



BÀI TẬP 'ƠNG TRÌNH LẬP TRÌNH VIÊN CÔNG NGHỆ JAVA

ĐODULE 3: LẬP TRÌNH WEB VỚI JAVA



BÀI 1: Tổng quan J2EE



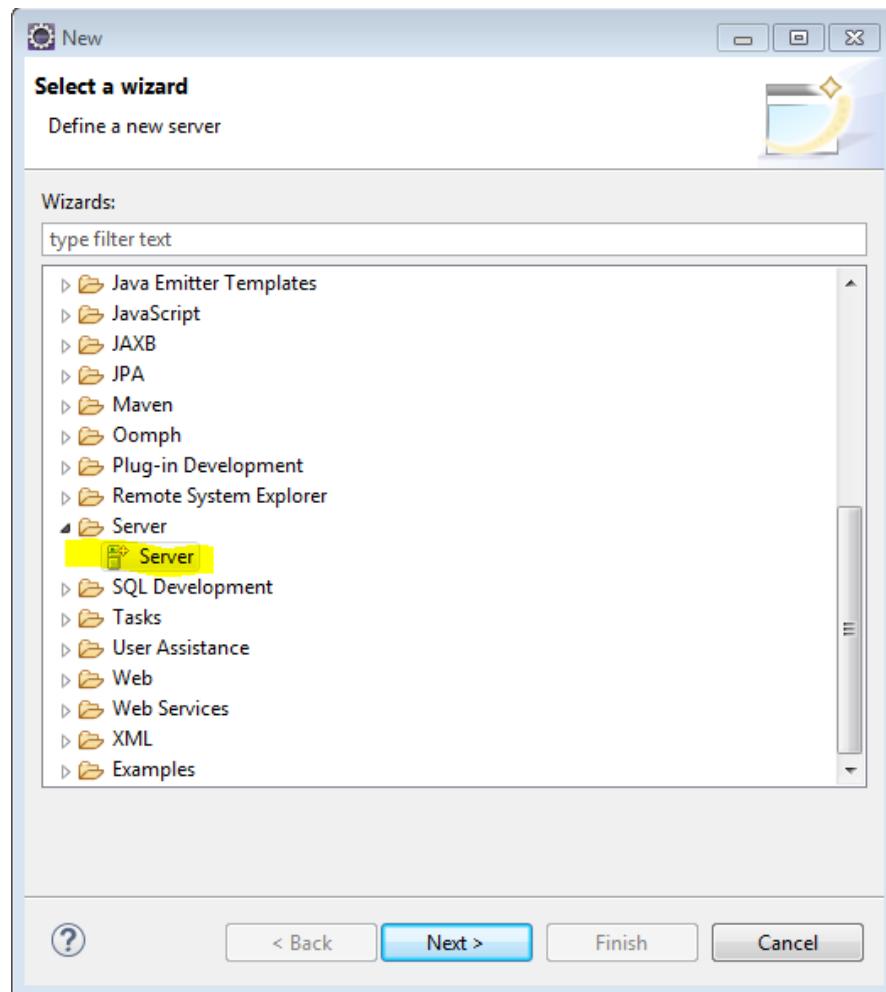
- Cài đặt và làm quen với môi trường phát triển ứng dụng web trên nền tảng công nghệ JEE
- Phát triển ứng dụng web tĩnh đơn giản sử dụng công nghệ html, javascript và css

1.1. Cài đặt môi trường Application Server

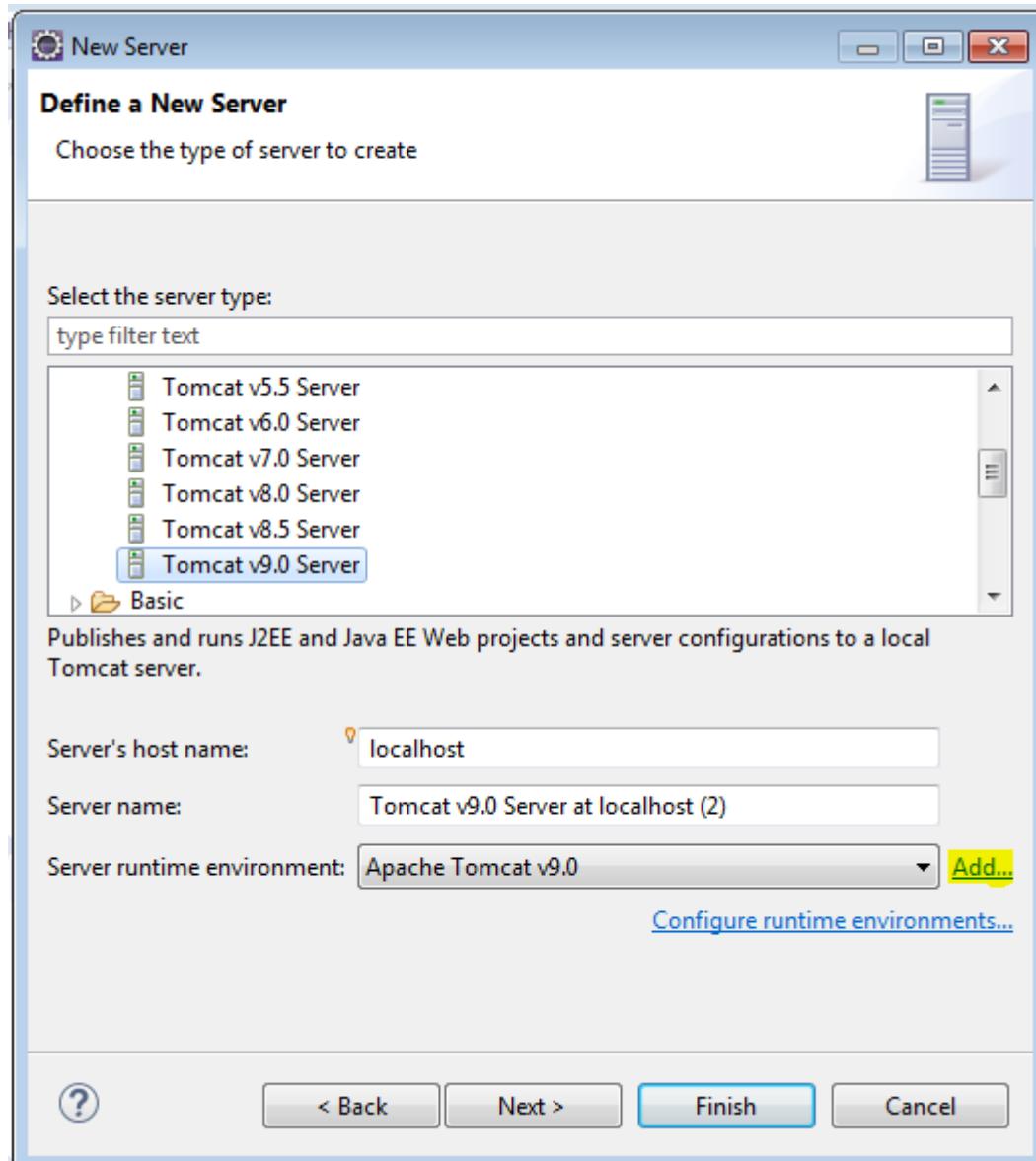
- **Yêu cầu:** Cài đặt Apache Tomcat 9 trong môi
- **Tóm tắt yêu cầu:**
 - Phải cài đặt trước: Java SDK, các trình duyệt (có thể là Firefox, IE, Chrome,..)
- **Hướng dẫn:**
 - Download Apache Tomcat phiên bản 9 tại địa chỉ <http://tomcat.apache.org/download-90.cgi> hoặc tìm kiếm download trên google
 - Giải nén apache-tomcat-9.0 và lưu trữ vào thư mục trên ổ đĩa

