



## I. Pour mener à bien la séquence...

Les logs de connexion ssh ne sont pas enregistrés par défaut :

Pour les enregistrer sur un poste client, modifier le fichier `/etc/profile`

```
function log2syslog
{
    declare command
    command=$(fc -ln -0)
    logger -p local1.notice -t bash -i -- $USER : $command
}
trap log2syslog DEBUG
```

## II. Gitlab

Gitlab est la solution retenue pour ce projet pour les fonctionnalités suivantes :

- La possibilité d'installer une instance locale (self manage) qui pourra être intégré à l'annuaire ldap de la section
- L'intégration native des outils DevOps
- La solution Open Source gratuite (pour la version communautaire)

Il existe cependant des solutions alternatives.

- Un serveur « git » simple : gogs ou gitea
- Des outils CI/CD externe Jenkins
- Une solution SAAS de Gitlab
- Github.

## III. Utilisation des programmes fournies

Deux programmes sont fournis :

- GestionCentraliseeDeLogs
- Flask\_API\_REST

### III.1 Gestion centralisée de logs

Le dépôt contient les fichiers du projet Gestion Centralisé de logs

Il contient également les fichiers pour l'intégration continue (gitlab) et les tests unitaires Catch2



Le programme est configuré pour QtCreator et la chaîne Cmake mais peut être porté sur une autre plate forme.

### **III.1.1 Branches et Itération**

- Le dépôt contient les 6 itérations du projet sur 6 branches.
- Chaque branche est le code attendu par l'étudiant à la fin de chaque itération
- Pour passer d'une branche à une autre (exemple Iteration1)

```
git checkout Iteration1
```

## **III.2 Flask\_API\_REST**

C'est un container docker avec un petit programme de démonstration d'utilisation d'une API REST

### **III.2.1 Usage**

- Lancer le container

```
docker compose up -d
```

- Ouvrir le site dans un navigateur

```
http://localhost:5000
```

- Appeler une route (exemple lister tous les éléments)

```
http://localhost:5000/api/v1/sshlog/all
```