

Bài tập bắt buộc của Tuần **XX** GV sẽ công bố cuối giờ thực hành. Bài tập bắt buộc sẽ gửi với Subject và tập tin nén kèm theo:

LTWWW_JAVA_{NGAYTHANGNAM}_TUAN{XX}_HOTENSINHVIEN

(trong đó **XX** sẽ là 01, 02 ... 10).

VD: LTWWW_JAVA_20082024_TUAN01_HOTENSINHVIEN

Sinh viên nộp sai yêu cầu bài tập Tuần nào xem như không nộp bài tập Tuần đó.

BÀI TẬP TUẦN 3 MÔN LẬP TRÌNH WEB NÂNG CAO (JAVA)

1. *Chương 3: Jakarta Server Page - JSP*

Mục tiêu:

- Hiểu và áp dụng được các cú pháp căn bản của JSPs trong việc xây dựng ứng dụng Web.
- Hiểu và áp dụng được các đối tượng ngầm định (implicit objects).
- Hiểu và áp dụng được JavaBean vào ứng dụng JSP.
- Hiểu và áp dụng được Expression Language vào ứng dụng Web.
- JSP Standard Tag Library (JSTL)

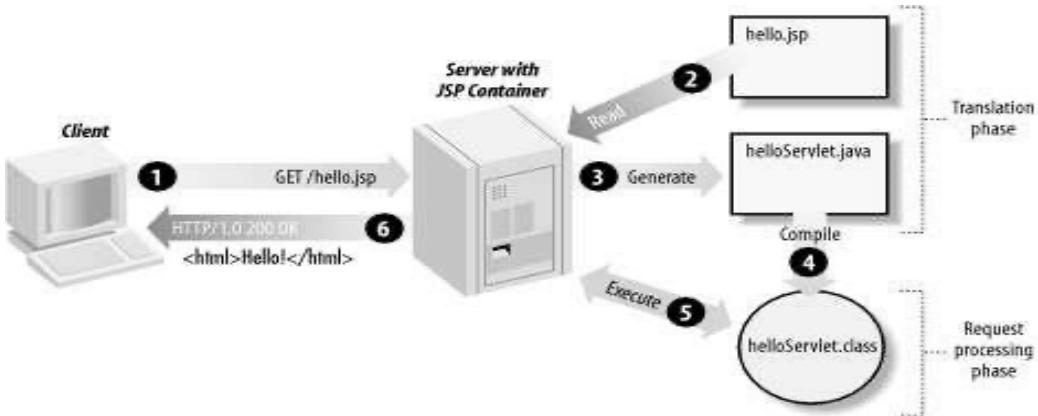
Yêu cầu:

- Tất cả các bài tập lưu trong thư mục: **T:\MaSV_HoTen\Tuan01**
- Tạo Project **MaSV_HoTen_Tuan01** trong thư mục **T:\MaSV_HoTen\Tuan01** trong Eclipse Java EE (Java Platform Enterprise Edition). Mỗi bài tập có thể lưu trong từng package riêng biệt.
- Cuối mỗi buổi thực hành, SV phải nén (.rar hoặc .zip) thư mục làm bài và nộp lại bài tập đã thực hiện trong buổi đó.

Dependencies:

- | | |
|--|--|
| 1. <!-- JSP and Servlets -->
jakarta.servlet-api: 6.0.0
jakarta.servlet.jsp-api: 3.1.1 | 4. <!-- MariaDB Client Java -->
mariadb-java-client: 3.4.1 |
| 2. <!-- JSTL -->
jakarta.servlet.jsp.jstl-api: 3.0.0
jakarta.servlet.jsp.jstl: 3.0.1 | 5. <!-- Hibernate validator -->
hibernate-validator: 8.0.8:Final |
| 3. <!-- Jakarta Persistence API -->
jakarta.persistence-api: 3.1.0
hibernate-core: 6.4.4.Final | |

JSP processing



1. Send an HTTP request to the Web server hello.jsp.
2. The web server recognizes that the HTTP request is for a JSP page and forwards it to a JSP engine.

This is done by using the URL or JSP page which ends with **.jsp** instead of **.html**.

3. The **JSP engine** loads the JSP page from disk and converts it into a servlet content. The code implements the corresponding dynamic behavior of the page.
4. The JSP engine compiles the servlet into an executable class and forwards the original request to a servlet engine.
5. The servlet engine loads the Servlet class and executes it. During execution, the servlet produces an output in HTML format. The output is further passed on to the web server by the servlet engine inside an HTTP response.
6. The web server forwards the HTTP response to your browser in terms of static HTML content. Finally, the web browser handles the dynamically-generated HTML page inside the HTTP response exactly as if it were a static page.

