TP 1 : Commandes et opérations sur GitHub

L'objectif de ce TP est de suivre l'évolution d'un projet ou plus simplement d'un programme, localement et à distance, même lorsque l'on collabore à plusieurs, prendre le risque de modifier ou supprimer le travail d'un collaborateur. Durant ce TP nous allons utiliser **GitHub** comme un outil de travail collaboratif**.**

1. Introduction

Git est un logiciel de gestion de versions décentralisé. C'est un logiciel libre créé par Linux et distribué selon les termes de la licence publique générale GNU version 2. En 2016, il s'agit du logiciel de gestion de versions le plus populaire qui était utilisé par des millions de personnes.

Le principe de l'utilisation de git repose sur le dépôt d'une version d'un projet stockée sur un serveur sur Internet. Les modifications faites sur votre ordinateur sont enregistrées sur un espace distant par git à travers la commande (commit). Git offre aussi la possibilité de partager vos travaux enregistrés sur le dépôt distant avec votre équipe à travers la commande (push). Aussi, vous pouvez récupérer le travail fait par vos co-équipiers enregistré sur le dépôt distant à travers la commande (pull).

2. Principe de fonctionnement

Le schéma suivant présente le principe de fonctionnement d'un système de gestion de versions de type **git** (VCS : Version Control System).

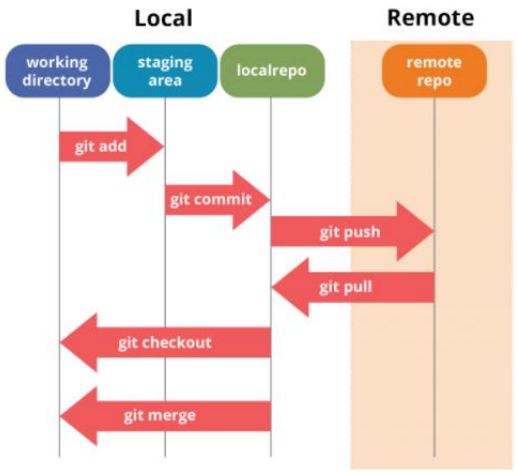


Figure 1. Fonctionnement d'un système de gestion de versions

2.1. Le stockage sur Git

Au début les fichiers de votre projet sont enregistrés sur votre ordinateur dans un dossier local appelé ***Working Directory***. Les modifications faites dans ce répertoire sont enregistrées dans un espace temporaire appelé ***Staging area*** qui contient les modifications prêtes à être « commitées » avant de les enregistrer dans le dépôt local appelé ***Local Repository***. Enfin, toutes les modifications sont enregistrées et synchronisées avec la version ancienne dans un dépôt distant appelé ***Remote Repository***.

2.2 Les opérations de bases de Git

Clone: *git clone <url_remote_repository>* permet de copier le contenu du dépôt distant dans le dossier courant. Cette commande est utilisée lorsqu'on n'a pas encore de dépôt local du projet.

Initialize: *git init* permet de créer un dépôt local git vide ou réinitialiser un existant.

Add: *git add <file>* permet de mettre à jour le contenu de l'arborescence de travail avant l'exécution de la commande commit.

Commit: *git commit -a -m <message>* permet d'enregistrer les modifications effectuées sur un projet dans un dépôt local.

Push: git push <remote-name> <branch-name> permet de partager votre projet dans un dépôt distant.

Pull: *git pull origin <branch-name>* Cette commande met à jour le dépôt local à partir du dépôt distant.

3. Github

GitHub est un service web d'hébergement et de gestion de développement de logiciels, qui se base sur le logiciel de gestion de versions Git. Ce service est développé en Ruby on Rails et Erlang par Chris Wanstrath, PJ Hyett et Tom Preston-Werner. GitHub propose des comptes professionnels payants, ainsi que des comptes gratuits pour les projets de logiciels libres. Le site assure également un contrôle d'accès et des fonctionnalités destinées à la collaboration comme le suivi des bugs, les demandes de fonctionnalités et la gestion de tâches.

Manipulation :

Dans ce TP, vous allez installer git sur votre ordinateur, créer un dépôt git distant sur GitHub et apprendre à utiliser ses fonctionnalités.

