Grille de correction détaillée – EC06

Informations générales

- **Épreuve :** EC06 – CI/CD et versionning
- **Durée :** 4h – Individuelle
- **Projet fil rouge :** FoodSafe – Intégration et déploiement continu
- **Technologies imposées :** Git + outil CI/CD (GitHub Actions, GitLab CI, Jenkins, etc.) + Docker

Critères d'évaluation et attentes

Critère	Intitulé officiel	Attentes côté correcteur	Points de vigilance
C7.3	Gestion Git	Dépôt structuré avec branches (`main` , `dev` , `feature/...`). Historique clair, commits explicites et cohérents.	Vérifier cohérence du workflow (Git Flow, trunk-based). Attention aux dépôts plats avec tout en `main` .
C9.4	Pipeline CI/CD	Fichier de configuration présent (`.gitlab-ci.yml` , `.github/workflows/ci.yml` , etc.). Doit inclure étapes install, lint, tests, build, éventuellement déploiement.	Vérifier exécution réelle (captures ou logs). Pipeline trop basique = perte de points.
C9.5	Déploiement sécurisé	Présence d'un `Dockerfile` fonctionnel et optimisé (multi-stage si possible). Script `deploy.sh` ou équivalent.	Vérifier que les secrets ne sont pas exposés dans le dépôt.

Critère	Intitulé officiel	Attentes côté correcteur	Points de vigilance
		Gestion sécurisée des variables d'environnement.	`.env` versionné = faute grave.

Livrables à corriger

- **Dépôt Git** : branches organisées, historique exploitable
- **Pipeline CI/CD** : configuration complète et fonctionnelle
- **Dockerfile** : image reproductible et optimisée
- **Scripts de déploiement** : automatisation claire et sécurisée
- **README.md** : documentation du processus complet (CI, Docker, déploiement)

Barème indicatif (sur 20 points)

Axe évalué	Points
Structuration Git (C7.3)	/6
Pipeline CI/CD (C9.4)	/7
Déploiement sécurisé (C9.5)	/7

> **Tolérance :** ±2 points selon la propreté du code, la documentation et la complétude des livrables.

Points positifs attendus

- Stratégie Git claire (workflow respecté, branches bien utilisées)
- Pipeline CI/CD complet : install → lint → tests → build → deploy
- Dockerfile multistage ou image légère optimisée pour la prod
- Gestion sécurisée des variables d'environnement via le pipeline

- Script de déploiement automatisé et fonctionnel (`deploy.sh` , `run.sh`)
- README clair expliquant le processus CI/CD avec exemples de commandes

Erreurs fréquentes à surveiller

- Commits peu clairs ou workflow Git inexistant
- Pipeline CI/CD incomplet (par ex. uniquement un build sans tests)
- Dockerfile trop basique ou inutilisable en production
- Secrets exposés dans le code ou versionnés dans `.env`
- Script de déploiement absent, incomplet ou inutilisable
- README minimal, sans explications pratiques

Rappel pédagogique

Cette épreuve doit confirmer que l'apprenant maîtrise :

- Le **versionning collaboratif** via Git et l'organisation professionnelle d'un dépôt
- L'**automatisation des builds, tests et déploiements** avec un pipeline CI/CD
- L'usage de **conteneurs Docker** pour garantir portabilité et reproductibilité
- La **sécurisation des déploiements** (gestion des secrets, variables d'environnement)

Elle valide la capacité à industrialiser un projet sensible comme FoodSafe, où la fiabilité, la traçabilité et la continuité de service sont essentielles.