Standard Tag Library (JSTL) trong JSP phân loại

- **Core Tags:** Nhóm thẻ cơ bản
- **Formatting tags:** Nhóm thẻ định dạng
- **SQL tags:** Nhóm thẻ SQL
- **XML tags:** Nhóm thẻ XML
- **JSTL Functions:** Nhóm hàm JSTL

Core Tags (JSTL)

Cú pháp

```
<%@ taglib prefix="c" uri="jakarta.tags.core" %>
```

Thẻ	Miêu tả
<code><c:out></code>	Giống <code><%= ... %></code> , nhưng cho các Expression
<code><c:set></code>	Thiết lập kết quả của 1 Expression trong một 'scope'
<code><c:remove></code>	Gỡ bỏ một biến mục tiêu (từ một biến scope cụ thể, nếu đã xác định)
<code><c:catch></code>	Bắt bất kỳ Throwable
<code><c:if></code>	Thẻ điều kiện đơn giản.
<code><c:choose></code>	Thẻ điều kiện đơn giản mà thiết lập một context cho các hoạt động điều kiện loại trừ, được đánh dấu bởi <code><when></code> và <code><otherwise></code>
<code><c:when></code>	Thẻ phụ của <code><choose></code> mà điều kiện được ước lượng là true
<code><c:otherwise></code>	Thẻ phụ của <code><choose></code> mà theo sau thẻ <code><when></code> và chỉ chạy nếu tất cả điều kiện là 'false'
<code><c:import></code>	Dùng để import.
<code><c:forEach></code>	Thẻ lặp cơ bản.
<code><c:forTokens></code>	Lặp qua các token, được phân biệt bởi các dấu phân tách (delimiter) đã cung cấp
<code><c:param></code>	Thêm một parameter tới một URL của thẻ đang chứa 'import'
<code><c:redirect></code>	Redirect tới một URL mới
<code><c:url></code>	Tạo một URL với các tham số truy vấn tùy ý

Formatting Tags (JSTL)

Cú pháp

```
<%@ taglib prefix="fmt" uri="jakarta.tags fmt" %>
```

Thẻ	Miêu tả
<code><fmt:formatNumber></code>	Trả lại giá trị số với định dạng cụ thể
<code><fmt:parseNumber></code>	Parse biểu diễn chuỗi của một số, tiền tệ, phần trăm
<code><fmt:formatDate></code>	Định dạng một date/time bởi sử dụng Style và Pattern đã cho
<code><fmt:parseDate></code>	Parse biểu diễn chuỗi của một date/time
<code><fmt:bundle></code>	Gán một Resource Bundle để được sử dụng bởi phần thân thẻ
<code><fmt:setLocale></code>	Lưu giữ Locale đã cho trong biến cấu hình locale
<code><fmt:setBundle></code>	Gán một Resource Bundle và lưu giữ trong biến scope đã đặt tên hoặc biến cấu hình bundle
<code><fmt:timeZone></code>	Xác định timezone cho bất kỳ định dạng time nào hoặc parse các action được lặp trong phần thân của tag.
<code><fmt:setTimeZone></code>	Gán timezone đã cung cấp biến cấu hình time zone đó
<code><fmt:message></code>	Hiển thị một thông báo đa ngôn ngữ
<code><fmt:requestEncoding></code>	Thiết lập mã hóa ký tự cho request

SQL Tags (JSTL)

Cú pháp

```
<%@ taglib prefix="sql" uri="jakarta.tags.sql" %>
```

Thẻ	Miêu tả
<code><sql:setDataSource></code>	Tạo một DataSource kết nối vào hệ quản trị cơ sở dữ liệu.
<code><sql:query></code>	Thực thi SQL query được định nghĩa trong tag đóng/mở hoặc thông qua thuộc tính sql
<code><sql:update></code>	Thực thi SQL update được định nghĩa trong tag đóng/mở hoặc thông qua thuộc tính sql
<code><sql:param></code>	Thiết lập một parameter trong một lệnh SQL
<code><sql:dateParam></code>	Thiết lập một parameter trong một lệnh SQL với giá trị java.util.Date đã xác định
<code><sql:transaction></code>	Cung cấp các phần tử database action được lặp với một Connection đã chia sẻ, thiết lập để thực thi tất cả các lệnh

XML Tags (JSTL)

Cú pháp

```
<%@ taglib prefix="x" uri="jakarta.tags.xml" %>
```

Thẻ	Miêu tả
<code><x:out></code>	Giống <code><%= ... %></code> , nhưng dành cho các XPath Expression
<code><x:parse></code>	Sử dụng để parse XML data được xác định hoặc thông qua một thuộc tính hoặc trong phần thân thẻ
<code><x:set></code>	Thiết lập một biến với giá trị của một XPath expression
<code><x:if></code>	Ước lượng XPath Expression và nếu nó là true, thì xử lý phần thân thẻ. Nếu điều kiện là false, phần thân bị bỏ qua
<code><x:forEach></code>	Lặp qua các node trong một tài liệu XML
<code><x:choose></code>	Điều kiện đơn giản mà thiết lập một context cho hoạt động điều kiện loại trừ, được đánh dấu bởi <code><when></code> và <code><otherwise></code> trong JSTL
<code><x:when></code>	Thẻ phụ của <code><choose></code> .
<code><x:otherwise></code>	Thẻ phụ của <code><choose></code> .
<code><x:transform></code>	Áp dụng một phép biến đổi XSL trên một tài liệu XML
<code><x:param></code>	Sử dụng cùng với thẻ transform để thiết lập một parameter trong XSLT stylesheet

