Công nghệ Java

Tên dự án: Spring boot Ecommerce

Dự án này là một sản phẩm MVP cho một ứng dụng mua sắm giày trực tuyến được xây dựng với Java Spring Boot. Mục tiêu chính là cho phép khách hàng duyệt và lọc sản phẩm, thêm các mặt hàng vào giỏ hàng của họ và đặt hàng. Dự án được xây dựng bằng Spring Boot frameword cho Back-end, Thymeleaf cho Front-end và MySQL làm hệ thống quản lý cơ sở dữ liệu.

Dự án thực hiện theo một số nguyên tắc, mẫu thiết kế sau trong quá trình phát triển:

Dependence Injection (DI): Cơ chế DI của Spring boot được dùng để tiêm phụ thuộc vào các lớp.

Mô hình MVC: Phần mềm được phát triển tuân theo một mẫu thiết kế MVC cổ điến, trong đó mô hình, giao diện và điểu khiển được phân tách rõ ràng, làm cho code dễ bảo trì hơn.

RESTful API Design: Một số API trong ứng dụng tuân theo nguyên tắc thiết kế RESTful.

Responsive Web Design: dự án sử dụng thymeleaf, thiết kế web đáp ứng có thể được áp dụng bằng cách sử dụng các truy vấn phương tiện CSS để điều chỉnh bố cục và kiểu dáng của các chế độ xem HTML dựa trên thiết bị hoặc kích thước màn hình.

Object-Relational Mapping (ORM): ánh xạ dữ liệu từ cơ sở dữ liệu quan hệ sang các đối tượng. Trong dự án này sử dụng MYSQL làm cơ sở dữ liệu, nên ORM được áp dựng bằng cách sử dụng Dữ liệu Spring JPA để ánh xạ các đối tượng Java vào các bảng cơ sở dữ liệu và ngược lại.

Repository Pattern: nhận yêu cầu từ các layer Service và thao tác với cơ sở dữ liệu. Các interface Repository được kế thừa từ JpaRepository

Hibernate: kết nối cơ sở dữ liệu

Luồng thực hiện trong dự án: View ⭢ Controller ⭢ Service ⭢ Reponsitory ⭢Database

Cấu trúc dự án:

- com.CNJAVA.Midterm: package chứa lớp ứng dụng chính (MidtermApplication)

- com.CNJAVA.Midterm.controller: package chứa các lớp controller để xử lý các request

- com.CNJAVA.Midterm.entity: package chứa các lớp đối tượng đại diện cho mô hình dữ liệu

- com.CNJAVA.Midterm.repository: package chứa các lớp repository để thực hiện các hoạt động truy cập dữ liệu

- com.CNJAVA.Midterm.res: package chứa các lớp Rest Controller xử lí api

- com.CNJAVA.Midterm.service: package chứa các lớp service thực hiện thao tác logic.

- resources/static/css: chứa các file css

- resources/static/images: chứa các hình ảnh cố định, logo của trang web

- resources/static/js: chứa các file js và bootstrap

- resources/static/uploads: chứa các hình ảnh sản phẩm do admin upload khi tạo sản phẩm

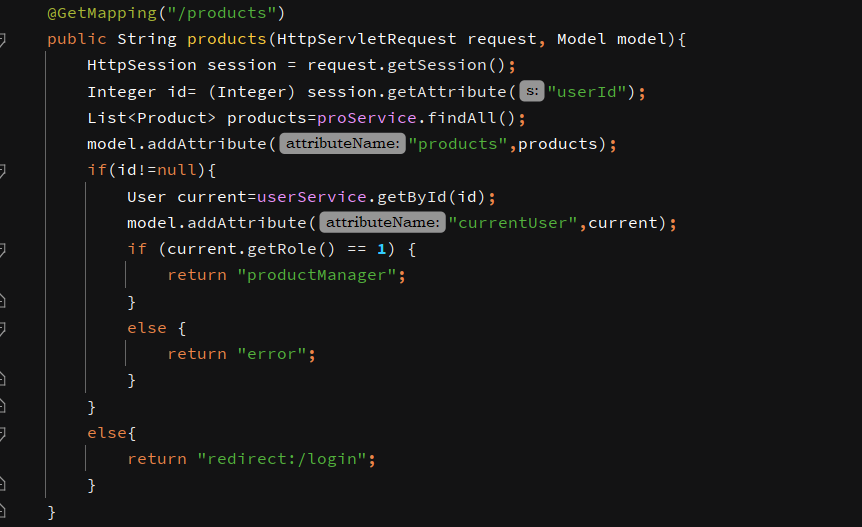
-resources/template:file html thiết kế giao diện các trang trên trang web

Gỉai thích code:

- Lớp AdminController:

+ Xử lý dữ liệu và trả về giao diện trang chủ admin khi người dùng truy cập đường dẫn có cú pháp /admin/ theo phương thức get

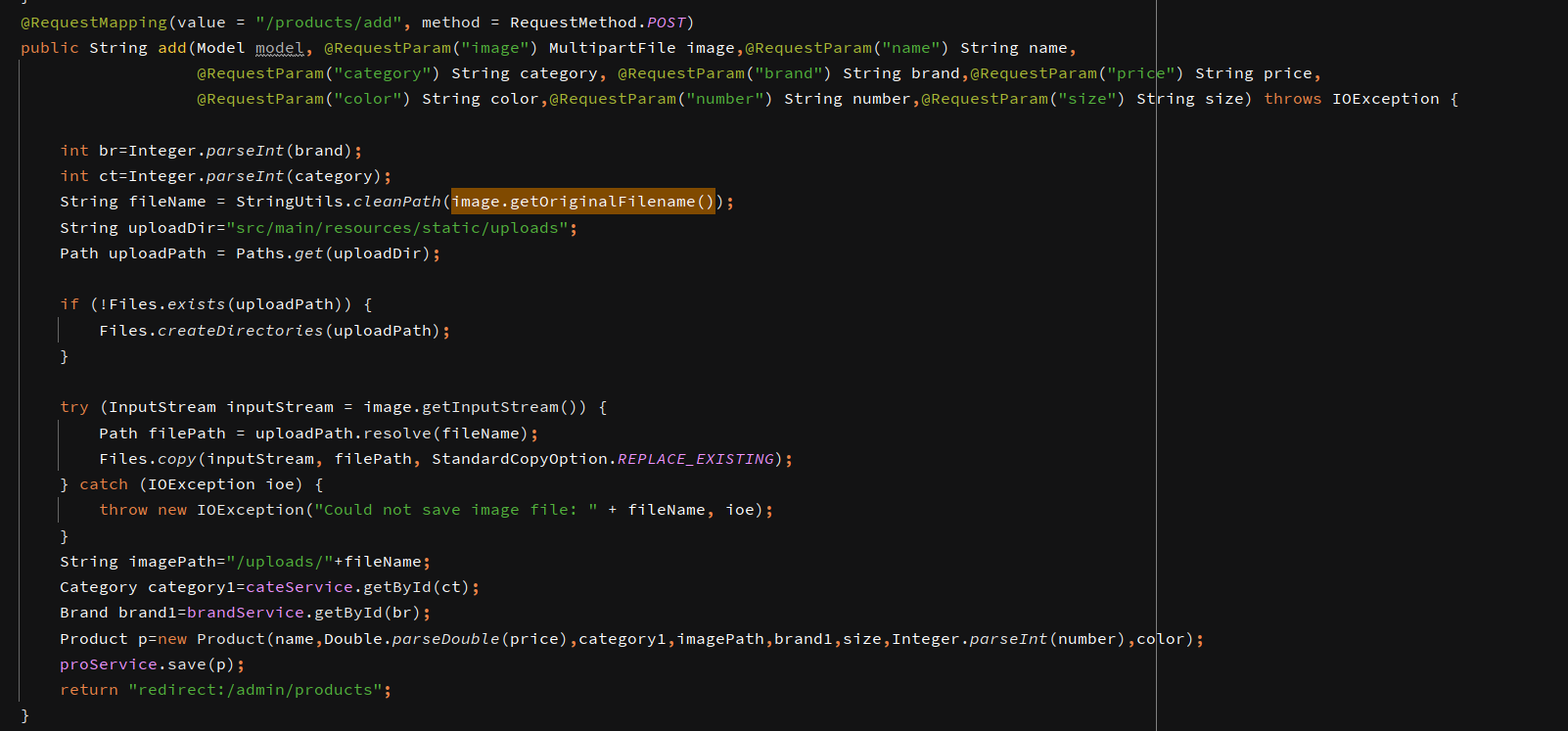


+ Lấy toàn bộ sản phẩm đang có (kể cả ngừng hoạt động) trả về giao diện quản lý sản phẩm

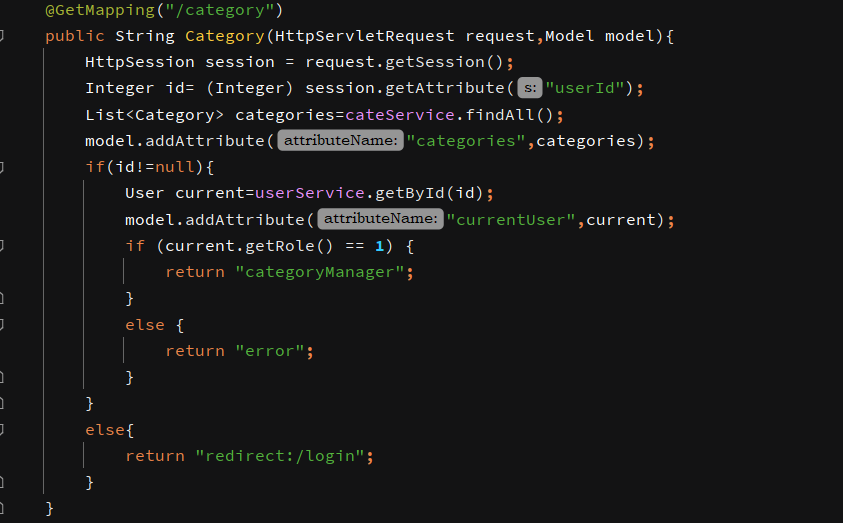
+ Xử lý yêu cầu và trả về giao diện thêm sản phẩm



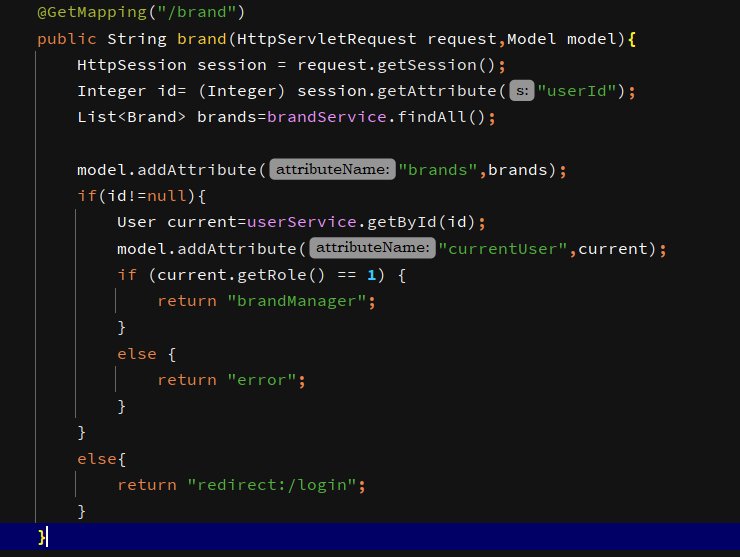
+ Xử lý form gửi từ người dùng, tải ảnh lên, lưu trữ hình ảnh, tạo và lưu sản phẩm mới vào cơ sở dữ liệu.



