Projet DevOps – Tests Pytest + Coverage + GitHub Actions

DRISS Ryma - M2 Data Engineer et Cloud

Objectif du projet

Ce projet a pour but de mettre en place une démarche **CI (Intégration Continue)** sur un mini-module Python (`df_ops.py`) :

- écrire des fonctions simples de manipulation de DataFrame (pandas)
- écrire des **tests unitaires** avec **pytest**
- mesurer la **couverture de code** avec **pytest-cov**
- automatiser l'exécution des tests dans **GitHub Actions** à chaque `push` / `pull_request`

Contenu du projet

- `src/df_ops.py` : fonctions de manipulation de DataFrame (moyennes, filtre, contrôle des colonnes...)
- `tests/test_df_ops.py` : tests unitaires pytest
- `.github/workflows/` : workflow CI GitHub Actions (tests automatiques)
- `coverage.xml` : rapport de couverture (généré par pytest-cov)

Structure du projet

```
NOM_PRENOM/
          src/
                __init__.py
                df_ops.py
          tests/
                test_df_ops.py
          requirements.txt
          .github/
              workflows/
                  ci.yml
```

Ce que j'ai fait (étapes)

1. Création du module `src/df_ops.py` :
   - création d'un DataFrame (`build_dataframe`)
   - calculs : `mean_age`, `mean_salary`
   - filtre : `filter_by_department`
   - compteur de lignes : `row_count`
   - vérification des colonnes : `_check_columns` (lève une erreur si colonne manquante)

2. Création des tests `tests/test_df_ops.py` avec pytest :
   - tests des résultats (moyennes, filtre, nombre de lignes)
   - test du cas d'erreur (colonne manquante) avec `pytest.raises(ValueError)`
3. Ajout de la couverture de code :
   - affichage des lignes manquantes (`-cov-report=term-missing`)
   - génération du rapport `coverage.xml`
4. Mise en place de GitHub Actions :
   - installation des dépendances
   - exécution automatique de `pytest` + coverage

Installation et prérequis

- Python 3.x
- pandas
- pytest
- pytest-cov

Installation (Ubuntu/WSL)

```
sudo apt update && sudo apt install -y python3-pandas python3-pytest python3-
   pytest-cov
```

Commandes utilisées

1) Lancer les tests

```
pytest
```

Permet de vérifier que toutes les fonctions du module fonctionnent correctement.

```
ryma@DRISSRYMAHP:/mnt/c/Users/DRISS/Desktop/projet_DEvops/DRISS_Ryma$ pytest
========================= test session starts ==========================
platform linux -- Python 3.12.3, pytest-7.4.4, pluggy-1.4.0
rootdir: /mnt/c/Users/DRISS/Desktop/projet_DEvops/DRISS_Ryma
plugins: cov-4.1.0
collected 5 items

tests/test_df_ops.py .....                                       [100%]
```

2) Lancer les tests avec détails

```
pytest -v
```

Affiche chaque test exécuté (utile pour comprendre ce qui passe/échoue).

3) Générer la couverture de code

```
pytest --cov=src --cov-report=term-missing
```

Mesure le % de lignes exécutées par les tests et affiche les lignes manquantes.

```
ryma@DRISSRYMAHP:/mnt/c/Users/DRISS/Desktop/projet_DEvops/DRISS_Ryma$ pytest -v
============================= test session starts ==============================
platform linux -- Python 3.12.3, pytest-7.4.4, pluggy-1.4.0 -- /usr/bin/python3
cachedir: .pytest_cache
rootdir: /mnt/c/Users/DRISS/Desktop/projet_DEvops/DRISS_Ryma
plugins: cov-4.1.0
collected 5 items

tests/test_df_ops.py::test_build_dataframe_structure_and_types PASSED   [ 20%]
tests/test_df_ops.py::test_means PASSED                                  [ 40%]
tests/test_df_ops.py::test_filter_by_department_it PASSED                [ 60%]
tests/test_df_ops.py::test_row_count PASSED                              [ 80%]
tests/test_df_ops.py::test_mean_salary_missing_column PASSED             [100%]

============================== 5 passed in 1.11s ===============================
```

4) Générer un fichier XML de couverture

```
pytest --cov=src --cov-report=xml
```

Crée `coverage.xml` pour l'exploitation en CI/CD

```
ryma@DRISSRYMAHP:/mnt/c/Users/DRISS/Desktop/projet_DEvops/DRISS_Ryma$ pytest --cov=src
============================= test session starts ==============================
platform linux -- Python 3.12.3, pytest-7.4.4, pluggy-1.4.0
rootdir: /mnt/c/Users/DRISS/Desktop/projet_DEvops/DRISS_Ryma
plugins: cov-4.1.0
collected 5 items

tests/test_df_ops.py .....                                               [100%]

---------- coverage: platform linux, python 3.12.3-final-0 -----------
Name             Stmts   Miss  Cover
------------------------------------
src/__init__.py      0      0   100%
src/df_ops.py       19      0   100%
------------------------------------
TOTAL               19      0   100%


============================== 5 passed in 1.25s ===============================
```

Résultat obtenu

- Tous les tests passent (100 %)
- Couverture de code : atteindre une couverture maximale (100% selon les tests)

GitHub Actions (CI)

Le workflow GitHub Actions est placé dans :

```
.github/workflows/
```

Il permet de :

- lancer automatiquement les tests à chaque push / pull request
- installer les dépendances
- générer la couverture

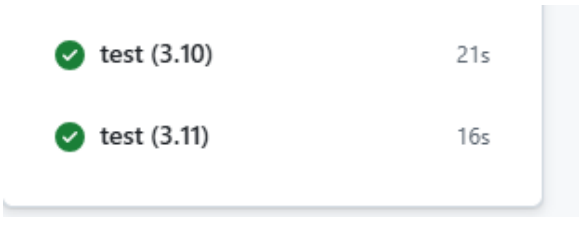

FIGURE 1 – les versions python utilisées

Run tests

```
▶ Run PYTHONPATH=src pytest -q --cov=src
.....                                                                    [100%]
================================ tests coverage ================================
_______________ coverage: platform linux, python 3.10.19-final-0 _______________

Name              Stmts   Miss  Cover
-----------------------------------
src/__init__.py       0      0   100%
src/df_ops.py        19      0   100%
-----------------------------------
TOTAL                19      0   100%
5 passed in 0.54s
```