Java 5

Lab 5

**JPA Repository**



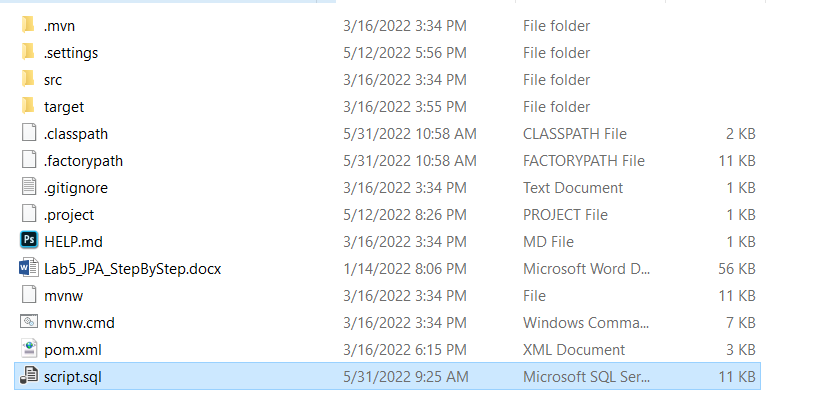
Nhóm : 3

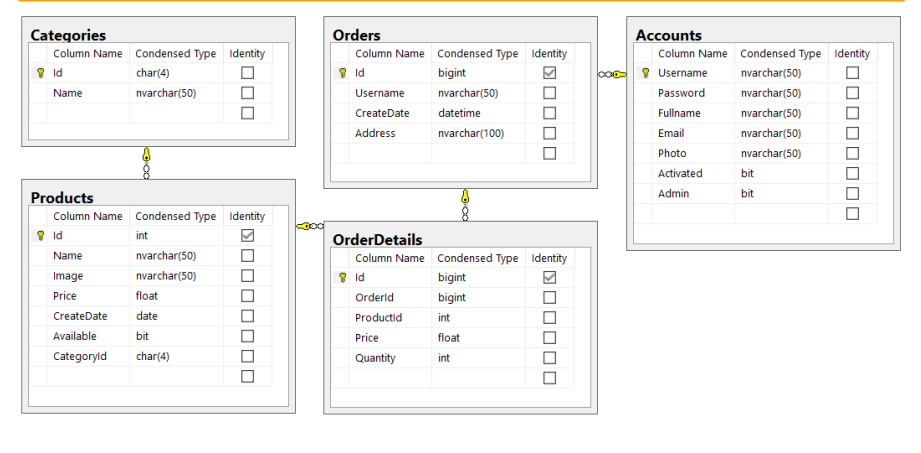
Lớp : IT16303

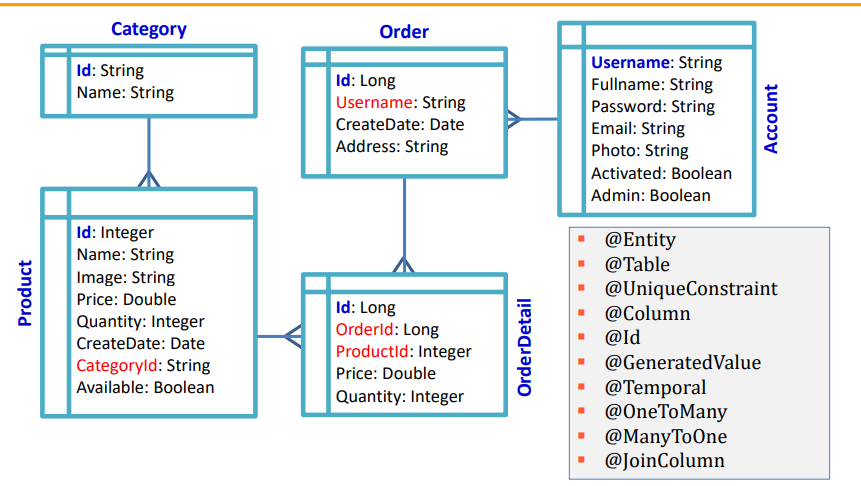
Giảng viên hướng dẫn : Lê Anh Tú

Bài 1: Xây dựng Dao làm việc với CSDL j5Shop.sql

* Chạy file script.sql để tạo DataBase







* Tạo Spring project và thêm 2 thư viện jpa và jdbc vào file Pom.xml

<**dependency**>

<**groupId**>org.springframework.boot</**groupId**>

<**artifactId**>spring-boot-starter-data-jpa</**artifactId**>

</**dependency**>

<**dependency**>

<**groupId**>com.microsoft.sqlserver</**groupId**>

<**artifactId**>mssql-jdbc</**artifactId**>

<**scope**>runtime</**scope**>

</**dependency**>

* Khai báo kết nối CSDL trong file **application.properties**

spring.datasource.url=jdbc:sqlserver://localhost:1433;database=J5Shop

spring.datasource.username=sa

spring.datasource.password=123456

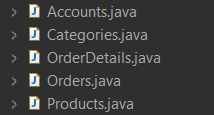
spring.datasource.driverClassName=com.microsoft.sqlserver.jdbc.SQLServerDriver

spring.jpa.hibernate.dialect=org.hibernate.dialect.SQLServer2014Dialect

spring.jpa.show-sql=false

spring.jpa.hibernate.ddl-auto = none

* Tạo các lớp Entity ánh xạ với CSDL



* + Category

@Data

@NoArgsConstructor

@AllArgsConstructor

@Entity

@Table(name = "categories")

public class Categories {

@Id

String id;

String name;

@OneToMany(mappedBy = "categories")

List<Products> products;

}

* + Product

@Data

@NoArgsConstructor

@AllArgsConstructor

