Grille de correction détaillée – EC04

Informations générales

- **Épreuve :** EC04 – API sécurisée et documentée
- **Durée :** 4h – Individuelle
- **Projet fil rouge :** FoodSafe – API publique (produits, profil, favoris)
- **Technologies imposées :** REST ou GraphQL, sécurisation obligatoire (JWT)

Critères d'évaluation et attentes

Critère	Intitulé officiel	Attentes côté correcteur	Points de vigilance
C8.3	API sécurisée	API dispose d'un système d'authentification robuste (JWT ou équivalent), middlewares pour autorisation, endpoints protégés. Organisation claire du code (contrôleurs, services, middlewares).	Vérifier qu'un endpoint protégé renvoie bien `401` sans token et `403` avec token invalide. Attention à la gestion des données sensibles (profils alimentaires).
C8.4	Documentation API	Présence d'un fichier Swagger/OpenAPI (`openapi.yaml` ou équivalent). Toutes les routes doivent être décrites (params, réponses, codes HTTP).	Vérifier que la doc est exploitable via Swagger UI ou Postman. Doc absente ou partielle = gros manque.
C8.5	Tests unitaires	Tests automatisés couvrant les endpoints critiques (authentification, recherche	Vérifier que les tests passent réellement. Couverture faible ou

Critère	Intitulé officiel	Attentes côté correcteur	Points de vigilance
		produit, compatibilité, favoris). Rapport de couverture fourni.	superficielle = perte de points.

Livrables à corriger

- **Code source** : projet API (`src/`), structuré et sécurisé
- **README.md** : instructions de test, exemples de requêtes (`curl` , Postman), gestion des tokens
- **Documentation** : fichier Swagger/OpenAPI complet
- **Tests** : code de tests + rapport de couverture

Barème indicatif (sur 20 points)

Axe évalué	Points
API sécurisée (C8.3)	/8
Documentation API (C8.4)	/6
Tests unitaires sur endpoints (C8.5)	/6

> **Tolérance :** ±1 point par critère selon la lisibilité du code, la propreté des tests et la complétude de la documentation.

Points positifs attendus

- Endpoints bien définis et cohérents (`/users` , `/products` , `/favorites`)
- Authentification JWT fonctionnelle avec tokens expirables
- Gestion claire des erreurs (`401` , `403` , `422` , `500`)
- Documentation Swagger/OpenAPI lisible et exploitable immédiatement
- Tests unitaires couvrant les cas critiques avec un bon taux de couverture

- README contenant des exemples concrets de requêtes et instructions de lancement

Erreurs fréquentes à surveiller

- Endpoints sensibles accessibles sans authentification
- JWT mal implémenté (expiration ignorée, token valide pour tous les utilisateurs)
- Documentation Swagger absente, incomplète ou obsolète
- Codes HTTP incohérents (`200` renvoyé en cas d'erreur, `500` récurrent)
- Tests unitaires superficiels ou non fonctionnels
- README minimal, sans explication sur la gestion des tokens

Rappel pédagogique

Cette épreuve doit confirmer que l'apprenant sait :

- Concevoir une **API sécurisée et maintenable**
- Rédiger une **documentation technique complète et exploitable**
- Automatiser les **tests des endpoints critiques**
- Respecter les contraintes de **protection des données sensibles** (santé, intolérances alimentaires)

Elle prépare l'intégration du front-end (EC02) avec le back-end sécurisé, ainsi que le déploiement cloud (EC05–EC06).