**Documentation du Projet CI/CD avec Jenkins sur GCP**

**Introduction**

Ce document décrit les étapes nécessaires pour créer et configurer un projet CI/CD avec Jenkins sur Google Cloud Platform (GCP). Il explique comment intégrer des Webhooks GitHub pour automatiser le processus d'intégration et de livraison continues (CI/CD). Grâce à cette configuration, chaque fois qu'un nouveau commit est poussé sur le dépôt GitHub, le pipeline Jenkins se déclenche automatiquement pour construire, tester et déployer le code.

**Étapes de Configuration**

**1. Création du Projet sur GitHub**

Nous avons créé un projet sur GitHub nommé ci-cd-pipeline-avec-jenkins-sur-gcp.

**2. Clonage du Projet**

Clonez le projet sur votre ordinateur en utilisant la commande suivante :

git clone https://github.com/Serchgalarza/ci-cd-pipeline-avec-jenkins-sur-gcp.git

Accédez ensuite au répertoire du projet :

cd ci-cd-pipeline-avec-jenkins-sur-gcp

**3. Création du Jenkinsfile**

Créez un fichier Jenkinsfile avec l'éditeur de texte nano :

nano Jenkinsfile

Ajoutez le contenu suivant au Jenkinsfile :

pipeline {

agent any

stages {

stage('Build') {

steps {

echo 'Building...'

}

}

stage('Test') {

steps {

echo 'Testing...'

}

}

stage('Deploy') {

steps {

echo 'Deploying...'

}

}

}

}

Ce Jenkinsfile définit les étapes de base pour le pipeline CI/CD, incluant les phases de construction, de test et de déploiement.

Pour sauvegarder les modifications dans nano, utilisez la combinaison de touches Ctrl + O, puis appuyez sur Enter. Pour fermer nano, utilisez Ctrl + X.

**4. Commit et Push des Modifications**

**4.1. Commit Direct à la Branche main**

Si vous choisissez de faire des modifications directement sur la branche main, utilisez les commandes suivantes :

git add .

git commit -m "Ajouter le Jenkinsfile pour configurer le pipeline CI/CD" -m "Ce commit ajoute le Jenkinsfile qui définit les étapes de construction, de test et de déploiement pour le pipeline CI/CD."

git push origin main

**4.2. Utiliser une Branche pour les Modifications**

Pour éviter des problèmes potentiels et suivre les bonnes pratiques, créez une nouvelle branche pour les modifications :

1. **Créer une Nouvelle Branche** :

git checkout -b feature/ajouter-jenkinsfile

1. **Ajouter et Valider les Changements** :

git add Jenkinsfile

git commit -m "Ajouter le Jenkinsfile pour configurer le pipeline CI/CD"

git push origin feature/ajouter-jenkinsfile

1. **Créer une Pull Request (PR)** :

Accédez à GitHub et créez une Pull Request pour fusionner les changements de votre branche feature/ajouter-jenkinsfile dans main.

* 1. Allez sur la page du dépôt GitHub.
  2. Cliquez sur l'onglet "Pull Requests".
  3. Cliquez sur "New Pull Request".
  4. Sélectionnez feature/ajouter-jenkinsfile comme branche source et main comme branche de destination.
  5. Ajoutez une description si nécessaire et soumettez la Pull Request.

1. **Révision et Fusion** :

Une fois la Pull Request soumise, elle doit être révisée et approuvée par les membres de l'équipe. Après approbation, fusionnez la Pull Request avec la branche main.

