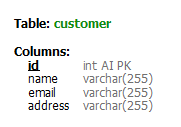


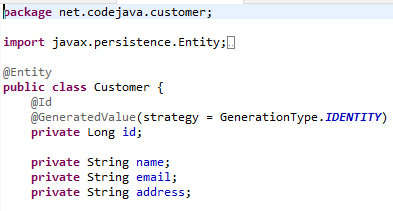
Bước đầu tiên chúng ta thấy rằng đối tượng Customer có 4 trường cần lưu vào cơ sở dữ liệu còn trường Action sẽ gán cứng vì đó là các action cố định .

**CHỨC NĂNG HIỆN LÊN TRANG CHỦ**

**Bước 1 :** Tạo cơ sở dữ liệu để chứa dữ liệu của Customer .



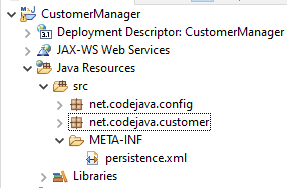
**Bước 2 :** Tạo lớp entity ở trong chương trình java để mapping với bảng trong database

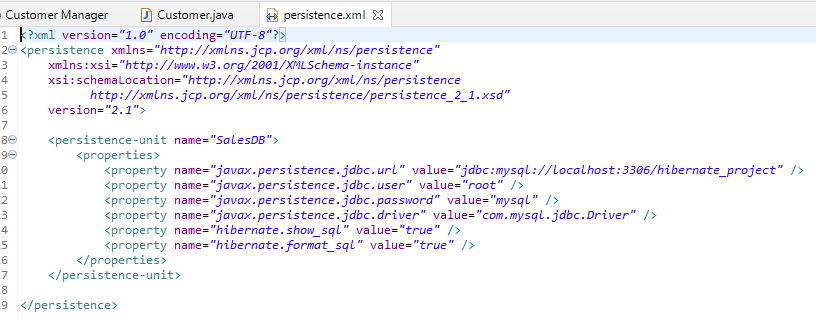


**Qui định :** Một entity phải có 1 khóa chính

-Nếu tên bảng trùng với tên bảng trong cơ sở dữ liệu thì không cần khai báo @Table.Tương tự với field.

**Bước 3 :** Cấu hình bằng jpa để tạo môi trường mapping giữa database và lớp trong java

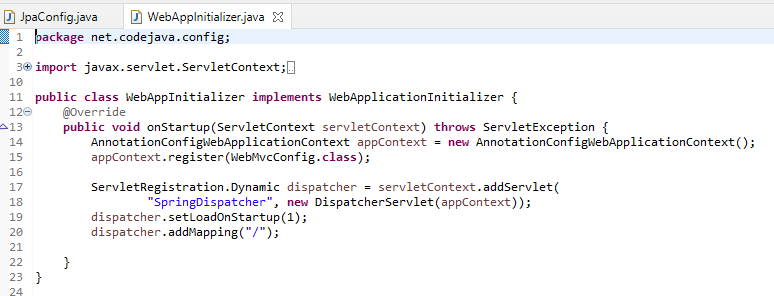




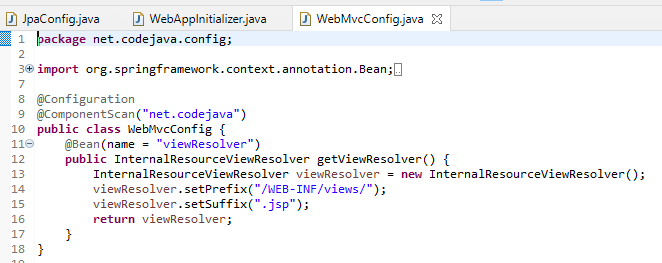
**Bước 4 :** Cấu hình jpa có 2 cách thứ nhất là dùng xml còn thứ 2 chúng ta cấu hình qua lớp



**Bước 5 :**



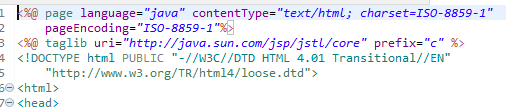
**Bước 6 :** Tạo file cấu hình kho chứa các file view và các file viết tắt không cần viết tên đuôi



**Bước 7 :** Như vậy qua 6 bước chúng ta đã hoàn thành việc kết nối giữa lớp entity trong java và lớp trong database .

Bước tiếp theo chúng ta sẽ thiết kế action New Customer.

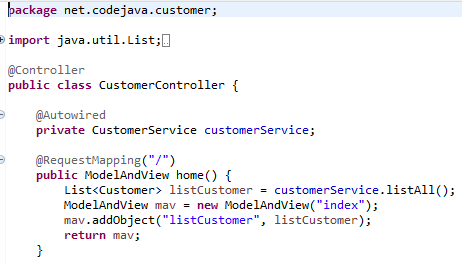
Trong WEB-INF chúng ta thiết lập giao diện index.jsp như sau :



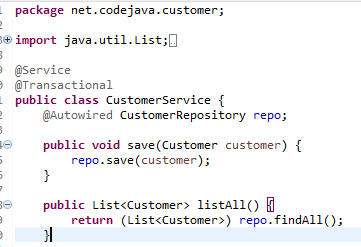


Lưu ý vào cách sử dụng cú pháp vòng lặp với jsp

**Bước 9 :** Như vậy khi chúng ta click vào đường link localhost:8080/CustomerManager/ thì sẽ hiện ra trang index như trên vì vậy ở controller chúng ta phải thiết lập một RequestMapping trả về view index.jsp



