



TP Cloud Engineering

Encadré par :  
Mme DAD Nisrine

2022/2023

# RAPPORT TP 1,2 ET 3

Réalisé par :  
LAOUINA Yassine



# Introduction

Durant les 3 premières séances du TP, j'ai appris à me familiariser avec les bases de la plateforme de Cloud computing Azure. On a découvert que le Cloud Azure offre une gamme de services et de fonctionnalités pour aider les entreprises à déployer, gérer et sécuriser leurs applications et leurs données dans le Cloud. Ce rapport vise à présenter brièvement l'utilité du Cloud Azure et les avantages qu'il peut offrir aux entreprises depuis ce qu'on a vu durant les 3 premières séances.

# Remarque :

Dans le premier chapitre du TP nous avons commencé à créer le notre compte sur portal.azure.com.

Suite à des complications hors de ma portée, j'ai eu la mauvaise souscription sur la plateforme, mes captures d'écrans ne seront donc pas les miennes.

Merci pour votre compréhension

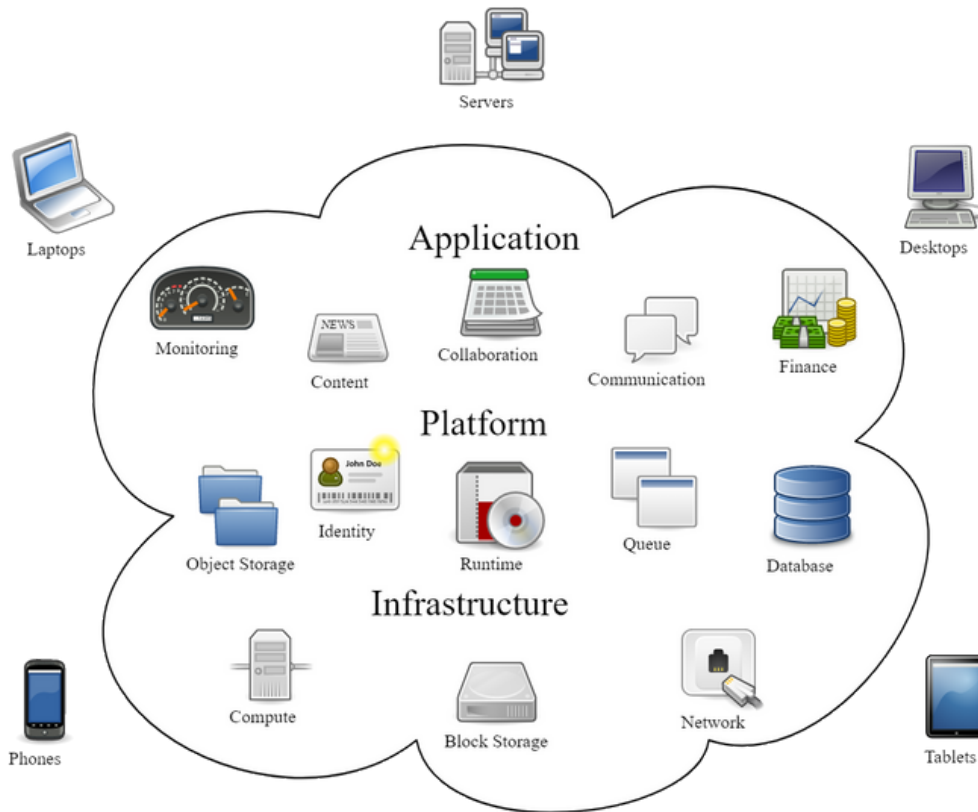


# **TP 1 :**

Les concepts de base du Cloud :