Après avoir installé Git sur votre machine, l'étape suivante consiste à créer un compte GitHub gratuit.

1. Visitez la page officielle de création de compte : Rejoindre GitHub
2. Choisissez un nom d'utilisateur, saisissez votre adresse électronique et choisissez un mot de passe.
3. Acceptez ou refusez de recevoir des mises à jour et des annonces en cochant/décochant la case Préférences de messagerie.
4. Vérifiez que vous n'êtes pas un robot en résolvant le puzzle Captcha.
5. Cliquez sur Créer un compte.
6. GitHub envoie un code de lancement à l'adresse électronique spécifiée. Copiez-collez le code dans le champ prévu à cet effet.

Vous avez maintenant créé avec succès un compte GitHub.

Partie I: Utilisation de l'interface de « Visual Studio Code »

Pour faciliter l'utilisation de git nous allons utiliser Visual Studio Code (VS Code).

VS Code est l'un des éditeurs les plus populaires dans le domaine du développement Web. Il a acquis une telle popularité grâce à ses nombreuses fonctionnalités intégrées telles que l'intégration du contrôle de version avec Git. L'exploitation de la puissance de Git à partir de VS Code peut rendre votre flux de travail plus efficace et plus robuste.

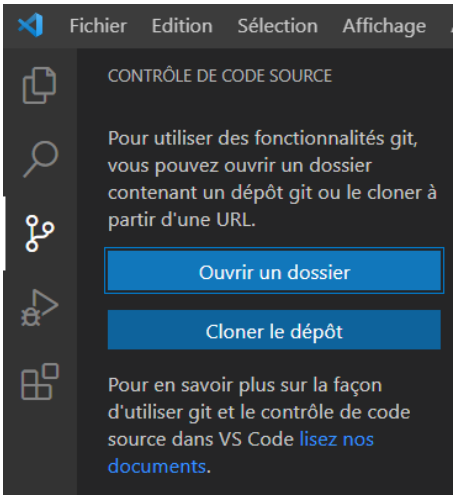
Pour ce faire, vous devez installer Visual Studio Code sur votre machine.

Étape 1: Initialisation du projet en tant que référentiel Git

Afin d'initialiser votre projet, vous devez suivre ces étapes :

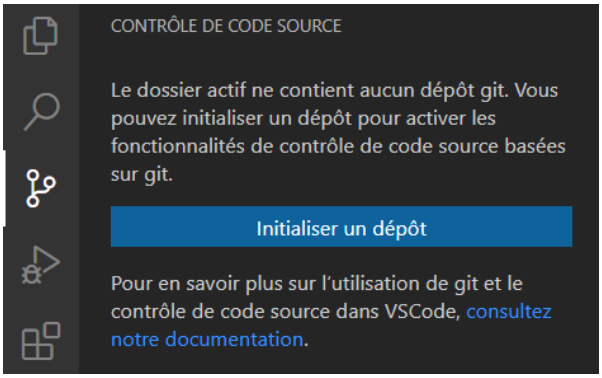
1. Ouvrir l'onglet **Contrôle de Code Source** (dans le panneau de gauche de VS Code) :

2. Cliquez sur **Ouvrir un dossier**



3. L'explorateur de fichiers du répertoire actuel est ouvert. Choisissez votre répertoire de projets, puis cliquez sur **Ouvrir**.

4. Choisir **Initialiser un dépôt**

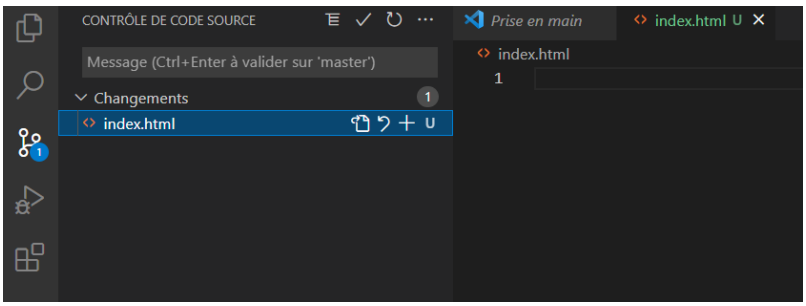


Votre système de fichiers, contient maintenant un répertoire .git.

