**✅ ÉTAPE 1 : Initialiser le dépôt Git**

**But** : dire à Git que ce dossier est un projet sous gestion de version.

**📌 Commande**

Dans le terminal (PyCharm ou autre), place-toi dans le dossier churn\_V1 et tape :

***git init***

**✅ Ce que ça fait :**

* **Cela crée un dossier caché .git à la racine de ton projet. Ce dossier contient toutes les informations de versionnement pour Git.**
* **Git te donne un message comme ça : Initialized empty Git repository in ...**

**Résultat attendu :**

* **Après avoir lancé git init, tu devrais voir dans le terminal une confirmation indiquant que le dépôt Git a bien été initialisé, comme par exemple :**

**Initialized empty Git repository in C:/Users/Username/Documents/Churn/churn\_V1/.git/**

**Vérification**

1. **Vérifie si le dossier .git a été créé** :
   * Ouvre l'Explorateur de fichiers sur ton PC.
   * Va dans ton dossier churn\_V1.
   * Assure-toi que les fichiers cachés sont visibles en activant "Afficher les fichiers cachés" dans l'explorateur (Sous **Affichage**, coche **Éléments masqués**).
   * Tu devrais voir un dossier .git dans ton répertoire churn\_V1.

**✅ Étape 2 : Créer le fichier .gitignore**

Le fichier **.gitignore** permet à Git de savoir quels fichiers il ne doit pas suivre. Par exemple, tu veux que Git ignore les fichiers temporaires, les fichiers système ou les fichiers de l'environnement virtuel (comme ceux créés par Python).

1. **Crée le fichier .gitignore** à la racine de ton projet :
   * Va dans le dossier churn\_V1 et crée un fichier nommé .gitignore.
   * Pour cela, tu peux créer un fichier texte classique et simplement le renommer en .gitignore (assure-toi que l'extension .txt soit bien supprimée, sinon ça ne fonctionnera pas).
2. **Ajoute des règles de base dans ce fichier** :  
   Voici quelques règles standard à ajouter dans ton fichier .gitignore pour ignorer les fichiers et dossiers courants qui ne doivent pas être suivis par Git :

**Contenu du fichier .gitignore** :

# Ignorer les fichiers de l'environnement virtuel

.env/

venv/

env/

# Ignorer les fichiers Python compilés

\*.pyc

\*.pyo

# Ignorer les fichiers d'IDE

.idea/

\*.iml

# Ignorer les fichiers système

Thumbs.db

.DS\_Store

* + **.env/** : ignorer les dossiers d'environnements virtuels.
  + \*\**.pyc, .pyo* : ignorer les fichiers Python compilés.
  + **.idea/** : ignorer les dossiers et fichiers créés par PyCharm (IDE).
  + **Thumbs.db, .DS\_Store** : ignorer les fichiers système générés sous Windows ou macOS.

1. **Sauvegarde le fichier .gitignore**.

**✅ Ce que ça fait :**

* Le fichier .gitignore indique à Git quels fichiers ou dossiers ne doivent pas être suivis ni envoyés vers le dépôt Git.
* En utilisant ce fichier, tu évites de pousser des fichiers inutiles ou sensibles vers ton dépôt Git, comme ton environnement virtuel ou des fichiers de configuration IDE.

**Vérification**

1. Si tu ouvres ton terminal dans PyCharm, tu peux vérifier si le fichier .gitignore a été bien ajouté à ton projet en utilisant cette commande :

dir

Sous Windows, cette commande listera les fichiers et dossiers de ton répertoire actuel. Tu devrais voir ton fichier .gitignore dans la liste.

1. **Vérifie aussi que les fichiers ignorés ne sont pas suivis par Git** :  
   Si tu as déjà ajouté des fichiers à Git (avant de créer .gitignore), tu peux vérifier les fichiers suivis avec :

git status

Cela te montrera les fichiers suivis ou modifiés par Git. Si des fichiers comme .env ou d'autres éléments ignorés apparaissent, cela signifie qu'ils sont déjà suivis.

**✅ Étape 3 : Ajouter les fichiers au staging (préparer le commit)**

Maintenant que ton fichier .gitignore est prêt et que tu as ton projet dans le dossier churn\_V1, tu vas **indiquer à Git quels fichiers tu veux suivre**. Pour cela, on les ajoute au "**staging area" (zone de préparation).**

**📌 Commande à exécuter :**

Ouvre le **terminal dans PyCharm** (Alt + F12 si besoin), puis tape :

***git add .***

✅ **Explication** :

* git add . : ajoute **tous les fichiers** (sauf ceux ignorés par .gitignore) au "staging area", c’est-à-dire la zone de préparation avant le commit.

💡 Tu peux aussi ajouter un fichier précis avec git add nom\_du\_fichier.py si tu veux être plus sélectif.

**🔍 Vérification :**

Après avoir ajouté les fichiers, tu peux vérifier ce qui est prêt à être "commit" en tapant :

***git status***

Tu verras les fichiers en vert = prêts à être commités ✅

✅ **Objectif** :  
Préparer une "photo" de ton projet à ce moment précis. Une fois les fichiers ajoutés, tu pourras faire un commit.