

SAE14 : se présenter sur internet

Repository et Versionning sur GITHUB

Prérequis :

- **Avoir un projet Symfony qui fonctionne**
- **Avoir un compte sur GitHub**
 - Si vous avez un compte, identifiez-vous en cliquant sur sign in
 - Sinon, vous devez créer un compte en cliquant sur sign up

Ce compte vous permettra de créer un repository (dépôt en français).

Sélectionner l'option gratuite.

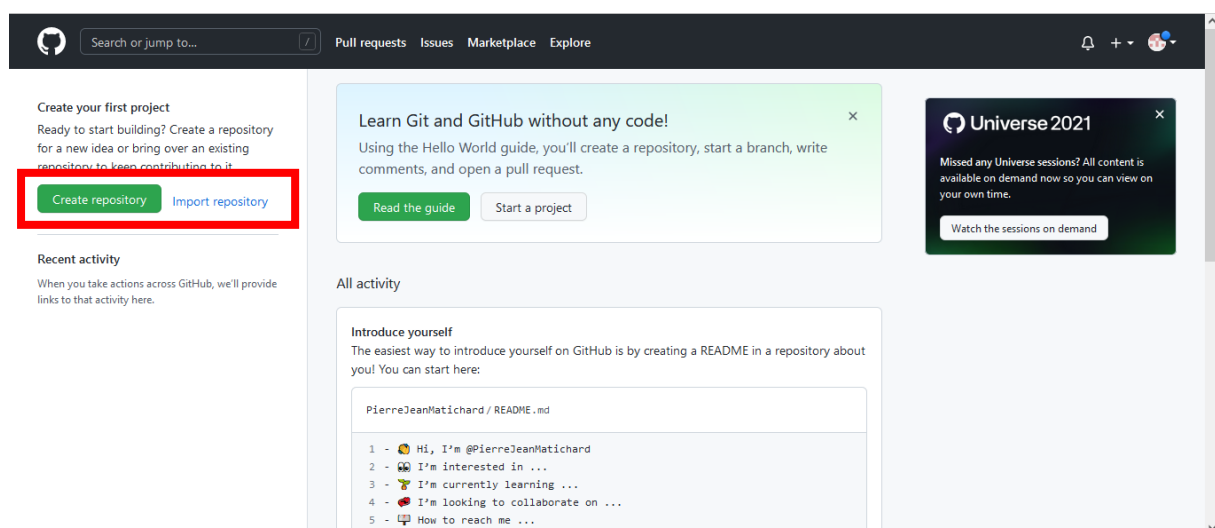
Vous avez déjà utilisé la commande : « git init »

Elle permet d'instancier un repository local. Les dossiers ont été créés en local, sur votre PC. Ils vont permettre de versionner notre code.

Remarque : Le « versionning » est un système permettant d'enregistrer les évolutions d'un fichier ou d'un ensemble de fichiers au cours du temps de manière à ce qu'on puisse rappeler une version antérieure d'un fichier à tout moment. Pour faire simple, ce système permet d'historier vos modifications sur vos fichiers et de revenir à une version antérieure à tout moment.

Vous pouvez faire la même chose sur GITHUB.

Pour cela, cliquer sur « create repository »




Puis cliquer sur « new » et suivre les instructions suivantes :

Noter un Nom pour votre repository

Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Owner * Repository name *

 PierreJeanMatichard /

Great repository names are short and memorable. Need inspiration? How about [vigilant-journey?](#)

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

J'ai appelé mon repository : **ProjetPortfolio**

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ **Add a README file**

This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**

Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**

A license tells others what they can and can't do with your code. [Learn more.](#)

[Create repository](#)

La méthode à utiliser est : « or push an existing repository from the command line ».

...or push an existing repository from the command line

```
git remote add origin https://github.com/PierreJeanMatichard/ProjetPortfolio.git
git branch -M main
git push -u origin main
```

Sur l'invite de commande, placer vous dans le dossier de votre projet Symfony.

Dans mon cas :

```
C:\Users\Administrateur>cd ../../  
C:\>cd symfony/projet/portfolio
```

Coller et exécuter la 1^{ère} commande :

git remote add origin <https://github.com/PierreJeanMatichard/ProjetPortfolio.git>

Donc :

```
C:\Symfony\Projet\portfolio>git remote add origin https://github.com/PierreJeanMatichard/ProjetPortfolio.git  
C:\Symfony\Projet\portfolio>
```

Elle ne retourne rien !

Pour vérifier, taper la commande : « **git status** »

```
C:\Symfony\Projet\portfolio>git status  
On branch master  
  
No commits yet  
  
Untracked files:  
  (use "git add <file>..." to include in what will be committed)  
    .env  
    .env.test  
    .gitignore  
    bin/  
    composer.json  
    composer.lock  
    config/  
    docker-compose.override.yml  
    docker-compose.yml  
    migrations/  
    phpunit.xml.dist  
    public/  
    src/  
    symfony.lock  
    templates/  
    tests/  
    translations/  
  
nothing added to commit but untracked files present (use "git add" to track)  
C:\Symfony\Projet\portfolio>
```

Vous pouvez remarquer que des fichiers (en rouges) ne sont pas « suivis » :

« untracked files : »

Ces fichiers en rouge ne seront pas présents lorsque vous aller pousser (push) votre projet Symfony sur github.