Utilisez le terminal afin d'explorer le répertoire du projet.

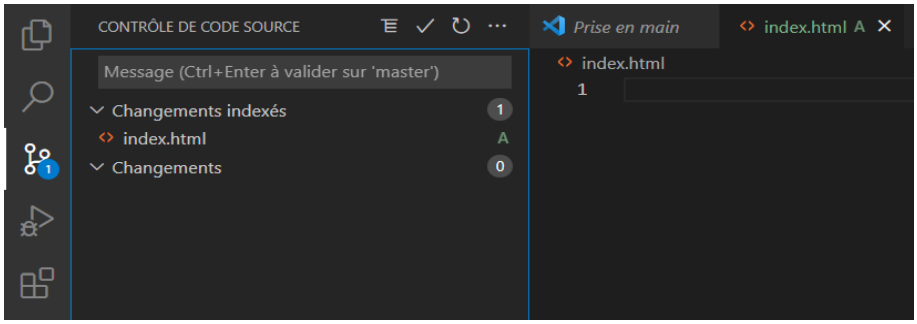
5. Créer le fichier « index.html ».

Dans le panneau **Contrôle de Code Source,** ce fichier apparaît avec la lettre **U** (*untracked file)*. Cela veut dire que ce fichier est nouveau, modifié ou non traqué, donc n'a pas été ajouté au référentiel.



➔ En cliquant sur l'icône **plus** (+), la lettre à côté du fichier se transforme en **A**.

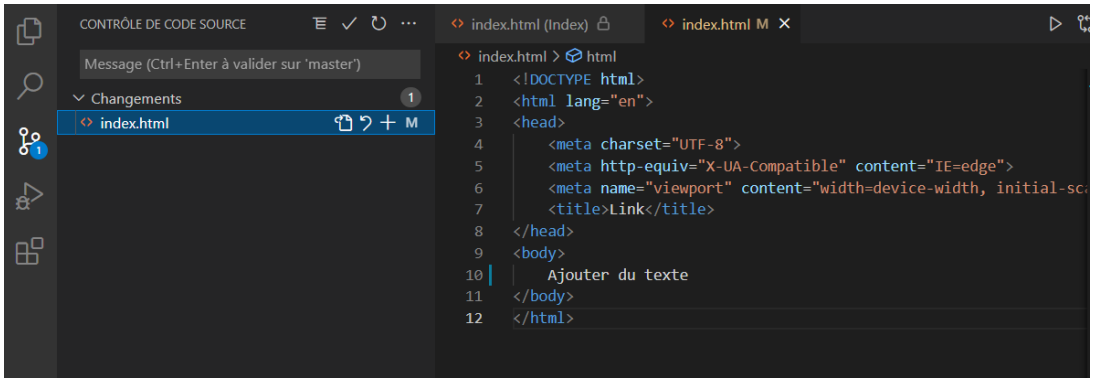
Donc, ce fichier qui a été bien ajouté au référentiel.



6. Pour la validation des modifications, il faut taper un message de commit en cliquant sur l'icône **check** dans la zone de saisie située en haut du panneau **Contrôle de Code Source**.

Étape 2: Modification du projet

1. Ajoutez du contenu au fichier index.html.



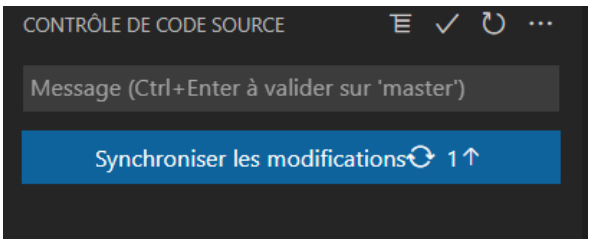
2. Cliquer sur l'icône **plus** (+), vous verrez que votre fichier a été modifié.

   La lettre **M** apparaîtra à côté représentant un dossier qui a été *modifié*.