1. **Supprimer la Branche (optionnel)** :

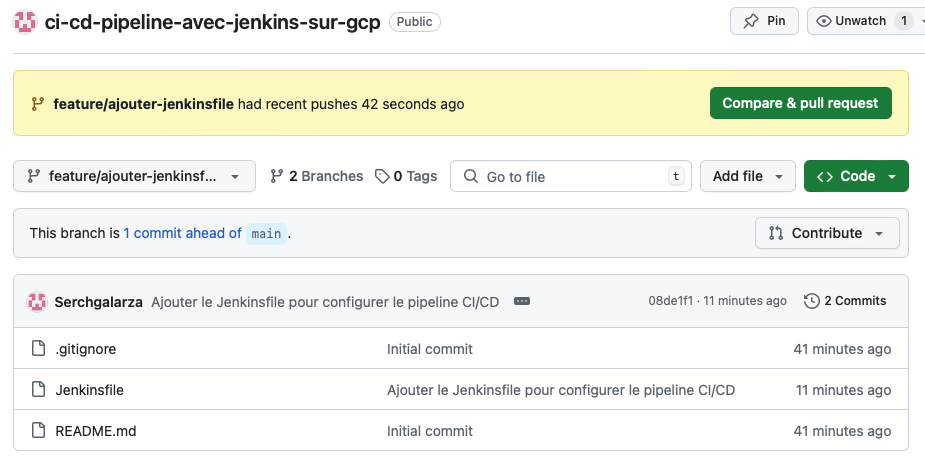
Après fusion, vous pouvez supprimer la branche de fonctionnalité si elle n'est plus nécessaire :

git branch -d feature/ajouter-jenkinsfile

git push origin --delete feature/ajouter-jenkinsfile

**5. Validation sur GitHub**

Une fois les modifications ajoutées et la branche créée, vérifiez les informations sur GitHub. Vous pouvez également basculer entre les branches pour confirmer que les changements ont été appliqués correctement.

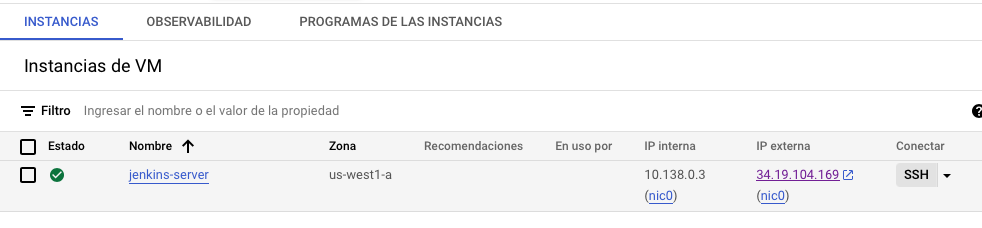


**Déploiement et Configuration de Jenkins sur GCP**

**1. Lancer l'Instance sur Google Cloud Platform (GCP)**

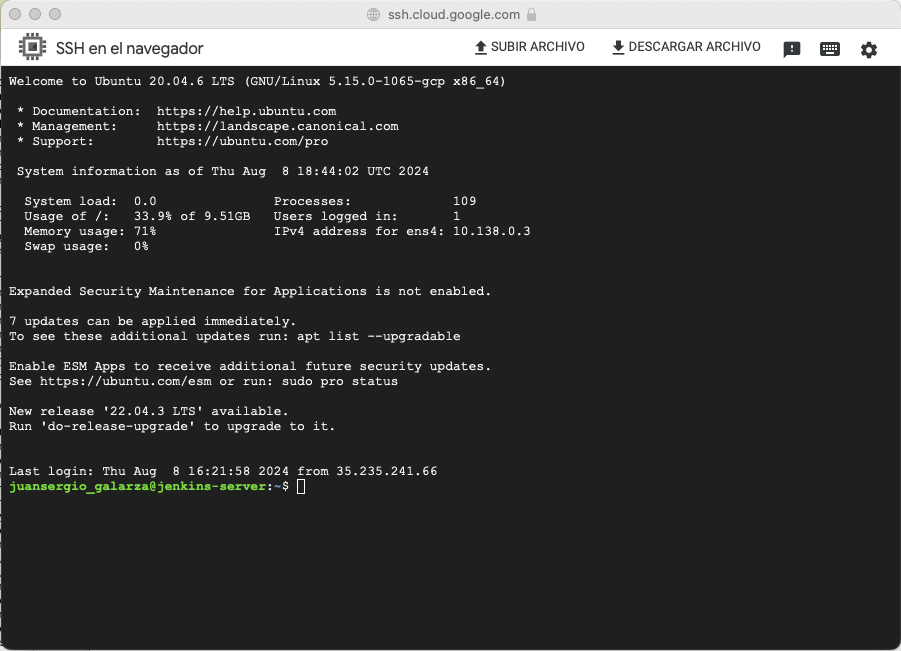
Pour notre démonstration, nous avons créé une instance de type micro avec Ubuntu comme système d'exploitation.

