# SpringCommerce

SpringCommerce là một ứng dụng web đơn giản nhưng mạnh mẽ, được thiết kế để phục vụ nhu cầu bán hàng thời trang trực tuyến của NHU STORE. Với giao diện dễ sử dụng và tối ưu hóa trải nghiệm người dùng, SpringCommerce giúp khách hàng dễ dàng tìm kiếm và lựa chọn các sản phẩm thời trang đa dạng như quần áo, phụ kiện và giày dép.

## Mô hình phát triển phần mềm

Dự án SpringCommerce của NHU STORE được phát triển theo mô hình Waterfall, phù hợp với các yêu cầu rõ ràng và cấu trúc hệ thống đơn giản. Mô hình này giúp đảm bảo tiến độ và sự kiểm soát chặt chẽ trong từng giai đoạn phát triển. Quá trình phát triển của dự án được chia thành các bước chính sau:

**1. Phân tích yêu cầu:** Thu thập và xác định các yêu cầu từ đề bài. Dự án sẽ hỗ trợ hiển thị sản phẩm, lọc theo tiêu chí và đặt hàng qua giỏ hàng.

**2. Thiết kế hệ thống:** Thiết kế cấu trúc cơ bản của ứng dụng web, bao gồm cơ sở dữ liệu và giao diện người dùng.

**3. Lập trình:** Triển khai các tính năng cốt lõi như hiển thị sản phẩm, lọc, và quản lý giỏ hàng.

**4. Kiểm thử:** Kiểm thử toàn bộ ứng dụng để đảm bảo tính năng hoạt động đúng theo yêu cầu.

**5. Triển khai:** Sau khi kiểm thử thành công, sản phẩm sẽ được triển khai ra thị trường.

## Chức năng chính

1. **Hiển thị sản phẩm:** Giao diện đơn giản hiển thị toàn bộ sản phẩm thời trang mà NHU STORE cung cấp. Người dùng có thể lọc và tìm kiếm sản phẩm theo các tiêu chí như danh mục, tên, giá, màu sắc và nhãn hiệu.
2. **Giỏ hàng và đặt hàng**: Người dùng có thể thêm sản phẩm vào giỏ hàng và tiến hành đặt hàng.
3. **Thanh toán khi nhận hàng:** Hiện tại ứng dụng chưa hỗ trợ thanh toán trực tuyến, người dùng sẽ thanh toán bằng tiền mặt khi sản phẩm được giao đến.

**Nguyên tắc và thực hành**

1. **Nguyên tắc đơn giản hóa:** Chỉ tập trung vào các tính năng thiết yếu để đáp ứng yêu cầu cơ bản của một MVP.
2. **Tính mở rộng:** Thiết kế hệ thống đơn giản nhưng dễ dàng mở rộng thêm các tính năng khác trong tương lai.
3. **Tách biệt logic và giao diện:** Các lớp chức năng và giao diện được thiết kế riêng biệt để dễ dàng bảo trì và nâng cấp.

**Cấu trúc mã nguồn**

Dự án SpringCommerce sử dụng cấu trúc chuẩn của Maven với các thư mục và gói như sau:

* com.example.SpringCommerce: Đây là gói chính của dự án, chứa các thành phần chính theo mô hình MVC (Model-View-Controller) để tổ chức mã nguồn một cách khoa học và dễ bảo trì.
* controller: Thư mục này chứa các lớp điều khiển, chịu trách nhiệm xử lý các yêu cầu từ phía người dùng, gọi các dịch vụ cần thiết và trả kết quả về cho giao diện người dùng.
* model: Chứa các lớp mô hình đại diện cho các đối tượng dữ liệu trong ứng dụng. Các lớp này ánh xạ với các bảng trong cơ sở dữ liệu và được sử dụng để lưu trữ và truy xuất dữ liệu.
* repository: Thư mục này chứa các lớp thao tác với cơ sở dữ liệu, sử dụng Spring Data JPA để quản lý dữ liệu một cách hiệu quả và tách biệt các thao tác cơ sở dữ liệu khỏi logic của ứng dụng.
* service: Thư mục này chứa các lớp dịch vụ xử lý logic nghiệp vụ của ứng dụng, đóng vai trò trung gian giữa controller và repository, thực hiện các xử lý cần thiết trước khi gửi dữ liệu tới các thành phần khác.
* SpringCommerceApplication.java: Đây là điểm khởi động của ứng dụng Spring Boot, chứa các phương thức để thêm dữ liệu vào cơ sở dữ liệu cũng như là phương thức để chạy ứng dụng.
* resources: Thư mục này chứa các tài nguyên cần thiết cho ứng dụng.
* static: Thư mục này chứa các tệp tĩnh bao gồm:

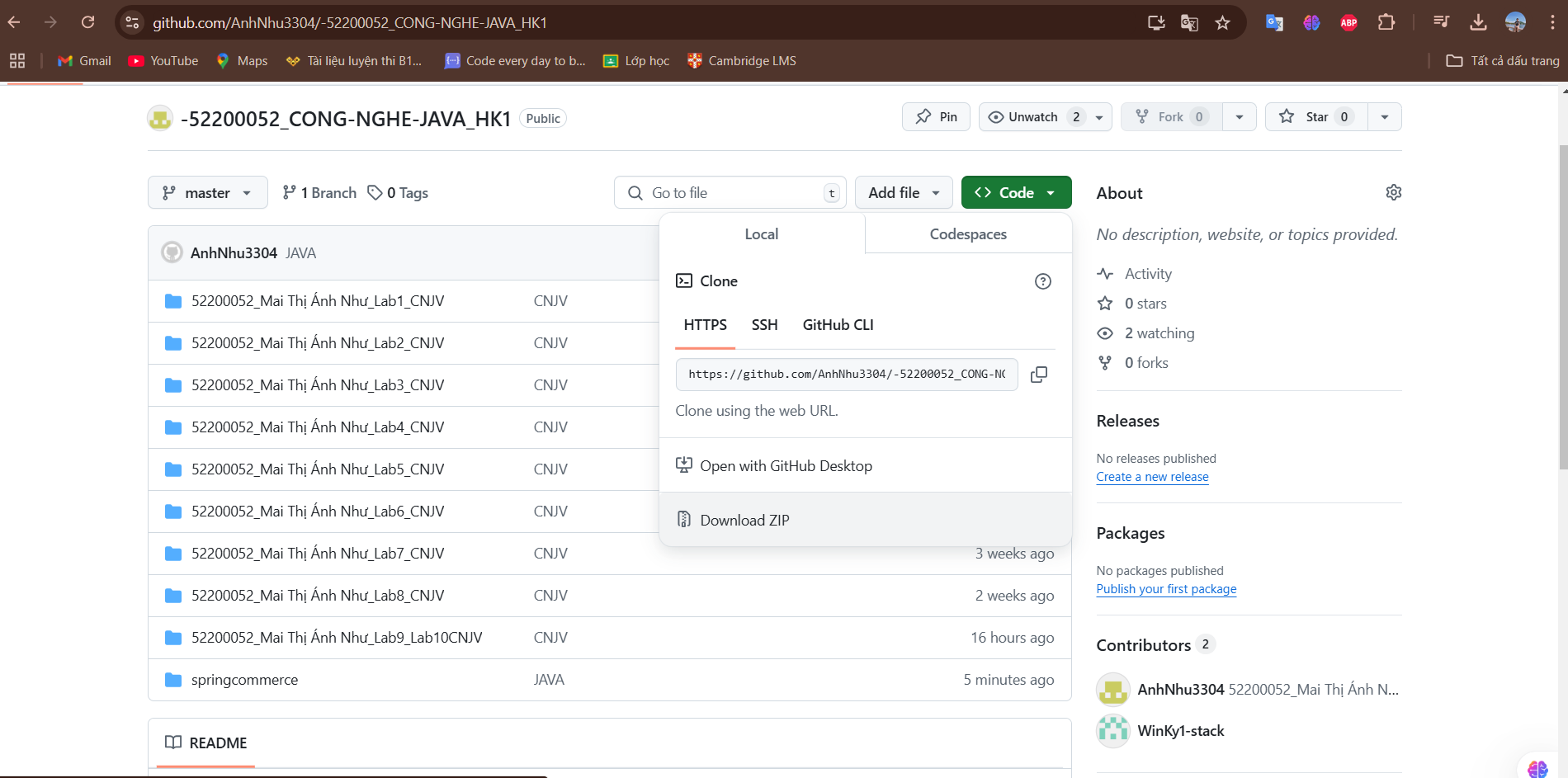
- images: Chứa các hình ảnh sử dụng trong giao diện.

* templates: Thư mục này chứa các tệp HTML sử dụng để hiển thị giao diện người dùng. Spring Boot sẽ sử dụng các tệp này để tạo trang web động, cho phép khách hàng xem sản phẩm, lọc sản phẩm, thêm vào giỏ hàng và thực hiện các thao tác đặt hàng.
* application.properties: Đây là nơi sẽ cấu hình cơ sở dữ liệu cho ứng dụng.