+Xử lý yêu cầu và trả dữ liệu là tất cả các danh mục đang có (kể cả ngừng hoạt động ) về trang quản lý danh mục

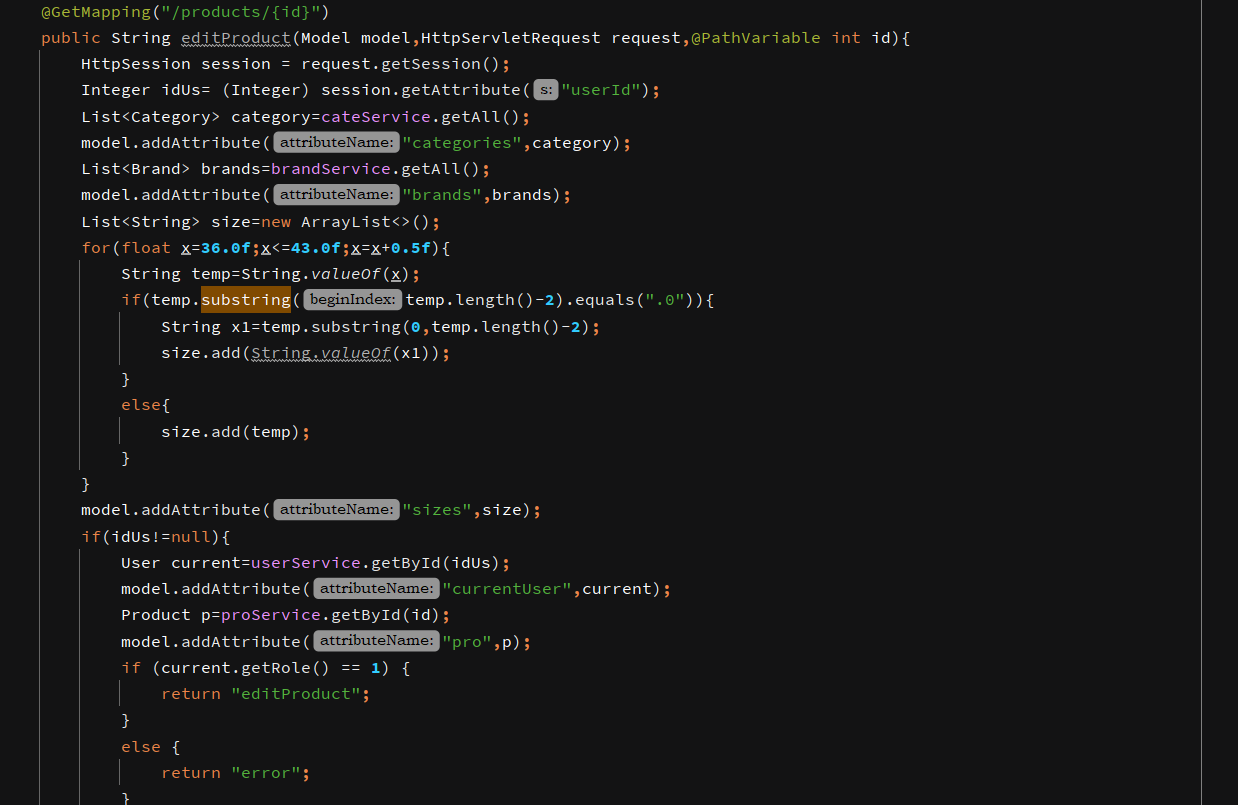


+ Xử lý yêu cầu và lấy tất cả dữ liệu về thương hiệu trả về trang quản lý thương hiệu

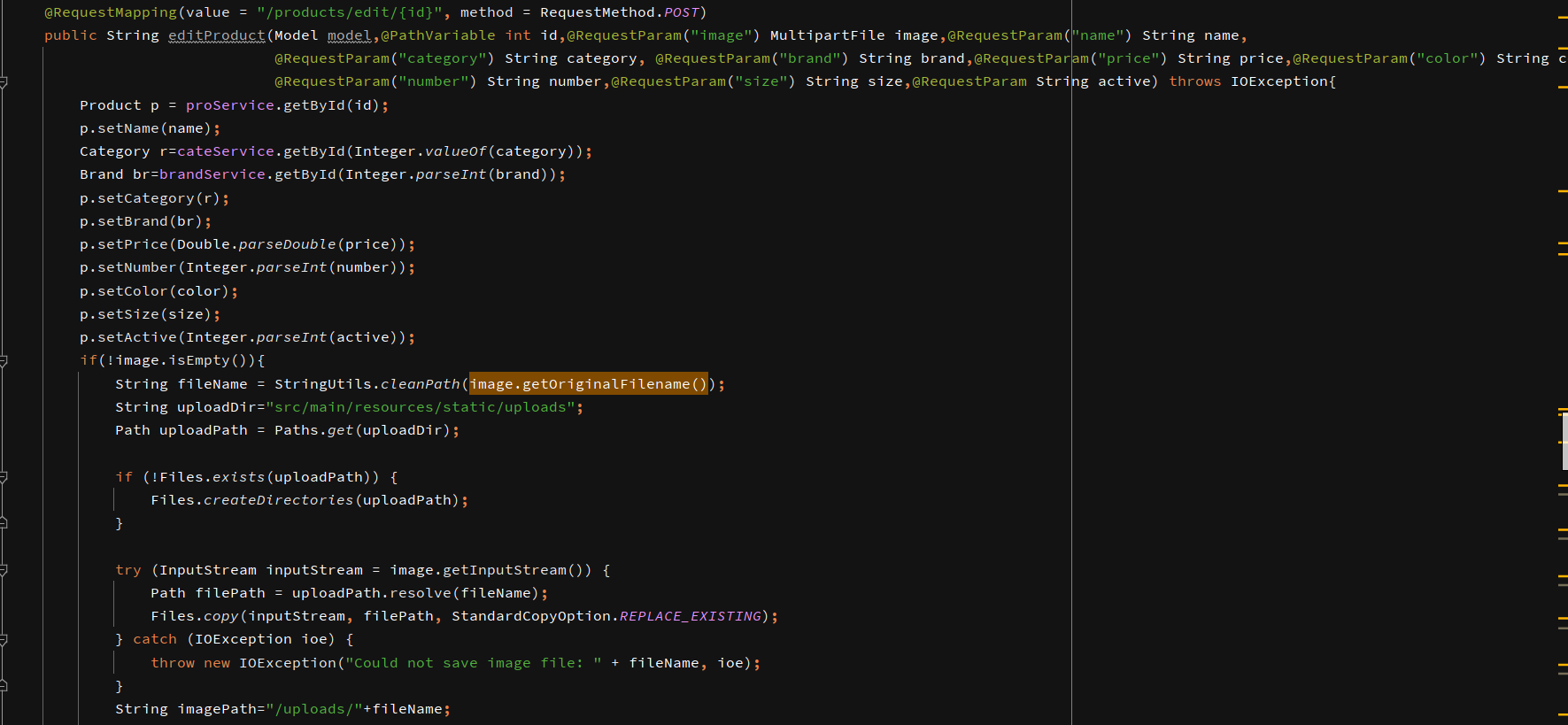


+ Xử lý yêu cầu và lấy tất cả dữ liệu về đơn đặt hàng trả về trang quản lý đơn đặt hàng

+ Xử lý yêu cầu và lấy dữ liệu sản phẩm theo id truyền qua từ đường dẫn và trả về trang chỉnh sửa sản phẩm



+ Xử lý yêu cầu khi người dùng gửi form thông tin sản phẩm đã được chỉnh sửa, cập nhật lại các trường thuộc tính, nếu hình ảnh thay đổi thì tiến hành tải lên và lưu trữ hình ảnh và cập nhật hình ảnh mới vào cơ sở dữ liệu, trả về trang quản lý sản phẩm sau khi cập nhật thành công



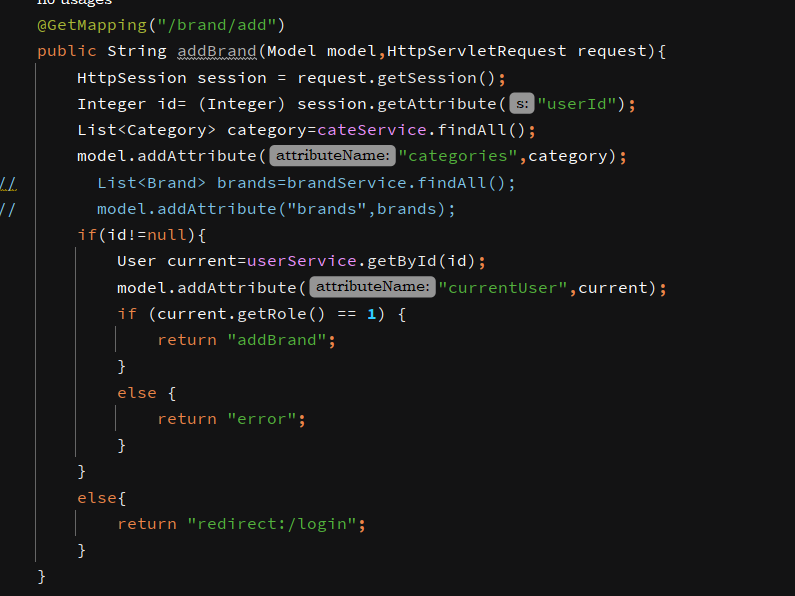
+Xử lý yêu cầu và hiển thị trang thêm danh mục



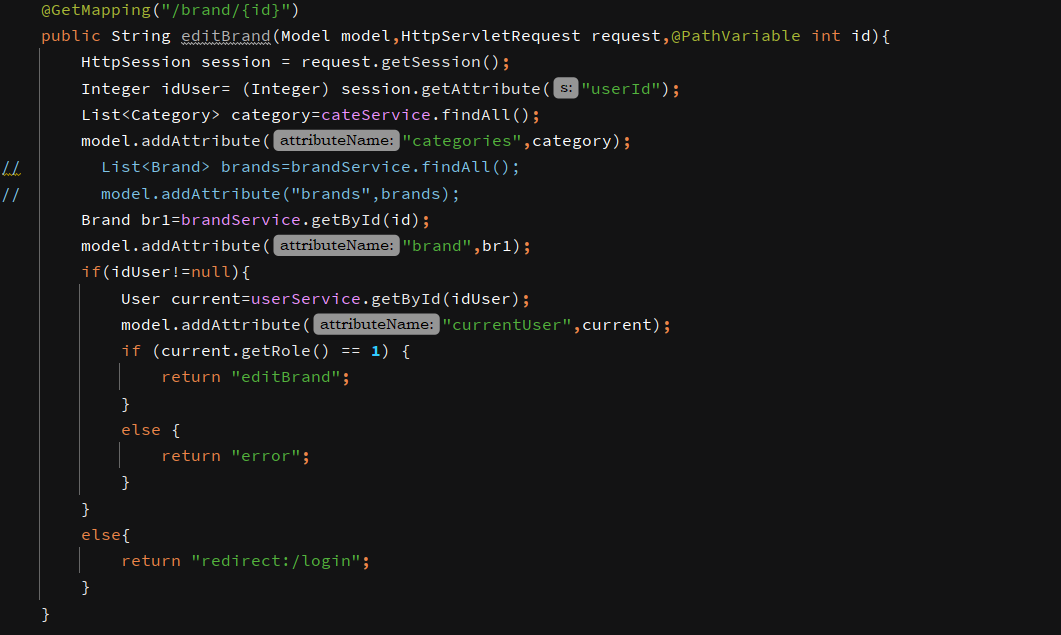
+ Xử lý yêu cầu, lấy danh mục theo id danh mục truyền từ đường dẫn và trả về trang chỉnh sửa danh mục



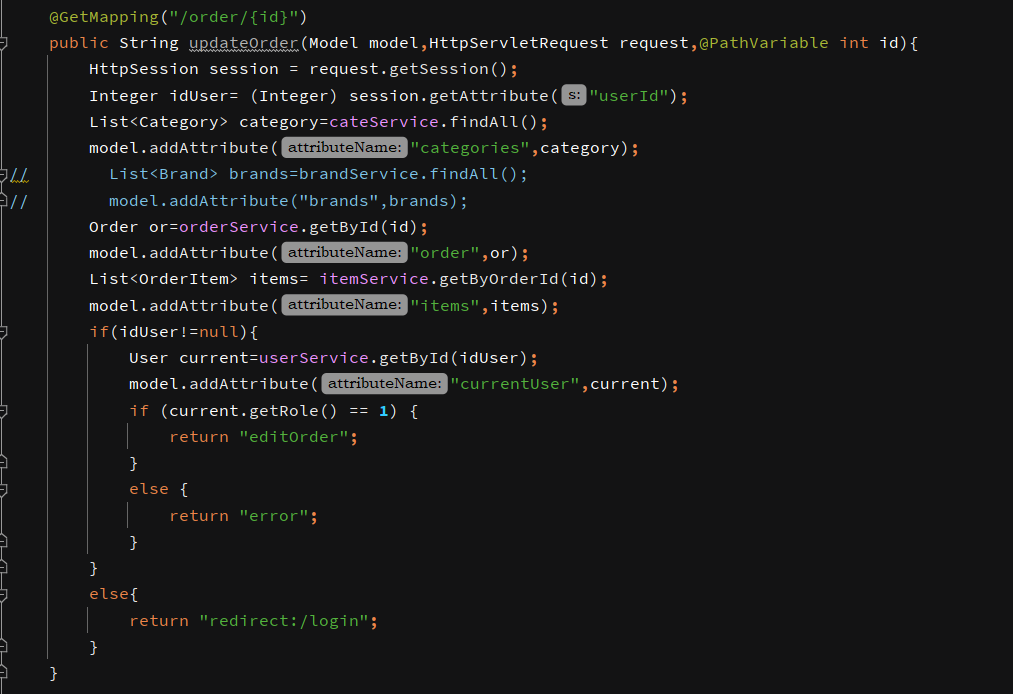
+ Xử lý và trả về giao diện trang thêm thương hiệu