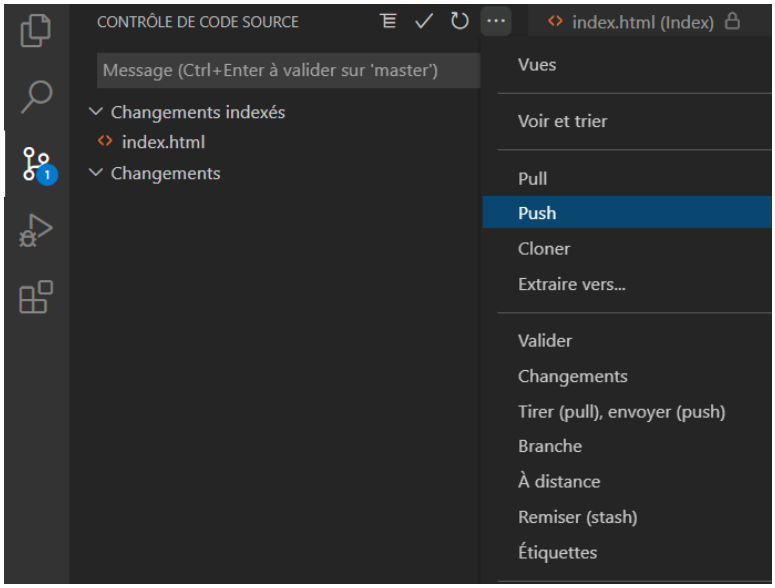
3. Afin de valider ces changements, cliquer sur l'icône **check** (pour créer un commit).

   Un bouton pour synchroniser les modifications apparaitra.

   En cliquant sur ce bouton, ces modifications seront enregistrées.



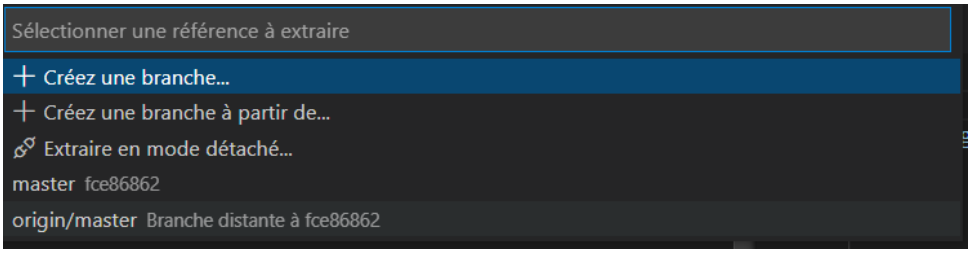
4. Une autre façon pour enregistrer ces modifications dans le dossier distant consiste à choisir la commande « Push » se trouvant dans le panneau **Contrôle de Code Source**



Étape 3 - Création des branches

1. Afin de créer et de changer de branche, cliquez sur l'icône de **contrôle de source** (celle qui ressemble à une fissure dans la route) suivie très probablement par master ou le nom de la branche en cours de fonctionnement.

2. Pour créer une branche, cliquez sur le nom de cette branche. Un menu devrait s'afficher pour vous permettre de créer une nouvelle branche :



3. Créez une nouvelle branche appelée test.

4. Modifiez votre fichier index.html qui signifie que vous êtes dans la nouvelle branche *Test*, comme l'ajout du texte *Une nouvelle branche*.

5. Validez ces changements à la branche *Test*. Ensuite, cliquez à nouveau sur le nom de la branche en bas à gauche pour revenir à la branche master.

6. Après être retourné(e) à la branche master, vous remarquerez que le texte *Une nouvelle branche* appliqué à la branche Test n'est plus présent.

Méthode 2: Utilisation des commandes Git:

Étape1: Créer un dépôt Git local

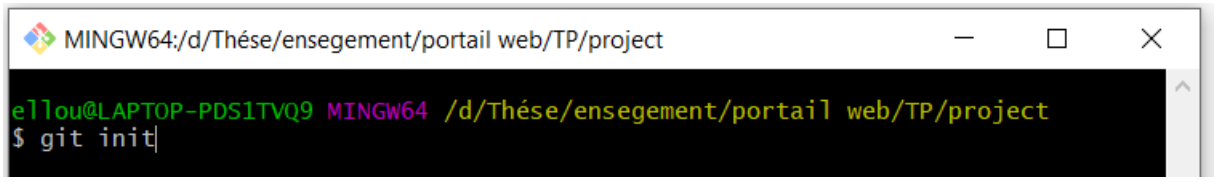
Après avoir installé ou mis à jour Git, l'étape suivante consiste à créer un dépôt Git local. Pour créer un dépôt Git, suivez les étapes ci-dessous :

