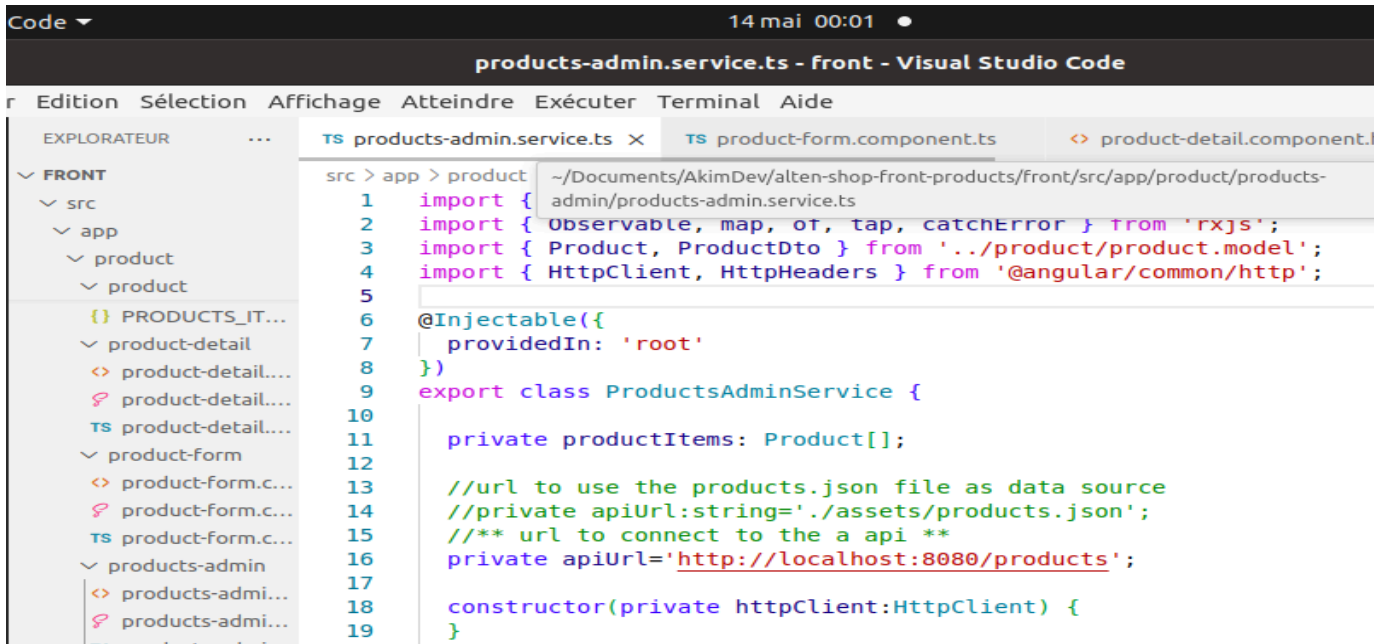


Le lien de l'api : <http://localhost:8080/products>

NB: Par défaut, le front est connecté au fichier product.json de ce fait l'ajout et la modification d'un produit ne sont pas opérationnels, l'ensemble des fonctionnalités de gestion sont disponibles que lorsque le front est connecté au back

Pour connecter le front au back, mettez en commentaire la ligne 14 du fichier ProductAdminService et décommentez la ligne 16 du même fichier (Voir Figure 1)



```
Code 14 mai 00:01
products-admin.service.ts - front - Visual Studio Code

Edition Sélection Affichage Atteindre Exécuter Terminal Aide

EXPLORATEUR ... TS products-admin.service.ts TS product-form.component.ts product-detail.component.ts

FRONT
src
  app
    product
      product
        PRODUCTS_IT...
        product-detail
          product-detail...
          product-detail...
          TS product-detail...
        product-form
          product-form.c...
          product-form.c...
          TS product-form.c...
        products-admin
          products-admi...
          products-admi...

src > app > product
1 import {
2 import { Observable, map, of, tap, catchError } from 'rxjs';
3 import { Product, ProductDto } from '../product/product.model';
4 import { HttpClient, HttpHeaders } from '@angular/common/http';
5
6 @Injectable({
7   providedIn: 'root'
8 })
9 export class ProductsAdminService {
10
11   private productItems: Product[];
12
13   //url to use the products.json file as data source
14   //private apiUrl:string='./assets/products.json';
15   /** url to connect to the a api **
16   private apiUrl='http://localhost:8080/products';
17
18   constructor(private httpClient:HttpClient) {
19   }
```

Figure 1: Connexion du front avec le back

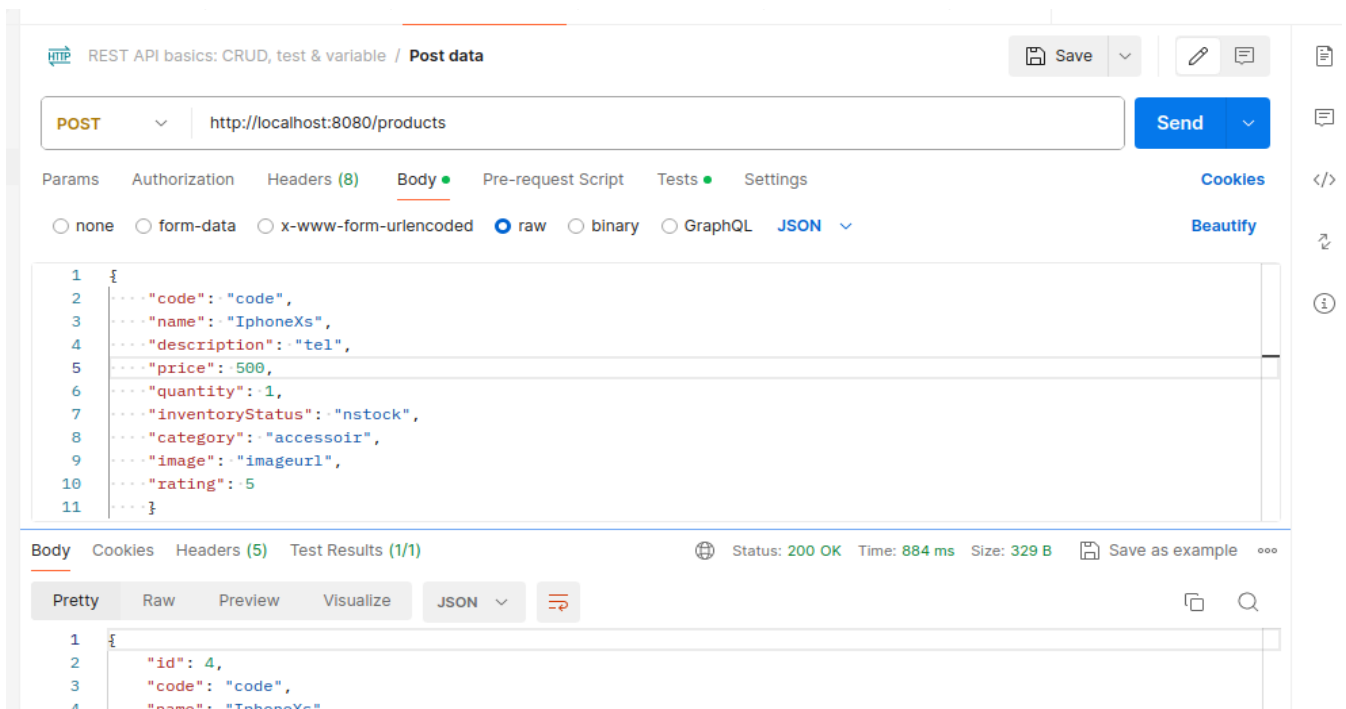


Figure 2: Exemple de la Méthode Post

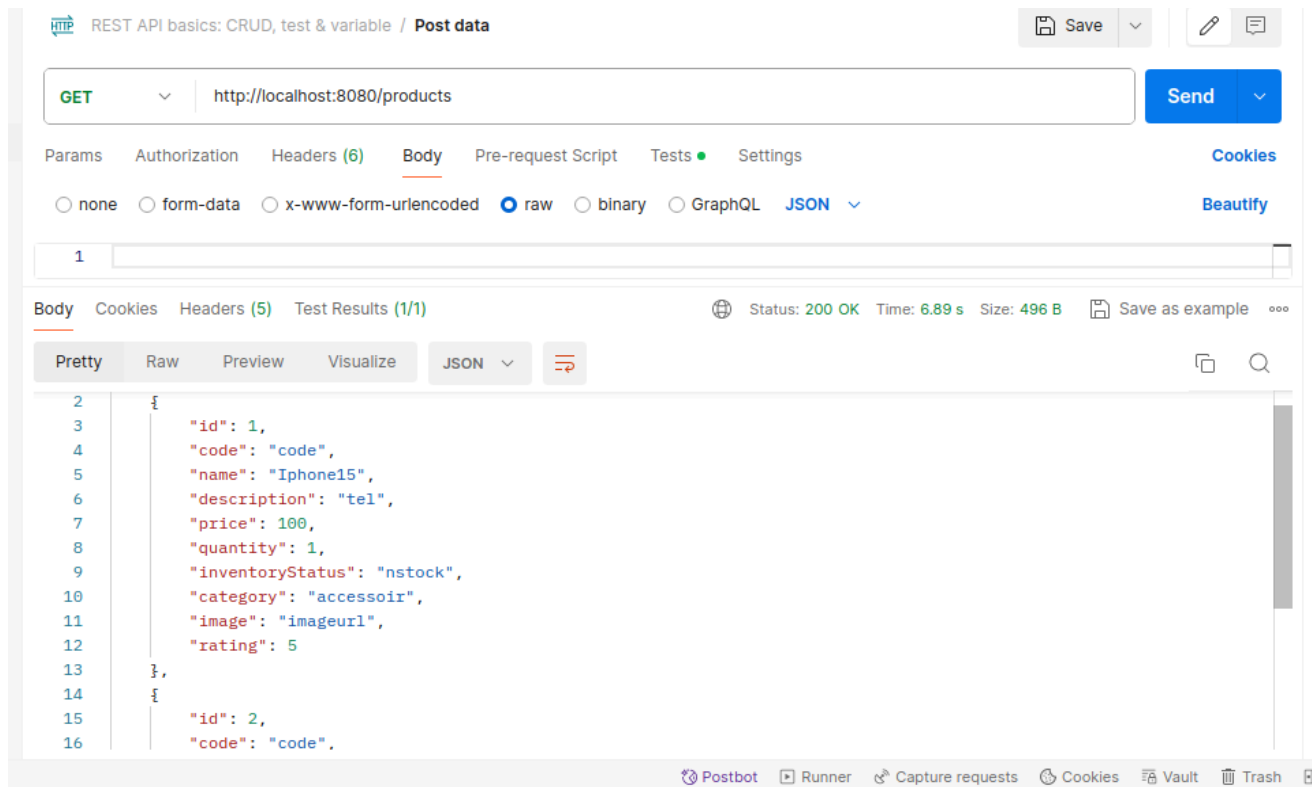


Figure 3: Exemple de la Méthode Get

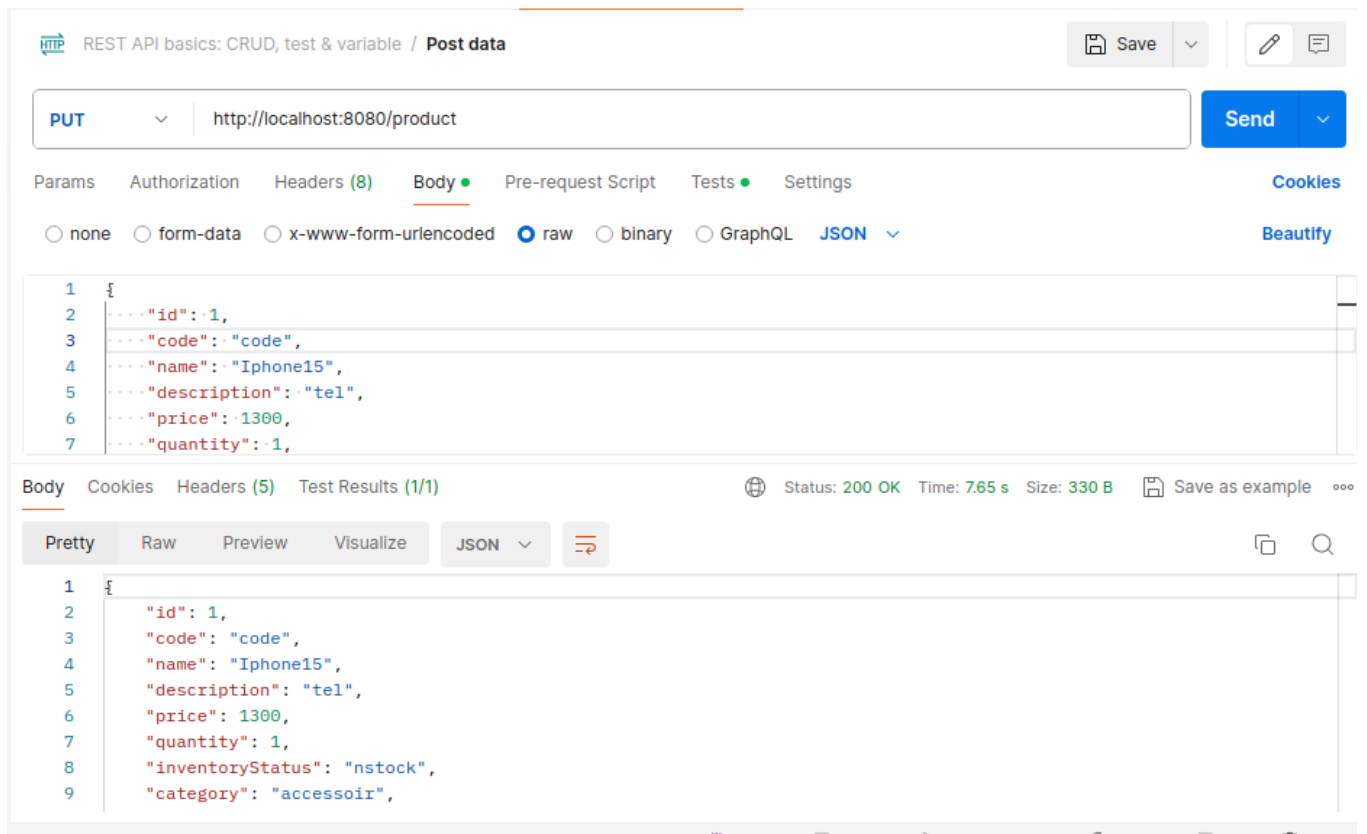


Figure 4: Exemple de la méthode Put

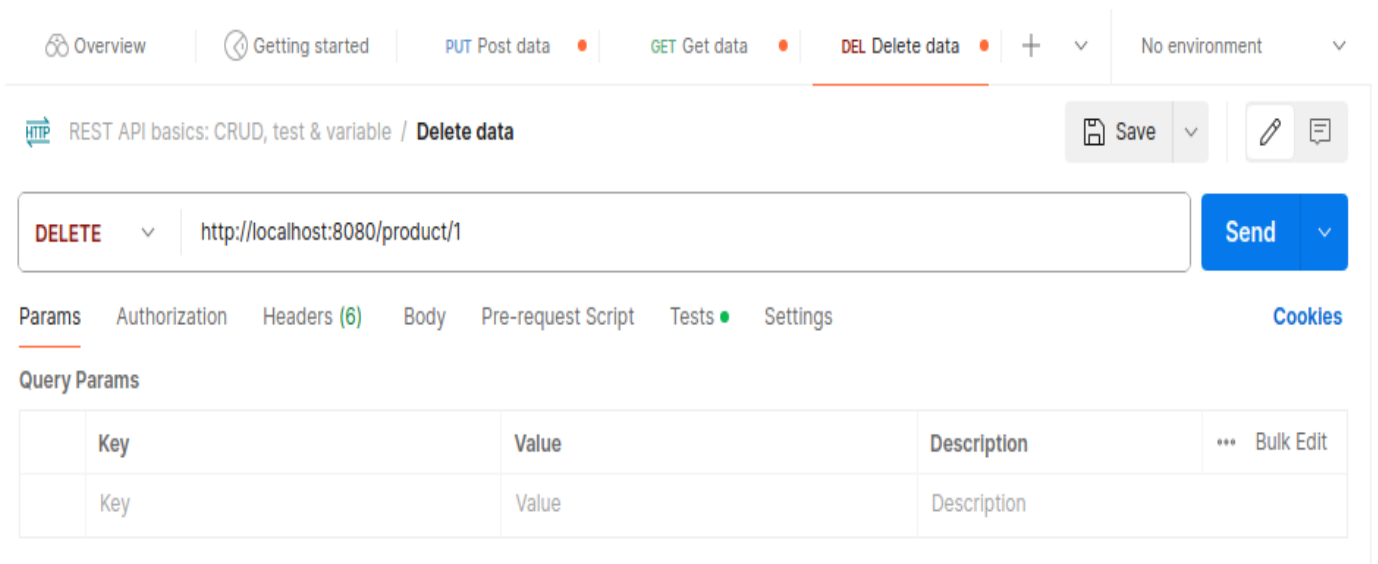


Figure 5: Exemple de la méthode Delete du produit dont id=1

POST

http://localhost:8080/products/array

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

{

2

→

"data":

→

[

3

→

→

→

{

4

→

→

→

"id":

→

1000,

5

→

→

→

"code":

→

"f230fh0g3",

6

→

→

→

"name":

→

"Bamboo Watch",

7

→

→

→

"description":

→

"Product Description",

8

→

→

→

"image":

→

"bamboo-watch.jpg",

9

→

→

→

"price":

→

65,

10

→

→

→

"category":

→

"Accessories",

11

→

→

→

"quantity":

→

24,

12

→

→

→

"inventoryStatus":

→

"INSTOCK",

13

→

→

→

"rating":

→

5

14

→

→

→

}

,

15

→

→

→

{

16

→

→

→

"id":

→

1001,

17

→

→

→

"code":

→

"nvklal433",

18

→

→

→

"name":

→

"Black Watch",

19

→

→

→

"description":

→

"Product Description",

Body

Cookies

Headers (7)

Test Results (1/1)

Status: 200 OK

Time: 4.91 s

Postbot

Runner

Capture re

Figure 6: Post d'une donnée contenant un **tableau de produit**