1.2. Cài đặt môi trường Application Server (Eclipse)

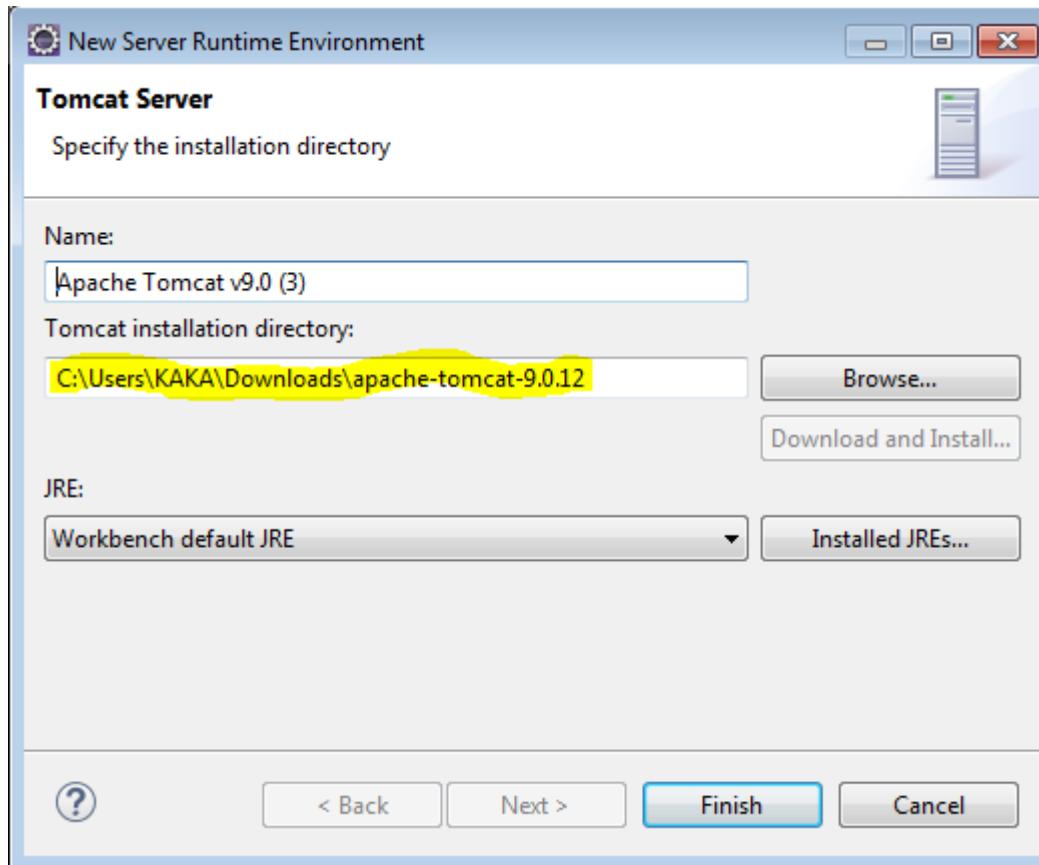
- **Yêu cầu:** Cài đặt Apache Tomcat trên môi trường soạn thảo Eclipse IDE
- **Hướng dẫn:**
 - Download Eclipse mới nhất download từ trang chủ <https://www.eclipse.org/downloads/packages/> chọn package Eclipse IDE for Java EE Developers
 - Giải nén tập tin eclipse-jee-2018-09.zip và lưu trữ vào thư mục
 - Mở Eclipse IDE
 - Tạo mới Server



- Chọn next qua bước tiếp theo chọn apache tomcat 9.0



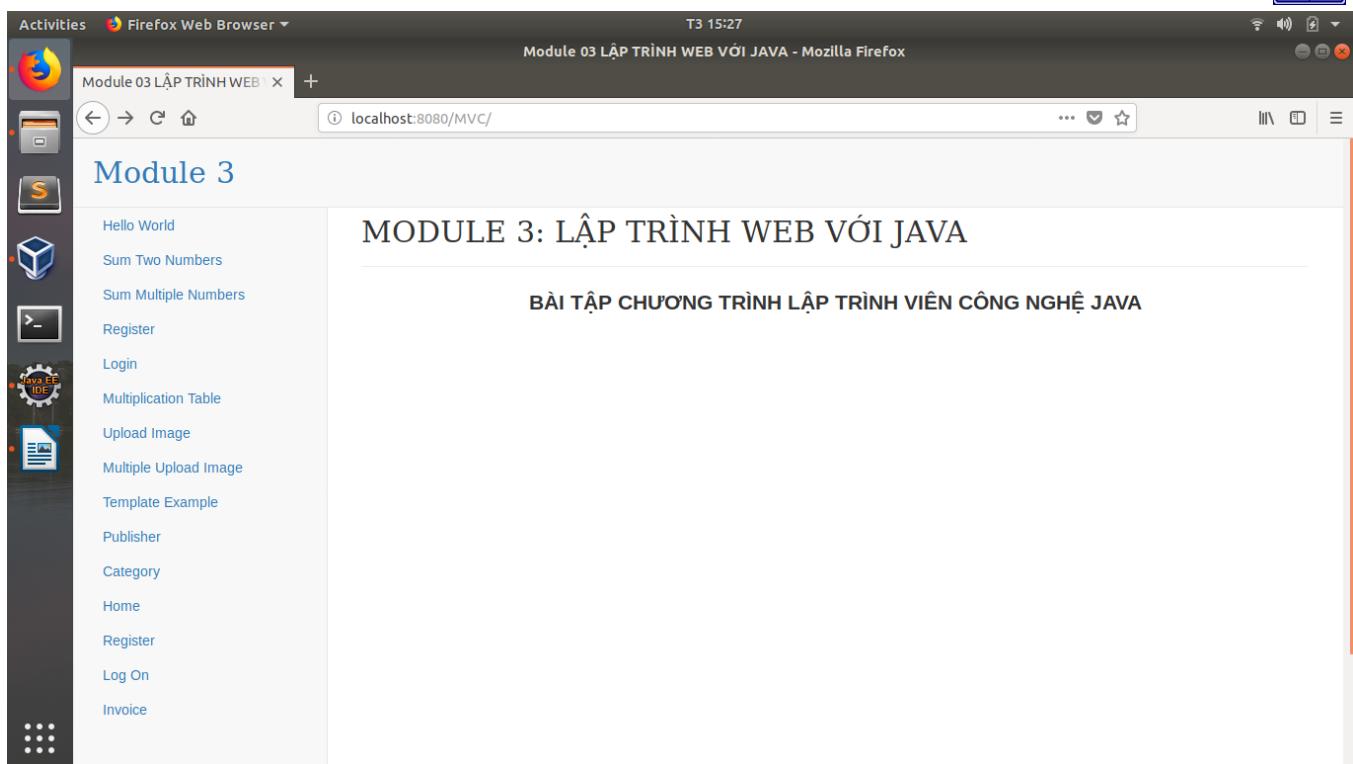
- Bước tiếp theo chọn Add và trả lời thư mục chứa Apache Tomcat



- Chọn finish và kết thúc quá trình cấu hình server

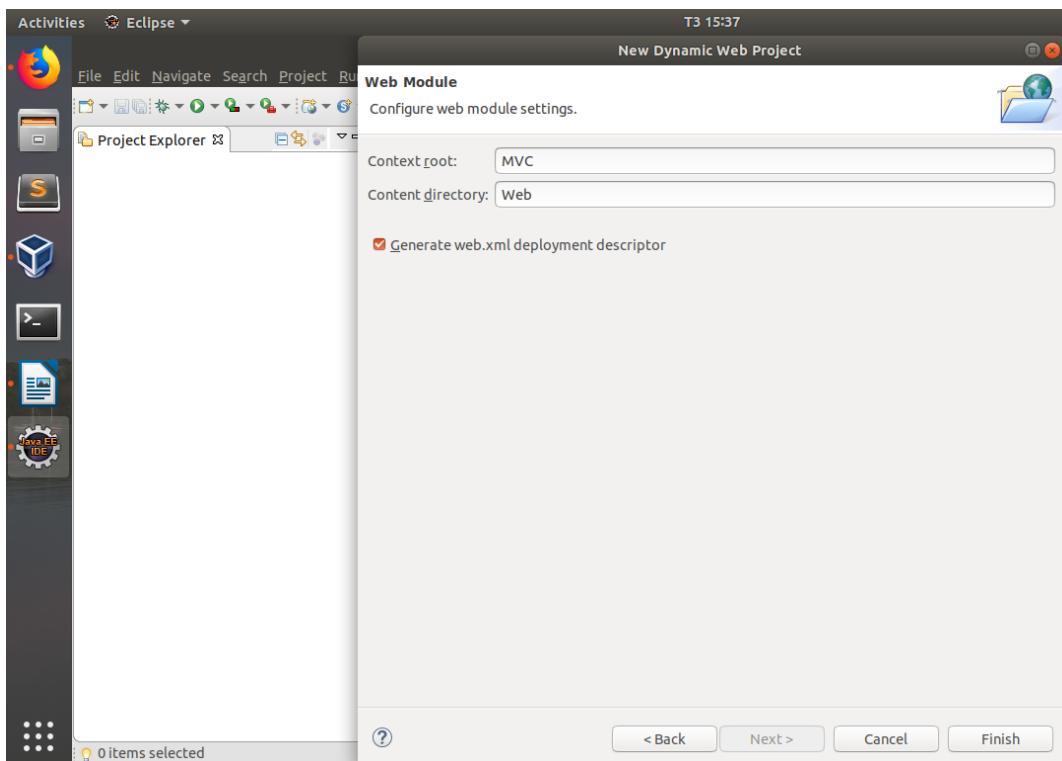
1.3. Phát triển ứng dụng web tĩnh

- **Yêu cầu:** Xây dựng ứng dụng web tĩnh trong Apache Tomcat
- **Tóm tắt yêu cầu:**
 - Thiết kế giao diện

