

1.Servlets Form Handling:

User opens studentForm.html

1. Fills the registration form
2. On submit, form data is sent to /register
3. RegisterServlet displays all the submitted data

2.Servlet Request Dispatcher Use Case:

1. User submits name and marks using a form.
2. First servlet receives the data and forwards it to a second servlet.
3. Second servlet processes the marks and sends back the grade.
4. Final output is shown to the user.

Files Needed:

- marksForm.html
- InputServlet.java
- GradeServlet.java

3.Servlet Hidden Form Fields Use Case:

1. User selects a product from a list.
2. On the next form, user enters quantity.
3. Hidden fields carry product name and price.
4. Final servlet calculates total and displays receipt.

Project Files:

File Name	Purpose
product.html	Product selection form
ProductServlet.java	Generates quantity form with hidden product data
OrderServlet.java	Final bill generation and display

4.Servlet Session Management:

1. User logs in with **account number and password**.
2. If credentials are valid, session stores:
 - Account holder's name
 - Account balance
3. Redirect to balance page
4. User can:
 - View balance (if session active)
 - Logout (which invalidates session)

Files Required:

File Name	Purpose
<code>login.html</code>	Login form
<code>LoginServlet.java</code>	Validates credentials, creates session
<code>BalanceServlet.java</code>	Displays balance if session exists
<code>LogoutServlet.java</code>	Invalidates session

5. Servlet Cookie:

Build a Servlet-based application where a user selects **items to add to a cart**, and the selected item names are stored as **cookies**.

When the user visits the "View Cart" page, the application should read all stored cookies and display the list of selected items.

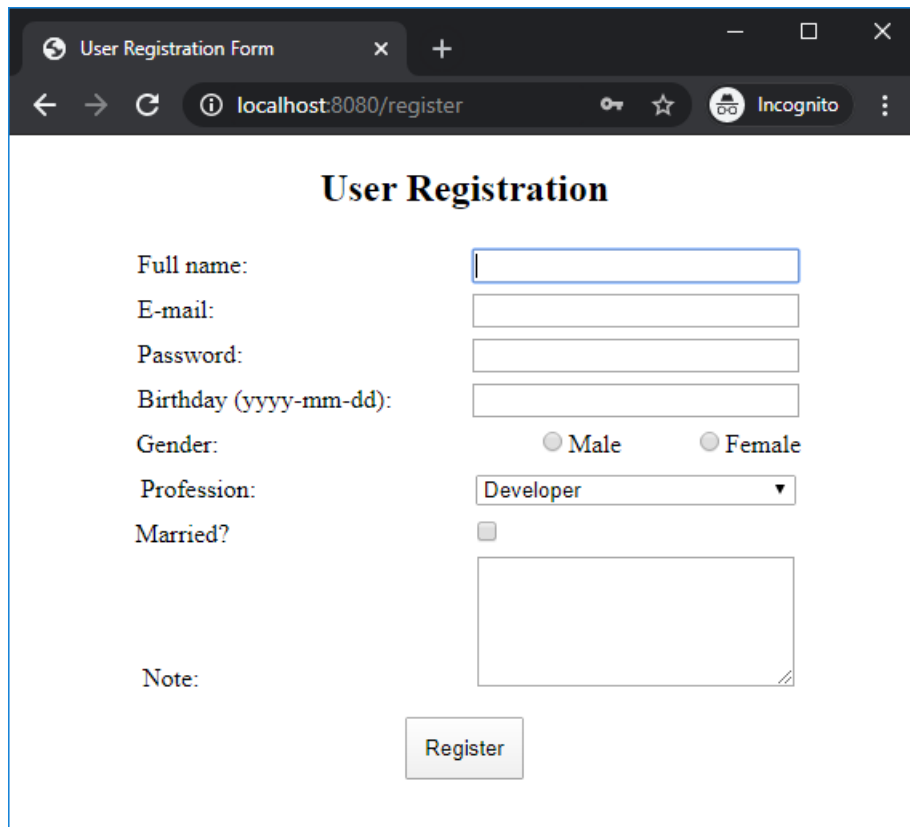
6. SERVLET + JDBC (MYSQL) + HTML:

Connect a Servlet to a **MySQL database**

- Use **JDBC** to execute a **SELECT** query
- Display Vendor records in an HTML table

=====

1. JSP LOGIN FORM::

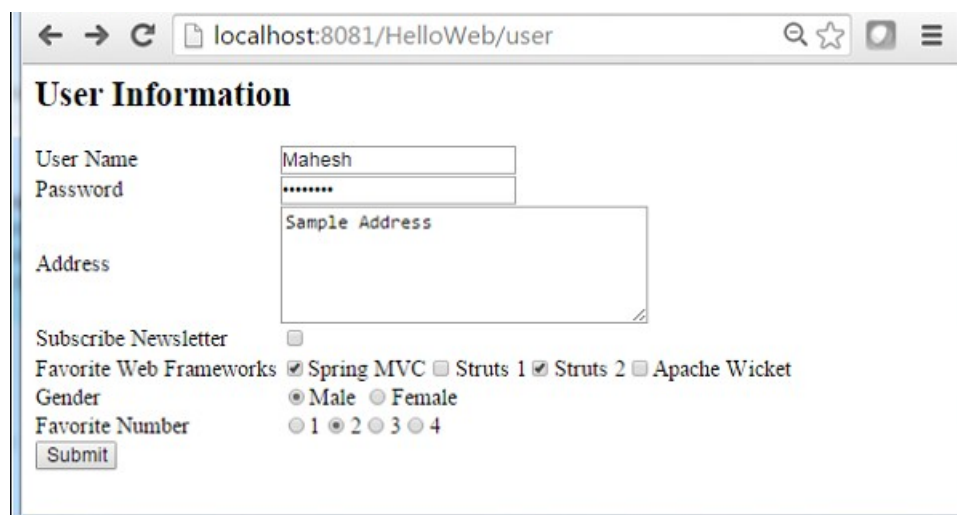


The screenshot shows a web browser window with the title 'User Registration Form'. The address bar displays 'localhost:8080/register'. The page content is titled 'User Registration' and contains the following form fields:

- Full name:
- E-mail:
- Password:
- Birthday (yyyy-mm-dd):
- Gender: ☐ Male ☐ Female
- Profession:
- Married? ☐
- Note:

A 'Register' button is located at the bottom of the form.

2. Expression Language (EL FORM) Design::



The screenshot shows a web browser window with the title 'User Information'. The address bar displays 'localhost:8081/HelloWeb/user'. The page content is titled 'User Information' and contains the following form fields:

- User Name:
- Password:
- Address:
- Subscribe Newsletter: ☐
- Favorite Web Frameworks: ☒ Spring MVC ☐ Struts 1 ☒ Struts 2 ☐ Apache Wicket
- Gender: ☒ Male ☐ Female
- Favorite Number: ☐ 1 ☒ 2 ☐ 3 ☐ 4

A 'Submit' button is located at the bottom of the form.

3. Design a JSP-based application with the following:

1. A form (`form.jsp`) that asks for **student name** and **marks**.
2. If $\text{marks} \geq 40$, forward the request to `pass.jsp`.
3. If $\text{marks} < 40$, forward to `fail.jsp`.
4. Both `pass.jsp` and `fail.jsp` should include a common footer using `<jsp:include>` from `footer.jsp`.

4. Create a JSP application that collects **product details** (Product ID, Name, Price, Quantity) from the user.

Store these values in a **JavaBean** using `<jsp:useBean>` and `<jsp:setProperty>`.

Display the product details in another JSP using `<jsp:getProperty>`.

5. Create a JSP application that takes **bank account details** (Account No, Name, Balance), stores them in a **JavaBean**, and displays the data using **Expression Language (EL)**.

Also display account type using session scope, and show whether the balance is **sufficient** using conditional expressions in EL.

