

# Rapport – Laboratoire 0

---

Nom : Samuel RICHARD

Code permanent : RICS10049804

Cours : LOG430 – Architecture Logicielle Session Été 2025

Groupe : 01

## Informations générales

Nom Github : Scirelgar

URL du dépôt GitHub/GitLab : <https://github.com/Scirelgar/LOG430-Laboratoires/>

URL du dépôt Lab0 : <https://github.com/Scirelgar/LOG430-Lab0.git>

## Auto-évaluation par section




### Structuration du dépôt

Éléments	Oui	Non	Commentaire / Justification
Le fichier README.md est présent et complet	<input checked="" type="checkbox"/>		
Un fichier .gitignore pertinent est utilisé	<input checked="" type="checkbox"/>		J'ai utilisé le template associé à python sur GitHub
Le dépôt présente une structure claire et logique	<input checked="" type="checkbox"/>		La structure suit les bonnes pratiques de projets python, avec le choix de faire les tests à l'extérieur du code de l'application



### Application minimale

Éléments	Oui	Non	Commentaire / Justification
L'application fonctionne localement	<input checked="" type="checkbox"/>		
L'application affiche un message ou répond sur un port web	<input checked="" type="checkbox"/>		En utilisant le serveur de développement de FastAPI



## Tests unitaires

Éléments	Oui	Non	Commentaire / Justification
Au moins deux tests unitaires sont présents			
Les tests passent avec succès			
Le framework de test est bien choisi et configuré			Utilisation de <code>pytest</code>







## Conteneurisation

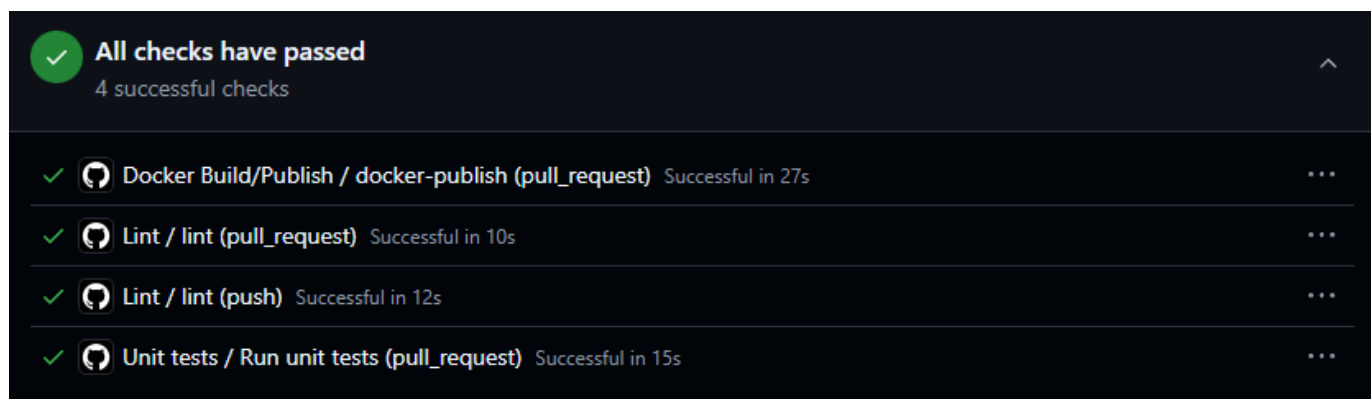
Éléments	Oui	Non	Commentaire / Justification
L'application est conteneurisée avec un Dockerfile			
L'image Docker fonctionne localement			
Un fichier .dockerignore est présent (si pertinent)			Pour ignorer les fichiers pycache

## Orchestration avec Docker Compose




Éléments	Oui	Non	Commentaire / Justification
Un fichier docker-compose.yml permet de lancer l'application			
Les instructions sont claires pour lancer l'application			Le README.md décrit deux méthodes d'installation et d'exécution

## Intégration continue (CI/CD)

Éléments	Oui	Non	Commentaire / Justification
Une pipeline CI/CD est configurée sur GitHub Actions ou GitLab CI			
L'étape lint fonctionne correctement			
L'étape test unitaire fonctionne correctement			
L'étape build Docker fonctionne correctement			
L'image est automatiquement poussée sur Docker Hub			L'utilisation de l'action <a href="#">docker/build-push-action@v6</a> combine les deux étapes de build et de publication.
Une preuve d'exécution réussie de la pipeline est incluse			Voir capture ci-après



## Documentation

Éléments	Oui	Non	Commentaire / Justification
Le README.md décrit le projet, les instructions de build et d'exécution			
Le fonctionnement de la CI/CD y est expliqué			
Les choix techniques sont justifiés			Ci-dessous quelques références utilisées pour la structure de projet.

## Références

- [Tests outside application code](#)
- [Testing FastAPI](#)

## Réflexion personnelle

### Qu'avez-vous appris dans ce laboratoire ?

Ayant déjà déployé des CI/CD en python auparavant, créer l'application de base et répondre aux exigences du laboratoire fût assez rapide. L'utilisation de Docker pour créer des images était nouveau pour moi et j'ai dû beaucoup parcourir la documentation pour comprendre comment construire une image convenablement. Il se trouve que simplement utiliser une image ne donne pas nécessairement une bonne intuition pour en construire une.

### Quels éléments vous ont posé le plus de difficulté ?

Écrire le fichier de configuration `compose.yaml`. Encore à l'heure actuelle je ne suis pas certain d'avoir bien saisi l'utilisation de ce fichier si tout ce qu'il accomplit peut déjà être fait en CLI avec un Dockerfile.

### Que souhaitez-vous améliorer pour les prochains laboratoires ?

Le fichier `compose.yaml`. Il me semble que je pourrais encore mieux exploiter ce fichier dans le cadre d'un build, publication et même exécution.