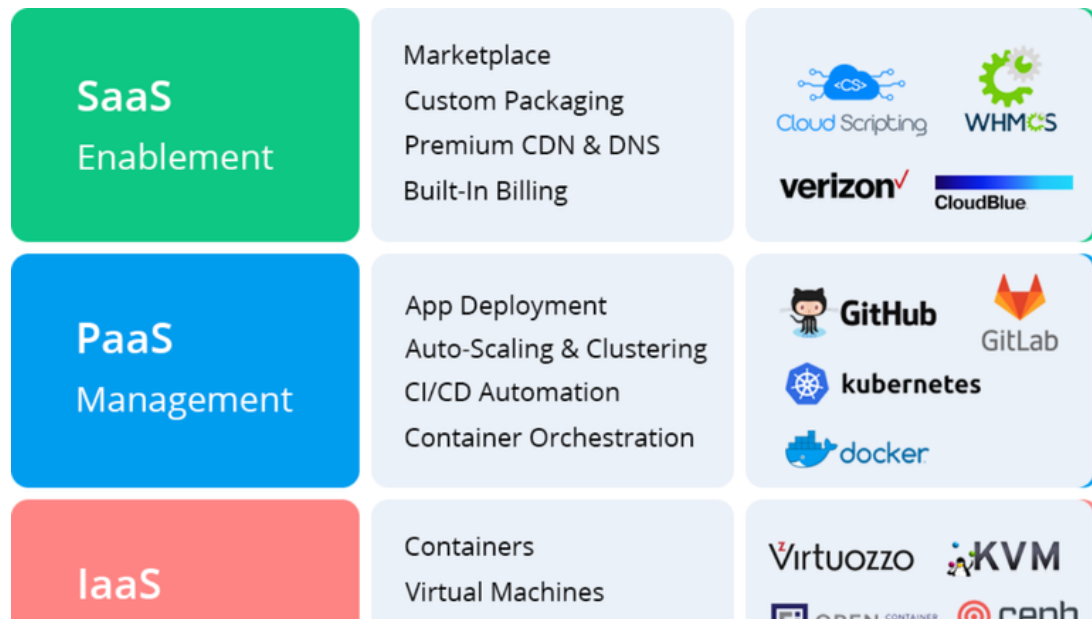
- 1.1) La définition du cloud computing
- 1.2) Les modèles de service
- 1.3) Azure CLI

# 1.1 La définition du cloud computing



Le cloud computing, également appelé informatique en nuage, est un modèle de prestation de services informatiques via Internet, permettant l'accès à des ressources informatiques à la demande, telles que des serveurs, du stockage, des bases de données, des réseaux et des logiciels, sans avoir à les posséder ou les gérer localement.

