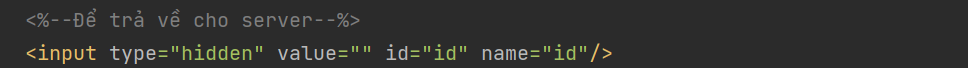
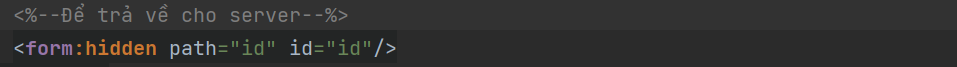
**BÀI 7.2: CRUD TRONG SPRING MVC**

**Submit API**

****

Tồn tại trong form để cho biết đây là cập nhật hay thêm mới: id rỗng là thêm mới còn id có giá trị là cập nhật. Ta có thể sửa thành dạng của Spring form



A black rectangle with white text

Description automatically generated

Để call được API phải trỏ đến được url của API đó, được định nghĩa trong hàm Ajax (url)



A computer screen shot of code

Description automatically generated

**Build phần service để cập nhật xuống database**

Quay lại khóa trước, khi update hoặc insert thì sẽ phát sinh một số khái niệm như transaction, commit, rollback. Khi thực hiện các thao tác với db (cập nhật, thêm mới, xóa) thì tất cả sẽ được đưa vào một transaction, nếu tất cả các thao tác đều done và không xảy ra lỗi thì sẽ thực hiện commit, còn nếu 1 trong các thao tác có vấn đề thì sẽ thực hiện rollback. Ở khóa trước các thao tác này phải tự build trong class Abstract (hàm update và insert)

Về bản chất thì trước khi tất cả các thao tác done hết thì dữ liệu sẽ được sửa ở trong một table ảo gọi là table tempotary. Chỉ khi nào toàn bộ các thao tác thành công thì mới commit, thực hiện train dữ liệu xuống database. Còn nếu không thì nó sẽ rollback toàn bộ dữ liệu có trong table ảo

Đối với Spring MVC – Spring data JPA, nó sẽ cung cấp một anotation tự động quản lý các thao tác trong một transaction của spring framework (đặt ở các hàm thao tác với dữ liệu)

A computer screen shot

Description automatically generated

Khi thêm một news, có mối quan hệ ManyToOne với category nên bắt buộc phải lấy được thông tin của category thì mới có thể thêm được 1 news. Do đó trong repository ở category cần thêm một hàm sau. Mục địch là để tìm category của news muốn thêm để thực hiện liên kết

A screen shot of a computer program

Description automatically generated

Khi lưu xuống database thì database cần là Entity. Nhưng do dữ liệu mà từ form đưa ra là một DTO do đó cần convert sang Entity.

Tóm tắt đoạn code bên dưới:

+ Đầu tiên tìm kiếm category theo mã code, có thể làm được như vậy là do dữ liệu được lưu ở dạng Map (xem bài trước phần spring form). Sau đó tiến hành convert từ DTO sang Entity, cuối cùng gắn thêm category vừa tìm được bên trên.

+ Hàm save bên trên sẽ có 2 chức năng và hoạt động tùy biến theo id. Nếu id là null thì nó sẽ ngầm hiểu là thêm mới. Còn nếu khác null thì nó sẽ thực hiện cập nhật. Sau khi lưu xong thì nó sẽ trả ra một news mới nhưng ở kiểu Entity. Nhiệm vụ là phải convert ngược lại sang DTO

A screen shot of a computer code

Description automatically generated

Khi cập nhật thì ta cần phải lấy lại thông tin của news cũ, mục đích là để gán lại các giá trị mà không thể sửa trên view (các giá trị trên view không có thông tin mà chỉ có thể lấy từ cái cũ). Do gán lại cho news mới mà cả 2 đều phải ở dạng Entity nên ta sử dụng tính chất Overiding trong OOP để xây dựng thêm một hàm trong Converter đóng vai trò convert giá trị từ entity này qua entity khác

A screen shot of a computer code

Description automatically generated

Và trước khi cập nhật thông tin cũ thì cũng cần phải tìm ra category để gắn vào (vì giá trị news hiện tại không mang category). Sau khi gán vào và cập nhật thông tin thì cũng sử dụng hàm save để update. Nó sẽ giống giống với hàm insert

A screen shot of a computer program

Description automatically generated

Do đó mà để tối ưu thì ta gộp cả 2 thành một hàm save

A computer screen shot of a program code

Description automatically generated

Chung quy lại: Đầu tiên cả 2 hàm đều phải find cái category theo code, sau đó nếu id khác null thì sẽ tiến hành cập nhật. Ở cập nhật, sẽ có đoạn lấy bài viết cũ lên, set cái category cho bài viết cũ và convert sang bài viết mới (dựa vào bài viết cũ để overide lại các thông tin thay đổi). Còn khi thêm mới thì đơn thuần DTO gửi lên như thế nào thì chỉ cần convert sang entity và set thêm cái category.

**Build lên phía API**