Pour remédier à ça, taper la commande suivante : « **git add .** »

Remarque : le « . » signifie « all »

```
C:\Symfony\Projet\portfolio>git add .
warning: LF will be replaced by CRLF in .env.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in .env.test.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in .gitignore.
```

Vérifier à l'aide de la commande « **git status** » que tous les fichiers sont suivis (en vert).

```
C:\Symfony\Projet\portfolio>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   .env
        new file:   .env.test
        new file:   .gitignore
        new file:   bin/console
        new file:   bin/phpunit
        new file:   composer.json
        new file:   composer.lock
        new file:   config/bundles.php
        new file:   config/packages/cache.yaml
```

Pour enregistrer les modifications que vous venez d'apporter, exécuter la commande suivante : **git commit -m "Initial commit"**

```
C:\Symfony\Projet\portfolio>git commit -m "Initial commit"
[master (root-commit) a6dbccb] Initial commit
49 files changed, 11190 insertions(+)
create mode 100644 .env
create mode 100644 .env.test
create mode 100644 .gitignore
create mode 100644 bin/console
create mode 100644 bin/phpunit
create mode 100644 composer.json
create mode 100644 composer.lock
create mode 100644 config/bundles.php
create mode 100644 config/packages/cache.yaml
create mode 100644 config/packages/dev/debug.yaml
create mode 100644 config/packages/dev/monolog.yaml
create mode 100644 config/packages/dev/web_profiler.yaml
```

REMARQUE : Il se peut qu'un message d'erreur s'affiche, vous devrez alors insérer votre adresse mail. Dans ce cas voici les instructions :

```
*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.
fatal: unable to auto-detect email address (got 'Mehdi@DESKTOP-RJ67DG9.(none)')
```

Pour cela, utiliser la commande suivante :

```
git config --global user.email "toto@etu.univ-st-etienne.fr"
```

Puis exécuter de nouveau la commande : `git commit -m "Initial commit"`

Si tout est ok, alors :

Alors, il faut taper les commandes suivantes sur les PC fixes de l'IUT.

Attention, sans ces commandes, ça ne fonctionne pas.

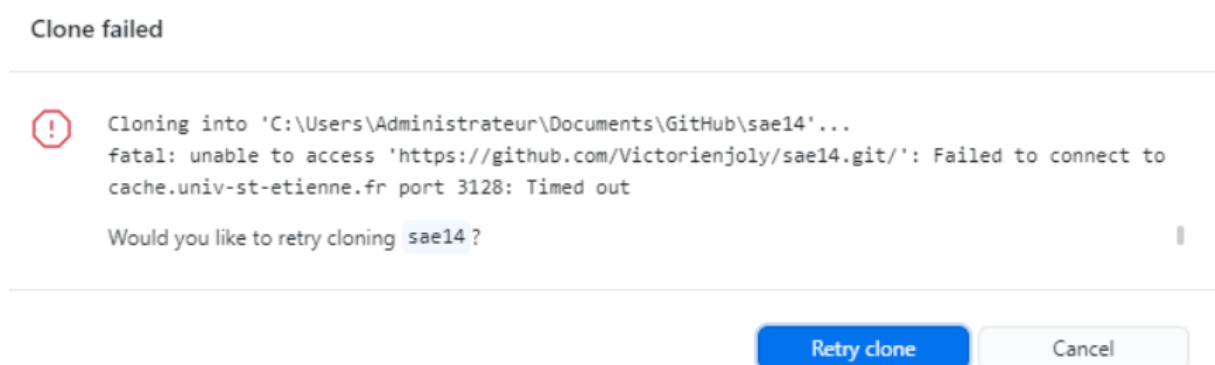
```
git config --global --unset http.proxy
```

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

Remarque : si vous ne les faites pas, il se peut qu'un message d'erreur s'affiche, exemple :

```
C:\Users\Administrateur\farissier>git push -u origin main
fatal: unable to access 'https://github.com/Simonfarissier/SAE14.git/': Failed to connect to cache.univ-st-etienne.fr port 3128: Timed out
```

Remarque : Il faudra les faire sur tous les PC fixes, de chaque salle, pour pouvoir cloner votre repository, sinon ce message s'affichera :



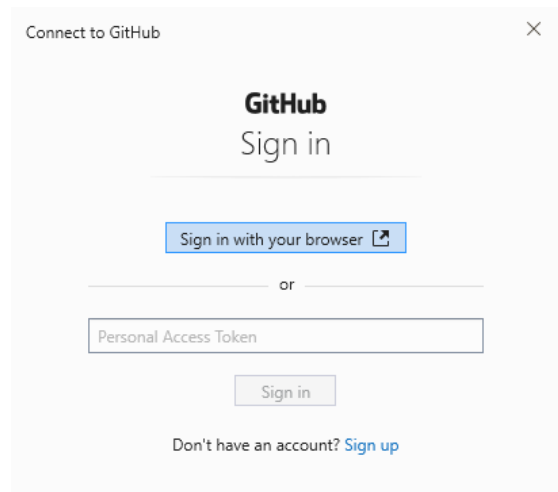
Puis, taper les commandes suivantes pour pousser vos fichiers sur GitHub :

git branch -M main

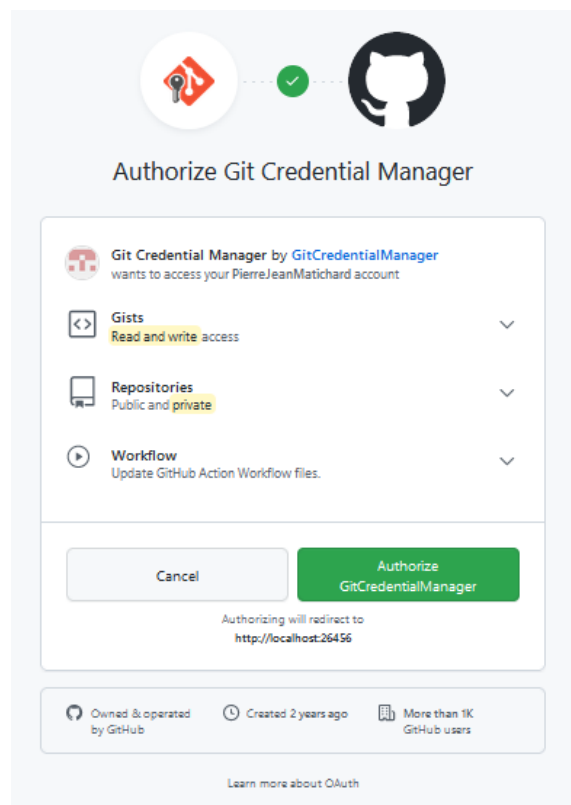
git push -u origin main

Une fenêtre devrait vous demander de vous identifier sur GitHub.