1. **Accéder à Google Cloud Console** :
   * Connectez-vous à Google Cloud Console.
2. **Créer une Nouvelle Instance** :
   * Allez dans le menu "Compute Engine", puis sélectionnez "VM instances".
   * Cliquez sur "Create Instance" pour lancer une nouvelle instance.
   * Sélectionnez un type de machine micro (par exemple, e2-micro).
   * Choisissez Ubuntu comme système d'exploitation dans les options de configuration de la machine.
3. **Configurer les Règles de Pare-feu (VPC)** :
   * Dans les paramètres de l'instance, configurez les règles de pare-feu pour permettre le trafic sur le port 8080, nécessaire pour accéder à Jenkins.
   * Vous pouvez le faire en cochant les options "Allow HTTP traffic" et "Allow HTTPS traffic", et en ajoutant une règle personnalisée pour le port 8080 si ce n’est pas déjà fait.
4. **Lancer l'Instance** :
   * Cliquez sur "Create" pour déployer l'instance.



**2. Accéder à la Configuration de l'Instance via SSH**

1. **Ouvrir la Console SSH** :
   * Dans Google Cloud Console, allez sur "Compute Engine" -> "VM instances".
   * Cliquez sur le bouton "SSH" pour ouvrir une session terminal directement dans le navigateur ou utilisez un client SSH pour vous connecter à l'instance.



**3. Installer et Configurer Jenkins**

1. **Mettre à Jour le Système** : Assurez-vous que le système est à jour :

sudo apt update

sudo apt upgrade -y

1. **Installer Java (si nécessaire)** : Jenkins nécessite Java. Installez OpenJDK :

sudo apt install openjdk-11-jdk -y

1. **Ajouter le Dépôt Jenkins et Installer Jenkins** : Ajoutez le dépôt Jenkins et installez Jenkins :

wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -

sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary > /etc/apt/sources.list.d/jenkins.list'

sudo apt update

sudo apt install jenkins -y

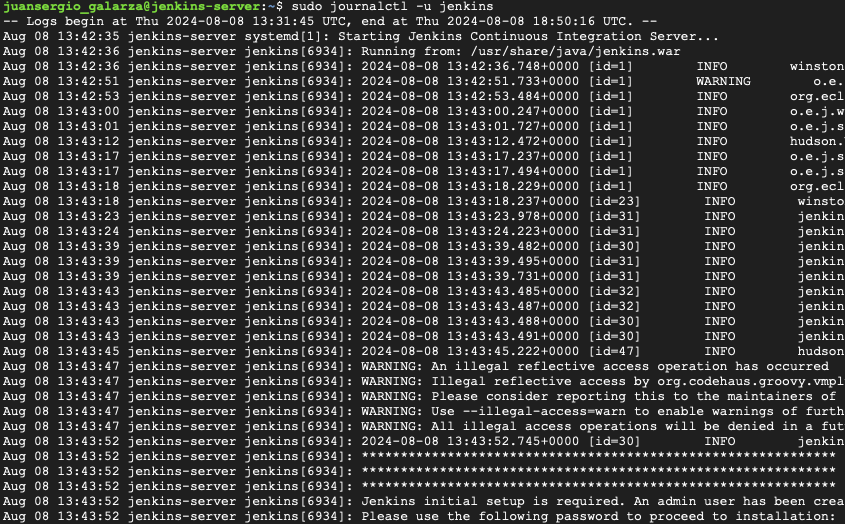
1. **Démarrer Jenkins** : Assurez-vous que Jenkins est démarré :