@Entity

@Table(name = "products")

public class Products {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

int id;

String name, image;

double price;

boolean available = true;

@Temporal(TemporalType.DATE)

@Column(name = "createdate")

Date createDate = new Date();

@ManyToOne @JoinColumn(name = "categoryid")

Categories categories;

@OneToMany(mappedBy = "products")

List<OrderDetails> orderDetails;

}

* + Account

@Data

@NoArgsConstructor

@AllArgsConstructor

@Entity

@Table(name = "accounts")

public class Accounts {

@Id

String username;

String email, fullname, password, photo;

boolean activated = true, admin = false;

@OneToMany(mappedBy = "accounts")

List<Orders> orders;

}

* + Order

@Data

@NoArgsConstructor

@AllArgsConstructor

@Entity

@Table(name = "orders")

public class Orders {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

int id;

String address;

@Temporal(TemporalType.DATE)

@Column(name = "createdate")

Date createDate = new Date();

@ManyToOne @JoinColumn(name = "username")

Accounts accounts;

@OneToMany(mappedBy = "orders")

List<OrderDetails> orderDetails;

}

* + Order detail

@Data

@NoArgsConstructor

@AllArgsConstructor

@Entity

@Table(name = "orderdetails")

public class OrderDetails {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

int id;

double price;

int quantity;

@ManyToOne @JoinColumn(name = "orderid")

Orders orders;

@ManyToOne @JoinColumn(name = "productid")

Products products;

