Grille de correction détaillée – EC06

Informations générales

- **Épreuve :** EC06 – CI/CD et versionning
- **Durée :** 4h – Individuelle
- **Projet fil rouge :** SkillHub – 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, messages de commit lisibles.	Vérifier que l'arborescence Git est cohérente (pas de tout en `main`). Mauvais messages de commit = perte de points.
C9.4	Pipeline CI/CD	Fichier de pipeline présent (`.gitlab-ci.yml` , `.github/workflows/ci.yml` ...), intégrant au minimum : build, lint/tests, éventuellement déploiement.	Vérifier exécution réelle (captures de builds ou logs). Pipeline trop minimal ou incomplet = points en moins.
C9.5	Déploiement sécurisé	Présence d'un `Dockerfile` fonctionnel, script de déploiement (`deploy.sh` ou équivalent), gestion	Vérifier que les secrets ne sont pas en clair dans

Critère	Intitulé officiel	Attentes côté correcteur	Points de vigilance
		des variables d'environnement sécurisée.	le dépôt. Attention aux `.env` versionnés par erreur.

Livrables à corriger

- **Dépôt Git :** organisation des branches, historique des commits
- **Pipeline CI/CD :** configuration complète et fonctionnelle
- **Dockerfile :** image reproductible et optimisée
- **Scripts de déploiement :** automatisation avec gestion des variables d'environnement
- **README.md :** description du processus CI/CD, instructions d'utilisation

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 :** ±1 point par critère en fonction de la propreté du code et de la complétude des livrables.

Points positifs attendus

- Branches Git clairement définies et utilisées (workflow explicite)
- Pipeline CI/CD exécutant les étapes essentielles (lint, tests, build, deploy)

- Dockerfile multistage ou optimisé (taille réduite, bonne reproductibilité)
- Scripts de déploiement clairs et sécurisés (variables d'environnement gérées correctement)
- README explicatif et opérationnel, permettant de rejouer le processus

Erreurs fréquentes à surveiller

- Absence de stratégie Git (tout en `main`, historique brouillon)
- Pipeline CI/CD trop basique ou inexistant
- Dockerfile incomplet ou inutilisable en production
- Secrets exposés dans le dépôt (mots de passe en clair dans `.env`)
- README incomplet ou inutilisable (aucune indication sur la CI/CD mise en place)

Rappel pédagogique

Cette épreuve doit confirmer que l'apprenant maîtrise :

- Le **versionning collaboratif** avec Git et l'organisation d'un dépôt professionnel
- L'**automatisation** de la qualité (lint, tests, builds) et des déploiements via CI/CD
- L'usage de **conteneurs Docker** pour garantir la portabilité et la reproductibilité
- Les bonnes pratiques de **sécurisation** des secrets et variables d'environnement

Elle constitue un jalon essentiel pour valider les compétences DevOps, en préparation aux futures étapes du projet (Cloud et exploitation en production).