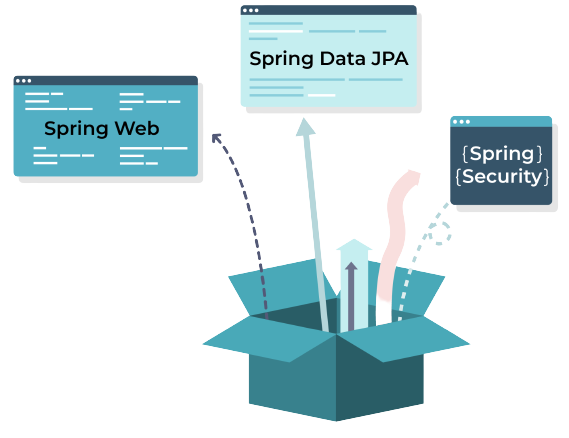
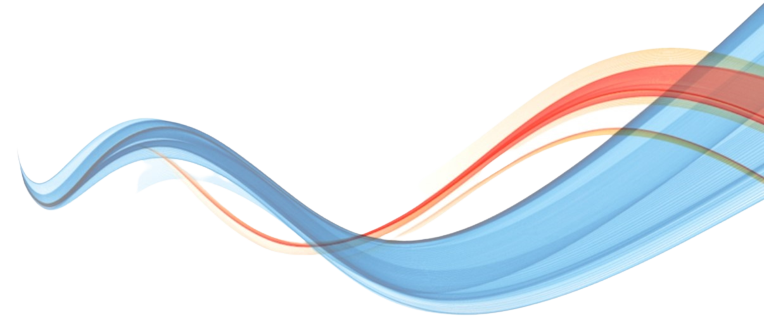
Compte rendu de :

SEANCE 09



**Réalisé Par :** AYOUJJIL Soukayna

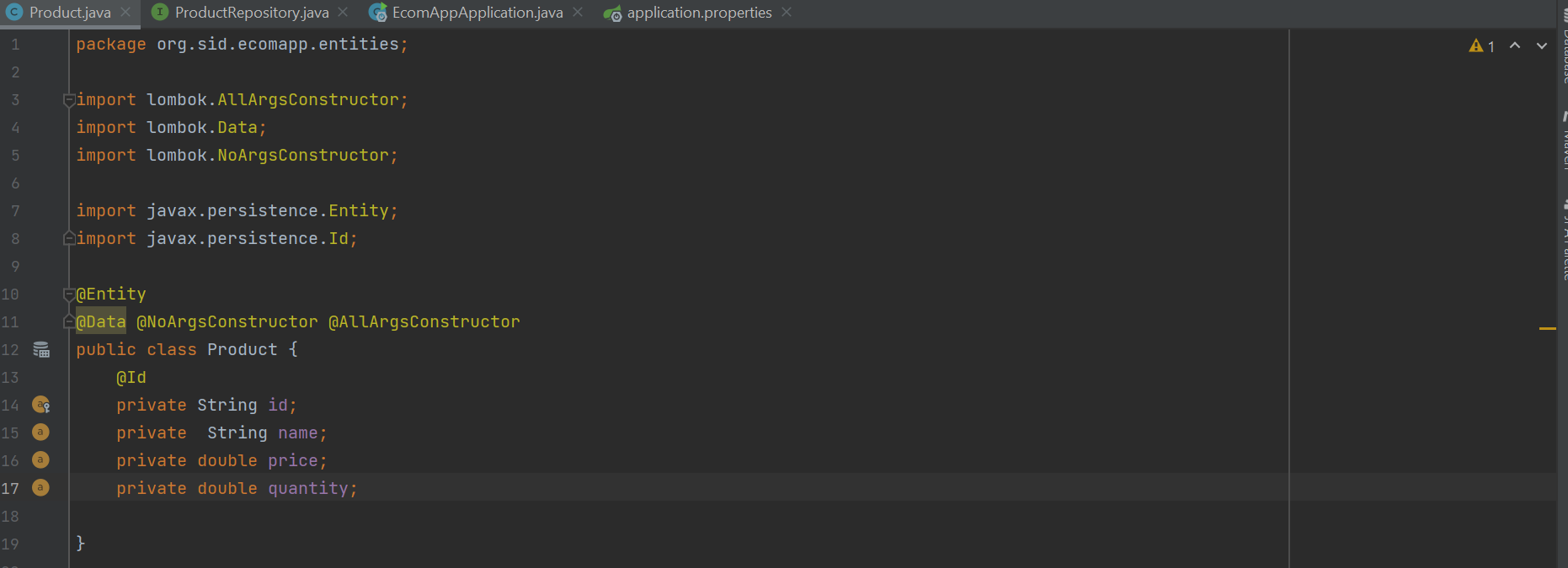
**Département  :** Mathématiques et Informatique

**Filière :** II-BDCC

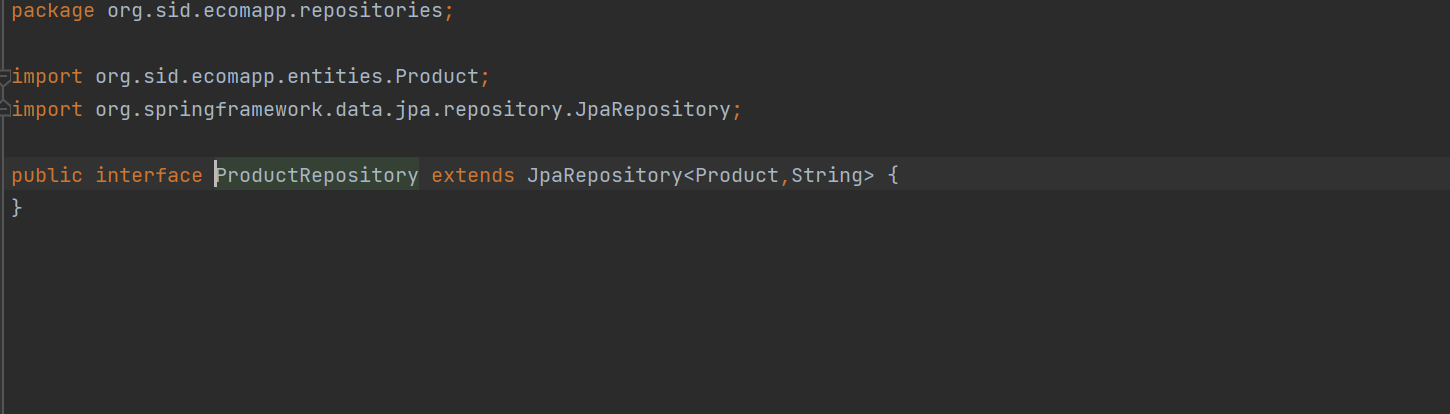
**Professeur :** M. Mohamed YOUSSFI

**Année Universitaire** **: 2021/2022**

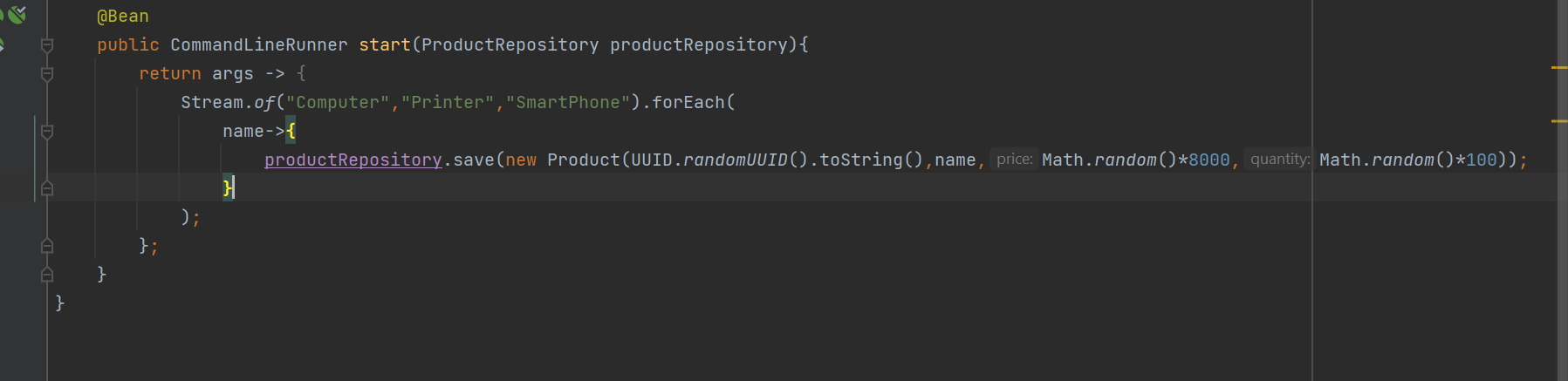
Etape 1 : Création de l’entité Product



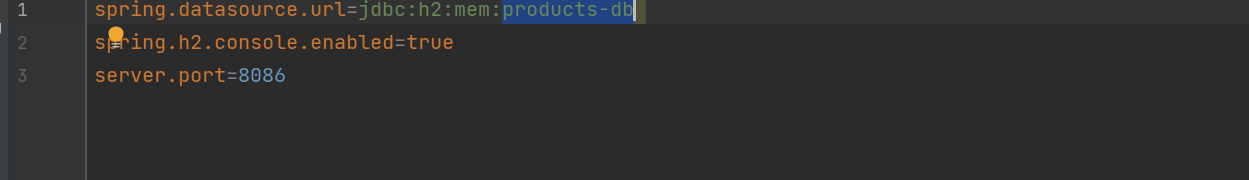
Etape 2 : Création de l’interface ProductRepository