sudo systemctl start jenkins

sudo systemctl enable jenkins

Pour voir les logs de Jenkins, utilisez :

sudo systemctl restart jenkins



1. **Configurer le Pare-feu pour Jenkins** : Vérifiez que le port 8080 est ouvert pour le trafic entrant (cela devrait être fait automatiquement si vous avez configuré les règles VPC précédemment).
2. **Accéder à Jenkins via le Navigateur** : Ouvrez un navigateur web et accédez à l'adresse de votre instance en utilisant le port 8080 :

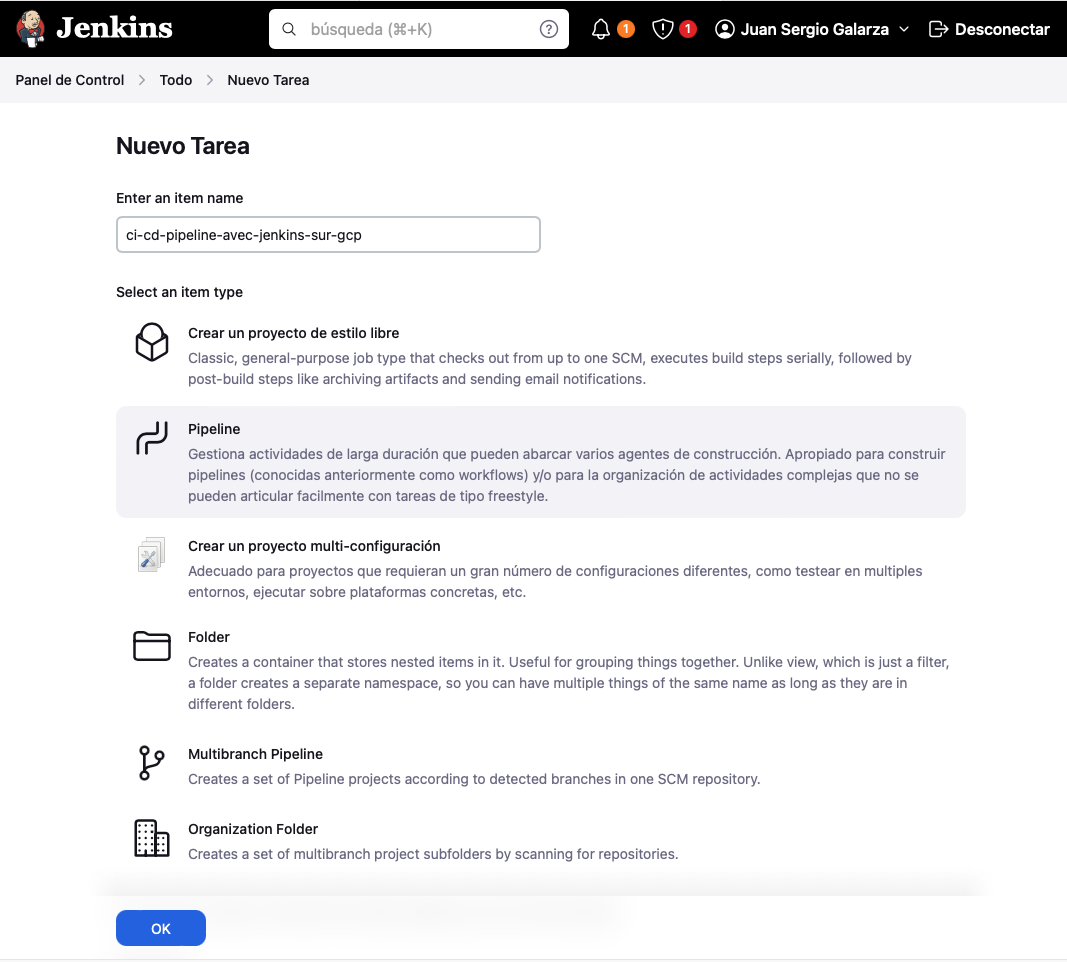
http://<external-ip>:8080

Dans cet exemple, l'URL est : http://34.19.104.169:8080.

