HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

BÁO CÁO

THỰC TẬP CƠ SỞ

Giảng viên: Kim Ngọc Bách

Tuần 8: 3/5/2025

Sinh viên: Nguyễn Khắc Trường B22DCCN884

MỤC LỤC

A. Kiến thức học được ở tuần này

Giới thiệu

Tiếp tục với kiến thức đã học được ở tuần trước, tuần này em tiếp tục tìm hiểu về Springboot

Phần 1: Xử lý ngoại lệ (Exception Handling) trong Spring Boot

Trong quá trình xây dựng ứng dụng, việc xuất hiện lỗi (ngoại lệ) là điều không thể tránh khỏi: có thể do dữ liệu đầu vào không hợp lệ, người dùng yêu cầu tài nguyên không tồn tại, lỗi truy xuất cơ sở dữ liệu, hoặc lỗi logic trong quá trình xử lý nghiệp vụ. Để ứng dụng không bị "sập" đột ngột và có thể trả về thông báo rõ ràng cho người dùng, Spring Boot cung cấp cơ chế **xử lý ngoại lệ tập trung (centralized exception handling)** rất linh hoạt.

1. Xử lý ngoại lệ bao gồm những gì?

Spring Boot hỗ trợ xử lý ngoại lệ qua một số cơ chế sau:

- **@ExceptionHandler**: Cho phép xử lý từng loại ngoại lệ cụ thể trong một @ControllerAdvice.
- **@ControllerAdvice**: Một lớp xử lý lỗi toàn cục cho toàn bộ controller, giúp gom logic xử lý lỗi lại một chỗ.
- **ResponseEntityExceptionHandler**: Lớp base có sẵn trong Spring để xử lý các lỗi phổ biến như MethodArgumentNotValidException, HttpMessageNotReadableException, v.v.
- **Custom Exception**: Người dùng có thể định nghĩa các exception riêng như ResourceNotFoundException, InvalidRequestException để phản ánh các lỗi nghiệp vụ.

2. Cách hoạt động

Khi một lỗi phát sinh trong quá trình xử lý request, thay vì để lỗi lan ra và trả về trang lỗi mặc định (trắng hoặc stack trace), Spring sẽ chuyển lỗi đến handler tương ứng trong @ControllerAdvice. Tại đây, developer có thể:

- Ghi log lỗi để tiện theo dõi.
- Trả về một đối tượng ResponseEntity có chứa HTTP status phù hợp (404, 400, 500,...) cùng thông điệp lỗi rõ ràng.
- Tùy chỉnh JSON response để frontend hoặc client API dễ xử lý.

3. Tác dụng

- **Tránh crash ứng dụng**: Ngăn ứng dụng dừng lại bất ngờ.
- **Tăng trải nghiệm người dùng**: Hiển thị lỗi có ý nghĩa, dễ hiểu.

- **Chuẩn hóa phản hồi**: Các lỗi đều tuân theo cùng một định dạng JSON → giúp frontend dễ xử lý hơn.
- **Giúp debug dễ dàng hơn**: Ghi log lỗi một cách có tổ chức.

Phần 2: Bảo mật (Security) trong Spring Boot

Trong bất kỳ ứng dụng web hoặc API nào, **bảo mật** luôn là ưu tiên hàng đầu, đặc biệt khi hệ thống có dữ liệu người dùng, tài chính hoặc tài nguyên nhạy cảm. Spring Boot tích hợp chặt chẽ với **Spring Security** – một framework mạnh mẽ và linh hoạt cho việc bảo vệ ứng dụng.

1. Spring Security bao gồm những gì?

- **Authentication** (xác thực): Kiểm tra danh tính người dùng (đăng nhập, token, OAuth, v.v.).
- **Authorization** (ủy quyền): Kiểm tra người dùng có quyền truy cập tài nguyên hay không (vai trò, quyền hạn).
- **CSRF protection**: Bảo vệ khỏi tấn công giả mạo yêu cầu từ bên thứ ba.
- **Password Encoding**: Mã hóa mật khẩu với BCryptPasswordEncoder.
- **Session Management**: Quản lý phiên đăng nhập (hữu ích cho web).
- **Security Filter Chain**: Chuỗi các bộ lọc bảo mật xử lý request/response.