Functions Tags (JSTL)

Cú pháp

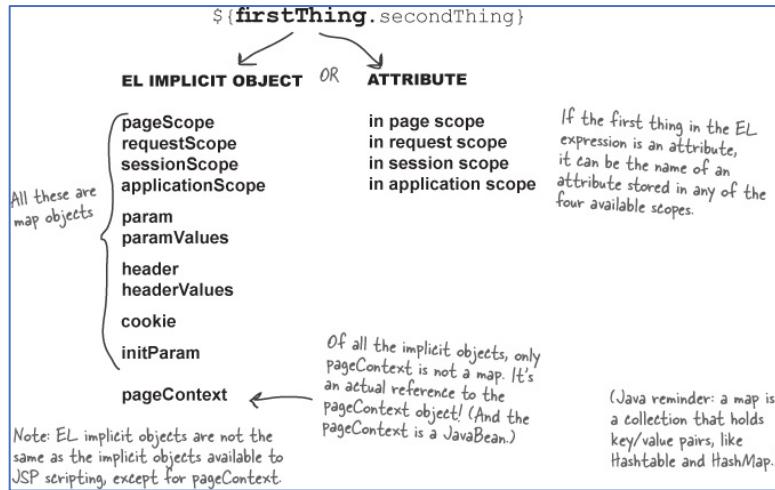
```
<%@ taglib prefix="fn" uri="jakarta.tags.functions" %>
```

Hàm	Miêu tả
Hàm fn:contains()	Kiểm tra nếu một chuỗi input chứa chuỗi phụ đã cho
Hàm fn:containsIgnoreCase()	Kiểm tra nếu một chuỗi input chứa chuỗi phụ đã cho trong trường hợp không phân biệt kiểu chữ
Hàm fn:endsWith()	Kiểm tra nếu một chuỗi input kết thúc với suffix đã cho
Hàm fn:escapeXml()	Các ký tự thoát mà có thể được phiên dịch như XML markup
Hàm fn:indexOf()	Trả về index bên trong một chuỗi về sự xuất hiện đầu tiên của chuỗi phụ
Hàm fn:join()	Kết hợp tất cả phần tử trong một mảng thành một chuỗi
Hàm fn:length()	Trả về số item trong một tập hợp, hoặc số ký tự trong một chuỗi
Hàm fn:replace()	Trả về một chuỗi là kết quả của việc thay thế một chuỗi input với một chuỗi đã cho
Hàm fn:split()	Chia một chuỗi thành một mảng các chuỗi phụ
Hàm fn:startsWith()	Kiểm tra nếu một chuỗi input bắt đầu với prefix đã cho
Hàm fn:substring()	Trả về một tập con của một chuỗi
Hàm fn:substringAfter()	Trả về một tập con của một chuỗi ở sau một chuỗi phụ đã cho
Hàm fn:substringBefore()	Trả về một tập con của một chuỗi ở trước một chuỗi phụ đã cho
Hàm fn:toLowerCase()	Biến đổi tất cả ký tự của một chuỗi thành chữ thường
Hàm fn:toUpperCase()	Biến đổi tất cả ký tự của một chuỗi thành chữ hoa
Hàm fn:trim()	Gỡ bỏ các khoảng trắng từ hai đầu của một chuỗi

EL expressions are ALWAYS within curly braces, and prefixed with the dollar sign

Cú pháp: \${expression}

VD: \${person.name}



Bài 1. JSP - Thao tác với Form

Các bài tập Java Servlets thực hiện bằng JSPs. Lưu ý dùng các actions khi cần thiết:

- jsp:forward, jsp:include
- jsp:useBean, jsp:setProperty, jsp:getProperty

Thực hiện form đăng ký khóa học cho sinh viên:

First name (max 30 characters a-z and A-Z)

Last name (max 30 characters a-z and A-Z)

Date of birth Day: Month: Year:

Email

Mobile number (10 digit number)

Gender Male Female

Address

City (max 30 characters a-z and A-Z)

Pin code (6 digit number)

State (max 30 characters a-z and A-Z)

Country India

Hobbies Drawing Singing Dancing Sketching Others

Qualification

Sl.No.	Examination	Board	Percentage	Year of Passing
1	Class X	<input type="text"/>	<input type="text"/>	<input type="text"/>
2	Class XII	<input type="text"/>	<input type="text"/>	<input type="text"/>
3	Graduation	<input type="text"/>	<input type="text"/>	<input type="text"/>
4	Masters	<input type="text"/>	<input type="text"/>	<input type="text"/>

(10 char max) (upto 2 decimal)

Course applies for BCA B.Com B.Sc B.A

B1. Tạo lớp **Student** bao gồm các thuộc tính mô tả trên form, các thuộc tính khai báo **private** kèm theo các phương thức **get/set** + **constructors**.