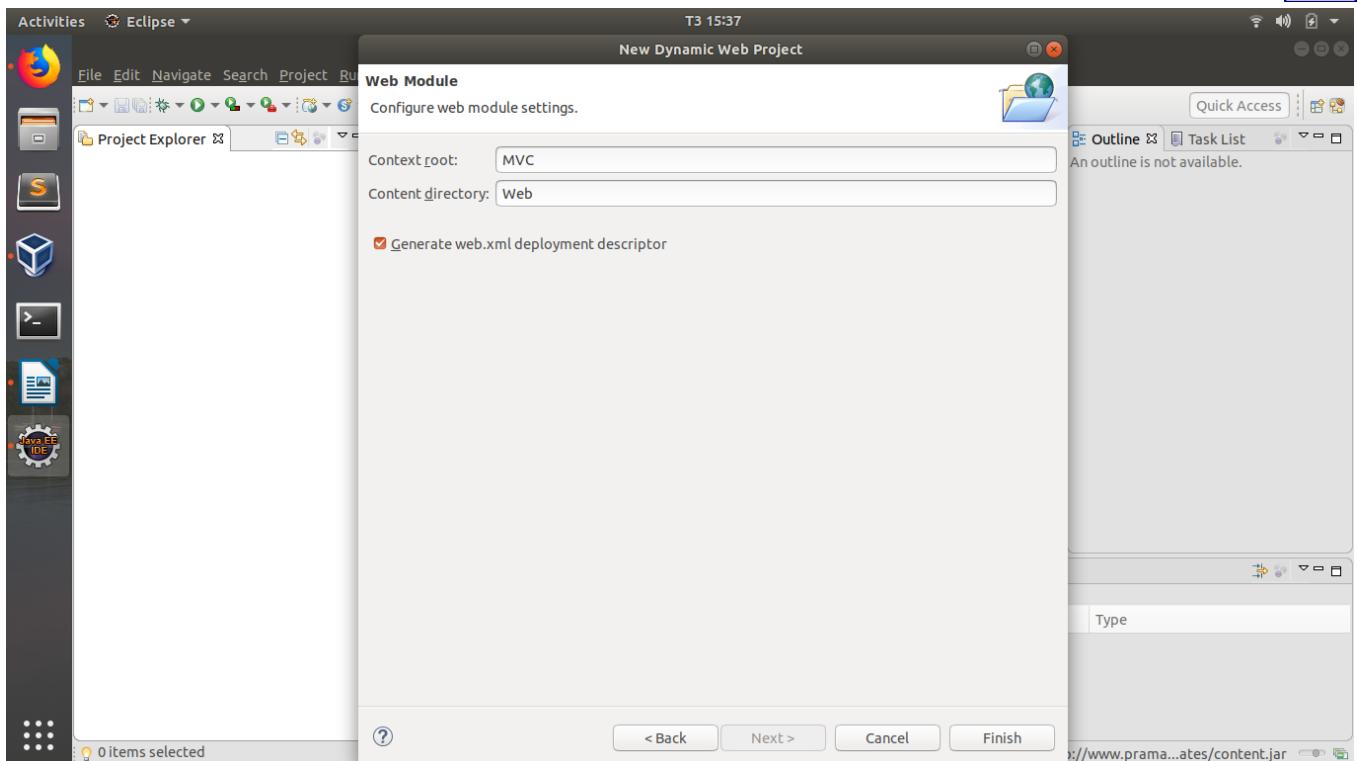


- **Hướng dẫn:**

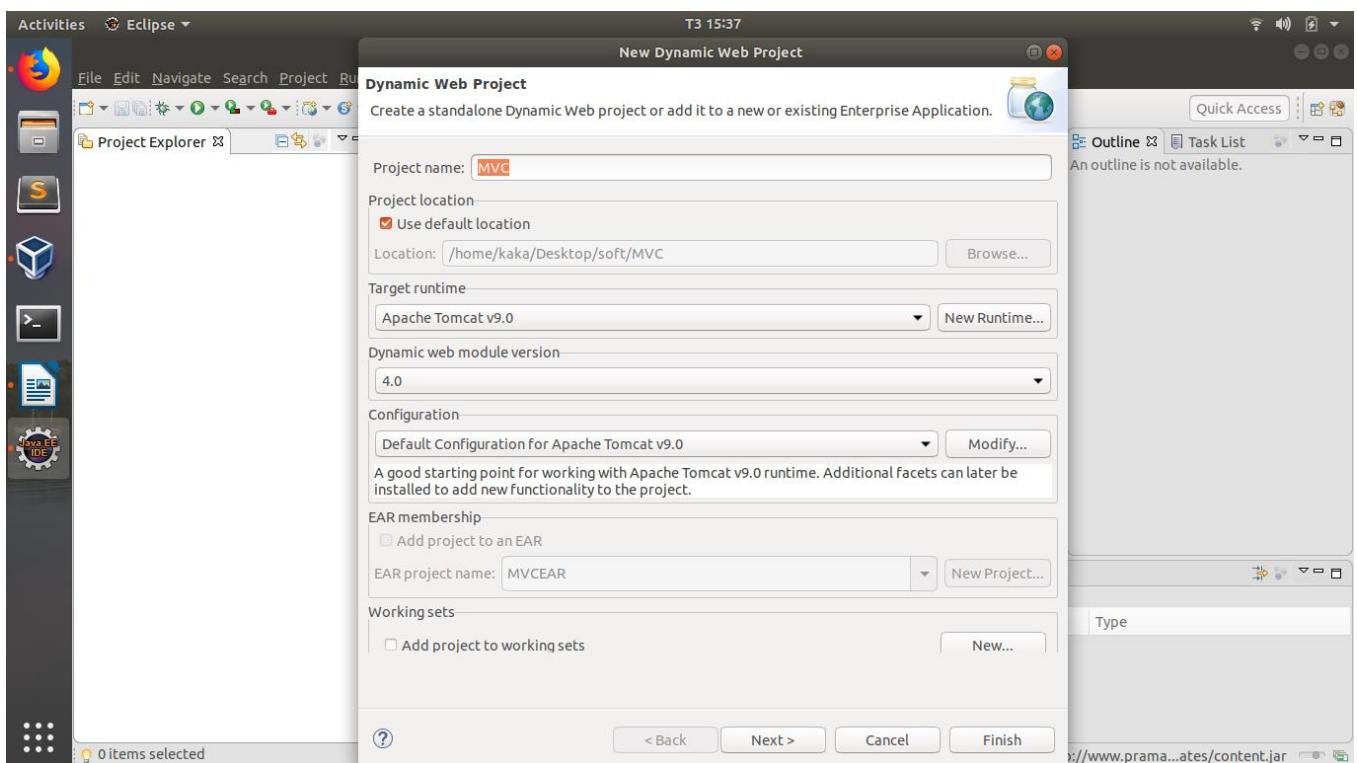
- Tạo mới project đặt tên là MVC



- Chọn Context root: MVC và thư mục Content directory là web



- Chọn Target Runtime trả về thư mục Tomcat 9.0



- Copy thư mục css vào trong thư mục Content Directory Web và tạo file index.html như hình sau

The screenshot shows the Eclipse IDE interface with the following components:

- Activities**: Shows the current activities: Eclipse.
- File Bar**: Contains File, Edit, Source, Navigate, Search, Project, Run, Window, Help.
- Toolbar**: Standard Eclipse toolbar icons.
- Project Explorer**: Shows the project structure:
 - MVC
 - Deployment Descriptor: MVC
 - JAX-WS Web Services
 - Java Resources
 - JavaScript Resources
 - build
 - Web
 - css
 - css.css
 - META-INF
 - WEB-INF
 - index.html
 - Servers
- Code Editor**: Displays the content of index.html:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="UTF-8">
5 <title>Module 03 LẬP TRÌNH WEB VỚI JAVA</title>
6 <link rel="stylesheet" type="text/css" href="/MVC/css/css.css">
7 </head>
8 <body>
9   <div class="title">
10    <div class="container">
11      <div class="brand"><a href="/MVC">Module 3</a></div>
12    </div>
13  </div>
14  <div class="sidebar">
15    <ul>
16      <li><a href="helloworld.html">Hello World</a></li>
17      <li><a href="sumtwonumbers.html">Sum Two Numbers</a></li>
18      <li><a href="summultinumbers.html">Sum Multiple Numbers</a></li>
19      <li><a href="register.jsp">Register</a></li>
20      <li><a href="login.jsp">Login</a></li>
21      <li><a href="multiplicationtable.jsp">Multiplication Table</a></li>
22      <li><a href="upload.html">Upload Image</a></li>
23      <li><a href="upload/multi.html">Multiple Upload Image</a></li>
24      <li><a href="template.jsp">Template Example</a></li>
25      <li><a href="admin/publisher.html">Publisher</a></li>
26      <li><a href="admin/category.html">Category</a></li>
27      <li><a href="home.html">Home</a></li>
28      <li><a href="auth/register.html">Register</a></li>
29      <li><a href="auth/logon.html">Log On</a></li>
30      <li><a href="admin/invoice.html">Invoice</a></li>
31    </ul>
32  </div>
33  <div class="main">
34    <div class="page-header">
35      MODULE 3: LẬP TRÌNH WEB VỚI JAVA

```
- Outline View**: Shows the DOM tree structure.

- Code viết HTML

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Module 03 LẬP TRÌNH WEB VỚI JAVA</title>
<link rel="stylesheet" type="text/css" href="/MVC/css/css.css">
</head>
<body>
    <div class="title">
        <div class="container">
            <div class="brand"><a href="/MVC">Module 3</a></div>
        </div>
    </div>
    <div class="sidebar">
        <ul>
            <li><a href="helloworld.html">Hello World</a></li>
            <li><a href="sumtwonumbers.html">Sum Two Numbers</a></li>
            <li><a href="summultiplenumbers.html">Sum Multiple Numbers</a></li>
            <li><a href="register.jsp">Register</a></li>
            <li><a href="login.jsp">Login</a></li>
            <li><a href="multiplicationtable.jsp">Multiplication Table</a></li>
            <li><a href="upload.html">Upload Image</a></li>
            <li><a href="upload/multi.html">Multiple Upload Image</a></li>
            <li><a href="template.jsp">Template Example</a></li>
            <li><a href="admin/publisher.html">Publisher</a>
            <li><a href="admin/category.html">Category</a></li>
            <li><a href="home.html">Home</a></li>
            <li><a href="auth/register.html">Register</a></li>
            <li><a href="auth/logon.html">Log On</a></li>
            <li><a href="admin/invoice.html">Invoice</a></li>
        </ul>
    </div>
    <div class="main">
        <div class="page-header">
            MODULE 3: LẬP TRÌNH WEB VỚI JAVA
        </div>
    </div>
</body>
```

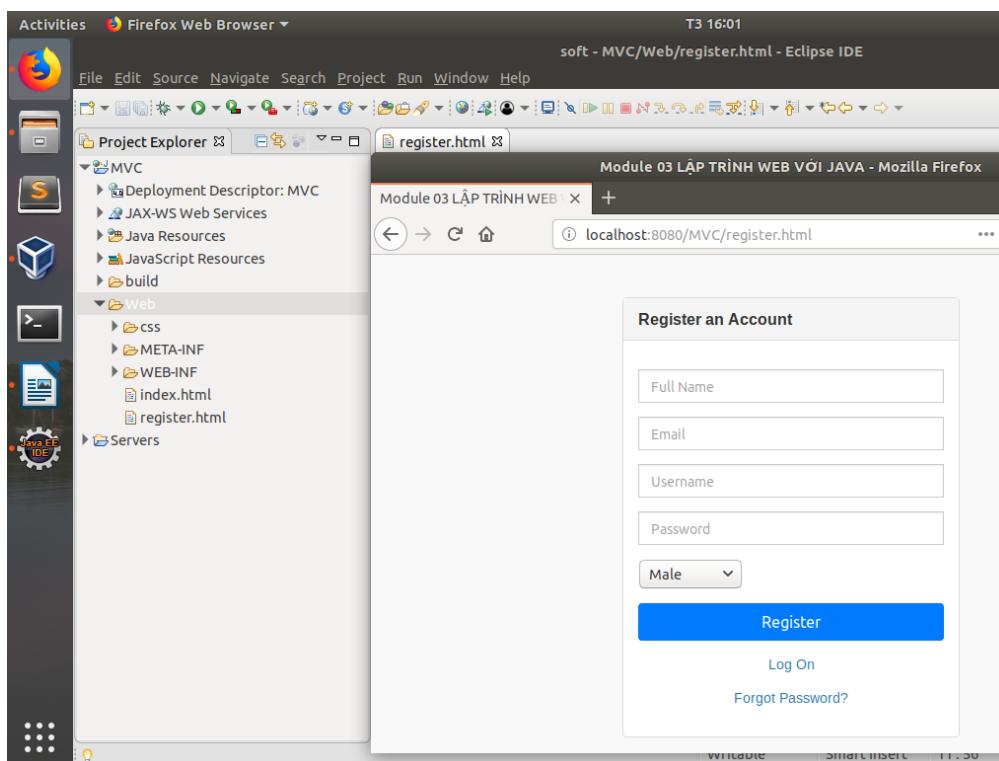
```

</div>
<h2 class="center">BÀI TẬP CHƯƠNG TRÌNH LẬP TRÌNH VIÊN CÔNG NGHỆ
JAVA</h2>
</div>
</body>
</html>

```

1.4. Web tĩnh Register HTML

- **Yêu cầu:** Xây dựng ứng dụng web tĩnh đăng ký thành viên và thực thi trong Apache Tomcat
- **Tóm tắt yêu cầu:**
 - Dựa vào tập tin css có sẵn thiết kế trang register.html đăng ký thành viên theo mẫu sau.



- **Hướng dẫn:**
 - Trong project MVC tạo tập tin html trong thư mục content web có tên là register.html.
 - Viết code HTML như sau

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Module 03 LẬP TRÌNH WEB VỚI JAVA</title>
<link rel="stylesheet" type="text/css" href="/MVC/css/css.css">
</head>
<body>
    <div class="panel-login">
        <div class="panel-heading">
            <h2 class="panel-title">Register an Account</h2>
        </div>
        <div class="panel-body">
            <form method="post">
                <p>
                    <input type="text" class="form-control" placeholder="Full Name" name="fullName">
                </p>
                <p>
                    <input type="email" class="form-control" placeholder="Email" name="email">
                </p>
                <p>
                    <input class="form-control" placeholder="Username" type="text" name="usr">
                </p>
                <p>
                    <input class="form-control" placeholder="Password" type="password" name="pwd">
                </p>
                <p>
                    <select name="gender">
                        <option value="0">Male</option>
                        <option value="1">Female</option>
                        <option value="2">Undefined</option>
                    </select>
                </p>
            </form>
        </div>
    </div>
</body>
```