2. Cách hoạt động

Spring Security hoạt động như một lớp chắn giữa client và controller:

- Khi một request đến, nó phải đi qua chuỗi filter bảo mật (SecurityFilterChain).
- Nếu request đó yêu cầu quyền xác thực hoặc ủy quyền:
- Spring Security sẽ kiểm tra token, session, hoặc thông tin login.
- Nếu không hợp lệ, trả về lỗi 401 Unauthorized hoặc 403 Forbidden.
- Nếu hợp lệ, cho phép truy cập controller phía sau.
- Các cấu hình bảo mật thường nằm trong lớp SecurityConfig, nơi lập trình viên xác định:
- Đường dẫn nào được phép truy cập công khai (permitAll()).
- Đường dẫn nào yêu cầu đăng nhập hoặc quyền cụ thể (hasRole("ADMIN")).
- Loại xác thực sử dụng (form, basic, JWT, OAuth2,...).

3. Tác dụng

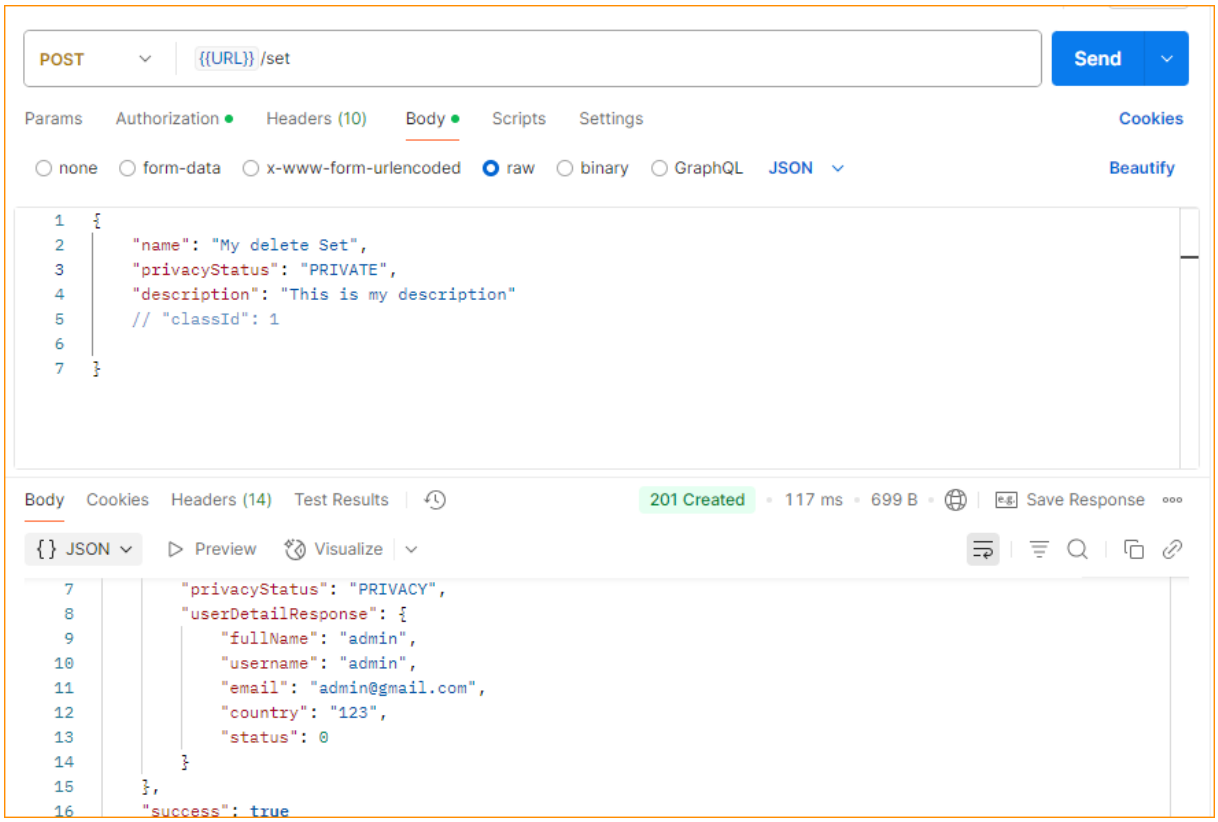
- **Ngăn truy cập trái phép**: Chặn người dùng chưa đăng nhập hoặc không có quyền.
- **Bảo vệ dữ liệu người dùng**: Đảm bảo dữ liệu cá nhân không bị rò rỉ.

- **Ngăn chặn các loại tấn công phổ biến**: CSRF, XSS, session hijacking,...
- **Dễ dàng tích hợp với các hệ thống xác thực khác**: như OAuth2, LDAP, SSO.
- **Tăng độ tin cậy của hệ thống**: Các API không thể bị lợi dụng dễ dàng.