Remplacez <external-ip> par l'adresse IP externe de votre instance.

**4. Configurer Jenkins pour Utiliser le Repository GitHub**

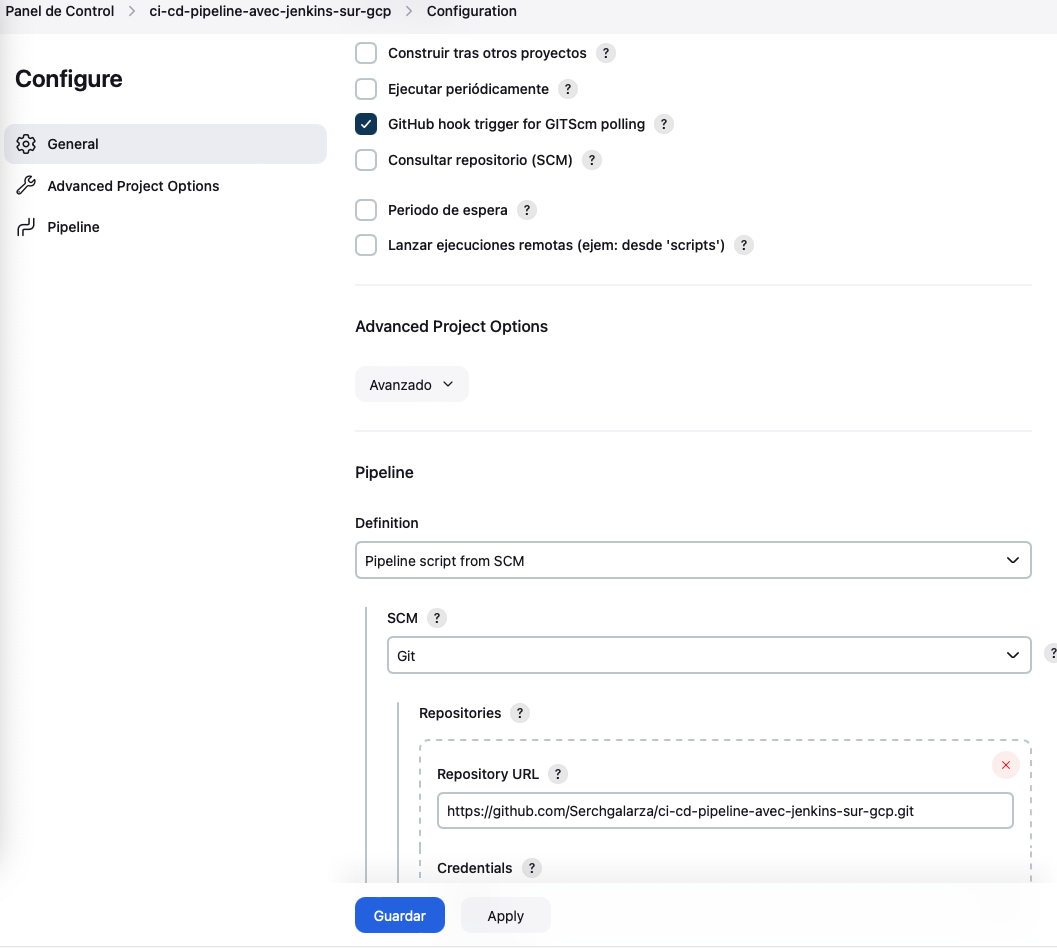
1. **Accéder à Jenkins** : Connectez-vous à votre instance Jenkins via l'URL http://34.19.104.169:8080/.
2. **Créer un Nouveau Job** :
   * Cliquez sur le symbole + ou "New Item" pour créer un nouveau job.
   * Entrez le nom de votre projet, par exemple : ci-cd-pipeline-avec-jenkins-sur-gcp.
   * Sélectionnez "Pipeline" comme type de job.
   * Cliquez sur "OK" pour créer le job.



