Get details about a specified subscription.

```
PS C:\Windows\system32> az account subscription list
Preview version of extension is disabled by default for extension installation, enabled for modules without stable versi
ons.
Please run 'az config set extension.dynamic_install_allow_preview=true or false' to config it specifically.
The command requires the extension account. Do you want to install it now? The command will continue to run after the ex
tension is installed. (Y/n): y
Run 'az config set extension.use_dynamic_install=yes_without_prompt' to allow installing extensions without prompt.
Command group 'account subscription' is experimental and under development. Reference and support levels: https://aka.ms
/CLI_refstatus
[
  {
    "authorizationSource": "RoleBased",
    "displayName": "Azure for Students",
    "id": "/subscriptions/[redacted]",
    "state": "Enabled",
    "subscriptionId": [redacted],
    "subscriptionPolicies": {
      "locationPlacementId": "Public_2014-09-01",
      "quotaId": "AzureForStudents_2018-01-01",
      "spendingLimit": "On"
    }
  }
]
PS C:\Windows\system32> az account subscription show
```

Avant de continuer, il est essential de savoir que les ressources crées et gérées par MS Azure sont organisées d'une manière hiérarchique à la manière d'un annuaire Active Directory. Pour pouvoir créer des ressources tel que les instances, espace de stockage, réseaux, volumes… , il est nécessaire de créer d'abord les ressources conteneurs comme expliquer ci-dessous :



Pour en savoir plus, consulter la documentation Microsoft Azure.

Partie 2 : Automatisation avec Terraform

I- Automatisation de la création d'un ressource group

Le groupe de ressources Azure est un conteneur logique dans lequel les ressources Azure telles que les applications Web, les bases de données et les comptes de stockage sont déployées et gérées. Dans cette partie du Lab, nous allons utiliser Terraform pour automatiser la création d'un groupe de ressources.

1. Création d'un projet Terraform :

```
PS C:\Windows\system32> mkdir terraform-azure


    Répertoire : C:\Windows\system32


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        22/02/2025     10:54                terraform-azure


PS C:\Windows\system32> cd terraform-azure
PS C:\Windows\system32\terraform-azure>
```

2. Dans ce répertoire, créez un fichier main.tf et ajoutez la configuration suivante :

```
terraform {
  required_version = ">=0.12"
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "~>4.20"
    }
  }
}
provider "azurerm" {
  features {}
  subscription_id = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
}
resource "random_pet" "rg-name" {
  prefix = var.resource_group_name_prefix
}
resource "azurerm_resource_group" "rg" {
  name     = random_pet.rg-name.id
  location = var.resource_group_location
}
```

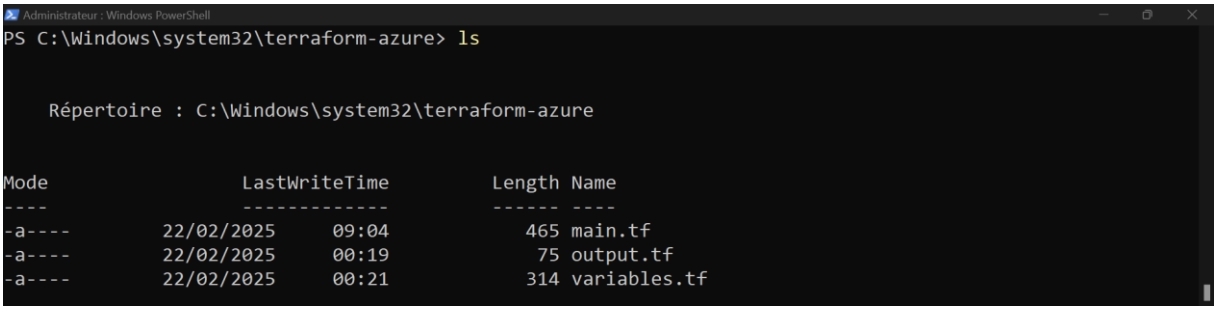
Remarque : Obtenez l'ID de votre abonnement avec la commande suivante : **az account show --query id -o tsv**, puis, remplacez xxxxxxxxx... par la valeur retournée.

