LAB6: **JPAREPOSITORY 2 – Java 5**

Tên sv: Nguyễn Hải Đăng PC03033

**Bài 1 (2 điểm)** Xây dựng trang tìm kiếm sản phẩm theo khoảng giá như hình sau

//ảnh

Để thực hiện theo yêu cầu trên, cần tạo và viết mã cho các thành phần cần thiết sau đây:

1. **ProductController**

@Autowired

**private** **final** ProductDAO dao;

@Autowired

**private** **final** SessionService session;

**public** ProductController(ProductDAO dao, SessionService session) {

**this**.dao = dao;

**this**.session = session;

}

@RequestMapping("/search")

**public** String search(Model model, @RequestParam("min") Optional<Double> min,

@RequestParam("max") Optional<Double> max) {

**double** minPrice = min.orElse(Double.***MIN\_VALUE***);

**double** maxPrice = max.orElse(Double.***MAX\_VALUE***);

List<Product> items = dao.findByPriceBetween(minPrice, maxPrice);

model.addAttribute("items", items);

**return** "product/search";

}

• Nhận các tham số **min** và **max**

• Gọi phương thức **findByPriceBetween()** của **ProductDAO**.

1. **ProductDAO**

// @Query("SELECT o FROM Product o WHERE o.price BETWEEN ?1 AND ?2")

// List<Product> findByPrice(double minPrice, double maxPrice);

• Sử dụng @Query(JPQL) để truy vấn dữ liệu

• Chú ý vị trí các tham số tương ứng với vị trí đối số của phương thức truy vấn

1. View **search.html**

<form action=*"/product/search"* method=*"get"*>

<input name=*"min"* th:value=*"${param.min}"* placeholder=*"Min Price?"*>

<input name=*"max"* th:value=*"${param.max}"* placeholder=*"Max Price?"*>

<button type=*"submit"*>Search</button>

</form>

<table>

<thead>

<tr>

<th>ID</th>

<th>Name</th>

<th>Price</th>

<th>Create Date</th>

</tr>

</thead>

<tbody>

<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>

</tbody>

</table>

**Bài 2 (2 điểm)** Xây dựng trang web tìm kiếm sản phẩm theo tên và kết hợp với phân trang các sản phẩm như hình sau:

//ảnh

Để thực hiện yêu cầu trên, cần phải viết mã cho các thành phần sau đây:

1. **ProductController**

@RequestMapping("/search-and-page")

**public** String searchAndPage(Model model, @RequestParam("keywords") Optional<String> kw,

@RequestParam("p") Optional<Integer> p) {

String kwords = kw.orElse((String) session.get("keywords", ""));

session.set("keywords", kwords);

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

Page<Product> page = dao.findAllByNameLike("%" + kwords + "%", pageable);

model.addAttribute("page", page);

**return** "product/search-and-page";

}

• Nhận từ khóa tìm kiếm keywords và trang số p thông qua tham số

• Ghi nhớ từ khóa vào session để duy trì trong quá trình phân trang

• Gọi findByKeywords() của ProductDAO để truy vấn trang theo từ khóa

1. **ProductDAO**

// @Query("SELECT o FROM Product o WHERE o.name LIKE ?1")

// Page<Product> findByKeywords(String keywords, Pageable pageable);

• Chú ý: **Pageable** và **Sort** nên là đối số cuối cùng để tránh nhầm lẫn với các vị trí tham số trong **JPQL**

**3. View search-and-page.html**

<form action=*"/product/search-and-page"* method=*"get"*>

<input name=*"keywords"* th:value=*"${keywords}"*>

<button type=*"submit"*>Search</button>

</form>

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

<thead>

<tr>

<th>ID</th>

<th>Name</th>

<th>Price</th>

<th>Create Date</th>

</tr>

</thead>

<tbody>

<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>

</tbody>

</table>

<ul>

<li th:if=*"${page.hasPrevious()}"*><a

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

<li th:if=*"${page.hasPrevious()}"*><a

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

<li th:if=*"${page.hasNext()}"*><a

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

<li th:if=*"${page.hasNext()}"*><a

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

</ul>

<ul>

<li>Số thực thể hiện tại: <span

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

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

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

<li>Tổng số thực thể: <span th:text=*"${page.totalElements}"*></span></li>

<li>Tổng số trang: <span th:text=*"${page.totalPages}"*></span></li>

</ul>

• Chú ý: các thuộc tính của **Page** giúp chúng ta triển khai việc phân trang một cách dễ dàng và thuận lợi.

**Bài 3 (1 điểm)** Xây dựng trang web cho phép thực hiện tổng hợp dữ liệu tồn kho như hình sau:  
//ảnh

Để thực hiện trang web theo yêu cầu thì cần viết mã cho các thành phần sau đây:

1. **ReportController**

@RequestMapping("/report/inventory-by-category")

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

List<Report> items = dao.getInventoryByCategory();

model.addAttribute("items", items);

**return** "product/inventory-by-category";

}

• Gọi phương thức **getInventoryByCategory**() của **ProductDAO** để lấy dữ liệu tổng hợp

• Cũng cần tạo một **entity class Report** để chứa dữ liệu

1. **ProductDAO**

@Query("SELECT new Report(o.category, sum(o.price), count(o)) " + "FROM Product o " + "GROUP BY o.category "

+ "ORDER BY sum(o.price) DESC")

List<Report> getInventoryByCategory();

• Do **Report** không ánh xạ đến một **table** nào nên chúng ta cần phải new **Report**() và **truyền** các đối số phù hợp để tạo đối tượng từ dữ liệu truy vấn được.

1. **Report**

@Entity

@Data

@NoArgsConstructor

@AllArgsConstructor

**public** **class** Report {

@Id

Serializable group;

Double sum;

Long count;

}

• Lưu ý: Để group by **được** thì biểu thức trong group by phải **Serializable**

1. **View inventory-by-category.html**

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

<tr>

<th>Loại hàng</th>

<th>Tổng giá</th>

<th>Số sản phẩm</th>

</tr>

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

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

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

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

</tr>

</table>

**PHẦNII**

**Bài 4 (2 điểm)** Hãy xây dựng trang web tìm kiếm như bài 1 (hình sau) nhưng sử dụng **DSL** để viết phương thức truy vấn thay cho **@Query(JPQL)**

Trong **ProductDAO** thay đoạn mã sau

// @Query("SELECT o FROM Product o WHERE o.price BETWEEN ?1 AND ?2")

// List<Product> findByPrice(double minPrice, double maxPrice);

Bằng đoạn mã này

List<Product> findByPriceBetween(**double** minPrice, **double** maxPrice);

**Bài 5 (2 điểm)** Hãy xây dựng trang web tìm kiếm theo từ khóa kết hợp với phân trang sản phẩm như bài 2 (hình sau) nhưng sử dụng **DSL** để viết phương thức truy vấn thay cho **@Query(JPQL)**

Trong **ProductDAO** thay đoạn mã sau

// @Query("SELECT o FROM Product o WHERE o.name LIKE ?1")

// Page<Product> findByKeywords(String keywords, Pageable pageable);

Bằng đoạn mã sau

Page<Product> findAllByNameLike(String keywords, Pageable pageable);

Chú ý: với **DSL**, các phương thức truy vấn có phân trang và tìm kiếm nên sử dụng **findAllBy**\_ thay vì **findBy**\_

**Bài 6 (1 điểm)**

Giảng viên cho thêm