CI/CD dans GitHub Actions

1. C'est quoi ?

- **CI (Intégration Continue)** : automatiser les tests, la vérification et la validation du code à chaque modification (push, pull request).
- **CD (Livraison/Déploiement Continu)** : automatiser la préparation ou la mise en production du code validé (build, zip, déploiement...).

2. Pourquoi on l'utilise ?

- S'assurer que le code reste fonctionnel à chaque modif.
- Éviter d'intégrer du code cassé.
- Garantir la qualité du code (lint, formatage).
- Livrer plus rapidement et plus sûrement.
- Dans un projet pro, c'est indispensable dès qu'on est plusieurs ou en prod.

3. Comment ça fonctionne ?

CI (exemple typique)

- À chaque push :
- Installer l'environnement Python
- Installer les dépendances
- Lancer `pytest`
- Lancer `flake8` ou `black` pour le style
- Afficher un résumé des erreurs si ça pète

CD (optionnel pour toi)

- Build d'un exécutable / archive / image Docker
- Déploiement automatique (cloud, FTP, serveur, etc.)

4. Vocabulaire essentiel

Terme	Explication
Workflow	Script d'actions GitHub (écrit en YAML)
Job	Suite d'étapes (build, test, etc.)
Step	Action unique (ex : `pip install`, `pytest`, etc.)
Runner	Machine virtuelle GitHub qui exécute le workflow
Trigger	Événement déclencheur (`push`, `pull_request`, etc.)

5. Syntaxe & Construction (GitHub Actions)

Fichier à créer : `.github/workflows/python-ci.yml`

```yaml
name: Python CI

on:
  push:
    branches: [ main ]
  pull_request:
    branches: [ main ]

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
    - name: Checkout code
      uses: actions/checkout@v3

    - name: Set up Python
      uses: actions/setup-python@v4
      with:
        python-version: '3.11'

    - name: Install dependencies
      run: |
        python -m pip install --upgrade pip
        pip install -r requirements.txt
        pip install flake8 pytest

    - name: Lint with flake8
      run: flake8 .

    - name: Run tests with pytest
      run: pytest
```

6. Représentation visuelle

```
  [Push sur main] ---> [CI GitHub Actions]
                       |--> [Install deps]
                       |--> [flake8]
                       |--> [pytest]
                       |--> [Succès / Erreurs]
```

✅Résultat attendu

Un fichier `.github/workflows/python-ci.yml` dans ton projet `machineMonitor` qui :

- s'exécute à chaque push sur `main`
- vérifie le code (`flake8`)
- lance les tests (`pytest`)
- résume le tout dans les Actions GitHub

Prochaine étape : écriture d'un vrai test, mise en place dans ton repo, push, vérification dans GitHub UI.