Cliquer sur « sign in with yours browser »



Puis il faut autoriser :



Enfin votre mot de passe :



Confirm access

Password

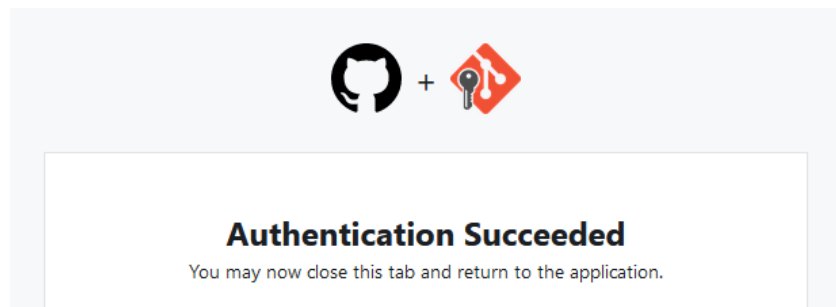
[Forgot password?](#)

Confirm password

Tip: You are entering [sudo mode](#). We won't ask for your password again for a few hours.

[Terms](#) [Privacy](#) [Security](#) [Contact GitHub](#)

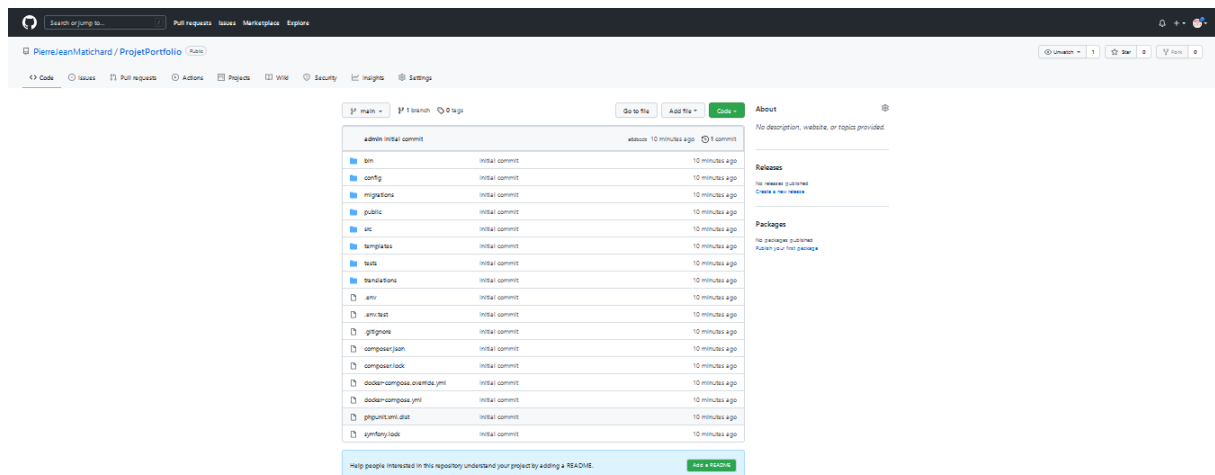
Ce message doit apparaître :



Il ne vous reste plus qu'à vérifier sur GitHub que tous vos dossiers et fichiers sont suivis.

Pour cela fermer la fenêtre précédente, aller sur GitHub, puis sur votre repository.

Le mien se nomme ProjetPortfolio :

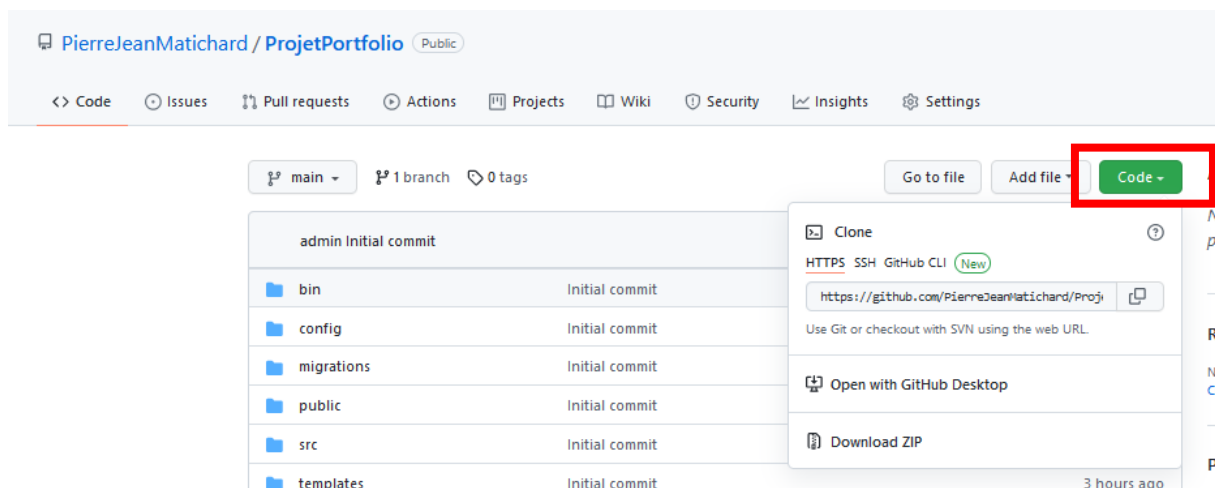


Pour ouvrir, travailler sur votre projet depuis n'importe quel ordinateur, voici une solution :

Prérequis pour appliquer cette solution : avoir téléchargé et installé :

- GitHub desktop
- Visual studio code

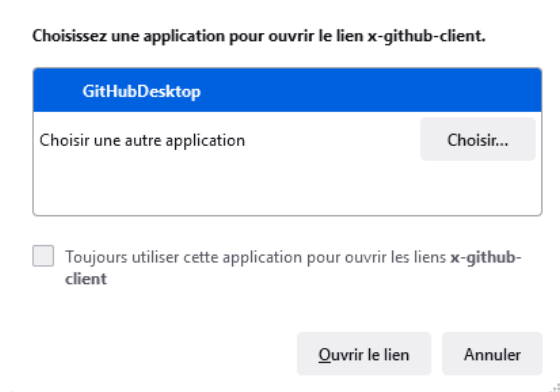
Puis sur votre GitHub, cliquer sur code, puis : Open with GitHub Desktop