3. Pour cet exemple, le préfixe à utiliser pour les noms des ressources groupe ainsi que la localisation seront introduit comme variables grâce à l'utilisation du fichier variables.tf comme suit :

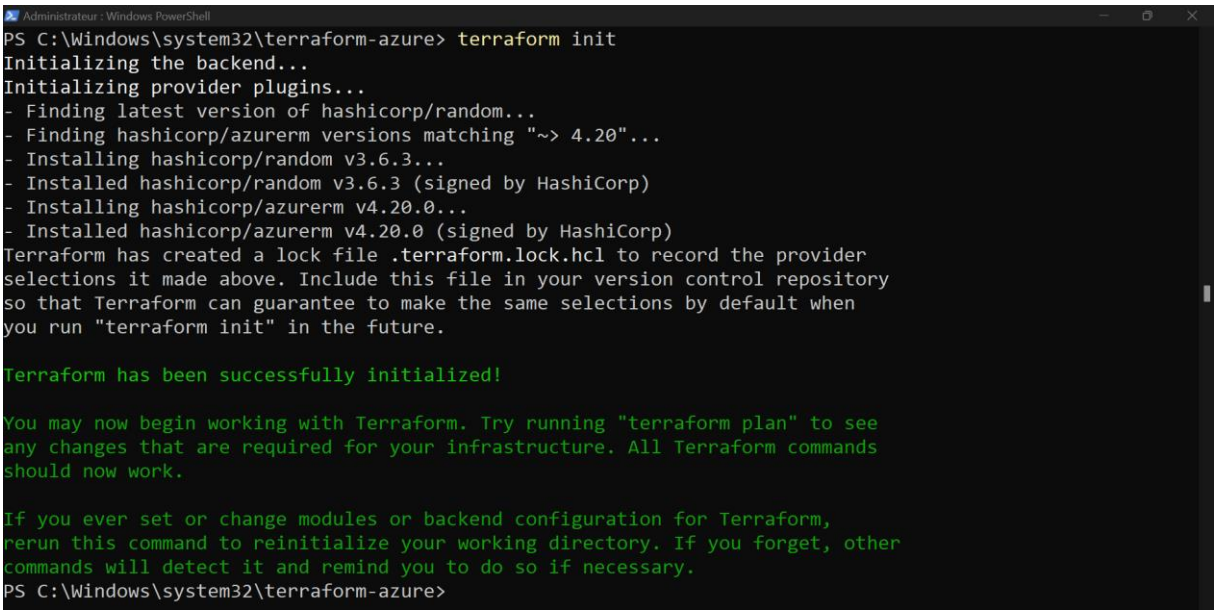
```
variable "resource_group_name_prefix" {
  default     = "rg"
  description = "Prefix of the resource group name that's combined with a random ID so name is unique in your Azure subscription."
}
variable "resource_group_location" {
  default     = "eastus"
  description = "Location of the resource group."
}
```

4. Enfin, pour configurer l'output à afficher à la fin de la création du ressource group, nous allons utiliser le fichier output.tf comme ci-dessous :

```
output "resource_group_name" {
  value = azurerm_resource_group.rg.name
}
```



5. Nous allons par la suite initialiser Terraform avec la commande : **Terraform init**



```
PS C:\Windows\system32\terraform-azure> ls


    Répertoire : C:\Windows\system32\terraform-azure


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        22/02/2025     11:04                .terraform
-a----        22/02/2025     11:05           2233 .terraform.lock.hcl
-a----        22/02/2025     09:04            465 main.tf
-a----        22/02/2025     00:19             75 output.tf
-a----        22/02/2025     00:21            314 variables.tf
```

6. Pour formater les fichiers de configuration dans l'espace de travail, nous allons utiliser la commande : **terraform fmt**

```
Administrateur : Windows PowerShell
PS C:\Windows\system32\terraform-azure> terraform fmt
PS C:\Windows\system32\terraform-azure>
```