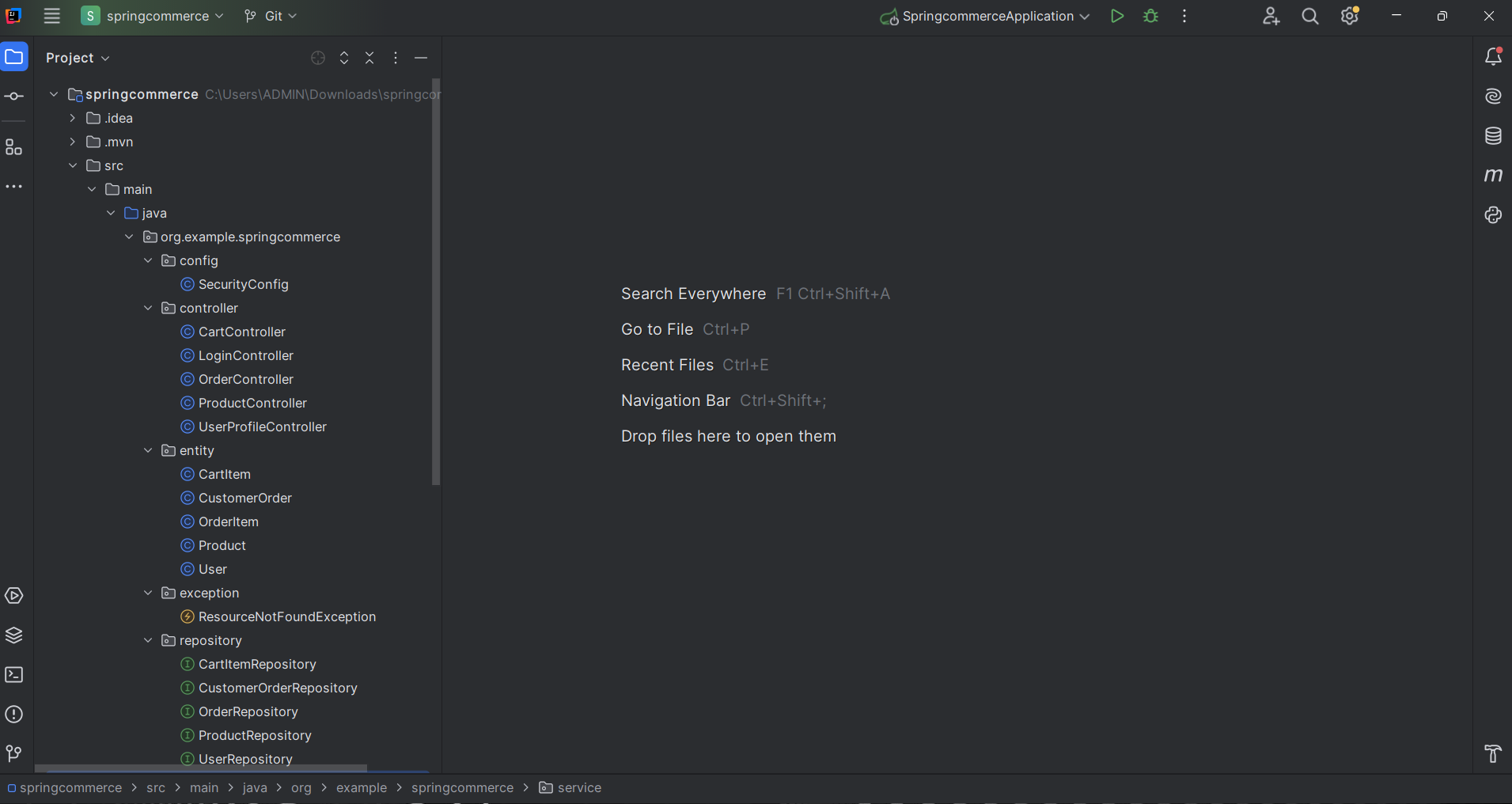
Cấu trúc này giúp ứng dụng dễ dàng bảo trì, mở rộng và đảm bảo tính nhất quán trong quy trình phát triển phần mềm theo mô hình Waterfall.

**Các bước để chạy ứng dụng**

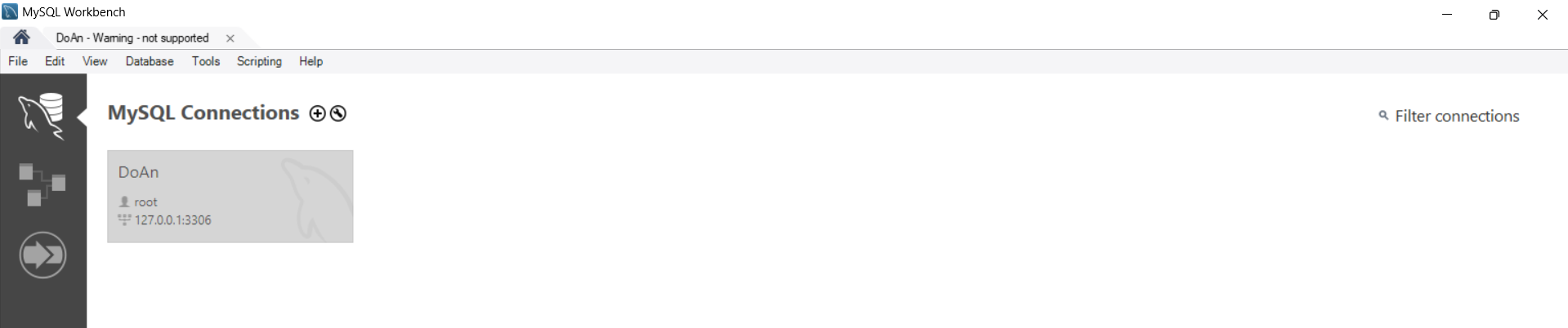
Đầu tiên ta sẽ nhấn vào phần Download ZIP để tải dự án về máy



Sau đó hãy giải nén và mở dự án bằng IntelliJ



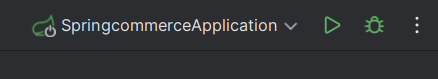
Tiếp đến ta mở MySQL Workbench để kết nối vào cơ sở dữ liệu



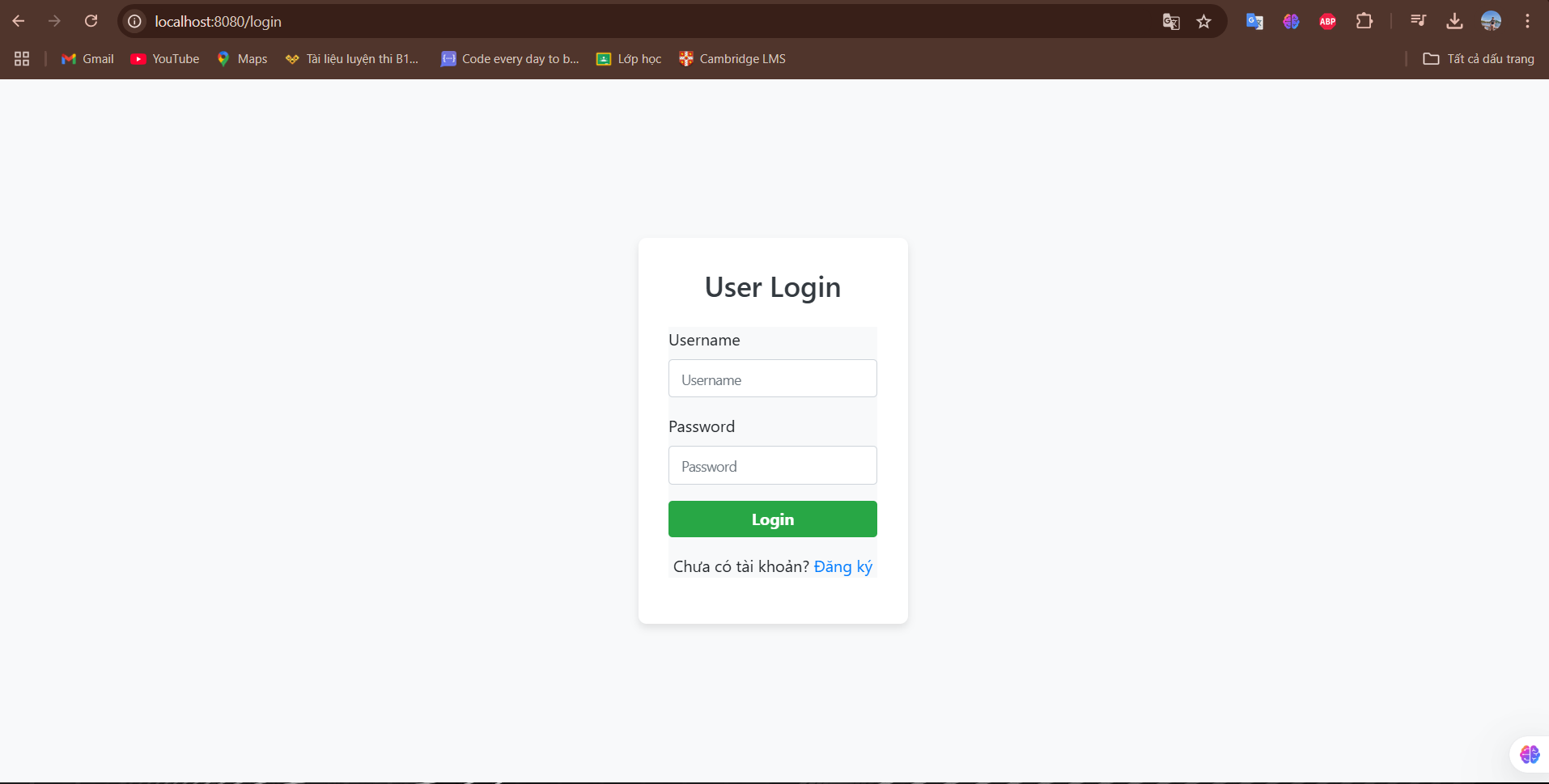
Tạo :