Ici, cliquer sur toujours autoriser Puis choisir une application

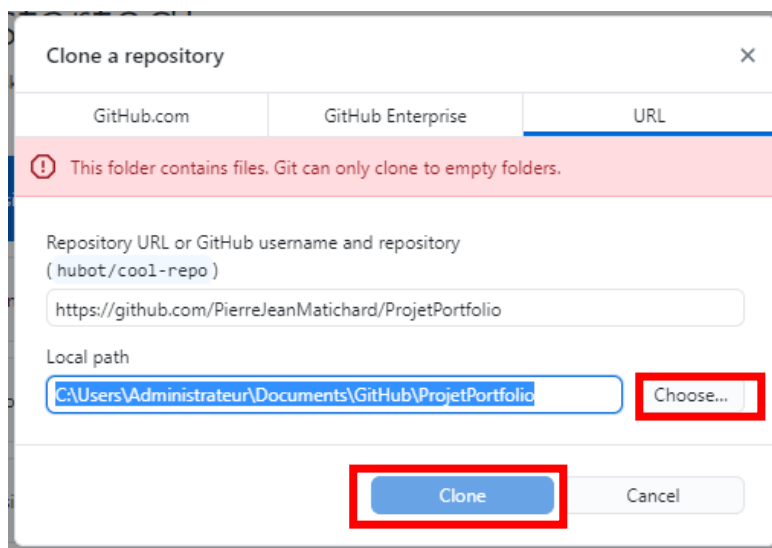


Choisir GitHub Desktop et cliquer sur ouvrir le lien :



Sélectionner le dossier dans lequel vous voulez ouvrir votre projet, dans mon cas, je suis toujours sur le même PC donc je sélectionne les mêmes dossiers.

Sinon cliquer sur « choose ».

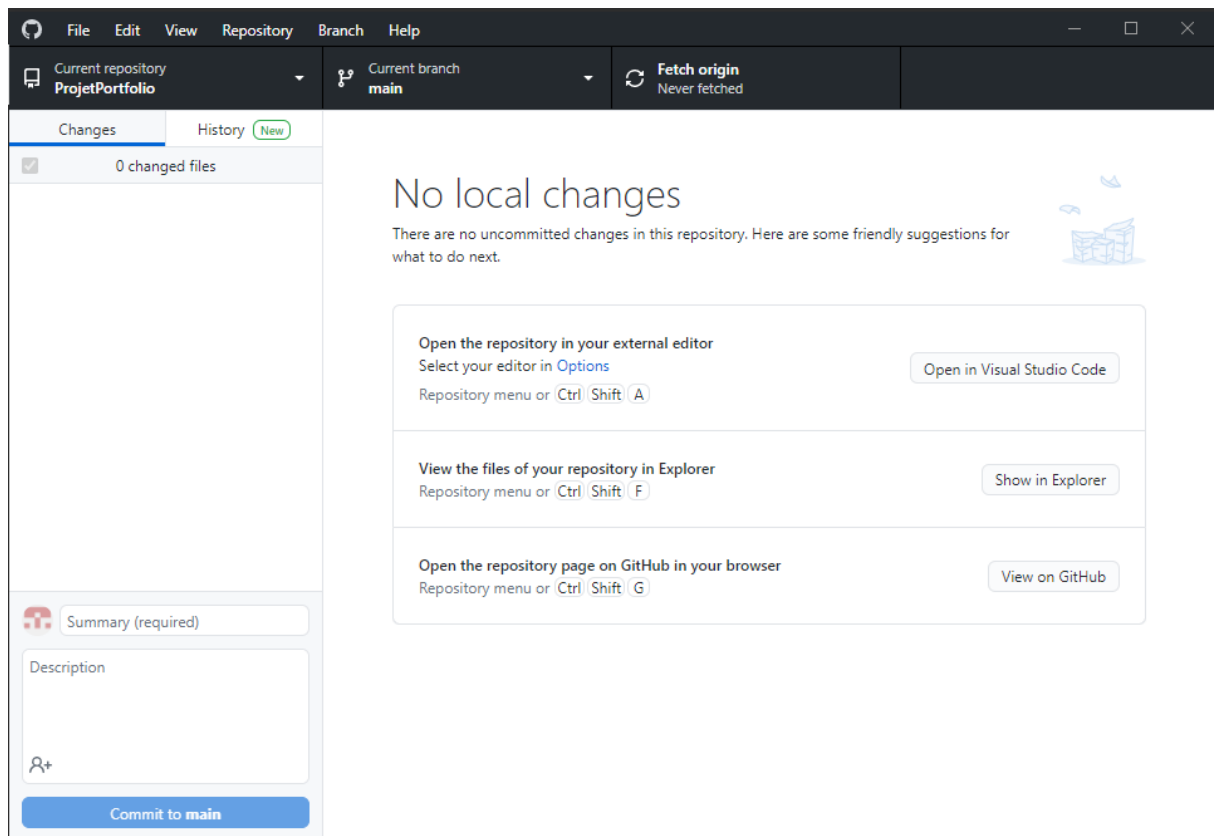


Puis cliquer sur « clone »