B2. Tạo form đăng ký

```
<form action="registration-form" name="formDangKy" method="GET">
```

B3. Tạo **Servlet** xử lý thao tác nhập dữ liệu của form:

```
@WebServlet("/registration-form")
public class RegistrationForm extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public RegistrationForm() {
        super();
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // TODO Auto-generated method stub
        response.getWriter().append("Served at: ").append(request.getContextPath());

        // Get data from Form
        String fname = request.getParameter("txtFName");
        String lname = request.getParameter("txtLName");
        String day = request.getParameter("day");
        String month = request.getParameter("month");
        String year = request.getParameter("year");
        String email = request.getParameter("txtEmail");
        String mobileNumber = request.getParameter("txtMobileNumber");
        String gender = request.getParameter("gender");
        String address = request.getParameter("txtAddress");
        String city = request.getParameter("txtCity");
        String pinCode = request.getParameter("txtPinCode");
        String state = request.getParameter("txtState");
        String country = request.getParameter("txtCountry");
        String hobbies = request.getParameter("chkHobbies");
        String course = request.getParameter("txtCourse");
        String birthdate = day + " " + month + " " + year;

        // Set data to Student
        Student sv = new Student();
        sv.setFirstName(fname);
        sv.setLastName(lname);
        sv.setEmail(email);
        sv.setGender(gender);
        sv.setBirthday(birthdate);

        // Set object student to request object
        request.setAttribute("student", sv);

        // Forward to result-form.jsp
        RequestDispatcher rd = request.getRequestDispatcher("result-form.jsp");
        rd.forward(request, response);
    }
}
```

B4. Tạo trang JSP hiển thị kết quả **result-form.jsp**

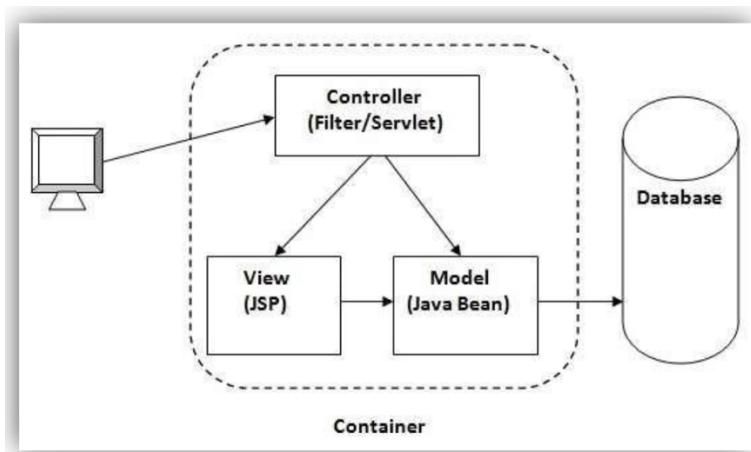
```

<%@page import="iuh.fit.se.entities.Student"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Result submit</title>
</head>
<body>
<%
    Student student = new Student();
    student = (Student)request.getAttribute("student");
    out.println(
        "First name:" + student.getFirstName()
        + "<br/> Last name: " + student.getLastName()
        + "<br/> Email : " + student.getEmail()
        + "<br/> Gender: " + student.getGender()
        + "<br/> Birthday: " + student.getBirthday()
    );
%
</body>
</html>

```

Bài 2. JSP - Model View Controller (MVC)

Thực hiện form đăng ký tài khoản với JSPs. Sau khi đăng ký thành công cập nhật danh sách tài khoản (không hiển thị password), Thực hiện theo mô hình MVC.



User Registration Form

First Name	Last Name	
Your Email		
Re-enter Email		
New Password		
Birthday		
Month	Day	Year
<input type="radio"/> Female	<input type="radio"/> Male	
Sign Up		

Bài 3. JSP - Xử lý đa ngôn ngữ

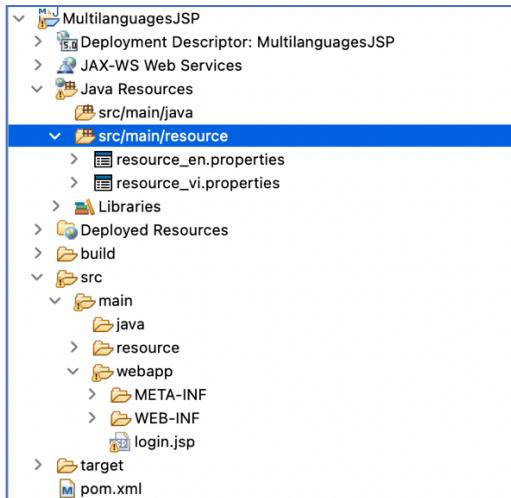
Thực hiện ứng dụng đa ngôn ngữ với JSP.