1. Ouvrez un terminal Git Bash et allez dans le répertoire où vous souhaitez conserver le projet sur votre machine locale. Par exemple :

2. Créez un dépôt Git dans le dossier sélectionné en exécutant la commande git init. La syntaxe est la suivante :

```
Boško@Bosko MINGW64 ~/Desktop/myproject
$ git init
Initialized empty Git repository in C:/Users/Snesko/Desktop/myproject/.git/

Boško@Bosko MINGW64 ~/Desktop/myproject (master)
$
```



3. Configuration du compte Github et l'espace de travail local sur votre ordinateur.

```
ellou@LAPTOP-PDS1TVQ9 MINGW64 /d/Thése/ensegement/portail web/TP/project (master
)
$ git config --global user.name "Nessrinetesting"
```

```
ellou@LAPTOP-PDS1TVQ9 MINGW64 /d/Thése/ensegement/portail web/TP/project (master
)
$ git config --global user.email nessrinetesting@gmail.com
```

Vous avez maintenant créé avec succès un dépôt Git local.

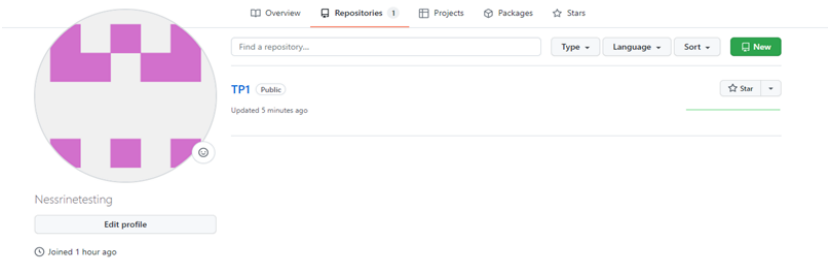
Étape 2: créer un nouveau dépôt sur GitHub

GitHub vous permet de garder la trace de votre code lorsque vous travaillez avec une équipe et que vous devez modifier le code du projet de manière collaborative.

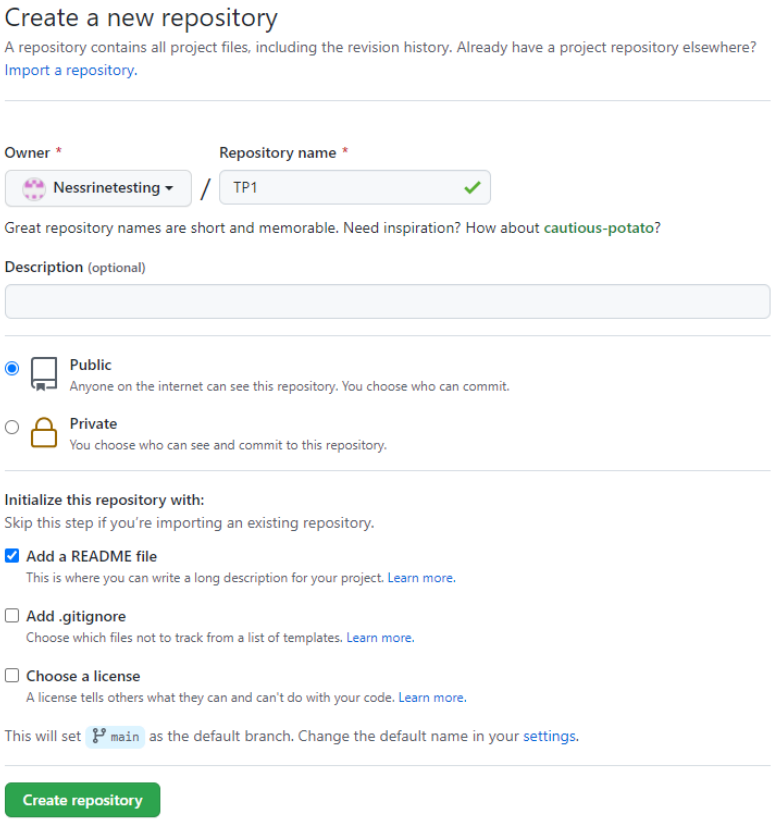
Suivez ces étapes pour créer un nouveau dépôt sur GitHub :

1. Connectez-vous et accédez à la page d'accueil de GitHub.

2. Trouvez l'option Nouveau dépôt sous le signe + à côté de votre photo de profil, dans le coin supérieur droit.



3. Saisissez un nom pour votre référentiel appelée TP1, fournissez une description et choisissez un paramètre de confidentialité.