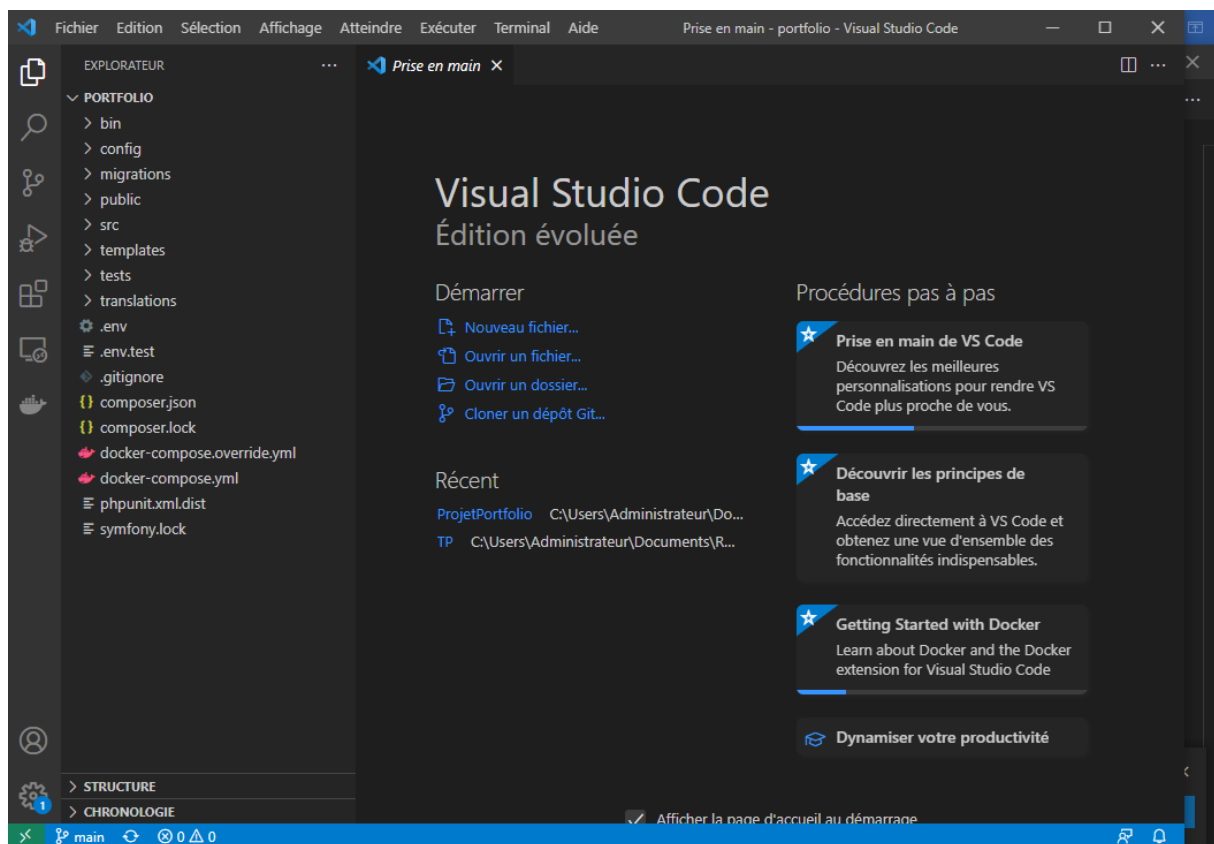
Remarque : si vous avez déjà utiliser ce même PC et que vous avez déjà cloner le projet depuis GitHub, vous devez créer et sélectionner un autre fichier en cliquant sur « choose ».

Maintenant, vous devez vérifier les informations, notamment « current repository »

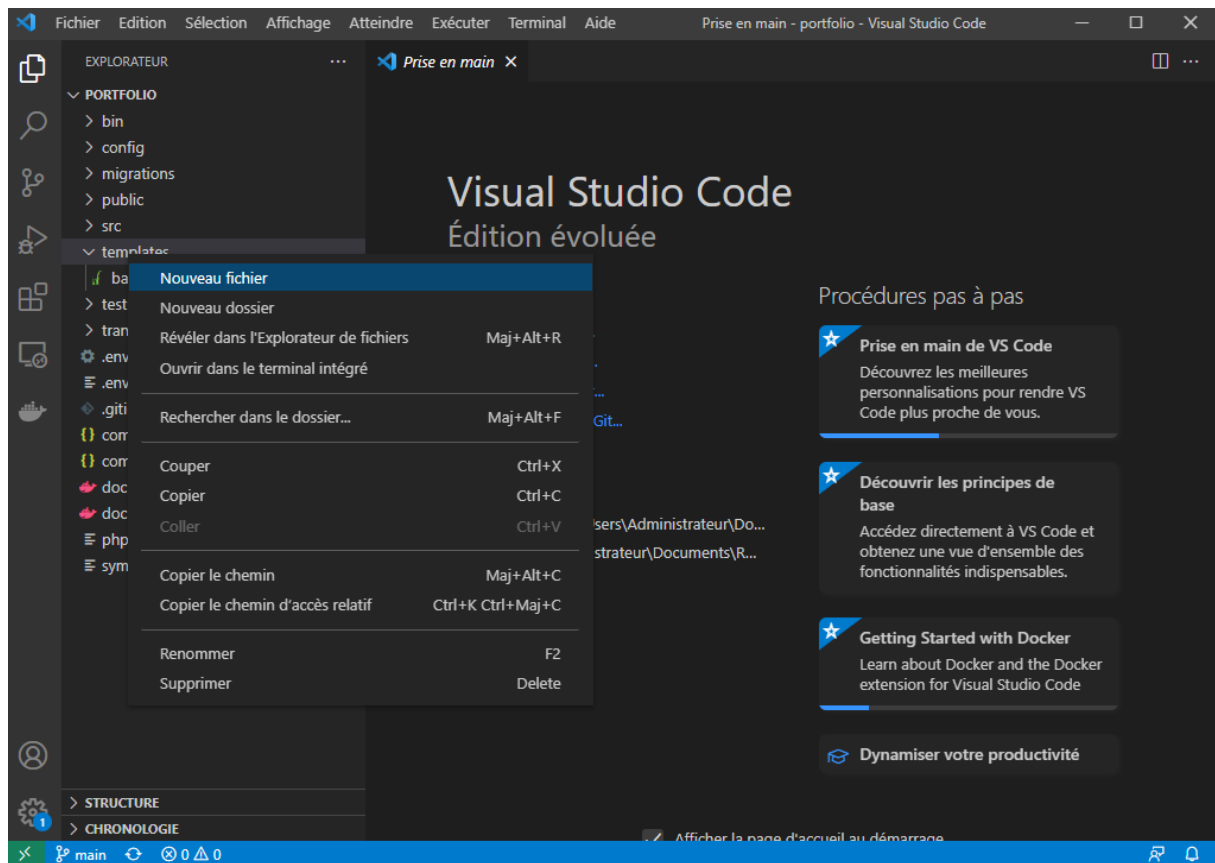
Et cliquer sur « open in Visual studio code »