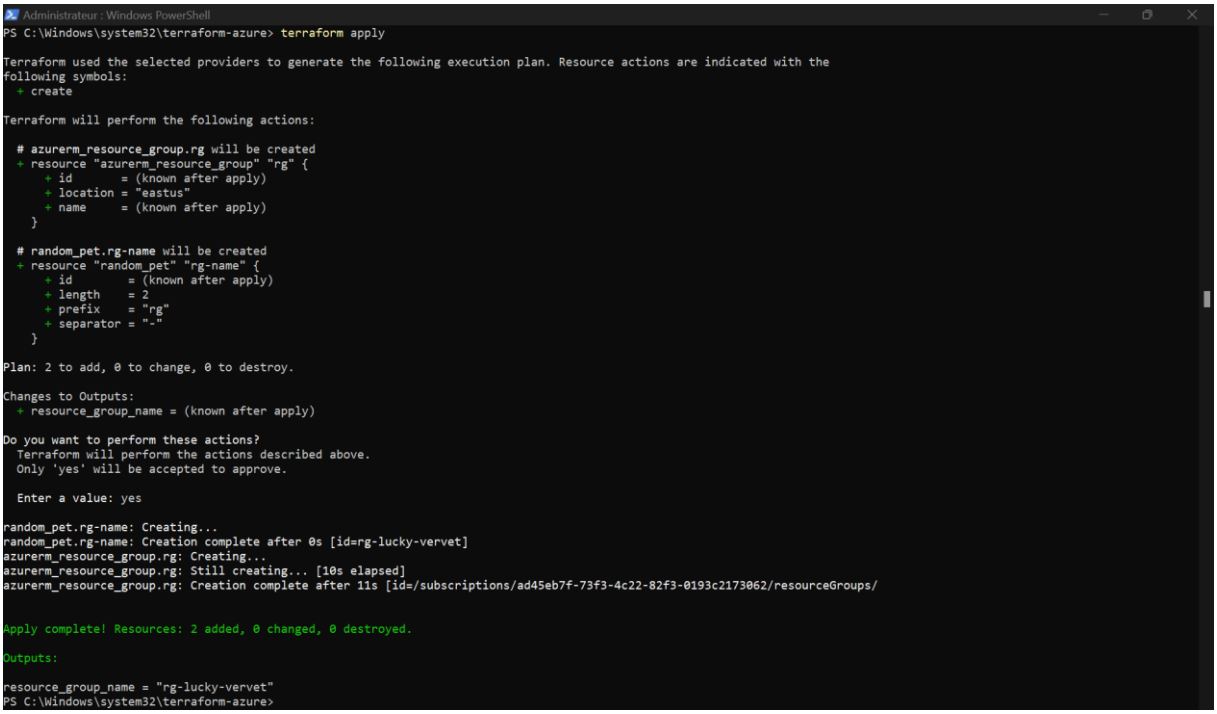
7. Pour faire la vérification syntaxique des fichiers de configuration, nous allons utiliser la commande : **terraform validate**

```
PS C:\Windows\system32\terraform-azure> terraform validate
Success! The configuration is valid.

PS C:\Windows\system32\terraform-azure>
```

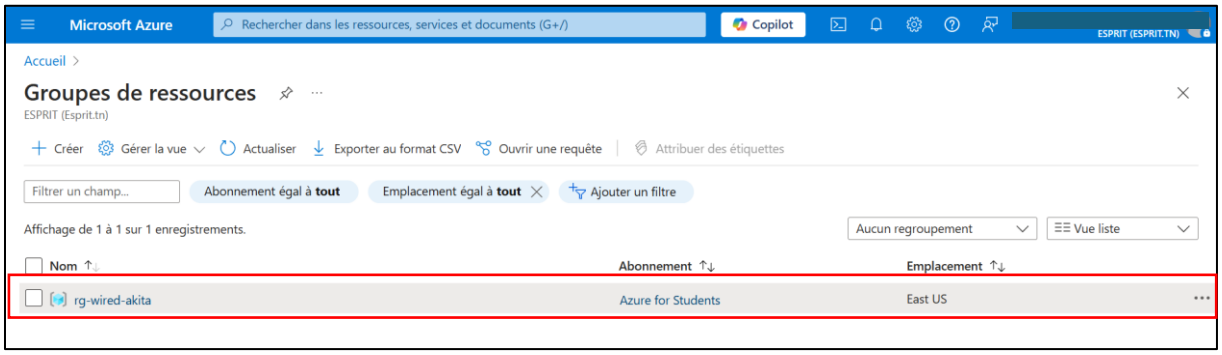
8. Il est maintenant possible de passer à la phase de génération du plan avec la commande : **terraform plan**

```
Administrateur : Windows PowerShell
PS C:\Windows\system32\terraform-azure> terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the follow
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 = "eastus"
      + name     = (known after apply)
    }

  # random_pet.rg-name will be created
  + resource "random_pet" "rg-name" {
      + id        = (known after apply)
      + length    = 2
      + prefix    = "rg"
      + separator = "-"
    }

Plan: 2 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + resource_group_name = (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:\Windows\system32\terraform-azure>
```

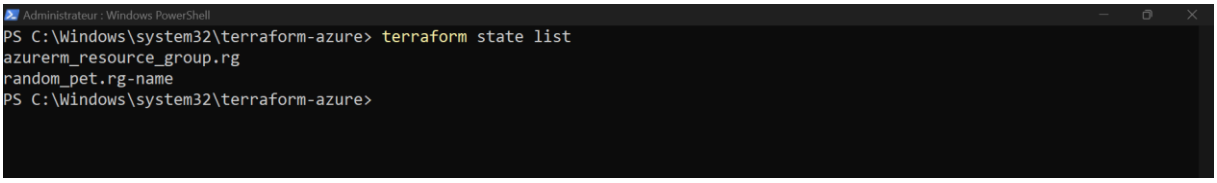
9. Enfin pour appliquer ces modifications, la commande suivante doit être exécutée : **terraform apply**



