**REST API**

# **HAL Explorer giúp gì cho bạn?**

**HAL Explorer** là **một giao diện web** giúp bạn **xem và kiểm tra API** mà bạn đã tạo trong Spring Boot.Nó giống như một trang web tự động liệt kê các API REST của bạn để bạn có thể xem và thử gửi yêu cầu (request) mà không cần dùng các công cụ như **Postman** hoặc **cURL**.

**Hiển thị API REST dễ dàng:**  
Bạn không cần phải nhớ từng endpoint hoặc tự kiểm tra thủ công, HAL Explorer liệt kê tất cả API của bạn.

**Tự động hỗ trợ liên kết (Hypermedia):**  
Nếu tài nguyên Address liên kết với tài nguyên khác (ví dụ: User), HAL Explorer sẽ hiển thị liên kết đó.

**Không cần thêm công cụ khác:**  
Bạn có thể kiểm tra và thử API ngay trong trình duyệt.

**Nếu không dùng HAL Explorer thì sao?**

Bạn sẽ phải sử dụng công cụ khác như **Postman** để gửi request đến từng endpoint.

Bạn phải tự nhớ hoặc tự viết tài liệu cho tất cả API.

**Tóm lại**

HAL Explorer là **một trang web tự động** được thêm vào ứng dụng của bạn, giúp bạn:

**Xem tất cả API REST đã tạo** mà không cần cấu hình gì thêm.

**Tương tác trực tiếp với API REST** ngay trong trình duyệt.

**Tiện lợi cho kiểm thử** trong quá trình phát triển.

# **Springdoc OpenAPI**

Khi bạn phát triển ứng dụng với API, đặc biệt là với REST API (dùng để giao tiếp giữa các hệ thống), bạn sẽ có rất nhiều endpoint (đường dẫn API).

**Vấn đề nếu không có tài liệu API:**

* Bạn sẽ phải **thủ công** tạo tài liệu cho từng API: giải thích cách sử dụng, các tham số yêu cầu, định dạng trả về.
* Các nhà phát triển khác hoặc người dùng API sẽ **khó hiểu** và không biết cách gọi đúng API.
* Khi bạn thay đổi API (thêm, sửa, xóa), bạn phải **cập nhật tài liệu thủ công**, điều này rất dễ bị bỏ sót và làm tài liệu bị lỗi.

**Giải pháp với Springdoc OpenAPI:**

* **Springdoc OpenAPI** tự động tạo **tài liệu API** từ mã nguồn của bạn. Nó "quét" tất cả các endpoint mà bạn đã xây dựng và tạo ra **tài liệu OpenAPI** (có thể xem như là một bản mô tả chi tiết về các API).
* **Swagger UI** cung cấp **giao diện web** trực quan để bạn có thể xem tài liệu API này và thậm chí **thử nghiệm API trực tiếp**.

**Lợi ích của Swagger UI và OpenAPI:**

**Lợi ích đối với nhà phát triển:**

1. **Tiết kiệm thời gian:**  
   Bạn không cần phải viết tài liệu thủ công cho API nữa. **Springdoc OpenAPI** làm điều này tự động.
2. **Dễ dàng kiểm thử API:**  
   **Swagger UI** là giao diện web cho phép bạn thử nghiệm API mà không cần phải sử dụng công cụ bên ngoài như **Postman** hoặc **cURL**. Bạn có thể gửi yêu cầu ngay từ trình duyệt và xem kết quả trực tiếp.
3. **Dễ dàng duy trì và cập nhật:**  
   Khi bạn thay đổi API (thêm, sửa, xóa endpoint), tài liệu sẽ được **cập nhật tự động**. Không cần phải lo lắng về việc quên cập nhật tài liệu.

**Lợi ích đối với các bên khác (ví dụ: các nhà phát triển khác, tester, khách hàng):**

1. **Dễ dàng hiểu API:**  
   Các nhà phát triển khác có thể truy cập Swagger UI và thấy ngay cách sử dụng API mà không cần phải hỏi bạn chi tiết.
2. **Tự động sinh ra thông tin:**  
   Swagger UI cung cấp mô tả chi tiết về các endpoint, tham số yêu cầu, kiểu dữ liệu, các lỗi có thể xảy ra, giúp người dùng API hiểu ngay cách gọi API.