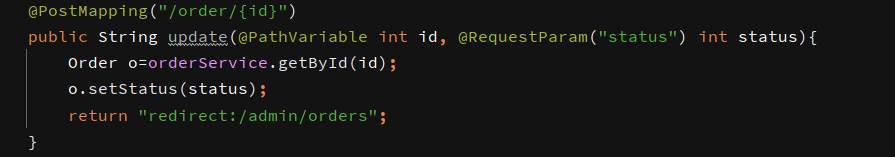
+ Xử lý yêu cầu và lấy thương hiệu theo id truyền từ đường dẫn và trả về trang chỉnh sửa thương hiệu



+ Xử lý yêu cầu, lấy thông tin đơn hàng từ id thông qua id truyền trên đường dẫn và trả về trang cập nhật đơn hàng

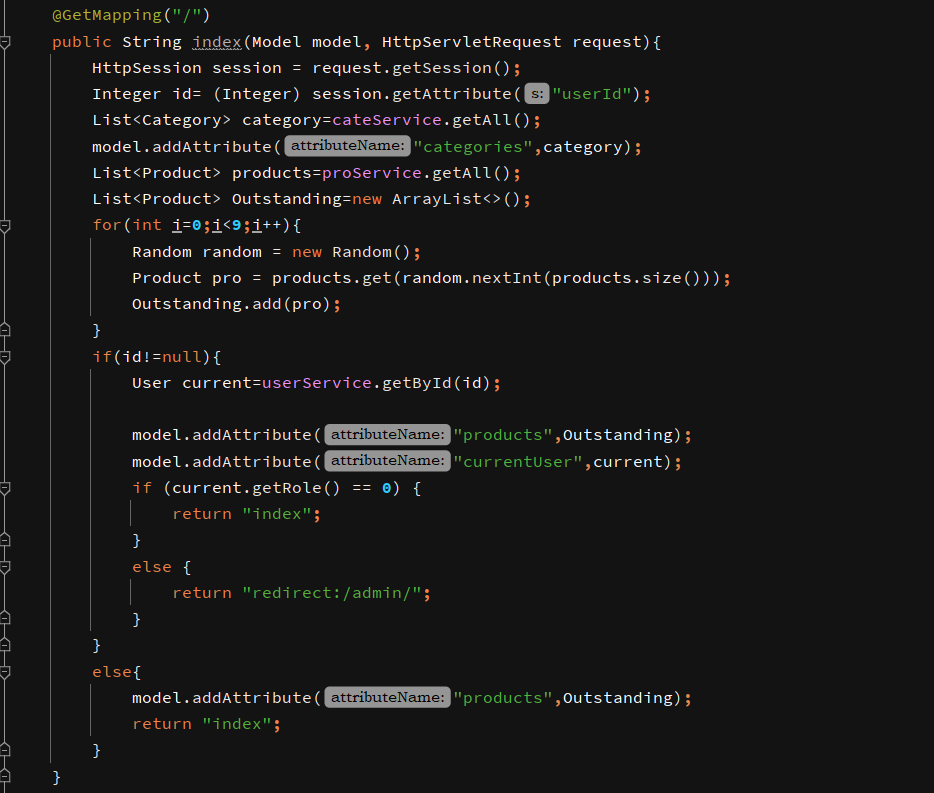


+ Nhận thông tin trạng thái đơn hàng và cập nhật vào đơn hàng có id truyền qua đường dẫn

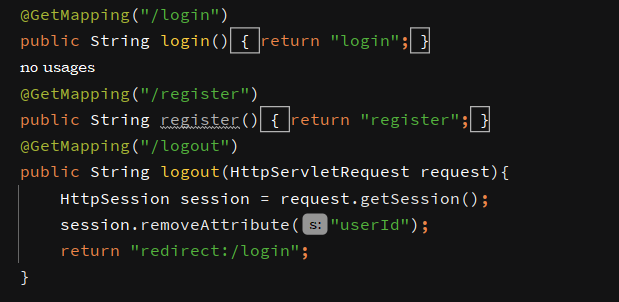


- HomeController

+ Xử lý yêu cầu, lấy các dữ liệu liên quan như người dùng hiện tại (current user), danh mục ( chỉ danh mục còn hoạt động) , 10 sản phẩm (còn hoạt động) và trả về trang chủ. Đây là trang chủ của người dùng là khách hàng, nếu người dùng là admin thì trả về trang chủ admin. Nếu người dùng chưa đăng nhập thì vẫn ở trang này.



+ Xử lý yêu cầu và trả về các giao diện đăng nhập, đăng ký và xử lý dữ liệu session khi người dùng đăng xuất

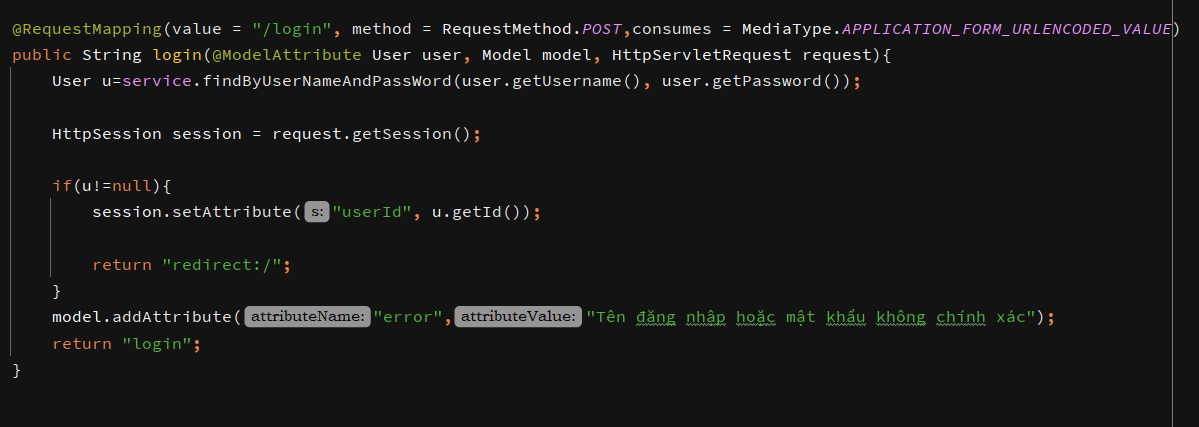


- UserController

+ Xử lý yêu cầu, nhận thông tin user từ form đăng ký và tiến hành kiểm tra số điện thoại, mật khẩu. Nếu các thông tin hợp lệ thì tạo và lưu mới một user vào cơ sở dữ liệu với vai trò khách hàng, sau khi đăng ký thành công thì chuyển hướng về trang đăng nhập.



+ Xử lý yêu cầu đăng nhập, nhận thông tin user, kiểm tra thông tin username và password do người dùng nhập với thông tin user trong cơ sở dữ liệu. Nếu tồn tại user trùng khớp với thông tin đăng nhập thì tiến hành lưu id người dùng vào session và chuyển hướng về trang chủ



- CategoryController

+ Xử lý yêu cầu, lấy các sản phẩm còn hoạt động theo mã danh mục truyền từ đường dẫn và trả về trang sản phẩm.

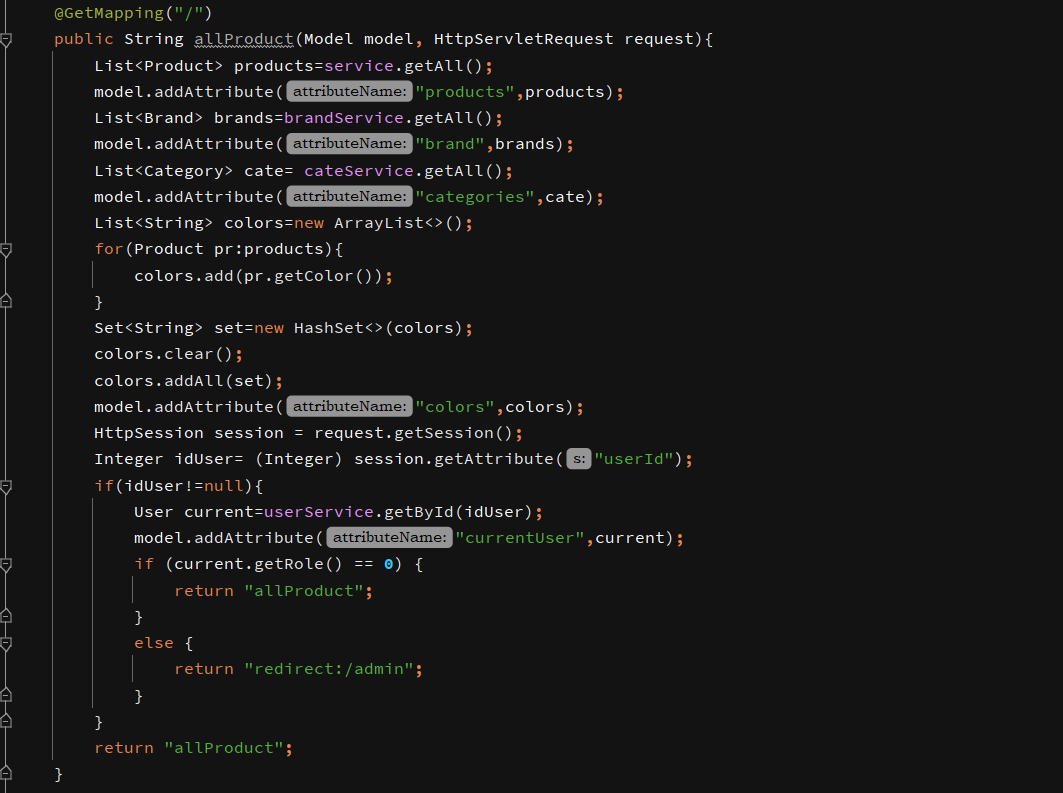


- ProductController

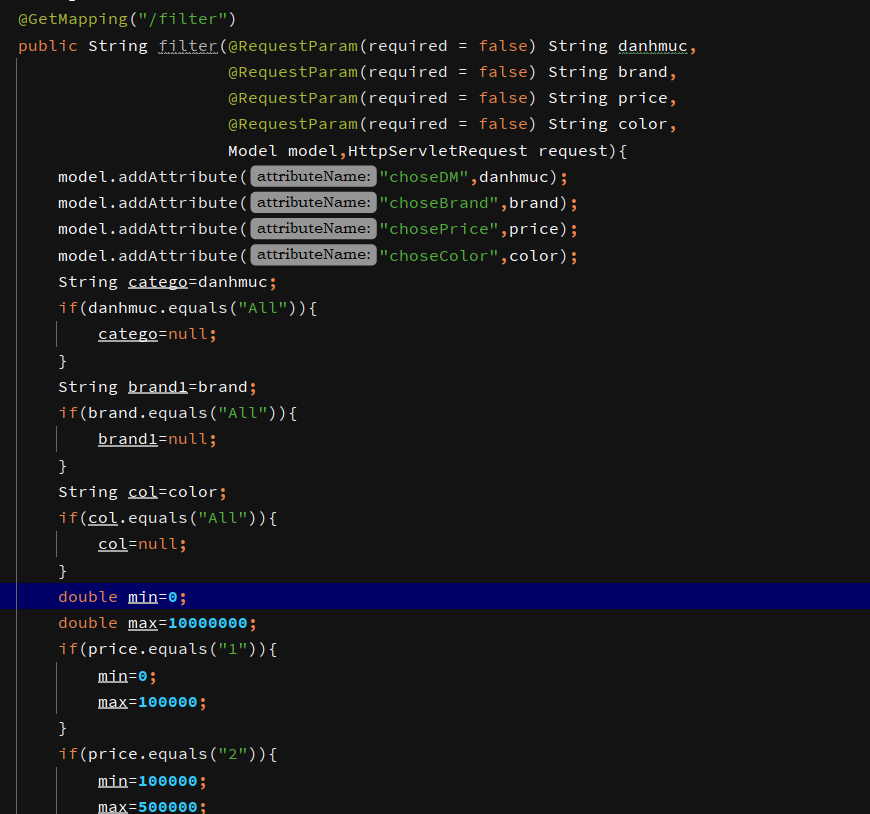
+ Xử lý yêu cầu, lấy thông tin các dữ liệu liên quan, thông tin sản phẩm thông qua id truyền từ đường dẫn và truyền về giao diện chi tiết sản phẩm. Có kiểm tra role của người dùng để trả về đúng giao diện



