ESGI

4 decembre 2025

SOUTENANCE CI/CD

Pipeline complet & DevSecOps pour application Python

PHAM Tung Duong; NGUYEN Duy Anh; AMARA Bilal

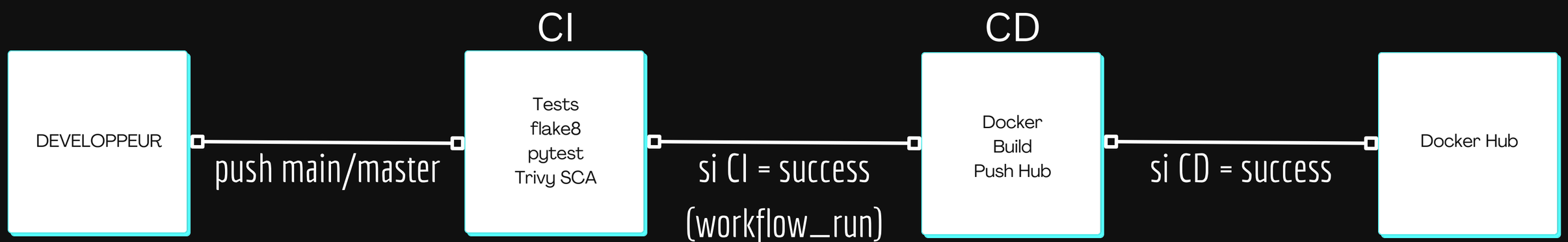
OBJECTIFS & CONTEXTE

- Construire un pipeline CI/CD complet avec GitHub Actions.

- Automatiser :
  - les tests multi-version (Python 3.8 / 3.9 / 3.10)
  - l’analyse de vulnérabilités avec Trivy
  - le build & push d’une image Docker vers Docker Hub
- Intégrer des bonnes pratiques de sécurité :
  - image Docker multi-stage
  - exécution non-root
  - scan de vulnérabilités dans la CI

Technologies : Python, GitHub Actions, Docker, Trivy, flake8, pytest.

ARCHITECTURE GLOBALE DU PIPELINE



CI - DÉCLENCHEMENT & PERMISSIONS

Points clés :

- Déclenchement automatique sur push
- Lancement manuel possible (workflow_dispatch)
- Principe du moindre privilège pour les permissions

.github > workflows > ci.yml

```yaml
name: CI

on:
  push:
    branches:
      - main
      - master
  workflow_dispatch:

permissions:
  contents: read
  security-events: write
  actions: read
```

CI : JOB TEST (MATRIX PYTHON)

```yaml
jobs:
  test:
    runs-on: ubuntu-latest
    strategy:
      matrix:
        python-version: ["3.8", "3.9", "3.10"]

    steps:
      - name: checkout
        uses: actions/checkout@v5

      - name: Python ${{ matrix.python-version }}
        uses: actions/setup-python@v6
        with:
          python-version: ${{ matrix.python-version }}

      - name: dependencies
        run: |
          python -m pip install --upgrade pip
          pip install flake8 pytest
```

CI : FLAKE8 & PYTEST

```yaml
36      - name: flake8
37        run: |
38          flake8 . --count --select=E9,F63,F7,F82 --show-source --statistics
39          flake8 . --count --exit-zero --statistics
40
41      - name: pytest
42        run: |
43          pytest tests/
```

RAPPELS :

- 1ère passe flake8 : bloquante, erreurs E9/F63/F7/F82
- 2e passe : non bloquante, statistiques
- pytest sur le dossier tests/

CI : JOB TRIVY-SCAN (SCA)

```yaml
trivy-scan:
  runs-on: ubuntu-latest

  steps:
    - name: checkout
      uses: actions/checkout@v5

    - name: trivy FS mode
      uses: aquasecurity/trivy-action@0.33.1
      with:
        scan-type: 'fs'
        scan-ref: '.'
        format: 'sarif'
        output: 'results.sarif'
        severity: 'CRITICAL,HIGH'

    - name: upload
      uses: github/codeql-action/upload-sarif@v4
      with:
        sarif_file: 'results.sarif'
```

TYPE D'ANALYSE :