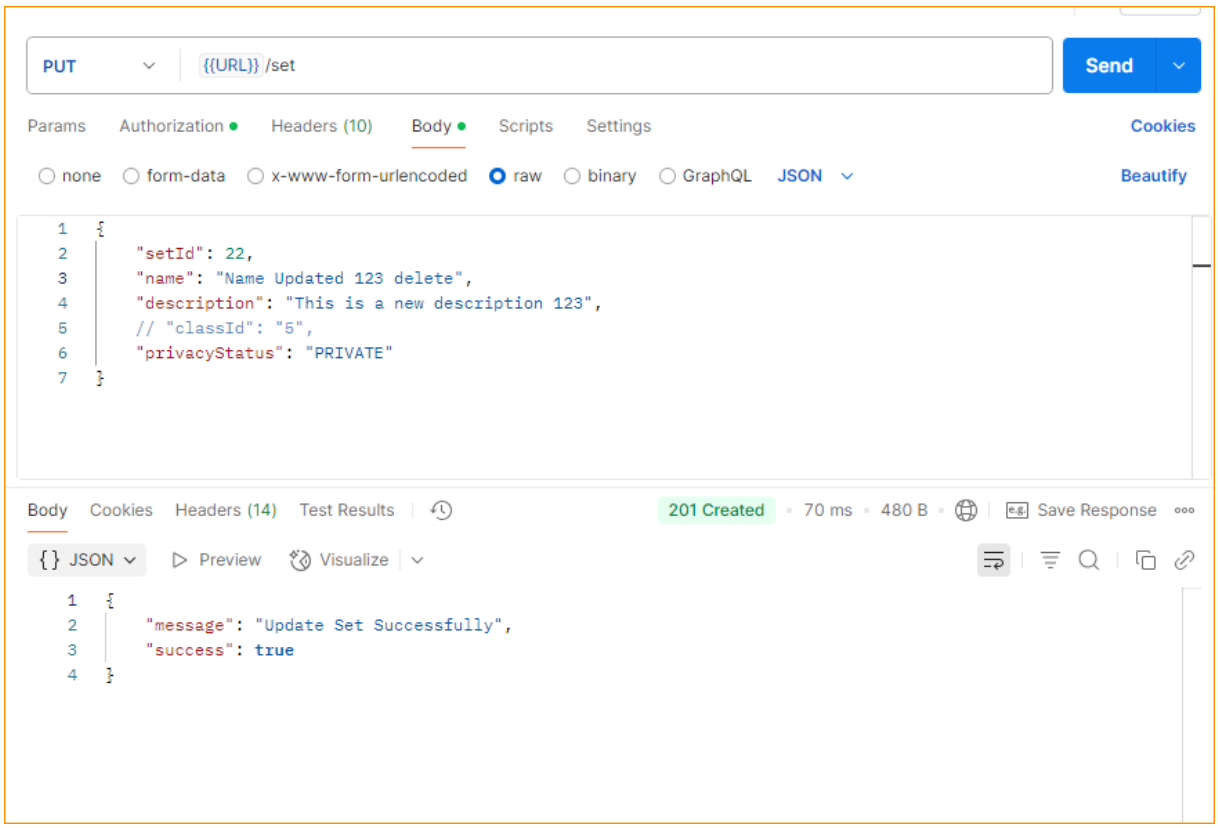
B. Tiến độ dự án

- Hoàn thiện các API liên quan đến các tập từ học(set) và từ học(word):

