

# Lab 1: Premiers pas avec Terraform et l'Infrastructure as Code

Cloud Computing

Université Claude Bernard Lyon 1 March 31, 2026

## Assignment Overview

### Document Contents:

1. Objectif .....	2
2. Pré-requis .....	2
2.1. Conseils .....	2
3. Contexte .....	2
4. Étapes .....	3
4.1. Étape 1: Lire les fichiers Terraform .....	3
4.2. Étape 2: Initialiser le projet .....	3
4.3. Étape 3: Lancer un plan .....	3
4.4. Étape 4: Appliquer la configuration .....	3
4.5. Étape 5: Vérifier les ressources .....	3
4.6. Étape 6: Inspecter le state .....	4
4.7. Étape 7: Détruire les ressources .....	4
5. Questions .....	5
6. Aller plus loin .....	5

### Assignment Details:

- **Course:** Cloud Computing
- **Instructor:** Hugo Blanc
- **Software:** Terraform, AWS, LocalStack
- **Duration:** ~1 hour
- **Lab Number:** Lab 1

# Objectif

---

Découvrir les fondamentaux de Terraform : `init`, `plan`, `apply`, `destroy`. Ce TP vous familiarisera avec la gestion du cycle de vie des ressources managées par Terraform.

## Pré-requis

---

- Avoir Terraform ( $\geq 1.5.0$ );
- avoir `tflocal` (`pip3 install terraform-local`);
- avoir LocalStack qui tourne;
- avoir la CLI AWS (voir la documentation officielle).

## Conseils

a Je vous recommande de faire les alias suivants:

```
alias tf=tflocal
alias aws="aws --endpoint-url=http://localhost:4566"
```

Également, pour simplifier l'usage de la CLI AWS, je vous recommande de définir des faux credentials pour éviter les erreurs:

```
export AWS_ACCESS_KEY_ID=test
export AWS_SECRET_ACCESS_KEY=test
export AWS_DEFAULT_REGION=us-east-1
```

## Contexte

---

**LocalStack** émule des services AWS en local sur la machine, sur le port 4566. La commande `tflocal` est un wrapper autour de Terraform qui permet de rediriger les appels d'API vers LocalStack.

# Étapes

---

## Étape 1: Lire les fichiers Terraform

Avant d'entrer des commandes, il vous faut lire attentivement chaque fichier `.tf` présent dans ce dossier.

### Question

Identifiez:

- Les variables déclarées et leurs valeurs;
- les ressources qui seront créées;
- Les sorties qui seront affichées.

## Étape 2: Initialiser le projet

```
tflocal init
```

Cette commande télécharge les providers déclarés dans le fichier `versions.tf` et initialise le dossier courant. Un dossier `.terraform/` est créé.

### Question

Inspectez le contenu du dossier `.terraform/` et du fichier `terraform.lock.hcl`. Que contiennent-ils ?

## Étape 3: Lancer un plan

```
tflocal plan
```

Terraform affiche les ressources qu'il prévoit de créer, modifier ou détruite. Aucun changement dans l'infrastructure n'est appliqué à cette étape, elle sert de vérification.

### Question

Lisez la sortie attentivement. Combien de ressources vont être créées ?

## Étape 4: Appliquer la configuration

```
tflocal apply
```

Terraform demande confirmation avant d'appliquer le changement de configuration d'infrastructure. Tapez `yes`. Les ressources seront créées et le fichier de state (`terraform.tfstate`) sera modifié.

Observez les sorties affichées à la fin de l'exécution.

## Étape 5: Vérifier les ressources

Utilisez la CLI AWS pour vérifier la bonne création des ressources :

```
aws --endpoint-url=http://localhost:4566 s3 ls
aws --endpoint-url=http://localhost:4566 s3 ls s3://<bucket-name>/
```

Vous pouvez également vérifier votre bucket et son objet en allant sur l'adresse : <http://localhost:4566/tp-cloud-YOURNAME-dev/index.html>

## Étape 6: Inspecter le state

```
tflocal show
tflocal state list
```

Ces commandes permettent d'inspecter le state actuellement comme Terraform le voit.

## Étape 7: Détruire les ressources

```
tflocal destroy
```

Toutes les ressources managées par Terraform dans le state sont détruites. Vérifiez que le bucket n'existe plus.

# Questions

---

1. **Que se passe-t-il si vous lancez `tflocal apply` une seconde fois sans modifier aucun fichier ?**  
Combien de ressources sont ajoutées, modifiées, détruites ?
2. **Ouvrez le fichier `terraform.tfstate`.** Que contient-il ? Pourquoi ce fichier est-il critique dans une optique de collaboration ?
3. **Supprimez manuellement le fichier `terraform.tfstate`, puis lancez `tflocal plan`.** Qu'observez-vous ? Qu'est-ce que ça indique sur le rôle du fichier de state ?
4. **Modifiez la variable `student_name` et relancez `tflocal apply`.** Quelles ressources sont modifiées ? Lesquelles sont re-crées ? Pourquoi ?
5. **Quelle est la différence entre `tflocal plan` et `tflocal apply` ?** Dans quel contexte la séparation de ces deux commandes peut être utile ?
6. **Quel est le but du fichier `.terraform.lock.hcl` ?** Devrait-il être commité dans un dépôt Git ?

## Aller plus loin

---

- Ajoutez un second objet S3 avec un contenu différent
- Ajoutez un tag supplémentaire au bucket, et observez le plan
- Essayez de renommer un bucket (changez la ressource Terraform) et observez ce que Terraform propose.