Sommaire

I. Introduction

Dans ce rapport, je vais passer en revue ce que j'ai appris aujourd'hui. Notre objectif était d'apprendre les bases de git et de comprendre comment travailler avec git localement. Git est un système de contrôle de version distribué permettant de suivre les modifications du code source au cours du développement du logiciel. Il est donc prudent de dire que c'est un outil précieux que nous, les étudiants de YOUCODE, devraient tous maîtriser si nous voulons nous appeler un jour des développeurs de logiciels chevronnée.

II. Structure de la journée

1. Explication de l'architecture de git par M. Tijani

M. Tijani nous a donné une explication simple mais très élaborée de l'architecture git.

2. Travail sur l'atelier

Nous avons commencé à travailler sur l'atelier donnée individuellement. Nous avons cherché sur le Web pour des solutions d'erreurs et avons lu la documentation de git. Et franchement, la recherche active de solutions nous a permis de mieux comprendre cet outil.

3. Live démo

J'ai fait une démonstration en direct au cours de laquelle j'ai expliqué à mes collègues ce que chaque commandement avait fait et pourquoi nous l'avions utilisé. Cette démonstration m'a permis de mieux comprendre le défi que représente le fait de tenter de transmettre des informations d'une manière compréhensible pour votre auditoire.

III. Description de l'atelier proposée

Cet atelier est assez simple mais il a été conçu de manière à faciliter la compréhension de cet outil par les débutants. Nous avons travaillé de manière individuelle, ce qui nous a permis de chercher et d'essayer de comprendre nous-mêmes

IV. L'atelier Git

J'ai utilisé la distribution Debian alors Pour installer git sur une Debian, il suffit de taper la commande :

```
apt-get install git-core gitk
```

Pour configurer le nom :

```
git config --global user.name "Med Mouiguina"
```

Pour configurer l'email :

```
git config --global user.email "mohamedmouiguina7@gmail.com"
```

Créer un répertoire de travail nommé “Atelier-Git”

```
mkdir Atelier-Git
```

Pour accedé au dossier / Atelier-Git

```
cd  /Atelier-Git
```

On se placer dans le dossier du projet. Puis on tape la commande suivante pour Initialiser le dépôt :

```
git init
```

Le dépôt est créé, un fichier .git est créé dans ce dossier. Il faudra ensuite créer des fichiers sources et les faire connaitre à Git. La commande “commit” sera utilisé, nous verrons cela un peu plus loin.

Vérifier le statut du répertoire :

```
git status
```

On crée une fichier index.html et l’ajoute quelque line de code avec un éditeur de code

```
touch index.html
```

Sauvegarder notre code

Lorsque nous modifions un fichier de notre code, il faut dire à Git de prendre les nouveaux changements dans celui-ci. Cette sauvegarde s’appelle un commit.

Lors de la commande :

```
git status
```

Nous avons vu que le fichier modifié était en rouge, ce qui veut dire que lors du commit il ne sera pas pris en compte. Il faudra lui préciser le nom lors du commit, il y a 3 possibilités :

```
git add fichier1 fichier2 # pour ajouter les 2 fichier au commit puis

git add . # pour ajouter tous les fichiers dans le répertoire faire attention au point

#Puis en faire une commit

git commit –m '' Initiale commit ''
```

Lorsque la commande commit est lancé, un éditeur de texte s'ouvre. Cela permet d'insérer un texte pour décrire le changement. Ce texte doit être court, par défaut, chaque commit ne devrait faire l'objet que d'un seul changement dans le code. Pour changer l'éditeur par défaut de git il suffit de faire :

```
git config --global core.editor "vim
```

Annuler un commit

Il peut arriver que nous ayons besoin d'annuler un commit que nous avons fait. Pour cela, nous avons la possibilité de vérifier tous les commits fait sur un projet :

```
git log
```

Avec cette commande, on peut voir tous les commits que nous avons effectué sur ce projet. Pour voir également les changements faits, on peut lancer la commande avec l'option -p :

```
git log -p
```

Pour annuler un commit de manière soft, voici plusieurs commandes possibles :

```
git reset –soft HEAD
```

On peut également utiliser plusieurs notations :

— HEAD : dernier commit

— HEAD^ : avant dernier commit

— HEAD^^ : avant-avant dernier commit

— HEAD~2 : pareil

— d6d98923868578a7f38dea79833b56d0326fcba1 : le numéro de commit.

Annuler/Supprimer un fichier avant un commit

Supposons que vous veniez d'ajouter un fichier à Git avec git add et que vous vous apprêtiez à le commiter. Cependant, vous vous rendez compte que ce fichier est une mauvaise idée et vous voudriez annuler votre git add. Il est possible de retirer un fichier qui avait été ajouté pour être commité en procédant comme suit :

```
git rm --cache index.html #suppose que le nom du fichier est index.html
```

V. Conclusion

Personnellement, j'ai eu une journée très productive, j'ai beaucoup appris. L'atelier était simple mais il me permet de mieux comprendre les commandes de base de git. Même si cet atelier était une tâche individuelle, nous expliquerions ce que nous avons appris les uns des autres, ce qui nous a vraiment permis de mieux comprendre le sujet.