Chọn ngôn ngữ	<input checked="" type="radio"/> Tiếng Việt	<input type="radio"/> Tiếng Anh	Chọn
Tên đăng nhập	<input type="text"/>		
Mật khẩu	<input type="password"/>		
Đăng nhập			

Please select your language	<input type="radio"/> Vietnamese	<input checked="" type="radio"/> English	Choose
Username	<input type="text"/>		
Password	<input type="password"/>		
Login			

Tham khảo hướng dẫn

Project gồm các file resource cho từng ngôn ngữ, cần khai báo thư viện cho JSTL.



resource_vi.properties		resource_en.properties	
name	value	name	value
languageMessage	Chọn ngôn ngữ	languageMessage	Please select your language
vn	Tiếng Việt	vn	Vietnamese
en	Tiếng Anh	en	English
chooseButton	Chọn	chooseButton	Choose
userName	Tên đăng nhập	userName	Username
pass	Mật khẩu	pass	Password
login	Đăng nhập	login	Login

B2. Thực hiện đa ngôn ngữ trong file JSP. Khai báo các taglib

```
<%@ taglib uri="jakarta.tags.core" prefix="c" %>
<%@ taglib uri="jakarta.tags.fmt" prefix="fmt" %>
<fmt:setLocale value="${languageCode}" scope="session" />=> lấy mã ngôn ngữ
cần xử lý
<fmt:setBundle basename="resource" scope="session" />=> gán file resource liên
quan đến ngôn ngữ
```

```

<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@ taglib uri="jakarta.tags.core" prefix="c" %>
<%@ taglib uri="jakarta.tags.fmt" prefix="fmt" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Multi-language Demo</title>
</head>
<body>
<c:set var="languageCode" value="${param.radLanguageCode}"></c:set>
<c:if test="${not empty languageCode }">
    <fmt:setLocale value="${languageCode}" scope="session"/>
</c:if>
<fmt:setBundle basename="resource" scope="session"/>
<form action="login.jsp" method="POST">
    <fmt:message key="languageMessage"></fmt:message>
    <input type="radio" name="radLanguageCode" value="vi"
        <c:if test="${languageCode == 'vi'}> checked</c:if>
    /><fmt:message key="vn"></fmt:message>
    <input type="radio" name="radLanguageCode" value="en"
        <c:if test="${languageCode == 'en'}> checked</c:if>
    /><fmt:message key="en"></fmt:message>

    <input type="submit" name="submit" value=<fmt:message key="chooseButton"></fmt:message>"/>
<table border="0">
    <tr>
        <td><fmt:message key="userName"></fmt:message></td>
        <td><input type="text" name="txtUserName" value="" /></td>
    </tr>
    <tr>
        <td><fmt:message key="pass"></fmt:message></td>
        <td><input type="password" name="txtPassword" value="" /></td>
    </tr>
    <tr>
        <td colspan="2">
            <input type="submit" name="submit" value=<fmt:message key="login"></fmt:message>"/>
        </td>
    </tr>
</table>
</form>
</body>
</html>

```

Bài 4. JSP - Thao tác với JSP Session

Thực hiện website mua bán sản phẩm đơn giản dùng JSP Session.

Hiển thị danh sách sản phẩm và cho thêm sản phẩm vào giỏ hàng

[View Cart](#)

<p>Nokia Lumia</p>  <p>Price: 99000.0</p> <p><input type="button" value="1"/></p> <p><input type="button" value="Add To Cart"/></p>	<p>BlackBerry Passport</p>  <p>Price: 48000.0</p> <p><input type="button" value="1"/></p> <p><input type="button" value="Add To Cart"/></p>	<p>Sony Xperia Z5</p>  <p>Price: 52000.0</p> <p><input type="button" value="1"/></p> <p><input type="button" value="Add To Cart"/></p>
<p>HTC One M9</p>  <p>Price: 83000.0</p> <p><input type="button" value="1"/></p> <p><input type="button" value="Add To Cart"/></p>	<p>Samsung Galaxy Note 5</p>  <p>Price: 71000.0</p> <p><input type="button" value="1"/></p> <p><input type="button" value="Add To Cart"/></p>	<p>iPhone 7 jet-black Plus</p>  <p>Price: 120000.0</p> <p><input type="button" value="1"/></p> <p><input type="button" value="Add To Cart"/></p>

Khi giỏ hàng không có gì, hiển thị thông báo

Model Description	Quantity	Unit Price	Total
Cart is currently empty -			
			Subtotal: \$0.0

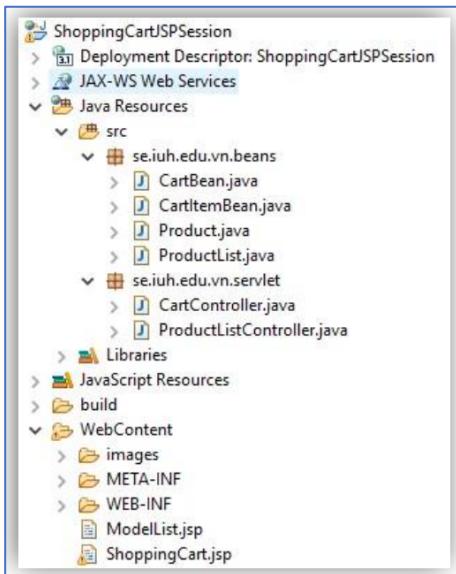
Khi giỏ hàng có sản phẩm, cho phép xóa, sửa các sản phẩm.

