Git et Git Hub Dans un Contexte DevOps

Partie 2 – Connaissances Avancées

1. Introduction :

Ce TP avancé vise à approfondir vos connaissances en Git et GitHub, en mettant l'accent sur les aspects plus avancés utilisés généralement par les ingénieurs DevOps. Vous allez vous familiariser avec des fonctionnalités avancées telles que les branches, les fusions, les conflits de fusion et les flux de travail de développement. Le TP est conçu pour durer au moins 3 heures.

Durée : 3 heures (minimum)

Prérequis :
- Un environnement de développement (IDE)
- Git installé sur votre machine
- Un compte GitHub

1. Exercice 1: Création et gestion des branches (30 minutes)

1. Clonez un repository GitHub vide sur votre machine en utilisant la commande `git clone <repository_url>`.
2. Créez une nouvelle branche appelée `feature/ma-nouvelle-fonctionnalite` avec la commande `git branch feature/ma-nouvelle-fonctionnalite`.
3. Passez sur la nouvelle branche en utilisant la commande `git checkout feature/ma-nouvelle-fonctionnalite`.
4. Effectuez quelques modifications et créez un commit sur cette branche.
5. Passez à la branche principale (généralement `master` ou `main`) avec la commande `git checkout main`.
6. Fusionnez la branche de fonctionnalité dans la branche principale avec la commande `git merge feature/ma-nouvelle-fonctionnalite`.
7. Supprimez la branche de fonctionnalité avec la commande `git branch -d feature/ma-nouvelle-fonctionnalite`.

2. Exercice 2: Gestion des conflits de fusion (60 minutes)

1. Créez une autre branche appelée `hotfix/correction-bug` à partir de la branche principale.
2. Effectuez des modifications dans la même partie de code sur la branche `hotfix/correction-bug` et sur une autre branche (peut-être une branche de fonctionnalité ou de développement).
3. Essayez de fusionner la branche `hotfix/correction-bug` dans la branche principale en utilisant `git merge hotfix/correction-bug`.
4. Identifiez et résolvez les conflits de fusion en modifiant manuellement les fichiers concernés.

5. Effectuez un commit pour résoudre les conflits et finaliser la fusion.
6. Poussez les modifications vers le repository distant.

3. Exercice 3 : Flux de travail de développement (60 minutes)

1. Identifiez un workflow de développement approprié pour votre projet (par exemple, le flux de travail GitFlow).
2. Créez différentes branches de fonctionnalités ou de développement pour simuler différentes tâches.
3. Effectuez des modifications et des commits sur chaque branche.
4. Fusionnez les branches de fonctionnalités dans la branche de développement principale.
5. Poussez les modifications vers le repository distant.
6. Optionnel : Effectuez une simulation de déploiement en utilisant une branche de déploiement ou une étiquette de version.

4. Exercice 4 : Collaboration avancée avec Git et GitHub (30 minutes)

1. Reprenez l'exercice de collaboration du TP précédent.
2. Utilisez les fonctionnalités de branches distantes et de pull requests dans GitHub pour demander à votre partenaire de fusionner vos modifications.
3. Revoyez les pull requests, commentez et fusionnez les modifications.
4. Effectuez des opérations de synchronisation pour récupérer les changements de votre partenaire localement en utilisant `git pull`.
5. Gérez les conflits, le cas échéant, en fusionnant les modifications sur votre machine.

Conclusion :

Ce TP avancé vous a permis de vous familiariser avec des fonctionnalités plus avancées de Git et GitHub, nécessaires dans les workflows DevOps. Vous avez appris à créer et gérer des branches, à résoudre les conflits de fusion et à mettre en place un flux de travail de développement efficace. Continuez à pratiquer régulièrement pour renforcer vos compétences en matière de gestion de versions et de collaboration dans un environnement DevOps.