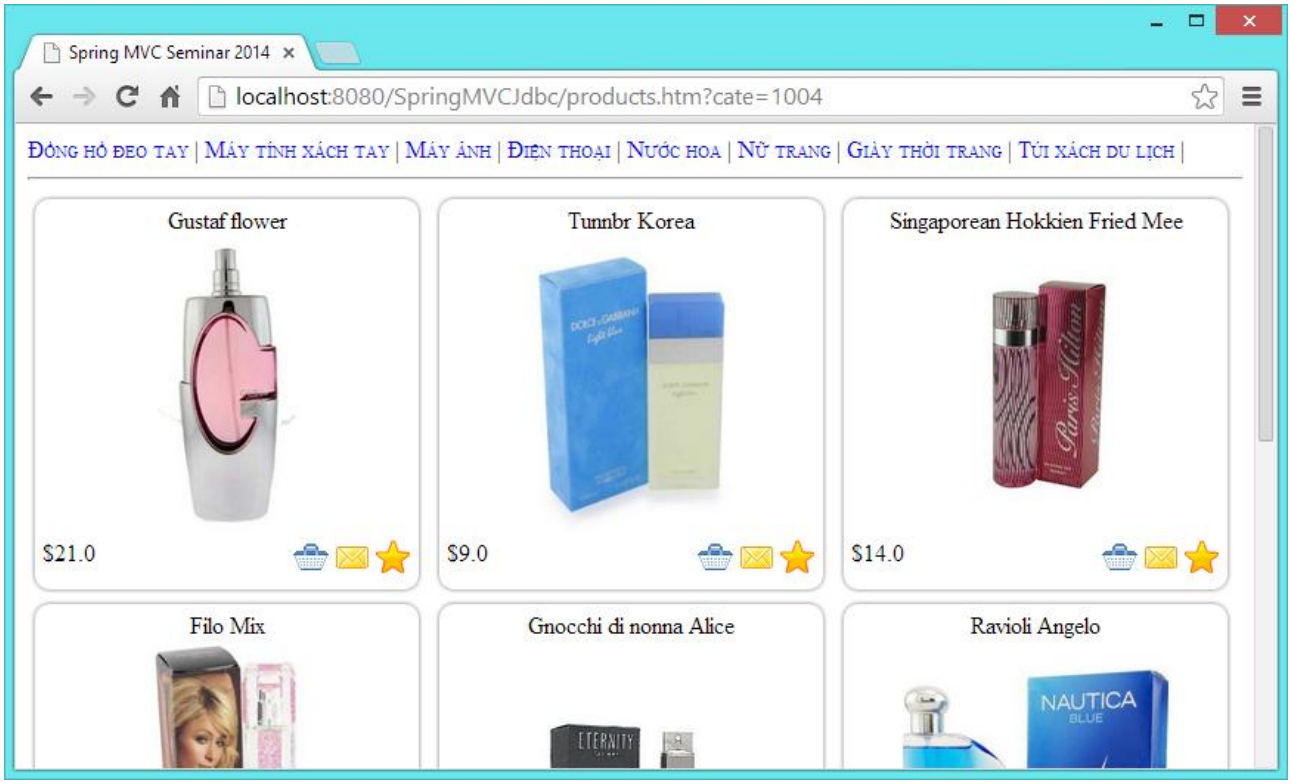
MỤC TIÊU

Kết thúc bài thực hành này, bạn có khả năng

- ✓ Tạo menu động
- ✓ Trình bày hàng hóa dạng cột với CSS

MÔ TẢ

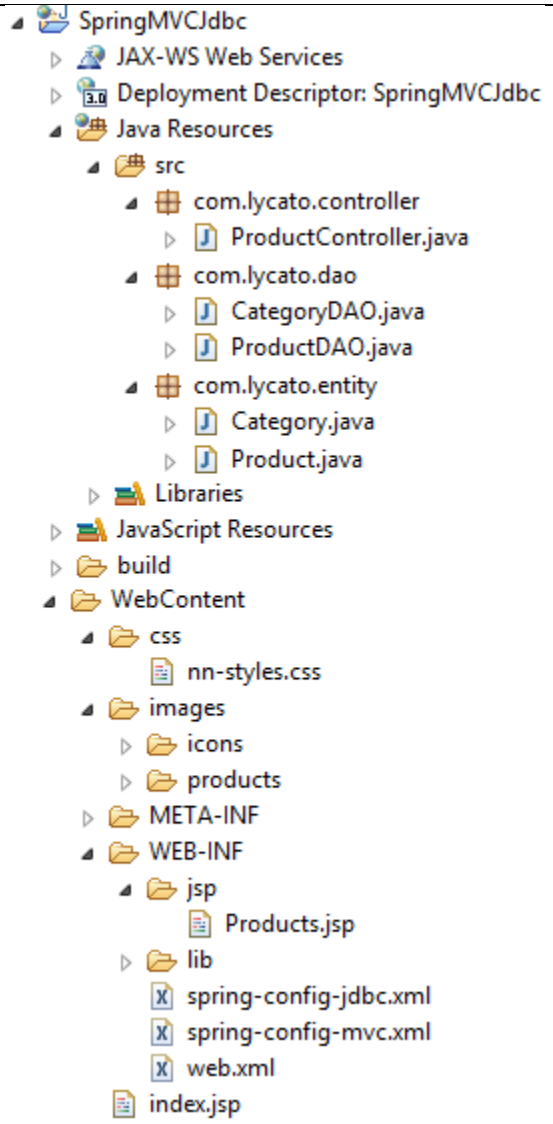


Hoạt động:

- ✓ Chạy products.htm hiển thị trang web gồm menu loại hàng hóa (đọc từ CSDL) và tất cả hàng hóa
- ✓ Nhấp [một loại hàng hóa] sẽ hiển thị các hàng hóa thuộc loại được chọn

THỰC HIỆN

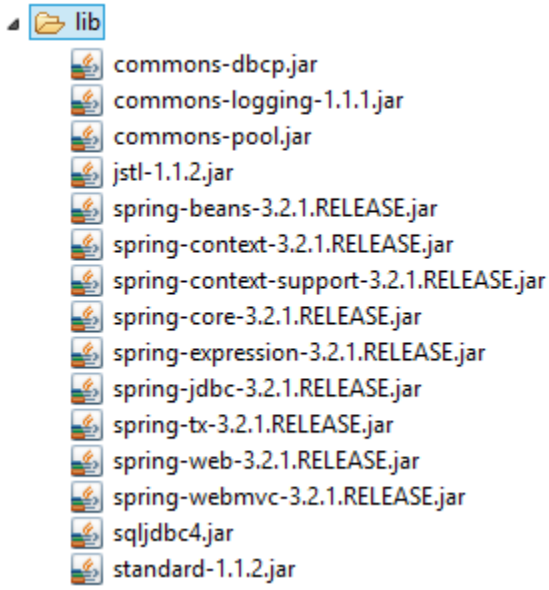
Trong bài này, bạn sẽ phải tạo một project có tổ chức như sau:



- ✓ Bước 1: Thư viện và cấu hình project
- ✓ Bước 2: CSDL
- ✓ Bước 4: Tạo lớp DAO và Entity
- ✓ Bước 5: Tạo giao diện
- ✓ Bước 6: Tạo Controller
- ✓ Bước 7: Chạy

Bước 1: Thư viện và cấu hình project

Thư viện



Bên cạnh các thư viện của Thư viện cần thiết cho ứng dụng

- ✓ SQLServerDriver
  - sqljdbc4.jar
- ✓ JdbcTemplate
  - commons-dbcp.jar
  - spring-jdbc-3.2.1.RELEASE.jar
  - spring-tx-3.2.1.RELEASE.jar

Cấu hình

❖ **Web.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app                                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd" id="WebApp_ID" version="3.0">
	<display-name>SpringMVCEmail</display-name>
	<welcome-file-list>
		<welcome-file>index.jsp</welcome-file>
	</welcome-file-list>
	<servlet>
		<servlet-name>dispatcher</servlet-name>
		<servlet-class>
			org.springframework.web.servlet.DispatcherServlet
		</servlet-class>
		<init-param>
			<param-name>contextConfigLocation</param-name>
			<param-value>/WEB-INF/spring-config-*.xml</param-value>
		</init-param>
		<load-on-startup>1</load-on-startup>
```