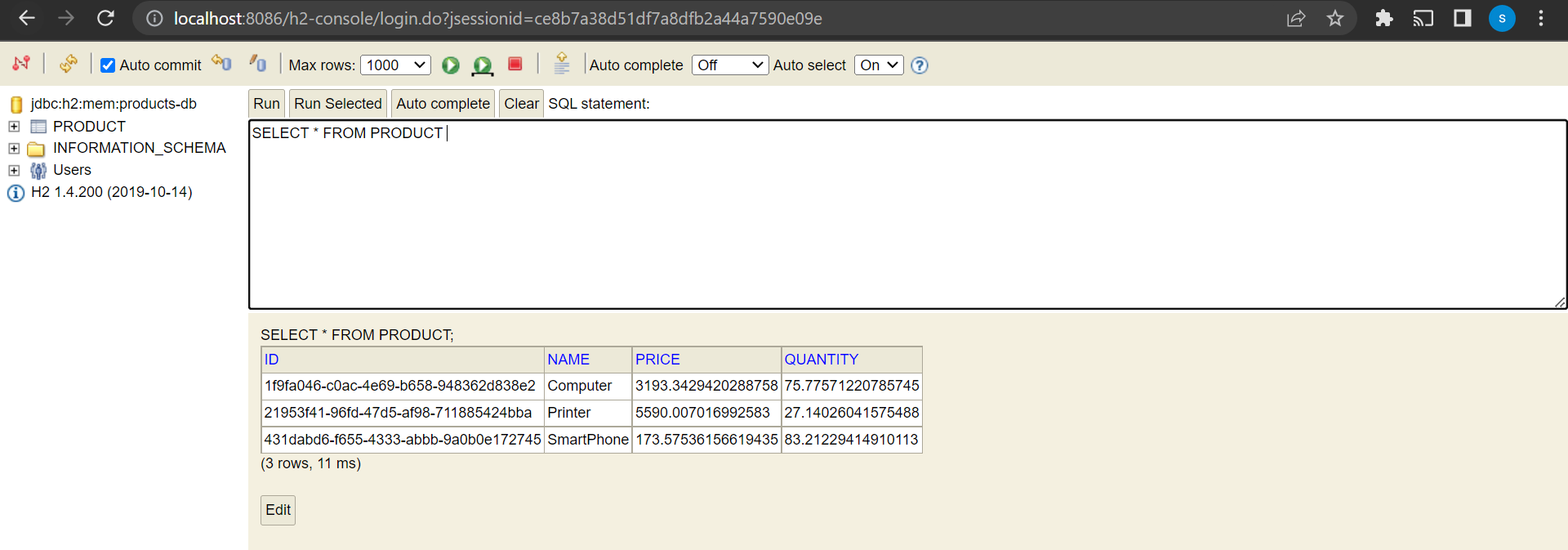
Etape 3 : Génération d’une liste des produits



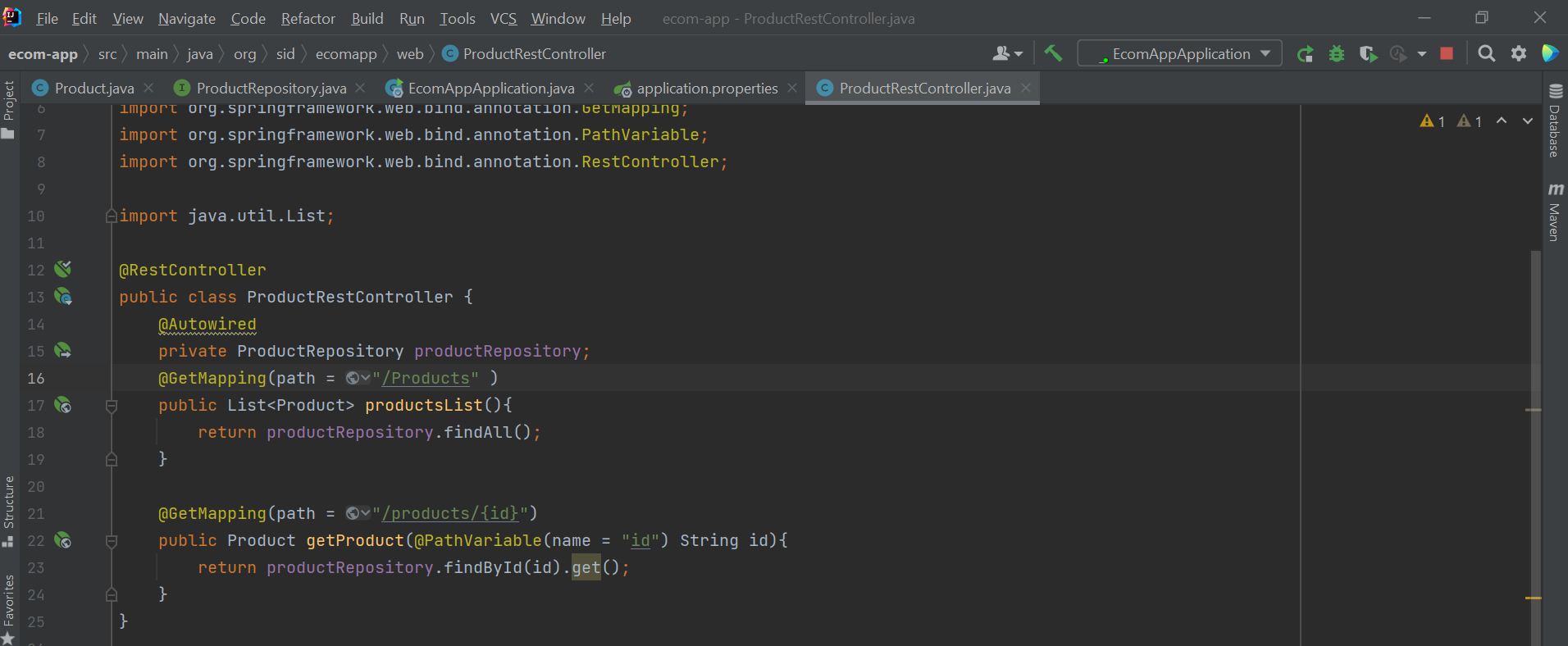
Etape 4 : Compléter le fichier application.properties