Model Description	Quantity	Unit Price	Total
PRO02 BlackBerry Passport	<input type="button" value="1"/> <input type="button" value="Update"/> <input type="button" value="Delete"/>	\$48000.0	\$48000.0
PRO06 iPhone 7 jet-black Plus	<input type="button" value="1"/> <input type="button" value="Update"/> <input type="button" value="Delete"/>	\$120000.0	\$120000.0
			Subtotal: \$168000.0

Tham khảo hướng dẫn

B1. Tạo Dynamic Web Project gồm các phần:

- Model:
 - Product (sản phẩm)
 - CartBean (giỏ hàng)
 - CartBeanItem(các sản phẩm trong giỏ hàng)
 - ProductList (danh sách các sản phẩm, lưu trong collection hoặc truy vấn từ CSDL)
- View:
 - ModelList (danh sách sản phẩm)
 - ShoppingCart (giỏ hàng)
- Controller:
 - CartController dùng để xử lý thao tác nghiệp vụ đối với giỏ hàng
 - ProductListController dùng để xử lý việc hiển thị danh sách sản phẩm.



B2. Tạo Model

```
package se.iuh.edu.vn.beans;
public class Product {
    private String id;
    private String model;
    private String description;
    private int quantity;
    private double price;
    private String imgURL;

    public String getImgURL() {..}

    public void setImgURL(String imgURL) {..}

    public String getId() {..}

    public void setId(String id) {..}

    public String getDescription() {..}

    public void setDescription(String description) {..}

    public int getQuantity() {..}

    public void setQuantity(int quantity) {..}

    public double getPrice() {..}

    public void setPrice(double price) {..}

    public String getModel() {..}

    public void setModel(String model) {..}

}
```

Lớp CartItemBean, lưu thông tin từng thành phần trong giỏ hàng

```
package se.iuh.edu.vn.beans;
public class CartItemBean {
    private String strPartNumber;
    private String strModelDescription;
    private double dblUnitCost;
    private int iQuantity;
    private double dblTotalCost;

    public String getPartNumber() {
        return strPartNumber;
    }

    public void setPartNumber(String strPartNumber) {
        this.strPartNumber = strPartNumber;
    }

    public String getModelDescription() {
        return strModelDescription;
    }

    public void setModelDescription(String strModelDescription) {
        this.strModelDescription = strModelDescription;
    }

    public double getUnitCost() {
        return dblUnitCost;
    }

    public void setUnitCost(double dblUnitCost) {
        this.dblUnitCost = dblUnitCost;
    }

    public int getQuantity() {
        return iQuantity;
    }

    public void setQuantity(int quantity) {
        iQuantity = quantity;
    }

    public double getTotalCost() {
        return dblTotalCost;
    }

    public void setTotalCost(double dblTotalCost) {
        this.dblTotalCost = dblTotalCost;
    }
}
```

B3. Xử lý các thao tác liên quan đến giỏ hàng: thêm sản phẩm vào giỏ hàng, xóa sản phẩm, cập nhật số lượng: CartBean.java

```
package se.iuh.edu.vn.beans;

import java.util.ArrayList;

public class CartBean {
    private ArrayList<CartItemBean> alCartItems = new ArrayList<CartItemBean>();
    private double dblOrderTotal;

    public int getLineItemCount() { }

    public void deleteCartItem(String strItemIndex) { }

    public void updateCartItem(String strItemIndex, String strQuantity) { }

    public void addCartItem(String strModelNo, String strDescription, String strUnitCost, String strQuantity) { }

    public void addCartItem(CartItemBean cartItem) { }

    public CartItemBean getCartItem(int iItemIndex) { }

    public ArrayList<CartItemBean> getCartItems() { }

    public void setCartItems(ArrayList<CartItemBean> alCartItems) { }

    public double getOrderTotal() { }

    public void setOrderTotal(double dblOrderTotal) { }

    protected void calculateOrderTotal() { }

}
```

Các hàm xử lý trong CartBean:

```
public int getLineItemCount() {
    return alCartItems.size();
}

public void deleteCartItem(String strItemIndex) {
    int iItemIndex = 0;
    try {
        iItemIndex = Integer.parseInt(strItemIndex);
        alCartItems.remove(iItemIndex - 1);
        calculateOrderTotal();
    } catch (NumberFormatException nfe) {
        System.out.println("Error while deleting cart item: " + nfe.getMessage());
        nfe.printStackTrace();
    }
}
```

```

public void updateCartItem(String strItemIndex, String strQuantity) {
    double dblTotalCost = 0.0;
    double dblUnitCost = 0.0;
    int iQuantity = 0;
    int iItemIndex = 0;
    CartItemBean cartItem = null;
    try {
        iItemIndex = Integer.parseInt(strItemIndex);
        iQuantity = Integer.parseInt(strQuantity);
        if (iQuantity > 0) {
            cartItem = (CartItemBean) alCartItems.get(iItemIndex - 1);
            dblUnitCost = cartItem.getUnitCost();
            dblTotalCost = dblUnitCost * iQuantity;
            cartItem.setQuantity(iQuantity);
            cartItem.setTotalCost(dblTotalCost);
            calculateOrderTotal();
        }
    } catch (NumberFormatException nfe) {
        System.out.println("Error while updating cart: " + nfe.getMessage());
        nfe.printStackTrace();
    }
}

