Nộp bài lab 5

Tên SV: Nguyễn Hải Đăng\_PC03033

**Bài 1 (2 điểm)**:Xây dựng **DAO** làm việc với **CSDL J5Shop.sql** (tải về từ tài nguyên học liệu) và thực hiện theo hướng dẫn sau:

• Chạy **MyShop.sql** để tạo CSDL **MyShop**

A screen shot of a computer

Description automatically generated with low confidence

**Sau khi chạy database MyShop thành công:**

A screenshot of a computer

Description automatically generated

**• Khai báo bổ sung thư viện cần thiết cho dự án**

A picture containing text, screenshot, font, line

Description automatically generated

**• Khai báo kết nối CSDL**

# MYSQL DB

spring.datasource.url=jdbc:sqlserver://localhost;database=MyShop;integratedSecurity=false;trustServerCertificate=true;

spring.datasource.username=sa

spring.datasource.password=123

spring.jpa.show-sql=false

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

spring.jpa.hibernate.dialect=org.hibernate.dialect.SQLServer2012Dialect

spring.jpa.hibernate.ddl-auto = none

**• Tạo các lớp Entity ánh xạ với CSDL**

**o Category**

@Data

@Entity

@Table(name = "Categories")

**public** **class** Category **implements** Serializable {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

Long id;

String name;

@OneToMany(mappedBy = "category")

List<Product> products;

}

**o Product**

@Data

@Entity

@Table(name = "Products")

**public** **class** Product **implements** Serializable{

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

Integer id;

String name;

String image;

Double price;

@Temporal(TemporalType.***DATE***)

@Column(name = "Createdate")

Date createDate = **new** Date();

Boolean available;

@ManyToOne @JoinColumn(name = "Categoryid")

Category category;

@OneToMany(mappedBy = "product")

List<OrderDetail> orderDetails;

}

**o Account**

@Data

@Entity

@Table(name = "Accounts")

**public** **class** Account **implements** Serializable{

@Id

String username;

String password;

String fullname;

String email;

String photo;

**boolean** activated;

**boolean** admin;

@OneToMany(mappedBy = "account")

List<Order> orders;

}

**o Order**

@Data

@Entity

@Table(name = "Orders")

**public** **class** Order **implements** Serializable{

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

Long id;

String address;

@Temporal(TemporalType.***DATE***)

@Column(name = "Createdate")

Date createDate = **new** Date();

@ManyToOne @JoinColumn(name = "Username")

Account account;

@OneToMany(mappedBy = "order")

List<OrderDetail> orderDetails;

}

**o OrderDetail**

@Data

@Entity

@Table(name = "Orderdetails")

**public** **class** OrderDetail **implements** Serializable{

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

Long id;

Double price;

Integer quantity;

@ManyToOne @JoinColumn(name = "Productid")

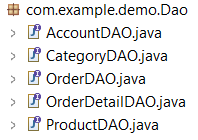
Product product;

@ManyToOne @JoinColumn(name = "Orderid")

Order order;

}

**• Tạo các interface DAO để làm việc với CSDL**

****

* AccountDAO

**public** **interface** AccountDAO **extends** JpaRepository<Account, String>{}

* CategoryDAO

**public** **interface** CategoryDAO **extends** JpaRepository<Category, String>{}

* OrderDAO

**public** **interface** OrderDAO **extends** JpaRepository<Order, Long>{}

* OrderDetailDAO

**public** **interface** OrderDetailDAO **extends** JpaRepository<OrderDetail, Long>{}

* ProductDAO

**public** **interface** ProductDAO **extends** JpaRepository<Product, Integer>{}

**Bài 2 (2 điểm)**:Xây dựng ứng dụng **CRUD** quản lý loại hàng **(Category)** như hình sau:

Để xây dựng ứng dụng **CRUD** quản lý loại hàng như trên, cần tạo các thành phần sau:

1. **CategoryController**

Chuẩn bị dữ liệu để hiển thị lên giao diện và xử lý các hành động tương tác của người sử dụng.