Etape 5 : Démarrer l’application sur le port 8086



Etape 6 : créer un Rest Controller

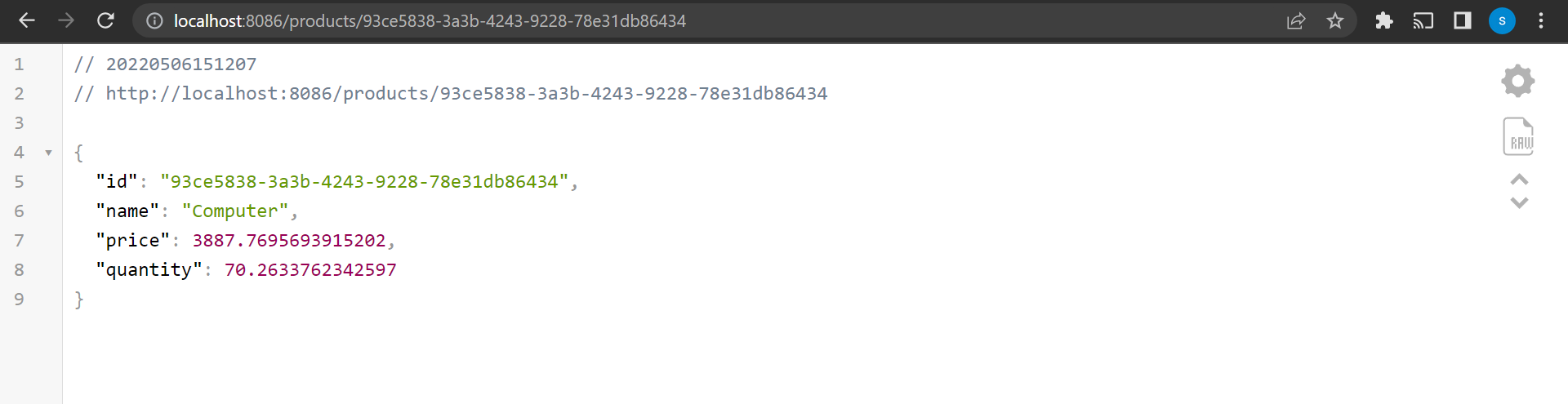


Etape 7 : Resultats

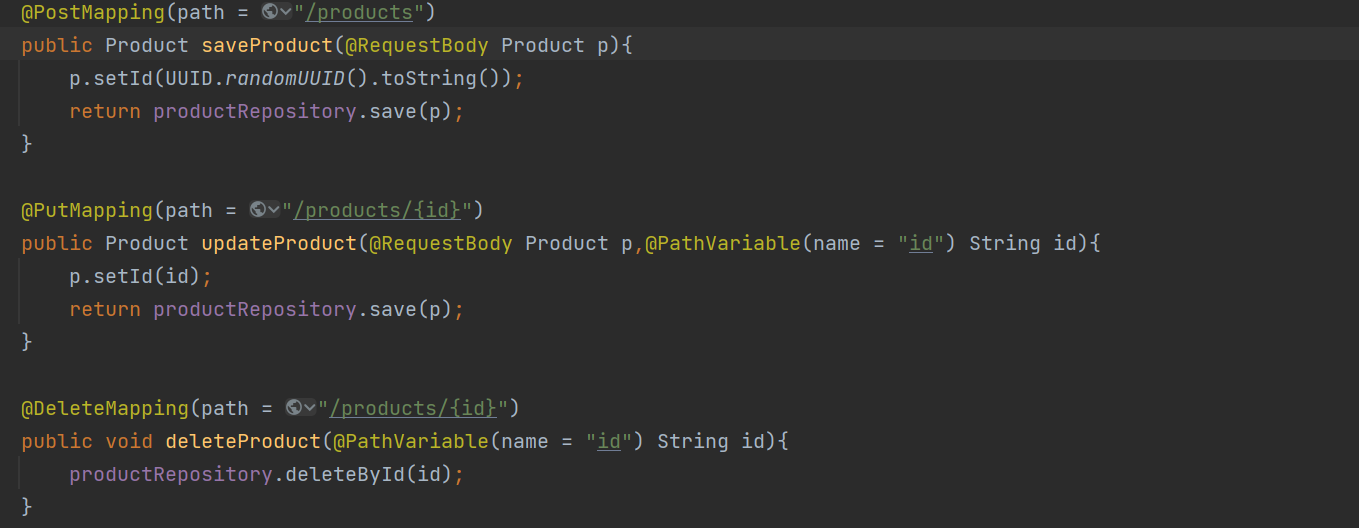
* Consulter Liste des produits



* Consulter un produit a l’aide de son id

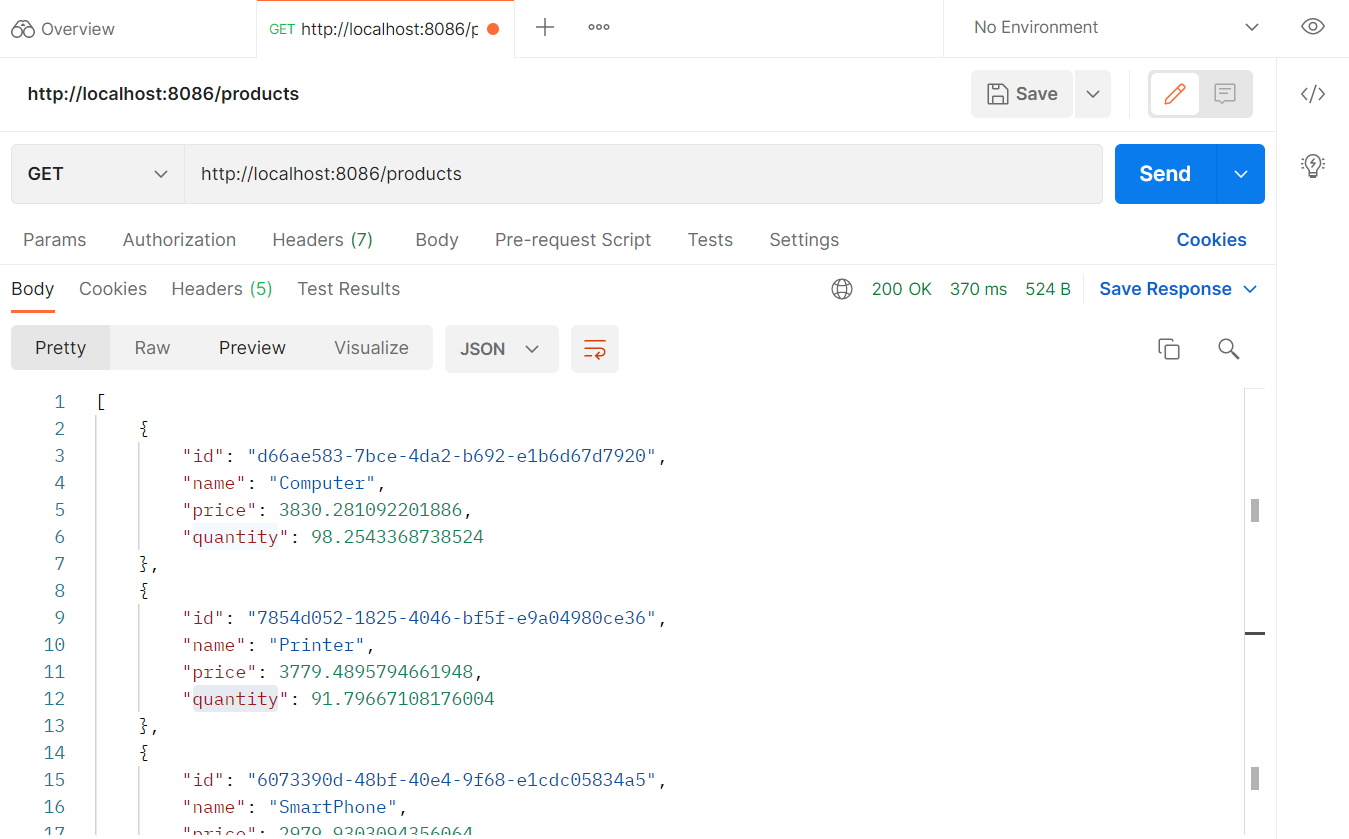


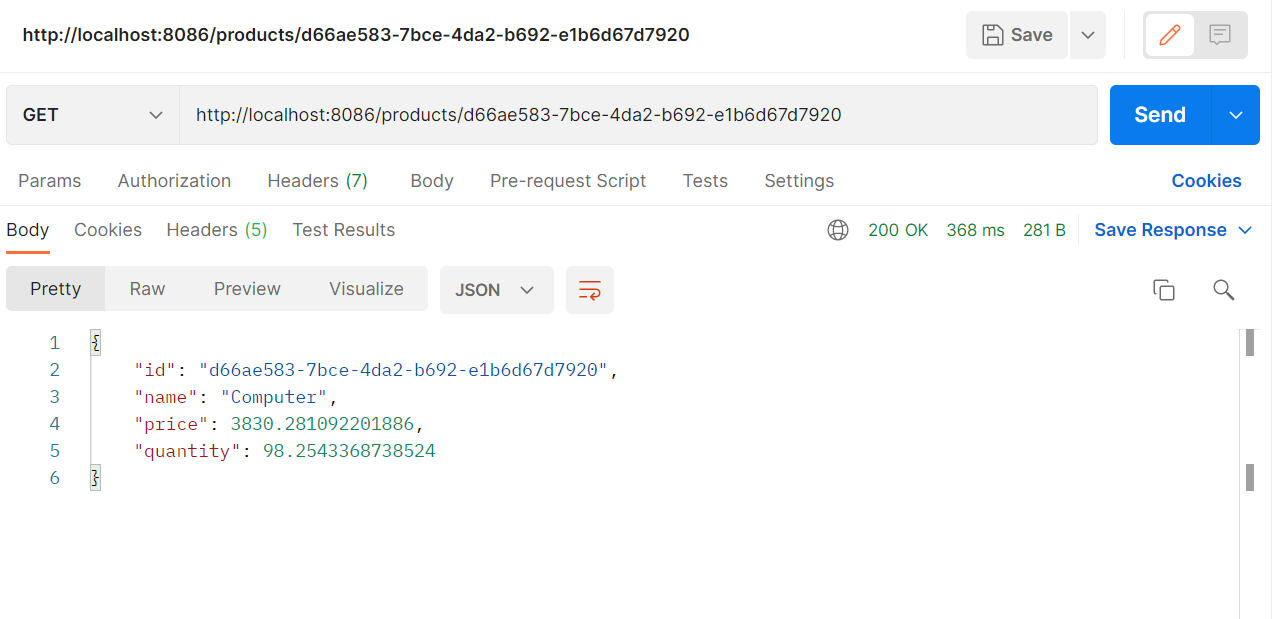
Etape 7 : Création des méthodes ajouter, modifier et supprimer un produit