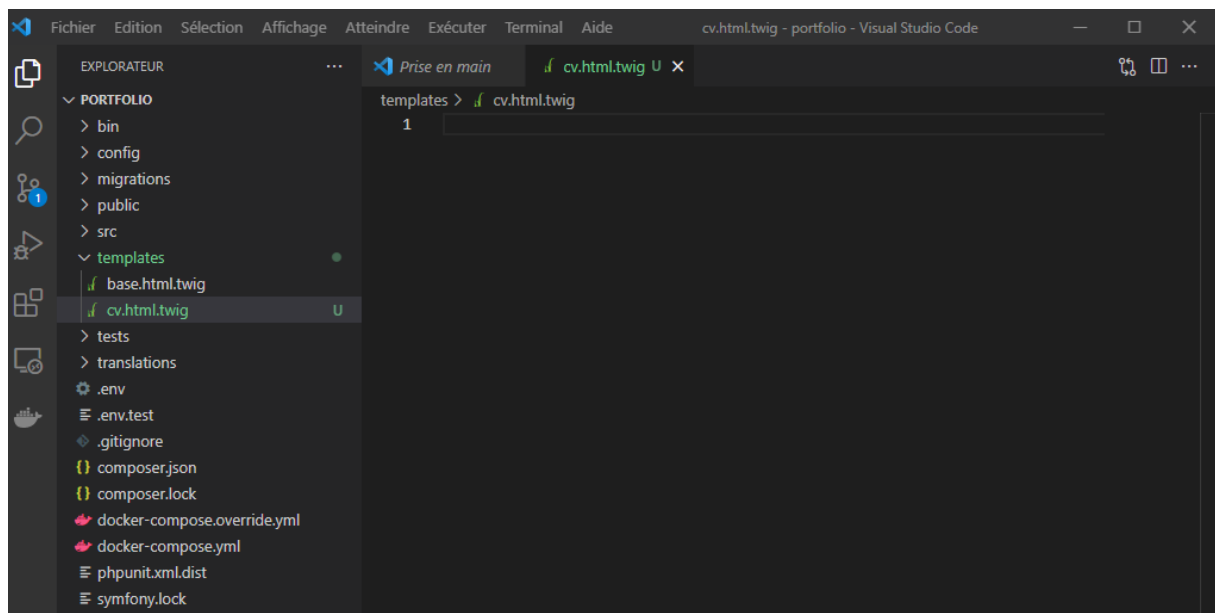
Cette page s'affiche après avoir cliqué sur faire confiance :



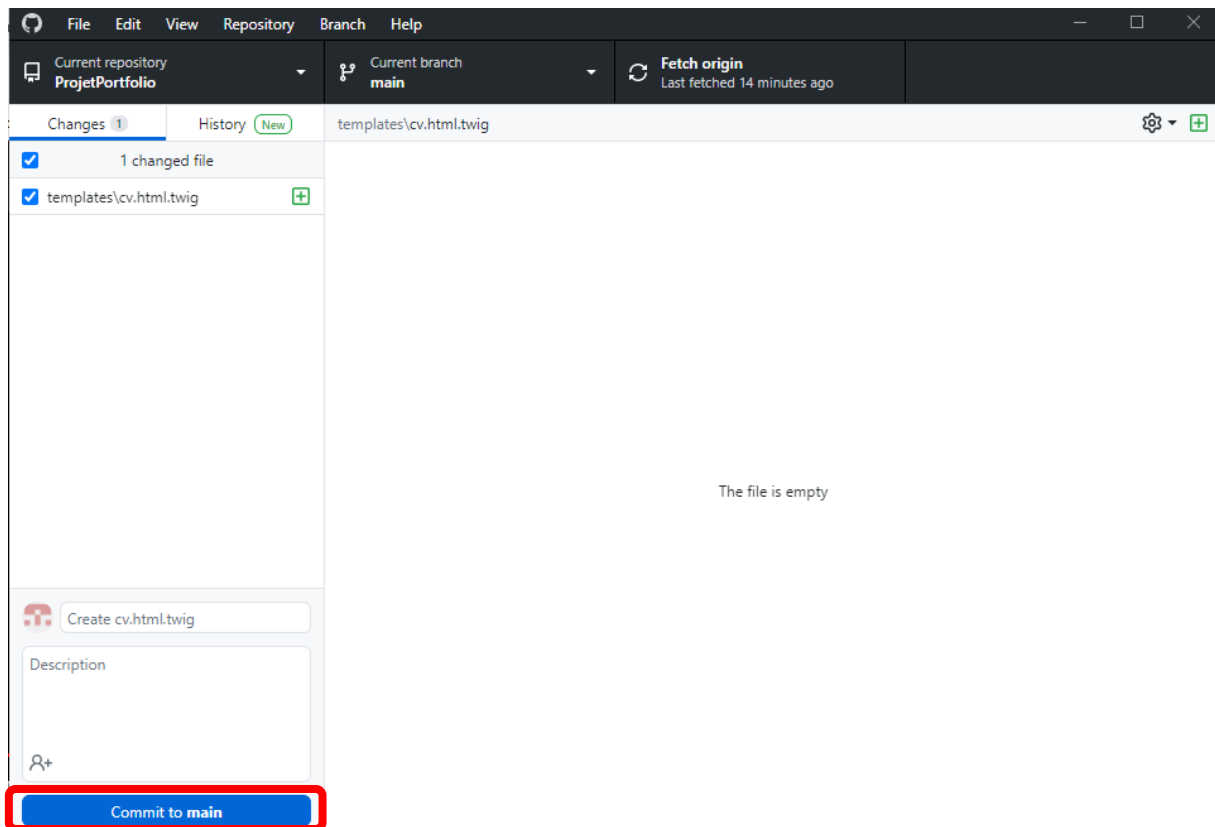
Maintenant lorsque vous créez ou modifiez votre projet, par exemple vous créez un nouveau « template » :



Je le nomme « cv.html.twig » :

