* **Web Service:** là 1 dịch vụ web, giúp các ứng dụng client – server có thể giao tiếp được với nhau thông qua giao thức HTTP/HTTPS. Là 1 giải pháp giúp các ứng dụng giao tiếp với nhau không phụ thuộc vào ngôn ngữ và nền tảng.
* **Các loại Web Service:** SOAP web service và restful web service
* **Spring** là 1 framework cung cấp 1 cơ sở hạ tầng toàn diện để hỗi trợ các lập trình viên phát triển ứng dụng java
* Spring boot là là 1 extension của pring famework loại bỏ đi những bước cấu hình phức tạp của spring
* API: tên api phải là danh từ còn hành động của nó thì do httpmethod quy định rồi.

**@Controller**

**Class API{**

@RequestMapping(value = "/news", method = RequestMethod.POST)

@ResponseBody

**public** NewDTO createNew(@RequestBody NewsDTO model) {

**return** model

* }
* **@RequestMapping(value = “/new”, method = RequestMethod.POST)** khai báo link cho api và http method cho api này, có nghĩa là khi truy cập vào đường link này và theo method post sẽ gọi đến method được gắn cái anotation này
* **@ ResponseBody** là anotation giúp chuyển đổi kiểu dữ liệu trả về của method thành kiểu json và trả về cho client
* **@RequestBody NewDTO model** là 1 anotation giúp lấy nội dung json của request và parse sang kiểu NewDTO và lưu vào biến model.
* **@RestController = @Controller + @ResponseBody**
* **Spring data jpa:** là 1 thư viện của spring framework, giúp ta thao với cơ sở dữ liệu. Nó giúp ta có thể dễ ràng mapping dữ liệu từ table sang đối tượng trong java, nó xây dựng hibernate query language giúp t truy vấn dữ liệu mà không cần phụ thuộc vào Cơ sở dữ liệu đang sử dụng

**+** Với jpa chúng ta chỉ việc tạo các entity và nó sẽ dựa vào các entity đó để generate ra các table

**+** đối với spring framework 4. thì chúng ta đã được tích hợp sẵn spring data jpa rồi

**+ để sử dụng spring data jpa:** ta phải gọi spring jpa, khai báo msql connector trong sping boot trong file pom.xml

* **Quan hệ 1 - n trong spring data jpa:** để tạo quan hệ 1 - n trong spring data jpa thì trong class Entity(1) ta sẽ khai báo **1** **list** các Entity(n) và gắn vào nó anotation @OneToMany và giá trị mappedBy là tên biến Entity(1) được khai báo bên class Entity(n).

**+ ví dụ:**

@OneToMany(mappedBy = "category") // category là map với category khai báo trong NewEntity

**private** List<NewEntity> news = **new** ArrayList<>();

Còn trong class Entity(n) sẽ khai báo **1** **biến** Entity(1) và gắn vào 1 anotation @ManyToOne và @JoinColumn(name = "category\_id") chính là tên field, khóa ngoại tham chiếu đến Entity(1)

**Ví dụ:**

@ManyToOne

@JoinColumn(name = "category\_id")

**private** CategoryEntity category;

* **Quan hệ n – n trong spring data jpa:** để tạo quan hệ thì trong Entity class của 1 trong 2 Entity class ta khai báo như sau:

Ví dụ khai báo trong **UserEntity:**

@ManyToMany

@JoinTable(name = "user\_role",

joinColumns = @JoinColumn(name = "user\_id"),

inverseJoinColumns = @JoinColumn(name = "role\_id"))

**private** List<RoleEntity> roles = **new** ArrayList<>();

**+** ta khai báo 1 list chứa các biến của Entity class kia

**+** trong đó user\_id chính là tên trường được tạo trong bảng trung gian tham chiếu đến khóa chính của Entity class chứa nó.

+ role\_id chính là tên trường được tạo trong bảng trung gian tham chiếu đến khóa chính trong bảng của Entity class thứ 2

Ví dụ khai báo trong **RoleEntity**:

@ManyToMany(mappedBy = "roles") // roles map với mảng roles được khai báo bên userEntity

**private** List<UserEntity> users = **new** ArrayList<>();

+ ta khai báo 1 list chứa các biến của Entity class kia

+ gắn vào 1 anotation@ManyToMany(mappedBy = "roles"), mappedBy chính là cái list được khai báo bên Entity class kia

* @Component là một [Annotation](https://loda.me/java-huong-dan-tu-tao-mot-annotations-loda1554297400922) (chú thích) đánh dấu trên các Class để giúp **Spring** biết nó là một Bean(dependency). Khi khởi chạy ứng dụng spring boot sẽ dò tìm toàn bộ các class cùng cấp or ở trong các package thấp hơn so với class app(có thể cấu hình khác), trong quá trình dò tìm khi được gặp 1 class được đánh dấu với @Component thì nó sẽ tạo 1 instance và đưa vào ApplicationContext(Bean Container)  để quản lí
* Về bản chất @Service và @Repository cũng chính là @Component
* @Target({ElementType.TYPE})
* @Retention(RetentionPolicy.RUNTIME)
* @Documented
* @Component // Cũng là một @Component
* public @interface Service {
* @AliasFor(
* annotation = Component.class
* )
* String value() default "";
* }

@Target({ElementType.TYPE})

@Retention(RetentionPolicy.RUNTIME)

@Documented

@Component

public @interface Repository {

@AliasFor(

annotation = Component.class

)

String value() default "";

}

* @Autowired là 1 anotation nói với **Spring Boot** hãy tự inject (tiêm) một instance của thuộc tính được đánh dấu vào thuộc tính này khi khởi tạo instance của class chứa nó.
* Trong spring boot muốn giao tiếp được với db ta sẽ tạo 1 package là repository và tạo ra các interface kế thừa từ JpaRepository
* **public** **interface** NewsRepository **extends** JpaRepository<NewEntity, Long>{
* }
* Jpa sẽ cung cấp cho t những hàm mà ta cần để tao tác với db