}

* Tạo các Interface DAO để làm việc với CSDL

public interface CategoryDAO extends JpaRepository<Categories, String>{}

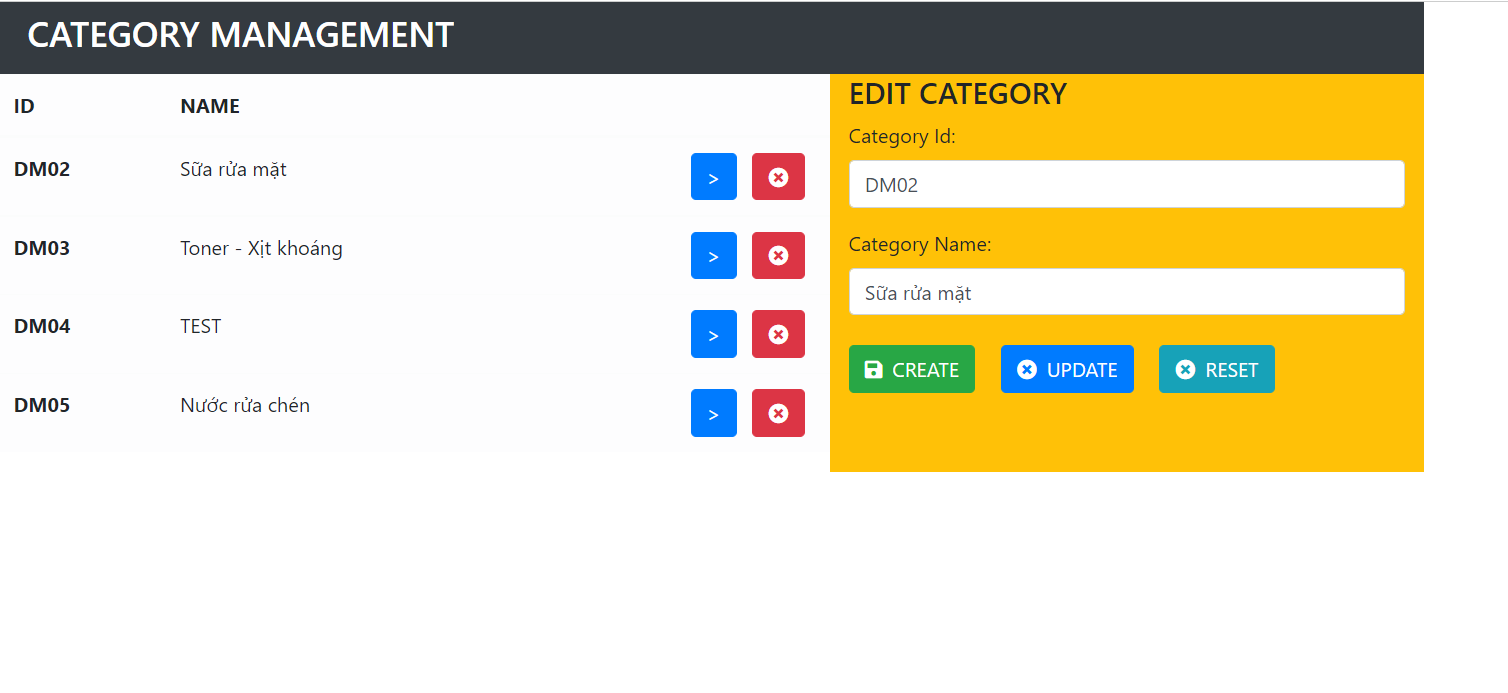
public interface ProductDAO extends JpaRepository<Products, Integer>{}

public interface AccountDAO extends JpaRepository<Accounts, String>{}

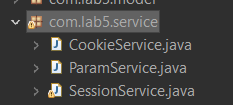
public interface OrderDAO extends JpaRepository<Orders, Integer>{}

public interface OrderDetailDAO extends JpaRepository<OrderDetails, Integer>{}

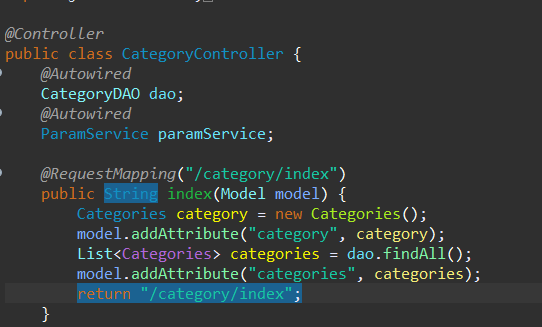
Bài 2: Xây dựng ứng dụng CRUD quản lý loại hàng (Category)

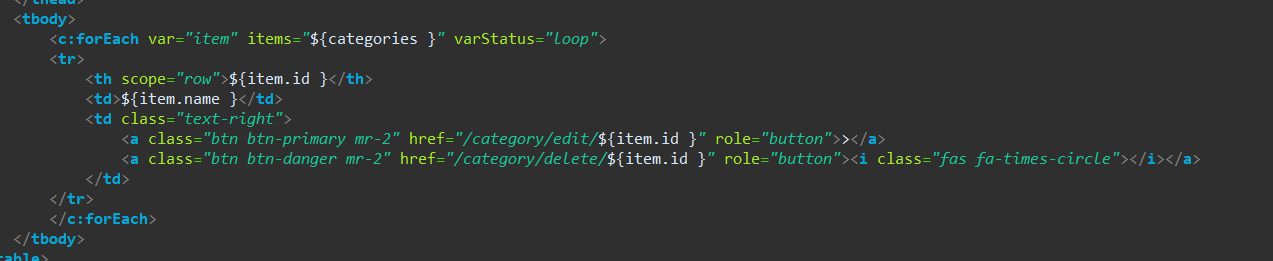


1. Tạo package service như bài lab4



1. Tạo lớp CategoryController





* Dùng @Autowired lên CategoryDao để sử truy vấn csdl
* Hàm index() gọi danh sách các loại sản phẩm lên thông qua CategoryDao và truyền vào index

