TP 1 Git

Lab 1 Basics

Lab 1, 1

Un répertoire .git a bien été créé après la commande **git init**

La commande **git status** nous propose d'utiliser la commande **git add** pour faire suivre le fichier file1.txt.

Après avoir utilisé la commande **git add**, **git status** nous indique que le fichier file1.txt sera suivi.

Le message de commit indique quels fichiers ont été créés ou modifiés.

Lab 1, 2

La commande **git diff** nous donne toutes les lignes modifiées pour chaque fichier.

La commande **git commit --amend** a permis de modifier le dernier message de commit.

git log nous donne des informations sur les derniers commits effectués.

La commande **git show** nous donne des informations sur un commit en particulier, ou sur le dernier commit si aucun n'est précisé.

git checkout -- a permis de supprimer les modifications faites sur un fichier.

La commande **git reset** a permis de retirer un fichier ajouté avec **git add**.

Lab 1, 3

La commande **git clone git@github.com:saussact/git-training-repo.git** a permis de cloner le repo distant par ssh.

La commande **git fetch** va récupérer les modifications qui ont été faites sur la branche courante.

La commande **git pull** va récupérer les modifications faites sur notre branche courante et les merge.

Puis, la commande **git push** va ajouter nos modifications de la branche courante locale sur la branche courante distante.

Lab 2 Branching

Lab 2,1

La commande **git branch lab2-blanchard** va créér une nouvelle branche, qui sera ensuite accessible avec la commande **git checkout**.

Après avoir effectué des modifications sur la branche créée, et avoir merge cette branche sur la branche main, j'ai obtenu le fast-forward suivant :

```
Fast-forward
 users/KillianBlanchard/file_2.txt | 1 +
 users/KillianBlanchard/file_3.txt | 0
 2 files changed, 1 insertion(+)
 create mode 100644 users/KillianBlanchard/file_3.txt
```

Lab 2, 2

Après avoir effectué un commit sur la branche lab2-blanchard en modifiant le fichier file_2.txt, un commit sur la branche main en modifiant le fichier file_1.txt, puis merge la branche lab2-blanchard sur main, et avoir écris un message de commit dans l'éditeur, j'ai optenu la sortie suivante :

```
Merge made by the 'recursive' strategy.
 users/KillianBlanchard/file_2.txt | 1 +
 1 file changed, 1 insertion(+)
```

La branche lab2-blanchard a ensuite été supprimée avec la commande **git branch -D**.

Lab 3 Merge conflicts

La commande **git checkout -b lab3-blanchard** a permis de créér une nouvelle branche, et de se déplacer directement dessus.

Après avoir effectué des commit sur la branche lab3-blanchard en modifiant le fichier file_1.txt, un commit sur la branche main en modifiant le fichier file_1.txt, puis en tentant de merge la branche lab3-blanchard sur main, j'ai obtenu le message suivant :

```
Fusion automatique de users/KillianBlanchard/file_1.txt
CONFLIT (contenu) : Conflit de fusion dans users/KillianBlanchard/file_1.txt
La fusion automatique a échoué ; réglez les conflits et validez le résultat.
```

Après avoir modifié le fichier file_1.txt pour résoudre le conflit, je l'ai ajouté avec **git add** puis effectué un commit.

Lab 4 Rebasing

Après avoir créé le fichier file_4.txt sur la branche main, et modifié le fichier file_1.txt sur la branche lab4-blanchard, la commande **git rebase main** depuis la branche lab4-blanchard a permis d'ajouter le fichier file_4.txt sur cette branche.

Lab 5 Workflow

La commande **git clone git@github.com:saussact/git-workflow.git** a permis de récupérer le workflow.

Après avoir effectué 4 commits sur la branche issue1-implement-feature-blanchard et utilisé la commande **git rebase -i main**, un éditeur s'est ouvert dans lequel j'ai remplacé le pick de 3 de mes commits par des squash.

Ensuite un nouvel éditeur s'est ouvert dans lequel j'ai rentré un message de commit.

J'ai ensuite merge cette branche sur la branche main, puis push mon repo local.