

Étape 1: Installation de Homebrew

Description :

Homebrew est un gestionnaire de paquets pour macOS qui facilite l'installation de logiciels et de bibliothèques. Il simplifie également la gestion des dépendances.

Instructions :

- a) Ouvrez le terminal sur votre Mac.
- b) Installez Homebrew en utilisant la commande suivante :

```
/bin/bash -c "$(curl -fsSL  
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

Étape 2: Installation de Terraform

Description :

Terraform est un outil d'infrastructure en tant que code (IaC) qui permet de définir et de provisionner des infrastructures cloud de manière programmable.

Instructions :

- a) Installation de HashiCorp

```
brew tap hashicorp/tap
```

- b) Installation de Terraform

```
brew install hashicorp/tap/terraform
```

Étape 3: Crédit du Compte AWS

Description :

AWS (Amazon Web Services) est un service de cloud computing qui offre une variété de services et de ressources informatiques à la demande.

Instructions :

- a) Accédez au site web d'AWS (<https://aws.amazon.com/>).

- b) Cliquez sur "Create an AWS Account" et suivez les instructions pour créer un compte AWS.
- c) Ici, on renseigne son compte Ynov.

The screenshot shows the AWS Console Home page. At the top, there's a navigation bar with the AWS logo, a 'Services' dropdown, a search bar, and user information ('Lucas Ynov'). Below the navigation is a 'Console Home' section with three main cards:

- Recently visited:** IAM, Billing, AWS Organizations.
- Welcome to AWS:**
 - Getting started with AWS**: Learn the fundamentals and find valuable information to get the most out of AWS.
 - Training and certification**: Learn from AWS experts and advance your skills and knowledge.
 - What's new with AWS?**: Discover new AWS services, features, and Regions.
- AWS Health:**
 - Open issues: 0 (Past 7 days)
 - Scheduled changes: 0 (Upcoming and past 7 days)
 - Other notifications: 0 (Past 7 days)

Étape 4: Installation de AWS CLI

Description :

AWS CLI (Command Line Interface) est un outil en ligne de commande qui permet d'interagir avec les services AWS directement depuis le terminal.

Instructions :

- a) Installez AWS CLI à l'aide de Homebrew avec la commande :

```
brew install awscli
```

- b) Configurez AWS CLI en exécutant la commande :

```
aws configure
```

- c) Création d'un user AMI

Fournissez les informations demandées, y compris votre clé d'accès AWS et votre région par défaut.

Étape 5: Création d'un Projet Terraform

Instructions :

- a) Créez un répertoire pour votre projet Terraform :

```
mkdir teralucas
```

- b) Naviguez dans le répertoire nouvellement créé :

```
cd teralucas
```

- c) Exécutez la commande pour initialiser Terraform :

```
terraform init
```

Étape 6: 01_hello_terraform. Écriture du Code Terraform (main.tf)

Description :

Créez un fichier main.tf qui contient la configuration Terraform, notamment la définition du fournisseur (AWS) et des ressources à déployer.

Instructions :

- a) Utilisez un éditeur de texte comme nano pour créer le fichier main.tf dans le répertoire de votre projet.
b) Ajoutez la configuration Terraform :

```
provider "aws" {  
    region = "us-west-1"  
}  
resource "aws_instance" "example" {  
    ami      = "ami-0aafdae616ee7c9b7"  
    instance_type = "t2.micro"  
}
```

Étape 7: Initialisation et Application Terraform

- a) Utilisation de terraform apply:

terraform apply

```
lucas@MacBook-Pro-de-Lucas teralucas % terraform apply

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:

# aws_instance.example will be created
+ resource "aws_instance" "example" {
    + ami                                = "ami-0aafdae616ee7c9b7"
    + arn                                = (known after apply)
    + associate_public_ip_address        = (known after apply)
    + availability_zone                  = (known after apply)
    + cpu_core_count                     = (known after apply)
    + cpu_threads_per_core              = (known after apply)
    + disable_api_stop                  = (known after apply)
    + disable_api_termination           = (known after apply)
    + ebs_optimized                      = (known after apply)
    + get_password_data                 = false
    + host_id                            = (known after apply)
    + host_resource_group_arn           = (known after apply)
    + iam_instance_profile              = (known after apply)
    + id                                 = (known after apply)
    + instance_initiated_shutdown_behavior = (known after apply)
    + instance_lifecycle                = (known after apply)
    + instance_state                    = (known after apply)
    + instance_type                     = "t2.micro"
    + ipv6_address_count                = (known after apply)
    + ipv6_addresses                     = (known after apply)
    + key_name                           = (known after apply)
    + monitoring                         = (known after apply)
    + outpost_arn                        = (known after apply)
    + password_data                     = (known after apply)
    + placement_group                   = (known after apply)
    + placement_partition_number         = (known after apply)
    + primary_network_interface_id      = (known after apply)
    + private_dns                        = (known after apply)
    + private_ip                         = (known after apply)
    + public_dns                         = (known after apply)
    + public_ip                          = (known after apply)
    + secondary_private_ips             = (known after apply)
    + security_groups                   = (known after apply)
    + source_dest_check                 = true
    + spot_instance_request_id          = (known after apply)

    + user_data_replace_on_change       = false
    + vpc_security_group_ids            = (known after apply)
}

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

aws_instance.example: Creating...
aws_instance.example: Still creating... [10s elapsed]
aws_instance.example: Still creating... [20s elapsed]
aws_instance.example: Still creating... [30s elapsed]
aws_instance.example: Still creating... [40s elapsed]
aws_instance.example: Still creating... [50s elapsed]
aws_instance.example: Still creating... [1m0s elapsed]
aws_instance.example: Still creating... [1m10s elapsed]
aws_instance.example: Still creating... [1m20s elapsed]
aws_instance.example: Still creating... [1m30s elapsed]
aws_instance.example: Creation complete after 1m35s [id=i-0f5dd5bf4eca3342c]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
lucas@MacBook-Pro-de-Lucas teralucas %
```

b) Fermeture de l'instance :

```
terraform destroy
```

```
lucas@MacBook-Pro-de-Lucas teralucas % terraform destroy
aws_instance.example: Refreshing state... [id=i-0f5dd5bf4eca3342c]

Terraform used the selected providers to generate the following execution plan. Resource actions are
indicated with the following symbols:
- destroy

Terraform will perform the following actions:

# aws_instance.example will be destroyed
- resource "aws_instance" "example" {
    - ami                               = "ami-0aafdaee616ee7c9b7" -> null
    - arn                               = "arn:aws:ec2:us-west-1:691750145950:instance/i-0f5dd5b
f4eca3342c" -> null
    - associate_public_ip_address      = true -> null
    - availability_zone                = "us-west-1c" -> null
    - cpu_core_count                  = 1 -> null
    - cpu_threads_per_core            = 1 -> null
    - disable_api_stop                = false -> null
    - disable_api_termination         = false -> null
    - ebs_optimized                   = false -> null
    - get_password_data               = false -> null
    - hibernation                     = false -> null
    - id                               = "i-0f5dd5bf4eca3342c" -> null
    - instance_initiated_shutdown_behavior = "stop" -> null
    - instance_state                  = "running" -> null
    - instance_type                   = "t2.micro" -> null
    - ipv6_address_count              = 0 -> null
    - ipv6_addresses                  = [] -> null
    - monitoring                      = false -> null
    - placement_partition_number       = 0 -> null
    - primary_network_interface_id    = "eni-020b8a253de05c9e7" -> null
    - private_dns                      = "ip-172-31-6-49.us-west-1.compute.internal" -> null
}
```

```
Enter a value: yes

aws_instance.example: Destroying... [id=i-0f5dd5bf4eca3342c]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 10s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 20s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 30s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 40s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 50s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 1m0s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 1m10s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 1m20s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 1m30s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 1m40s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 1m50s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 2m0s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 2m10s elapsed]
aws_instance.example: Still destroying... [id=i-0f5dd5bf4eca3342c, 2m20s elapsed]
aws_instance.example: Destruction complete after 2m24s
```

Annexe :

Site utilisé :