# ESGI - Conteneurs et Devops

Thibaut LACUISSE – ESGI M1 SI

https://github.com/TiboJhjj/art-explorer.git

CI/CD pour \*Chicago Art Explorer\* – Guide d'installation (utilisateur \*\*tibo\*\*)

## 1. Vue d’ensemble

1. \*\*Installer Docker\*\* sur Debian 11

2. \*\*Lancer Jenkins\*\* dans un conteneur Docker

3. Effectuer la \*\*configuration initiale\*\* de Jenkins (déverrouillage, plugins, utilisateur)

4. \*\*Ajouter un agent distant\*\* via SSH (clé, credentials, nœud)

5. Mettre en place un \*\*pipeline type\*\* pour :

- exécuter les \*tests unitaires\*

- construire l’\*image Docker\* de production

- \*déployer\* automatiquement sur une VM distante

- envoyer une \*notification\* en cas d’échec de build

Cet environnement assure une CI/CD fiable pour l’application \*\*Chicago Art Explorer\*\*

---

## 2. Installation de Docker sur Debian 11

Mettre à jour les paquets

sudo apt update && sudo apt upgrade -y

Installer les dépendances

sudo apt install -y apt-transport-https ca-certificates curl gnupg lsb-release

Ajouter la clé GPG Docker + dépôt officiel

curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] \

https://download.docker.com/linux/debian $(lsb\_release -cs) stable" \

| sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

Installer Docker Engine

sudo apt update

sudo apt install -y docker-ce docker-ce-cli containerd.io

Ajouter l'utilisateur tibo au groupe docker

sudo usermod -aG docker tibo

Se déconnecter / reconnecter pour prise en compte

Vérifier

docker run hello-world

---

## 3. Lancement de Jenkins dans Docker

Créer un réseau dédié

docker network create jenkins

Volume persistant

docker volume create jenkins\_home

Conteneur Jenkins LTS

docker run -d \

--name jenkins \

--restart unless-stopped \

--network jenkins \

-p 8080:8080 -p 50000:50000 \

-v jenkins\_home:/var/jenkins\_home \

jenkins/jenkins:lts

Mot de passe initial

docker exec jenkins cat /var/jenkins\_home/secrets/initialAdminPassword

Accédez ensuite à `http://192.168.45.130:8080` et collez le mot de passe.

---

## 4. Configuration initiale de Jenkins

1. \*\*Déverrouillage\*\* : saisir la clé initiale.

2. \*\*Plugins\*\* : \*Install suggested plugins\*.

3. \*\*Compte admin\*\* : créer un utilisateur administrateur.

4. \*\*Git\*\* :

- Installer \*\*Git Plugin\*\* si nécessaire

- \*Manage Jenkins → Global Tool Configuration\* → Git executable : `/usr/bin/git`

5. \*\*Plugins clés\*\* :

- Docker Pipeline

- SSH Agent

- Publish Over SSH

- Email Extension (notifications mail)

---

## 5. Ajout d’un agent SSH distant (utilisateur \*\*tibo\*\*)

5.1 Sur la VM distante

Installer OpenSSH

sudo apt update

sudo apt install -y openssh-server

sudo systemctl enable ssh --now

Préparer l'utilisateur tibo

sudo mkdir -p /home/tibo/.ssh

sudo chown tibo:tibo /home/tibo/.ssh

sudo chmod 700 /home/tibo/.ssh

5.2 Sur le serveur Jenkins

Générer la paire de clés (sans passphrase)

ssh-keygen -t ed25519 -f ~/.ssh/tibo\_agent -N ""

Copier la clé publique sur l’agent

ssh-copy-id -i ~/.ssh/tibo\_agent.pub tibo@192.168.45.130

5.3 Ajouter les identifiants SSH dans Jenkins

\*Manage Jenkins → Credentials → System → Global credentials → Add Credentials\*

| Champ | Valeur |

|-------|--------|

| \*\*Kind\*\* | SSH Username with private key |

| \*\*Username\*\* | `tibo` |

| \*\*Private Key\*\* | \*Enter directly\* — collez le contenu de `~/.ssh/tibo\_agent` |

| \*\*ID\*\* | `ssh-agent-key` |

| \*\*Description\*\* | Clé SSH pour agent distant |

Une image contenant texte, capture d’écran, nombre, Police

Le contenu généré par l’IA peut être incorrect.

5.4 Déclarer le nœud

\*Manage Jenkins → Manage Nodes and Clouds → New Node\*

| Paramètre | Valeur |

|-----------|--------|

| \*\*Name\*\* | `agent-ssh` |

| \*\*Type\*\* | \*Permanent Agent\* |

| \*\*Remote root dir\*\* | `/home/tibo` |

| \*\*Launch method\*\* | \*Launch agents via SSH\* |

| \*\*Host\*\* | `<IP\_AGENT>` |

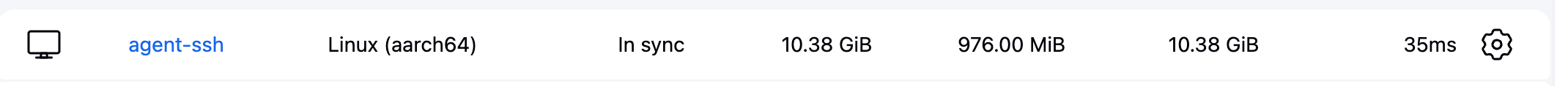
| \*\*Credentials\*\* | `ssh-agent-key` |

| \*\*Host Key Verification\*\* | \*Non verifying\* (tests) ou \*Manually trusted key\* |

Une image contenant texte, reçu, capture d’écran

Le contenu généré par l’IA peut être incorrect.

Sauvegardez puis lancez ; le statut passe à \*\*online\*\*.



---

## 6. Pipeline type (aperçu rapide)

Le `Jenkinsfile` du dépôt automatise :

1. \*\*Tests unitaires\*\* (`pytest`)

2. \*\*Build & push\*\* d’une image Docker

3. \*\*Déploiement\*\* sur la VM via SSH

4. \*\*Notification\*\* (Gmail) en cas d’échec de build

Cela boucle l’intégration et la livraison continues pour \*Chicago Art Explorer\*.

* Pas finalisé, je reviens sur cette partie à la fin

---

## 7. Notifications par Gmail

7.1 Créer l'identifiant Jenkins

Manage Jenkins → Credentials → (Global) → Add Credentials

Kind: Username with password

Username: ton.mail@gmail.com

Password: <App Password Gmail>

ID: gmail-creds

Description: Gmail SMTP

7.2 Configurer le serveur SMTP

Manage Jenkins → Configure System → E‑mail Notification

SMTP server: smtp.gmail.com

Use SMTP Authentication: ✔️

Credentials: gmail-creds

SMTP port: 587

Use TLS: ✔️

Charset: UTF-8

## Problème

Je n’ai pas réussi à faire fonctionner la notification Gmail. Je n’ai pas compris comment bien configurer cette partie sur l’interface Jenkins donc la pipeline ne fonctionne pas. Malgré tout, des pipelines plus simples fonctionne bien puisque le reste du TP est opérationnel.

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.