```
	</servlet>
	<servlet-mapping>
		<servlet-name>dispatcher</servlet-name>
		<url-pattern>*.htm</url-pattern>
	</servlet-mapping>
</web-app>
```

Cấu hình để Spring MVC nạp nhiều file cấu hình: spring-config-*.xml. Dấu * sẽ đại diện cho nhóm ký tự bất kỳ. Cụ thể ở bài này là mvc, gmail và upload

- **spring-config-mvc.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xmlns:context="http://www.springframework.org/schema/context"
	xmlns:p="http://www.springframework.org/schema/p"
	xmlns:tx="http://www.springframework.org/schema/tx"
	xmlns:mvc="http://www.springframework.org/schema/mvc"
	xsi:schemaLocation="http://www.springframework.org/schema/beans
			http://www.springframework.org/schema/beans/spring-beans-3.2.xsd
			http://www.springframework.org/schema/context
			http://www.springframework.org/schema/context/spring-context-3.2.xsd
			http://www.springframework.org/schema/tx
			http://www.springframework.org/schema/tx/spring-tx-3.2.xsd
			http://www.springframework.org/schema/mvc
			http://www.springframework.org/schema/mvc/spring-mvc-3.2.xsd">

	<!-- Declare a view resolver -->
	<bean id="viewResolver" p:prefix="/WEB-INF/jsp/" p:suffix=".jsp"
	class="org.springframework.web.servlet.view.InternalResourceViewResolver"/>

	<!-- Spring MVC Annotation -->
	<mvc:annotation-driven />
	<context:annotation-config />

	<!-- Where to find component -->
	<context:component-scan base-package="com.lycato" />
</beans>
```

- ✓ Khai báo bean InternalResourceViewResolver để xử lý view
- ✓ Chỉ rõ package tìm kiếm các component là com.lycato
- ✓ Chỉ rõ ứng dụng Spring này được phép sử dụng annotation

- **spring-config-jdbc.xml**

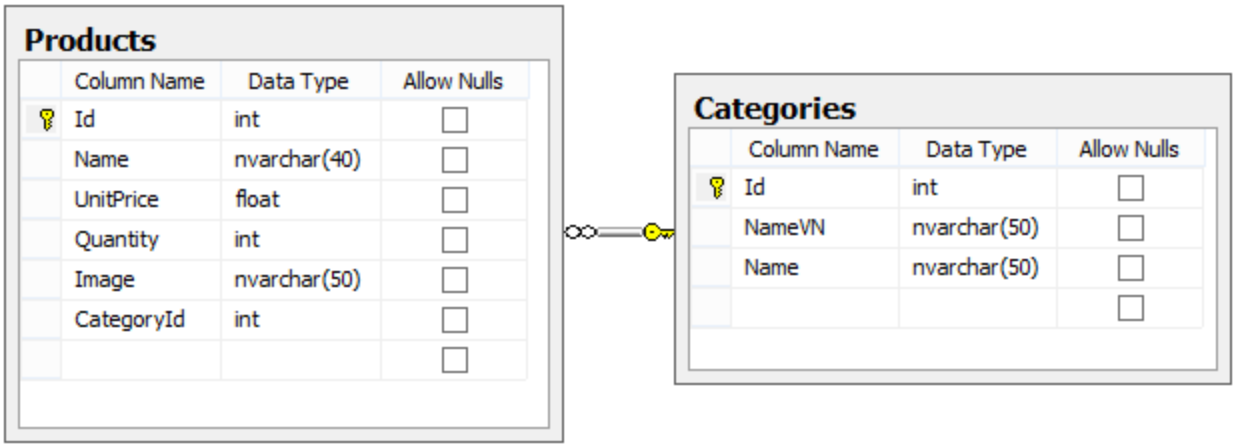
```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xmlns:context="http://www.springframework.org/schema/context"
	xmlns:p="http://www.springframework.org/schema/p"
	xmlns:tx="http://www.springframework.org/schema/tx"
	xmlns:mvc="http://www.springframework.org/schema/mvc"
	xsi:schemaLocation="http://www.springframework.org/schema/beans
			http://www.springframework.org/schema/beans/spring-beans-3.2.xsd
			http://www.springframework.org/schema/context
			http://www.springframework.org/schema/context/spring-context-3.2.xsd
			http://www.springframework.org/schema/tx
			http://www.springframework.org/schema/tx/spring-tx-3.2.xsd
```

```
                    http://www.springframework.org/schema/mvc
                    http://www.springframework.org/schema/mvc/spring-mvc-3.2.xsd">

        <bean id="dataSource" destroy-method="close"
                class="org.apache.commons.dbcp.BasicDataSource"
                p:driverClassName="com.microsoft.sqlserver.jdbc.SQLServerDriver"
                p:url="jdbc:sqlserver://localhost:1433;DatabaseName=Seminar"
                p:username="sa"
                p:password="songlong"/>

        <bean id="jdbcTemplate"
                class="org.springframework.jdbc.core.JdbcTemplate">
                <constructor-arg ref="dataSource"/>
        </bean>

</beans>
```

