ATTENTION!!

Toujours faire un fetch avant de commencer à travailler, pour vérifier les dernières modifications et les inclure dans son code avec un pull.

On fait une branche différentes pour le developpement, voir par fonctionnalité. Quand le code est terminé on fera un merge vers la branche supérieure et on le communiquera à l'équipe.

COMMANDES GIT

Configuration:

git config --list --> pour voir les informations de configuration

git config --global user.name "votre nom" --> ajouter votre nom aux infos de la config

git config --global user.email "votre email" --> ajouter votre email aux infos de la config

Repository: (faire cd leCheminDuDossier)

git init --> création du directory git

git add "nom fichier" --> ajout du fichier dans l'Index (préparation de la validation => staging)

git add . --> ajout de tous les fichiers dans l'Index

git commit -m "votre commentaire" --> validation des modifications (en local)

git status --> permet de voir l'état du repository (fichier unstaged ou staged...)

git log --> permet de voir l'historique des commits

git log --oneline --> permet de voir l'historique des commits de manière plus courte

Remote Repository:

git clone "remote url" --> copie du repository distant sur votre machine locale

git pull --> mise à jour du repository local à partir du serveur (pull = fetch + merge)

git push --> pousse le travail sur le serveur (vers exterieur --> vers serveur)

git remote add origin "remote url" --> préparation pour envoyer projet existant vers serveur (faire un push par la suite)

git fetch --> récupère les données distantes sans rien modifier en local

Branches:

git branch --> indique le nom de la branche où nous nous trouvons

git branch "nom branche" --> création d'une branche

git checkout "nom branche" --> sélection de la branche sur laquelle nous voulons travailler

git checkout -b "nom branche" --> création + sélection de la branche

Merge:

git checkout "nom branche" --> se mettre sur la branche où nous voulons rapatrier le code

git merge "nom branche à merger" --> indique la branche où nous voulons prendre le code pour le fusionner

exemple: merge de la branche FeatureA sur master

git checkout master

git merge FeatureA

Modifications:

git reset HEAD "nom fichier" --> enlève le fichier de l'Index (unstaged)

git revert HEAD --> Annuler le dernier commit (inverse les modifications)

git checkout "NumeroCommit " "Nom fichier" --> Revenir en arrière sur les modifications du fichier (faire un commit ensuite)

git reset "NumeroCommit" --> permet de revenir en arrière sur l'historique (attention, on modifiera l'historique)(faire add et commit)