1. **Configurer le Job** :
   * **Description** : Ajoutez une petite description pour votre job si nécessaire.
   * **Build Triggers** : Sous la section "Build Triggers", sélectionnez l'option "GitHub hook trigger for GITScm polling". Cela permet à Jenkins de déclencher le job automatiquement à chaque commit sur GitHub.
   * **Pipeline** : Sous la section "Pipeline", choisissez l'option "Pipeline script from SCM".
   * **SCM** : Sélectionnez "Git" comme système de gestion de versions.
     + **URL du Repository** : Entrez l'URL de votre repository GitHub :

https://github.com/Serchgalarza/ci-cd-pipeline-avec-jenkins-sur-gcp.git

* + - **Branch** : Pour cet exemple, utilisez la branche feature/ajouter-jenkinsfile. Vous pouvez également utiliser main ou master si c'est la branche principale.
  + Cliquez sur "Save" pour enregistrer les configurations du job.



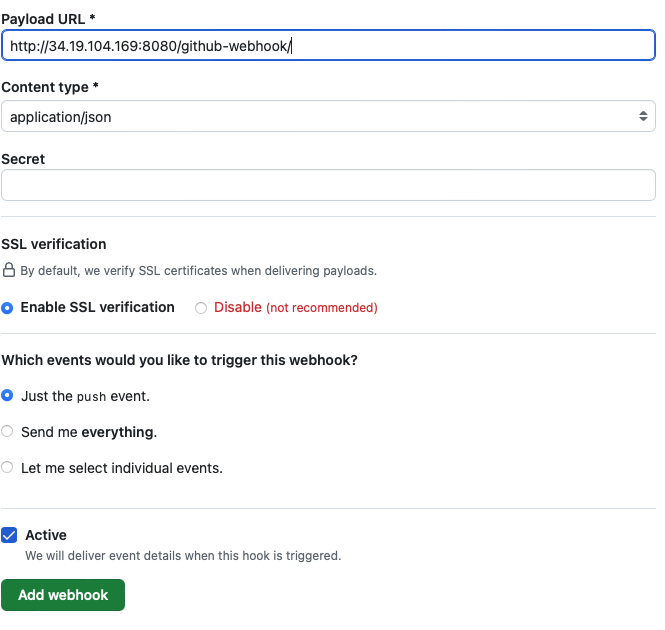
**5. Configurer le Webhook sur GitHub**

Pour que Jenkins se déclenche automatiquement à chaque commit, vous devez configurer un Webhook dans votre dépôt GitHub :

1. **Accéder aux Paramètres du Dépôt** :
   * Allez sur la page de votre dépôt GitHub : https://github.com/Serchgalarza/ci-cd-pipeline-avec-jenkins-sur-gcp.
   * Cliquez sur l'onglet "Settings".
2. **Ajouter un Webhook** :
   * Dans le menu de gauche, sélectionnez "Webhooks".
   * Cliquez sur "Add webhook".
   * **Payload URL** : Entrez l'URL suivante :

http://34.19.104.169:8080/github-webhook/

* + **Content type** : Sélectionnez application/json.
  + **Which events would you like to trigger this webhook?** : Sélectionnez "Just the push event".
  + Cliquez sur "Add webhook" pour enregistrer le webhook.



**Conclusion**

Une fois ces étapes complètes, votre Jenkins est configuré pour surveiller les commits dans votre dépôt GitHub et déclencher automatiquement le pipeline CI/CD défini dans votre Jenkinsfile. Vous pouvez vérifier le bon fonctionnement en effectuant un commit sur votre dépôt GitHub et en observant si le pipeline Jenkins se déclenche correctement.