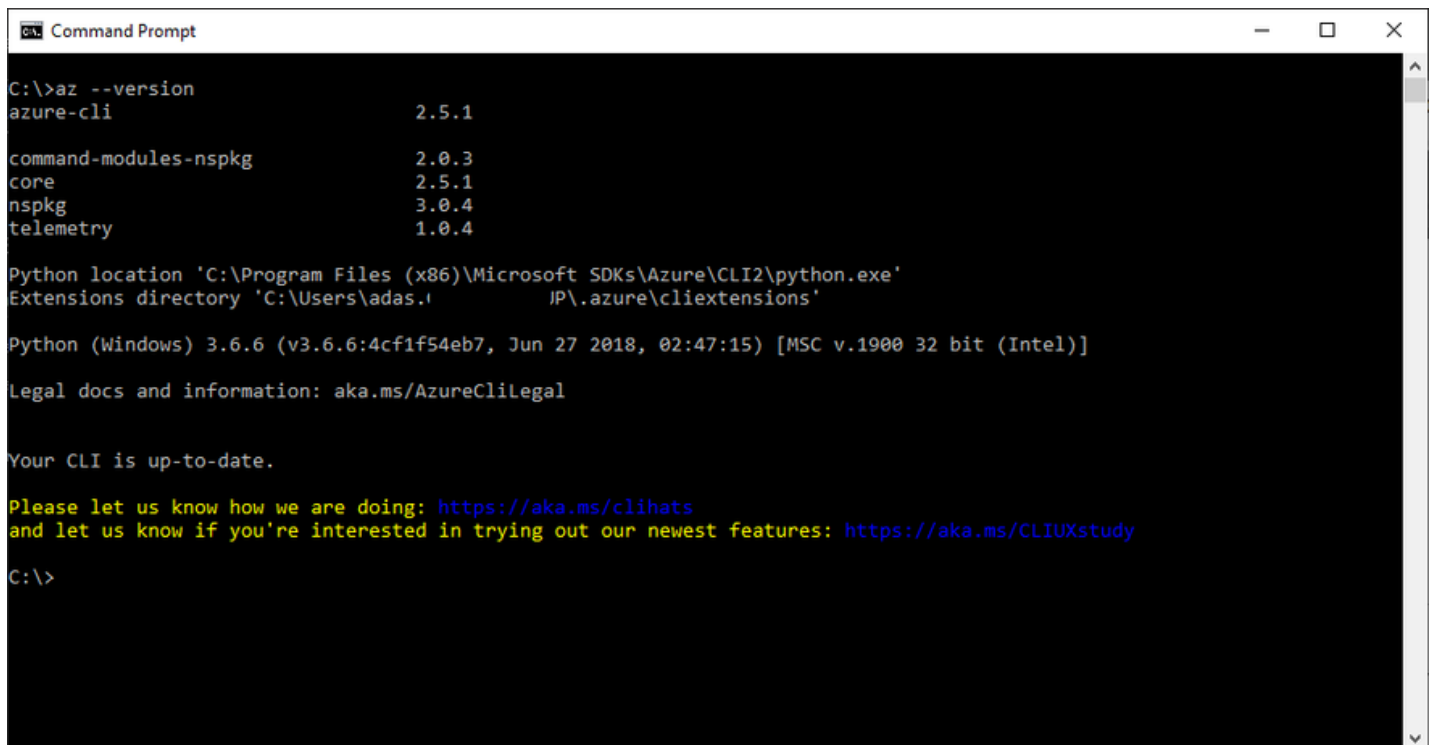
# 1.2 Les modèles de service du cloud



- **IaaS** (Infrastructure as a Service) : Il s'agit de la fourniture d'une infrastructure informatique virtuelle, telle que des serveurs virtuels, des réseaux et des espaces de stockage, via Internet.
- **PaaS** (Platform as a Service) : Il s'agit de la fourniture d'une plateforme de développement et de déploiement d'applications, qui inclut généralement des outils, des bibliothèques et des services pour faciliter le développement, le test, le déploiement et la gestion des applications.
- **SaaS** (Software as a Service) : Il s'agit de la fourniture d'applications logicielles via Internet, généralement sous la forme d'un service en ligne accessible via un navigateur web.

## 1.3 Azure CLI

Azure CLI (Azure Command-Line Interface) est un outil en ligne de commande fourni par Microsoft pour interagir avec les services Azure depuis l'invite de commandes. On peut y créer ou configurer ou supprimer l'ensemble des services ou ressources sur Azure



```
Command Prompt

C:\>az --version
azure-cli                2.5.1

command-modules-nspkg    2.0.3
core                     2.5.1
nspkg                    3.0.4
telemetry                1.0.4

Python location 'C:\Program Files (x86)\Microsoft SDKs\Azure\CLI2\python.exe'
Extensions directory 'C:\Users\adas.1\JP\azure\cliextensions'

Python (Windows) 3.6.6 (v3.6.6:4cf1f54eb7, Jun 27 2018, 02:47:15) [MSC v.1900 32 bit (Intel)]

Legal docs and information: aka.ms/AzureCliLegal

Your CLI is up-to-date.

Please let us know how we are doing: https://aka.ms/cliats
and let us know if you're interested in trying out our newest features: https://aka.ms/CLIUXstudy

C:\>
```



## **TP 2 :**

- 2.1) Créer et configurer des machines virtuelles Windows et Linux.
- 2.2) Dimensionner, mettre à l'échelle des machines virtuelles.
- 2.3) Configurer les disques et choix de redondance.
- 2.4) Monitoring

## 2.1 Création des ressources

Pour créer une VM sur Azure il faut spécifier le Nom de la machine virtuelle, Emplacement, Taille de la machine virtuelle, Disques, Système d'exploitation et Réseau.

Il existe des outils plus agnostiques, qui sont indépendants de la plateforme ou l'interface graphique et qui résistent au changement de ces derniers au cours des mises à jours.

Exemple Terraform :

Sur Terraform, les infrastructures sont décrites sous forme de code (IaC) dans un fichier de configuration déclaratif écrit en langage de description d'infrastructure (HCL). Ce fichier contient la définition des ressources, leurs configurations, leurs dépendances, et les relations entre elles. On utilise ce fichier de configuration pour planifier et créer les ressources d'infrastructure nécessaires.