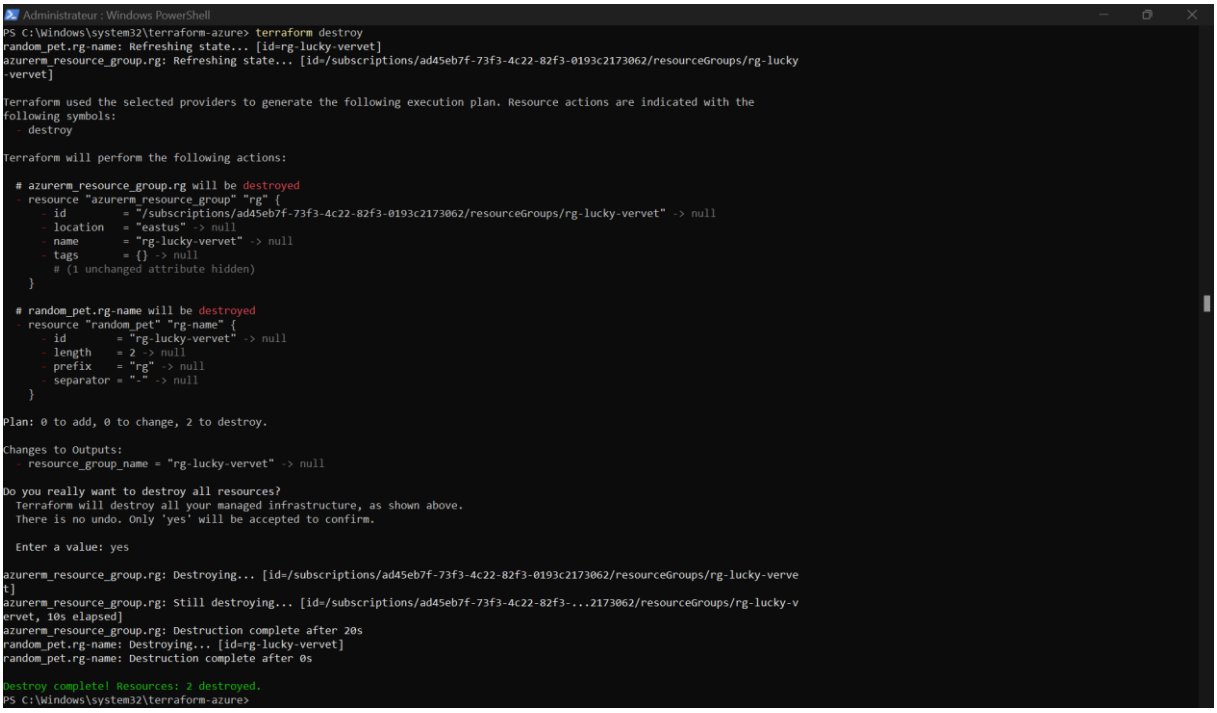
Il est possible de vérifier que le ressource group a bien été créer à travers le portail Azure :



Il est aussi possible de lister les ressources crées à partir de Terraform avec la commande : **terraform state list**



10. Enfin, pour supprimer les ressources que nous venons de créer, nous allons utiliser la commande : **terraform destroy**