Sau đây là địa chỉ **URL (@/category/index)** chuẩn bị dữ liệu

Trong đó:

• Biến **item**: buộc lên form

• Biến **items**: hiển thị lên bảng

1. **View** **index.html, form.html** và **table.html**

• **Index.html**

<!DOCTYPE html>

<html xmlns:th=*"http://www.thymeleaf.org"*>

<head>

<meta charset=*"UTF-8"*>

<title>Category Management</title>

</head>

<body>

<h3>CATEGORY MANAGEMENT</h3>

<div th:include=*"fragments/form"*></div>

<hr>

<div th:include=*"fragments/table"*></div>

</body>

</html>

**• form.html**

<!DOCTYPE html>

<html xmlns:th=*"http://www.thymeleaf.org"*>

<head>

<meta charset=*"UTF-8"*>

</head>

<body>

<form action=*"/category/index"* method=*"post"* th:object=*"${item}"*>

<input type=*"text"* th:field=*"\*{id}"* placeholder=*"Category Id?"*/>

<input type=*"text"* th:field=*"\*{name}"* placeholder=*"Category Name?"*/>

<hr>

<button type=*"submit"* formaction=*"/category/create"*>Create</button>

<button type=*"submit"* formaction=*"/category/update"*>Update</button>

<a th:href=*"@{/category/delete/{id}(id=${item.id})}"*>Delete</a>

<a th:href=*"@{/category/index}"*>Reset</a>

</form>

</body>

</html>

**• table.html**

<!DOCTYPE html>

<html xmlns:th=*"http://www.thymeleaf.org"*>

<head>

<meta charset=*"UTF-8"*>

</head>

<body>

<table border=*"1"* style="width:*100%*">

<tr>

<th>Id</th>

<th>Name</th>

<th></th>

</tr>

<tr th:each=*"item : ${items}"*>

<td th:text=*"${item.id}"*></td>

<td th:text=*"${item.name}"*></td>

<td><a th:href=*"@{/category/edit/{id}(id=${item.id})}"*>Edit</a></td>

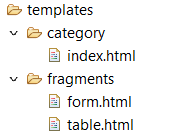
</tr>

</table>

</body>

</html

Thông tin tổ chức code html:

****

**Tổng hợp các hoạt động trên các trang jsp tương tác với phía server gồm**

1. Hoàn thiện **CategoryController** để xử lý các hành động **Edit, Create, Update và Delete.**

Bổ sung vào **CategoryController** các phương thức và ánh xạ với các địa chỉ **URL** có trên giao diện để xử lý các **request** của người sử dụng như sau:

@RequestMapping("/category/index")

**public** String index(Model model) {

Category item = **new** Category();

model.addAttribute("item", item);

List<Category> items = dao.findAll();

model.addAttribute("items", items);

**return** "category/index";

}

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

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

Category item = dao.findById(id).get();

model.addAttribute("item", item);

List<Category> items = dao.findAll();

model.addAttribute("items", items);

**return** "category/index";

}

@RequestMapping("/category/create")

**public** String create(Category item) {

dao.save(item);

**return** "redirect:/category/index";

}

@RequestMapping("/category/update")

**public** String update(Category item) {

dao.save(item);

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

}

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

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

dao.deleteById(id);

**return** "redirect:/category/index";

}

**Bài 3 (1 điểm)**:Giảng viên cho thêm quản lý **CRUD** của một table khác trong **CSDL MyShop**

**Tạo controller Account:**

@Controller

**public** **class** AccountController {

@Autowired

AccountDAO dao; // Làm việc với bảng Accounts

@RequestMapping("/account/index")

**public** String index(Model model) {

Account item = **new** Account();

model.addAttribute("item", item);

List<Account> items = dao.findAll();

model.addAttribute("items", items);

**return** "account/index";

}

@RequestMapping("/account/edit/{username}")

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

Account item = dao.findById(username).get();

model.addAttribute("item", item);

List<Account> items = dao.findAll();

model.addAttribute("items", items);

**return** "account/index";

}

@RequestMapping("/account/create")

**public** String create(Account item) {

dao.save(item);

**return** "redirect:/account/index";

}

@RequestMapping("/account/update")

**public** String update(Account item) {

dao.save(item);

**return** "redirect:/account/edit/" + item.getUsername();

}

@RequestMapping("/account/delete/{username}")

**public** String delete(@PathVariable("username") String username) {

dao.deleteById(username);

**return** "redirect:/account/index";

}

}

Tạo thêm file index.html trong thư mục account

<!DOCTYPE html>

<html xmlns:th=*"http://www.thymeleaf.org"*>

<head>

<meta charset=*"UTF-8"*>

<title>Account Management</title>

</head>

<body>

<h3>ACCOUNT MANAGEMENT</h3>

<form th:action=*"@{/account/index}"* th:object=*"${item}"* method=*"post"*>

<input th:field=*"\*{username}"* placeholder=*"Username"* /><br/>

<input th:field=*"\*{password}"* placeholder=*"Password"* /><br/>

<input th:field=*"\*{fullname}"* placeholder=*"Fullname"* /><br/>

<input th:field=*"\*{email}"* placeholder=*"Email"* /><br/>

<input th:field=*"\*{photo}"* placeholder=*"Photo"* /><br/>

<input type=*"checkbox"* th:field=*"\*{activated}"* /><label>Activated</label><br/>