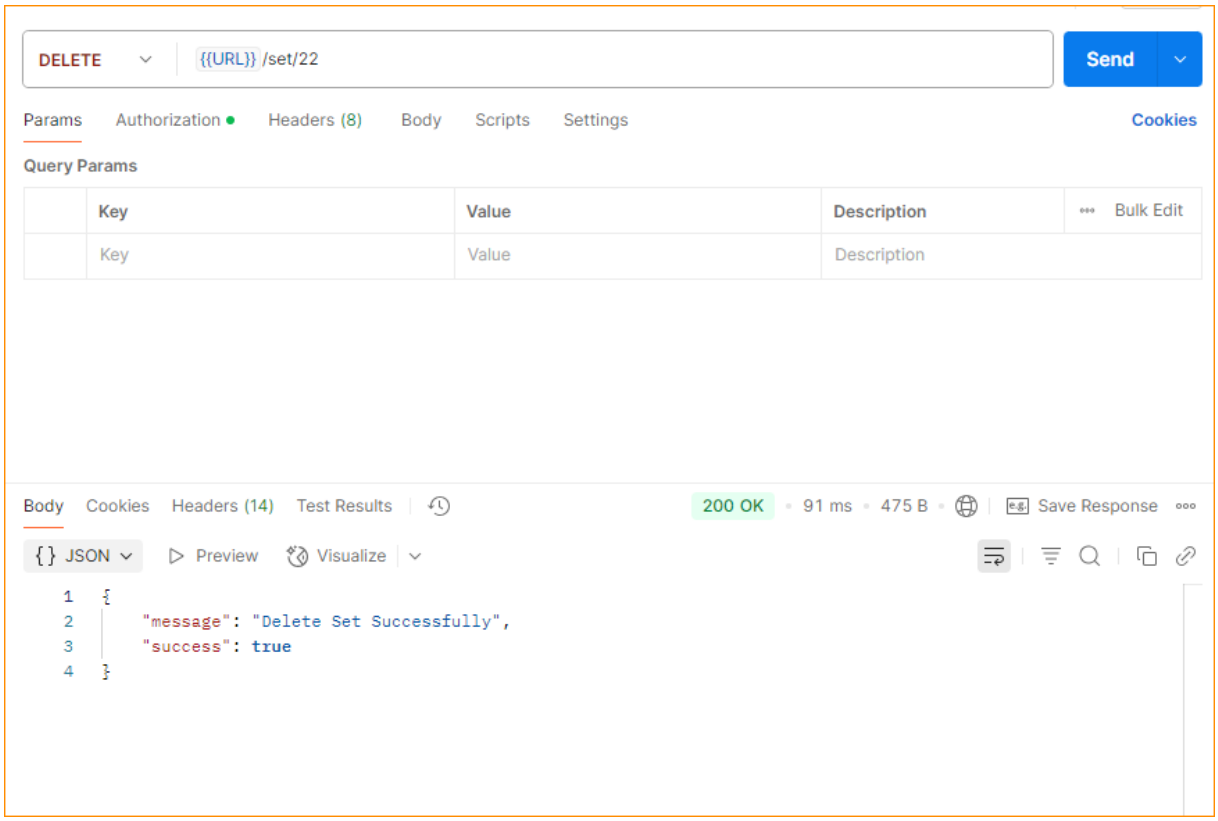
+ Tạo set: Người dùng nhập tên, chế độ riêng tư, mô tả, lớp(nếu để chế độ lớp) cho tập từ mình học.



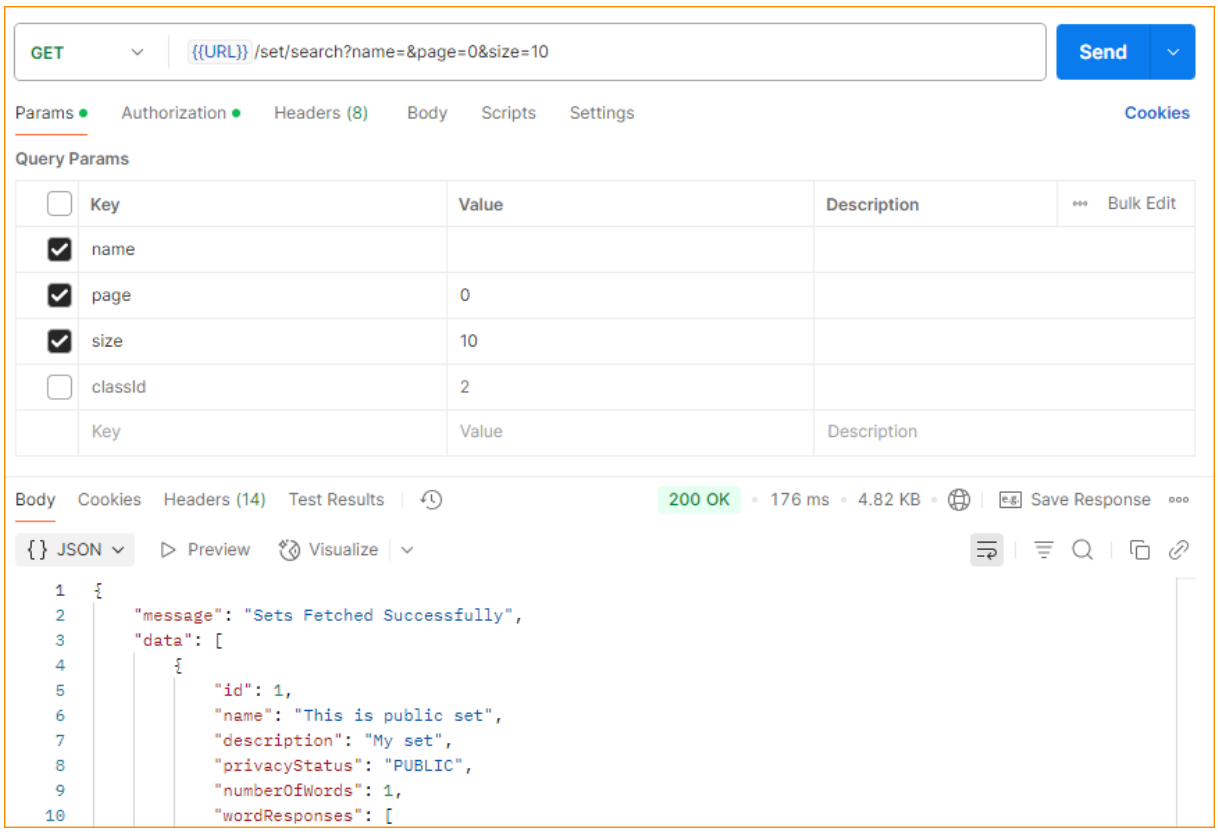
+ Sửa set: Tương tự phần thêm nhưng người dùng cần nhập cả lớp muốn sửa.