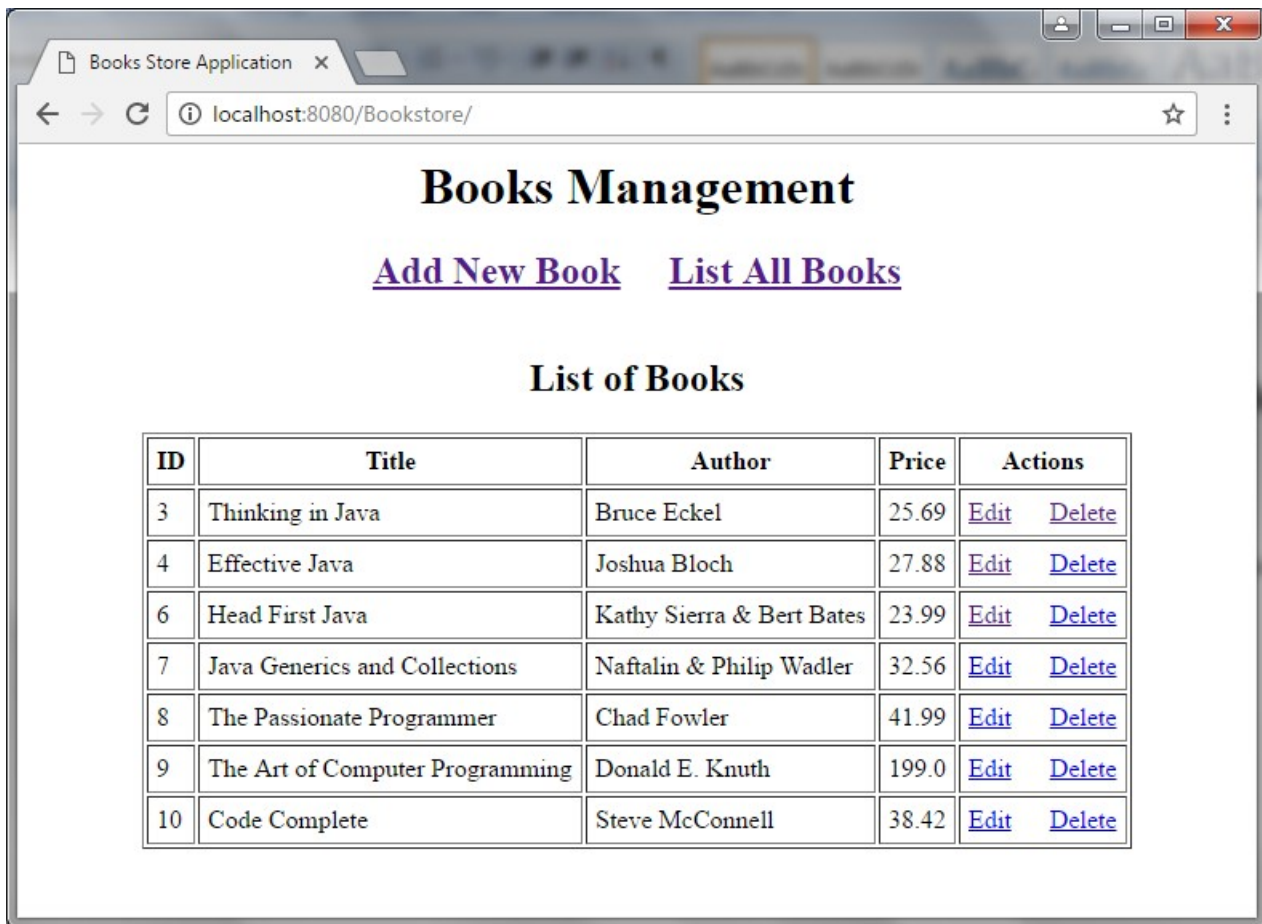
6. **MVC APPLICATION (MODEL(STUDENT) + VIEW(JSP) +CONTROLLER (SERVLET) + MYSQL CRUD)**

Books Management

[Add New Book](#) [List All Books](#)

Add New Book

Title:	<input type="text"/>
Author:	<input type="text"/>
Price:	<input type="text"/>
<input type="button" value="Save"/>	



HIBERNATE:

1. -----
Hibernate Item CRUD

1. Add
 2. View All
 3. Update
 4. Delete
 5. Get by ID
 0. Exit
- Choose option: _

Choose option: 1
Enter name: Pen
Enter price: 15.5
Item added.