[**http://localhost:8080/swagger-ui/index.html**](http://localhost:8080/swagger-ui/index.html)

**Có thể đổi** springdoc.swagger-ui.path=/my-custom-docs

[**http://localhost:8080/my-custom-docs/index.html**](http://localhost:8080/my-custom-docs/index.html)

**Tổng kết**

* **HAL Explorer**:
  + HAL là một chuẩn để mô tả API, nơi mà các tài nguyên của bạn được mô tả bằng **dữ liệu** cùng với **liên kết** tới các tài nguyên khác. Nói cách khác, API sử dụng **HAL** giúp bạn dễ dàng **điều hướng** qua các tài nguyên liên quan mà không cần phải biết trước chúng ở đâu.
  + **Ví dụ**: Khi bạn truy vấn danh sách "nhân viên", mỗi nhân viên có thể chứa liên kết tới tài nguyên "chi tiết nhân viên" hoặc "sửa thông tin nhân viên". Bạn chỉ cần theo các liên kết này để tiếp tục truy vấn, không cần phải biết rõ cấu trúc API.
* **Springdoc OpenAPI**:
  + OpenAPI là một tiêu chuẩn mô tả **tài liệu chi tiết** về API của bạn. Nó mô tả tất cả các **endpoints** (điểm cuối API), các **phương thức HTTP** (GET, POST, PUT, DELETE), các tham số cần thiết, kiểu dữ liệu đầu vào/đầu ra, và các thông tin khác như lỗi có thể xảy ra.
  + **Swagger UI** là một giao diện người dùng mà Springdoc OpenAPI cung cấp, giúp bạn dễ dàng **thử nghiệm API** trực tiếp và xem tài liệu API mà không cần phải mở mã nguồn.

**Kết hợp cả hai có được không?**

**Câu trả lời là có thể kết hợp được cả hai**! Bạn có thể sử dụng cả **HAL Explorer** và **Springdoc OpenAPI** trong một dự án để đạt được những lợi ích sau:

* **HAL Explorer** giúp bạn mô tả cách điều hướng API của bạn qua các liên kết tự động (hyperlinks), mang đến sự **dễ dàng trong việc khám phá tài nguyên**.
* **Springdoc OpenAPI** giúp bạn tạo **tài liệu chi tiết về API**, cung cấp một **giao diện Swagger UI** để người dùng có thể thử nghiệm API trực tiếp, và đảm bảo API của bạn có tài liệu mô tả rõ ràng.

Kết hợp cả hai sẽ giúp bạn có một API vừa dễ dàng **khám phá thông qua các liên kết** (HAL Explorer) và có **tài liệu chi tiết để hiểu và thử nghiệm API** (Springdoc OpenAPI).

# **Logging**

Cấu hình này thiết lập **Logging** trong ứng dụng Spring Boot để theo dõi và ghi lại các sự kiện, lỗi hoặc trạng thái trong quá trình ứng dụng chạy. Logging rất quan trọng để bạn dễ dàng kiểm tra và sửa lỗi khi phát triển ứng dụng.

**<root level="INFO">:**

* Đặt mức độ ghi log mặc định cho toàn bộ ứng dụng là INFO (chỉ ghi log từ mức INFO trở lên).
* Các mức độ log gồm:
  + TRACE: Ghi chi tiết nhất, thường chỉ dùng debug.
  + DEBUG: Thông tin cho nhà phát triển.
  + INFO: Thông tin quan trọng, thường sử dụng nhất.
  + WARN: Cảnh báo nhưng ứng dụng vẫn chạy được.
  + ERROR: Báo lỗi nghiêm trọng.

**<appender-ref>:**

* **CONSOLE**: Ghi log ra terminal.
* **FILE**: Ghi log vào file.