+ Xử lý yêu cầu, lấy toàn bộ sản phẩm, danh mục, thương hiệu còn hoạt động, và các dữ liệu về giá cả, màu sắc và truyền về trang tất cả danh mục. Có kiểm tra role của người dùng để trả về đúng giao diện

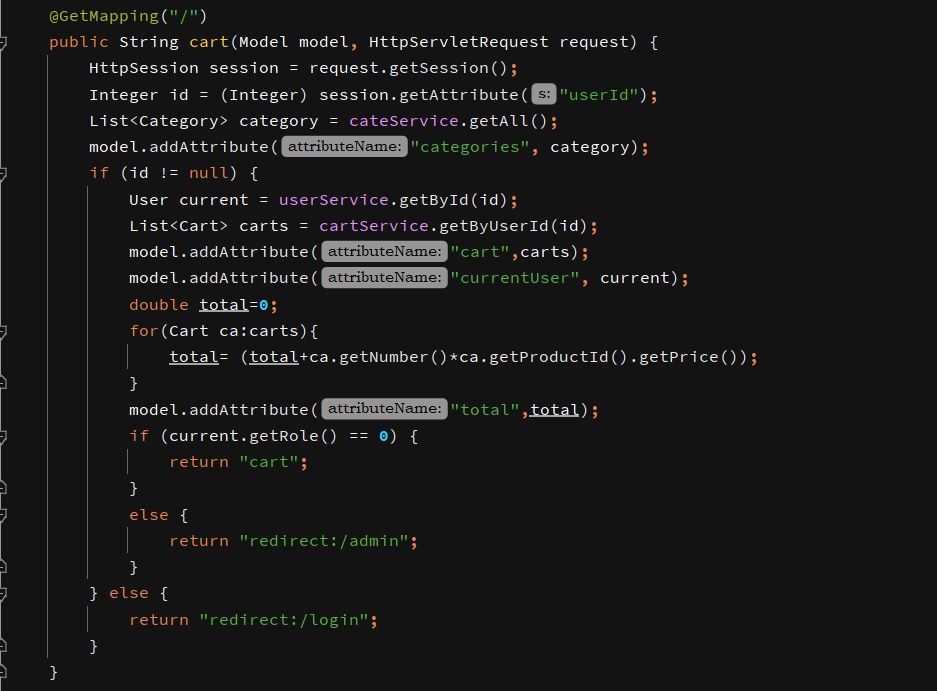


+ Xử lý yêu cầu tìm kiếm theo những lựa chọn của người dùng. Lọc theo danh mục, theo thương hiệu, khoảng giá và màu sắc. Lấy thông tin các sản phẩm phù hợp với yêu cầu người dùng và trả về trang sản phẩm.

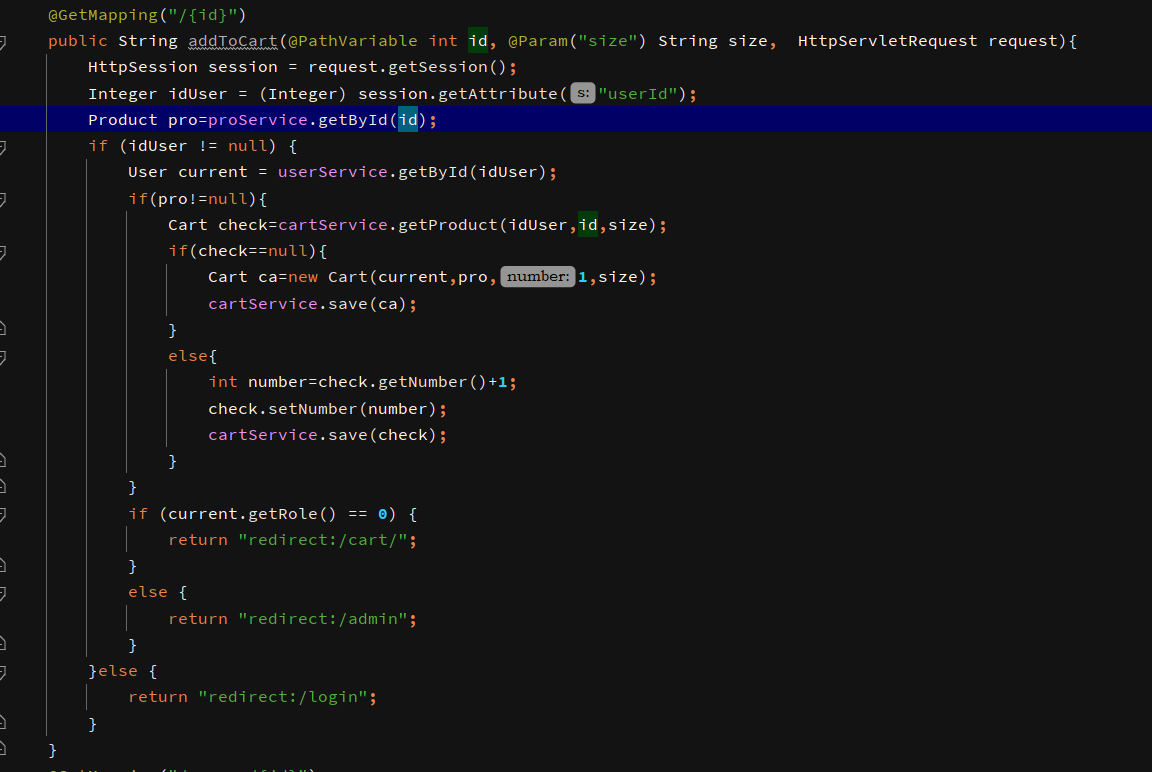




- CartController

+ Xử lý yêu cầu tới trang cart, Lấy thông tin người dùng từ session, nếu không có session userID thì chuyển về trang đăng nhập, tìm và lấy ra danh sách giỏ hàng của người dùng thông qua id người dùng. Trả về giao diện cart để hiện thị thông tin. 

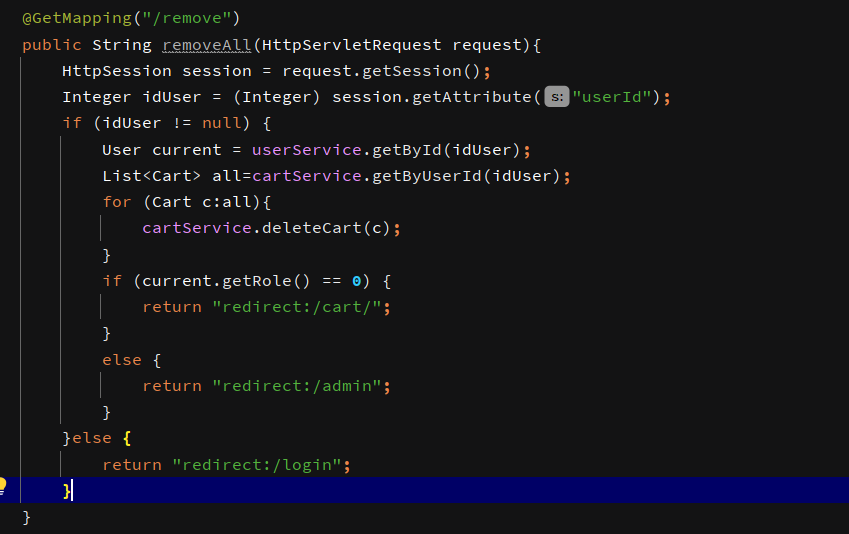
+ Xử lý yêu cầu thêm sản phẩm vào giỏ hàng thông qua id sản phẩm truyền từ đường dẫn. Lấy thông tin người dùng từ session, nếu không có session userID thì chuyển về trang đăng nhập. Nếu đã đăng nhập thì tiến hành lấy sản phẩm từ id sản phẩm và kiểm tra tồn tại trong giỏ hàng hay chưa. Nếu sản phẩm chưa tồn tại thì thêm mới vào giỏ hàng, nếu đã tồn tại thì tăng số lượng sản phẩm mà không thêm dòng mới.



+ Xử lý yêu cầu xóa dòng dữ liệu cart trong giỏ hàng thông qua id cart. Tìm dòng dữ liệu cart thông qua id và xóa khỏi cơ sở dữ liệu.



+ Xử lý yêu cầu xóa toàn bộ giỏ hàng.