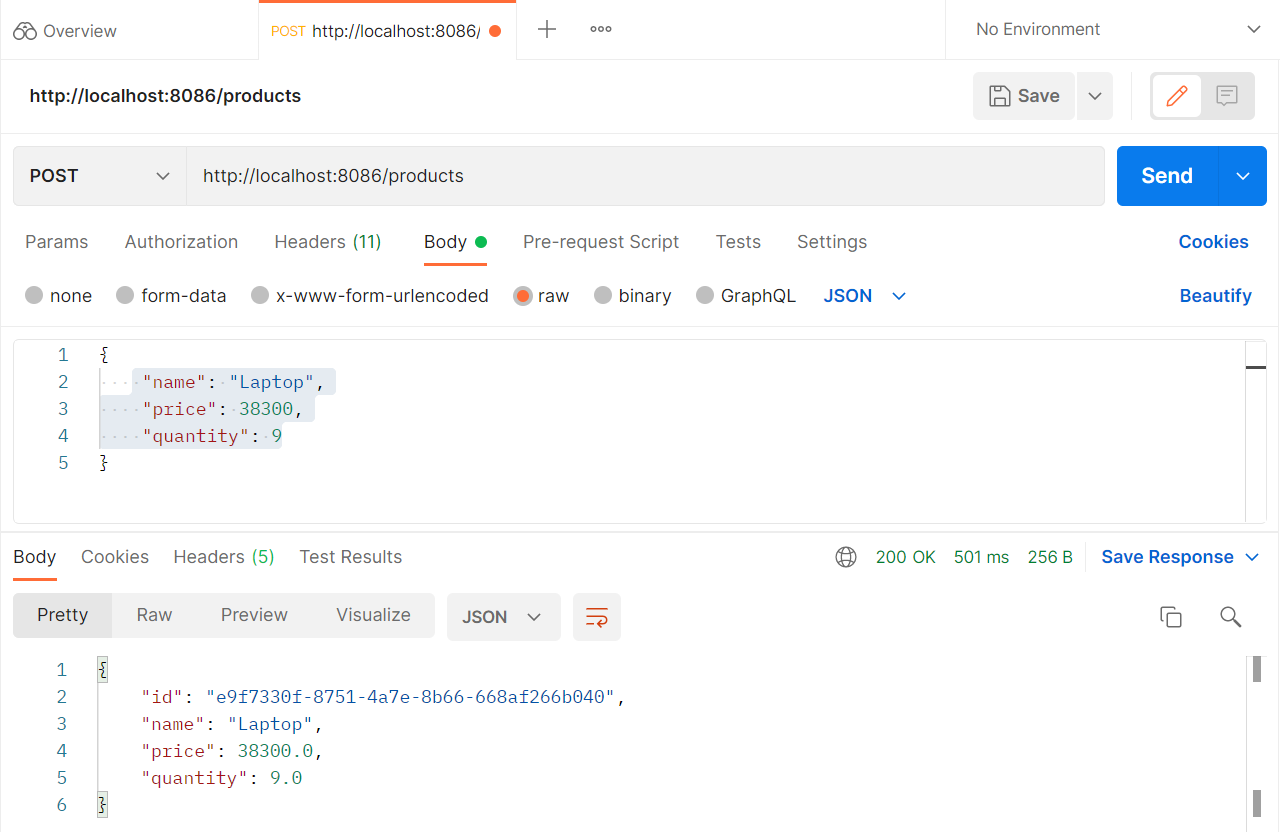
Etape 8 : Resultats

* Consulter Liste des produits / un seul produit

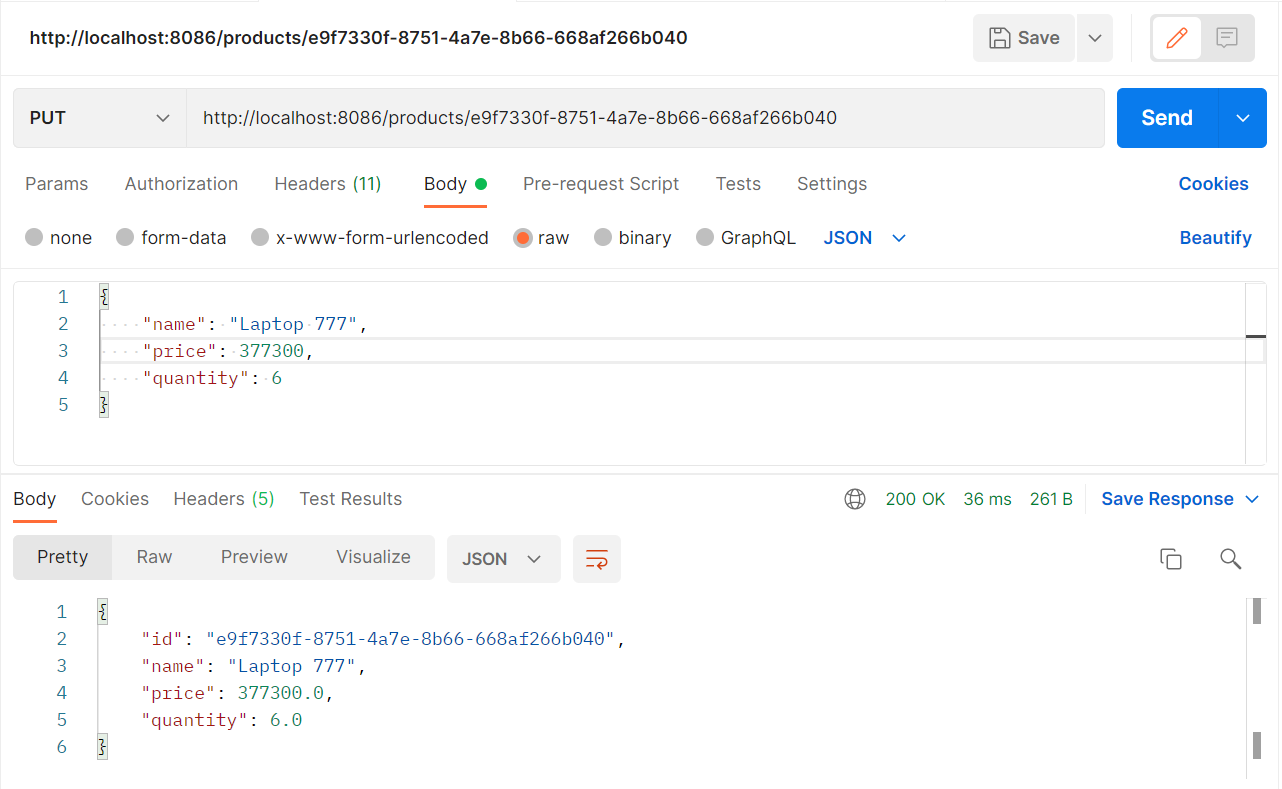




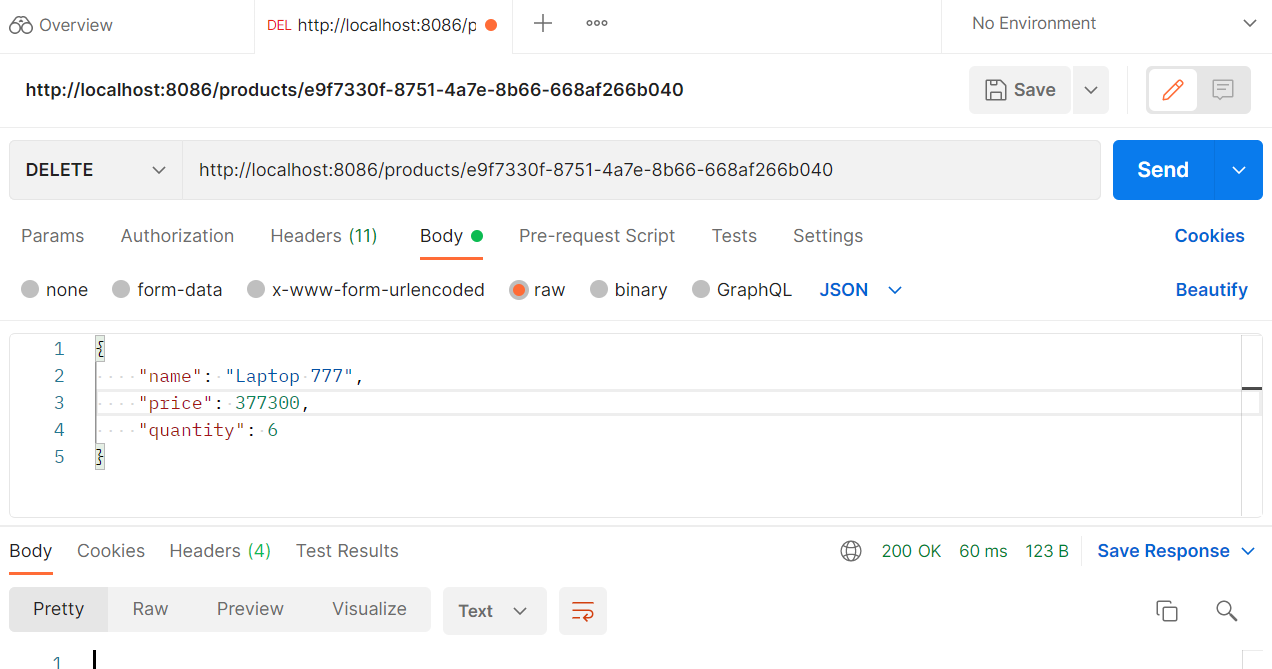
* Ajouter un produit



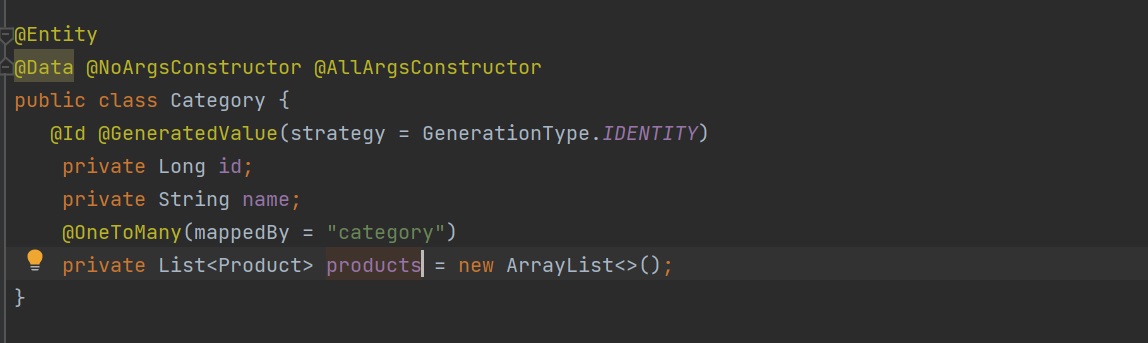