# Exercice 1 :



HashiCorp

# Terraform

```
provider "azurerm" {
  features {}
}

resource "azurerm_virtual_machine" "example" {
  name          = "VM1-Windows"
  location      = "France Central"
  availability_set_id =
"/subscriptions/xxxxx/resourceGroups/xxxxx/providers/Microsoft.Compute/availabilit
ySets/xxxxx"
  vm_size       = "Standard_B1s"

  storage_os_disk {
    name          = "myosdisk"
    caching       = "ReadWrite"
    create_option = "FromImage"
    managed_disk_type = "Standard_LRS"
  }

  storage_image_reference {
    publisher = "MicrosoftWindowsServer"
    offer     = "WindowsServer"
    sku       = "2016-Datacenter"
    version   = "latest"
  }

  os_profile {
    computer_name = "myvm"
    admin_username = "myusername"
    admin_password = "mypassword"
  }

  network_interface_ids = [

"/subscriptions/xxxxx/resourceGroups/xxxxx/providers/Microsoft.Network/networkIn
terfaces/xxxxx"
  ]
}
```

# Exercise 2 :



HashiCorp

# Terraform

```
provider "azurerm" {
  features {}
}

resource "azurerm_availability_set" "example" {
  name            = "my-availability-set"
  location        = "France Central"
  platform_fault_domain_count = 2
  platform_update_domain_count = 5
}

resource "azurerm_virtual_machine" "windows_vm" {
  name            = "my-windows-vm"
  location        = "France Central"
  availability_set_id = azurerm_availability_set.example.id
  vm_size         = "Standard_DS1_v2"

  storage_os_disk {
    name            = "myosdisk"
    caching         = "ReadWrite"
    create_option   = "FromImage"
    managed_disk_type = "Standard_LRS"
  }

  storage_image_reference {
    publisher = "MicrosoftWindowsServer"
    offer     = "WindowsServer"
    sku       = "2016-Datacenter"
    version   = "latest"
  }

  os_profile {
    computer_name = "my-windows-vm"
    admin_username = "myusername"
    admin_password = "mypassword"
  }
}
```

```

network_interface_ids = [
  azurerm_network_interface.example_windows_vm.id
]
}

resource "azurerm_virtual_machine" "linux_vm" {
  name          = "my-linux-vm"
  location      = "France Central"
  availability_set_id = azurerm_availability_set.example.id
  vm_size       = "Standard_DS1_v2"

  storage_os_disk {
    name          = "myosdisk"
    caching       = "ReadWrite"
    create_option = "FromImage"
    managed_disk_type = "Standard_LRS"
  }

  storage_image_reference {
    publisher = "Canonical"
    offer     = "UbuntuServer"
    sku       = "18.04-LTS"
    version   = "latest"
  }

  os_profile {
    computer_name = "my-linux-vm"
    admin_username = "myusername"
    admin_password = "mypassword"
  }

  network_interface_ids = [
    azurerm_network_interface.example_linux_vm.id
  ]
}

```

## Exercise 3 :

```
resource "azurerm_virtual_machine_scale_set"
"example" {
  name          = "my-vmss"
  resource_group_name =
azurerm_resource_group.example.name
  location      =
azurerm_resource_group.example.location
  sku           = "Standard_DS1_v2"
  instances     = 3

  storage_profile_image_reference {
    publisher = "Canonical"
    offer     = "UbuntuServer"
    sku       = "18.04-LTS"
    version   = "latest"
  }

  os_profile {
    computer_name_prefix = "my-vm"
    admin_username       = "myusername"
    admin_password       = "mypassword"
  }
}
```

## **TP 3 :**

- 3.1) Héberger une application web avec Azure App Service.
- 3.2) Publier une application web sur Azure avec votre IDE.
- 3.3) Planifier un déploiement d'application web pour le test et la restauration avec les emplacements de déploiement App Service.
- 3.4) Effectuer le scale-up/scale-out d'une application web App Service pour répondre efficacement à la demande

# Exercice 1 :



HashiCorp

# Terraform

```
provider "azurerm" {
  features {}
}

resource "azurerm_resource_group" "example" {
  name     = "webApp-RG"
  location = "East US"
}

resource "azurerm_app_service_plan" "example" {
  name                = "plan-B1"
  location            = azurerm_resource_group.example.location
  resource_group_name = azurerm_resource_group.example.name

  sku {
    tier = "Basic"
    size = "B1"
  }
}

resource "azurerm_app_service" "example" {
  name                = "votre-nom-et-prenom" # UNIQUE
  location            = azurerm_resource_group.example.location
  resource_group_name = azurerm_resource_group.example.name
  app_service_plan_id = azurerm_app_service_plan.example.id

  site_config {
    java_version = "1.8"
    linux_fx_version = "JAVA|9.0"
  }

  app_settings = {
    "WEBSITE_HTTPLOGGING_RETENTION_DAYS" = "7"
    "WEBSITE_DOCUMENTROOT"              = ""
  }

  identity {
    type = "SystemAssigned"
  }
}
```