```
</p>
<p>
    <button class="btn btn-primary btn-lg">Register</button>
</p>
<p class="center"><a href="logon.html">Log On</a></p>
<p class="center"><a href="forgot.html">Forgot  
Password?</a></p>
</form>
</div>
</div>
</body>
</html>
```

- Truy cập vào liên kết register.html và xem kết quả

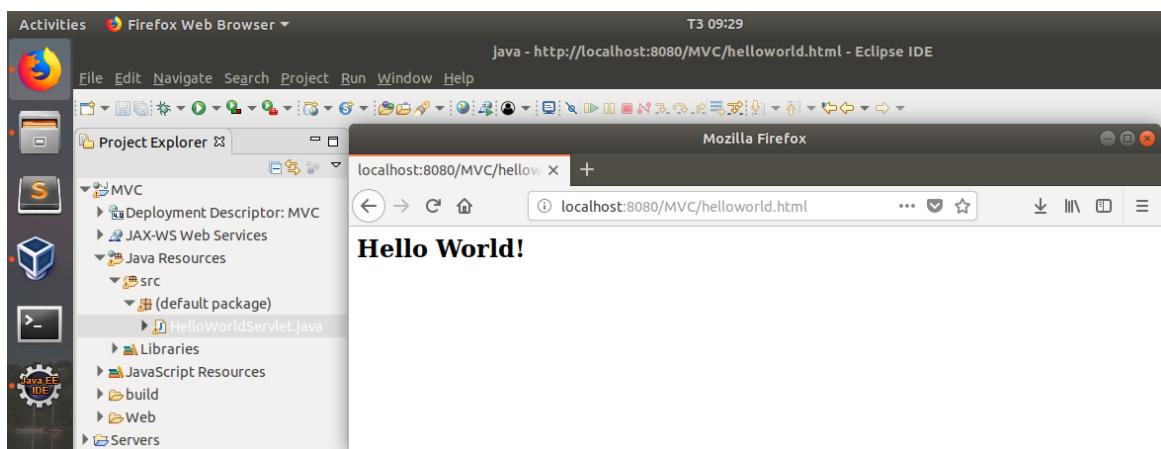
BÀI 2: Servlet



- Xây dựng một Servlet đơn giản và cấu trúc thư mục của ứng dụng Servlet
- Xây dựng cấu trúc triển khai web.xml trong ứng dụng web

2.1. Xây dựng cấu trúc ứng dụng Servlet

- **Yêu cầu:** Xây dựng một trang Servlet xuất ra một câu chào như sau:



- **Hướng dẫn:**

- Tạo project Dynamic Web đặt tên là **MVC**
- Tạo ra class có tên **HelloWordServlet.java** kế thừa từ **GenericServlet**. Annotation đến trang này là **helloworld.html**
- Hiện thực phương thức **service** để xuất ra lời chào **Hello World!** (ở trang **HelloWordServlet.java**)
- Gợi ý code cho phương thức **service**

```

@.WebServlet("/helloworld.html")
public class HelloWorldServlet extends GenericServlet {
    @Override
    public void service(ServletRequest request, ServletResponse response) throws
    ServletException, IOException {
        response.setContentType("text/html;charset=utf-8");
        try(PrintWriter pw = response.getWriter()){
            pw.write("<h2>Hello World!</h2>");
        }
    }
}

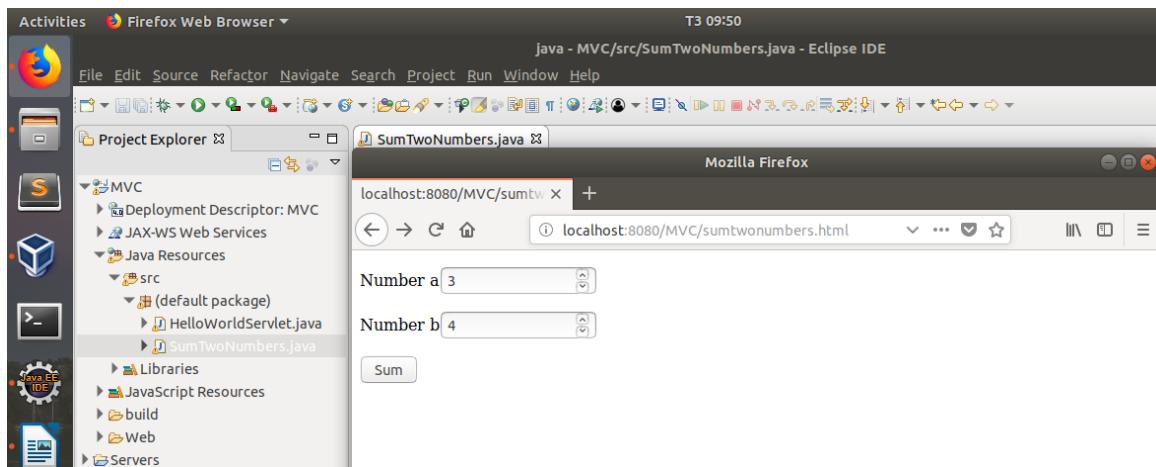
```

{}

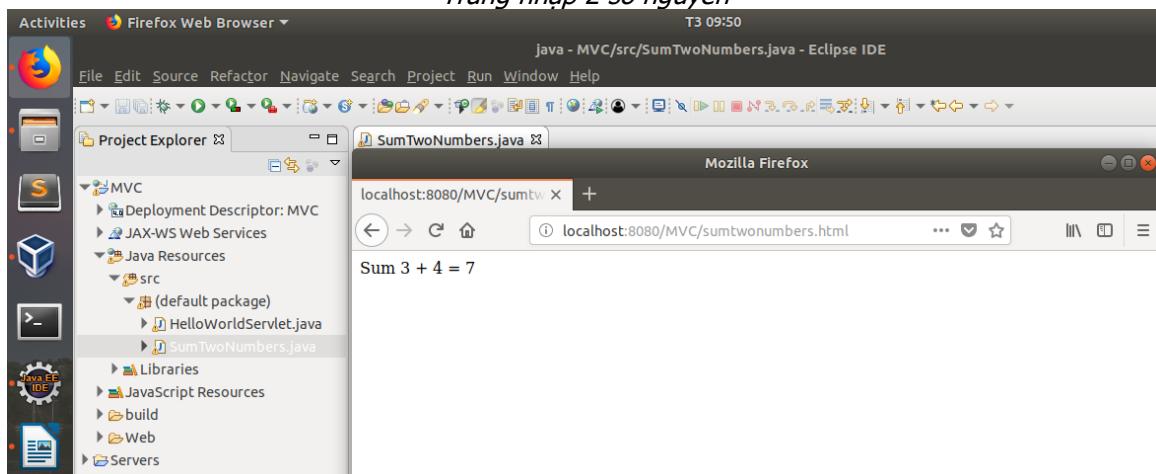
- Start Tomcat và truy cập trong trình duyệt, chạy thử link **helloworld.html** và xem kết quả.

2.2. Tính tổng 2 số

- **Yêu cầu:** Xây dựng trang Servlet để tính tổng 2 số nguyên nhập vào từ giao diện sau:



Trang nhập 2 số nguyên



Trang hiển thị kết quả tổng 2 số nhập vào

- **Hướng dẫn sử dụng:**

- Người dùng nhập vào **Number a** và **Number b** => Nhấn "Sum" => hiển thị kết quả

- **Hướng dẫn:**

Chuẩn bị phần html cho form nhập dữ liệu như sau

```

1 <form method="post">
2   <p>
3     <label>Number a</label>
4     <input type="number" name="a">
5   </p>
6   <p>
7     <label>Number b</label>
8     <input type="number" name="b">
9   </p>
10  <p>
11    <button>Sum</button>
12  </p>
13 </form>

```

- Tạo class **SumTwoNumbers** kế thừa từ **HttpServlet**, Annotation **WebServlet("/sumtwonumbers.html")** và viết phương thức **doGet** như sau:

```

@WebServlet("/sumtwonumbers.html")
public class SumTwoNumbers extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try(PrintWriter pw = response.getWriter()){
            pw.write("<form method=\"post\">");
            pw.write("<p>");
            pw.write("<label>Number a</label>");
            pw.write("<input type=\"number\" name=\"a\">");
            pw.write("</p>");
            pw.write("<p>");
            pw.write("<label>Number b</label>");
            pw.write("<input type=\"number\" name=\"b\">");
            pw.write("</p>");
            pw.write("<p>");
            pw.write("<button>Sum</button>");
            pw.write("</p>");
            pw.write("</form>");
        }
    }
}

```

}

- Viết thêm phương thức **doPost** cho class **SumTwoNumbers** như sau:

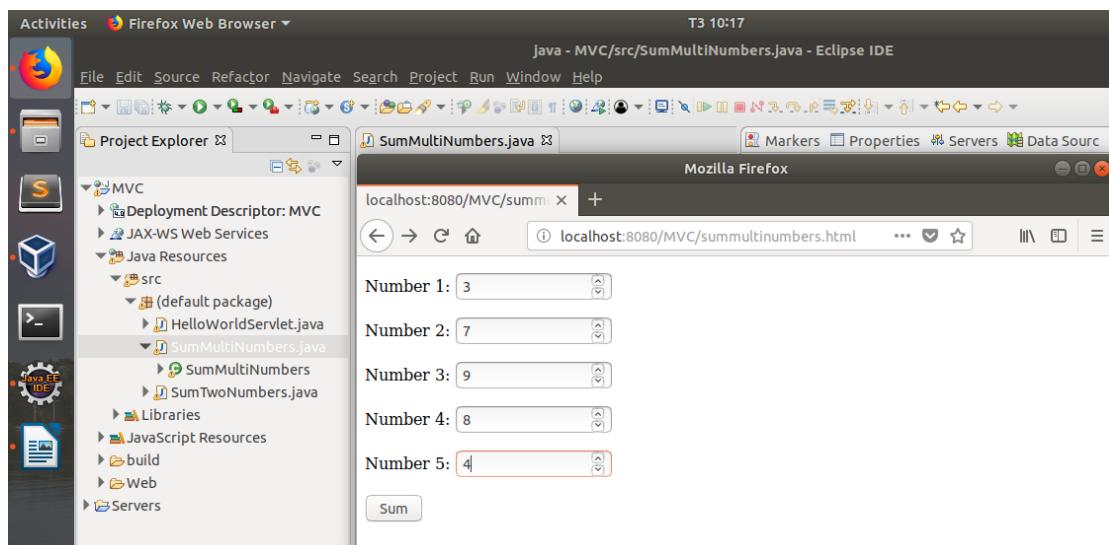
```
@WebServlet("/sumtwonumbers.html")
public class SumTwoNumbers extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        int a = Integer.parseInt(request.getParameter("a"));
        int b = Integer.parseInt(request.getParameter("b"));
        try(PrintWriter pw = response.getWriter()){
            pw.printf("<p>Sum %d + %d = %d</p>", a, b, a + b );
        }
    }
}
```

- Chạy thử chương trình truy cập vào link **sumtwonumbers.html** và xem kết quả

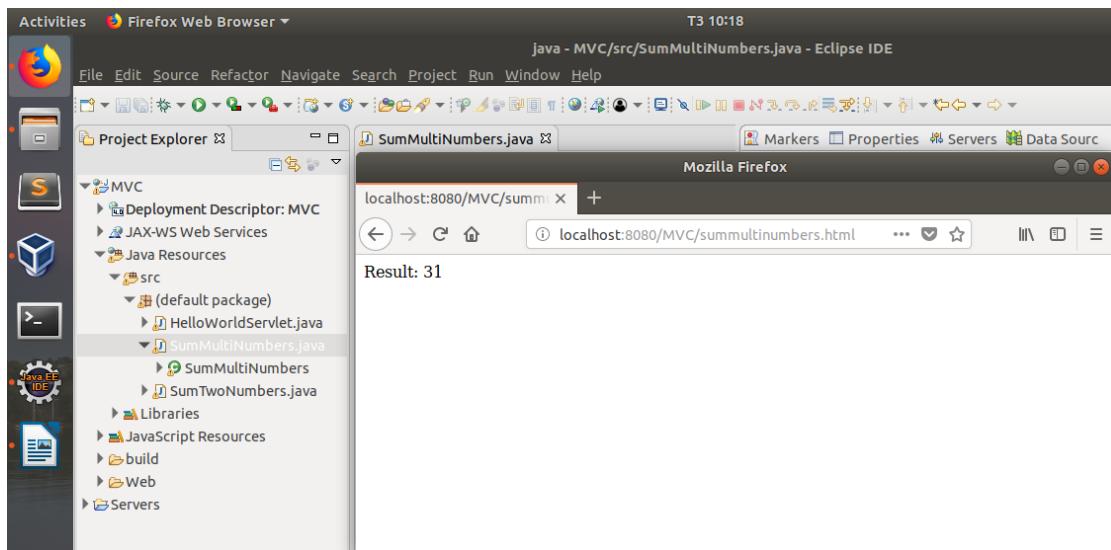
2.3. Sum Multiple Numbers

- **Yêu cầu:** Viết ứng dụng nhập vào n số nguyên và hiển thị tổng của n số nguyên đó:

Từ trang **summultiplnumbers.html**



Trang **summultiplnumbers.html GET**



Trang *summultiplnumbers.html* **POST**

- Hướng dẫn:

Trong ứng dụng web MVC tạo class **SumMultiNumbers** kế thừa từ HttpServlet và viết phương thức doGet như sau:

```
@WebServlet("/summultiplnumbers.html")
public class SumMultiNumbers extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try(PrintWriter pw = response.getWriter()){
            pw.append("<form method=\"post\">");
            for(int i = 1; i <= 5; i++) {
                pw.format("<p><label>Number %d: </label><input type=\"number\" name=\"arr\"></p>", i);
            }
            pw.append("<p><button>Sum</button></p>");
            pw.append("</form>");
        }
    }
}
```

Viết thêm phương thức **doPost** vào class **SumMultiNumbers** để xử lý kết quả:

```
@WebServlet("/summultiplnumbers.html")
public class SumMultiNumbers extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
```

```
throws ServletException, IOException {  
    String []arr = request.getParameterValues("arr");  
    int s = 0;  
    for(String item : arr) {  
        s += Integer.parseInt(item);  
    }  
    try(PrintWriter pw = response.getWriter()){  
        pw.format("<p>Result: %d</p>", s);  
    }  
}  
}
```

- Thực thi và xem kết quả link: **summultinumbers.html**

BÀI 3: JSP, EL & JSTL



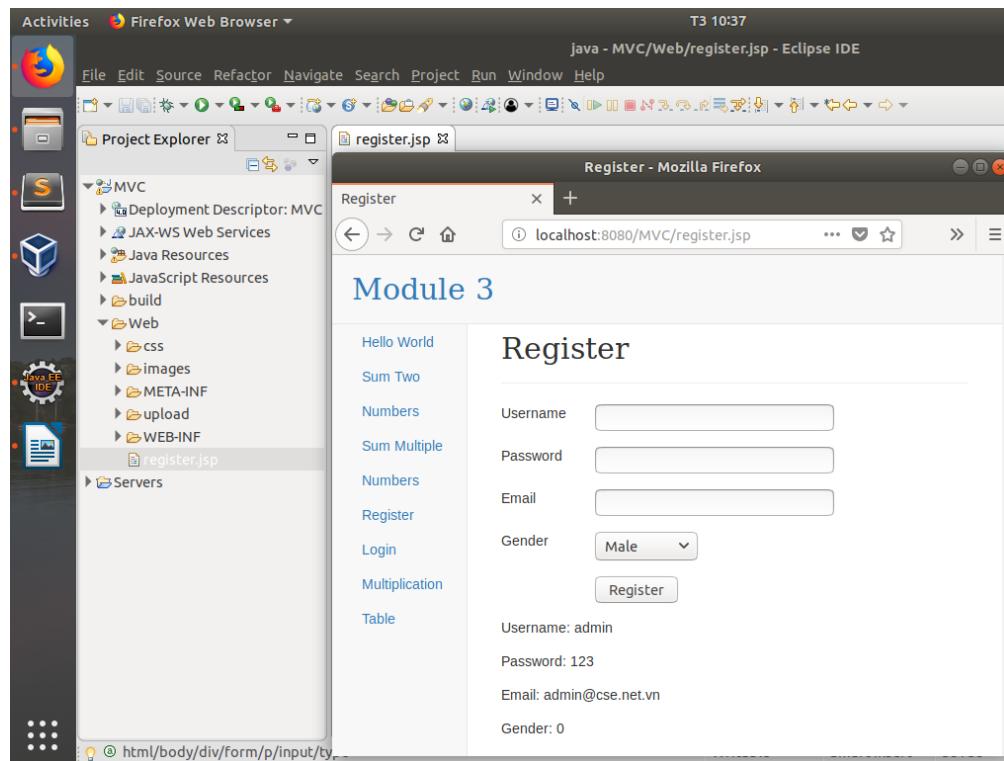
- Sử dụng các thẻ trong trang JSP
- Sử dụng các biến ẩn trong trang JSP
- Sử EL và JSTL

3.1. Register Simple

- **Yêu cầu:** Tạo ra register.jsp nhập vào thông tin thành viên và xuất ra thông tin của thành viên đó
- **Hướng dẫn sử dụng:**
 - Người dùng nhập Username, Password, Email, chọn Gender và nhấn “Register”.
- **Tóm tắt yêu cầu:**
 - Thiết kế giao diện:

The screenshot shows the Eclipse IDE interface. On the left is the Project Explorer view, which lists a project named 'MVC' containing 'Deployment Descriptor: MVC', 'JAX-WS Web Services', 'Java Resources', 'JavaScript Resources', 'build', and 'Web' folders. Inside the 'Web' folder are 'css', 'images', 'META-INF', 'upload', and 'WEB-INF' sub-folders, with 'register.jsp' highlighted. To the right of the Project Explorer is the Java Editor view showing the code for 'register.jsp'. Below these is the Eclipse Navigator view. On the far right is the Mozilla Firefox browser window titled 'Register - Mozilla Firefox'. The address bar shows 'localhost:8080/MVC/register.jsp'. The page content includes a header 'Module 3' and a form titled 'Register' with fields for 'Username' (admin), 'Password' (hidden), 'Email' (admin@cse.net.vn), and 'Gender' (Male). A 'Register' button is at the bottom of the form.

Trang register.jsp **GET**

Trang register.jsp **POST**

- Hướng dẫn:

- Tạo ra trang **register.jsp** trong project **MVC** với nội dung như sau