* Modifier un produit



* Supprimer un produit



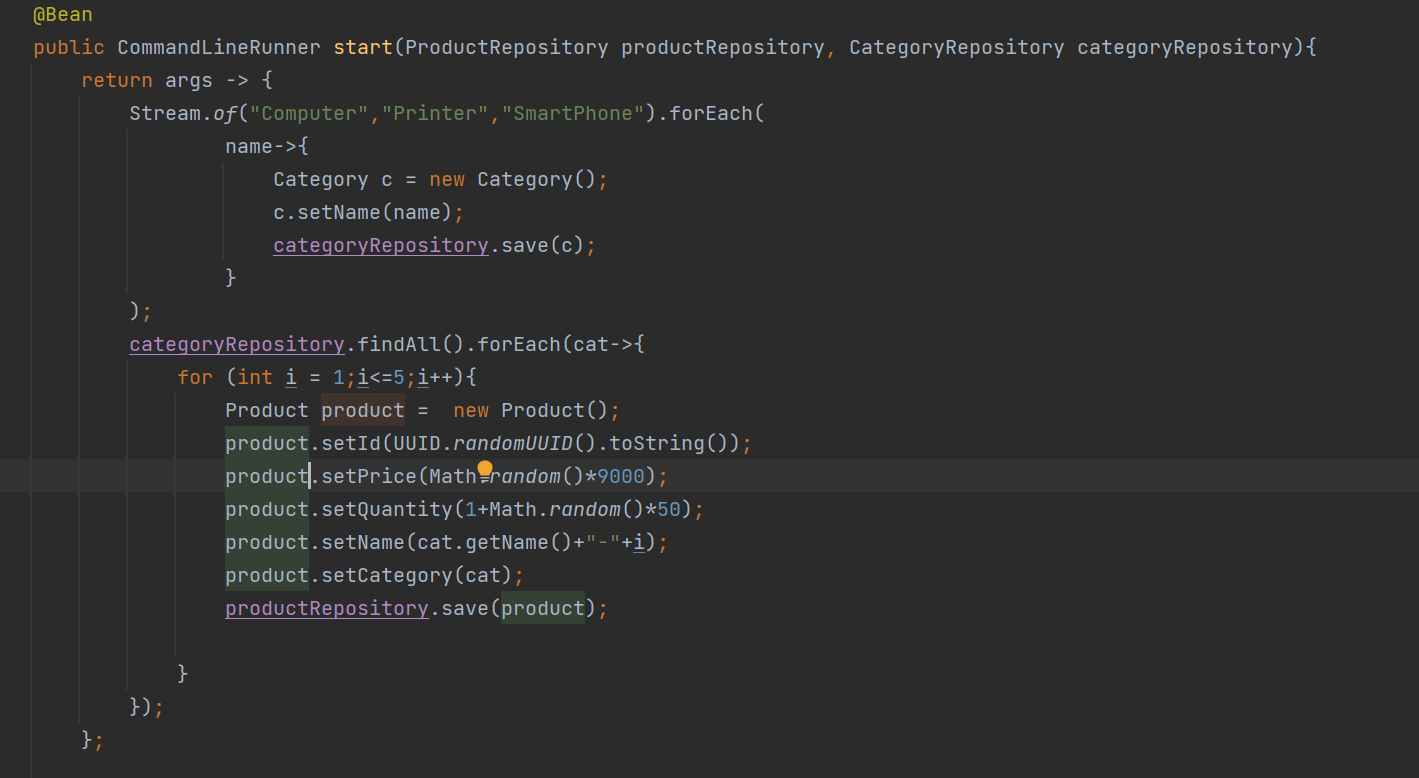
Etape 9 : Création de l’entité Category



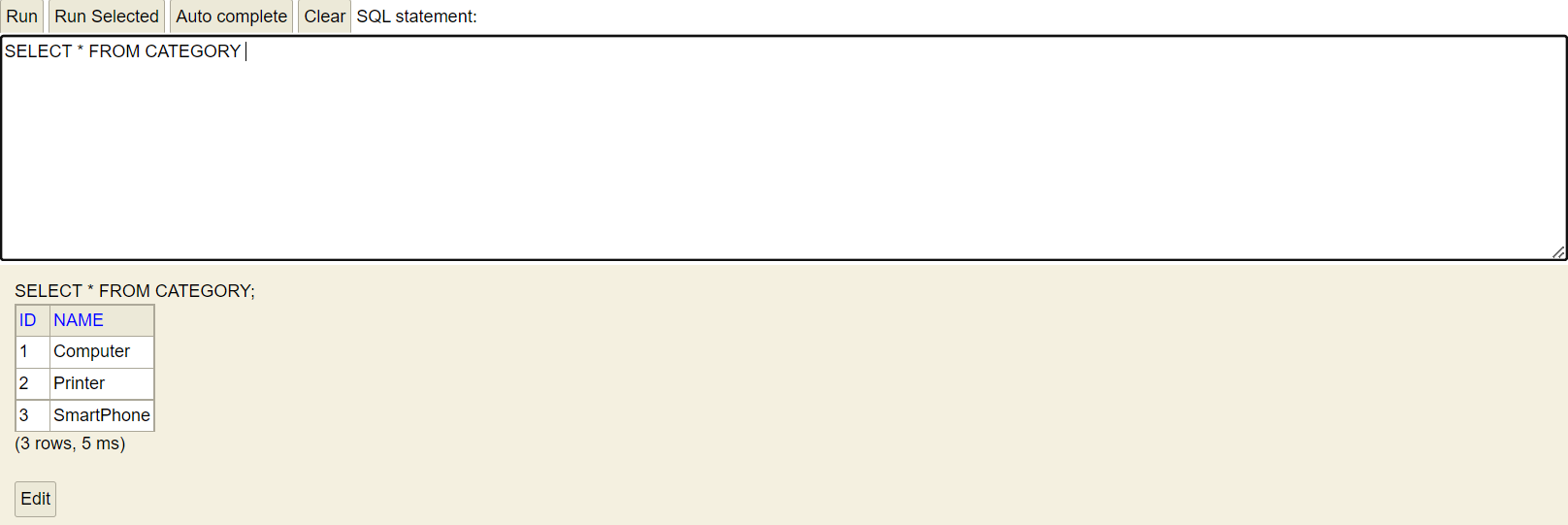
Etape 10 : Création du repository CategoryRespository