<input type=*"checkbox"* th:field=*"\*{admin}"* /><label>Admin</label><br/>

<hr/>

<button type=*"submit"* formaction=*"/account/create"*>Create</button>

<button type=*"submit"* formaction=*"/account/update"*>Update</button>

</form>

<table border=*"1"* style="width:*100%*">

<tr>

<th>Username</th>

<th>Fullname</th>

<th>Email</th>

<th>Actions</th>

</tr>

<tr th:each=*"item : ${items}"*>

<td th:text=*"${item.username}"*></td>

<td th:text=*"${item.fullname}"*></td>

<td th:text=*"${item.email}"*></td>

<td>

<a th:href=*"@{/account/edit/{username}(username=${item.username})}"*>Edit</a>

<a th:href=*"@{/account/delete/{username}(username=${item.username})}"*>Delete</a>

</td>

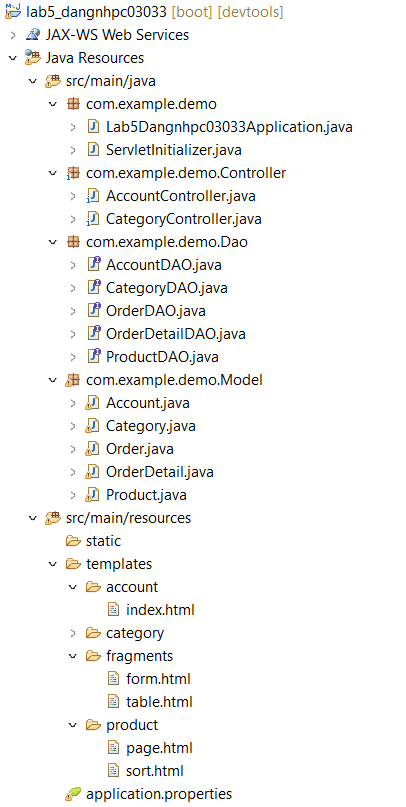
</tr>

</table>

</body>

</html>

Tổ chức dự án code:



**PHẦNII**

**Bài 3 (2 điểm)**

Xây dựng trang web cho phép sắp xếp giảm dần theo cột được chọn trên header của table như hình sau, khởi đầu thì sắp theo giá.

1. **Bổ sung code vào CategoryController**

@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<Product> items = productDAO.findAll(sort);

model.addAttribute("items", items);

**return** "product/sort";

}

1. **View sort.jsp hiển thị các sản phẩm**

<!DOCTYPE html>

<html xmlns:th=*"http://www.thymeleaf.org"*>

<head>

<meta charset=*"UTF-8"*>

<title>Sorting</title>

</head>

<body>

<h3>SORTING BY <span th:text=*"${field}"*></span></h3>

<table border=*"1"* style="width:*100%*">

<tr>

<th><a th:href=*"@{/product/sort(field='id')}"*>Id</a></th>

<th><a th:href=*"@{/product/sort(field='name')}"*>Name</a></th>

<th><a th:href=*"@{/product/sort(field='price')}"*>Price</a></th>

<th><a th:href=*"@{/product/sort(field='createDate')}"*>Date</a></th>

</tr>

<tr th:each=*"item : ${items}"*>

<td th:text=*"${item.id}"*></td>

<td th:text=*"${item.name}"*></td>

<td th:text=*"${item.price}"*></td>

<td th:text=*"${item.createDate}"*></td>

</tr>

</table>

</body>

</html>

Trang web trên đây chưa được sắp xếp, chỉ hiển thị tất cả. Mong muốn khi click váo các liên kết sau sẽ tiến hành sắp xếp theo giá trị của tham số field:

1. **Bổ sung thêm mã sắp xếp**

• **Optional** là vì mới chạy chưa chọn cột, trong trường hợp này sẽ sắp xếp theo **price**.