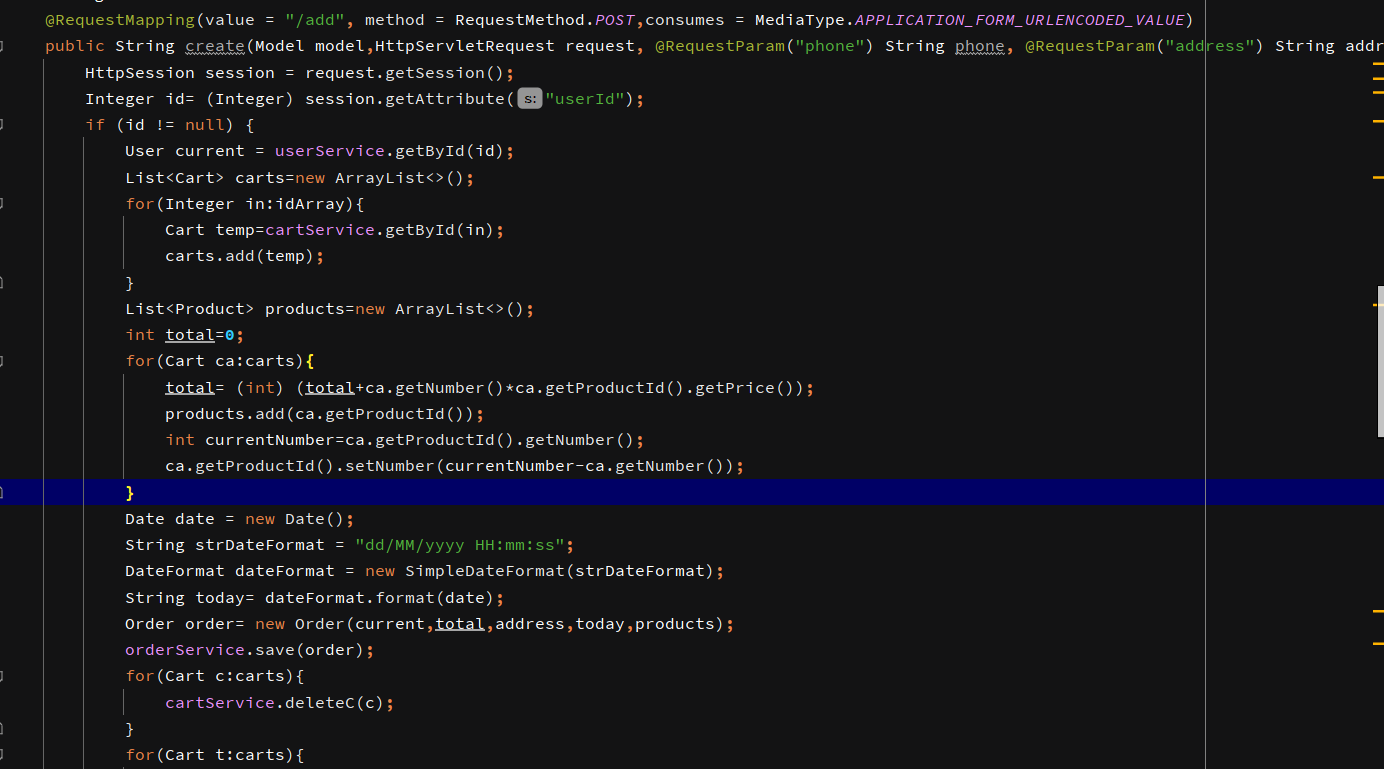


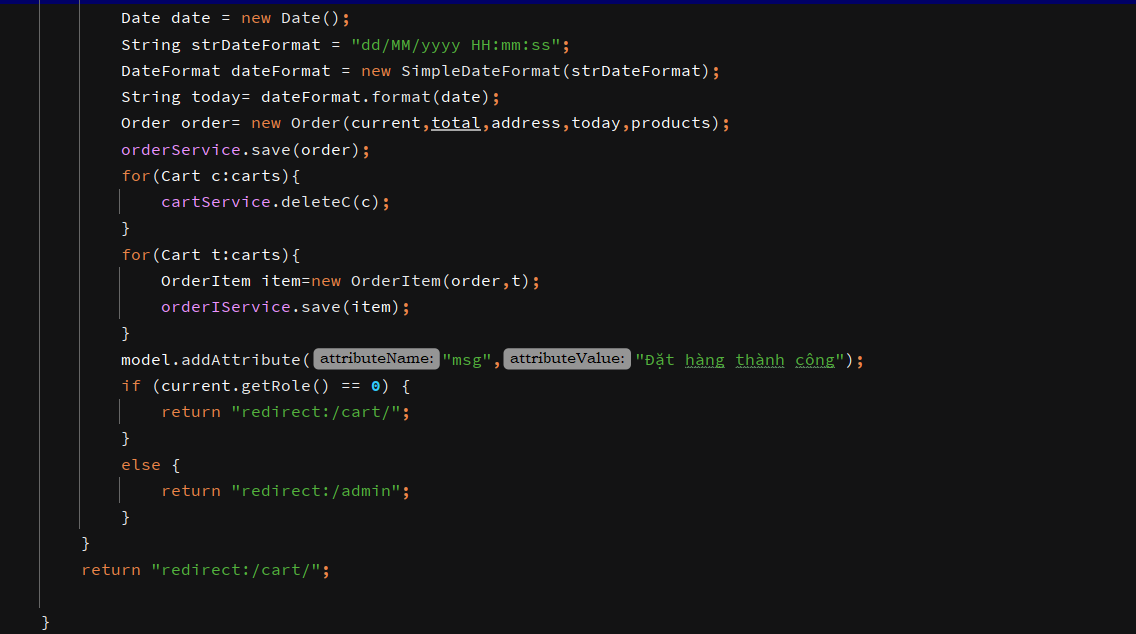
- OrderController

+ Xử lý yêu cầu, lấy thông tin dữ liệu liên quan đến cart. Từ id những dòng cart được người dùng chọn để thanh toán để lấy ra những thông tin cart và chuyển đến trang order để người dùng tiến hành kiểm tra đơn đặt, điền thông tin và đặt hàng.

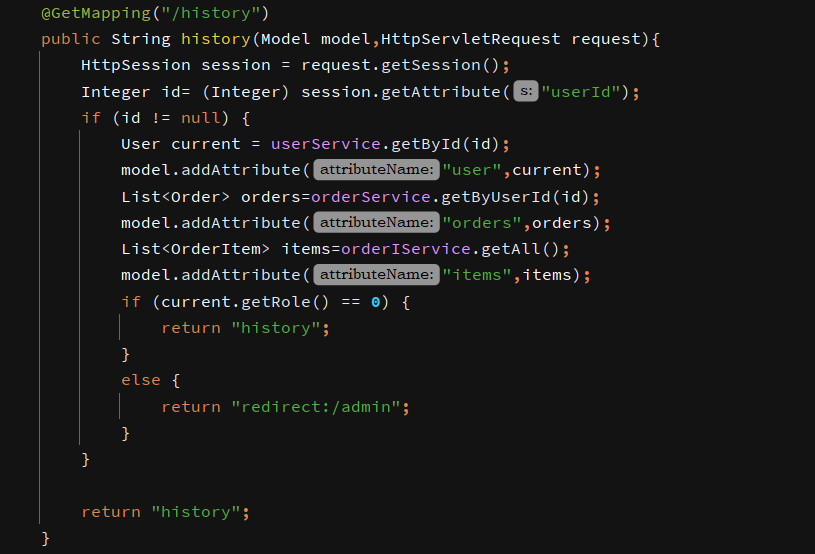


+ Xử lý yêu cầu đặt hàng và tạo ra đơn đặt hàng lưu vào cơ sở dữ liệu.

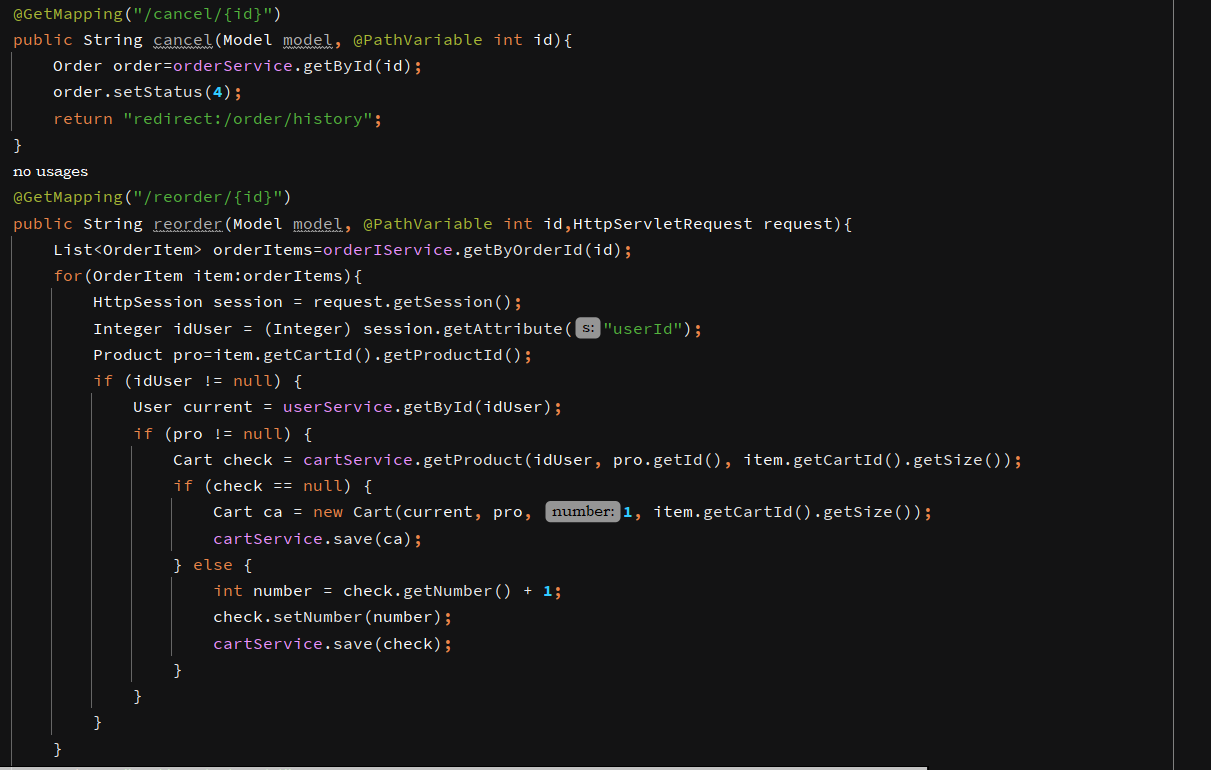




+ Xử lý yêu cầu người dùng muốn xem lịch sử các đơn đặt hàng. Lấy thông tin người dùng từ session và lọc các đơn đặt hàng theo id người dùng kèm theo là danh sách item của đơn đặt hàng đó và truyền về giao diện.

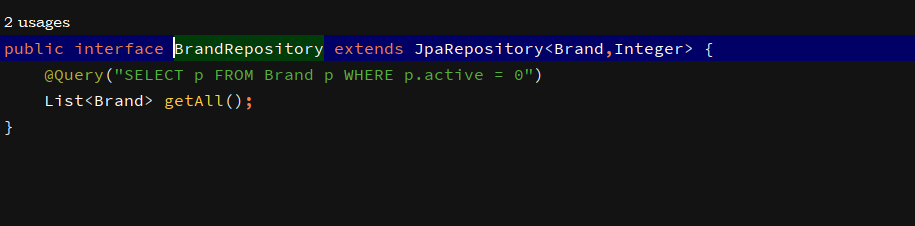


+ Xử lý yêu cầu người dùng muốn hủy đơn hàng hoặc đặt lại đơn hàng ( chỉ được đặt lại khi đơn hàng đã hủy hoặc giao thành công)



- BrandRepository

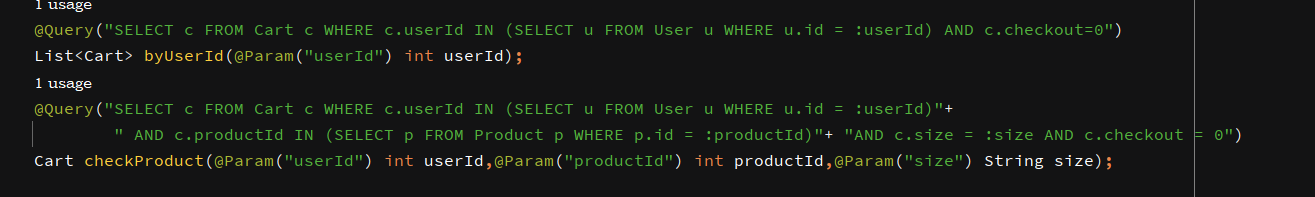
+ Query lấy tất cả sản phẩm còn hoạt động



- CartRepository

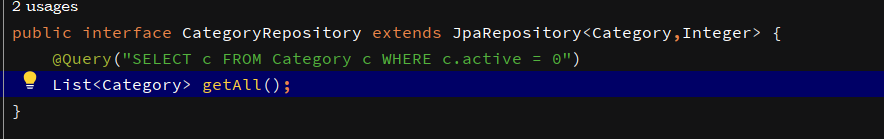
+ Query lấy tất cả các cart chưa đặt hàng theo id người dùng

+ Query lấy dòng dữ liệu cart với thông số id user, id product, size và chưa được đặt hàng (Dùng để kiểm tra sản phẩm với thông số đặt hàng là size đã có trong giỏ hàng của người dùng hay chưa)



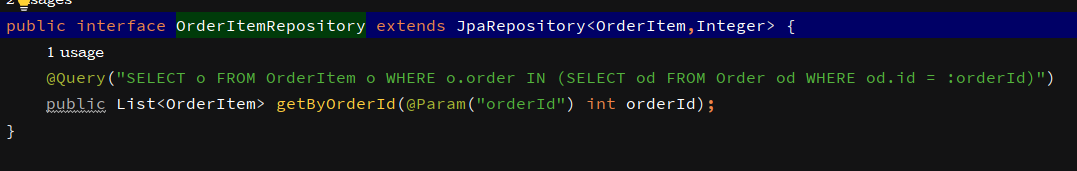
- CategoryRepository

+ Query lấy tất cả các danh mục còn hoạt động



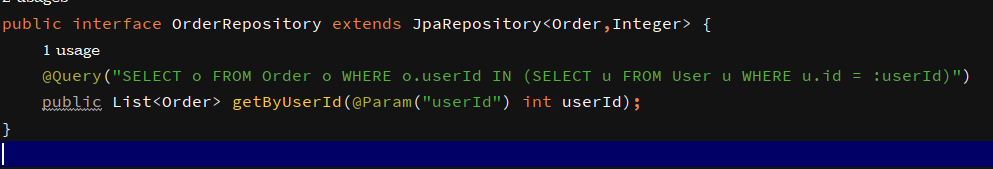
- OrderItemRepository

+Query lấy các dòng dữ liệu cart của 1 đơn đặt hàng theo id đơn đặt hàng



-OrderRepository

+Query lấy tất cả các đơn đặt hàng theo id người dùng



-ProductRepository

+ Query chọn những sản phẩm thỏa những yêu cầu của người dùng (lọc theo danh mục, thương hiệu, màu sắc, mức giá)

+ Query lấy tất cả sản phẩm theo id danh mục

+ Query lấy tất cả sản phẩm còn hoạt động

+ Query lây tất cả sản phẩm theo id thương hiệu (cả hoạt động và ngừng hoạt động)