Kế đến ta sẽ nhấn vào nút chạy như ảnh dưới



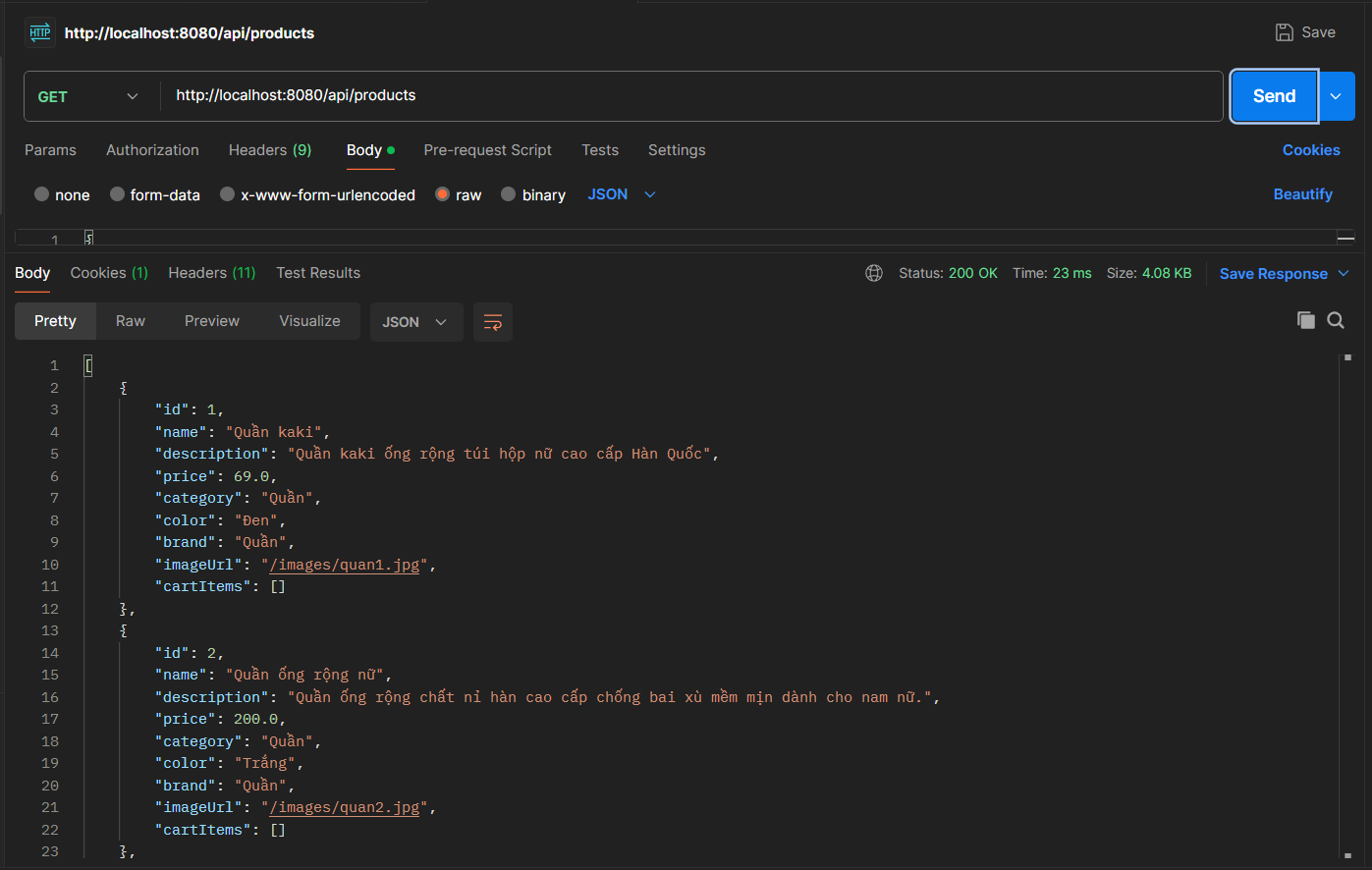
Cuối cùng ta vào Google Chrome hoặc các trình duyệt khác và chỉa tới đường dẫn localhost:8080, nếu như thành công ta sẽ được dẫn đến trang đăng nhập

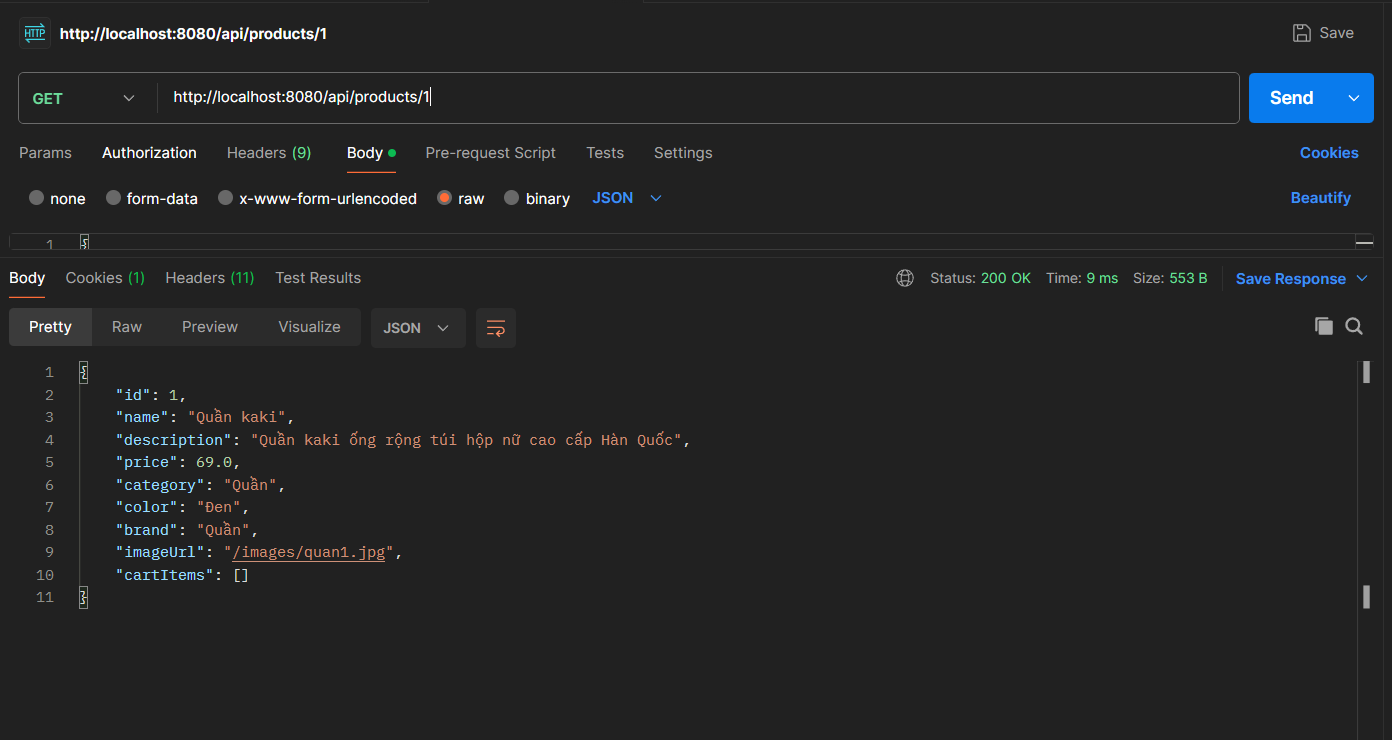


## Kiểm thử API

1. Đối với Product ta sẽ có các phương thức:

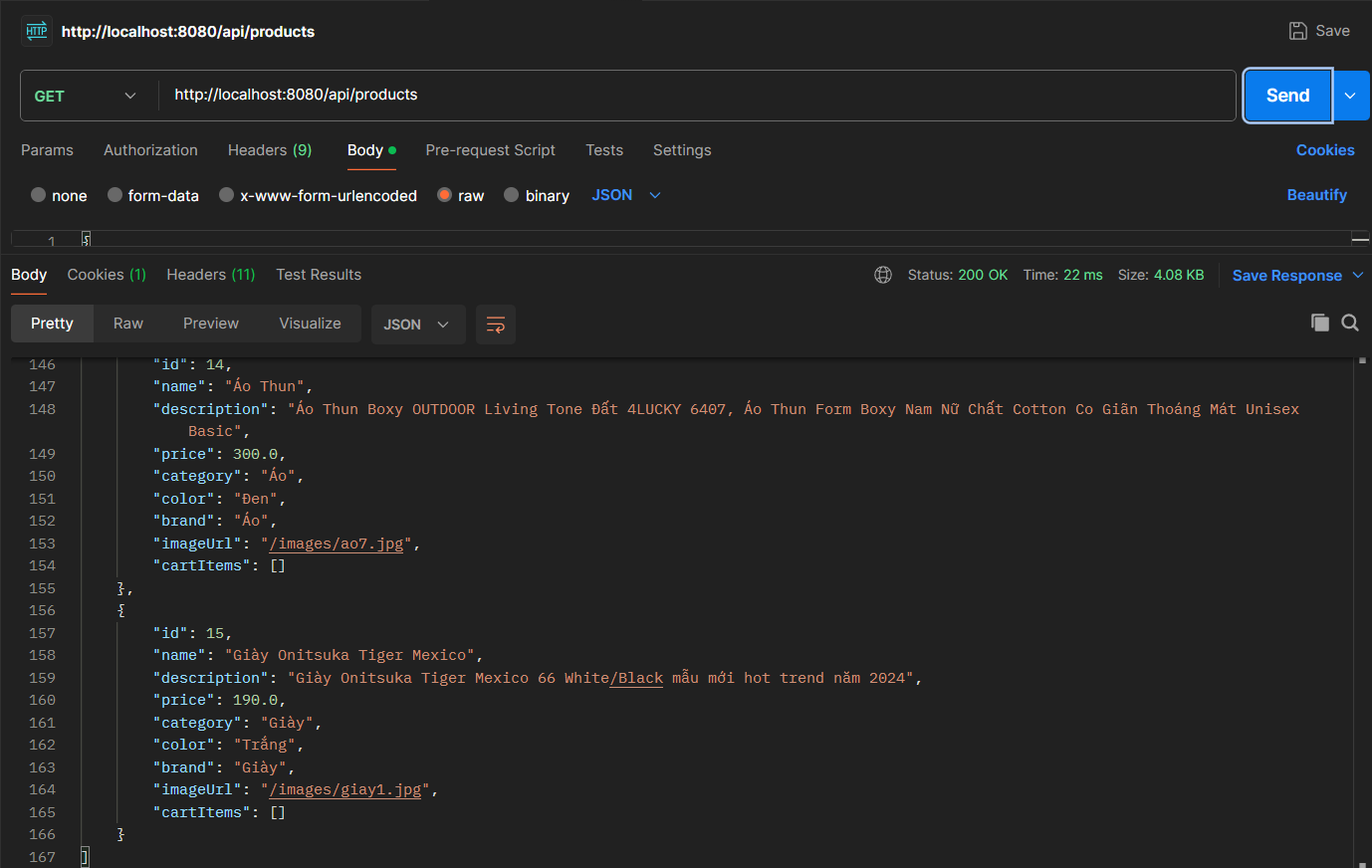
**GET**:

* <http://localhost:8080/api/products>: dùng để hiện thị tất cả các sản phẩm có trong cơ sở dữ liệu  
   
* <http://localhost:8080/api/product/{id}>: dùng để hiện thị sản phẩm ứng với id của sản phẩm có trong cơ sở dữ liệu

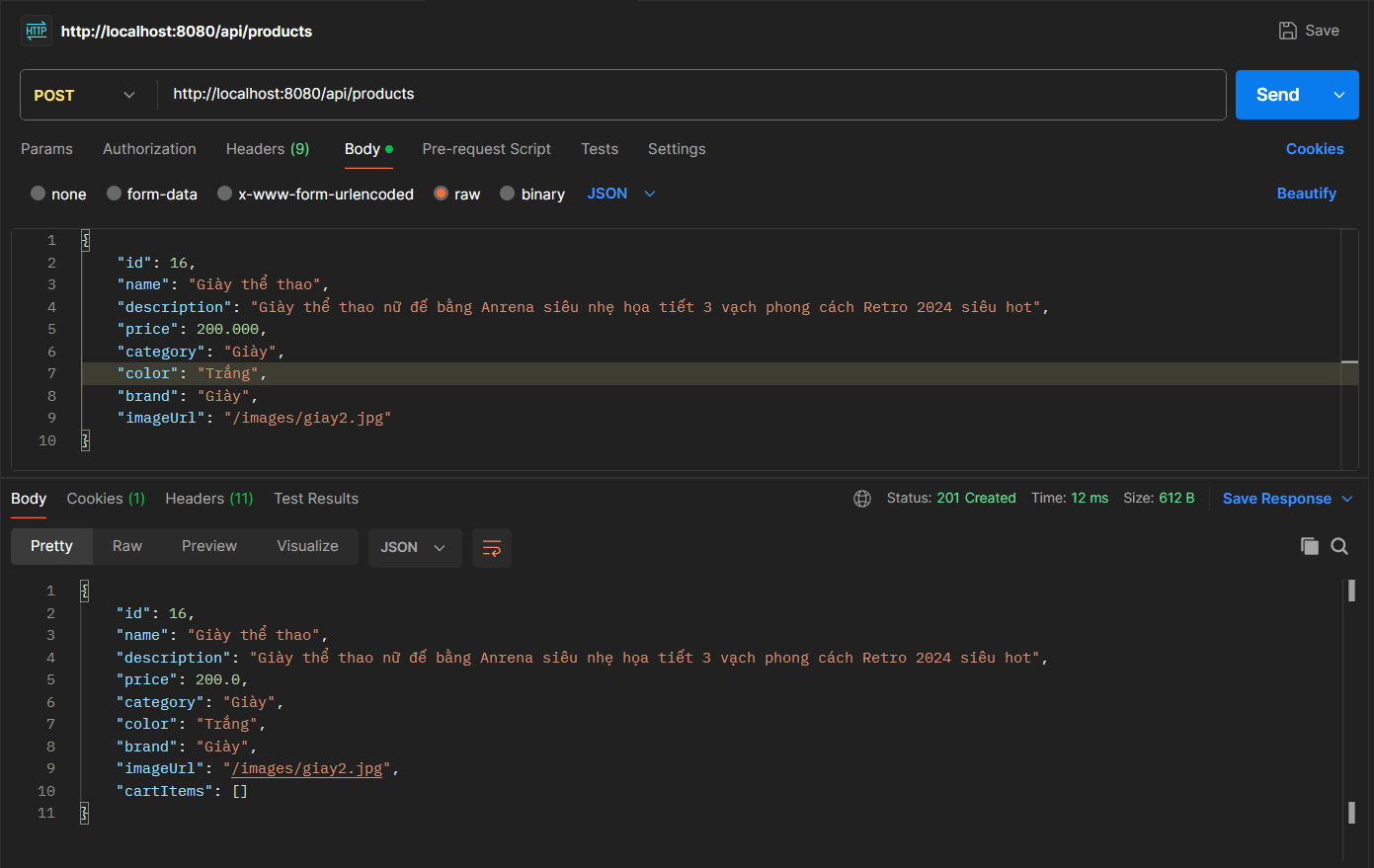


**POST**: <http://localhost:8080/api/product>: dùng để thêm sản phẩm vào cơ sở dữ liệu

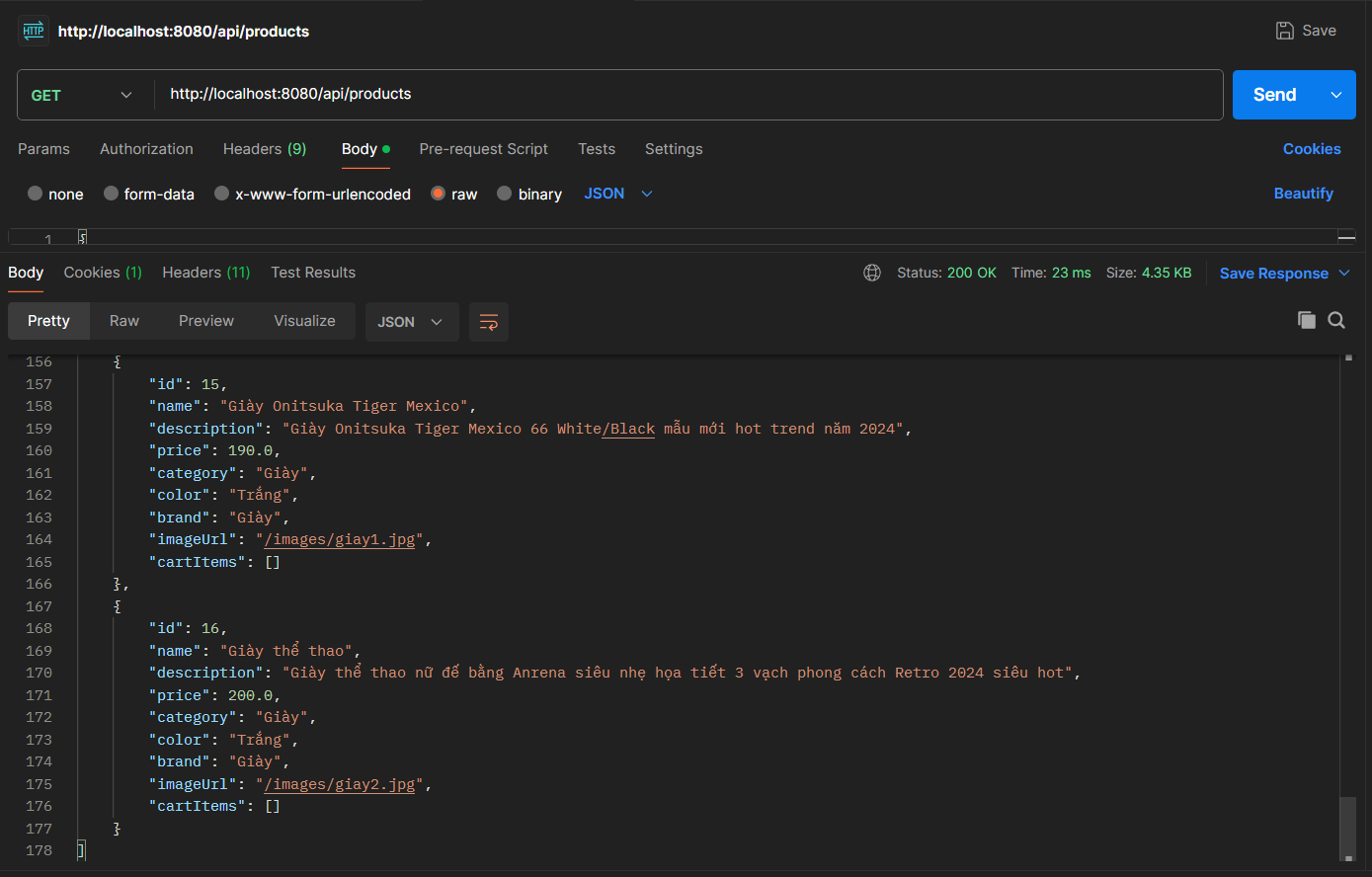
* Cơ sở dữ liệu hiện tại đang có id lớn nhất là 15:



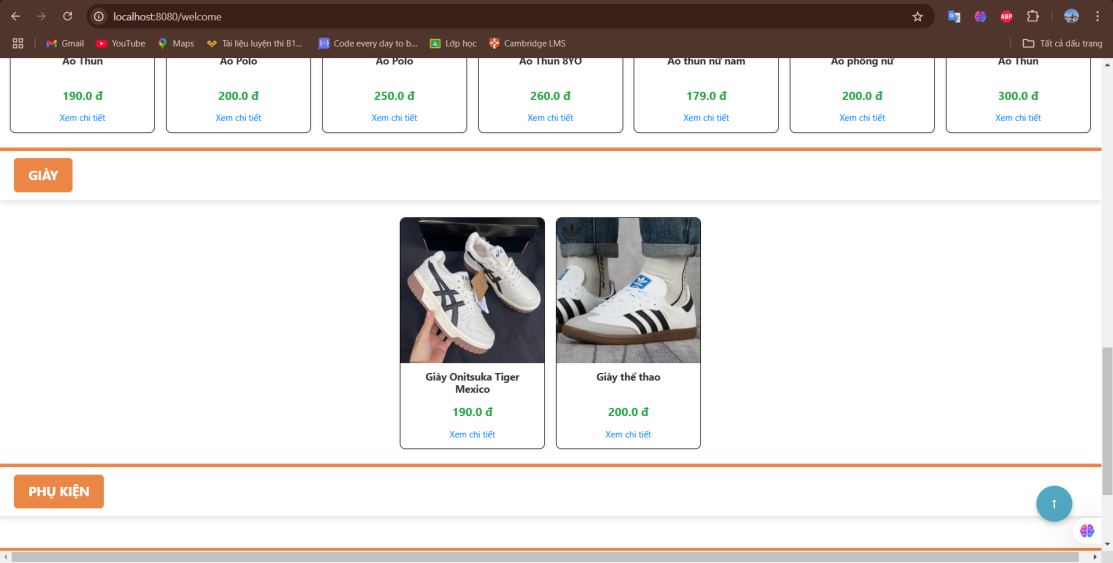
* Ta sẽ thêm như sau bằng phương thức **POST**:



* Cơ sở dữ liệu sau khi thêm sản phẩm mới vào:

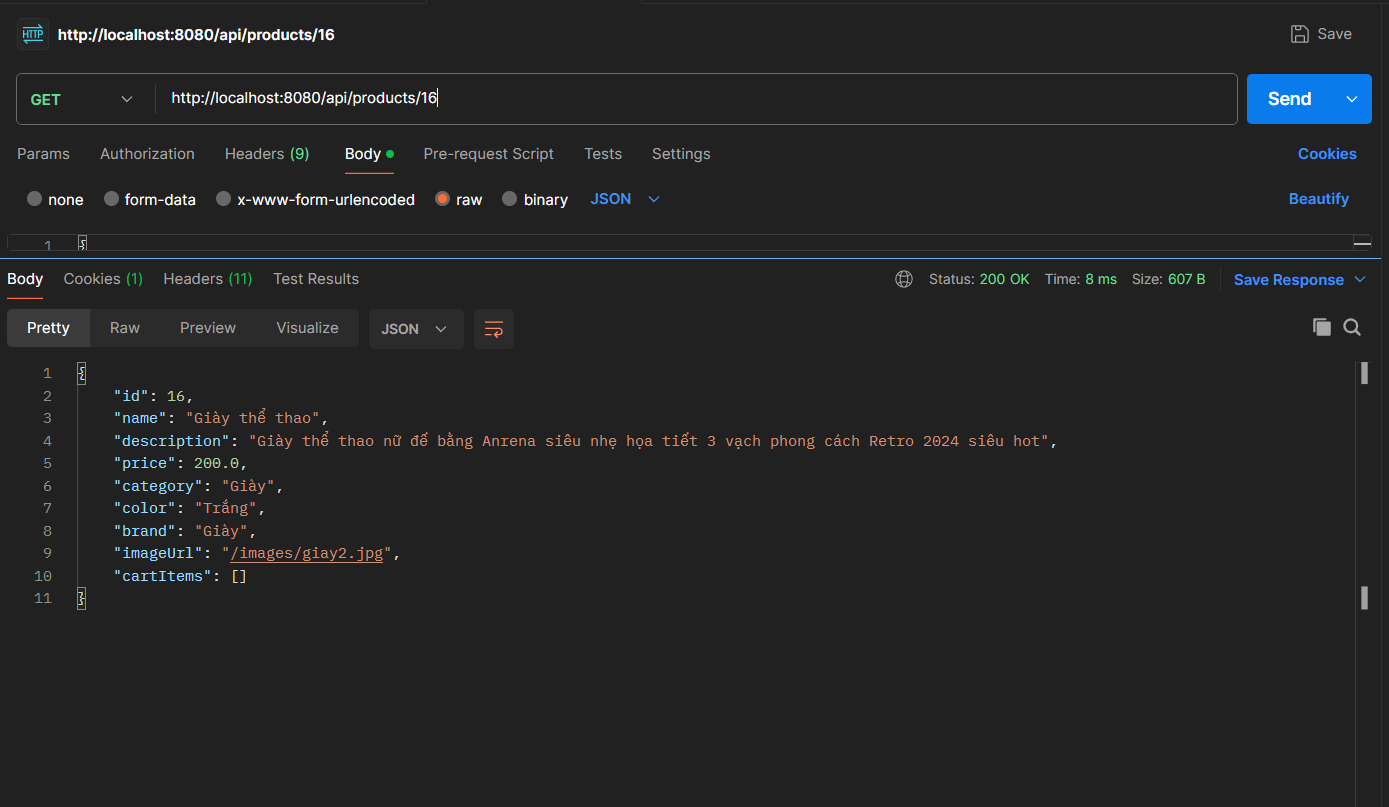


* Cuối cùng là ở localhost ta cũng đã thêm sản phẩm “Giày thể thao” sau khi sử dụng phương thức **POST**

