# Atelier : Mise en place des outils de développement WEB

**Objectifs :** Nous allons mettre en oeuvre différents outils de développement WEB :

* GIT : pour sécuriser son travail et collaborer
* PhpStorm : un des meilleurs IDE du marché (et européen !)

**Travail à rendre :**

* Il y a des tâches, des questions ou des travaux à faire tout le long de ce support :il faudra les traiter !
* Vous y répondrez dans un fichier type “word” envoyé en format modifiable (pas de pdf) par mail : jbaubry25@gmail.com
* Le document sera structuré (garde, sommaire)
* Forme :
  + Les questions seront rappelées,
  + les réponses rédigées,
  + les captures commentées.
* Réalisation au fur et à mesure de la lecture de ce document.
* Date limite : mardi 18 Octobre 18h00

# Introduction à GIT

Nous allons étudier son fonctionnement et voir ce qu’il apporte comme fonctionnalités.

# I°) Quelques définitions

* Qu'est-ce qu’un gestionnaire de versions ?

Un gestionnaire de versions est un programme qui permet aux développeurs de conserver un historique des modifications et des versions de tous leurs fichiers.

* versioning : L’action de contrôler les versions
* outils de versionning :

Ce type d’outil a différentes grandes fonctionnalités :

1. Revenir à une version précédente de votre code en cas de problème.
2. Suivre l’évolution de votre code étape par étape.
3. Travailler à plusieurs sans risquer de supprimer les modifications des autres collaborateurs.

Git est de loin le système de contrôle de versions le plus largement utilisé aujourd'hui.

## II°) Intérêts

* **En travaillant seul** : garder l’historique des modifications et pouvoir revenir à une version précédente facilement.
* **En travaillant en équipe**: gérer les versions et fusionner les modifications des différents acteurs travaillant simultanément sur un même fichier / projet.

## III°) Git et GitHub

Git et GitHub sont deux choses différentes.

* **Git** : gestionnaire de versions (logiciel installé localement et logiciel serveur de dépôts distants). Vous l'utilisez pour créer un dépôt local et gérer les versions de vos fichiers.
* **GitHub** : service en ligne qui va héberger votre dépôt. Dans ce cas, on parle de dépôt distant puisqu’il n’est pas stocké sur votre machine.

# IV°) Différents types de dépôts

## A°) Dépôts ?

Un dépôt est comme un dossier qui conserve un historique des versions et des modifications d’un projet. Il peut être local ou distant.

Dans la documentation en ligne ou en milieu professionnel, on parle souvent de “*repository”*.

## B°) Dépôt local

* Un entrepôt virtuel d’un projet.
* Permet d'enregistrer les versions de votre code et d'y accéder au besoin. Endroit où l’on stocke, sur sa machine, une copie d’un projet, ses différentes versions et l’historique des modifications.
* Un dépôt local se manipule avec des commandes dédiées à sa manipulation

## C°) Le dépôt distant

* Différent de dépôt local.
* Stocke les différentes versions de votre code afin de garder un historique délocalisé, c'est-à-dire un historique hébergé sur Internet ou sur un réseau. Associé à un compte sur un service de versionning
* Version dématérialisée du dépôt local, que ce soit sur Internet ou sur un réseau. Il permet de centraliser le travail des développeurs dans un projet collectif.

Les interactions avec un dépôt distant se font avec d’autres commandes que pour un dépôt local.

# 

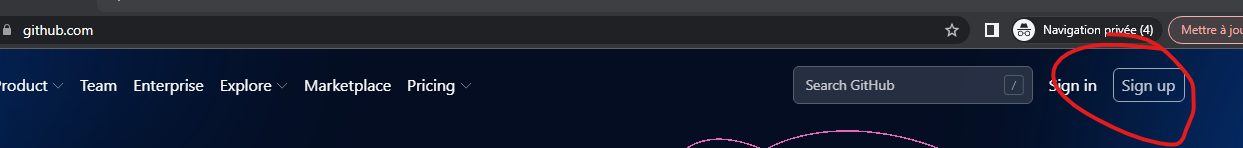
# ETAPE N°1 : Premier dépôt

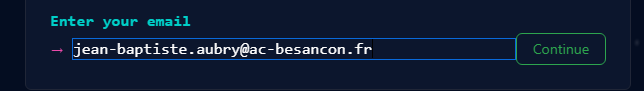
# I°) Créer un nouveau compte sur Github

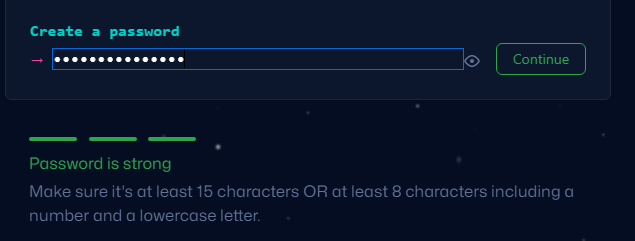
Pour ce compte l’adresse mail a peu d’importance. Je vous conseille cependant de prendre une adresse mail qui survivra à votre passage à Pergaud !

(sauf si vous en avez déjà un !)

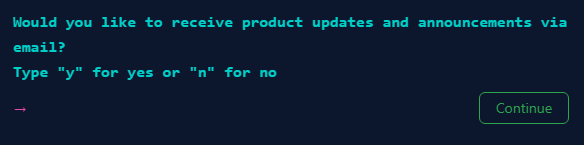
URL : Github.com

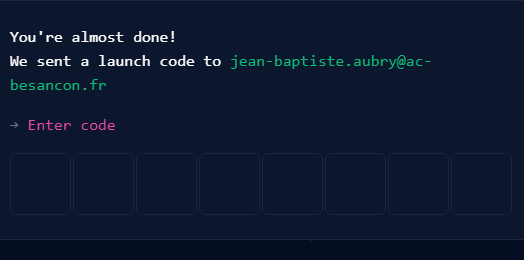


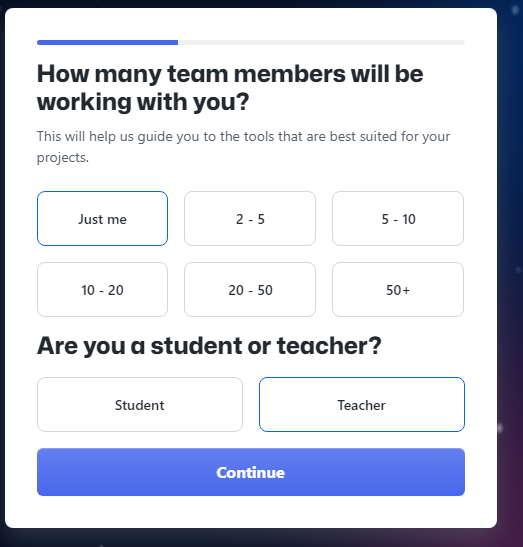




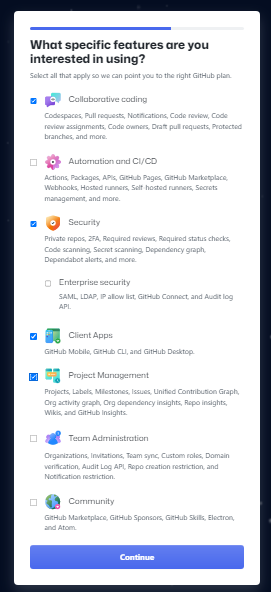






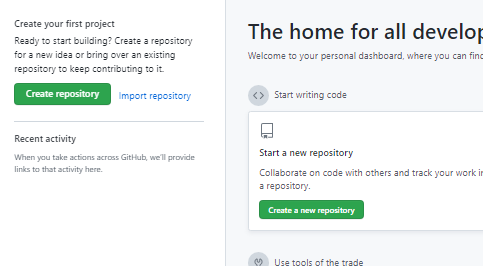


Cocher ce qui va nous intéresser :

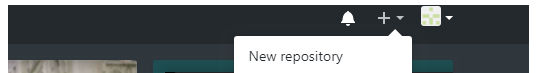


(vous ce sera student !)

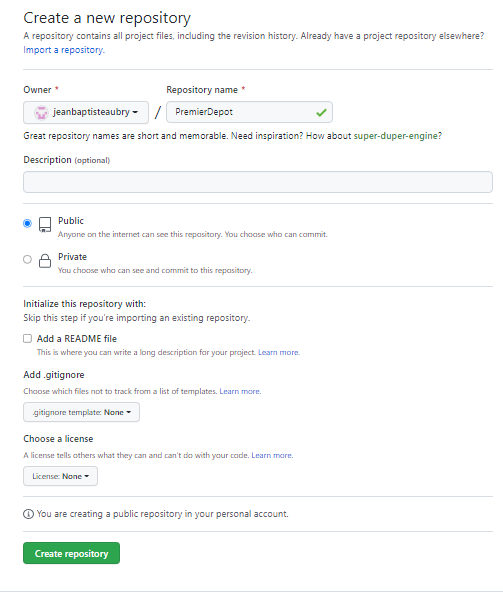
# II°) Création d’un dépôt distant



ou



Vous appellerez ce dépôt : PremierDepot



##### **Questions :**

1. Quelles seront les commandes GIT pour créer un dépôt local en le reliant à ce dépôt distant ?  
echo "# premierdepot" >> README.md

git init

git add README.md

git commit -m "first commit"

git branch -M main

git remote add origin https://github.com/titoulz/premierdepot.git

git push -u origin main

2. Quelles seront les commandes GIT pour relier un dépôt local existant à ce dépôt distant ?

git remote add origin https://github.com/titoulz/premierdepot.git

git branch -M main

git push -u origin main

# III°) Le dépôt local

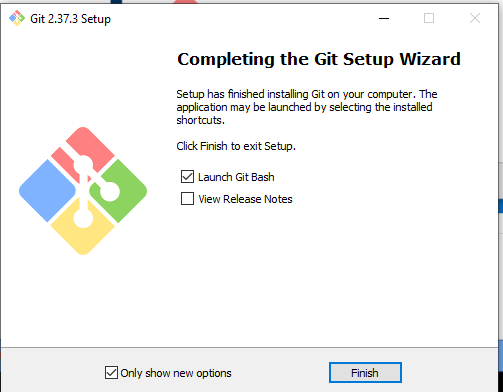
## A°) Prérequis : avoir GIT d’installer ! (Déjà fait au lycée)

* Télécharger git : <https://git-scm.com/downloads>

Pour windows : <https://github.com/git-for-windows/git/releases/download/v2.37.3.windows.1/Git-2.37.3-64-bit.exe>

Nous ferons une installation avec tous les paramètres par défaut. Il est important d’avoir la dernière version car des paramètres ou des fonctionnalités peuvent évoluer et devenir bloquant pour le couplage avec github.

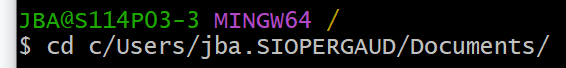
A la fin de l’installation, vous lancerez le gitbash :



## B°) Initialiser GIT.

Pourquoi ? Git a des paramètres globaux et d’autres liés à chaque dépôt. Nous allons commencer par les paramètres génériques (ils peuvent être surchargés par projet).

Quelques commandes utiles pour git-bash : cd

* cd (change directory)
  + cd c:
    - Aller sur le lecteur c:^
  + cd user/<utilisateur AD>/Documents
    - Aller dans le dossier documents de l’utilisateur connecté (ici JBA)
  + 
* mkdir : Make Directory
  + mkdir premierprojet
    - Créer le dossier “premierprojet” dans le dossier courant
* pwd : print working directory :
* dir ou ls : list directory

Définir l’utilisateur :

Définition du nom :

git config --global user..name "titoulz”

Définition du mail :

git config --global user.email "jean-baptiste.aubry@ac-besancon.fr"

Exemple : 



Si vous souhaitez, pour un projet spécifique, changer votre nom d’utilisateur, vous devrez repasser cette ligne mais sans le --global.

##### **Tâches (capture) :**

3. Faire les paramètres généraux !

## C°) Créer un dépôt local : deux méthodes !

Maintenant que vous avez configuré les paramètres de base, vous pouvez créer votre fameux dépôt local. Pour ce faire, deux solutions sont possibles :

* créer un dépôt local vide : pour accueillir un nouveau projet. La procédure est expliquée ci-dessous ;
* cloner un dépôt distant : importer l’historique d’un dépôt distant en local. (plus tard !)

### 1°) Initialiser un dépôt en créant un dépôt local vide

Première étape : créer un dossier sur votre ordinateur (avec le nom de notre projet, c’est mieux). CF commandes bashs

##### **Tâches (capture) :**

4. Création d’un nouveau répertoire dans “Mes documents”. Depuis gitbash :

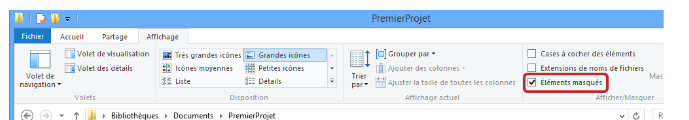
* + Aller dans le lecteur c: (cd c:)
  + Aller dans le dossier users\<<nom d’utilisateur>>\documents\
  + Créez le dossier “PremierProjet”.
  + Accédez à cd <nouveau dossier>.

La commande qui permet d’initialiser git pour travailler dans un dossier est :

git init

##### **Question :**

5. Quels sont les changements dans votre dossier PremierProjet ?

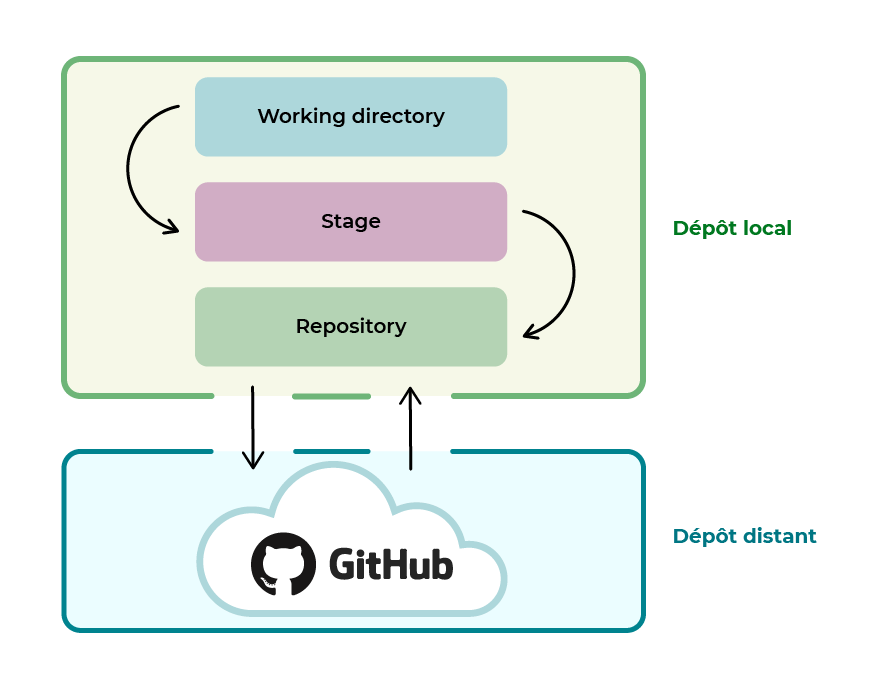
(indice)

.git a apparu en masqué

ETAPE N°2 : Fonctionnement de git

# I°) Principe

Ce schéma résume le fonctionnement de git :



## Le Working directory

Cette zone correspond au dossier du projet sur votre ordinateur.

## Le Stage ou index

Cette zone est un intermédiaire entre le working directory et le repository. Elle représente tous les fichiers modifiés que vous souhaitez voir apparaître dans votre prochaine version de code.

## Le Repository

Lorsque l’on crée de nouvelles versions d’un projet (vous vous souvenez, les 3 versions différentes du gâteau ?), c’est dans cette zone qu’elles sont stockées.

Ces 3 zones sont donc présentes dans votre ordinateur, en local.

# II°) Ajout de fichiers à l’index

##### **Tâches (capture)** :

6. Créer ces fichiers dans votre dossier :

* + fichier.html

<!DOCTYPE html>

<html>

<head>

<title></title>

<link rel="stylesheet" type="text/css" href="styles.css">

</head>

<body>

<h1>J’apprend GIT</h1>

</body>

</html>

* + styles.css

h1 {

color: red;

}

* 7. Ajouter les fichiers à l’index :

git add fichier.html styles.css

# III°) Création d’une nouvelle version

Créez une nouvelle version avec la commande git commit

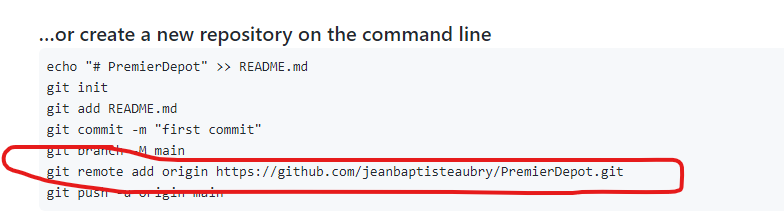
git commit -m "Ajout des fichiers html et css de base"

* m (comme message) :
  + un argument obligatoire.
  + définit un message particulier rattaché au commit effectué.
  + Il faut utiliser des guillemets droites " et non “” (faire ctrl + z après avoir appuyé sur “).

# IV°) Lier votre dépôt local à un dépôt distant et gestion des branches

## A°) Liaison à un dépôt distant

Vous trouverez la commande sur github après la page du dépôt :



git remote add origin https://github.com/jeanbaptisteaubry/PremierDepot.git

(vous adapterez en fonction du contexte )

## B°) Sélection de la branche de travail :

git branch -M main

## C°) Envoie de l’historique local vers le serveur

git push -u origin main

Là, tous les soucis peuvent arriver !