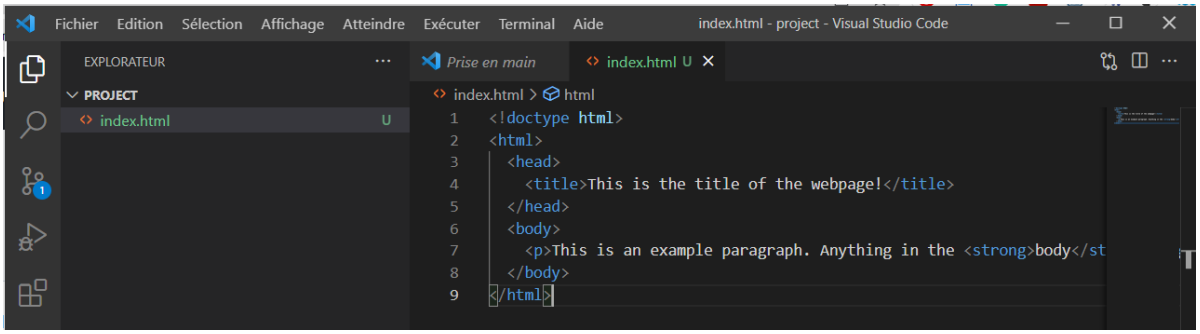
4. Cliquez sur le bouton Créer un référentiel.

GitHub vous permet d'ajouter un dépôt existant que vous avez créé localement. Pour pousser un dépôt local de votre machine vers GitHub, utilisez la syntaxe suivante :

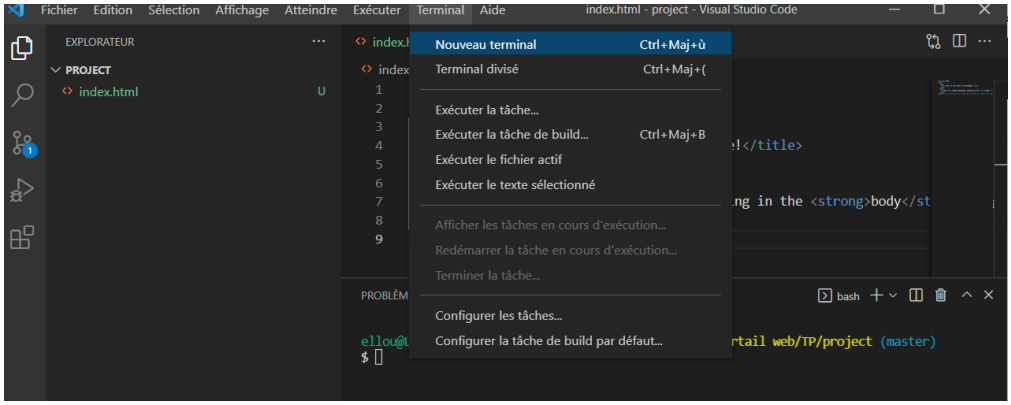
```
ellou@LAPTOP-PDS1TVQ9 MINGW64 /d/Thése/ensegement/portail web/TP/project (master
)
$ git remote add origin https://github.com/Nessrinetesting/TP1.git
```

Étape 3: Ajouter un fichier au référentiel

1. Ouvrir VS Code et créer un nouveau fichier appelé index.html



2. Ouvrir un nouveau terminal sous VS Code



3. Ajouter le fichier index.html en utilisant la commande git commit.

Vous pouvez vérifier les fichiers suivis par Git en exécutant :

```
ellou@LAPTOP-PDS1TVQ9 MINGW64 /d/Thése/ensegement/portail web/TP/project (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        index.html

nothing added to commit but untracked files present (use "git add" to track)
```

Étape 4: Créer un commit

Après avoir ajouté les fichiers spécifiés à l'environnement de transit, demandez à Git de regrouper les fichiers dans un commit en utilisant la commande git commit. Git stocke alors cette version du fichier. Vous pouvez revoir la version stockée à tout moment.