Trong file cấu hình này bạn phải khai báo 2 bean.

- ✓ BasicDataSource: bean này cấu hình các thông số kết nối CSDL
- ✓ JdbcTemplate: bean này được khai báo đến làm việc với CSDL được tiêm vào và sử dụng sau này trong ứng dụng

Bước 2: CSDL



Hình: CSDL Seminar chứa 2 bảng Products và Categories

Bước 4: Tạo lớp mô tả và truy xuất dữ liệu

Lớp mô tả dữ liệu (Entity)

Category.java

```
package com.lycato.entity;

public class Category {
        Integer id;
        String name, namevn;

        public Integer getId() {
                return id;
        }
```

```
	public void setId(Integer id) {
		this.id = id;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getNamevn() {
		return namevn;
	}
	public void setNamevn(String namevn) {
		this.namevn = namevn;
	}
}
```

Product.java

```
package com.lycato.entity;

public class Product {
	Integer id, quantity, categoryId;
	String name, image;
	Double unitPrice;

	public Integer getId() {
		return id;
	}
	public void setId(Integer id) {
		this.id = id;
	}
	public Integer getQuantity() {
		return quantity;
	}
	public void setQuantity(Integer quantity) {
		this.quantity = quantity;
	}
	public Integer getCategoryId() {
		return categoryId;
	}
	public void setCategoryId(Integer categoryId) {
		this.categoryId = categoryId;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getImage() {
		return image;
	}
	public void setImage(String image) {
		this.image = image;
	}
	public Double getUnitPrice() {
		return unitPrice;
	}
	public void setUnitPrice(Double unitPrice) {
		this.unitPrice = unitPrice;
	}
```

```
}
```

Lớp truy xuất dữ liệu (DAO)

Lớp này chứa các phương thức thao tác dữ liệu (thêm, sửa, xóa) và truy vấn dữ liệu.

- ✓ Insert(): thêm
- ✓ Update(): sửa
- ✓ Delete(): xóa
- ✓ getXyz(): truy vấn

Lớp nào được chú thích bởi @Repository để có thể tiêm vào ProductController trong ứng dụng bởi @Autowire để sử dụng sau này.