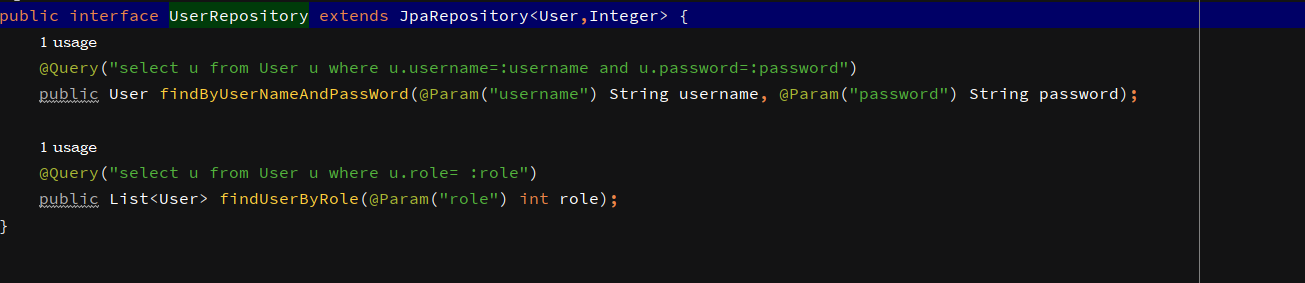
+ Query Query lấy tất cả sản phẩm theo id danh mục và còn hoạt động



- UserRepository

+ Query lấy user theo username và password

+ Query lấy user theo role



Để chạy ứng dụng trên máy tính cục bộ, hãy làm theo các bước sau:

Điều kiện chạy:

- Java JDK 8 trở lên

- MySQL hoặc Postgres database

- Git

Các bước chạy dự án:

1. Clone dự án từ github hoặc gitlab:

Git hub: <https://github.com/yvyt/52000170_Java_Midterm.git>

Git lab: <https://gitlab.duthu.net/S52000170/52000170_java_midterm.git>

2. Chuyển hướng đến thư mục code vừa clone về

3. Chỉnh sửa file application.properties để phù hớp với cấu hình cơ sở dữ liệu:

spring.datasource.url=jdbc:mysql://{database-host}:{port}/{database-name}?useSSL=false&serverTimezone=UTC

spring.datasource.username={database-username}

spring.datasource.password={database-password}

3. import file springcommerce.sql vào database

4. build và run chương trình:

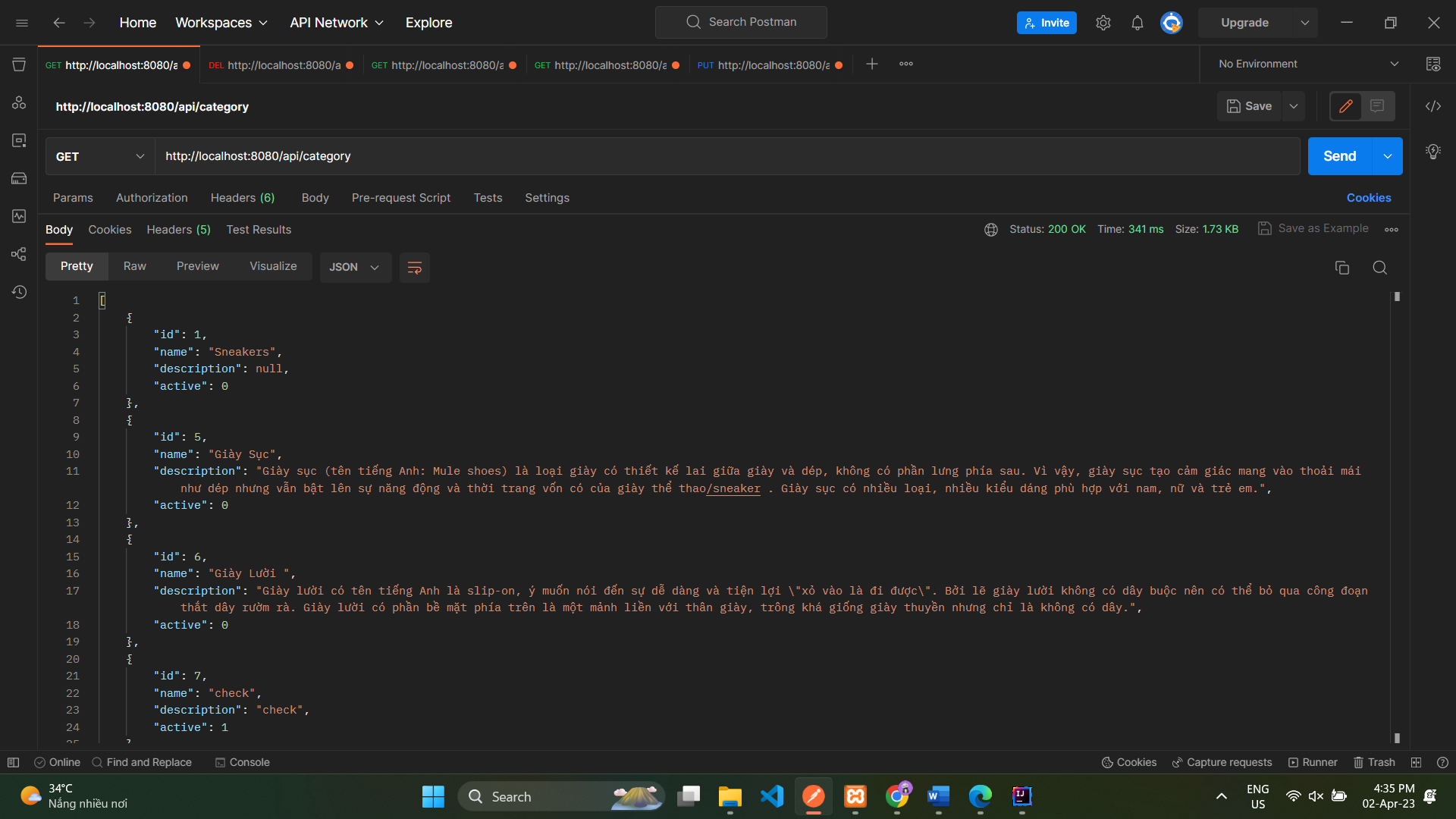
mvnw spring-boot:run

Minh chứng API

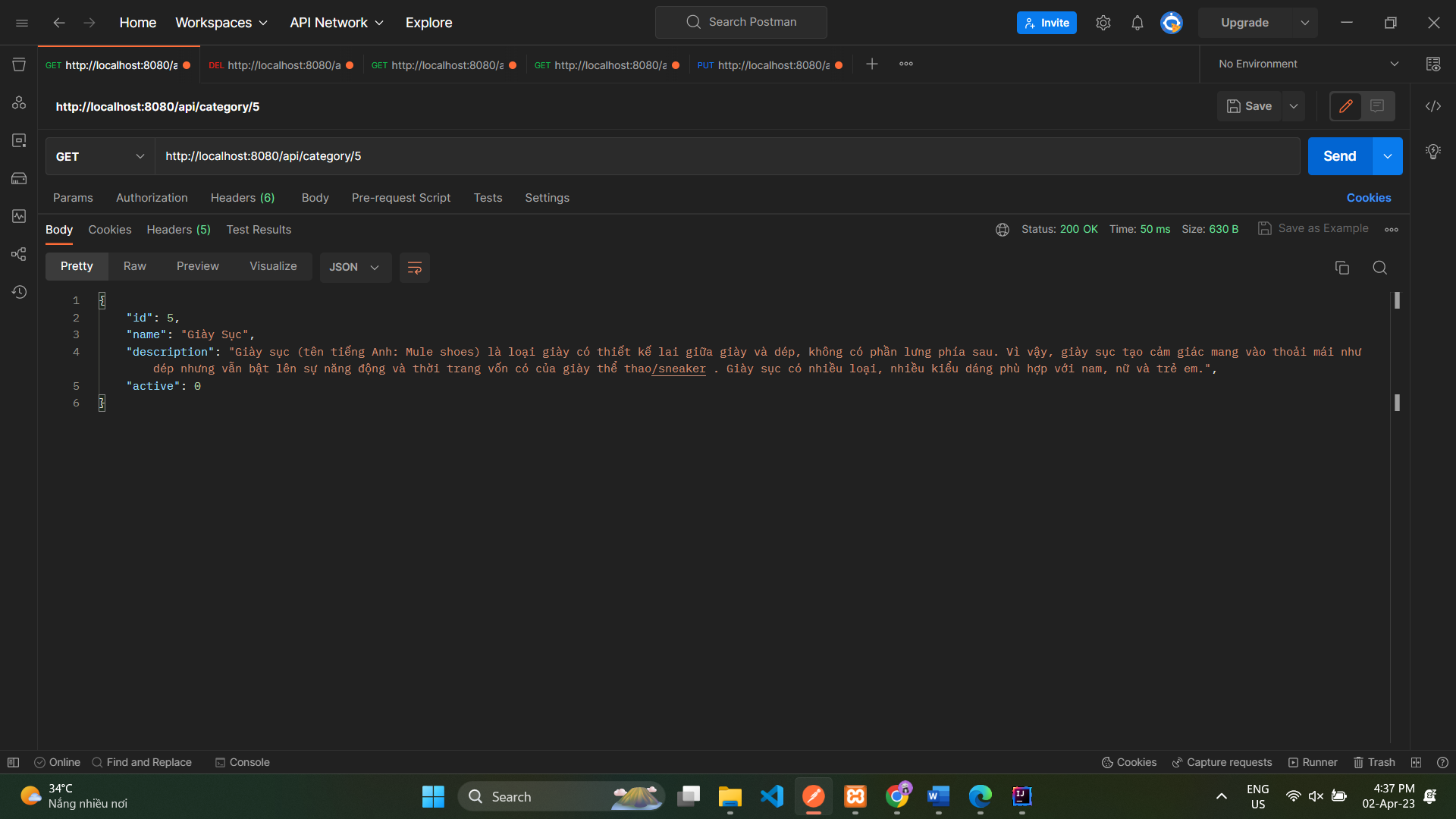
- dự án sử dụng API để thêm, xóa, sửa category và brand của sản phẩm:

Category:

GET: /api/category



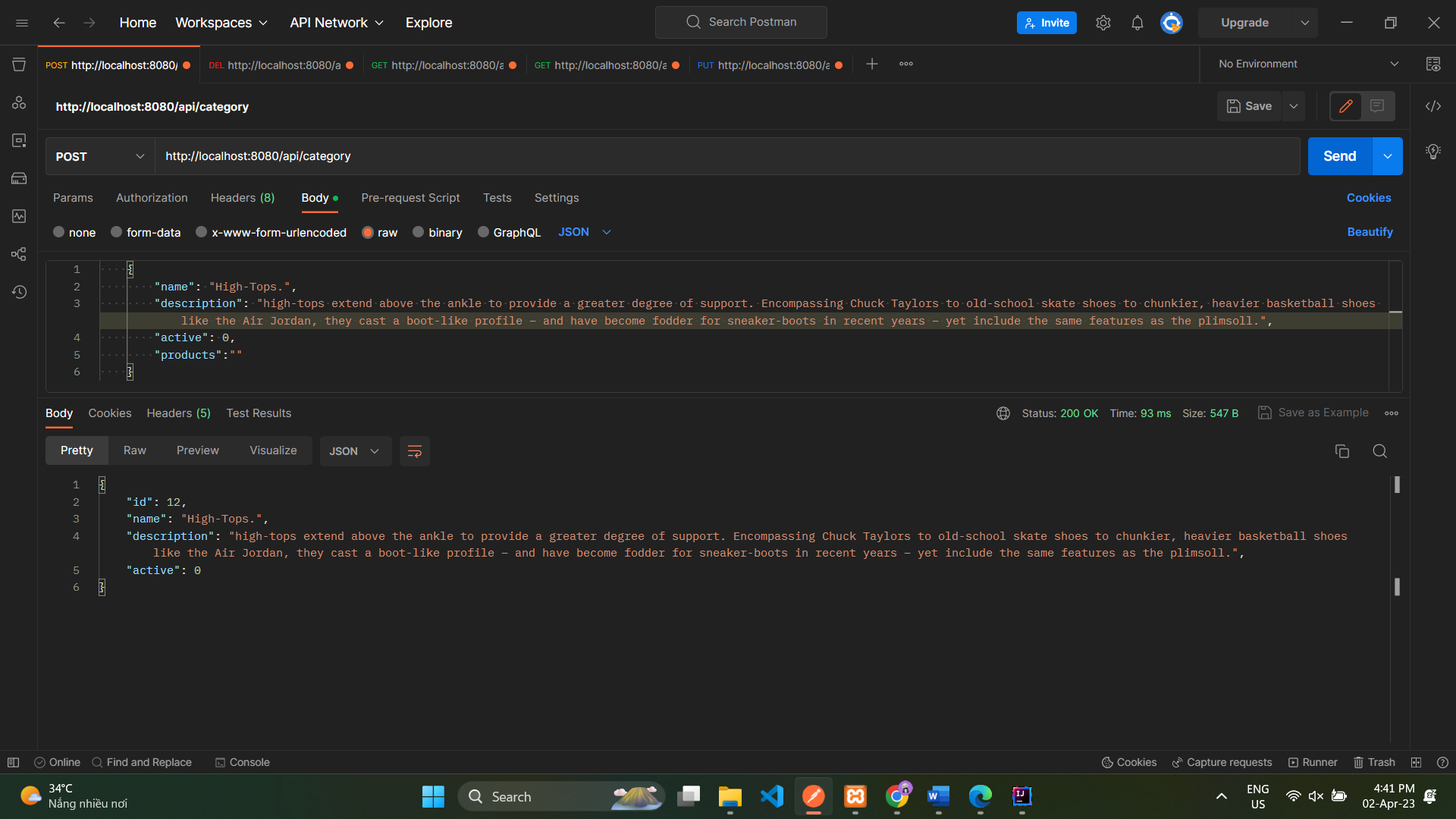
GET: /api/category/{id}



PUT: /api/category/{id}

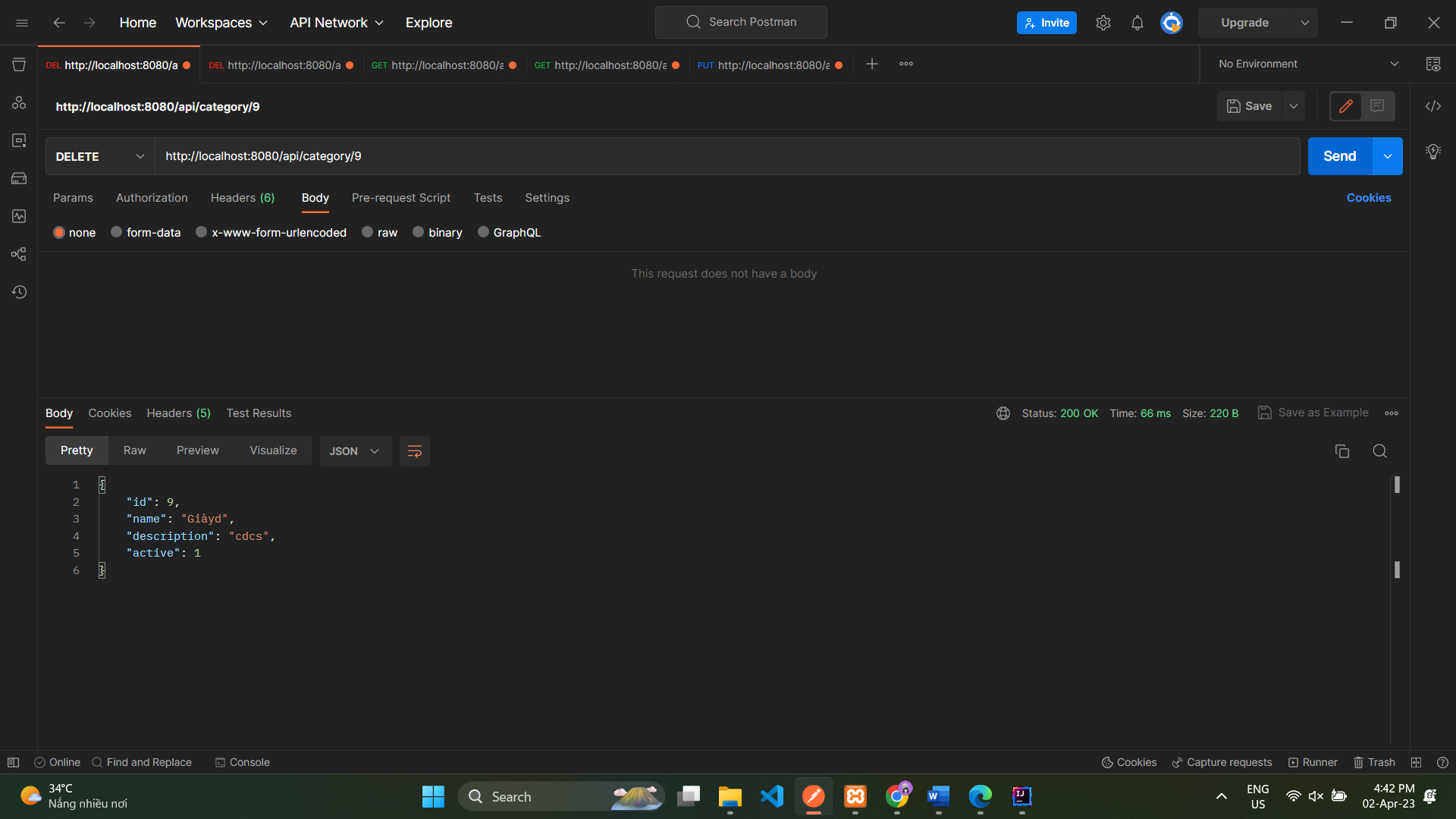


POST: /api/category



DELETE: /api/category/{id}

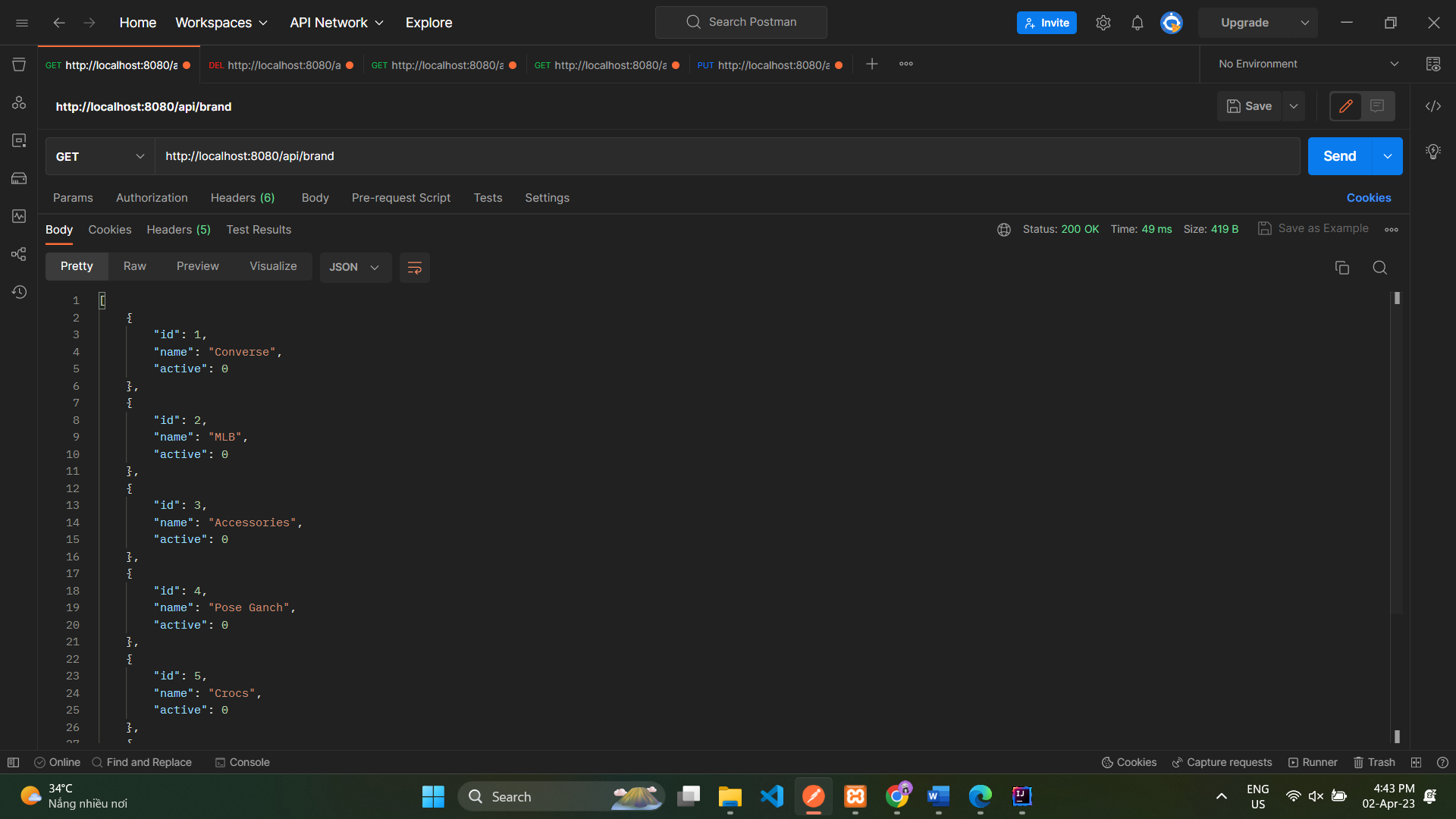
(Khi xóa 1 danh mục, nó sẽ không mất đi khỏi database, nó chỉ được đánh dấu đã xóa)



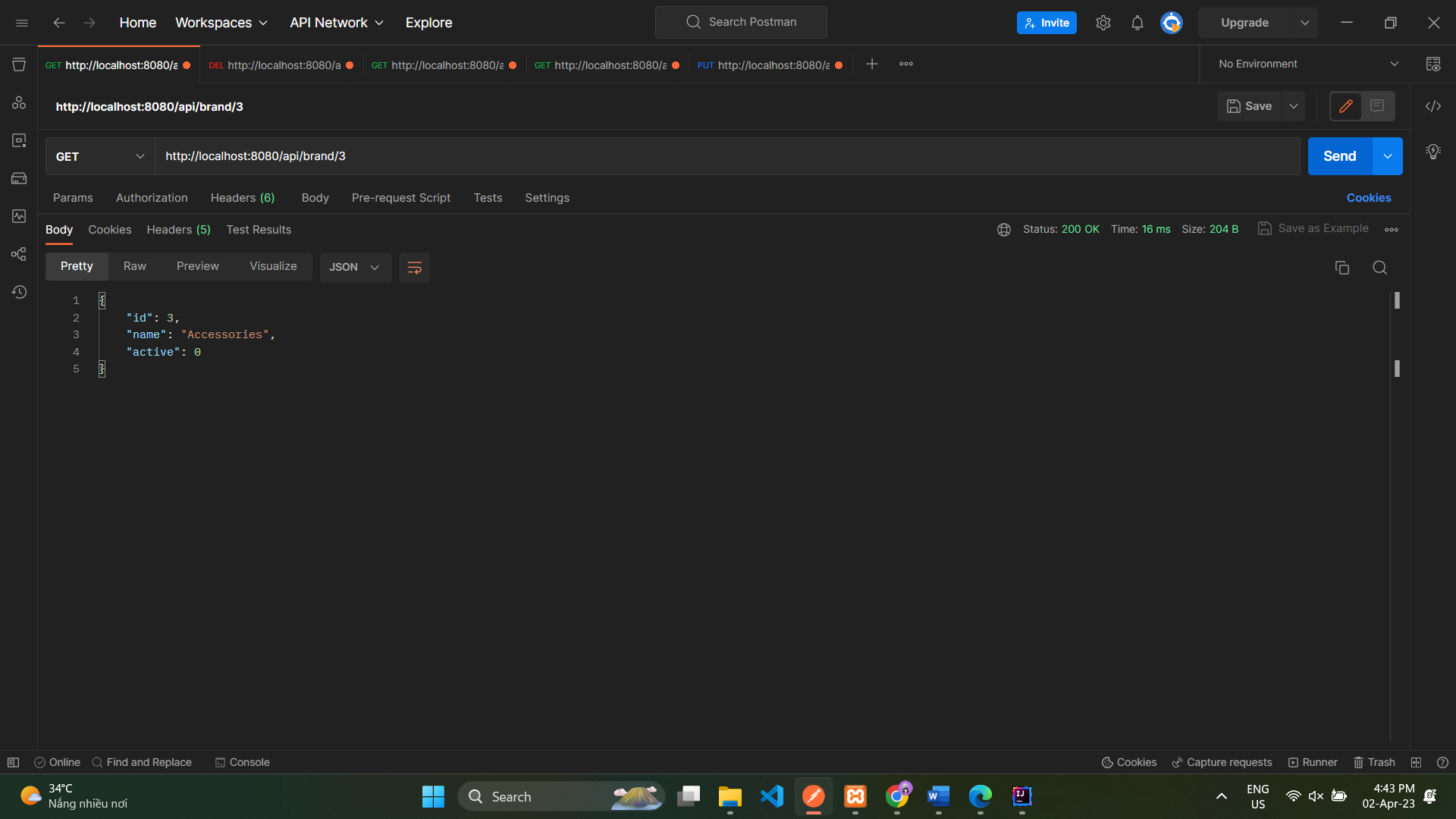
Khi active =1 tức là danh mục đã xóa (ngừng hoạt động)