Cette commande requiert que vous soyez authentifié sur le serveur. Vous devez avoir votre navigateur principale connecté à votre session github en mode normal (pas en mode privé donc)

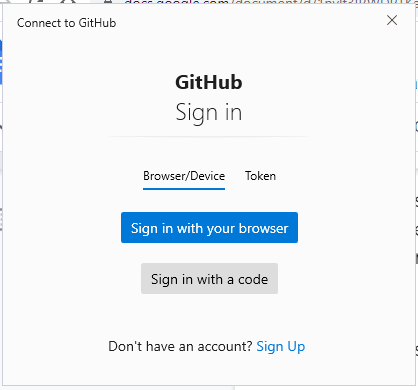
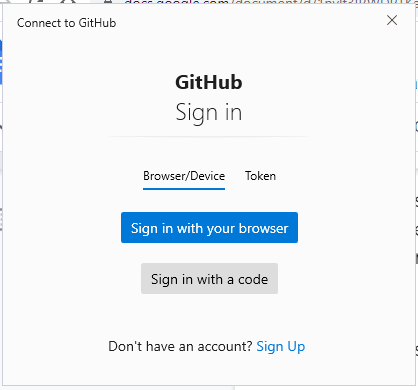
* désactiver le proxy pour git :

$ git config --global --unset https.proxy

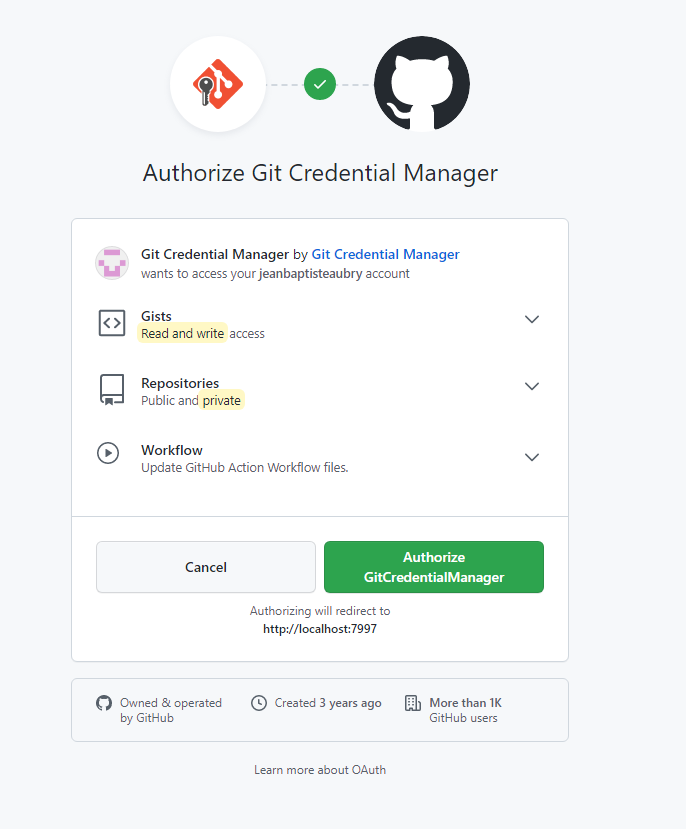
$ git config --global --unset http.proxy

$ git config --global --unset ftp.proxy

**Solution 1 : Authentification pour navigateur**

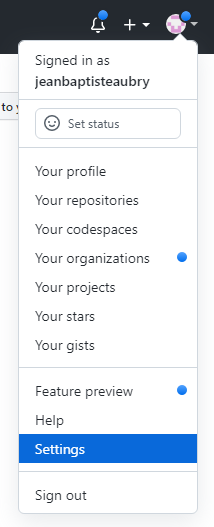
* Sélectionner Sign up with your browser :

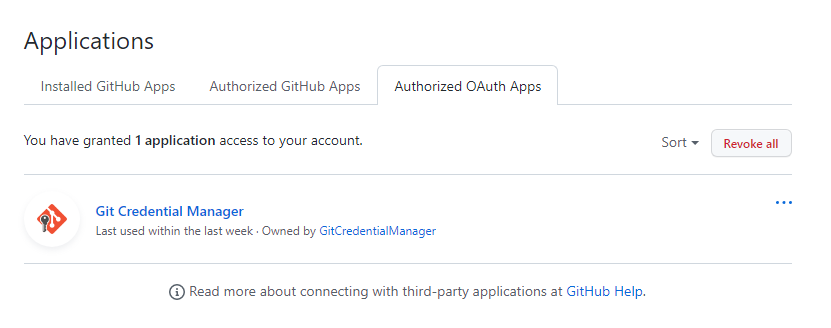
Normalement, si votre navigateur est ouvert, et que vous êtes déjà connecté à github : magie !



* Cliquer sur “Authorize”

(Pour info : Vous retrouverez l’autorisation ici :

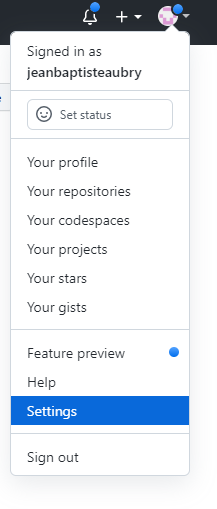


puis : )

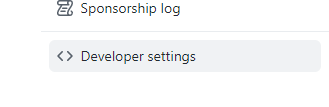
**ATTENTION !! DANS L'ÉVENTUALITÉ OÙ CETTE AUTHENTIFICATION BLOQUE, VOUS POUVEZ FAIRE UNE AUTHENTIFICATION PAR TOKEN**

**Solution 2 : Authentification pour Token**

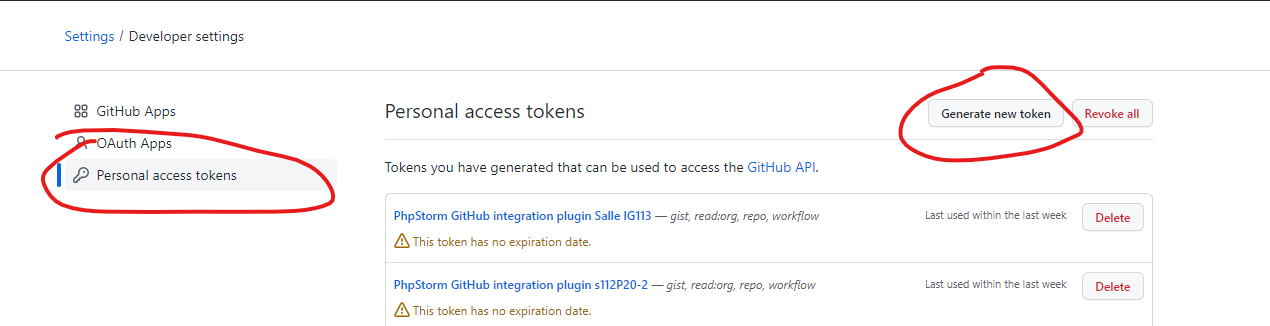
* Génération du token : (Si ça ne s’ouvre pas automatiquement)
  + Mon compte ⇒ Settings



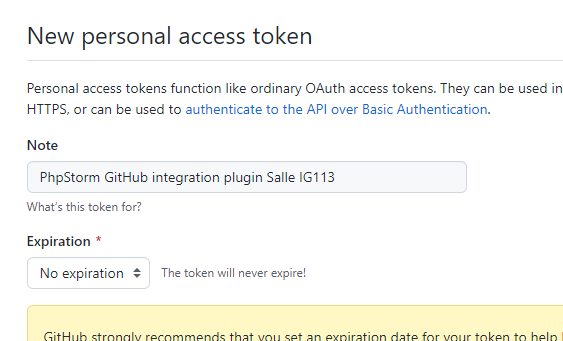
* + Developer settings (tout en bas !)



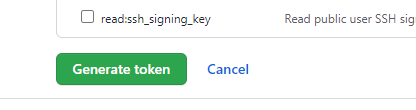
* + Puis génération du token :



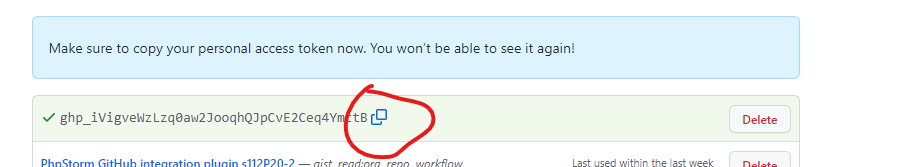
* + Donner un nom au nouveau token et une durée (on va mettre illimité, même si ce n’est pas la grande sécurité)
    - et cliquer “Générate token”



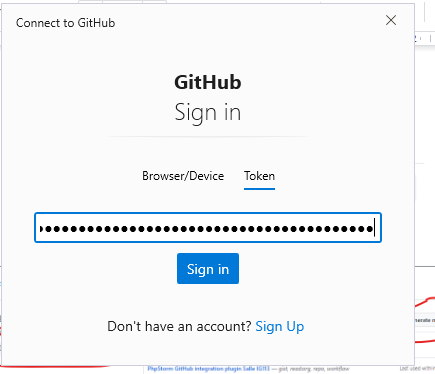
[...]



* COPIER le token



* puis le coller dans la demande d’authentification :



* Normalement, c’est bon…

##### **Question**

8. Pourquoi le proxy n’est pas nécessaire ?

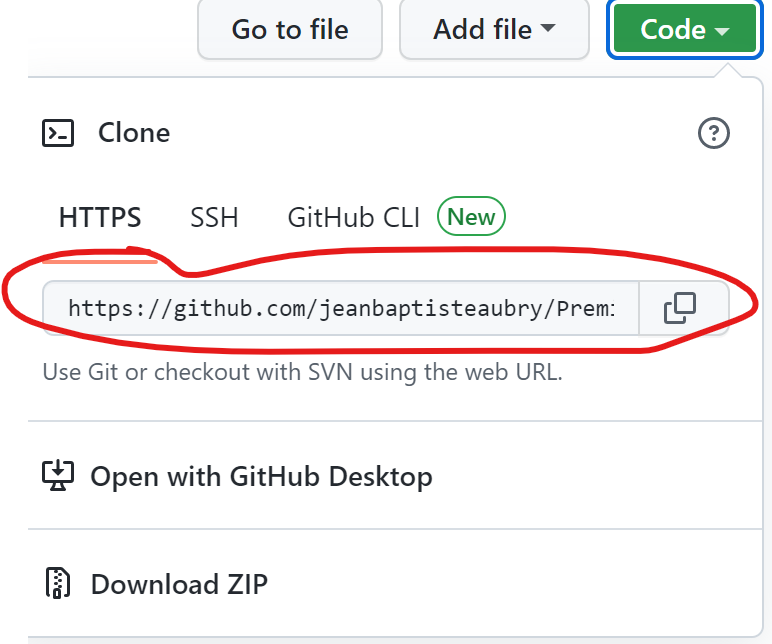
Il y a une exception au pare-feu qui autorise toutes les communications sans authentification auprès de GIT.

# V°) Travailler à distance :

## A°) Cloner un dépôt existant

git clone “URL du dossier”

Où trouver l’URL ? ici =>

on trouve cettte url dans notre repository premierdepot sur github

## B°) Localiser vers un dépôt existant:

git remote add URL

(Recommandé de faire d’abord un git push pour récupérer le repository puis add, puis commit !) Il faut que le dépôt distant soit créé.

## C°) Soumettre des modifications à un dépôt existant dont on est pas le propriétaire :

git pull request (sur le dépôt de quelqu’un d’autre)

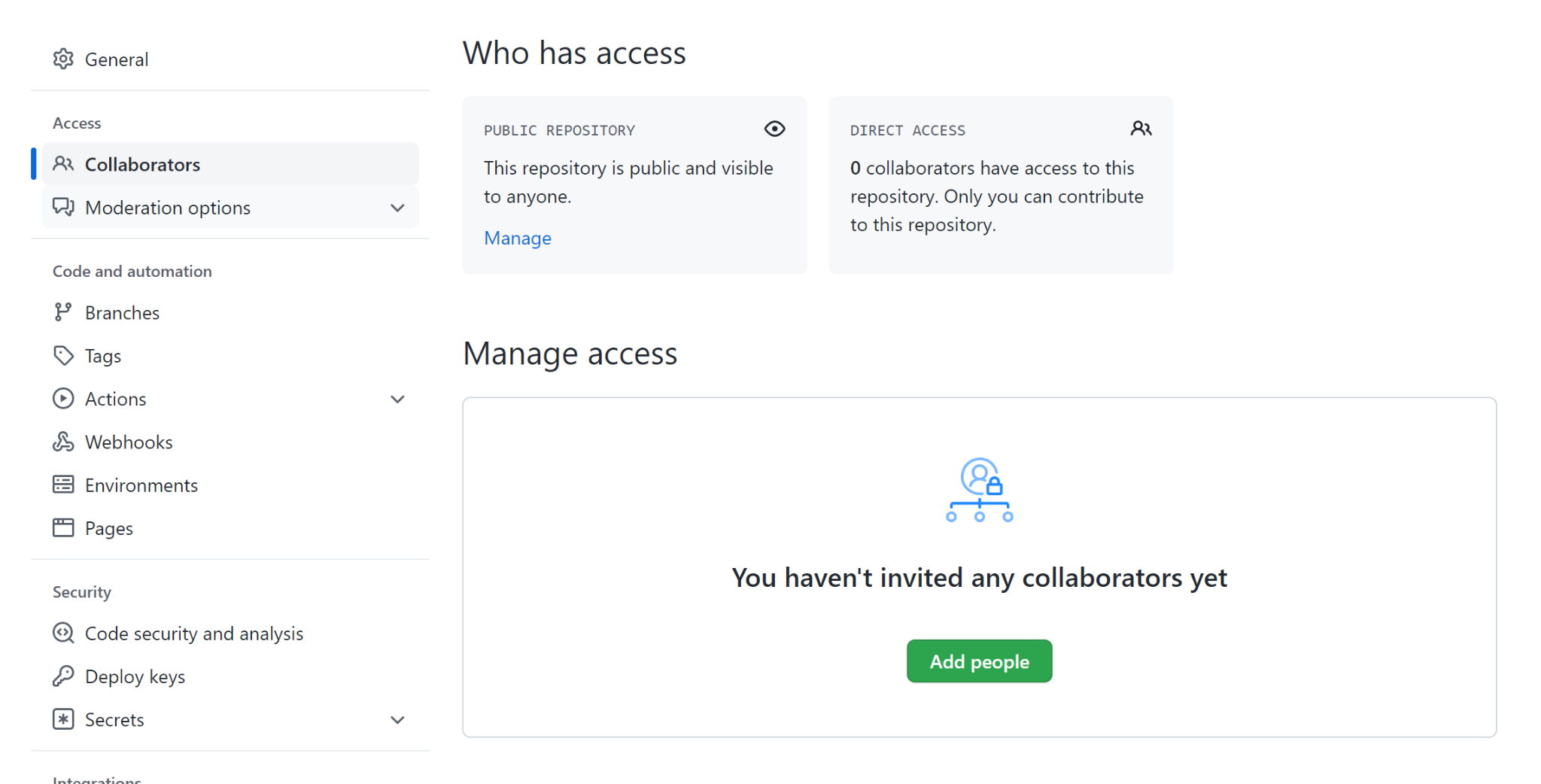
##### **Tâche (capture) :**

9. Faire un clone du projet d’un de vos camarades, faites une modification et faites un pull request.

10. Accepter le pull request

## D°) Travailler à plusieurs

Vous pouvez inviter des utilisateurs GitHub à se joindre à votre projet :



##### **Tâches et questions :**

11. Inviter votre camarade

12. Il doit faire des modifications :

* + nouveaux fichiers
  + commit
  + push

13. Que constatez-vous ?

les modifs s’applique dés que l’on pull la version la + recente que le camarade vient de deposer sur notre repositories

14. Avec votre camarade, chacun de votre côté modifiez le même fichier.

* + add, commit et push !

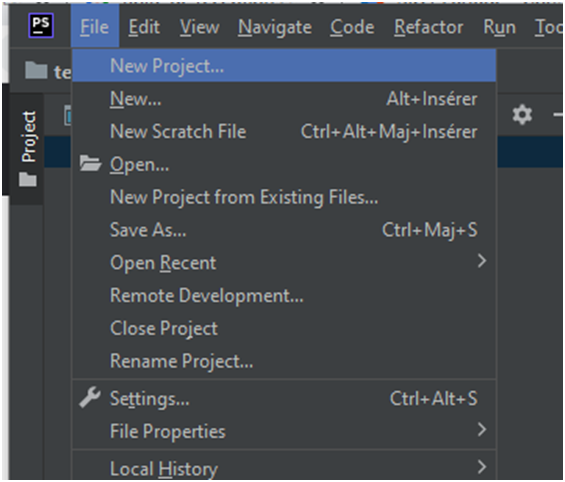
15. Trouvez une solution ! (même problème quand on merge des branches ayant modifié différemments le même fichier :) ) (indice, après “push/pull” conflictuel éditer le fichier incriminé)

# ETAPE N°4 : Git et PHPStorm

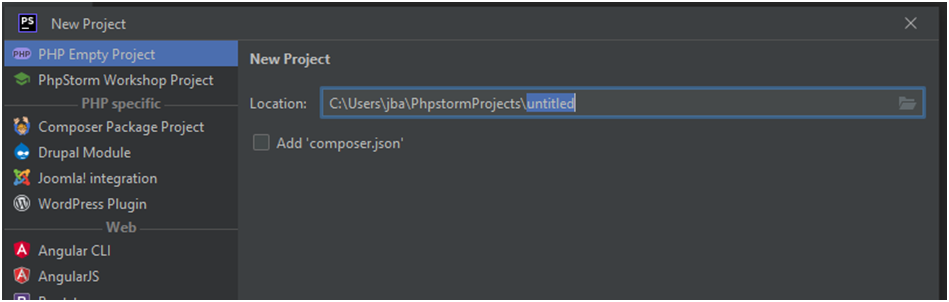
Git est couplé dans phpstorm. Les explications sur le fonctionnement de git sont à l’étape 2.