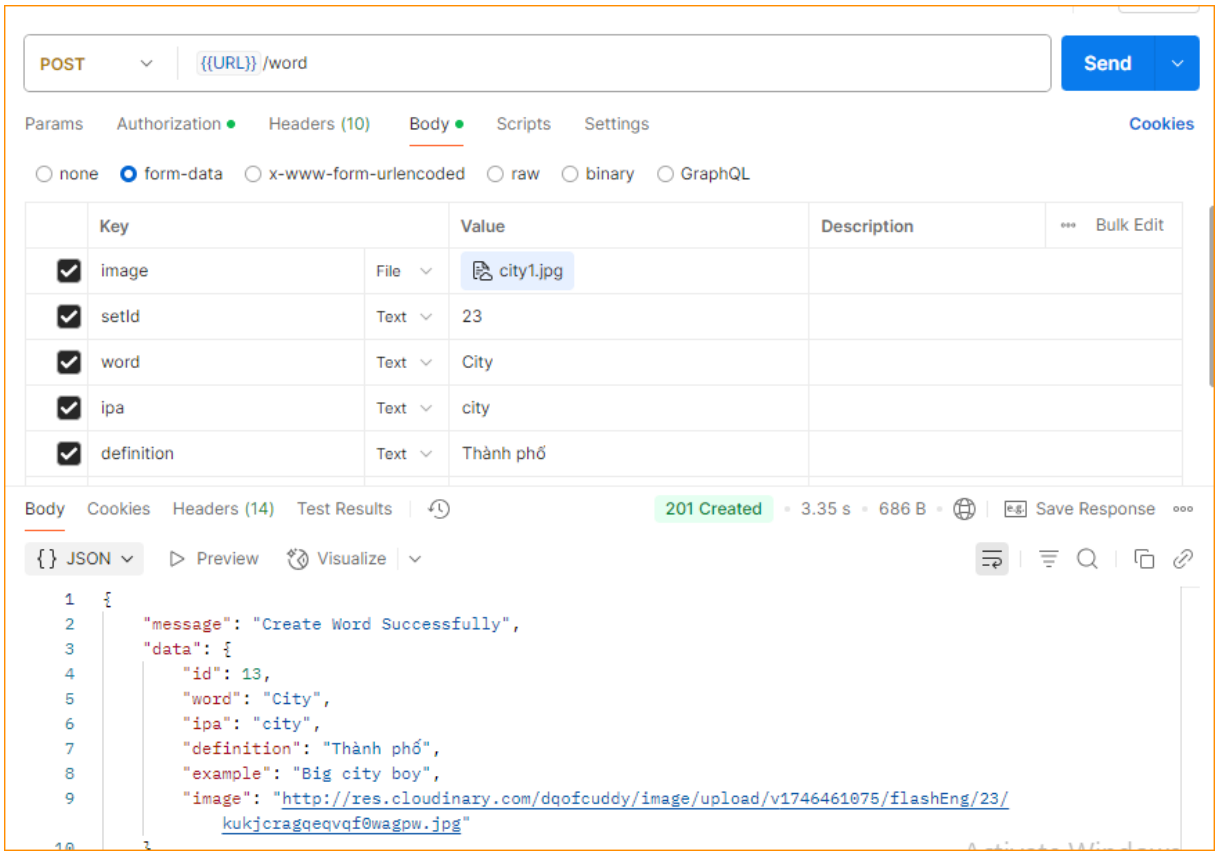
+ Xóa set: Người dùng gửi id lớp muốn xóa và token để xác nhận xem có quyền xóa không.