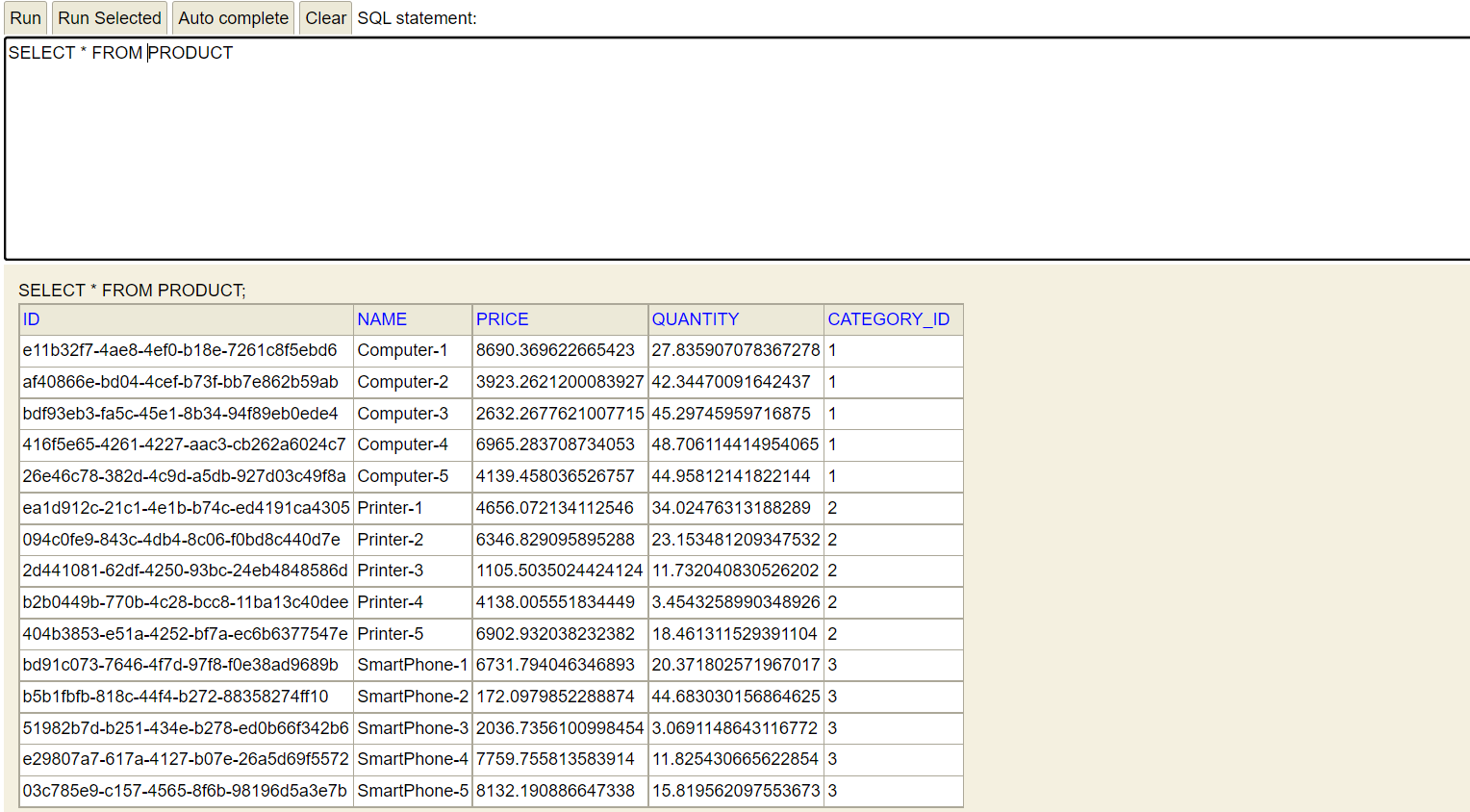


Etape 11 : Création des produits et categories



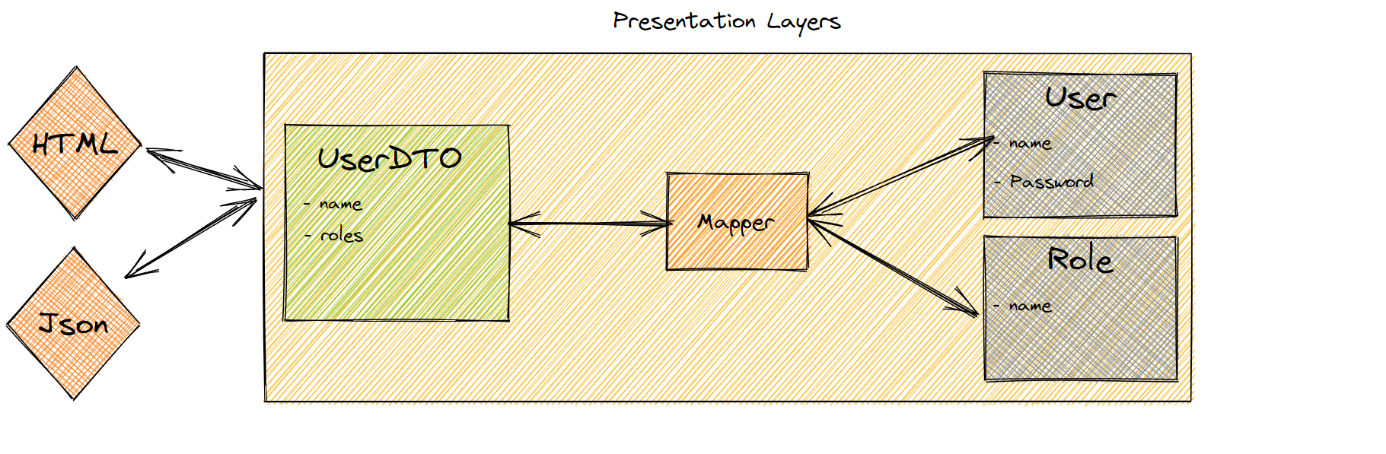
Etape 12 : Resultats





Etape 13 : Création du modèle DTO (objet de transfert de données)

* **Comment l'utiliser ?**



<https://www.baeldung.com/java-dto-pattern#how-to-use-it>

* **Quand l'utiliser :**

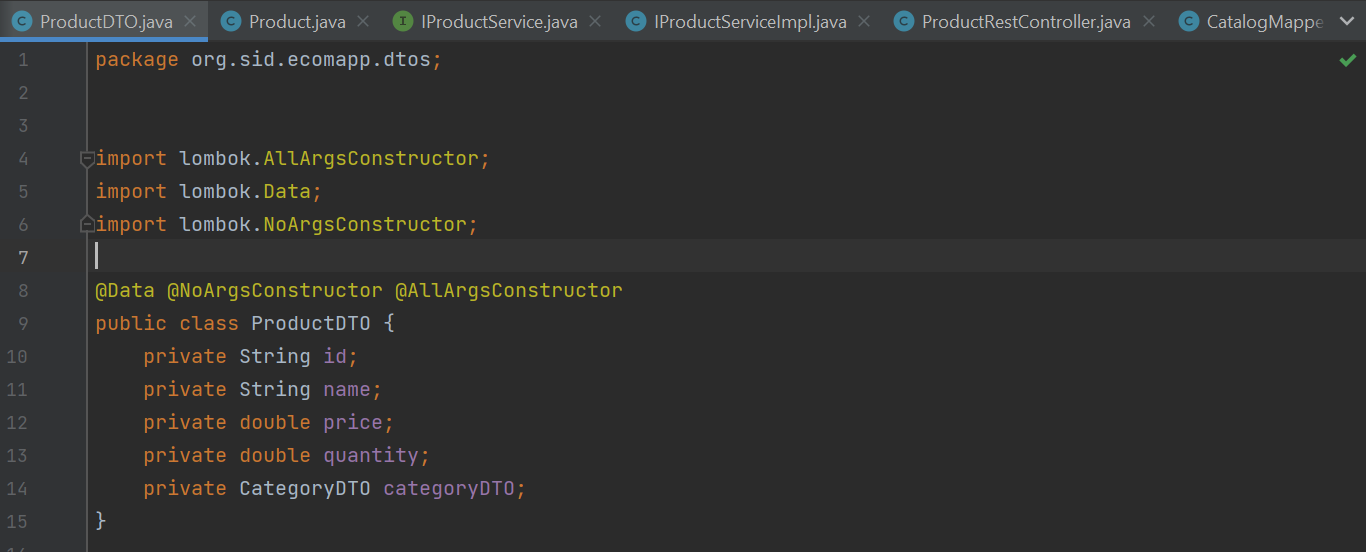
Les DTO sont utiles dans les systèmes avec des appels à distance, car ils aident à en réduire le nombre.

Les DTO sont également utiles lorsque le modèle de domaine est composé de nombreux objets différents et que le modèle de présentation a besoin de toutes leurs données en même temps, ou ils peuvent même réduire les allers-retours entre le client et le serveur.

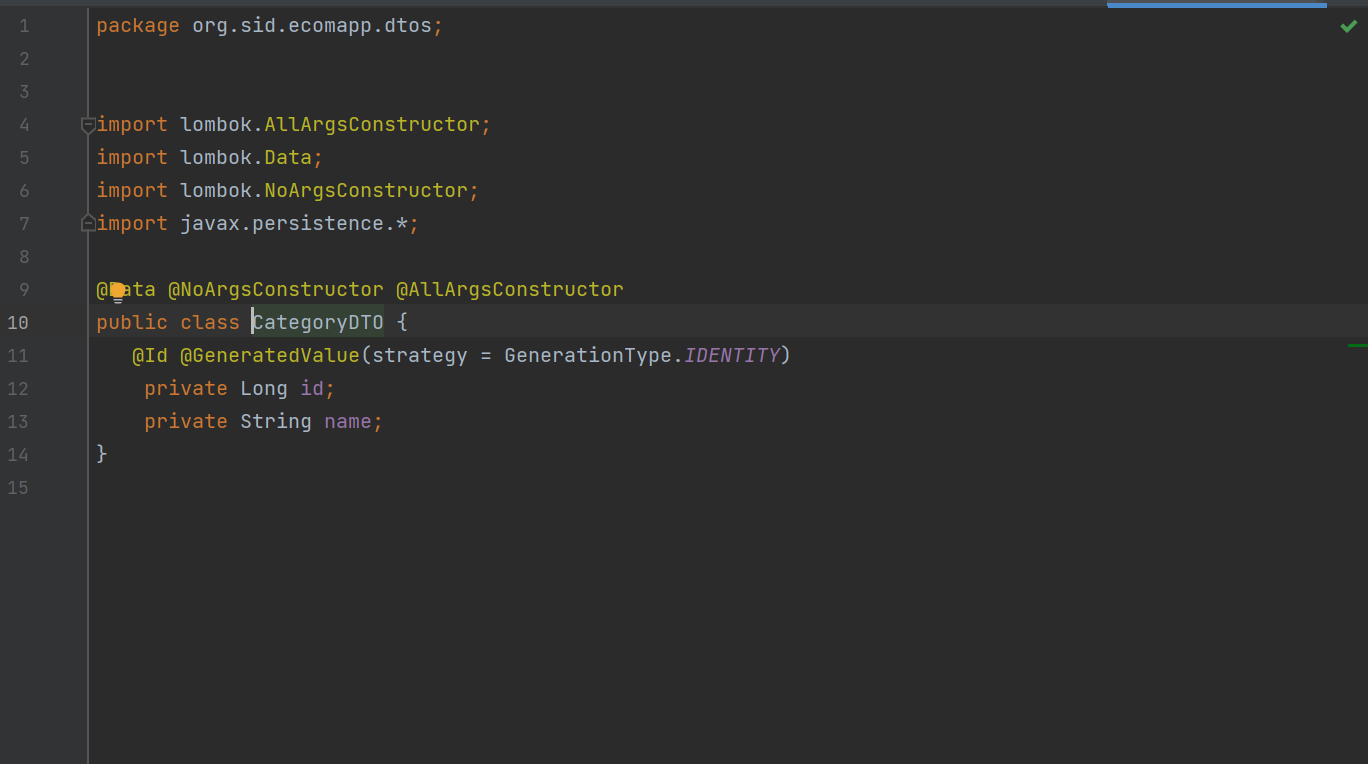
**Avec les DTO, nous pouvons créer différentes vues à partir de nos modèles de domaine**, ce qui nous permet de créer d'autres représentations du même domaine mais en les optimisant en fonction des besoins des clients sans affecter notre conception de domaine. Une telle flexibilité est un outil puissant pour résoudre des problèmes complexes.