- **Trivy en mode FS = Software Composition Analysis (scan des dépendances & système de fichiers)**

CD : DÉCLENCHEMENT & LOG-IN DOCKER

```yaml
name: CD

on:
  workflow_run:
    workflows: ["CI"]
    types:
      - completed

jobs:
  build:
    runs-on: ubuntu-latest
    if: ${{ github.event.workflow_run.conclusion == 'success' }}

    steps:
      - name: checkout
        uses: actions/checkout@v5

      - name: login
        uses: docker/login-action@v3
        with:
          username: ${{ secrets.DOCKER_USERNAME }}
          password: ${{ secrets.DOCKER_PASSWORD }}
```

CD : BUILD & PUSH DOCKER

```yaml
24      - name: build and push
25        id: push
26        uses: docker/build-push-action@v6
27        with:
28          context: .
29          file: ./Dockerfile
30          push: true
31          tags: ${{ secrets.DOCKER_USERNAME }}/calculator:latest
```

RÉSULTAT :

- Image username/calculator:latest publiée sur Docker Hub.

DOCKERFILE MULTI-STAGE & SÉCURITÉ

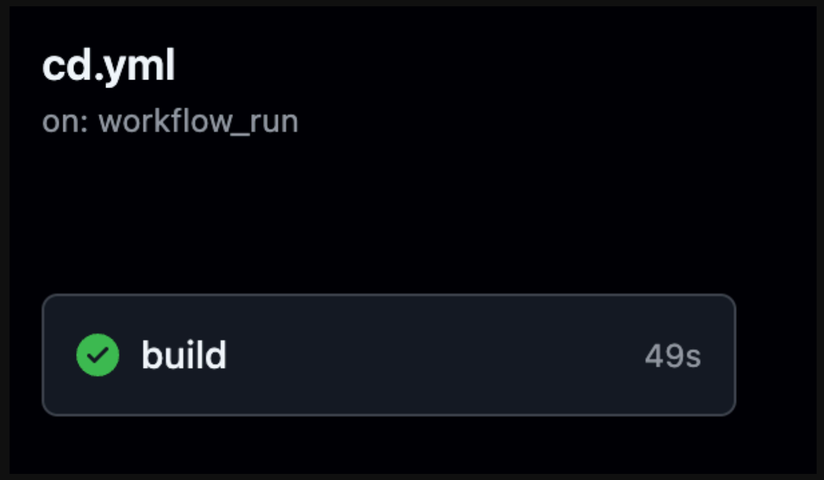
```dockerfile
FROM python:3.10-alpine AS builder

WORKDIR /app

RUN apk add --no-cache gcc python3-dev musl-dev linux-headers

COPY requirements.txt .

RUN python -m venv /opt/venv && \
    /opt/venv/bin/pip install --no-cache-dir --upgrade pip && \
    /opt/venv/bin/pip install --no-cache-dir -r requirements.txt

FROM python:3.10-alpine

RUN addgroup -S appgroup && \
    adduser -S appuser -G appgroup && \
    mkdir -p /app && \
    chown -R appuser:appgroup /app

WORKDIR /app

COPY --from=builder --chown=appuser:appgroup /opt/venv /opt/venv

COPY --chown=appuser:appgroup calculator/ ./calculator/

ENV PATH="/opt/venv/bin:$PATH" \
    PYTHONUNBUFFERED=1 \
    PYTHONDONTWRITEBYTECODE=1

USER appuser
```

HEALTHCHECK & DÉPENDANCES

```
32  HEALTHCHECK --interval=30s --timeout=3s --start-period=5s --retries=3 \
33      CMD python -c "from calculator.calculator import add; assert add(1,1) == 2" || exit 1
34
35  CMD ["python", "calculator/calculator.py"]
```

```
1   flake8==7.1.1
2   pytest==8.3.3
3   pytest-cov==5.0.0
4   safety==3.2.10
5   bandit==1.8.0
```

RÉSULTATS, SCREENSHOTS & CONCLUSION



RÉSULTATS DU PIPELINE :

- **CI → tests + lint + Trivy**
- **CD → build & push Docker**
- **Image calculator:latest → présente sur Docker Hub**
- **Rapports Trivy visibles dans GitHub Security**