```
<form method="post" class="form">
    <p>
        <label>Username</label>
        <input type="text" name="usr">
    </p>
    <p>
        <label>Password</label>
        <input type="password" name="pwd">
    </p>
    <p>
        <label>Email</label>
        <input type="text" name="email">
    </p>
    <p>
        <label>Gender</label>
        <select name="gender">
            <option value="0">Male</option>

```

```

        <option value="1">Female</option>
        <option value="2">Undefined</option>
    </select>
</p>
<p>
    <button>Register</button>
</p>
</form>

```

- Viết thêm phần xử lý **POST** cho trang **register.jsp** như sau:

```

<%
    if(request.getMethod().equals("POST")){
%>

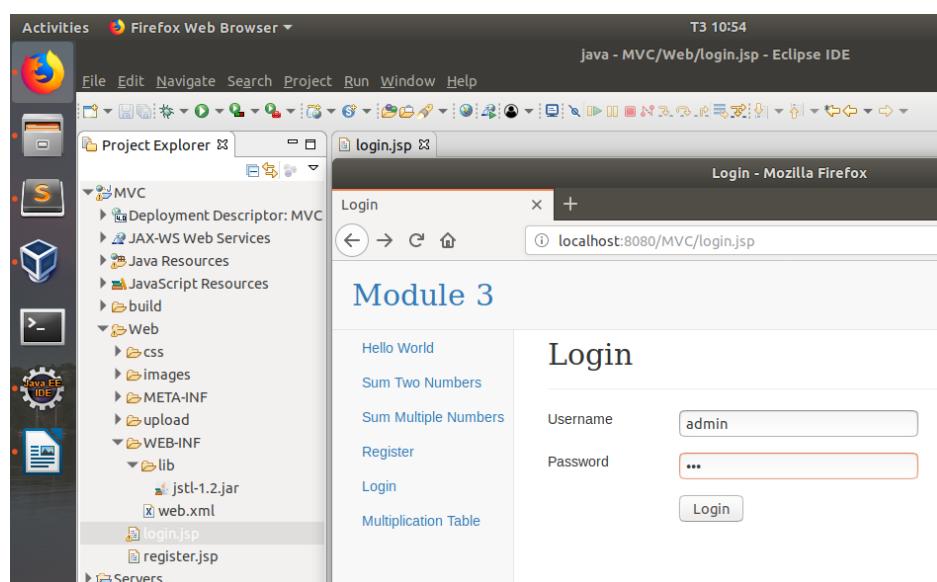
    <p>Username: <%= request.getParameter("usr") %></p>
    <p>Password: <%= request.getParameter("pwd") %></p>
    <p>Email: <%= request.getParameter("email") %></p>
    <p>Gender: <%= request.getParameter("gender") %></p>
<%
}
%>

```

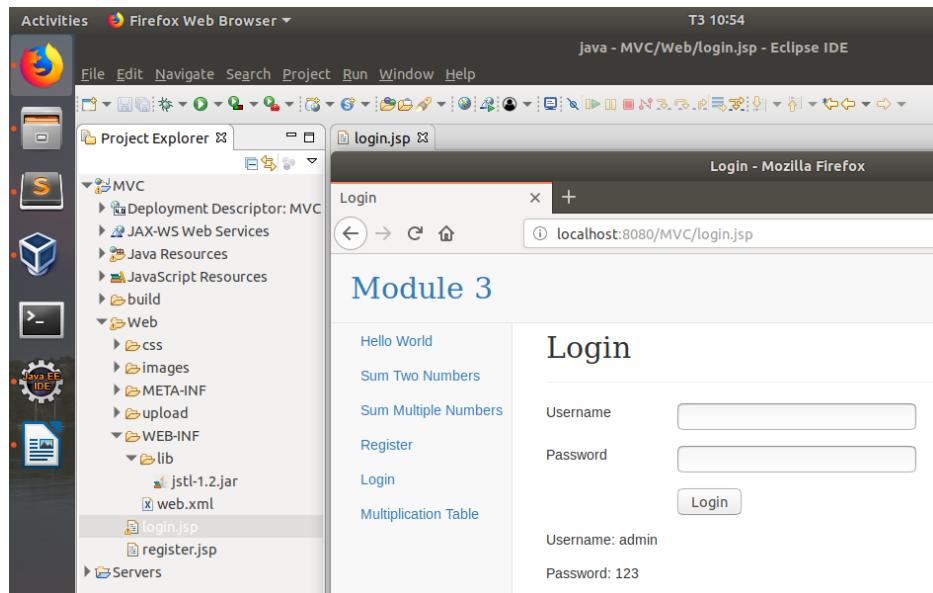
- Chạy thử **register.jsp**

3.2. Login Simple (Kết hợp JSTL)

- **Yêu cầu:** tạo trang Login.jsp sử dụng **JSTL** để hiển thị thông tin người dùng nhập vào:



Kết quả trang login.jsp **GET**



Kết quả trang login.jsp **POST**

- Hướng dẫn:

- Trong project **MVC**: Thêm thư viện **JSTL** vào thư mục **lib**
- Tạo trang **login.jsp** với nội dung như sau:

```
<form method="post" class="form">
    <p>
        <label>Username</label>
        <input type="text" name="usr">
    </p>
    <p>
        <label>Password</label>
        <input type="password" name="pwd">
    </p>
    <p>
        <button>Login</button>
    </p>
</form>
```

- Trong trang login.jsp thêm taglib phía trên
- Trong trang login.jsp bổ sung phía dưới phần xử lý thông tin như sau:

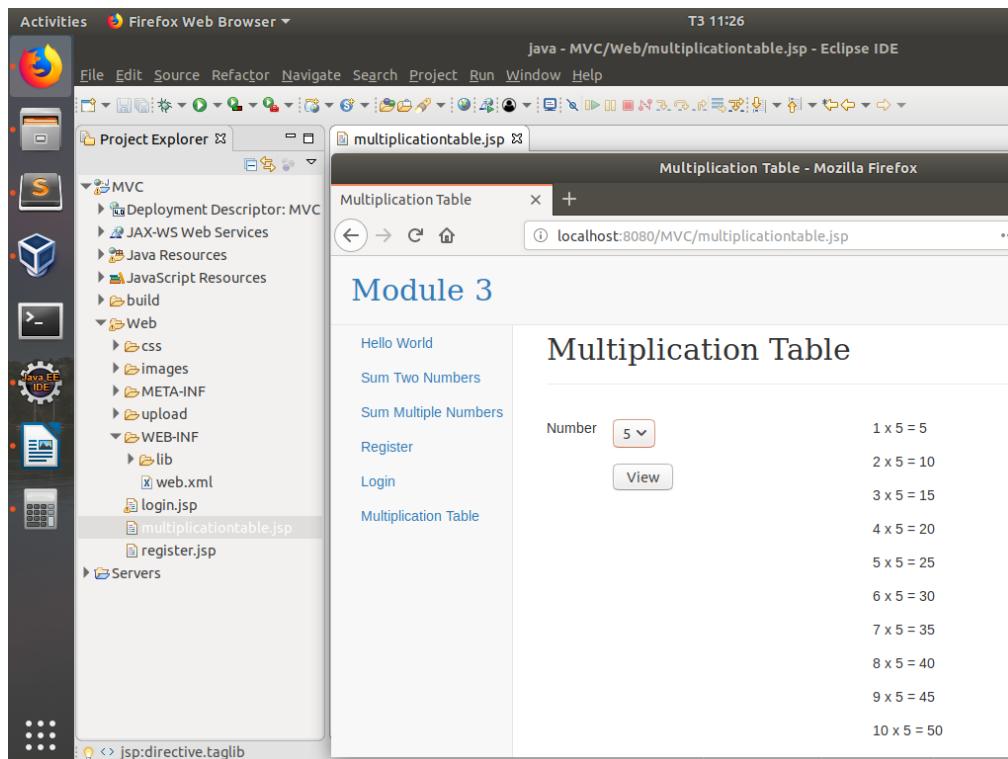
```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```

```
<form method="post" class="form">  
    <p>  
        <label>Username</label>  
        <input type="text" name="usr">  
    </p>  
    <p>  
        <label>Password</label>  
        <input type="password" name="pwd">  
    </p>  
    <p>  
        <button>Login</button>  
    </p>  
</form>  
<c:if test="${pageContext.request.method == 'POST'}">  
    <p>Username: ${param.usr}</p>  
    <p>Password: ${param.pwd}</p>  
</c:if>
```