<https://www.baeldung.com/java-dto-pattern#how-to-use-it>

* Classe ProductDTO

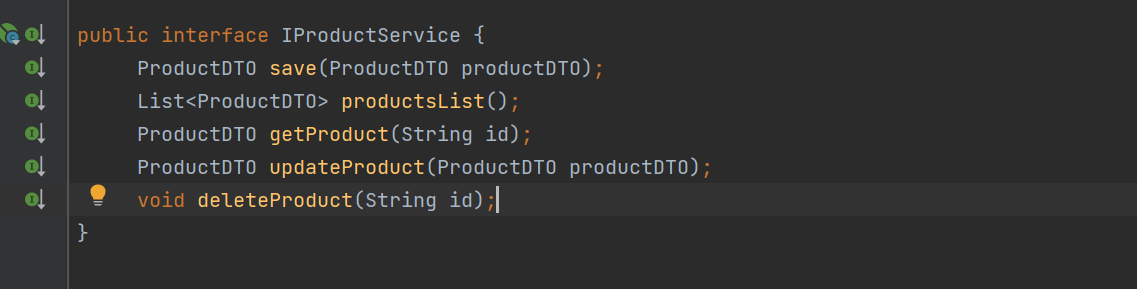


* Classe CategoryDTO

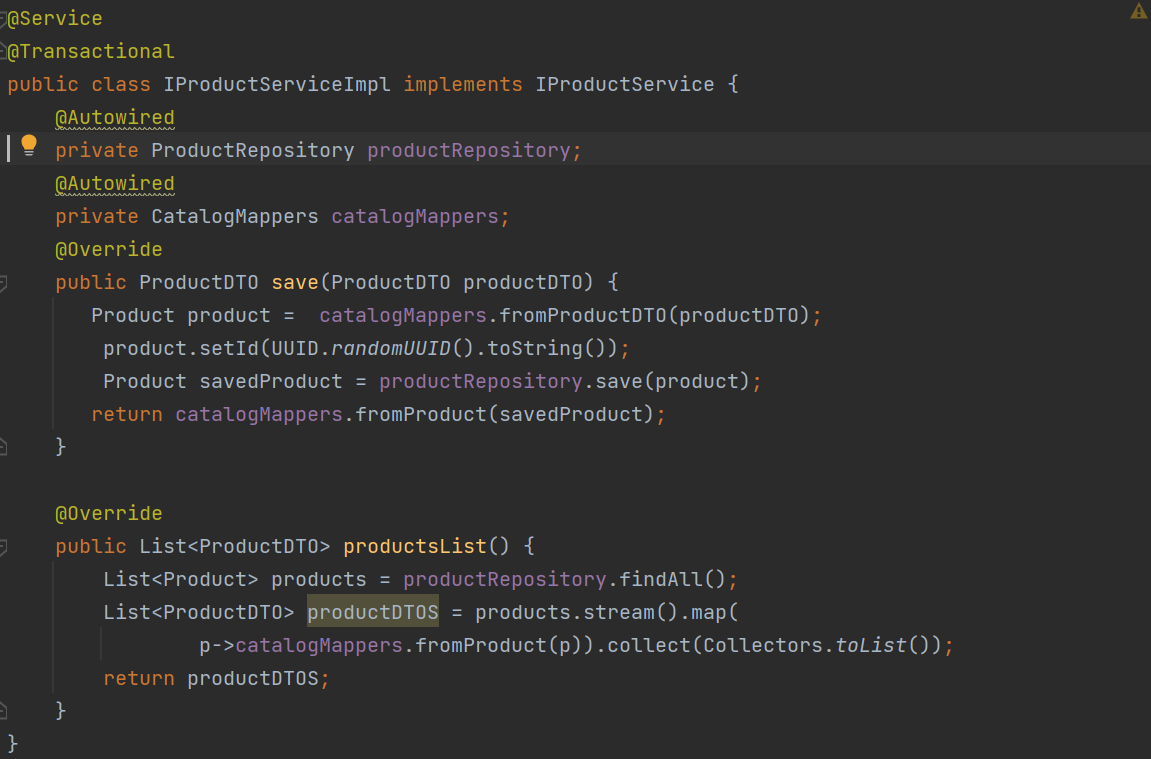


Etape 14 : Création de la couche service : partie métier de l’application

* Interface IProductService



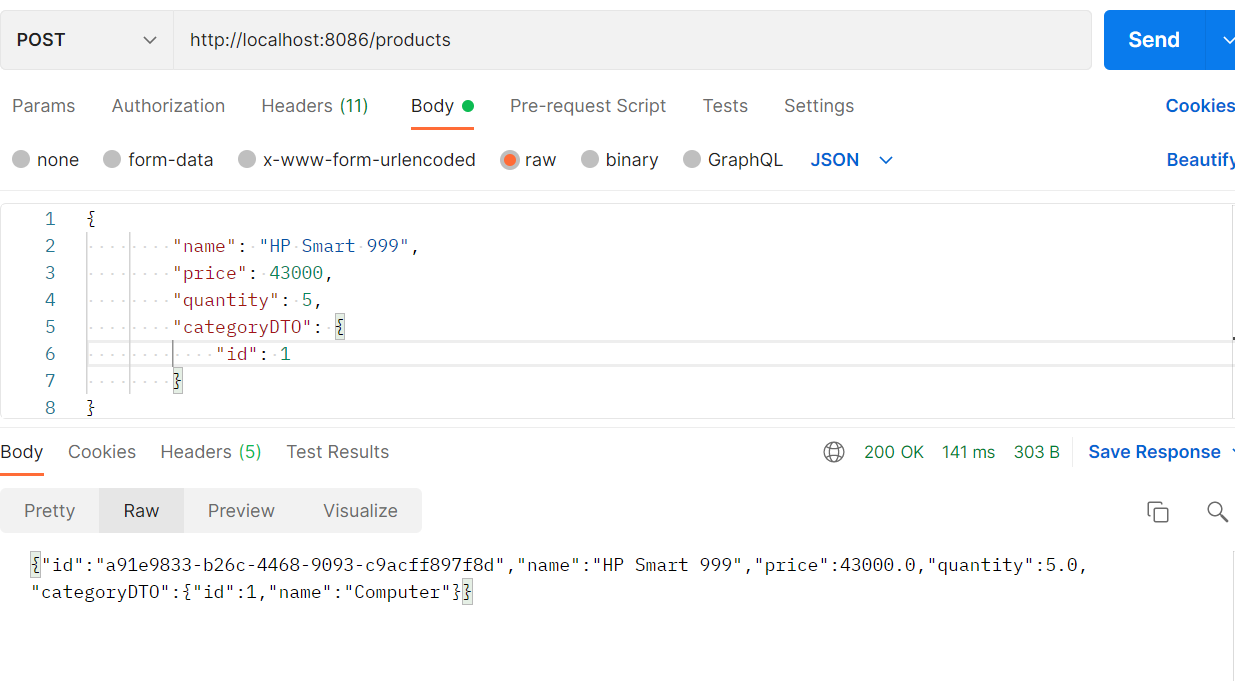
* Implémentation de l’interface IProductService