II- Automatisation d'une Machine Virtuelle

1. Création d'un projet Terraform-VM :

2. Ajoutez cette configuration à main.tf pour créer une VM :

```
# Configure the Microsoft Azure Provider
terraform {
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "~>4.20"
    }
  }
}
```

```
provider "azurerm" {
  features {}
  subscription_id = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
}
# Create a resource group if it doesn't exist
resource "azurerm_resource_group" "myterraformgroup" {
  name     = "myResourceGroupTerraform"
  location = "eastus"
  tags = {
    environment = "Terraform Demo"
  }
}
# Create virtual network
resource "azurerm_virtual_network" "myterraformnetwork" {
  name                = "myVnet"
  address_space       = ["10.0.0.0/16"]
  location            = "eastus"
  resource_group_name = azurerm_resource_group.myterraformgroup.name
  tags = {
    environment = "Terraform Demo"
  }
}
# Create subnet
resource "azurerm_subnet" "myterraformsubnet" {
  name                 = "mySubnet"
  resource_group_name  = azurerm_resource_group.myterraformgroup.name
  virtual_network_name = azurerm_virtual_network.myterraformnetwork.name
  address_prefixes     = ["10.0.1.0/24"]
}
# Create public IPs
resource "azurerm_public_ip" "myterraformpublicip" {
  name                = "myPublicIP"
  location            = "eastus"
  resource_group_name = azurerm_resource_group.myterraformgroup.name
  allocation_method   = "Static"   # Changé de "Dynamic" à "Static"
  sku                 = "Standard" # Assurez-vous que la SKU est définie sur "Standard"
  tags = {
    environment = "Terraform Demo"
  }
} # Create Network Security Group and rule
resource "azurerm_network_security_group" "myterraformnsg" {
  name                = "myNetworkSecurityGroup"
  location            = "eastus"
  resource_group_name = azurerm_resource_group.myterraformgroup.name
  security_rule {
    name                       = "SSH"
    priority                   = 1001
    direction                  = "Inbound"
    access                     = "Allow"
    protocol                   = "Tcp"
```

```
    source_port_range          = "*"
    destination_port_range     = "22"
    source_address_prefix      = "*"
    destination_address_prefix = "*"
  }
  tags = {
    environment = "Terraform Demo"
  }
}
# Create network interface
resource "azurerm_network_interface" "myterraformnic" {
  name                = "myNIC"
  location            = "eastus"
  resource_group_name = azurerm_resource_group.myterraformgroup.name
  ip_configuration {
    name                          = "myNicConfiguration"
    subnet_id                     = azurerm_subnet.myterraformsubnet.id
    private_ip_address_allocation = "Dynamic"
    public_ip_address_id          = azurerm_public_ip.myterraformpublicip.id
  }
  tags = {
    environment = "Terraform Demo"
  }
}
# Connect the security group to the network interface
resource "azurerm_network_interface_security_group_association" "example" {
  network_interface_id      = azurerm_network_interface.myterraformnic.id
  network_security_group_id = azurerm_network_security_group.myterraformnsg.id
}
# Generate random text for a unique storage account name
resource "random_id" "randomId" {
  keepers = {
    # Generate a new ID only when a new resource group is defined
    resource_group = azurerm_resource_group.myterraformgroup.name
  }
  byte_length = 8
}
# Create storage account for boot diagnostics
resource "azurerm_storage_account" "mystorageaccount" {
  name                     = "diag${random_id.randomId.hex}"
  resource_group_name      = azurerm_resource_group.myterraformgroup.name
  location                 = "eastus"
  account_tier             = "Standard"
  account_replication_type = "LRS"
  tags = {
    environment = "Terraform Demo"
  }
}
# Create (and display) an SSH key
resource "tls_private_key" "example_ssh" {
```

```
  algorithm = "RSA"
  rsa_bits  = 4096
}
output "tls_private_key" {
  value     = tls_private_key.example_ssh.private_key_pem
  sensitive = true
}
# Create virtual machine
resource "azurerm_linux_virtual_machine" "myterraformvm" {
  name                  = "myVM"
  location              = "eastus"
  resource_group_name   = azurerm_resource_group.myterraformgroup.name
  network_interface_ids = [azurerm_network_interface.myterraformnic.id]
  size                  = "Standard_DS1_v2"
  os_disk {
    name                 = "myOsDisk"
    caching              = "ReadWrite"
    storage_account_type = "Premium_LRS"
  }
  source_image_reference {
    publisher = "Canonical"
    offer     = "UbuntuServer"
    sku       = "18.04-LTS"
    version   = "latest"
  }
  computer_name                   = "myvm"
  admin_username                  = "azureuser"
  disable_password_authentication = true
  admin_ssh_key {
    username   = "azureuser"
    public_key = tls_private_key.example_ssh.public_key_openssh
  }
  boot_diagnostics {
    storage_account_uri = azurerm_storage_account.mystorageaccount.primary_blob_endpoint
  }
  tags = {
    environment = "Terraform Demo"
  }
}
```

Remarque : Obtenez l'ID de votre abonnement avec la commande suivante : **az account show --query id -o tsv**, puis, remplacez xxxxxxxxx... par la valeur retournée.

```
PS C:\Windows\system32\terraform-VM> ls


    Répertoire : C:\Windows\system32\terraform-VM


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        22/02/2025     10:09           5064 main.tf
```

3. Faire les commandes terraform **init**, **fmt, validate**, **plan** et **apply**.
4. Vérifiez la création de la VM à travers le portail Azure.