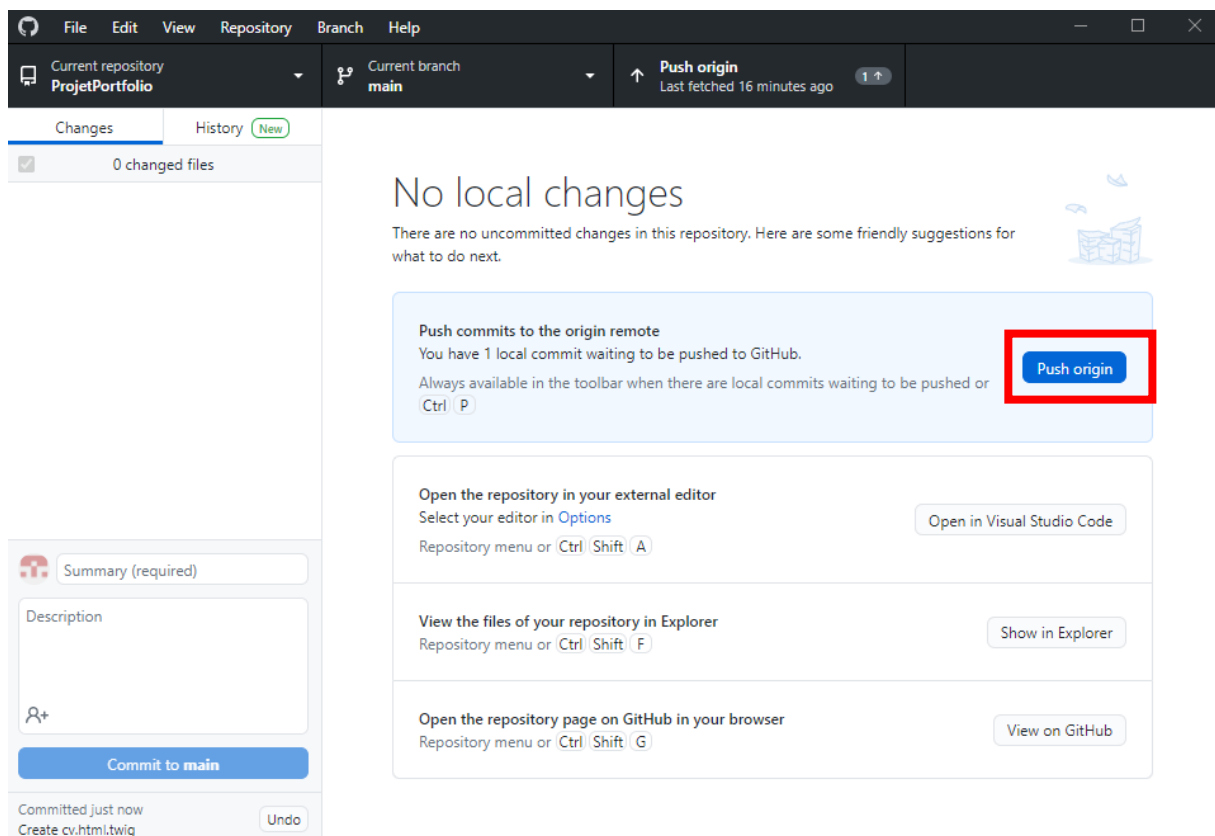


Dès lors, sur GitHub Desktop, le fichier apparaît :

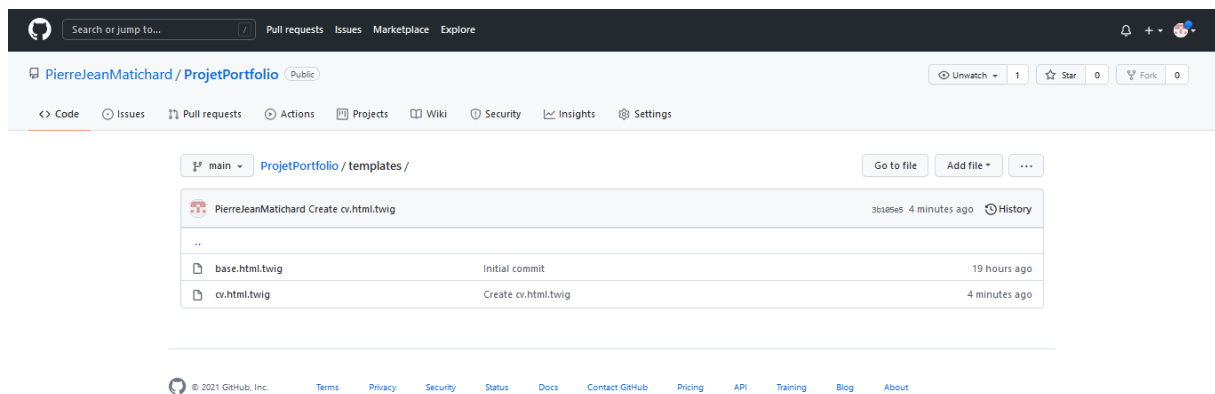


Si vous voulez l'ajouter à votre repository GitHub, alors cliquer sur : « Commit to main », puis sur « push origin ».