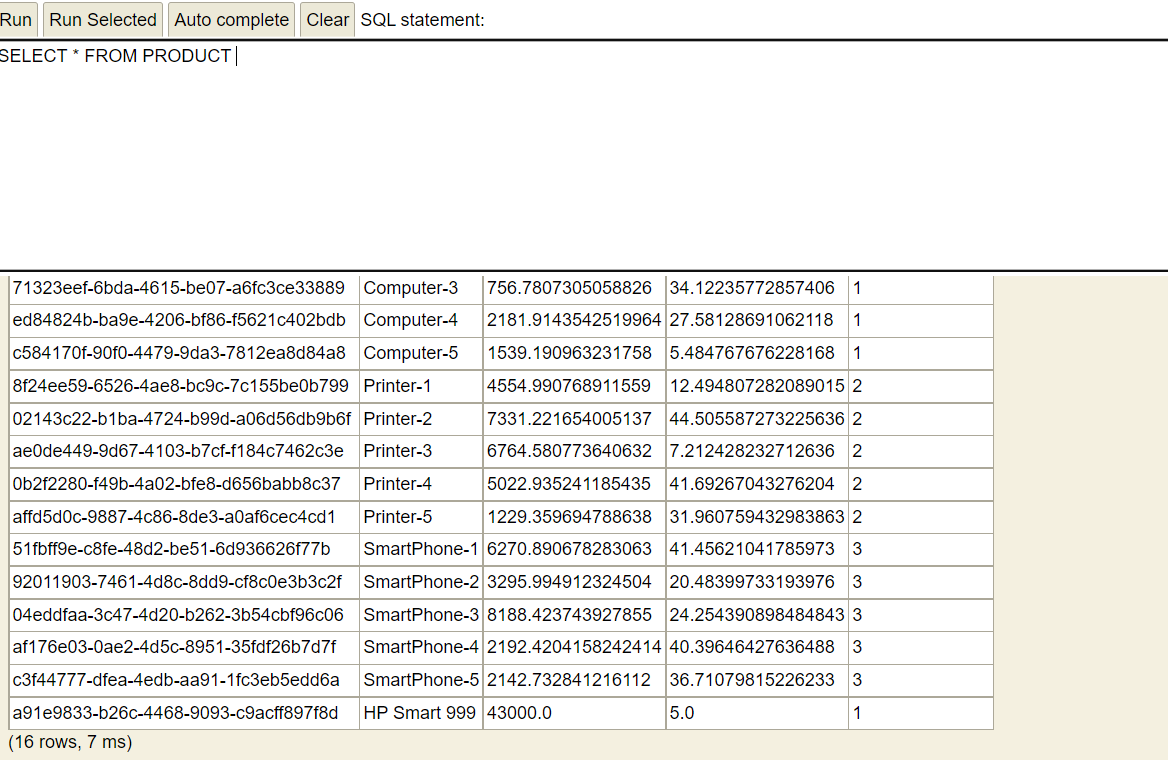


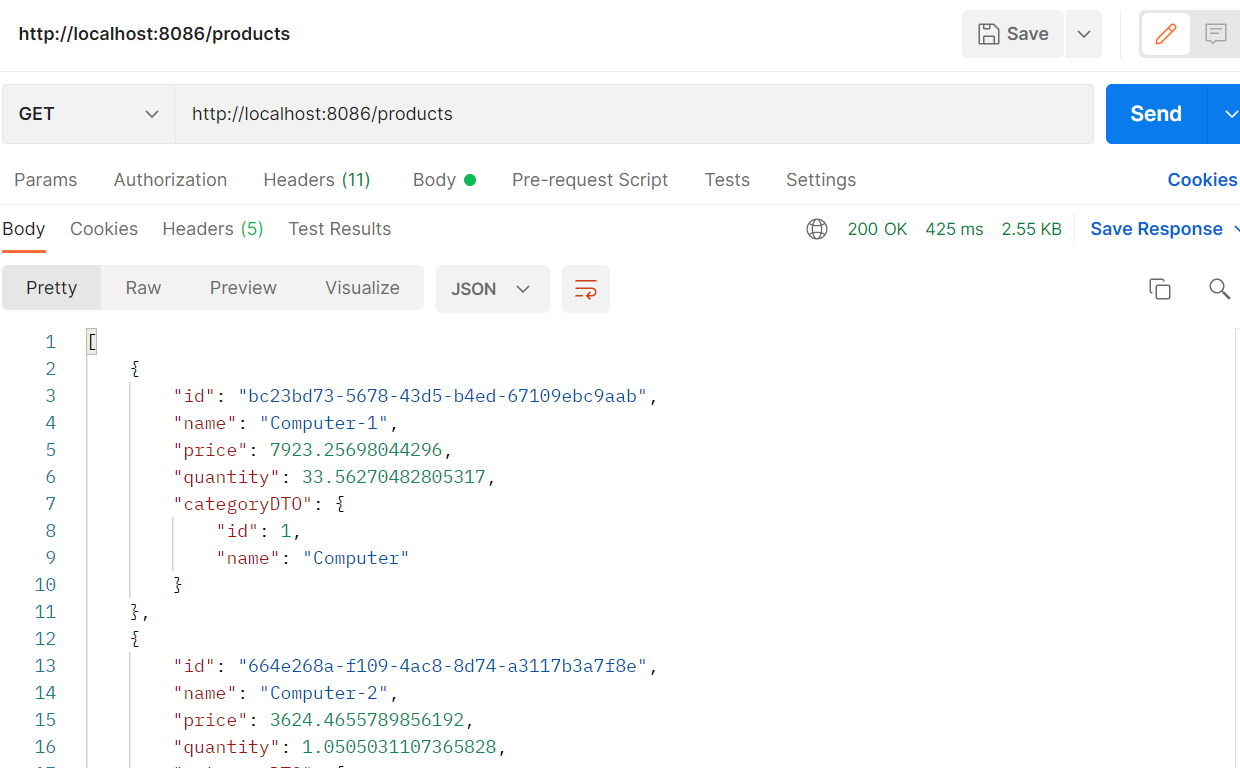


Etape 15 : Resultats

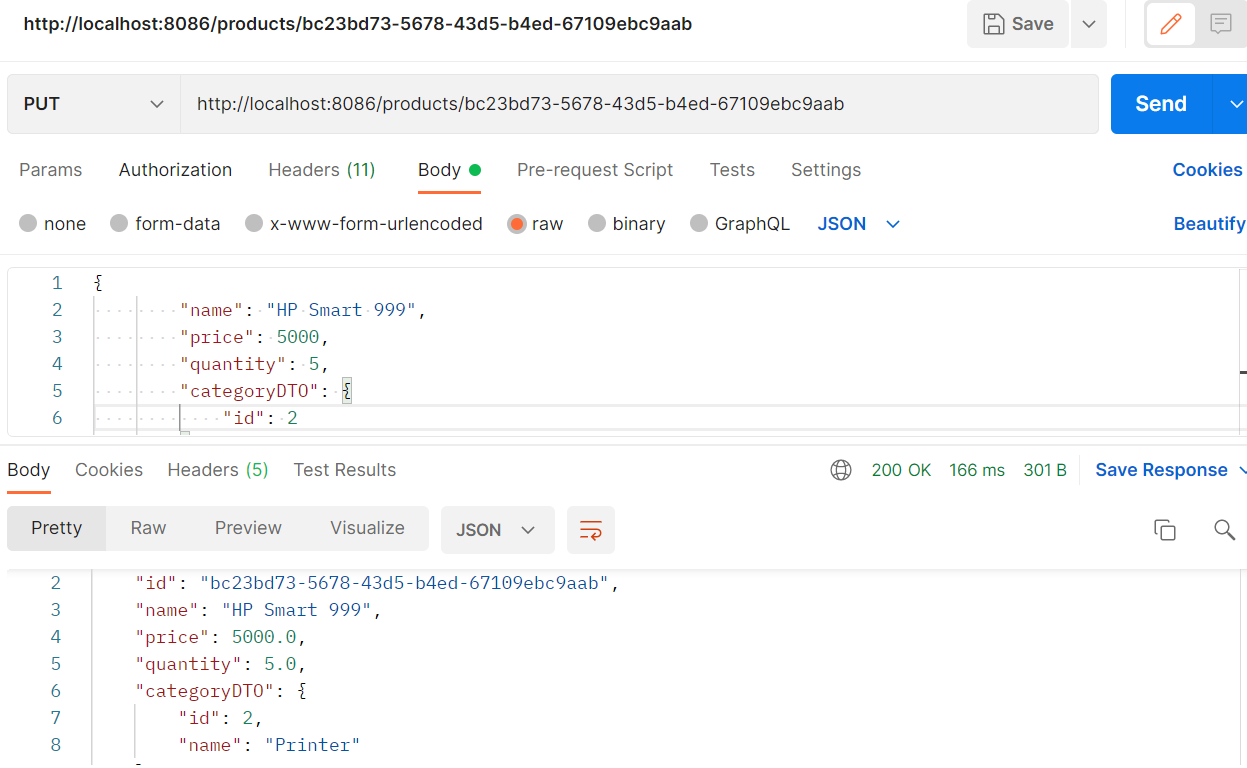
* Ajouter un produit

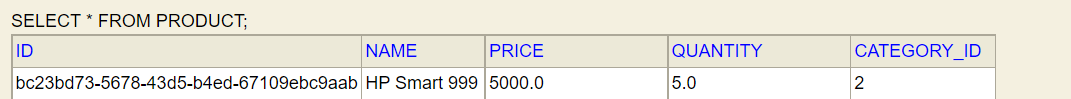






* Modifier un produit





* Supprimer un produit

