Durant le projet, nous avons dû utiliser l’outil de gestion de versions Git ainsi que le service web d’hébergement de projet GitHub. Nous nous en sommes servi pour pouvoir implémenter des parties au projet après la réalisation de chaque tâche. Au tout début du projet, nous avons adopté une méthode d’utilisation de GitHub qui consistait à avoir chacun une branche à notre nom et pousser nos modifications à l’intérieur de cette même branche, puis nous faisions un « merge » de tous les branches dans le « main » à la fin du sprint. À la fin du premier sprint nous avons remarqué que ce n’était pas une bonne méthode de travail, ce n’est pas une bonne façon d’utiliser GitHub. Nous avons donc mis au point une nouvelle façon de travailler avec GitHub et les tâches que nous avions à réaliser. À chaque fois qu’un membre de l’équipe avait une nouvelle tâche d’assigner, il allait écrire le numéro de la tâche, son nom et la date dans un tableau dans OneNote, donc on pouvait garder une trace de qui s’est assignée quelle tâche. Ensuite, sur GitHub, il allait se créer une branche avec le nom « request#??? » les points d’interrogation étant bien sur le numéro de la tâche. Une fois que la requête avait été effectuée, la personne effectuait un « push » sur sa branche pour ensuite faire un « pull » de la branche « dev » sur sa branche pour pouvoir régler les conflits et s’assurer que tout fonctionnerait bien une fois dans le dev. Si tout fonctionnait et que l’assurance qualité avait été faite, il allait sur la branche « dev » et effectuait un « merge » avec sa branche. Donc, nous avons utilisé GitHub pour garder des traces des versions antérieures, modifications précédentes, des personnes qui ont effectué telle modification ainsi que pour pouvoir se partager les parties de notre projet entre les membres de notre équipe.

Tout au long du projet, nous avons utilisé ces commandes dans Git pour nous permettre de bien l’utiliser

* Git init – Pour initier un repository Git local
* Git clone – Pour cloner un repository Git sur le poste local
* Git branch – Pour créer une branche
* Git checkout – Pour changer de branche
* Git pull origin – Pour aller chercher la dernière version d’une branche
* Git push origin – Pour pousser les modifications dans une branche
* Git merge – Pour synchroniser deux branches ensemble
* Git remote -v – Pour savoir à quelle repository le projet actuel est connecté
* Git remote add origin – Pour ajouter un lien vers un repository
* Git remote rm origin – Pour supprimer un lien vers un repository
* Git add – Pour ajouter les modifications effectuées
* Git commit – Pour commit les modifications effectuées
* Git status – Pour voir les fichiers modifiés ou non depuis la dernière version

Voici la petite documentation que nous nous étions faits au cours de notre projet pour nous aider à utiliser Git.

Lien du repos : <https://github.com/Dolotboy/GreenHouse-Tech>

Pour commencer :

Git clone <https://github.com/Dolotboy/GreenHouse-Tech>

Git branch VOTRENOM

Git checkout VOTRENOM

Git remote set-url origin git remote add origin <https://github.com/Dolotboy/GreenHouse-Tech.git>

Pour changer le remote :

git remote rm origin

git remote add origin <https://github.com/Dolotboy/GreenHouse-Tech.git>

Quand une tâche est fini :

git add …

git commit –m "Push request#…"

git push origin request#...

git checkout dev

git pull origin dev

git merge request#…

git push origin dev

git branch (nouvelle requete)

git checkout (nouvelle branche)

----On peut commencer à travailler-----

Git status

Git add \*.extension

Git status

Git commit -m "Message significatif"

Git pull origin VOTRENOM

Git push origin VOTRENOM

**Important de faire une branche à votre nom!!**

**NE PAS PUSH DANS LE MASTER**