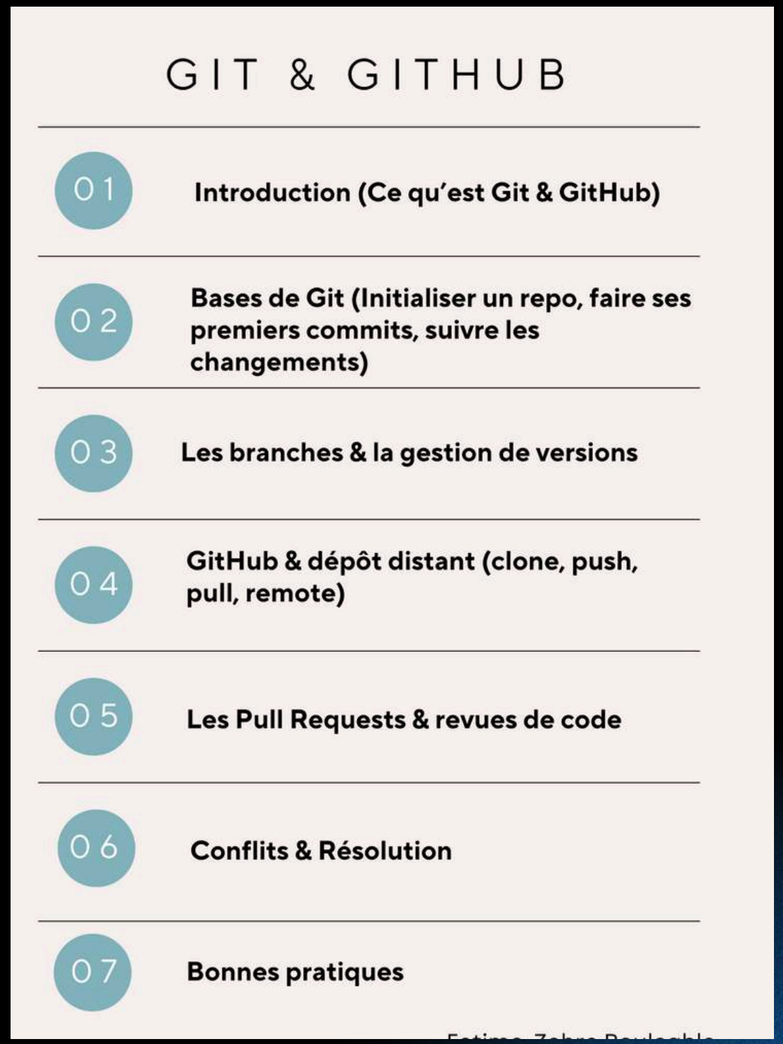
29 Novembre 2025

GIT & GITHUB

Fatima-Zahra BOULAGHLA
Mohammed SABER

Objectifs du Workshop



Git

C'est un système de contrôle de version distribué (DVCS).
Il prend des instantanés (snapshots) du projet, pas des différences de fichiers.

Chaque utilisateur a une copie complète de l'historique du dépôt (d'où le terme distribué).

Installé en local sur votre machine pour versionner du code

```
git --version
```

```
where git      # sous Windows
which git      # sous Linux/Mac
```

```
git config --global user.name "Ton Nom"
git config --global user.email "ton.email@example.com"
```

GitHub

GitHub est une plateforme web qui héberge des dépôts Git. Elle est essentielle pour le travail d'équipe et la collaboration en fournissant :

- Un lieu centralisé pour cloner et pousser (push) le travail.
- Un mécanisme pour l'ouverture de requêtes de tirage (Pull Requests), permettant la discussion et la révision de code avant la fusion (merge).