CategoryDAO.java

```
package com.lycato.dao;

import java.io.Serializable;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.stereotype.Repository;

import com.lycato.entity.Category;

@Repository
public class CategoryDAO{
	/**
	 * Inject từ <bean class="...JdbcTemplate>
	 */
	@Autowired
	protected JdbcTemplate jdbc;

	/**
	 * Thêm mới 1 thực thể
	 * @param entity là thực thể mới
	 */
	public void insert(Category entity) {
		String sql = "INSERT INTO Categories (Name, NameVN) VALUES (?,?)";
		jdbc.update(sql, entity.getName(), entity.getNamevn());
	}

	/**
	 * Cập nhật thực thể
	 * @param entity là thực thể cần cập nhật
	 */
	public void update(Category entity) {
		String sql = "UPDATE Categories SET name=?, Description=? WHERE Id=?";
		jdbc.update(sql, entity.getName(), entity.getNamevn(), entity.getId());
	}

	/**
	 * Xóa thực thể theo mã
	 * @param id mã thực thể cần xóa
	 */
	public void delete(Serializable id) {
```

```
		String sql = "DELETE FROM Categories WHERE Id=?";
		jdbc.update(sql, id);
	}

	/**
	 * Truy vấn 1 thực thể theo mã
	 * @param id mã thực thể cần truy vấn
	 * @return thực thể truy vấn được
	 */
	public Category getById(Serializable id) {
		String sql = "SELECT * FROM Categories WHERE Id=?";
		return jdbc.queryForObject(sql, getRowMapper(), id);
	}

	/**
	 * Truy vấn tất cả các thực thể
	 * @return danh sách thực thể truy vấn được
	 */
	public List<Category> getAll() {
		String sql = "SELECT * FROM Categories";
		return getBySql(sql);
	}

	/**
	 * Truy vấn các thực thể theo câu lệnh sql
	 * @param sql câu lệnh truy vấn
	 * @return danh sách thực thể truy vấn được
	 */
	protected List<Category> getBySql(String sql) {
		return jdbc.query(sql, getRowMapper());
	}

	/**
	 * Ánh xạ cấu trúc bản ghi theo thuộc tính của bean
	 * @return ánh xạ bản ghi theo thuộc tính bean
	 */
	private RowMapper<Category> getRowMapper() {
		return new BeanPropertyRowMapper<Category>(Category.class);
	}
}
```

ProductDAO.java

```
package com.lycato.dao;

import java.io.Serializable;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.stereotype.Repository;

import com.lycato.entity.Product;

@Repository
public class ProductDAO{
	/**
	 * Inject từ <bean class="...JdbcTemplate>
	 */
	@Autowired
	protected JdbcTemplate jdbc;
```

```
	/**
	 * Thêm mới 1 thực thể
	 * @param entity là thực thể mới
	 */
	public void insert(Product entity) {
		String sql = "INSERT INTO Products (Name, UnitPrice, Quantity, Image,
CategoryId) VALUES (?,?,?,?,?)";
		jdbc.update(sql, entity.getName(), entity.getUnitPrice(),
entity.getQuantity(), entity.getImage(), entity.getCategoryId());
	}

	/**
	 * Cập nhật thực thể
	 * @param entity là thực thể cần cập nhật
	 */
	public void update(Product entity) {
		String sql = "UPDATE Products SET Name=?, UnitPrice=?, Quantity=?, Image=?,
CategoryId=? WHERE Id=?";
		jdbc.update(sql, entity.getName(), entity.getUnitPrice(),
entity.getQuantity(), entity.getImage(), entity.getCategoryId(), entity.getId());
	}

	/**
	 * Xóa thực thể theo mã
	 * @param id mã thực thể cần xóa
	 */
	public void delete(Serializable id) {
		String sql = "DELETE FROM Products WHERE Id=?";
		jdbc.update(sql, id);
	}

	/**
	 * Truy vấn 1 thực thể theo mã
	 * @param id mã thực thể cần truy vấn
	 * @return thực thể truy vấn được
	 */
	public Product getById(Serializable id) {
		String sql = "SELECT * FROM Products WHERE Id=?";
		return jdbc.queryForObject(sql, getRowMapper(), id);
	}

	/**
	 * Truy vấn tất cả các thực thể
	 * @return danh sách thực thể truy vấn được
	 */
	public List<Product> getAll() {
		String sql = "SELECT * FROM Products";
		return getBySql(sql);
	}

	/**
	 * Truy vấn các thực thể theo câu lệnh sql
	 * @param sql câu lệnh truy vấn
	 * @return danh sách thực thể truy vấn được
	 */
	protected List<Product> getBySql(String sql) {
		return jdbc.query(sql, getRowMapper());
	}

	/**
	 * Truy vấn thực thể theo tên
	 * @param name tên của thực thể cần truy vấn
```

```
	 * @return danh sách thực thể truy vấn được
	 */
	public List<Product> getByName(String name) {
		String sql = "SELECT * FROM Products WHERE Name LIKE ?";
		return jdbc.query(sql, getRowMapper(), "%" + name + "%");
	}

	/**
	 * Ánh xạ cấu trúc bản ghi theo thuộc tính của bean
	 * @return ánh xạ bản ghi theo thuộc tính bean
	 */
	private RowMapper<Product> getRowMapper() {
		return new BeanPropertyRowMapper<Product>(Product.class);
	}

	/**
	 * Truy vấn thực thể theo mã loại
	 * @param categoryId là mã loại
	 * @return danh sách thực thể truy vấn được
	 */
	public List<Product> getByCategoryId(Integer categoryId) {
		String sql = "SELECT * FROM Products WHERE CategoryId=?";
		return jdbc.query(sql, getRowMapper(), categoryId);
	}
}
```