@Autowired

CategoryDAO dao;

@Autowired

ParamService paramService;

@RequestMapping("/category/index")

public String index(Model model) {

Categories category = new Categories();

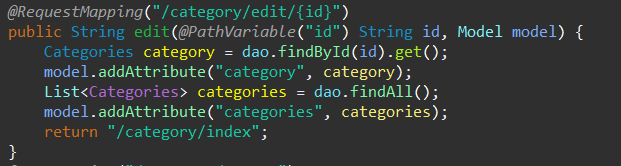
model.addAttribute("category", category);

List<Categories> categories = dao.findAll();

model.addAttribute("categories", categories);

return "/category/index";

}



* Hàm edit lấy biến id, sau đó tìm category phù hợp và truyền lại lên form

@RequestMapping("/category/edit/{id}")

public String edit(@PathVariable("id") String id, Model model) {

Categories category = dao.findById(id).get();

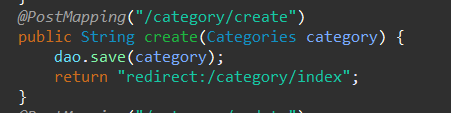
model.addAttribute("category", category);

List<Categories> categories = dao.findAll();

model.addAttribute("categories", categories);

return "/category/index";

}



* Hàm create sẽ tạo mới 1 category mới

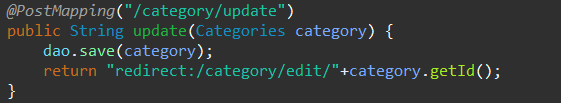
@PostMapping("/category/create")

public String create(Categories category) {

dao.save(category);

return "redirect:/category/index";

}



* Hàm update cập nhật lại thông ti của category trong csdl và truyền lại lên form

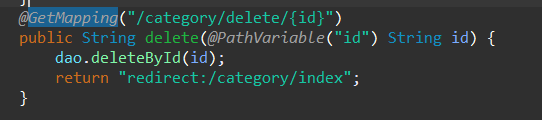
@PostMapping("/category/update")

public String update(Categories category) {

dao.save(category);

return "redirect:/category/edit/"+category.getId();

}



Hàm delete sẽ xóa category thông qua id của category được chọn

@GetMapping("/category/delete/{id}")

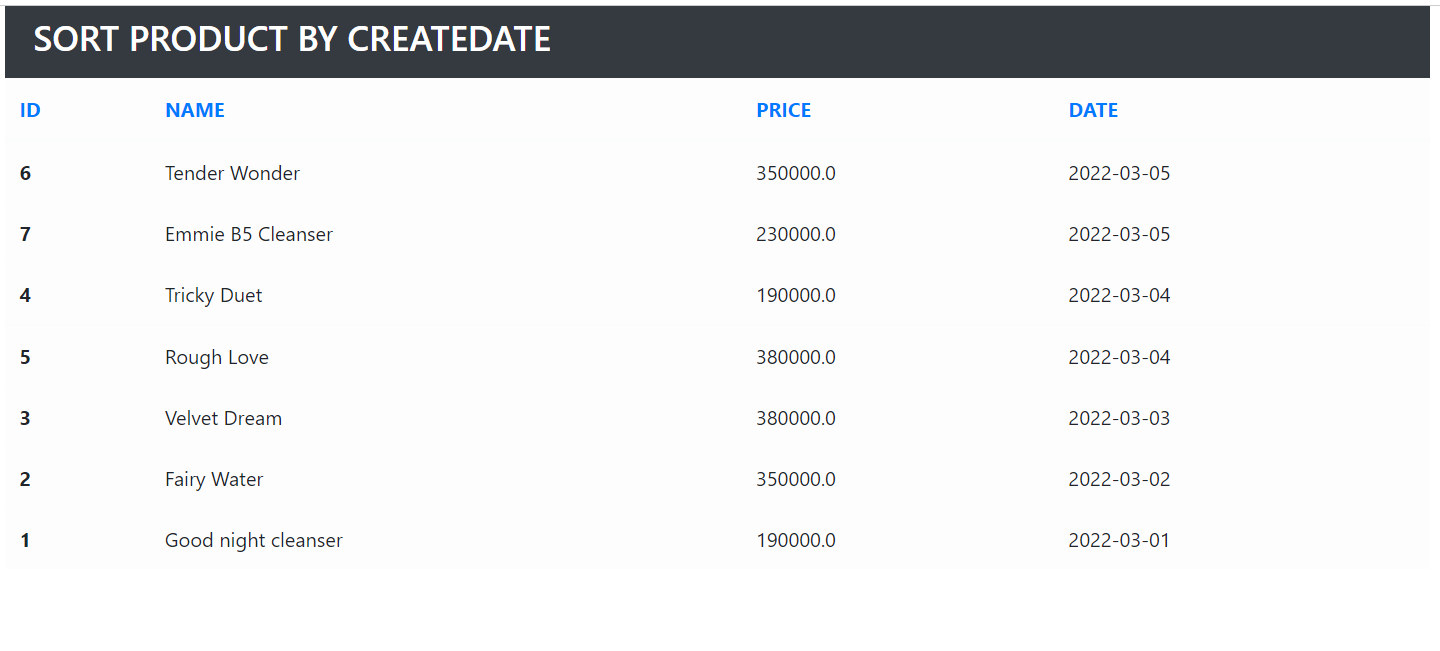
public String delete(@PathVariable("id") String id) {