Connecter GIT à GitHub avec une clé SSH valide

```
git@github.com: Permission denied (publickey).
fatal: Could not read from remote repository.
```

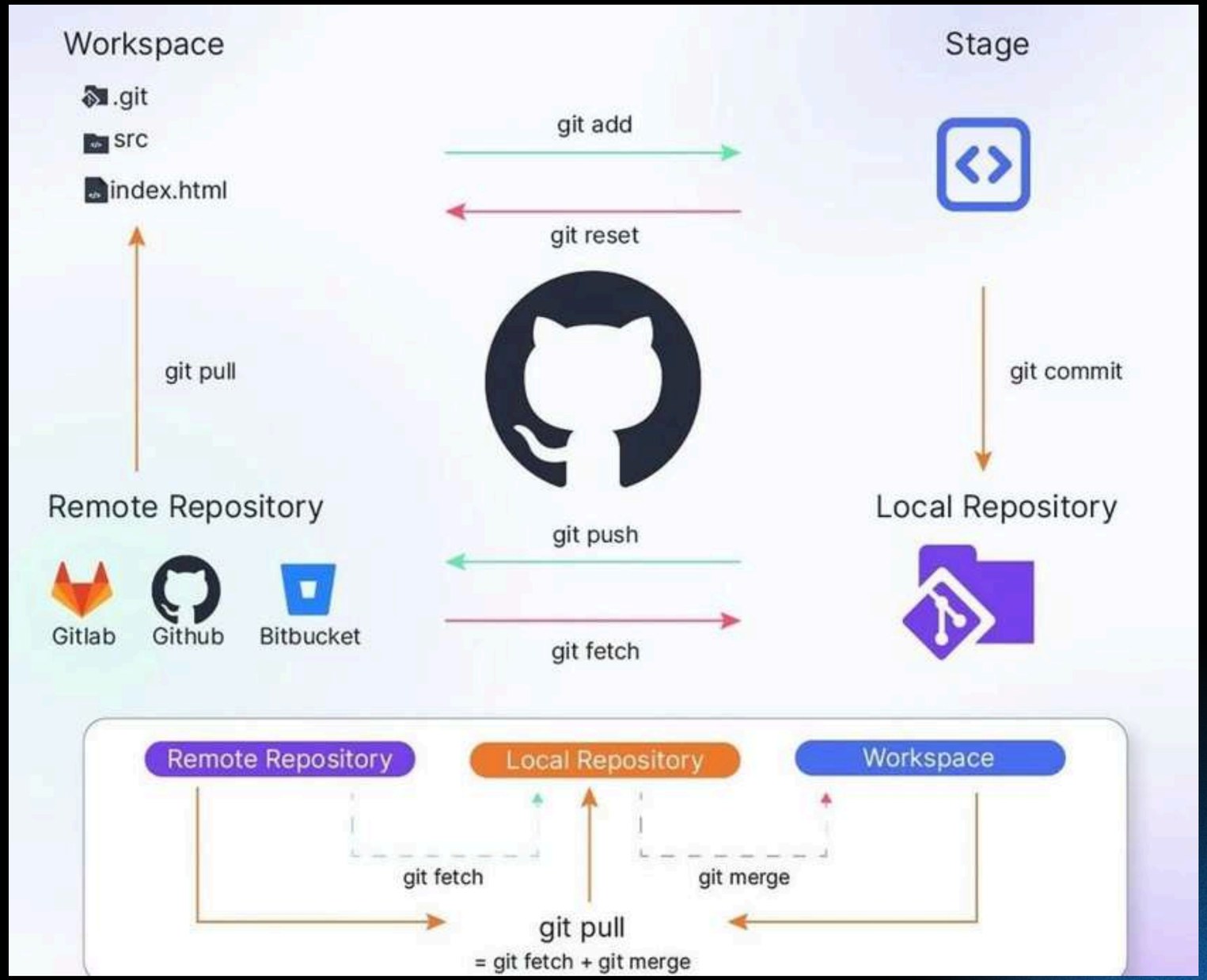
```
ssh-keygen -t ed25519 -C "ton.email@example.com"
```

```
 eval "$(ssh-agent -s)"
 ssh-add ~/.ssh/id_ed25519
```

```
Get-Content ~/.ssh/id_ed25519.pub
```

```
Hi FZ-Boulaghla! You've successfully authenticated, but GitHub does not provide shell access.
```