# Deuxième Web App :

```
provider "azurerm" {  
  features {}  
}  
  
resource "azurerm_resource_group" "rg" {  
  name     = "webApp-RG"  
  location = "East US"  
}  
  
resource "azurerm_app_service_plan" "plan" {  
  name                = "plan-B1"  
  location             = azurerm_resource_group.rg.location  
  resource_group_name = azurerm_resource_group.rg.name  
  
  sku {  
    tier = "Basic"  
    size = "B1"  
  }  
}  
  
resource "azurerm_app_service" "app" {  
  name                = "VotreNomPrenom1"  
  location             = azurerm_resource_group.rg.location  
  resource_group_name = azurerm_resource_group.rg.name  
  app_service_plan_id = azurerm_app_service_plan.plan.id  
  
  site_config {  
    java_version = "1.8"  
    linux_fx_version = "TOMCAT|9.0"  
  }  
}
```

# Troisième Web App :

```
provider "azurerm" {  
  features {}  
}  
  
resource "azurerm_resource_group" "rg" {  
  name     = "webApp-RG"  
  location = "France Central"  
}  
  
resource "azurerm_app_service_plan" "plan" {  
  name                = "plan-B1"  
  location             = azurerm_resource_group.rg.location  
  resource_group_name = azurerm_resource_group.rg.name  
  
  sku {  
    tier = "Basic"  
    size = "B1"  
  }  
}  
  
resource "azurerm_app_service" "app" {  
  name                = "VotreNomPrenom2"  
  location             = azurerm_resource_group.rg.location  
  resource_group_name = azurerm_resource_group.rg.name  
  app_service_plan_id = azurerm_app_service_plan.plan.id  
  
  site_config {  
    python_version = "3.11"  
    windows_fx_version = ""  
  }  
}
```

# Troisième Web App :

```
provider "azurerm" {  
  features {}  
}  
  
resource "azurerm_resource_group" "rg" {  
  name     = "webApp-RG"  
  location = "France Central"  
}  
  
resource "azurerm_app_service_plan" "plan" {  
  name                = "plan-B1"  
  location             = azurerm_resource_group.rg.location  
  resource_group_name = azurerm_resource_group.rg.name  
  
  sku {  
    tier = "Basic"  
    size = "B1"  
  }  
}  
  
resource "azurerm_app_service" "app" {  
  name                = "VotreNomPrenom2"  
  location             = azurerm_resource_group.rg.location  
  resource_group_name = azurerm_resource_group.rg.name  
  app_service_plan_id = azurerm_app_service_plan.plan.id  
  
  site_config {  
    python_version = "3.11"  
    windows_fx_version = ""  
  }  
}
```

# Réponses aux questions :

- Combien d'applications sont hébergées? :

Une seule application est hébergée

- Combien d'instances sont en cours d'exécution? :

3 instances.

- Peut-on effectuer un scale-out? Pourquoi? :

Oui, on peut effectuer un scale-out manuel pour augmenter le nombre d'instances en cours d'exécution. Cela permet de gérer la charge et la disponibilité de l'application en répartissant le trafic entre plusieurs instances.

# Réponses aux questions :

- Effectuez un scale-out manuel de 1. Quel est le nombre maximal d'instance qu'on peut avoir?

Pourquoi? : Le nombre maximal d'instances qu'on peut avoir avec le plan "B1" est 3. Cela est dû aux limites imposées par le plan "B1" qui permet d'exécuter jusqu'à 3 instances maximum pour l'application.

- Assurez-vous que le nombre d'instance a augmenté après le scale-out :

Après avoir effectué un scale-out manuel de 1, le nombre d'instances en cours d'exécution augmentera à 2.

- Peut-on effectuer un scale-out automatique? Justifiez votre réponse :

Non, le plan "B1" du niveau "Basic" ne prend pas en charge le scale-out automatique. Cette fonctionnalité est réservée aux plans de niveau supérieur, tels que "Standard", "Premium", ou "Isolated".



```
provider "azurerm" {  
  features {}  
}  
  
resource "azurerm_app_service_plan" "this" {  
  name          = "app-service-plan-name"  
  location      = "East US"  
  resource_group_name = "webApp-RG"  
  sku {  
    tier = "Basic"  
    size = "D1"  
  }  
}  
  
resource "azurerm_web_app" "this" {  
  name          = "web-app-name"  
  location      = "East US"  
  resource_group_name = "webApp-RG"  
  server_farm_id   = azurerm_app_service_plan.this.id  
  ...  
}
```

Accéder à [portal.azure.com](https://portal.azure.com) pour monitorer la VM



```
autoscale_setting {  
  direction = "Increase"  
  metric_trigger {  
    metric_name =  
      "\\Processor\\PercentProcessorTime"  
    metric_resource_id = azurerm_web_app.this.id  
    time_grain      = "PT1M"  
    statistic       = "Average"  
    time_window     = "PT5M"  
    time_aggregation = "Average"  
    operator        = "GreaterThan"  
    threshold       = 70  
    metric_namespace = "Microsoft.Web/sites"  
  }  
  scale_action {  
    direction = "Increase"  
    type      = "ChangeCount"  
    value     = "1"  
    cooldown  = "PT5M"  
    min_cooldown = "PT5M"  
  }  
}
```



```
autoscale_setting {  
  direction = "Decrease"  
  metric_trigger {  
    metric_name = "\\Processor\\PercentProcessorTime"  
    metric_resource_id = azurerm_web_app.this.id  
    time_grain = "PT1M"  
    statistic = "Average"  
    time_window = "PT5M"  
    time_aggregation = "Average"  
    operator = "LessThan"  
    threshold = 20  
    metric_namespace = "Microsoft.Web/sites"  
  }  
  scale_action {  
    direction = "Decrease"  
    type = "ChangeCount"  
    value = "-1"  
    cooldown = "PT5M"  
    min_cooldown = "PT5M"  
  }  
}
```



# Suite : Suppression



HashiCorp

# Terraform

Pour supprimer les deux dernières applications web créées avec Terraform, on peut utiliser la commande suivante :

```
terraform destroy -  
target=azurerm_web_app.<nom_app_web1> -  
target=azurerm_web_app.<nom_app_web2>
```

Pour le reste, je suis dans l'incapacité de créer un emplacement (slot) de déploiement pour le test, déployer le même code modifié.

# Suite : DNS

Pour la résolution de nom de domaine : j'utilise  
genious pour mon propre portfolio  
<https://cellardoor.info> hébergé sur DigitalOcean  
et configuré comme suit :

10 Enregistrement(s) trouvé(s), Page 1 de 1			
Name	Value	TTL	Status
@.cellardoor.info	68.183.13.168	86400	Actif
averti.cellardoor.info	68.183.13.168	86400	Actif
cliniquesolis.cellardoor.info	68.183.13.168	86400	Actif
dotnetstore.cellardoor.info	68.183.13.168	86400	Actif
old.cellardoor.info	68.183.13.168	86400	Actif
pgadmin.cellardoor.info	68.183.13.168	86400	Actif
reactresto.cellardoor.info	68.183.13.168	88400	Actif
telerp.cellardoor.info	68.183.13.168	86400	Actif
weavent.cellardoor.info	68.183.13.168	86400	Actif
www.cellardoor.info	68.183.13.168	86400	Actif

# Conclusion

En conclusion, l'utilisation de Terraform pour automatiser la création, la gestion et la suppression des ressources dans Azure offre de nombreux avantages pour le déploiement d'applications web. En utilisant la configuration Terraform, On peut définir et gérer l'infrastructure comme du code, ce qui vous permet de versionner, partager, auditer et réutiliser facilement notre infrastructure.

Dans ces TPs, nous avons utilisé Terraform pour créer plusieurs ressources dans Azure, notamment des groupes de ressources, des machines virtuelles, des machines virtuelles à l'échelle, des plans App Service et des applications web.

Nous avons également configuré des paramètres tels que la région, la pile, le serveur, le système d'exploitation, le plan d'hébergement, ainsi que des règles de mise à l'échelle automatique basées sur des métriques de performance telles que le CPU.

L'automatisation avec Terraform permet d'améliorer la reproductibilité, la cohérence et l'efficacité des déploiements d'applications web dans Azure, en évitant les erreurs de configuration manuelle et en facilitant la gestion des ressources à grande échelle.

Il est important de toujours vérifier et valider les configurations Terraform avant de les appliquer pour éviter les erreurs potentielles et garantir la sécurité et la conformité des déploiements.

En somme, Terraform offre une approche puissante et flexible pour gérer l'infrastructure comme du code dans Azure, permettant aux équipes de développement et d'exploitation de travailler de manière plus efficace et de garantir une gestion robuste et automatisée des ressources pour les applications web.

**LAOUINA Yassine - 411R9**