# Grille de correction détaillée – EC09

## Informations générales

- **Épreuve :** EC09 – Guide de bonnes pratiques
- **Durée :** 4h – Individuelle
- **Projet fil rouge :** RebootCamp – Guide qualité et éco-conception
- **Nature :** Épreuve écrite (guide structuré et exemples)

## Critères d'évaluation et attentes

| Critère | Intitulé officiel | Attentes côté correcteur | Points de vigilance |
|---|---|---|---|
| **C8.6** | Conventions de code et documentation | Guide structuré et concret : normes de nommage, organisation du code, conventions Git, documentation minimale. | Vérifier que les règles soient applicables, pas juste théoriques. |
| **C9.6** | Stratégie de tests | Présentation claire des types de tests (unitaires, intégration, E2E), outils proposés, plan de couverture. | Attention aux guides qui citent des tests sans donner de méthodologie concrète. |
| **C14.3** | Qualité logicielle globale | Approche réaliste pour assurer un code maintenable, sécurisé et sobre. Intégration des pratiques d'éco-conception. | Surveiller l'absence de volet éco-conception (critique pour RebootCamp). |

## Livrables à corriger

- **Guide de bonnes pratiques** (format `.md` ou `.pdf`)
- **Exemples de conventions** (Git, code, structure projet)
- **Plan de test et couverture** (tableaux, outils, méthodo)
- **Annexes éventuelles** (`exemples/`, README explicatif)

# Barème indicatif (sur 20 points)

| Axe évalué | Points |
|---|---|
| Conventions de code + documentation (C8.6) | /7 |
| Stratégie de tests (C9.6) | /6 |
| Qualité globale + éco-conception (C14.3) | /7 |

> **Tolérance :** ±2 points selon la précision, la structuration et l'applicabilité des recommandations.

# Points positifs attendus

- Conventions claires : nommage (`camelCase`, `PascalCase`, `snake_case`), structure des dossiers, organisation des modules
- Normes Git bien définies : branches (`feat/`, `fix/`, `hotfix/`, `release/`), pull requests, politique de merge
- Checklist de revue de code : lisibilité, duplications, sécurité (XSS, injections), performance, accessibilité
- Stratégie de tests : tableau avec niveaux (unitaires, intégration, E2E) et outils (Jest, PHPUnit, Cypress…)
- Documentation minimale obligatoire : `README.md`, commentaires ciblés sur modules critiques, schémas d'architecture
- Pratiques d'éco-conception intégrées : lazy loading, suppression du code mort, compression d'images, dark mode, CI/CD sobre
- Références à des outils de monitoring : EcoIndex, Lighthouse, GreenIT Analysis

# Erreurs fréquentes à surveiller

- Guide trop théorique ou incomplet (copie de standards génériques sans contextualisation)
- Absence d'exemples concrets (code, Git, tests) rendant le guide inutilisable en équipe
- Oubli de la dimension **éco-conception** (critique pour RebootCamp)
- Manque de structuration (texte brut sans titres, tableaux ni checklist)
- Plan de tests absent, trop flou ou non relié à des outils précis

# Rappel pédagogique

Cette épreuve valide la capacité à :

- **Structurer et formaliser** des règles concrètes de qualité logicielle
- Mettre en œuvre une **stratégie de tests réaliste et mesurable**
- Intégrer les **pratiques d'éco-conception** au sein d'un projet numérique durable
- Valoriser la **documentation technique** comme outil d'onboarding et de collaboration

Elle donne aux apprenants un socle de pratiques pour travailler efficacement en équipe et maintenir un **niveau élevé de qualité et de durabilité** dans leurs productions logicielles.