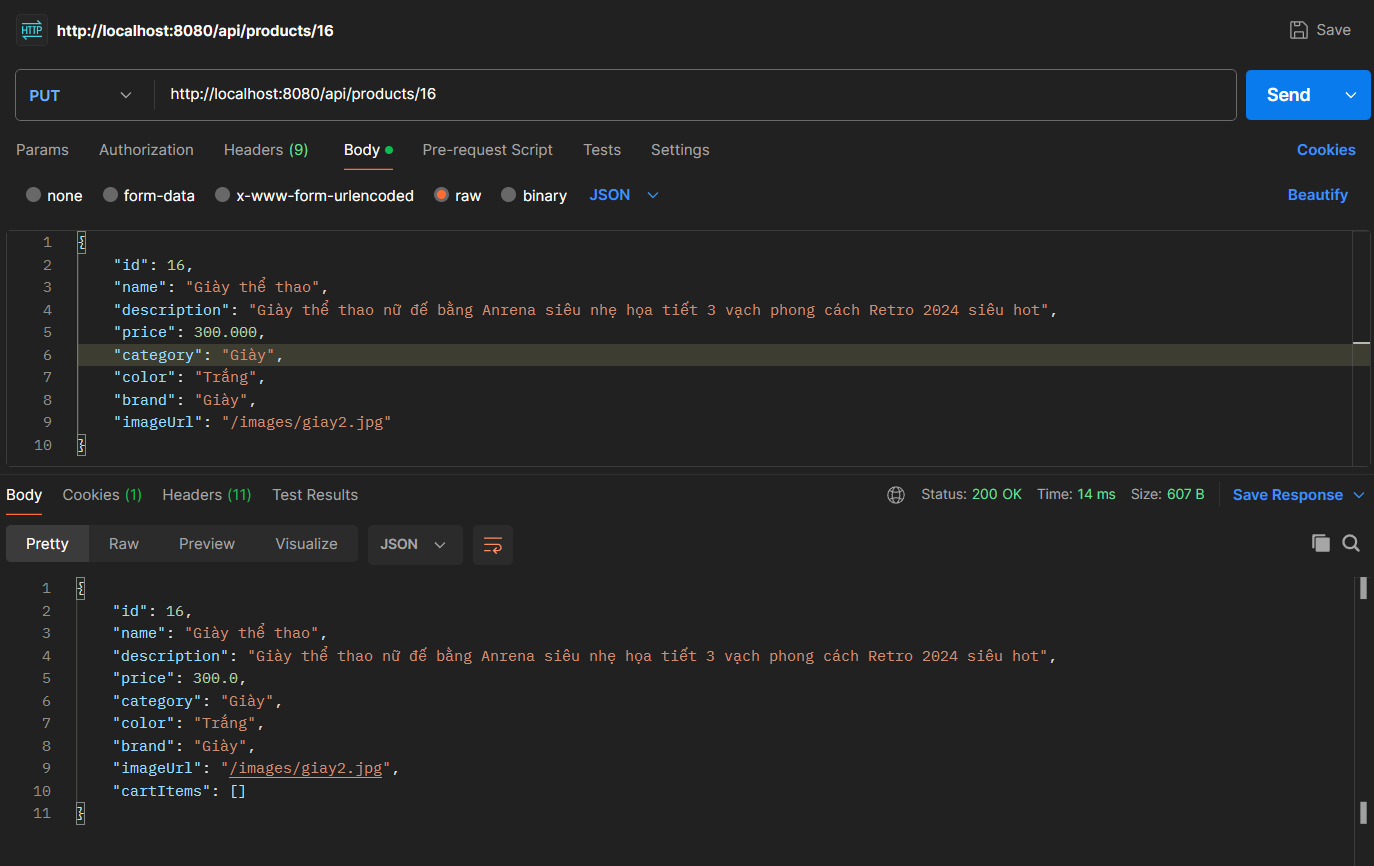


**PUT**: <http://localhost:8080/api/product/{id}>: dùng để chỉnh sửa sản phẩm ứng với id sản phảm phẩm có trong cơ sở dữ liệu

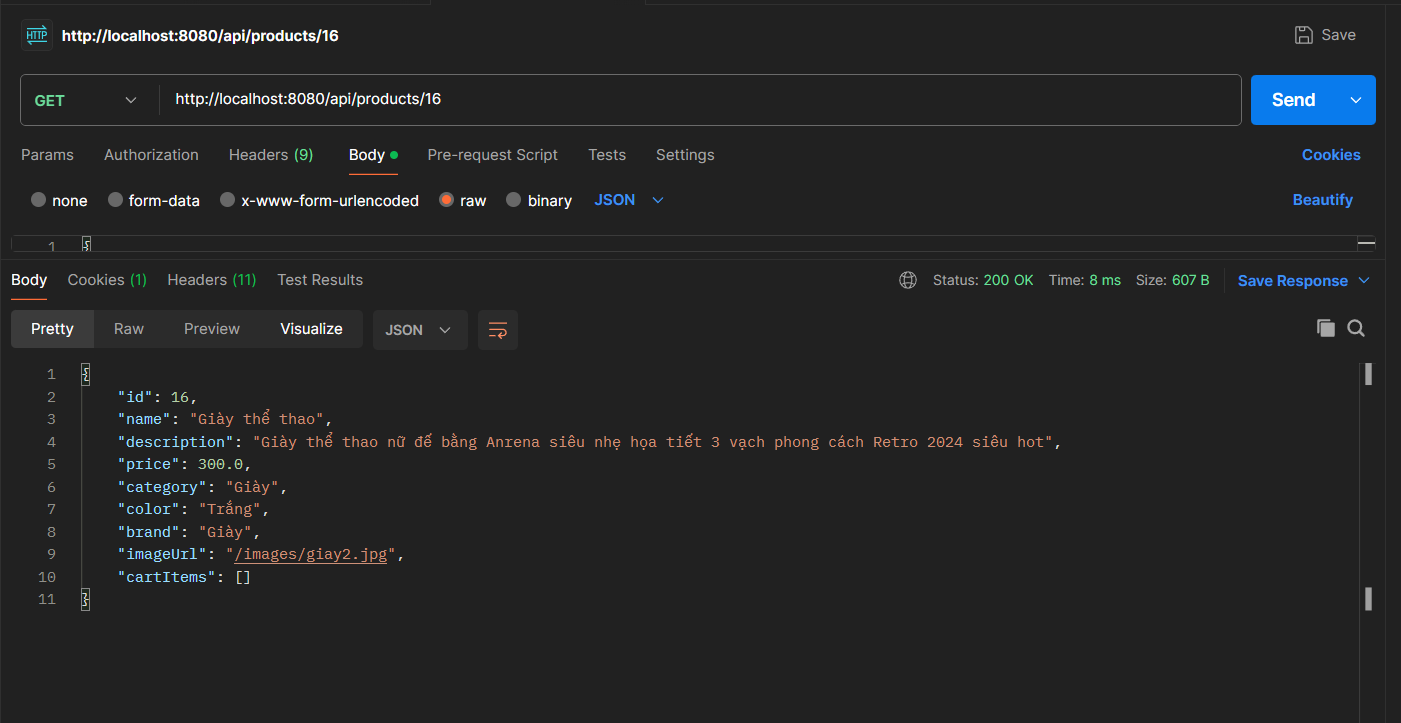
* Ta sẽ sử dụng phương thức **GET** để lấy sản phẩm với id tùy chọn, ảnh dưới đây sẽ chọn id là 16:



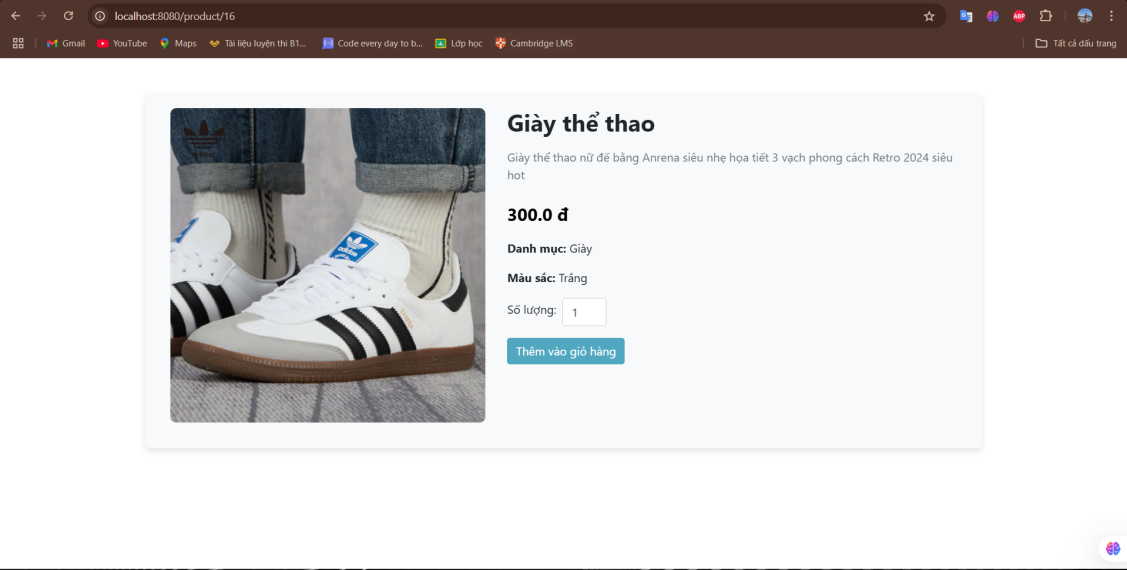
* Sau đó sử dụng phương thức **PUT** để chỉnh sửa như sau, ảnh dưới sẽ chỉnh sửa giá từ 200.000đ thành 300.000đ:



* Tiếp đến ta sẽ kiểm tra lại bằng phương thức **GET**:

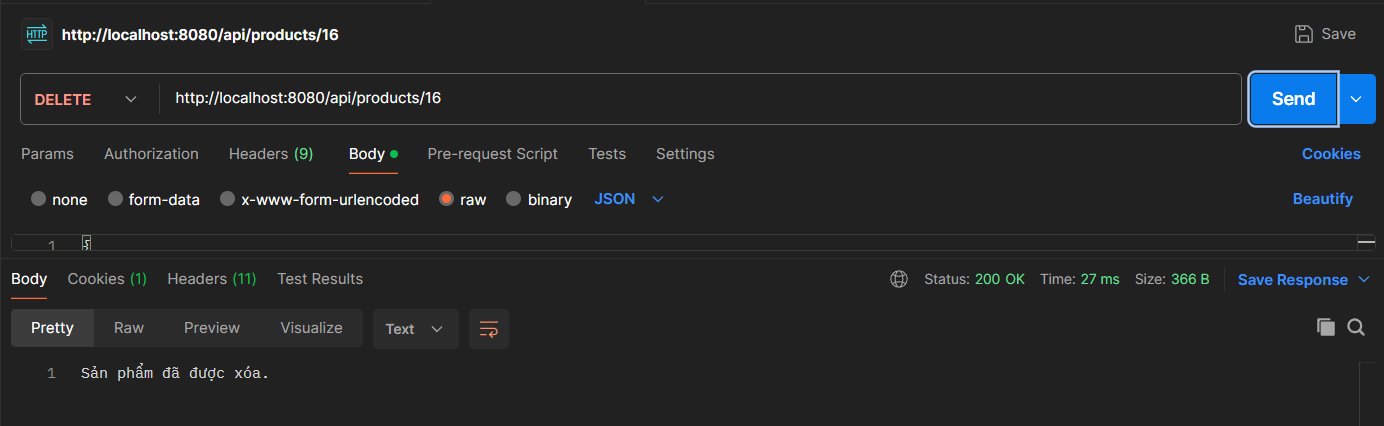


* Cuối cùng là kiểm tra trên localhost:

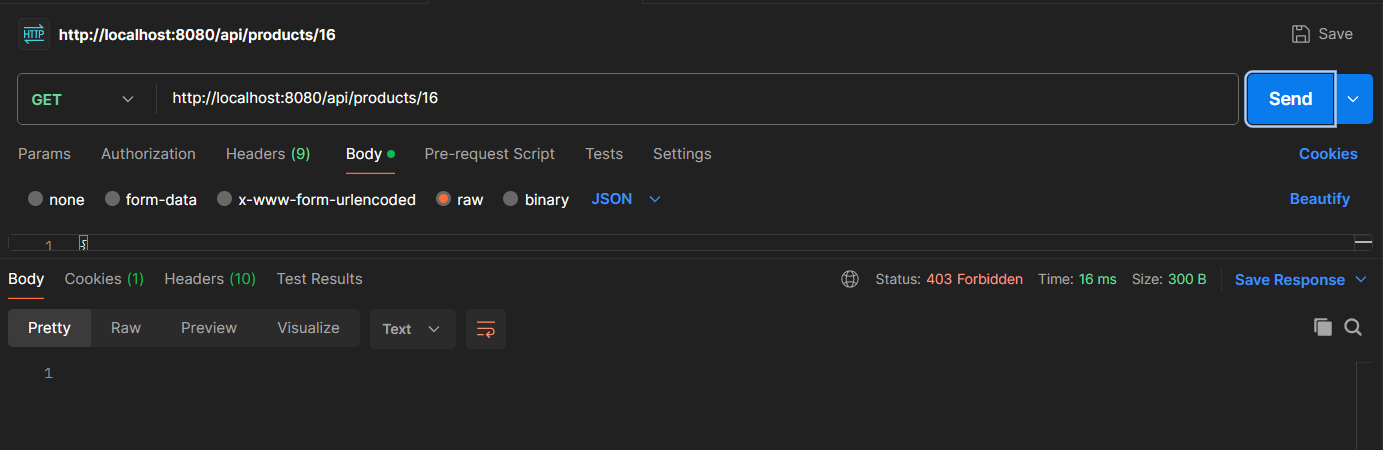


**DELETE**: <http://localhost:8080/api/product/{id}>: dùng để xóa sản phẩm ứng với id sản phẩm có trong cơ sở dữ liệu

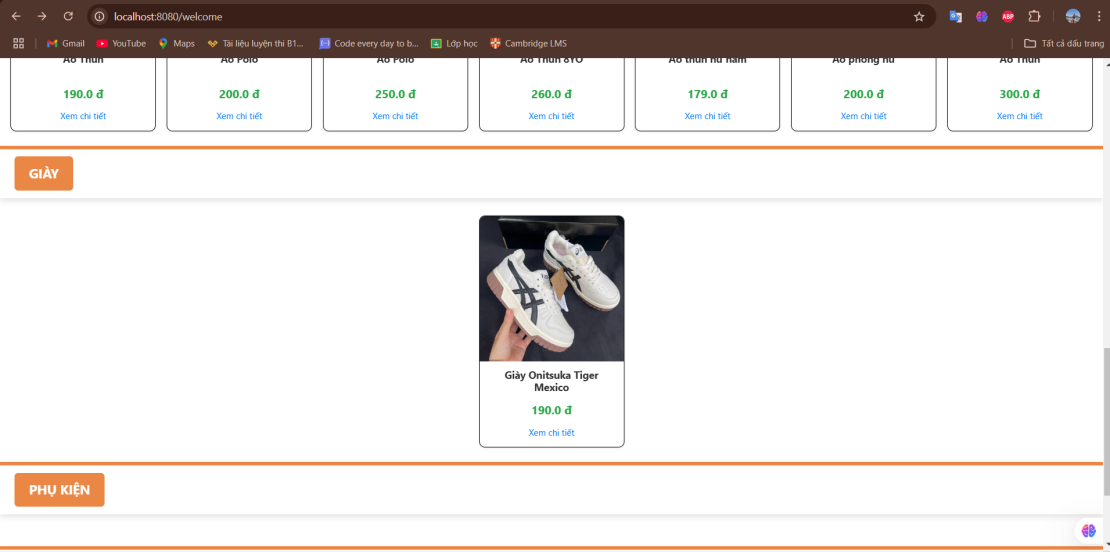
* Ta sẽ sử dụng phương thức **DELETE** để xóa sản phẩm với id bất kì, ảnh dưới sẽ chọn id là 16 tương ứng với “Giày thể thao”:



* Sau khi xóa ta kiểm tra lại bằng phương thức **GET**:

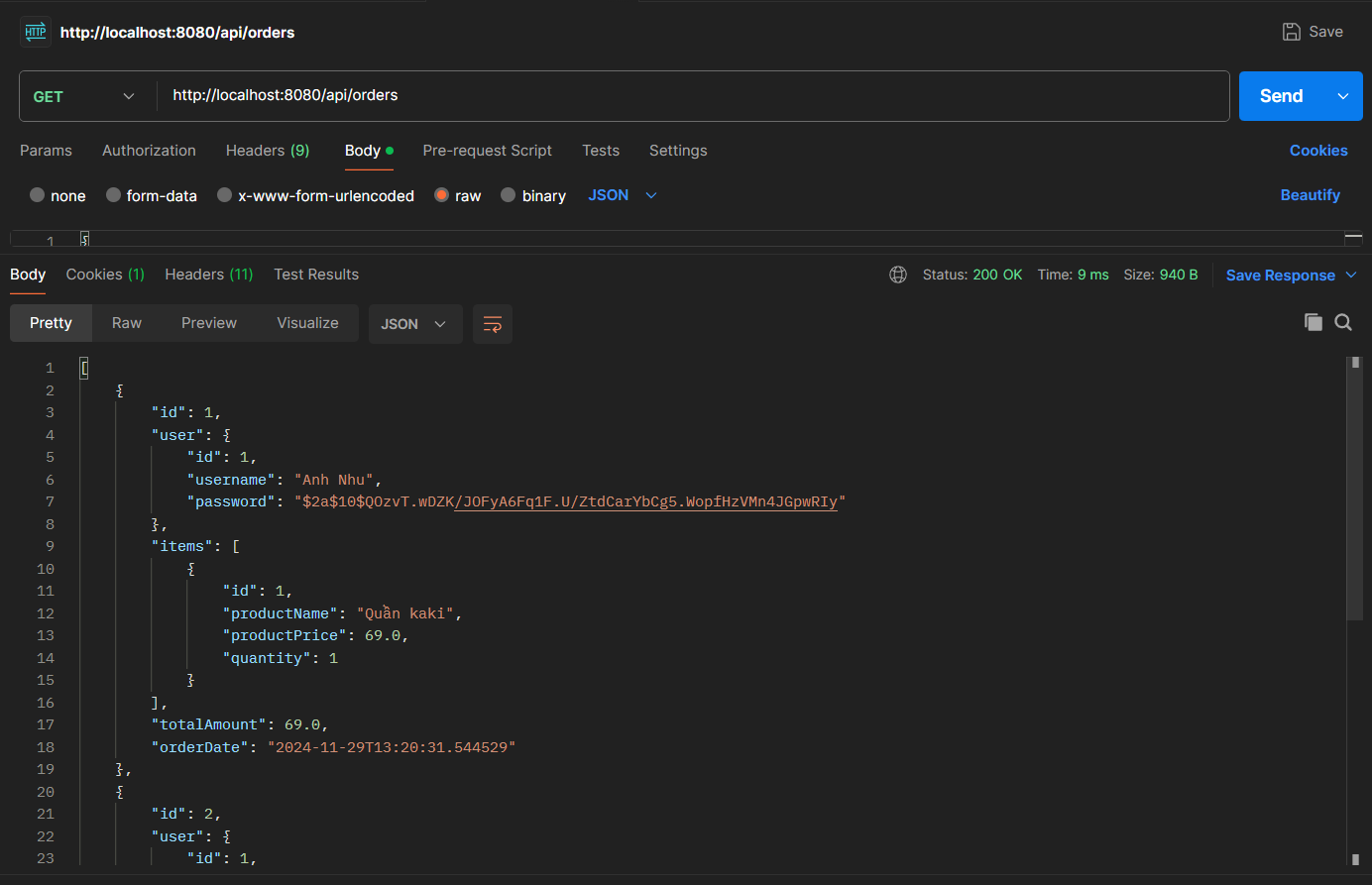


* Cuối cùng ta kiểm tra trên localhost:

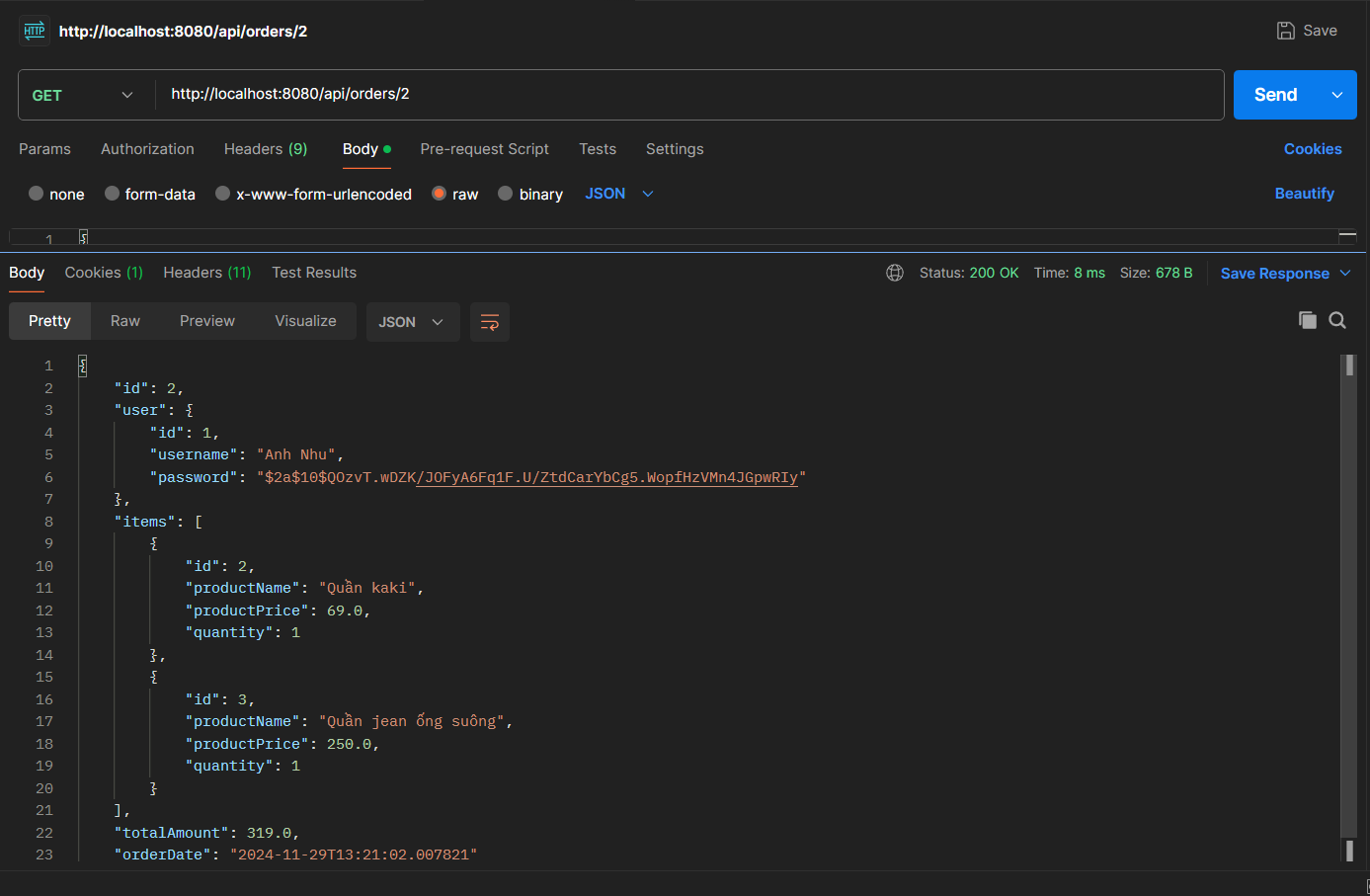


1. Đối với Order ta sẽ có các phương thức:

**GET**:<http://localhost:8080/api/orders>: dùng để hiện thị tất cả các hóa đơn có trong cơ sở dữ liệu

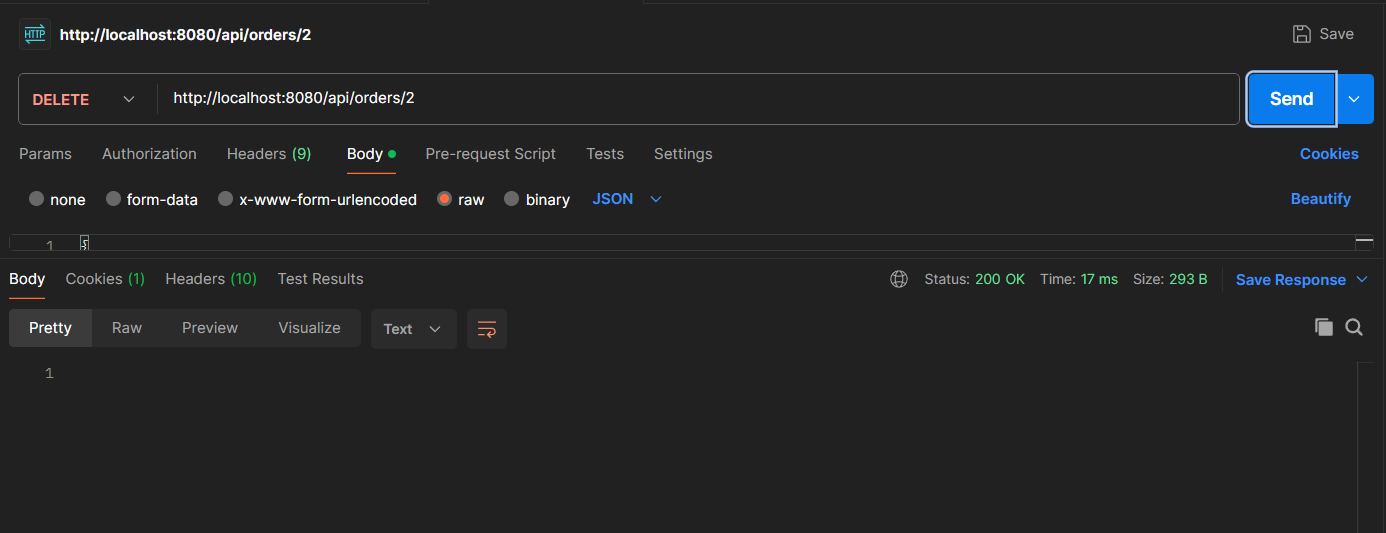


* <http://localhost:8080/api/orders/{id}>: dùng để hiện thị hóa đơn ứng với id của hóa đơn có trong cơ sở dữ liệu

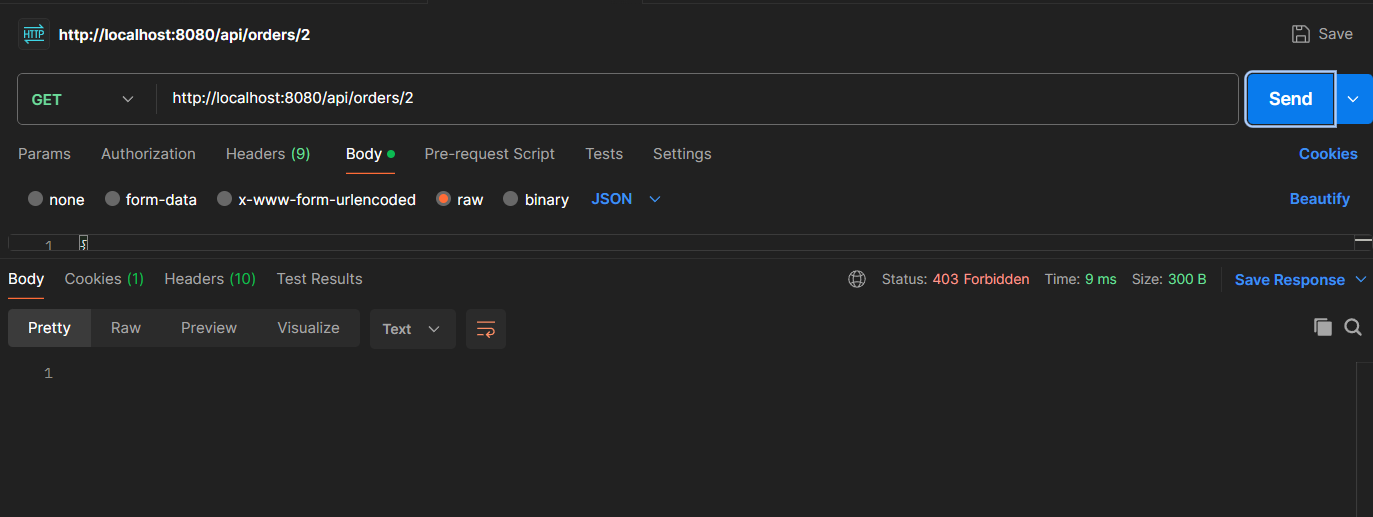


**DELETE**: <http://localhost:8080/api/order/{id}>: dùng để xóa hóa đơn ứng với id hóa đơn có trong cơ sở dữ liệu

* Ta sẽ sử dụng phương thức **DELETE** để xóa hóa đơn với id bất kì, ảnh dưới sẽ chọn id là 2:



* Sau khi xóa ta kiểm tra lại bằng phương thức **GET**:



* Cuối cùng ta kiểm tra trên localhost:

