Procédures Git pour la gestion des branches et du rebase

Étape de Création d'une Branche

Cette section décrit le processus standard pour créer une nouvelle branche de développement.

1. **Se positionner sur la branche principale du sprint en cours :**
   - **Exemple :** `SPRINT-111`
2. **Créer une nouvelle branche :**
   - **Utilisez la commande : `git checkout -b Nouvelle_branche`**
3. **Finaliser les développements :**
   - **Une fois les modifications terminées, effectuez un commit.**
4. **Créer une Merge Request (dans GitLab) :**
   - **Ouvrez une Merge Request pour qu'une autre personne puisse effectuer la revue de votre code.**
5. **Prendre en compte les retours :**
   - **Si des retours sont faits lors de la revue, assurez-vous de les intégrer.**

Étape du Rebase

Cette section explique comment effectuer un rebase lorsque votre branche de développement est en retard par rapport à la branche principale.

- **Scénario : Votre branche (Nouvelle_branche `PW_PRI_SIM_APA`), créée à partir de `SPRINT-111`, n'a que 3 commits locaux et est en retard par rapport à la branche principale.**

1. **Se positionner sur la branche principale :**
   - `git checkout SPRINT-111`
2. **Récupérer les dernières modifications distantes :**
   - `git fetch origin`
3. **Réinitialiser la branche locale à la version distante :**
   - `git reset --hard origin/SPRINT-111`
4. **Retourner sur votre branche de développement :**
   - `git checkout PW_PRI_SIM_APA`
5. **Effectuer le rebase :**
   - `git rebase --onto origin/SPRINT-111 PW_PRI_SIM_APA~3 PW_PRI_SIM_APA`
6. **Forcer le push de la branche rebasée :**
   - `git push origin PW_PRI_SIM_APA -f`

Note importante : **Effectuez un `git pull origin` uniquement sur les branches Master ou pre-master.**

Dans le Cadre d'un E2E (End-to-End) avec Forte Liaison

Cette section aborde les opérations Git pour des développements E2E qui dépendent fortement les uns des autres.

Scénario 1 : Je travaille sur `branche_E2E_1`

1. **Ajouter et commiter les modifications :**
   - `git add .`
   - `git commit -m "Message de commit"`
2. **Pousser les modifications sur la branche :**
   - `git push branche_E2E_1`

Scénario 2 : Je crée `branche_E2E_2` à partir de `branche_E2E_1`

1. **Créer la nouvelle branche :**
   - `git checkout -b branche_E2E_2`

Scénario 3 : `branche_E2E_1` est mergée et je dois rebaser `branche_E2E_2`

1. **Se positionner sur la branche principale :**
   - `git checkout SPRINT-111`
2. **Récupérer les dernières modifications distantes :**
   - `git fetch origin`
3. **Réinitialiser la branche locale à la version distante :**
   - `git reset --hard origin/SPRINT-111`
4. **Retourner sur branche_E2E_1 :**
   - `git checkout branche_E2E_1`
5. **Effectuer le rebase de branche_E2E_2 sur la branche principale :**
   - `git rebase --onto origin/SPRINT-111 branche_E2E_2~nbCommit branche_E2E_2` **(où nbCommit représente le nombre de commits sur branche_E2E_2)**
6. **Pousser les modifications de `branche_E2E_2` :**
   - `git push origin branche_E2E_2`

***Note importante :* Dans le cas où `branche_E2E_1` n'est pas encore mergée, lors de la création de la Merge Request pour `branche_E2E_2`, définissez `branche_E2E_1` comme branche cible.**