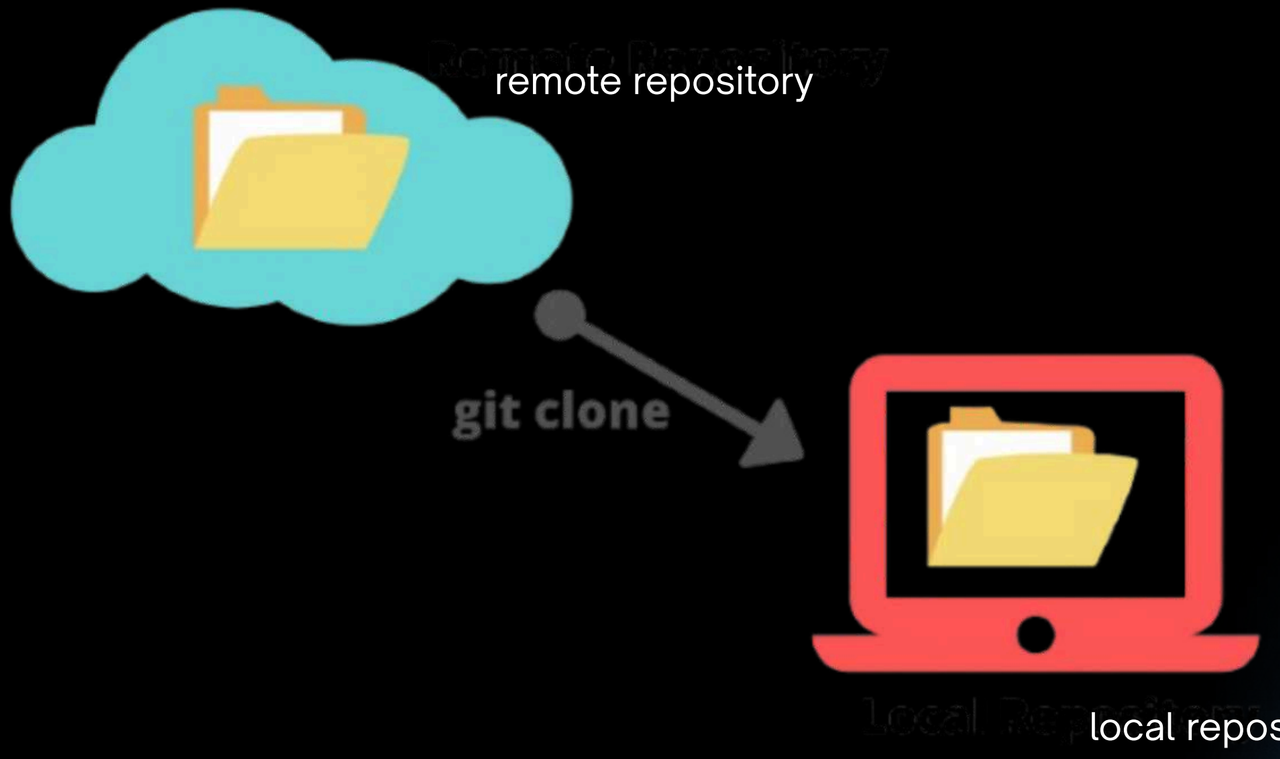
Comment fonctionne Git avec Github?



Workflow

Pour commencer un projet Git, il y a 2 méthodes :