```

```

public CartItemBean getCartItem(int iItemIndex) {
    CartItemBean cartItem = null;
    if (alCartItems.size() > iItemIndex) {
        cartItem = (CartItemBean) alCartItems.get(iItemIndex);
    }
    return cartItem;
}

public ArrayList<CartItemBean> getCartItems() {
    return alCartItems;
}

public void setCartItems(ArrayList<CartItemBean> alCartItems) {
    this.alCartItems = alCartItems;
}

public double getOrderTotal() {
    return dblOrderTotal;
}

public void setOrderTotal(double dblOrderTotal) {
    this.dblOrderTotal = dblOrderTotal;
}

protected void calculateOrderTotal() {
    double dblTotal = 0;
    for (int counter = 0; counter < alCartItems.size(); counter++) {
        CartItemBean cartItem = (CartItemBean) alCartItems.get(counter);
        dblTotal += cartItem.getTotalCost();
    }
    setOrderTotal(dblTotal);
}

```

```

public void addCartItem(String strModelNo, String strDescription, String strUnitCost, String strQuantity)
{
    double dblTotalCost = 0.0;
    double dblUnitCost = 0.0;
    int iQuantity = 0;
    CartItemBean cartItem = new CartItemBean();
    try {
        dblUnitCost = Double.parseDouble(strUnitCost);
        iQuantity = Integer.parseInt(strQuantity);
        if (iQuantity > 0) {
            dblTotalCost = dblUnitCost * iQuantity;
            cartItem.setPartNumber(strModelNo);
            cartItem.setModelDescription(strDescription);
            cartItem.setUnitCost(dblUnitCost);
            cartItem.setQuantity(iQuantity);
            cartItem.setTotalCost(dblTotalCost);
            alCartItems.add(cartItem);
            calculateOrderTotal();
        }
    } catch (NumberFormatException nfe) {
        System.out.println("Error while parsing from String to primitive types: " + nfe.getMessage());
        nfe.printStackTrace();
    }
}

public void addCartItem(CartItemBean cartItem) {
    alCartItems.add(cartItem);
}

```

B4. Tạo danh sách sản phẩm (dùng Collection, có thể lấy từ CSDL cũ thẻ khác)

```

package se.iuh.edu.vn.beans;

import java.util.ArrayList;
import java.util.List;

public class ProductList {

    private static final List<Product> ds = new ArrayList<Product>();
    static {
        initData();
    }

    public static List<Product> queryProducts() {
        return ds;
    }

    private static void initData() {
        Product sp = new Product();
        sp.setId("PRO01");
        sp.setDescription("");
        sp.setPrice(99000);
        ds.add(sp);

        sp = new Product();
        sp.setId("PRO02");
        sp.setDescription("");
        sp.setPrice(48000);
        ds.add(sp);

        sp = new Product();
        sp.setId("PRO03");
        sp.setDescription("");
        sp.setPrice(52000);
        ds.add(sp);

        sp = new Product();
        sp.setId("PRO04");
        sp.setDescription("");
        sp.setPrice(83000);
        ds.add(sp);

        sp = new Product();
        sp.setId("PRO05");
        sp.setDescription("");
        sp.setPrice(71000);
        ds.add(sp);
    }
}

```

B5. Tạo Servlet nhận yêu cầu xử lý request/response của Client: CartController.java

```
public class CartController extends HttpServlet {  
    private static final long serialVersionUID = 1L;  
  
    public void doPost(HttpServletRequest request, HttpServletResponse response) {}  
  
    protected void deleteCart(HttpServletRequest request) {}  
    protected void updateCart(HttpServletRequest request) {}  
    protected void addToCart(HttpServletRequest request) {}  
}
```