A travers cette procédure, nous allons voir comment créer un nouveau projet puis comment le coupler à GIT

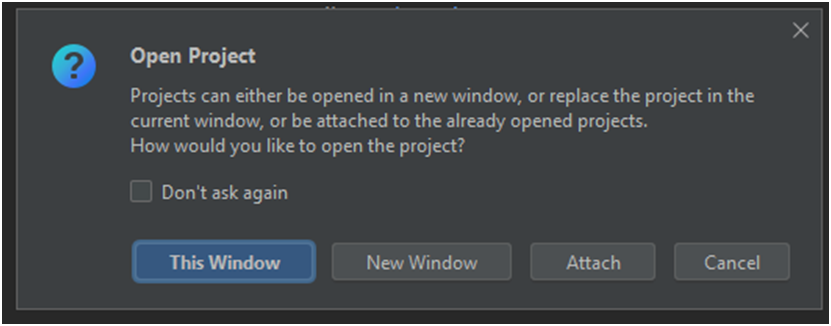
## Création d’un projet :



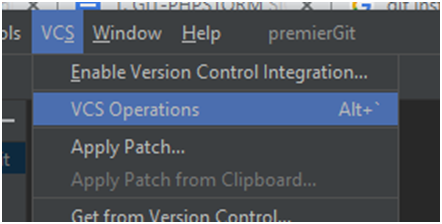
* Remplacer “untitled” par le nom du projet :

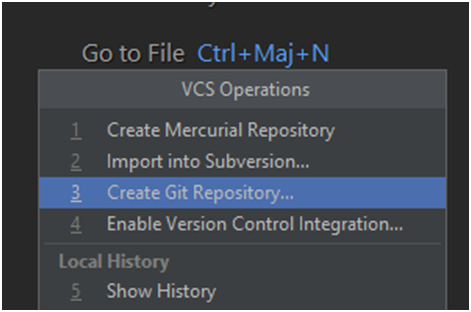


* puis choisissez où ouvrir ce nouveau projet :

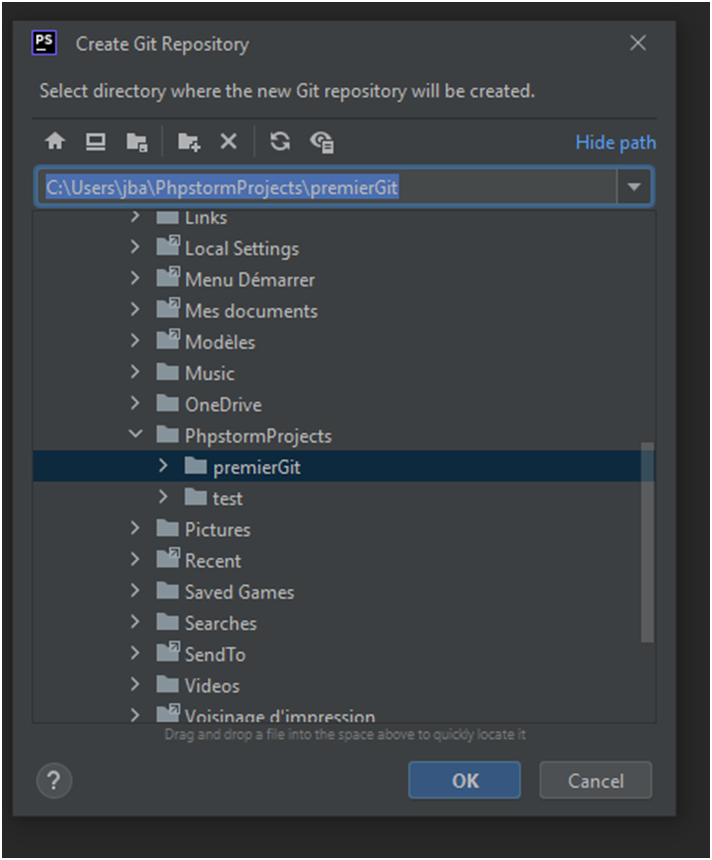


## Initialisation de git :



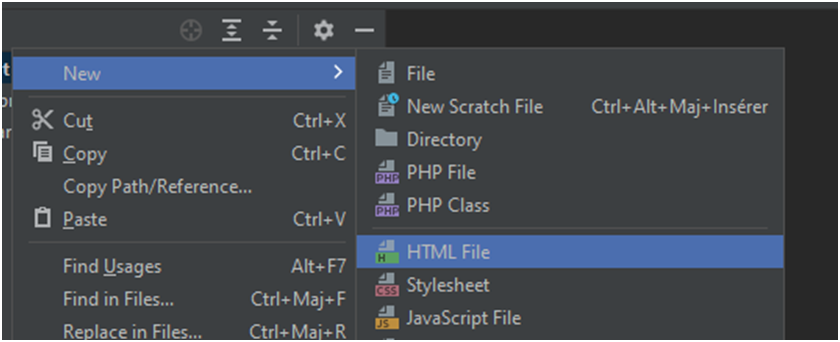


* Laisser l’emplacement par défaut :

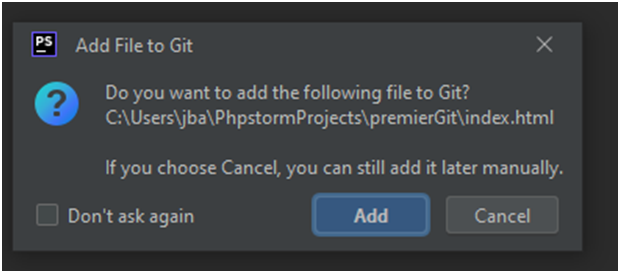


## Commit : ajout de fichier et validation :

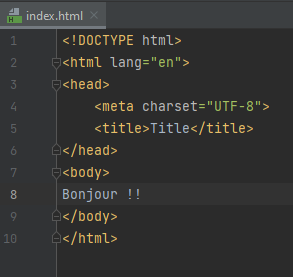
* 1. Ajout d’un fichier pour l’exemple



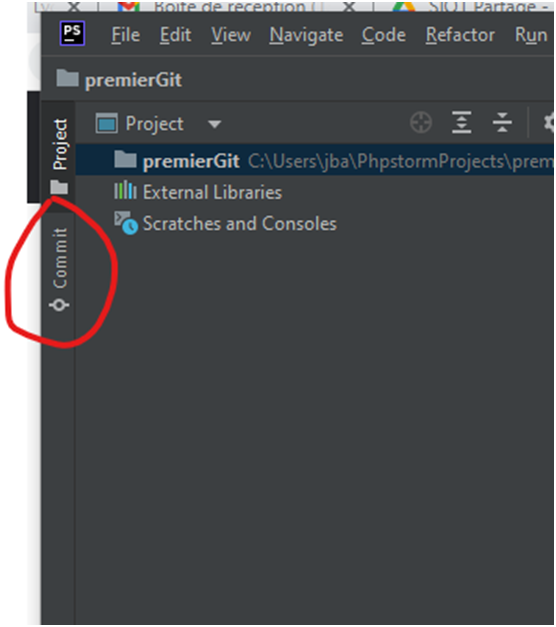
* Confirmer l’ajout du fichier à GIT :