Bước 5: Tạo giao diện

Products.jsp

```
<%@page pageEncoding="utf-8" contentType="text/html; charset=utf-8" %>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<title>Spring MVC Seminar 2014</title>
	<link href="css/nn-styles.css" rel="stylesheet">
</head>
<body>
	<!--MENU LOẠI HÀNG-->
	<c:forEach var="c" items="${categories}">
		<a href="products.htm?cate=${c.id}">${c.namevn}</a> |
	</c:forEach>
	<hr>
	<!--DANH SÁCH HÀNG HÓA-->
	<c:forEach var="p" items="${products}">
	<ul>
		<li>${p.name}</li>
		<li><img src="images/products/${p.image}"></li>
		<li>
			<div style="float: left">$${p.unitPrice}</div>
			<div style="float: right">
				<img src="images/icons/Basket.png">
				<img src="images/icons/Letter.png">
				<img src="images/icons/Favourites.png">
			</div>
		</li>
	</ul>
	</c:forEach>
</body>
```

```
</html>
```

nn-styles.css

Định nghĩa style trình bày cho mỗi mặt hàng.

```
ul{
	padding: 5px;
	margin: 5px;
	list-style: none;
	display: inline-block;
	width: 250px;
	text-align: center;
	border-radius: 10px;
	box-shadow:0 0 5px gray;
}
li>img{
	max-width: 180px;
	height: 200px;
}
a{
	text-decoration: none;
	color:blue;
	font-variant: small-caps;
}
a:hover {
	color:red;
}
```

Bước 6: Tạo Controller

```
package com.lycato.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;

import com.lycato.dao.CategoryDAO;
import com.lycato.dao.ProductDAO;
import com.lycato.entity.Category;

@Controller
public class ProductController {
	/**
	 * Inject từ @Repository ProductDAO
	 */
	@Autowired
	ProductDAO pdao;

	/**
	 * Inject từ @Repository CategoryDAO
	 */
	@Autowired
	CategoryDAO cdao;

	/**
	 * Truy vấn List<Customer> và đặt vào model
```

```
	 */
	@ModelAttribute("categories")
	public List<Category> getCategories() {
		return cdao.getAll();
	}

	/**
	 * GET|POST: products.htm
	 */
	@RequestMapping("products")
	public String showProducts(ModelMap model,
			@RequestParam(value="cate", required=false) Integer categoryId) {
		if(categoryId == null){
			model.addAttribute("products", pdao.getAll());
		}
		else{
			model.addAttribute("products", pdao.getByCategoryId(categoryId));
		}
		return "Products";
	}
}
```

Khi bạn tương tác vào chương trình tương:

STT	Hành động	Phương thức	Mô tả
1	Chạy products.htm	showProducts(), GET	Hiển thị menu loại và danh sách hàng hóa
2	Nhấp [loại]	showProducts(categoryId), POST	Hiển thị menu loại và danh sách hàng hóa của loại được chọn

Bước 7: Chạy

http://localhost:8080/SpringMVCJdbc/products.htm