**Review code java**

**Remarque** :

1. le principe d'abstraction n'est pas respecter donc on doit séparer la définition du logique métier de l'implémentation réelle du service en utilisant les interfaces.

Pour cela il faut créer deux interfaces suivantes :

* ProductService  qui contient :

Product save(Product product);

Product update(Product product,Long id);

List<Product> findAll();

Product find(Long id);

**void** delete(Long id);

* InventoryService qui contient :

Inventory getProductsInventory();

Par la suite ces deux dernières seront implémentées par ProductServiceImpl et par la suite nous les injectons dans ProductController.

1. il est généralement recommandé d'exposer les données sous forme de DTO (Data Transfer Objects)- en utilisant model mapper plutôt que d'exposer directement les entités de la base de données.
2. Fusionner @RequestBody et @Controller en @RestController
3. La méthode findAll peut être simplifiée en utilisant simplement la méthode toList() de Java Stream pour convertir l'objet Iterable renvoyé par la méthode findAll en une liste d'objets Product.
4. Ajouter des petits changements dans la méthode getProductsInventory afin d’éviter l’exception nullpointerexception
5. Pour que le code soit plus clair, on peut renommer l’InventoryItem en ProductInventory , productBarcodes pourrait être renommé en productBarcodeList et C par getProductsInventory
6. Ajouter la vérification de l’existence du produit dans la base de donnée avant sa suppression en levant une exception NotFound() si c’est le cas
7. Documenter les ends-points, de préférence en utilisant le freamwork swager