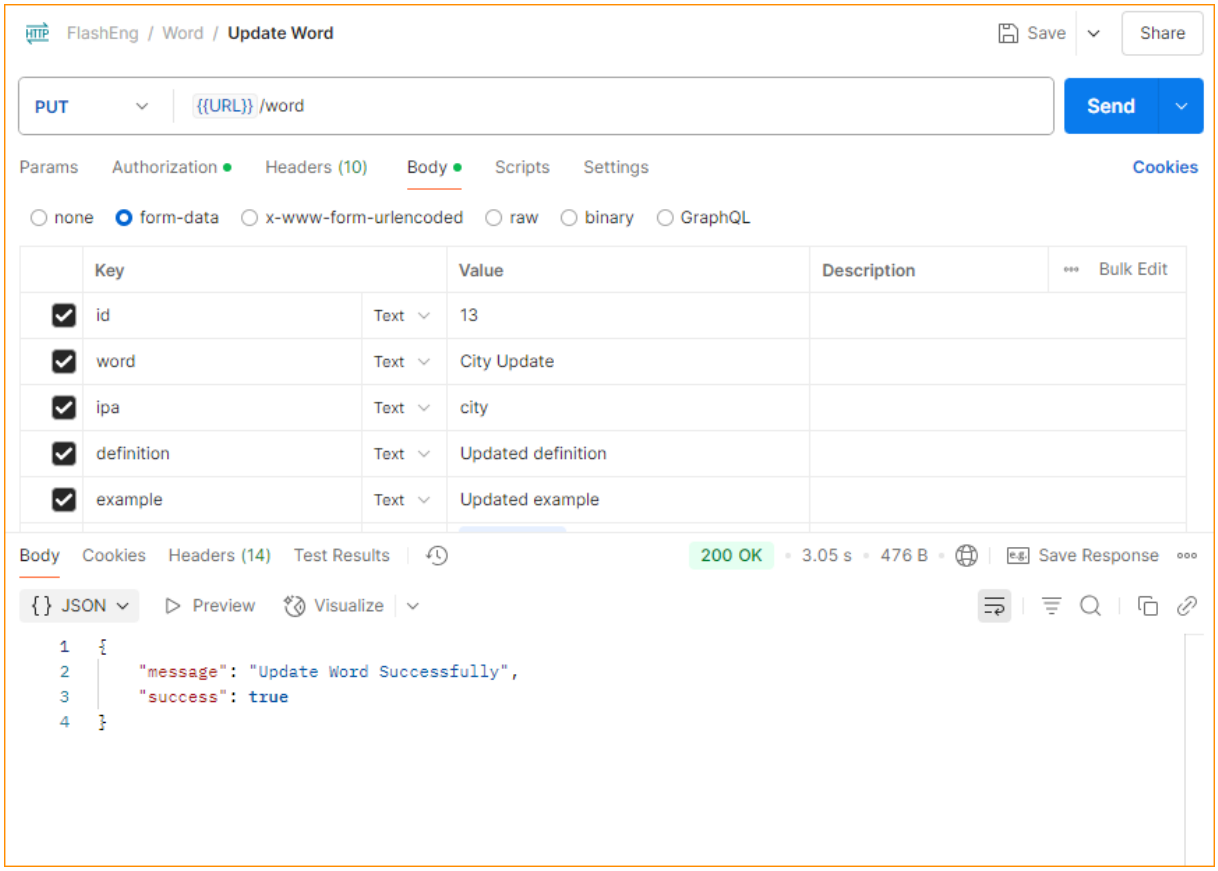
+ Tìm set: Người dùng nhập id hoặc tên set để tìm kèm theo là page và size và giới hạn số lượng trả về



+ Tạo word: Người dùng nhập các thông tin từ, nghĩa,… và setId để tạo word trong set.



+ Sửa word: các trường nhập tương tự phần thêm nhưng bỏ setId và thêm wordId.



+ Xóa word: Người dùng nhập wordId cần xóa kèm theo token để xác định xem có quyền xóa hay không.