1. Initialiser un dépôt en local (git init)
2. Cloner un dépôt distant (git clone <url_du_repo>)



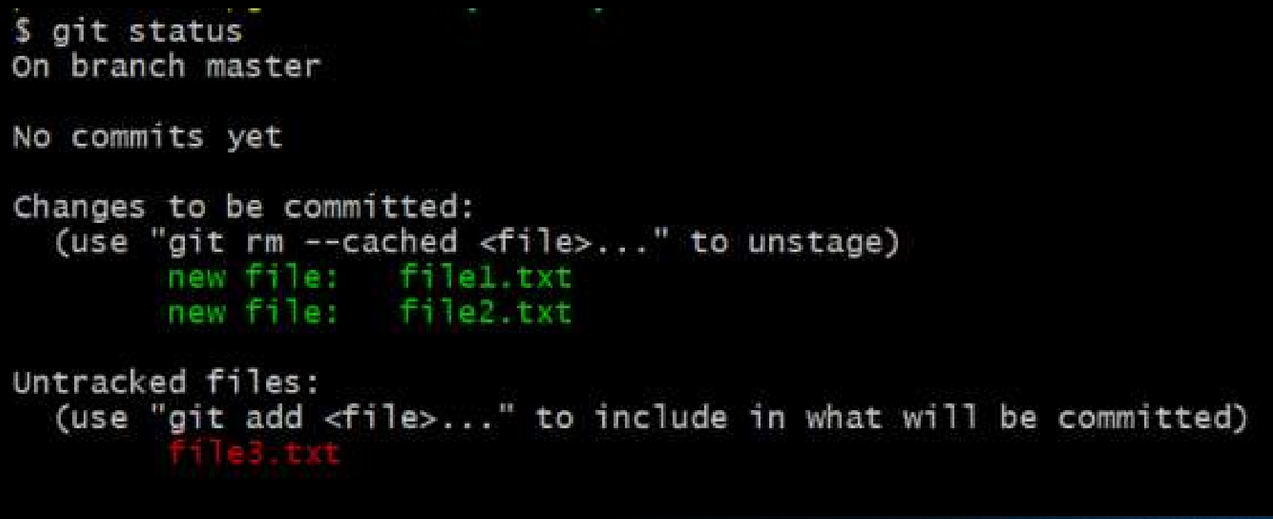
Les 3 États d'un Fichier (git status)

1. **Untracked (Non suivi)**

2. **Modified**
   - A été modifié depuis le dernier commit

3. **Staged (Indexé)**
   - Fichier ajouté à la zone de staging
   - Prêt à être commité

4. **Committed**
   - Données sauvegardées dans la base Git locale
   - Snapshot permanent dans l'historique



git commit:

La Commande de Base (Option -m):

```
git commit -m "feat(project-demo): First commit" -m "ce code permet de ...
j'ai fait ça ..."
```

La Commande de Base (Sans -m):

```
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch master
# Changes to be committed:
#       modified:   index.html
fix(index.html):nouvelles modifications
pour tester 'git commit' j'ai ajouté des nouvelles modifications
~
~
```

Résultat:

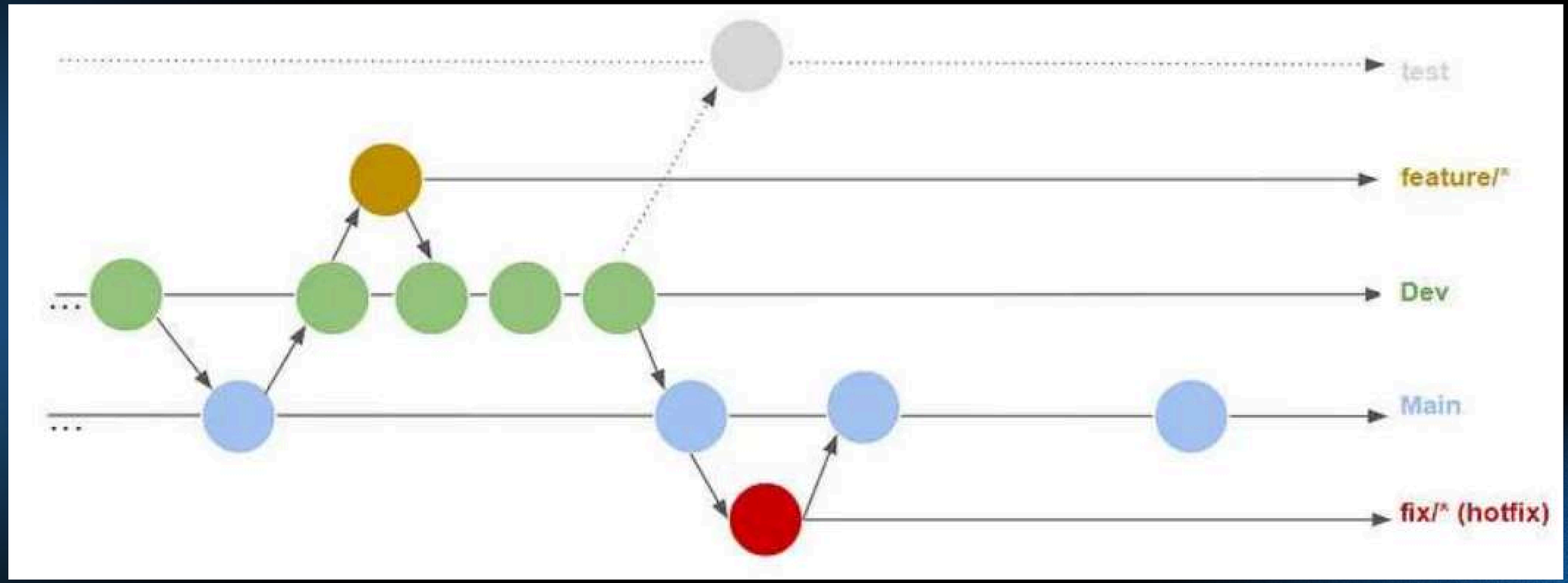
```
f5994c5] fix(index.html):nouvelles modifications pour tester 'git commit' j'ai ajouté des nouvelles modifi
changed, 1 insertion(+), 1 deletion(-)
```