• Chú ý: **findAll(sort)**

**Bài 4 (2 điểm)**:Xây dựng trang web cho phép phân trang (mỗi trang 5 sản phẩm) như hình sau:

1. **Bổ sung code vào CategoryController**

@RequestMapping("/product/page")

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

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

Page<Product> page = productDAO.findAll(pageable);

model.addAttribute("page", page);

**return** "product/page";

}

1. **View page.jsp hiển thị các sản phẩm của trang và thông tin phân trang**

<!DOCTYPE html>

<html xmlns:th=*"http://www.thymeleaf.org"*>

<head>

<meta charset=*"UTF-8"*>

<title>Pagination</title>

</head>

<body>

<h3>PAGINATION</h3>

<table border=*"1"* style="width:*100%*">

<tr>

<th>Id</th>

<th>Name</th>

<th>Price</th>

<th>Date</th>

</tr>

<tr th:each=*"item : ${page.content}"*>

<td th:text=*"${item.id}"*></td>

<td th:text=*"${item.name}"*></td>

<td th:text=*"${item.price}"*></td>

<td th:text=*"${item.createDate}"*></td>

</tr>

</table>

<a th:href=*"@{/product/page(p=0)}"*>First</a>

<a th:href=*"@{/product/page(p=${page.number - 1})}"* th:if=*"${page.number > 0}"*>Previous</a>

<a th:href=*"@{/product/page(p=${page.number + 1})}"* th:if=*"${page.number + 1 < page.totalPages}"*>Next</a>

<a th:href=*"@{/product/page(p=${page.totalPages - 1})}"* th:if=*"${page.totalPages > 1}"*>Last</a>

<ul>

<li>Current entities: <span th:text=*"${page.numberOfElements}"*></span></li>

<li>Page number: <span th:text=*"${page.number}"*></span></li>

<li>Page size: <span th:text=*"${page.size}"*></span></li>

<li>Total entities: <span th:text=*"${page.totalElements}"*></span></li>

<li>Total pages: <span th:text=*"${page.totalPages}"*></span></li>

</ul>

</body>

</html>

Hiệu chỉnh **CategoryController** để khi click vào các liên kết điều hướng thì truy vấn trang theo tham số

**Bài 5 (1 điểm) Giảng viên cho thêm kết hợp phân trang và sắp xếp**

@GetMapping("/account/sortPage")

**public** String getCategories(@RequestParam(defaultValue = "0") **int** page,

@RequestParam(defaultValue = "3") **int** pageSize,

Model model) {

// Tạo đối tượng Pageable với trang hiện tại, kích thước trang và sắp xếp theo email

PageRequest pageable = PageRequest.*of*(page, pageSize, Sort.*by*("username"));

// Lấy trang dữ liệu từ DAO sử dụng Pageable

Page<Account> accountPage = dao.findAll(pageable);

model.addAttribute("item", accountPage.getContent());

model.addAttribute("page", accountPage.getNumber());

model.addAttribute("totalPages", accountPage.getTotalPages());

**return** "product/index";

}

Đầu tiên đưa page đầu là 0

Và set mỗi page sẽ có 3 dữ liệu

Sao đó thì sẽ sắp xếp theo họ tên của account thông qua lệnh Sort.*by*("username") và dùng PageRequest để trả về dữ liệu gồm trang và số lượng của mỗi trang và cách sắp xếp tiếp theo đó là hiển thị toàn bộ qua lệnh findAll và đưa vào List accountPage và chuyển toàn bộ qua cho giao diện.

Thông tin form index.html:

<!DOCTYPE html>

<html xmlns:th=*"http://www.thymeleaf.org"*>

<head>

<meta charset=*"UTF-8"*>

<title>Insert title here</title>

<!-- Thêm liên kết CSS của Bootstrap -->

<link rel=*"stylesheet"*

href=*"https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"*>

</head>

<body>

<div class=*"container"*>

<table class=*"table table-striped"*>

<thead>

<tr>

<th>STT</th>

<th>Email</th>

</tr>

</thead>

<tbody>

<tr th:each=*"account, accountCount : ${item}"*>

<td th:text=*"${accountCount.index + 1}"*></td>

<td th:text=*"${account.email}"*></td>

</tr>

</tbody>

</table>

<!-- Hiển thị phân trang -->

<nav aria-label=*"Page navigation example"*>

<ul class=*"pagination"*>

<li class=*"page-item"*><a class=*"page-link"* href=*"#"*

th:href=*"@{/account/sortPage(page=${page - 1})}"* th:if=*"${page > 0}"*>Previous</a>

</li>

<li class=*"page-item"*

th:each=*"pageNumber : ${#numbers.sequence(0, totalPages - 1)}"*>

<a class=*"page-link"* href=*"#"*

th:href=*"@{/account/sortPage(page=${pageNumber})}"*

th:text=*"${pageNumber + 1}"*></a>

</li>

<li class=*"page-item"*><a class=*"page-link"* href=*"#"*

th:href=*"@{/account/sortPage(page=${page + 1})}"*

th:if=*"${page + 1 < totalPages}"*>Next</a></li>

</ul>

</nav>

</div>

<!-- Thêm liên kết tới thư viện JavaScript của Bootstrap -->

<script

src=*"https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"*></script>

</body>

</html>