Formation Git & GitHub - Supports de cours

Introduction à Git & GitHub

Pourquoi Git et GitHub ?

- Git : système de gestion de versions qui permet de suivre les modifications du code.
- GitHub : plateforme en ligne qui facilite la collaboration autour de projets versionnés avec Git.
- Avantages : suivi des modifications, collaboration simplifiée, gestion des conflits.

Concepts clés

- Repository (dépôt) : Espace de stockage contenant les fichiers et l'historique des modifications.
- Commit : Enregistrement d'une modification dans l'historique du projet.
- Branch (branche) : Version parallèle d'un projet pour travailler sans affecter la version principale.
- Pull Request (PR) : Proposition de fusionner une branche avec une autre après révision du code.
- Issue : Ticket utilisé pour signaler un problème ou une nouvelle tâche à accomplir.

Utilisation de GitHub sans ligne de commande

1 Créer un repository

1. Aller sur GitHub.
2. Cliquer sur le bouton "+" > New Repository.
3. Donner un nom au projet, ajouter une description et choisir Public ou Private.
4. Cocher "Initialize this repository with a README".
5. Cliquer sur "Create Repository".

2 Travailler avec les Issues

1. Aller sur l'onglet Issues.
2. Cliquer sur "New Issue" et donner un titre + description.
3. Ajouter des labels, assigner des collaborateurs si nécessaire.
4. Cliquer sur "Submit new issue".

3 Création d'une branche depuis une Issue

1. Dans l'Issue, chercher "Create a branch" et cliquer dessus.
2. Valider la création de la branche.
3. La branche est automatiquement créée à partir de main.

4 Modifier des fichiers et faire un commit

1. Aller sur la branche créée.
2. Modifier un fichier directement via l'interface web.
3. Ajouter un message de commit clair.
4. Cliquer sur "Commit changes".

5 Créer une Pull Request (PR)

1. Aller sur l'onglet Pull Requests.
2. Cliquer sur "New Pull Request".
3. Vérifier que la branche source est bien sélectionnée.
4. Ajouter un titre et une description clairs.
5. Assigner des reviewers et labels si nécessaire.
6. Cliquer sur "Create Pull Request".

6 Fusionner une Pull Request

1. Une fois la PR validée par un reviewer, cliquer sur "Merge pull request".
2. Supprimer la branche si elle n'est plus nécessaire.

Ajouter des collaborateurs sur GitHub

1 Accéder aux paramètres du repository

1. Aller sur GitHub et ouvrir le repository concerné.
2. Cliquer sur l'onglet "Settings" ().
3. Dans le menu de gauche, cliquer sur "Collaborators" (sous "Access").

2 Ajouter un collaborateur

1. Cliquer sur le bouton "Add people".
2. Entrer le nom d'utilisateur GitHub ou l'adresse email du collaborateur.
3. Cliquer sur "Add collaborator".

4. Le collaborateur recevra une invitation par email et devra l'accepter.

3 Définir les permissions

- Read (Lecture) Peut voir le repo, mais ne peut pas modifier.
- Write (Écriture) Peut pousser des commits, gérer des issues et pull requests.
- Admin Peut modifier les paramètres du repo et gérer les collaborateurs.
- Maintain Peut gérer les issues, pull requests et branches.
- Triage Peut gérer et organiser les issues et pull requests.

4 Ajouter des collaborateurs via une organisation (si applicable)

Si le repository appartient à une organisation, tu peux :

- Ajouter des membres à l'organisation via Settings > Members.
- Assigner des rôles via Teams pour mieux gérer les accès.

5 Vérifier les accès

Le collaborateur peut maintenant cloner, pousser des commits et gérer les pull requests selon les permissions accordées.

Commande pour cloner :

```bash
git clone https://github.com/ton-utilisateur/nom-du-repository.git
```

Bonnes pratiques GitHub

- Ne jamais travailler directement sur main.
- Créer des branches pour chaque nouvelle fonctionnalité ou correction.
- Faire des commits clairs et atomiques.
- Utiliser des Pull Requests pour toute modification.
- Assigner des reviewers pour valider les changements.

Exercice pratique

Objectif : Simuler une collaboration sur un projet GitHub.

1. Créer un repository avec un README.md.
2. Créer une Issue décrivant une modification à apporter.
3. Créer une branche associée à l'Issue.
4. Modifier un fichier et faire un commit.
5. Créer une Pull Request et demander une revue.
6. Fusionner la Pull Request après validation.

Vous êtes maintenant prêts à utiliser GitHub pour gérer vos projets en équipe !