# Documentation de l'Environnement du Projet DevOPTIQ

## 1. Introduction

Ce document détaille les étapes nécessaires pour configurer, maintenir et relancer l'environnement de développement pour le projet DevOPTIQ. Il contient également des instructions pour résoudre les problèmes courants.

## 2. Outils requis

Voici la liste des outils nécessaires pour travailler sur le projet :

* - Git (version 2.47.1 ou supérieure)
* - Python 3.10 avec pip pour gérer les dépendances
* - PowerShell 7 ou un terminal compatible avec Git Bash
* - Microsoft Teams et OneDrive pour la gestion des fichiers
* - Un éditeur de texte ou IDE (comme VS Code ou PyCharm).

## 3. Structure des fichiers

La structure des fichiers du projet DevOPTIQ est organisée comme suit :

* - Archives/ - Contient les fichiers archivés.
* - Bases de données/ - Répertoire pour les fichiers liés aux bases de données.
* - Code/ - Contient le code source du projet.
* - Templates/ - Répertoire pour les fichiers modèles.
* - Venv/ - Répertoire pour l'environnement virtuel Python.
* - requirements.txt - Liste des dépendances Python.
* - .gitignore - Liste des fichiers à ignorer par Git.
* - test.txt - Fichier de test pour vérifier le dépôt Git.

## 4. Étapes pour relancer l'environnement

Suivez les étapes ci-dessous pour relancer correctement l'environnement après un redémarrage :

1. - Ouvrez PowerShell 7 ou Git Bash.
2. - Naviguez vers le répertoire du projet : `cd "C:/Users/Hubert.AFDEC/A.F.D.E.C/Projet OPTIQ - DevOPTIQ"`
3. - Activez l'environnement virtuel Python : `source Venv/Scripts/activate` (ou `Venv\Scripts\activate` sous Windows).
4. - Assurez-vous que les dépendances sont installées : `pip install -r requirements.txt`.
5. - Vérifiez la connexion avec le dépôt Git : `git remote -v`.
6. - Tirez les dernières modifications depuis GitHub : `git pull`.

## 5. Bonnes pratiques

Voici quelques bonnes pratiques à suivre :

* - Commitez régulièrement vos modifications avec des messages clairs.
* - Mettez à jour votre dépôt local avant de travailler (`git pull`).
* - N'oubliez pas d'ajouter des fichiers dans Git après leur création (`git add`).
* - Assurez-vous que tous les fichiers sensibles sont inclus dans `.gitignore`.

## 6. Résolution des problèmes courants

Voici quelques problèmes courants et leurs solutions :

Problème : Erreur de permission lors de l'ajout d'un fichier à Git  
Solution : Ajoutez le fichier problématique à `.gitignore`.

Problème : Fichiers non synchronisés avec GitHub  
Solution : Vérifiez votre connexion Internet et assurez-vous que l'origine Git est correcte (`git remote -v`).

Problème : Erreur lors de l'activation de l'environnement virtuel  
Solution : Assurez-vous que le chemin vers l'environnement virtuel est correct.