Déclencher un Pipeline Jenkins avec GitHub

***Objectif :*** Automatiser l'exécution d'un script "Hello World" à l'aide d'un pipeline Jenkins qui se déclenche à chaque fois que le fichier README.md est modifié et poussé sur GitHub.

**Prérequis :**

- Un compte GitHub.

- Un serveur Jenkins avec le plugin GitHub installé et configuré.

- Connaissances de base en Git.

**Instructions Détaillées**

**Partie 1 : Préparation du Dépôt GitHub**

**1. Créer un Nouveau Dépôt GitHub :**

- Connectez-vous à GitHub et créez un nouveau dépôt nommé jenkins-hello-world.

- Clonez le dépôt sur votre machine locale.

**2. Ajouter un Script "Hello World" :**

- ***Pour Python :*** Créez un fichier hello.py avec le contenu suivant :

print("Hello, World from Jenkins Pipeline!")

- ***Pour Java :*** Créez un fichier HelloWorld.java avec le contenu suivant :

public class HelloWorld {

public static void main(String[] args) {

System.out.println("Hello, World from Jenkins Pipeline!");

}

}

- Ajoutez un fichier README.md avec une description simple de votre projet.

**3. Poussez vos Modifications :**

- Utilisez Git pour ajouter, commettre et pousser vos fichiers sur GitHub.

**Partie 2 : Configuration du Pipeline Jenkins**

**1. Créer un Nouveau Job Pipeline dans Jenkins :**

- Dans Jenkins, sélectionnez "Nouveau Job".

- Nommez votre job (par exemple, HelloWorldPipeline), sélectionnez "Pipeline" et créez-le.

**2. Configurer le Pipeline pour Utiliser Votre Dépôt GitHub :**

- Dans la configuration du job, dans la section "Pipeline", choisissez "Pipeline script from SCM".

- Sélectionnez "Git" comme SCM.

- Entrez l'URL de votre dépôt GitHub et spécifiez la branche (par exemple, main).

- Dans le champ "Script Path", entrez le nom de votre Jenkinsfile (que vous allez créer dans l'étape suivante).

**3. Créer un Jenkinsfile dans Votre Dépôt :**

- Créez un fichier nommé Jenkinsfile à la racine de votre dépôt avec le contenu suivant :

pipeline {

agent any

stages {

stage('Build') {

steps {

script {

// Choisissez la commande en fonction de votre script

sh 'python hello.py' // Pour Python

// sh 'javac HelloWorld.java && java HelloWorld' // Pour Java

}

}

}

}

}

- Poussez le Jenkinsfile dans votre dépôt GitHub.

**4. Configurer le Pipeline pour Utiliser Votre Dépôt GitHub avec Vérification Périodique :**

* + Dans la configuration du job, dans la section "Pipeline", choisissez "Pipeline script from SCM".
  + Sélectionnez "Git" comme SCM.
  + Entrez l'URL de votre dépôt GitHub et spécifiez la branche (par exemple, main).
  + Dans le champ "Script Path", entrez le nom de votre Jenkinsfile.
  + Pour activer la vérification périodique, allez dans la section "Build Triggers" de la configuration du job et sélectionnez "Poll SCM". Dans le champ de programmation, vous pouvez entrer un horaire selon la syntaxe cron. Par exemple, H/5 \* \* \* \* pour vérifier le dépôt toutes les cinq minutes.

**Partie 3 : Testez Votre Configuration**

- Modifiez le fichier README.md dans votre dépôt GitHub, puis commettez et poussez la modification.

- Cette action devrait déclencher automatiquement votre pipeline Jenkins.

- Vérifiez que le pipeline s'exécute correctement et affiche le message "Hello, World from Jenkins Pipeline!" dans la console de sortie Jenkins.

Dans cet exercice, la modification du README.md sert de déclencheur pour votre pipeline Jenkins, illustrant comment les modifications de code peuvent automatiquement déclencher des processus CI/CD, renforçant les pratiques d'intégration et de déploiement continu.

**5. (BONUS) À la place de la vérification périodique, configurer le Webhook GitHub pour Déclencher le Pipeline :**

- Dans votre dépôt GitHub, allez dans "Settings" > "Webhooks" et ajoutez un nouveau webhook.

- L'URL du webhook sera l'URL de votre serveur Jenkins suivi de /github-webhook/ (par exemple, http://your-jenkins-server.github-webhook/).

- Choisissez "Just the push event".

- Activez le webhook.

**Livrable :**

Soumettez un rapport sur l'exercice "Déclencher un Pipeline Jenkins avec GitHub", incluant des captures d'écran comme preuves :

**1. Dépôt GitHub :** Montrez le dépôt avec hello.py/HelloWorld.java, README.md, et Jenkinsfile.

**2. Configuration Jenkins :** Capture de la configuration du pipeline Jenkins, montrant "Pipeline script from SCM" avec l'URL du dépôt et le chemin du Jenkinsfile.

**3. Vérification SCM :** (Si applicable) Montrez la programmation cron dans "Build Triggers" pour la vérification périodique.

**4. Pipeline en Action :** Capture du pipeline Jenkins exécutant suite à une modification du README.md.

**5. Résultat du Pipeline :** Montrez la sortie de la console Jenkins, avec l'exécution réussie du script.

**6. (Optionnel) Webhook GitHub :** Si un webhook a été configuré, incluez sa configuration.

**Consignes :**

- Soyez clair et organisé.

- Chaque capture doit être décrite brièvement.

- Assurez-vous de la lisibilité des captures.

- Soumettez en PDF ou Word.