Branches & Fusion

- Pourquoi créer des branches?
- Travailler sur une fonctionnalité sans toucher main
- Fusion de branches

```
git checkout -b ma-branche
# modifier index.html
git add .
git commit -m "Ajout sur branche"
git switch main
git merge ma-branche
```

GITFLOW



Branche	Rôle	Qui l’utilise / Quand
main (ou master)	Branche stable en production	Tout le monde la respecte , seules les releases & hotfix vont dessus
develop (ou dev)	Branche d’intégration :Contient le code des nouvelles features testées	Les devs fusionnent leurs feature/* ici
feature/ma-fonction	Développement d’une fonctionnalité	Créée pour bosser sur une feature précise
release/v1.2	Prépare une mise en prodTests finaux, corrections	Permet de figer le code avant passage en main
hotfix/bug-urgent	Répare un bug en prod	Branche créée en urgence à partir de main
test (optionnelle)	Branche de pré-prod	Pour tester des trucs sans impacter develop

Pull Request (PR) / Merge Request

Une PR est une demande pour fusionner ton travail dans la branche principale (main).
Ça permet aux autres de voir ce que tu as changé, de commenter, et de valider avant d’intégre

- Important : Les Pull Requests se font sur GitHub, pas directement en ligne de commande.

Conflits

Le Problème : Un conflit survient uniquement lorsque deux branches différentes ont modifié les mêmes lignes dans le même fichier, ou si un fichier a été supprimé dans une branche et modifié dans l'autre.

La Cause : Git ne peut pas deviner quelle modification est correcte,il s'arrête et demande au développeur d'intervenir manuellement.

Pull Request

`git checkout main`	Se positionner sur la branche principale
`git checkout -b feat/ajout-bouton`	Créer une branche pour une fonctionnalité
`git add .`	Ajouter / modifier
`git commit -m "feat: ajout d'un bouton de test"`	Commit
`git push -u origin feat/ajout-bouton`	Pousser la branche
`git checkout main`	Aller sur main
`git merge feat/ajout-bouton`	Fusionner la branche
`git push origin main`	Pousser les changements

Conflits

`git merge nom-de-branche`	Fusionner une branche
`code fichier.txt`	Ouvrir le fichier (Visual Studio Code)

Bonnes pratiques, ressources

- Rédiger des messages de commit clairs;
- Ne sauvegardez pas les données sensibles (passwords, keys);
- Utiliser le fichier **.gitignore**

https://git-scm.com/docs
Pro Git eBook

MERCI!