Brand:

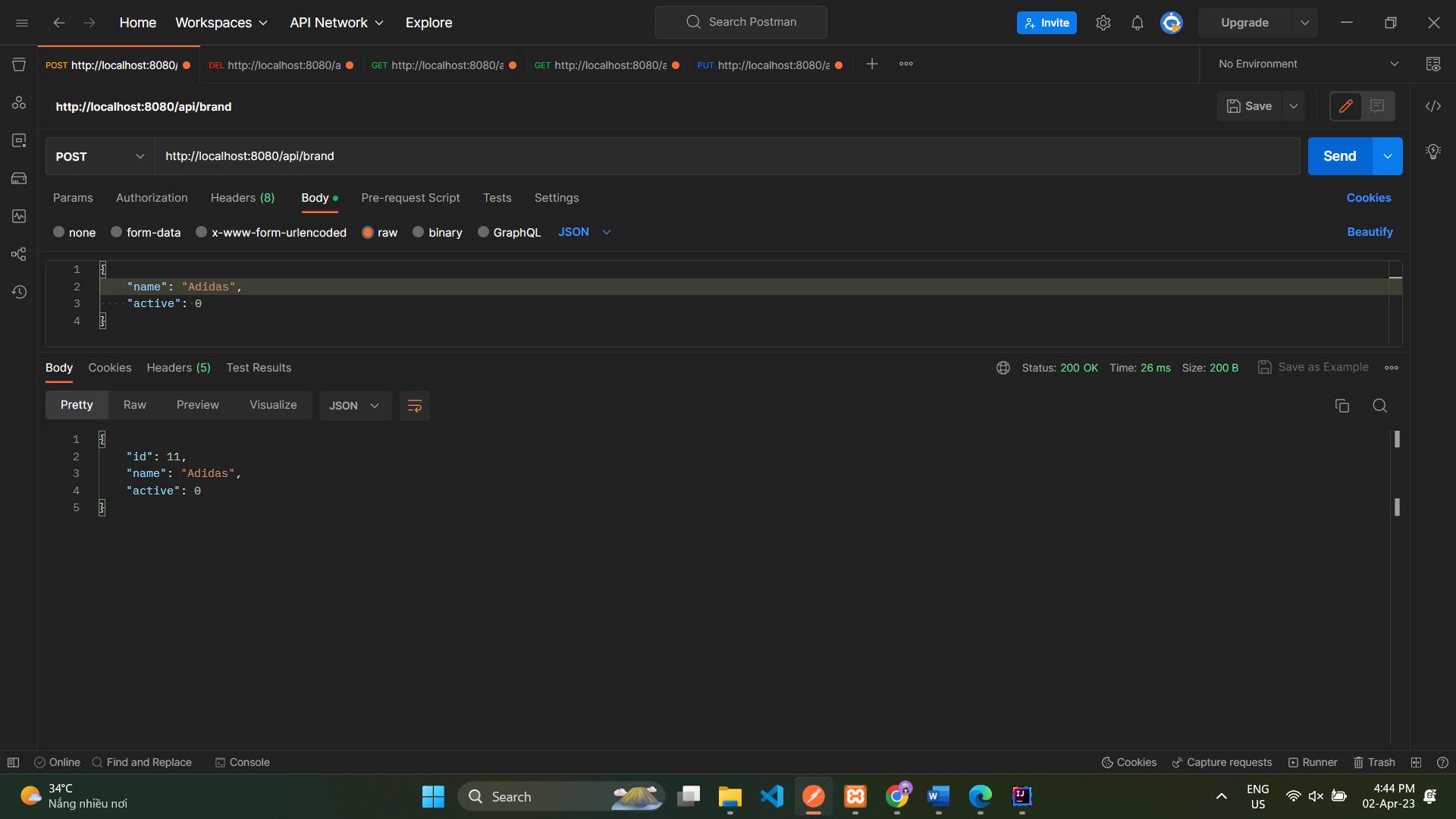
GET: /api/brand



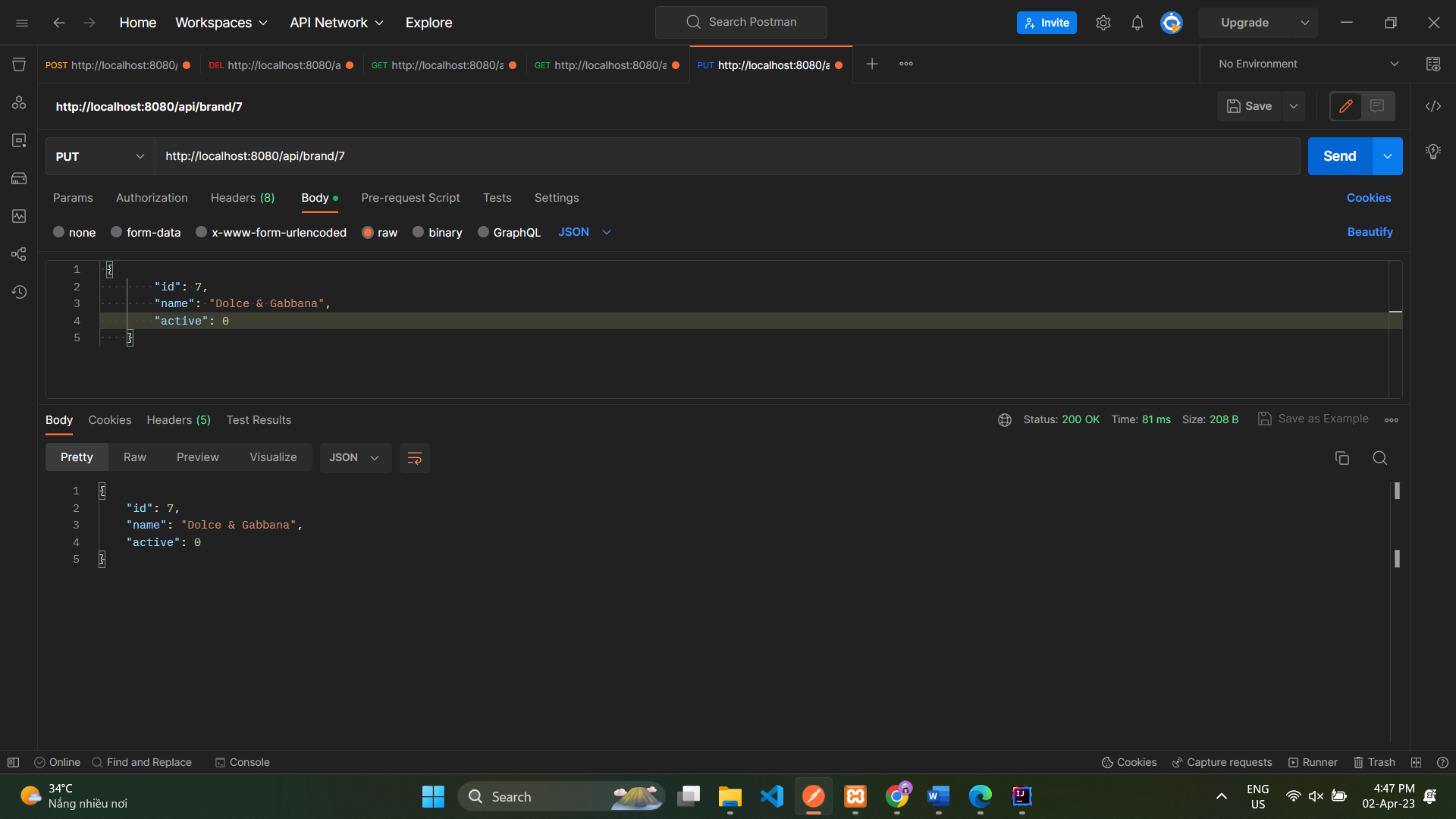
GET: /api/brand/{id}



POST: /api/brand

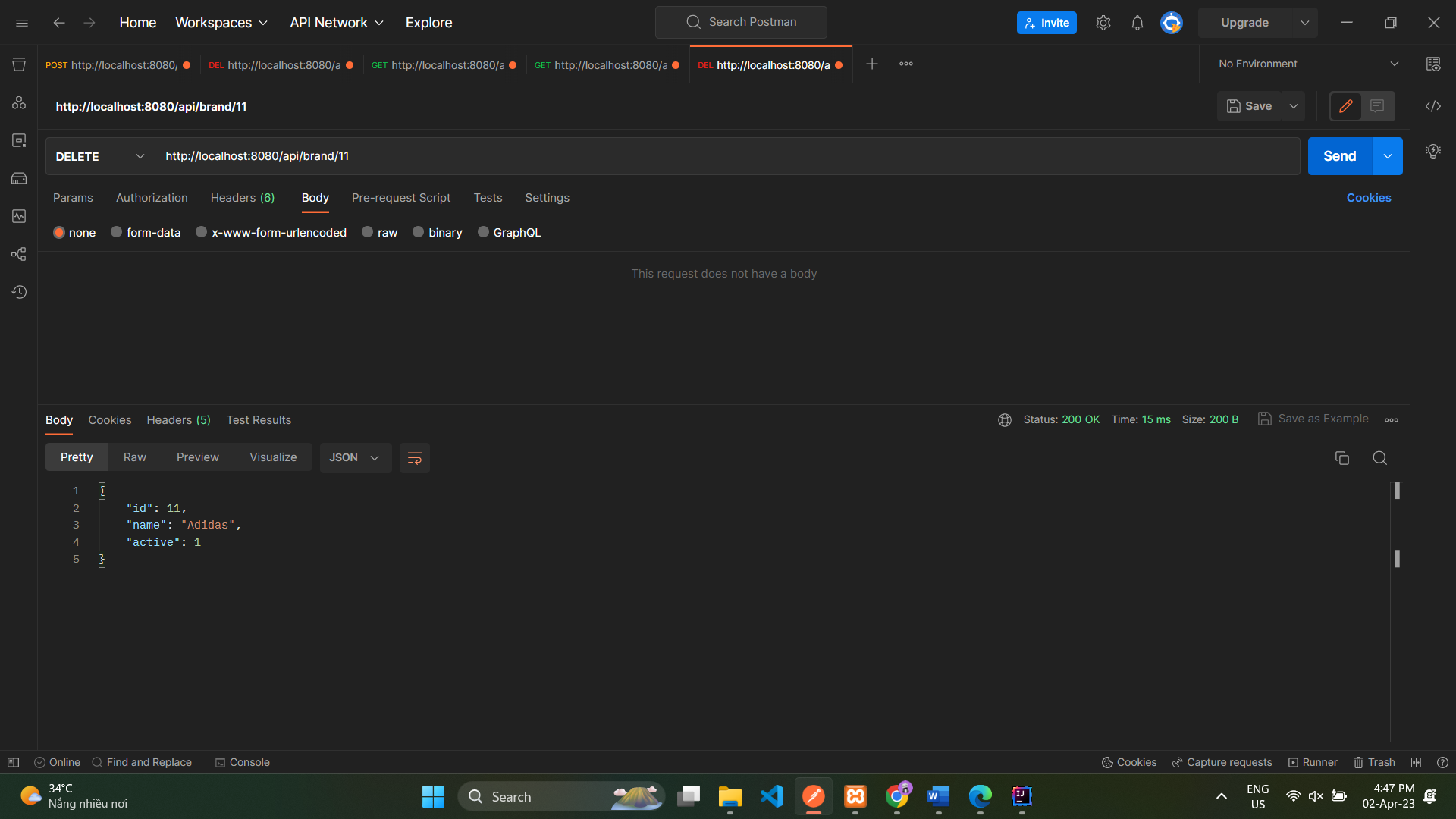


PUT: /api/brand/{id}



DELETE: /api/brand/{id}

(Khi xóa 1 brand, nó sẽ không mất đi khỏi database, nó chỉ được đánh dấu đã xóa)



Khi active =1 tức là brand đã xóa (ngừng hoạt động)