\*\*Rapport de Stage : Gestion de Version avec Git\*\*

Ce rapport présente les principaux concepts et commandes liés à l'utilisation de Git, un système de gestion de version distribué, pour le suivi et la gestion des modifications de code source dans un projet.

\*\*Création d'un Projet et Communication avec GitHub :\*\*

- Pour commencer, un projet est créé sur GitHub, une plateforme d'hébergement de code en ligne.

- Les fichiers du projet sont copiés localement en utilisant Git, établissant ainsi une connexion entre le référentiel local et GitHub.

\*\*Gestion des Branches :\*\*

- Git permet une gestion flexible des branches, que ce soit en local ou à distance.

- Les branches sont créées localement avec la commande `git branch` et on bascule sur une branche spécifique avec `git checkout nom-de-la-branche`.

- Utilisation de `git pull` pour récupérer les dernières modifications d'une branche à distance.

\*\*Fusion de Branches :\*\*

- Pour fusionner une branche avec la branche principale (par exemple `main`), on suit ces étapes :

1. Se positionner sur la branche à fusionner avec `git checkout nom-de-la-branche`.

2. Mettre à jour la branche principale avec `git pull origin main`.

3. Utiliser `git merge` pour fusionner les modifications de la branche dans la branche principale.

4. En cas de conflits, résoudre les conflits, ajouter les fichiers modifiés avec `git add`, puis valider les modifications avec `git commit`.

\*\*Revenir en Arrière après une Fusion Incorrecte :\*\*

- Si une fusion a été effectuée incorrectement, on peut revenir en arrière avec `git reset --hard HEAD^`.

\*\*Suppression des Branches :\*\*

- Suppression d'une branche en local : `git branch -d nom-de-la-branche`.

- Suppression d'une branche à distance : `git push origin --delete nom-de-la-branche`.

\*\*Suivi des Modifications :\*\*

- Utilisation de `git log` pour visualiser toutes les opérations effectuées uniquement sur la branche courante.

- Utilisation de `git branch -a` pour voir toutes les branches locales et à distance.

\*\*Enregistrement des Modifications :\*\*

- `git add .` pour préparer les modifications à être incluses dans le prochain commit.

- `git commit -m "message de commit"` pour enregistrer les modifications avec un message descriptif.

- `git push` pour envoyer les modifications vers GitHub.

En résumé, Git est un outil puissant pour la gestion de version, permettant le suivi efficace des modifications, la collaboration en équipe et le contrôle des branches de développement. L'utilisation de commandes spécifiques permet de maintenir un historique clair et de gérer les modifications de manière méthodique dans un projet de développement logiciel.