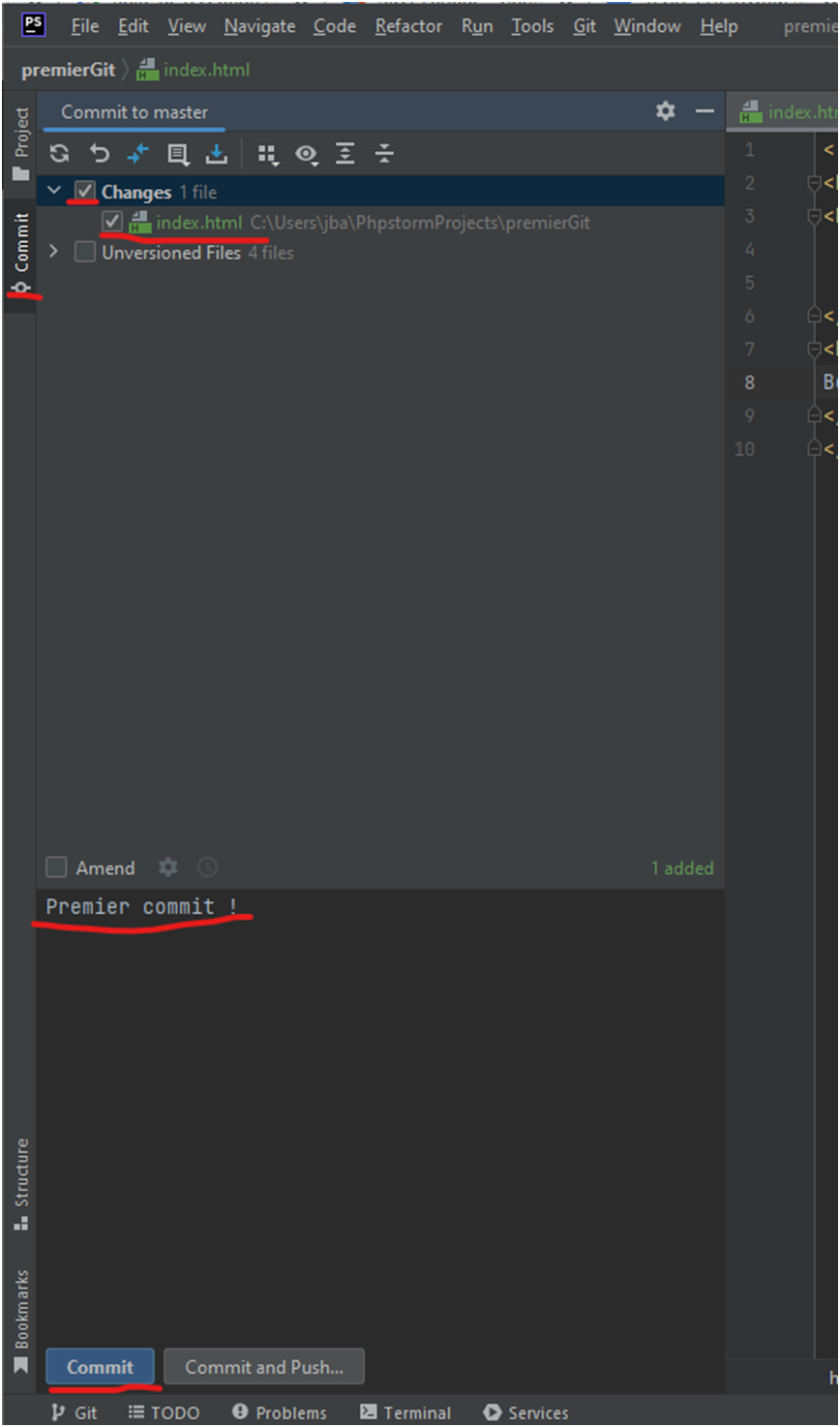
* Contenu de l’exemple :



* 1. La fonctionnalité :
     1. Le menu “commit” :

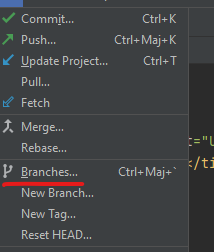


* + 1. Le “commit” : sélection des fichiers à archiver, du message et commit !

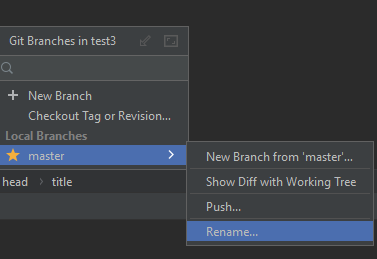


## Branche : renommer la branche “Master” en “main” :

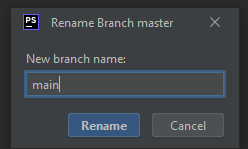
La branche principale de github est main et non master



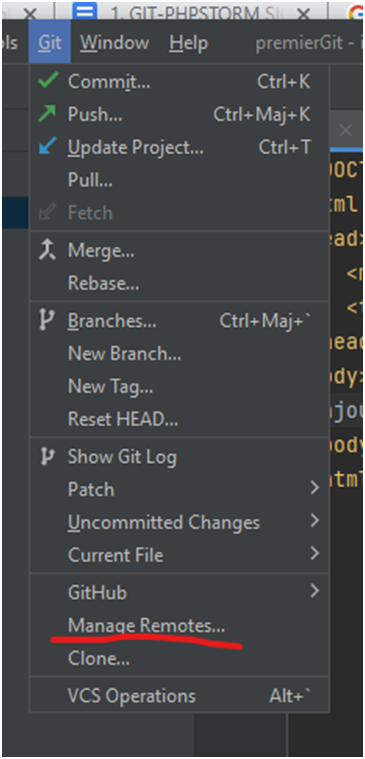
* Renommer la branche :



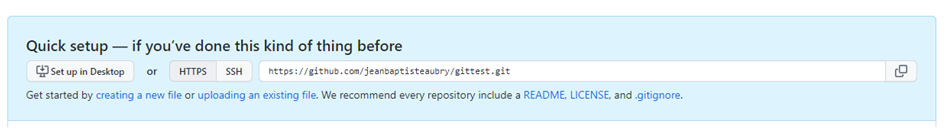
* Nommage de la branche :



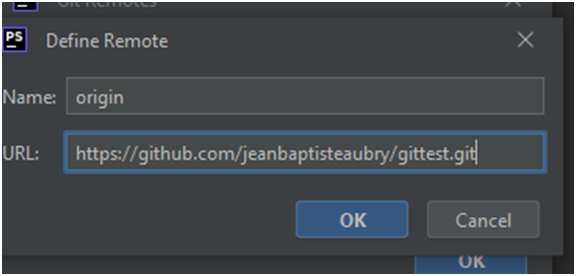
## Remote : la gestion des dépôts distants :



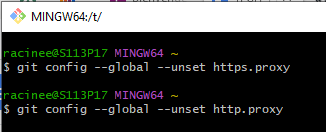
* + 1. Récupérer le lien du dépôt sous GitHub (le dépôt qui a été créé à l’étape 1):



* + 1. Copier le :



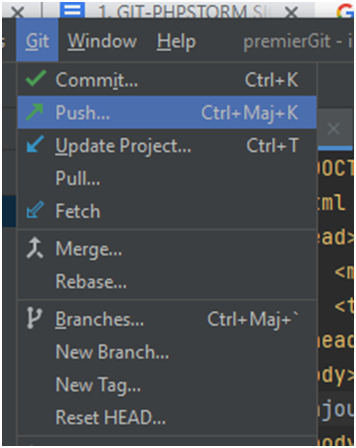
* Si vous avez le message d’erreur “407 proxy”, faire dans git bash :



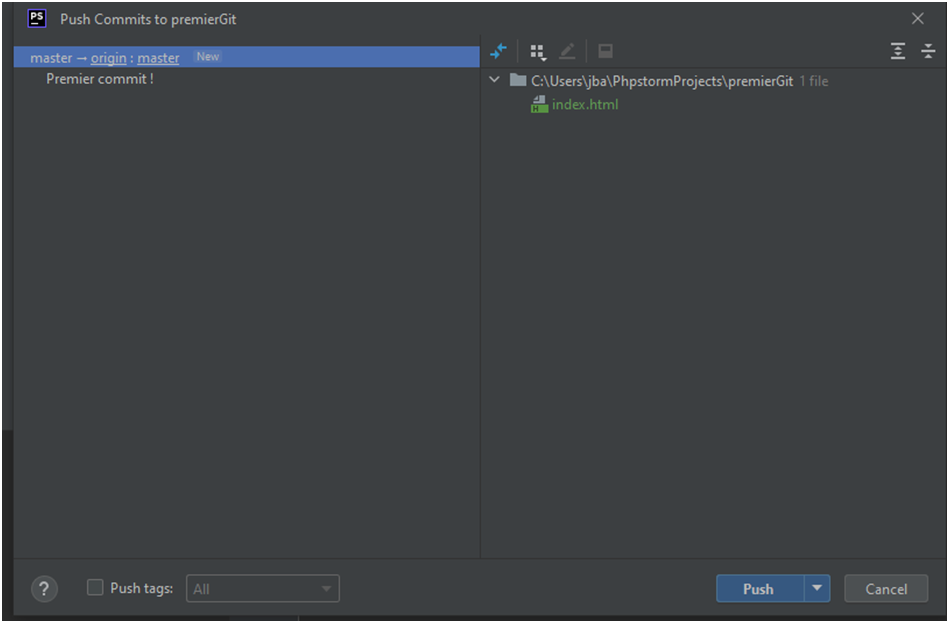
* Et réessayer !