Các hàm xử lý trong CartController.java

```
public void doPost(HttpServletRequest request, HttpServletResponse response)  
        throws ServletException, IOException {  
  
    String strAction = request.getParameter("action");  
  
    if (strAction != null && !strAction.equals("")) {  
        if (strAction.equals("add")) {  
            addToCart(request);  
        } else if (strAction.equals("Update")) {  
            updateCart(request);  
        } else if (strAction.equals("Delete")) {  
            deleteCart(request);  
        }  
    }  
    response.sendRedirect("/ShoppingCartJSPSession/ShoppingCart.jsp");  
}  
  
protected void deleteCart(HttpServletRequest request) {  
    HttpSession session = request.getSession();  
    String strItemIndex = request.getParameter("itemIndex");  
    CartBean cartBean = null;  
  
    Object objCartBean = session.getAttribute("cart");  
    if (objCartBean != null) {  
        cartBean = (CartBean) objCartBean;  
    } else {  
        cartBean = new CartBean();  
    }  
    cartBean.deleteCartItem(strItemIndex);  
}
```

```
protected void updateCart(HttpServletRequest request) {
    HttpSession session = request.getSession();
    String strQuantity = request.getParameter("quantity");
    String strItemIndex = request.getParameter("itemIndex");

    CartBean cartBean = null;

    Object objCartBean = session.getAttribute("cart");
    if (objCartBean != null) {
        cartBean = (CartBean) objCartBean;
    } else {
        cartBean = new CartBean();
    }
    cartBean.updateCartItem(strItemIndex, strQuantity);
}

protected void addToCart(HttpServletRequest request) {
    HttpSession session = request.getSession();
    String strModelNo = request.getParameter("modelNo");
    String strDescription = request.getParameter("description");
    String strPrice = request.getParameter("price");
    String strQuantity = request.getParameter("quantity");

    CartBean cartBean = null;

    Object objCartBean = session.getAttribute("cart");

    if (objCartBean != null) {
        cartBean = (CartBean) objCartBean;
    } else {
        cartBean = new CartBean();
        session.setAttribute("cart", cartBean);
    }
    cartBean.addCartItem(strModelNo, strDescription, strPrice, strQuantity);
}
```

B5. Tạo Servlet hiển thị danh sách: ProductListController.java

```
public class ProductListController extends HttpServlet {
    private static final long serialVersionUID = 1L;

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        List<Product> list = ProductList.queryProducts();
        request.setAttribute("ds", list);
        RequestDispatcher dispatcher = getServletContext().getRequestDispatcher("/ModelList.jsp");
        dispatcher.forward(request, response);
    }

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        doGet(request, response);
    }
}
```

B6. Tạo giao diện hiển thị danh sách và cho phép thêm sản phẩm vào giỏ hàng (View): ModelList.jsp

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Product List</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<style>
.a {
    width: 160px; height: 200px; border: 1px solid black;
    padding: 5px; margin: 10px;
    float: left; text-align: center;
}

.hinh {
    width: 80px; height: 100px;
}
</style>
</head>

<body>
    <p><a href="/ShoppingCartJSPSession/ShoppingCart.jsp">View Cart</a></p>

    <c:forEach items="${ds}" var="sp">
        <div class="a">
            <form name="modelDetail" method="POST"
                  action="/ShoppingCartJSPSession/CartController">
                ${sp.model} <br />
                <img src='/ShoppingCartJSPSession${sp.imgURL}' class='hinh'> <br />
                Price: ${sp.price} <br />
                <input type="text" size="2" value="1" name="quantity"><br>
                <input type="hidden" name="modelNo" value="${sp.id}">
                <input type="hidden" name="price" value="${sp.price}">
                <input type="hidden" name="description" value="${sp.model}">
                <input type="hidden" name="action" value="add">
                <input type="submit" name="addToCart" value="Add To Cart">
            </form>
        </div>
    </c:forEach>
</body>
</html>
```

B7: Tạo giao diện hỗ trợ xử lý trong giỏ hàng: JSP ShoppingCart.jsp

```
<jsp:useBean id="cart" scope="session" class="se.iuh.edu.vn.beans.CartBean" />

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head><title>Shopping Cart</title>
</head>
<body>
    <p><a href="/ShoppingCartJSPSession/DSSP">Product List</a></p>
    <table width="100%" border="1">
        <tr bgcolor="#CCCCCC">
            <td>Model Description</td> <td>Quantity</td>
            <td>Unit Price</td> <td>Total</td>
        </tr>

        <c:if test="${cart.lineItemCount==0}">
            <tr>
                <td colspan="4">Cart is currently empty -<br />
            </tr>
        </c:if>
        <c:forEach var="cartItem" items="${cart.cartItems}" varStatus="counter">
            <form name="item" method="POST" action="/ShoppingCartJSPSession/cartController">
                <tr>
                    <td>
                        <c:out value="${cartItem.partNumber}" /></b><br />
                        <c:out value="${cartItem.modelDescription}" /></td>
                    <td>
                        <input type='hidden' name='itemIndex' value='<c:out
value="${counter.count}" />'>
                        <input type='text' name="quantity" value='<c:out
value="${cartItem.quantity}" />' size='2'>
                        <input type="submit" name="action" value="update">
                        <input type="submit" name="action" value="Delete">
                    </td>
                    <td><c:out value="${cartItem.unitCost}" /></td>
                    <td><c:out value="${cartItem.totalCost}" /></td>
                </tr>
            </form>
        </c:forEach>
        <tr>
            <td colspan="2"></td>
            <td></td>
            <td>Subtotal: <c:out value="${cart.orderTotal}" /></td>
        </tr>
    </table>
</body>
</html>
```