dao.deleteById(id);

return "redirect:/category/index";

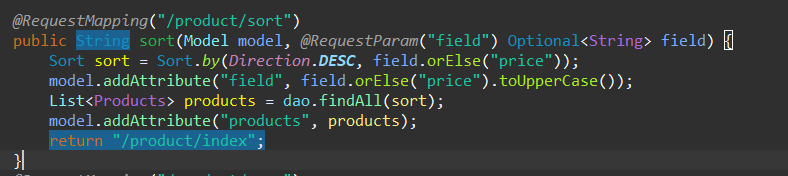
}

Bài 3: Xây dựng trang web cho phép sắp xếp giảm dần theo cột được chọn





* Thẻ <a> sẽ truyền thâm số field cho server



* Hàm sort sẽ sắp xếp lại bảng thông qua giá trị của field sau đó load lại trang

@Autowired

ProductDAO dao;

@RequestMapping("/product/sort")

public String sort(Model model, @RequestParam("field") Optional<String> field) {

Sort sort = Sort.by(Direction.DESC, field.orElse("price"));

model.addAttribute("field", field.orElse("price").toUpperCase());

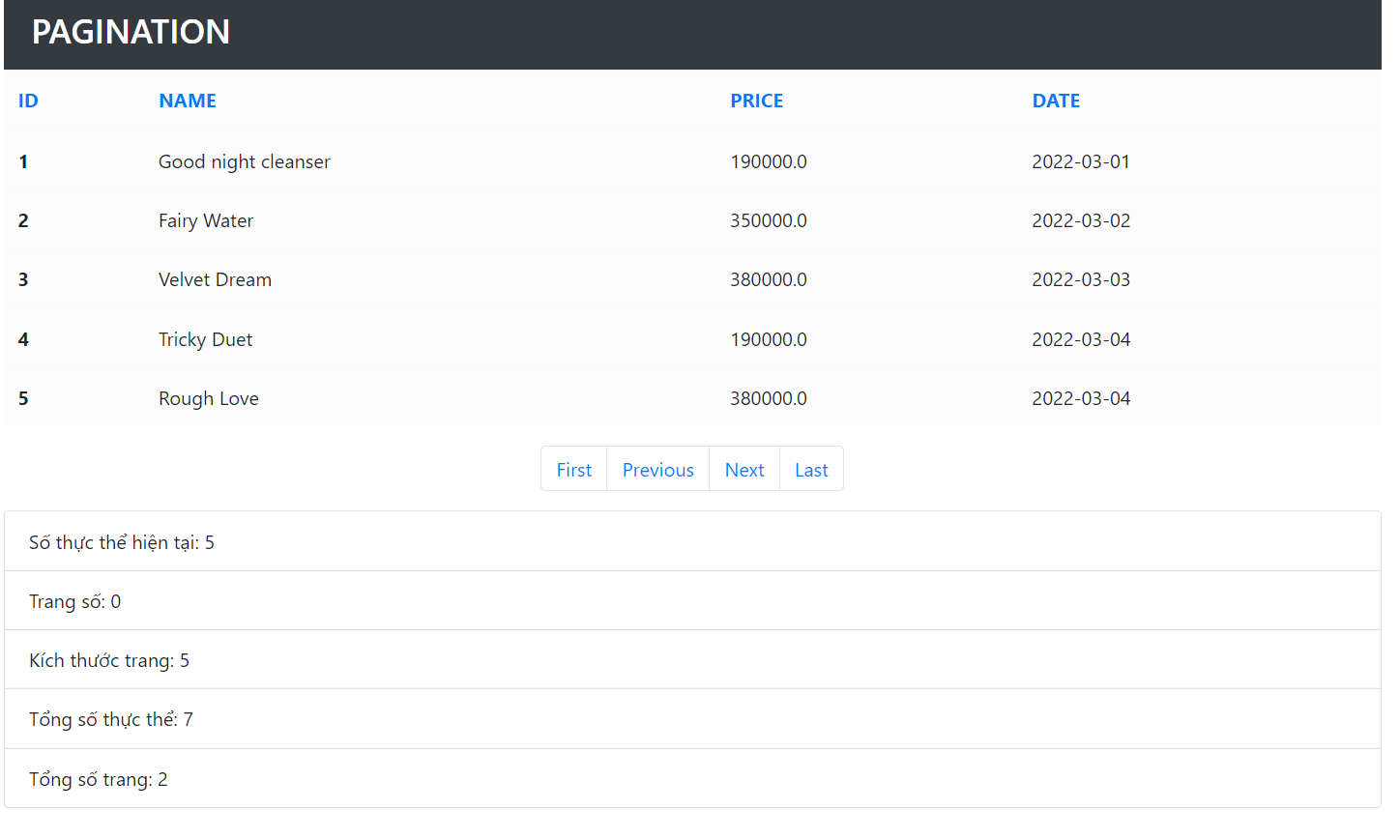
List<Products> products = dao.findAll(sort);

