**Terraform-based Azure Infrastructure Deployment**

**🧰 Prerequisites**

* ✅ Azure CLI installed [Download](https://learn.microsoft.com/en-us/cli/azure/install-azure-cli)
* ✅ Terraform installed Download
* ✅ Azure Subscription (with Contributor access or above)
* ✅ Code editor (e.g., VS Code)

**🪜 Step-by-Step Guide**

**🧾 Step 1: Login to Azure**

bash

CopyEdit

az login

🔐 Log into your Azure account. A browser window will open for authentication.

**📁 Step 2: Create a New Directory for Your Terraform Project**

bash

CopyEdit

mkdir terraform-azure-infra && cd terraform-azure-infra

📦 This keeps your Terraform files organized.

**📄 Step 3: Create Terraform Configuration Files**

**🔹 main.tf**

Defines the resources to provision.

hcl

CopyEdit

provider "azurerm" {

features {}

}

resource "azurerm\_resource\_group" "example" {

name = "example-resources"

location = "East US"

}

**🔹 variables.tf**

To declare input variables.

hcl

CopyEdit

variable "location" {

default = "East US"

}

**🔹 outputs.tf**

To define outputs after deployment.

hcl

CopyEdit

output "resource\_group\_name" {

value = azurerm\_resource\_group.example.name

}

**🔍 Step 4: Initialize Terraform**

bash

CopyEdit

terraform init

📦 Initializes the working directory and downloads necessary providers.

**📝 Step 5: Validate and Format the Configuration**

bash

CopyEdit

terraform fmt

terraform validate

🔍 Ensures that your config is syntactically valid and well-formatted.

**📋 Step 6: Plan the Deployment**

bash

CopyEdit

terraform plan

🔎 Shows what Terraform will do without making changes.

**🚀 Step 7: Apply the Configuration**

bash

CopyEdit

terraform apply

💥 Provisions the infrastructure. Confirm when prompted.

**📦 Step 8: Check Outputs**

bash

CopyEdit

terraform output

📄 Displays the values defined in outputs.tf.

**🧹 Step 9: Destroy the Infrastructure (When No Longer Needed)**

bash

CopyEdit

terraform destroy

⚠️ Tears down the infrastructure. Use carefully.

**📘 Example Resources to Add**

You can expand your infra by adding:

* 💻 Virtual Machines
* 🛡️ Network Security Groups
* 🌐 Virtual Networks (VNet)
* 📊 Storage Accounts
* 🔄 Load Balancers

**📁 Sample Folder Structure**

css

CopyEdit

terraform-azure-infra/

├── main.tf

├── variables.tf

├── outputs.tf

└── README.md

**👥 Deploy Azure Active Directory Users using Terraform**

⚠️ Requires Terraform **AzureAD** provider, not just azurerm

**📄 Step-by-Step**

1. **Add the AzureAD Provider**

hcl

CopyEdit

terraform {

required\_providers {

azuread = {

source = "hashicorp/azuread"

version = "~> 2.0"

}

}

}

provider "azuread" {

}

1. **Create User**

hcl

CopyEdit

resource "azuread\_user" "example" {

user\_principal\_name = "john.doe@yourdomain.onmicrosoft.com"

display\_name = "John Doe"

mail\_nickname = "johndoe"

password = "TempPassword@123" # Use Key Vault or variable

force\_password\_change = true

}

**🔐 Deploy Azure Key Vault using Terraform**

1. **Add to your main.tf**

hcl

CopyEdit

resource "azurerm\_key\_vault" "example" {

name = "exampleKeyVault123"

location = azurerm\_resource\_group.example.location

resource\_group\_name = azurerm\_resource\_group.example.name

tenant\_id = data.azurerm\_client\_config.current.tenant\_id

sku\_name = "standard"

soft\_delete\_enabled = true

purge\_protection\_enabled = true

access\_policy {

tenant\_id = data.azurerm\_client\_config.current.tenant\_id

object\_id = data.azurerm\_client\_config.current.object\_id

key\_permissions = ["Get", "List"]

secret\_permissions = ["Get", "Set", "Delete"]

}

}

data "azurerm\_client\_config" "current" {}

1. **Store a Secret**

hcl

CopyEdit

resource "azurerm\_key\_vault\_secret" "example" {

name = "exampleSecret"

value = "SuperSecretValue"

key\_vault\_id = azurerm\_key\_vault.example.id

}

**🌍 Deploy Azure Kubernetes Service (AKS) using Terraform**

1. **Required Providers & Variables**

hcl

CopyEdit

provider "azurerm" {

features {}

}

resource "azurerm\_kubernetes\_cluster" "example" {

name = "exampleAKSCluster"

location = azurerm\_resource\_group.example.location

resource\_group\_name = azurerm\_resource\_group.example.name

dns\_prefix = "exampleaks"

default\_node\_pool {

name = "default"

node\_count = 1

vm\_size = "Standard\_DS2\_v2"

}

identity {

type = "SystemAssigned"

}

tags = {

Environment = "Dev"

}

}

**☁️ Deploy Hybrid Cloud Network (VNet Peering, VPN Gateway, etc.)**

1. **Define VNets**

hcl

CopyEdit

resource "azurerm\_virtual\_network" "on\_prem\_vnet" {

name = "OnPremVNet"

address\_space = ["10.0.0.0/16"]

location = azurerm\_resource\_group.example.location

resource\_group\_name = azurerm\_resource\_group.example.name

}

resource "azurerm\_virtual\_network" "cloud\_vnet" {

name = "CloudVNet"

address\_space = ["10.1.0.0/16"]

location = azurerm\_resource\_group.example.location

resource\_group\_name = azurerm\_resource\_group.example.name

}

1. **Add VNet Peering**

hcl

CopyEdit

resource "azurerm\_virtual\_network\_peering" "onprem\_to\_cloud" {

name = "OnPremToCloud"

resource\_group\_name = azurerm\_resource\_group.example.name

virtual\_network\_name = azurerm\_virtual\_network.on\_prem\_vnet.name

remote\_virtual\_network\_id = azurerm\_virtual\_network.cloud\_vnet.id

allow\_forwarded\_traffic = true

allow\_gateway\_transit = false

}