## Push : archiver en ligne votre dépôt local

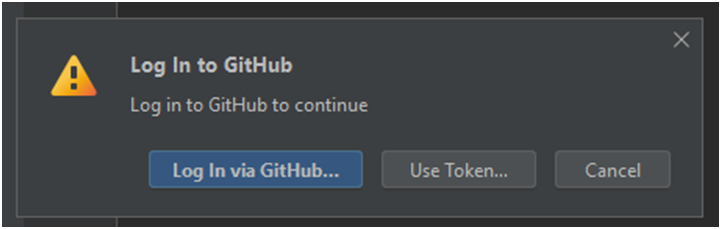
* 1. La fonctionnalité :



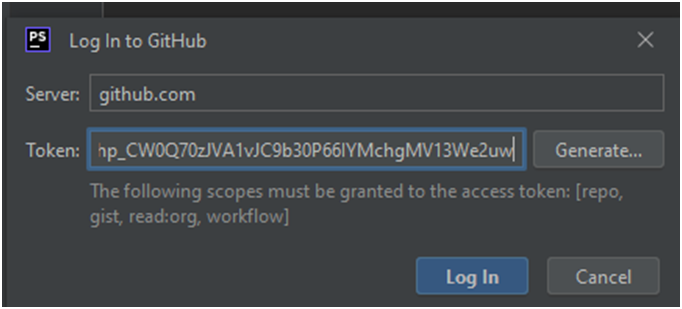
* 1. Valider le push :



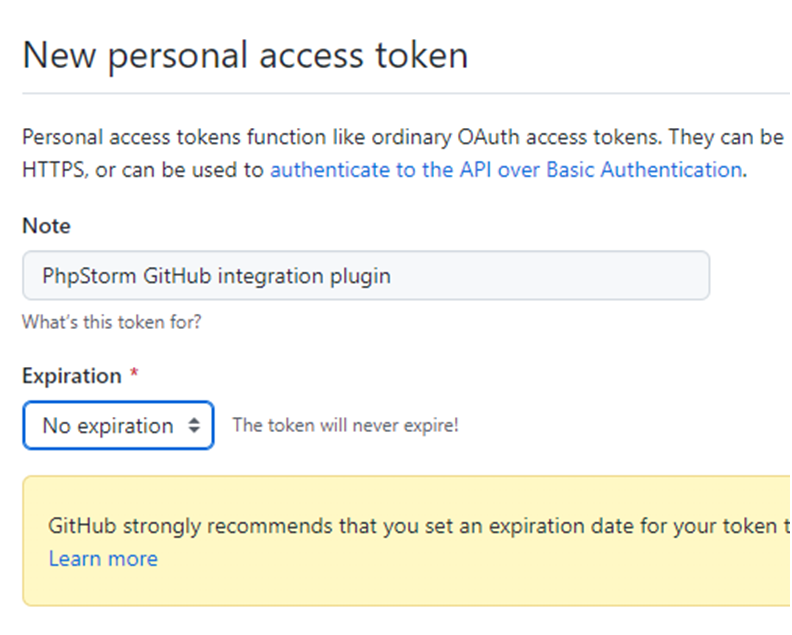
* 1. Connection à GitHub : choisir **“USE TOKEN” :**



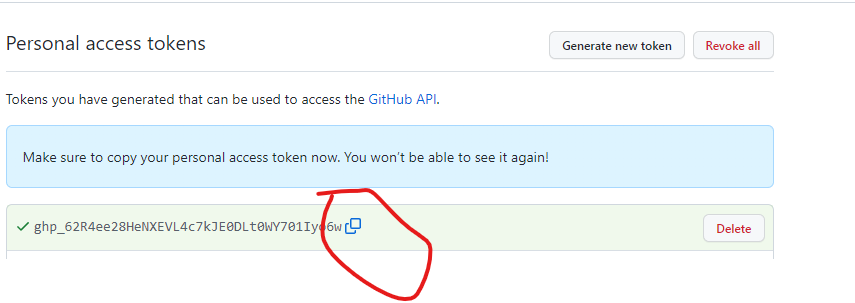
* 1. Sélectionner “Generate” :



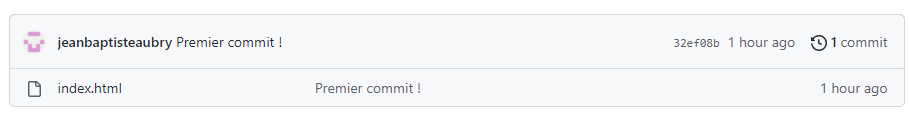
* Authentifier vous sous github
* Générer un nouveau token :



* Normalement, le token est copié automatiquement, si ce n’est le cas, vous pouvez le faire à la main :



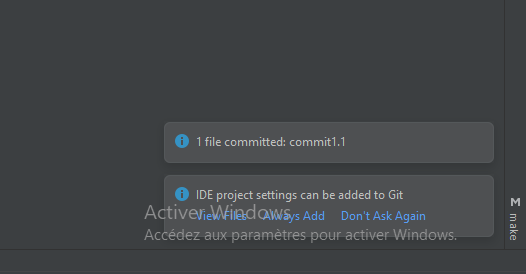
* 1. Github : vous pouvez retrouver votre commit :



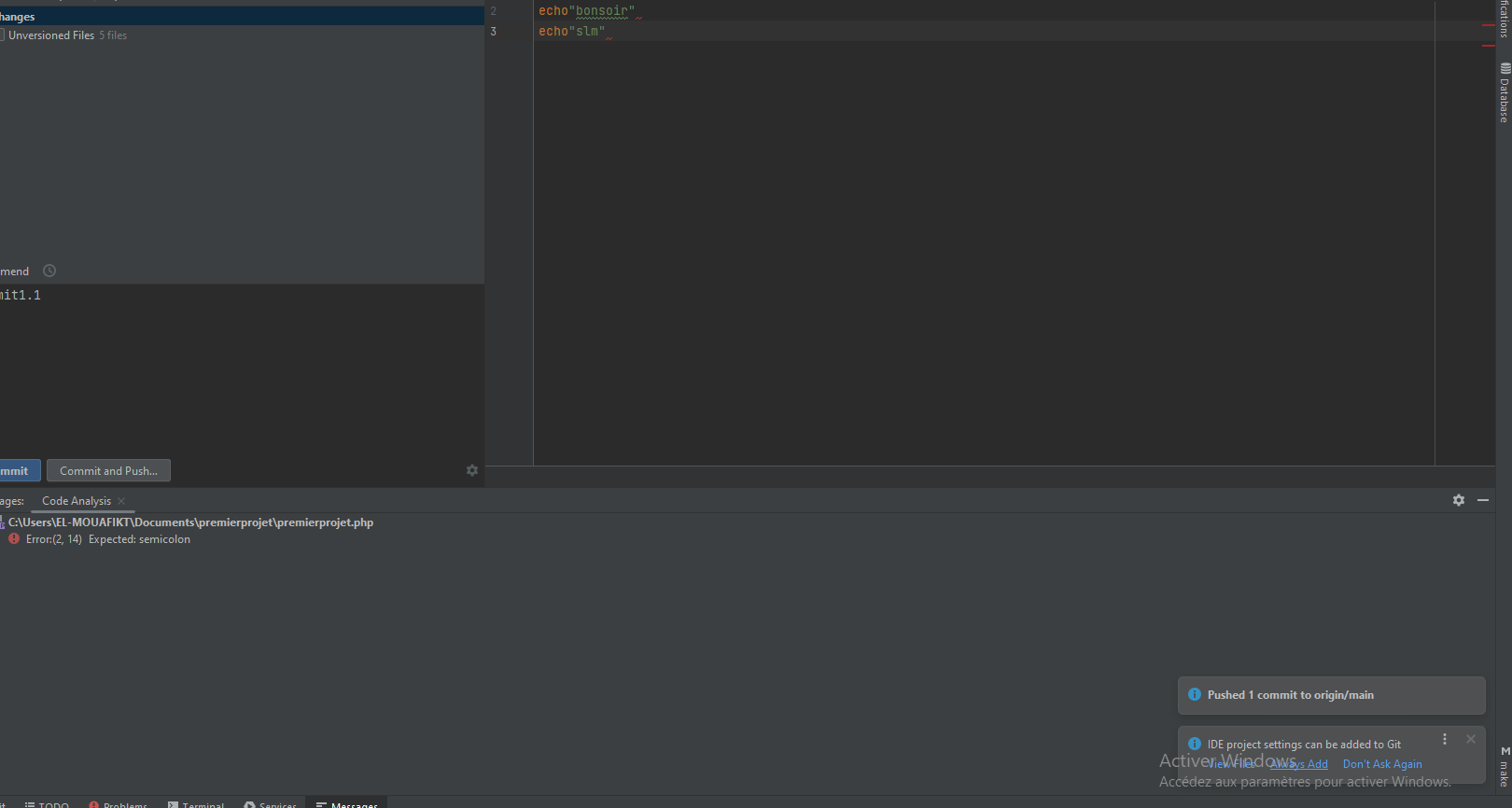
##### **Tâches (capture) :**

17. Illustrer le ‘gitage’ de votre projet sur github :

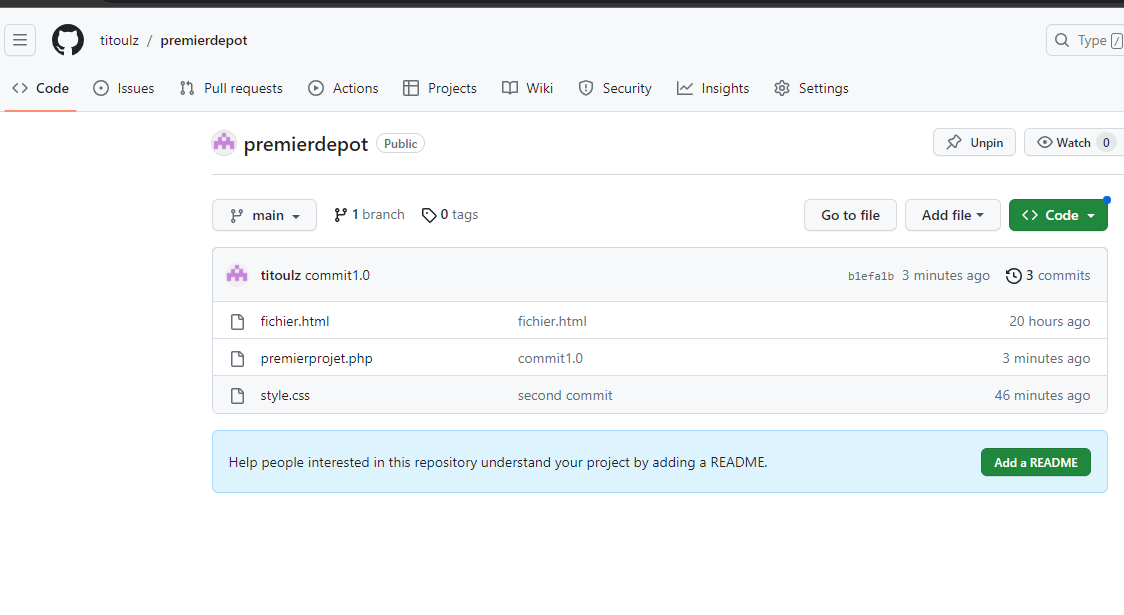
1. phpstorm : faire nouveau commit (il faudra modifier qqc !)

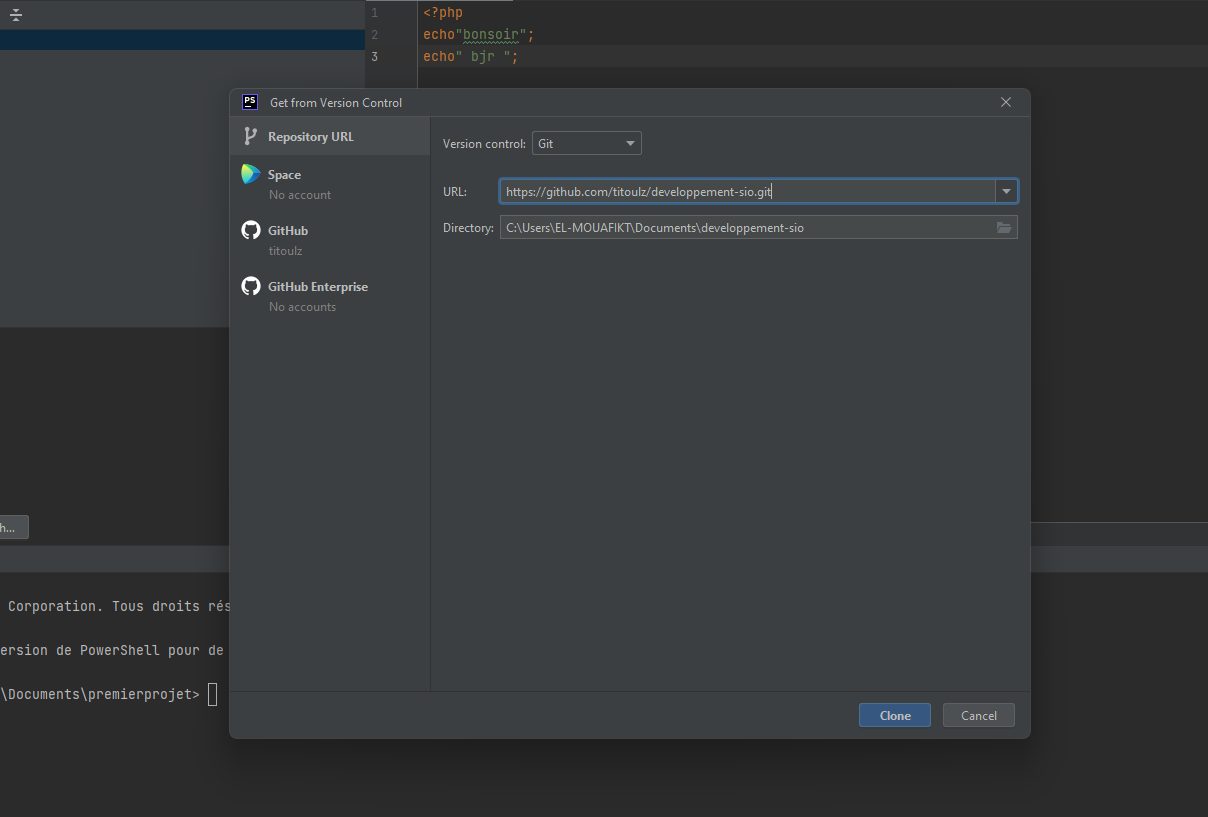
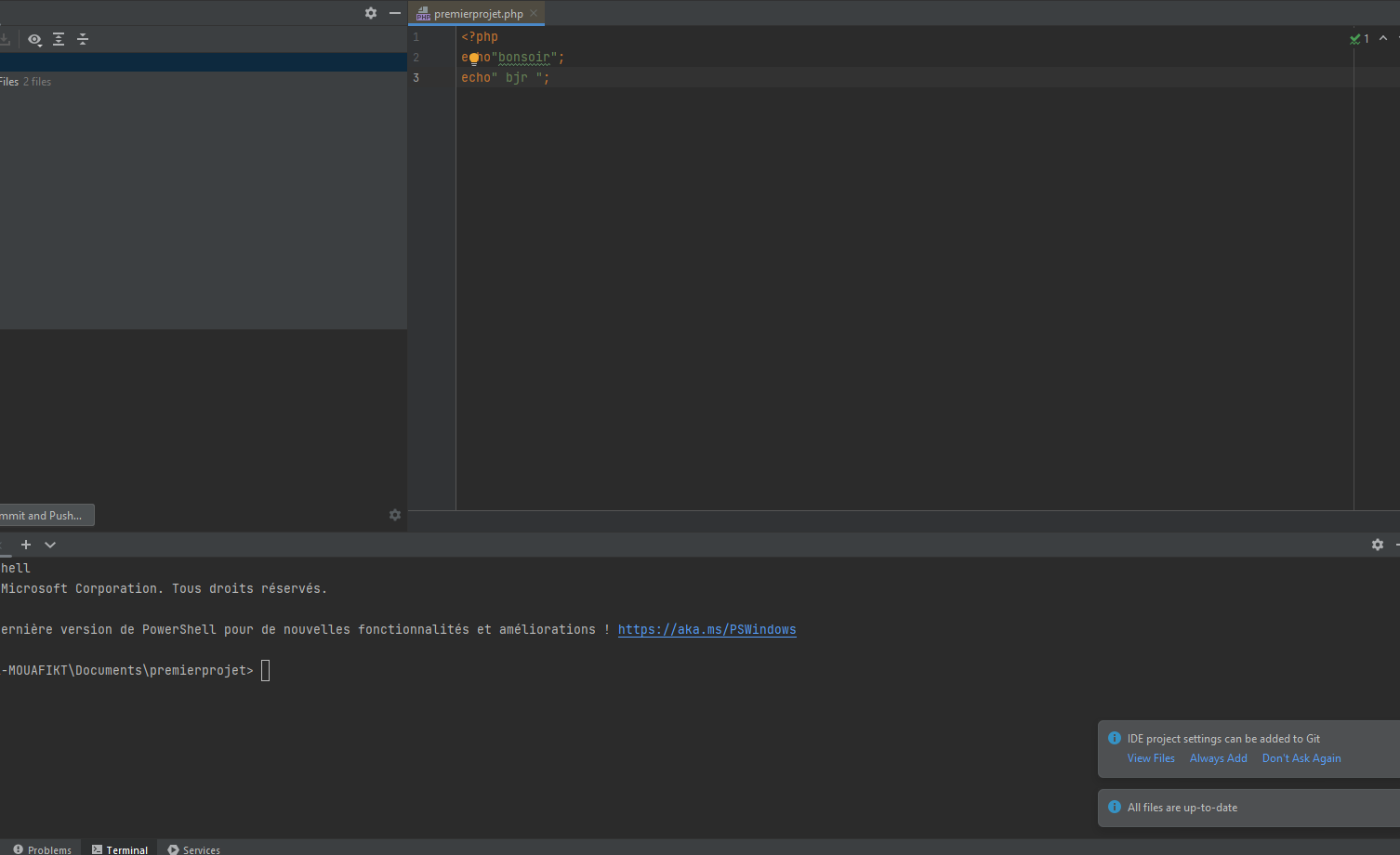


1. phpstorm : envoyer les modifications sur le dépôt : git push



1. github : consulter les modifications sur le dépôt



1. phpstorm : faire le clonage du dépôt d’un de vos camarades (git clone …)
2. 
3. phpstorm : récupérer des modifications du projet déjà cloné (git pull ou update project) en 4.
4. 
5. phpstorm : le changement de remote du projet cloné en 4 vers un **nouveau** dépôt à vous (git → manage remote → …)