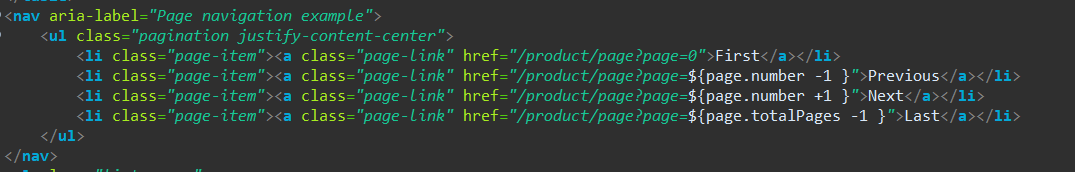
model.addAttribute("products", products);

return "/product/index";

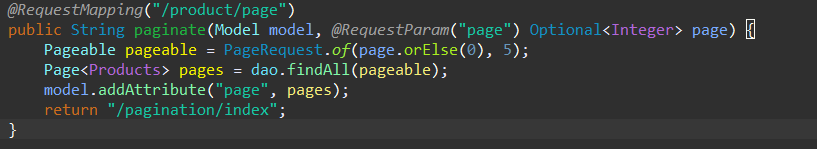
}

Bài 4: Xây dựng trang web cho phép phân trang

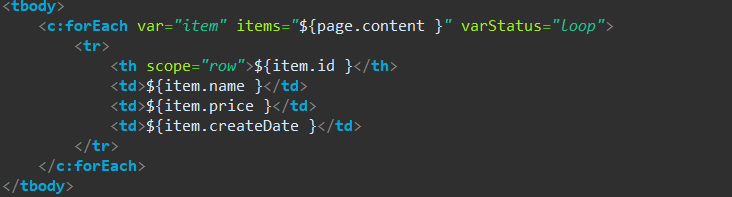




* Thẻ <a> truyền tham số page đến server:
  + First (trang thứ nhất): page = 0
  + Prev (trang trước): page = trang hiện tại - 1
  + Prev (trang sau): page = trang hiện tại + 1
  + First (trang cuối cùng): page = tổng số trang - 1



Hàm paginate sẽ lấy giá trị của “page” sau đó truy vấn trang theo tham số “page”



Ở vòng forEach để giá trị của items=”page.content” vì page content chứa các thực thể trong trang tương ứng

@RequestMapping("/product/page")

public String paginate(Model model, @RequestParam("page") Optional<Integer> page) {

Pageable pageable = PageRequest.of(page.orElse(0), 5);

Page<Products> pages = dao.findAll(pageable);

model.addAttribute("page", pages);

return "/pagination/index";

}

----Hết----