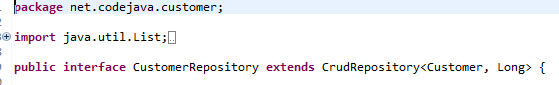
**Bước 10 :** Chúng ta tạo ra một lớp CustomerService để xử lý việc lấy dữ liệu lên từ database



Lưu ý với annotation @Transactional

<https://thorben-janssen.com/transactions-spring-data-jpa/>

**Bước 11 :** Chúng ta tạo 1 interface CustomerRepository



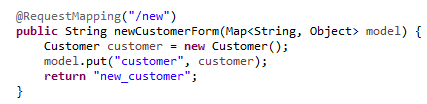
Như vậy qua 11 bước chúng ta đã hoàn thành xong chức năng hiện ra trang chủ .

**CHỨC NĂNG ADD NEW CUSTOMER**

**Bước 1 :** Trong view index.jsp chúng ta thêm đường link như sau :

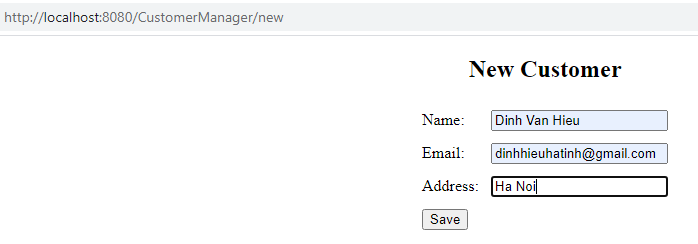


**Bước 2 :** Trên controller chúng ta thiết lập requestmapping là new với method là get như sau :

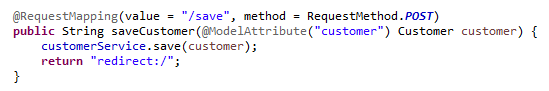


**Bước 3 :** Thiết lập trang view **new\_customer.jsp**

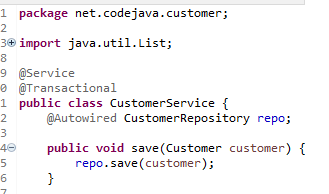




**Bước 4 :** Sau khi click vào save thì controller sẽ tiếp nhận và xử lý lưu vào database như sau



**Bước 5 :** Ở tầng Service



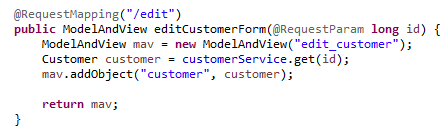
Như vậy chúng ta đã hoàn thành chức năng save một customer vào cơ sở dữ liệu

**CHỨC NĂNG EDIT**

**Bước 1 :** Đầu tiên chúng ta gán href cho edit



**Bước 2 :** Sau khi click vào chữ edit thì có nghĩa là client đã gửi lên controller một request mapping như href với method là post .Nhiệm cụ của controller này là trả về một trang khai báo customer có chứa dữ liệu như id truyền trên url



**Bước 3:** Thiết lập trang edit\_customer.jsp để hiện ra dữ liệu cần sửa

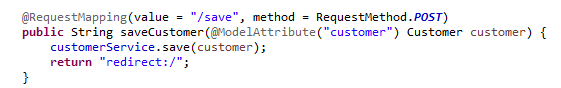


**Bước 4 :** Trong service chúng ta viết phương thức get để lấy dữ liệu lên



Như vậy qua 4 bước chúng ta đã hiện ra được trang cần edit

**Bước 5 :** Sau khi click vào nút submit thì nghĩa là client đã gửi lên server một request với thông số như trong th:action và method cũng vậy

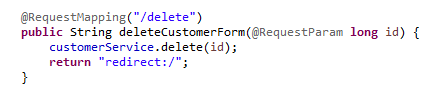


**CHỨC NĂNG DELETE**

**Bước 1 :** Tại nút delete chúng ta thiết lập một đường link như sau :



**Bước 2 :** Sau khi click vào nút delete nghĩa là client đã gửi lên server một request như href và một method là get

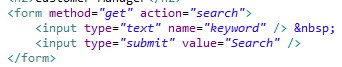


**Bước 3 :** Trong tầng service tiến hành viết phương thức delete như sau :

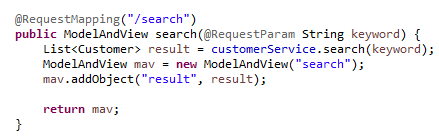


**CHỨC NĂNG SEARCH**

**Bước 1 :** Tại trang index chúng ta thêm vào 1 form như sau :



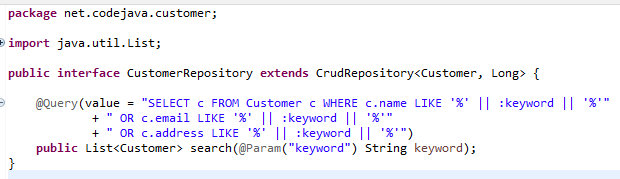
**Bước 2:** Sau khi chúng ta click vào nút submit nghĩa là chúng ta đã gửi lên controller một request với method action name ..



**Bước 3 :** Trong tầng service chúng ta thiết lập phương thức search như sau :



**Bước 4 :** Trong CustomerService chúng ta thiết lập phương thức search như sau :



<https://stackjava.com/spring/huong-dan-annotation-query-nativequery-trong-spring-data.html>