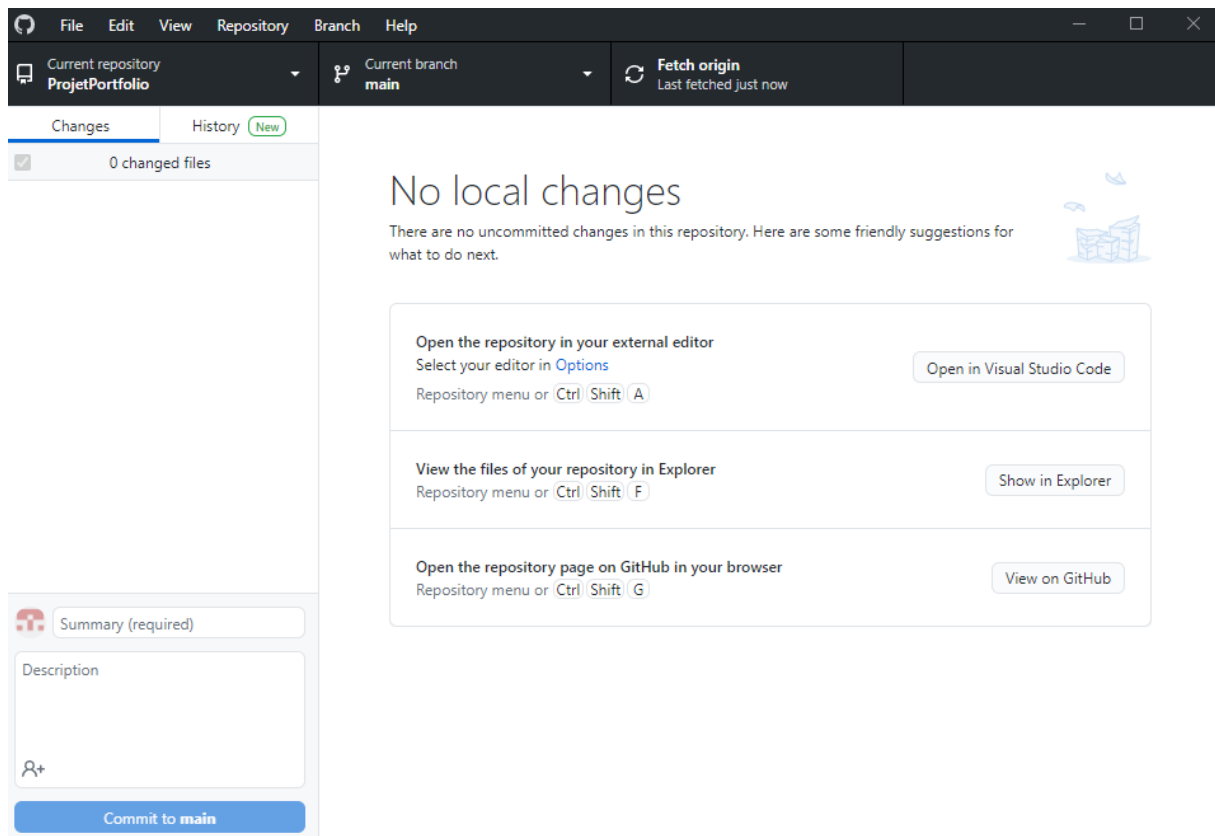
Attention, il faut les pousser 1 par 1.



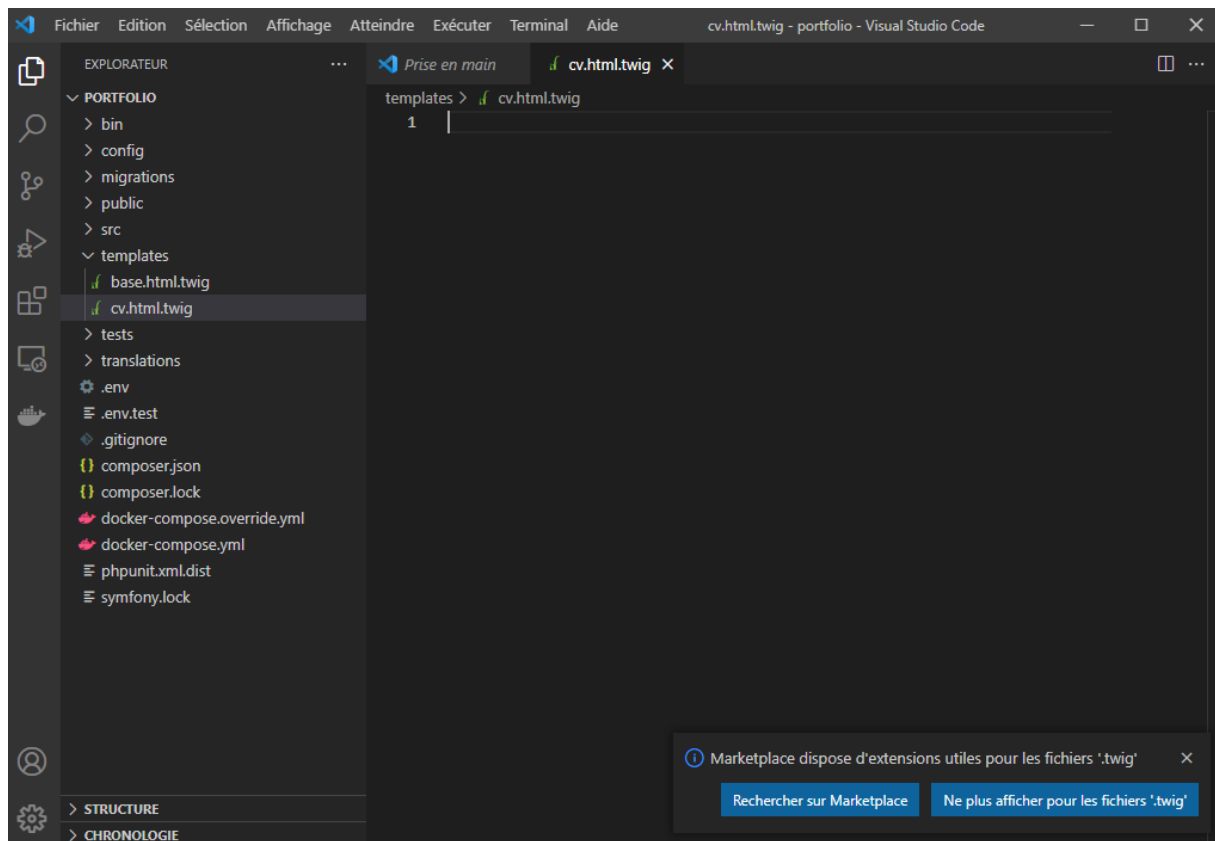
On constate que GitHub Desktop à versionner votre repository sur GitHub :



Et qu'il a été chercher cette dernière version (fetch origin). Donc si vous ouvrez de nouveau votre projet en cliquant sur « Open in Visual studio code » (il faut avoir fermer Visual studio code pour ça) :



Le fichier « cv.html.twig » est bien présent :



Pour lancer le serveur symfony, il faut aller dans le dossier GitHub créé lors du clonage.

C:\users\administrateur\documents\github\ suivi du nom de votre projet

Puis taper la commande suivante :

composer require symfony/runtime