La syntaxe est la suivante :

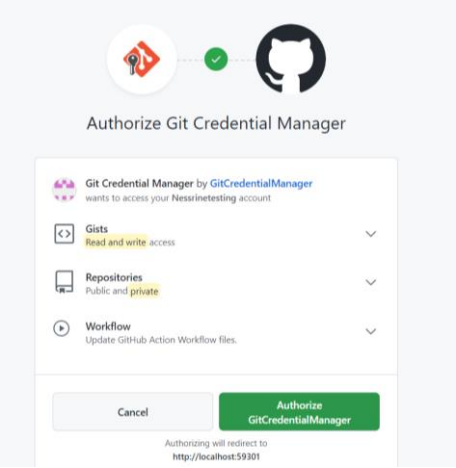
```
ellou@LAPTOP-PDS1TVQ9 MINGW64 /d/Thése/ensegement/portail web/TP/project (master)
$ git commit -m "version 1"
[master (root-commit) b9e7b59] version 1
 1 file changed, 9 insertions(+)
 create mode 100644 index.html
```

Étape 5: Ajouter les paramètres d'accès au GitHub sous VS Code

1. Utiliser la commande **git config --global credential.helper store** pour enregistrer les paramètres de connexion sous VS Code.

```
ellou@LAPTOP-PDS1TVQ9 MINGW64 /d/Thése/ensegement/portail web/TP/project (
master)
$ git config --global credential.helper store
```

2. Cliquer sur Autoriser.

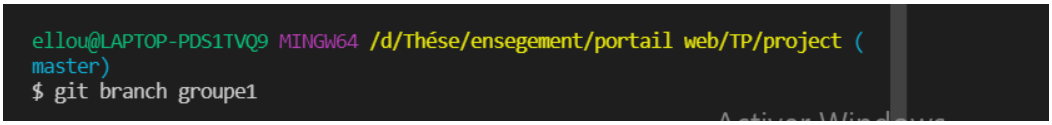


3. Enregistrer les modifications dans le dossier TP1 distant (remote) dans la branche master en utilisant la commande **git push origin master.**

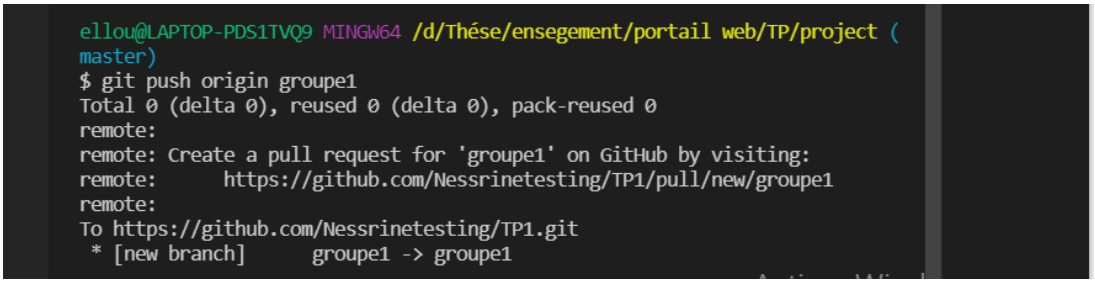
```
ellou@LAPTOP-PDS1TVQ9 MINGW64 /d/Thése/ensegement/portail web/TP/project (
master)
$ git push origin master
info: please complete authentication in your browser...
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 382 bytes | 382.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/Nessrinetesting/TP1/pull/new/master
remote:
To https://github.com/Nessrinetesting/TP1.git
 * [new branch]      master -> master
```

Étape 6: Ajouter une nouvelle branche « groupe1 »

1. Créer une nouvelle branche appelée groupe1



2. Enregistrer les modifications sous la branche groupe1



Étape 7: Invitation des collaborateurs sur le projet TP1

Cliquer sur le répertoire **TP1** et sélectionnez l'onglet **Settings** puis **collaborators** pour ajouter des Coéditeurs sur ce projet.