- Chạy lại trang login.jsp để xem kết quả

3.3. Multiplication Table

- **Yêu cầu:** tạo trang **multiplictiontable.jsp** yêu cầu người dùng chọn 1 số từ select list rồi hiển thị bảng tính phép nhân này với giá trị vừa chọn:



Kết quả hiển thị bảng cửu chương 5

- Hướng dẫn:

- Xây dựng nội dung trang multiplicationtable.jsp với nội dung như sau

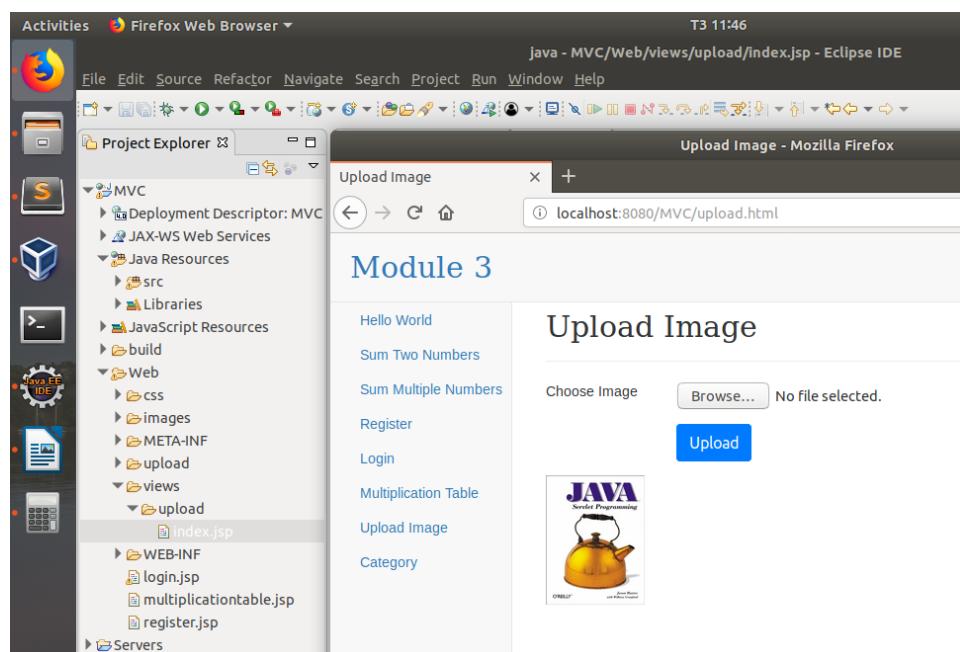
```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<form method="post" class="form">
    <p>
        <label>Number</label>
        <select name="num">
            <c:forEach var="i" begin="2" end="9" step="1">
                <option value="${i}">${i}</option>
            </c:forEach>
        </select>
    </p>
    <p>
        <button>View</button>
    </p>
</form>
<c:if test="#{pageContext.request.method == 'POST'}">
    <c:forEach var="i" begin="1" end="10" step="1">
```

```
<p>${i} x ${param.num} = ${i * param.num}</p>
</c:forEach>
</c:if>
```

- Chạy lại trang **multipletable.jsp** để xem kết quả

3.4. Upload Image (JSP và Servlet)

- **Yêu cầu:** Tạo trang Upload Image cho người dùng chọn 1 hình ảnh sau đó hiển thị ảnh đó lên trang vừa chọn:



- **Hướng dẫn:**

- Tạo trang **index.jsp** trong thư mục views/upload với nội dung như sau
- Lưu ý sử dụng **enctype="multipart/form-data"**

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<form method="post" class="form" enctype="multipart/form-data">
    <p>
        <label>Choose Image</label>
        <input type="file" name="f">
    </p>
    <p>
        <button class="btn btn-primary">Upload</button>
    </p>
</form>
```

```
<c:if test="#{not empty img}">
    
</c:if>
```

- Tạo class UploadServlet lưu ý sử dụng phần Annotation **@MultipartConfig** với nội dung phần **doGet** như sau

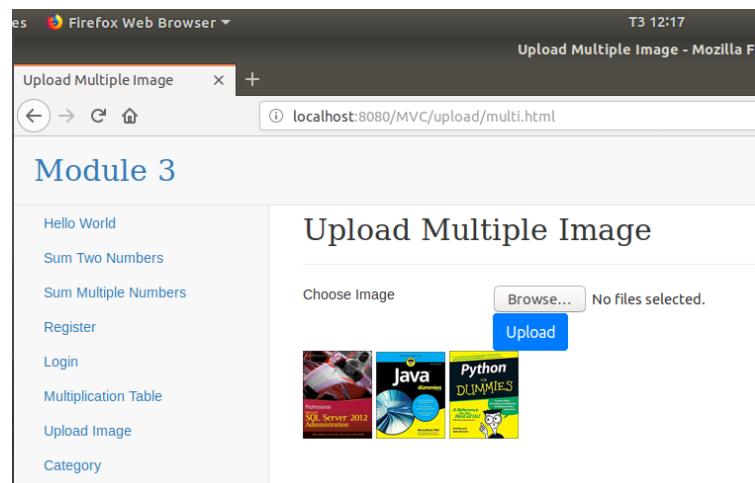
```
@WebServlet("/upload.html")
@MultipartConfig
public class UploadServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
    ServletException, IOException {
        request.getRequestDispatcher("/views/upload/index.jsp").forward(request, response);
    }
    private static String upload(String path, Part part) throws IOException {
        String fileName = part.getSubmittedFileName();
        try(InputStream is = part.getInputStream()){
            try(OutputStream os = new FileOutputStream(new File(path + fileName))){
                int len = 0;
                byte[] bytes = new byte[1024];
                while ( ( len = is.read(bytes, 0, 1024)) > 0 ) {
                    os.write(bytes, 0, len);
                }
            }
        }
        return fileName;
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
    ServletException, IOException {
        Part part = request.getPart("f");
        String path = request.getServletContext().getRealPath("/upload/");
        String fileName = upload(path, part);
        request.setAttribute("img", fileName);
        doGet(request, response);
    }
}
```

{}

- Chạy lại trang upload.html để xem kết quả

3.5. Upload Multiple Image (bài làm thêm)

- **Yêu cầu:** Xây dựng trang upload/multi.html người dùng có thể chọn upload nhiều hình ảnh sau đó hiển thị lên web:



Trang upload/multi.html

- **Hướng dẫn:**

- Xây dựng trang multi.jsp trong thư mục views/upload
- Phần dưới trong multi.jsp có sử dụng code để hiển thị kết quả hình ảnh
- Phần trên cùng sử dụng tablib JSLT

```
<form method="post" class="form" enctype="multipart/form-data">
    <div>
        <label>Choose Image</label>
        <input type="file" multiple="multiple" name="f">
    </div>
    <div>
        <button class="btn btn-primary">Upload</button>
    </div>
</form>
<c:if test="${not empty list}">
    <c:forEach var="img" items="${list}">
```

```

</c:forEach>
</c:if>
```

- Tạo servlet **UploadMultiple** với nội dung như sau

```
@WebServlet("/upload/multi.html")
@MultipartConfig
public class MultiUploadServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        request.getRequestDispatcher("/views/upload/multi.jsp").forward(request, response);
    }
    private static String upload(String path, Part part) throws IOException {
        String fileName = part.getSubmittedFileName();
        try(InputStream is = part.getInputStream()){
            try(OutputStream os = new FileOutputStream(new File(path + fileName))){
                int len = 0;
                byte[] bytes = new byte[1024];
                while ( ( len = is.read(bytes, 0, 1024)) > 0) {
                    os.write(bytes, 0, len);
                }
            }
        }
        return fileName;
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        String path = request.getServletContext().getRealPath("/upload/");
        List<String> list = new LinkedList();
        for(Part part : request.getParts()) {
            String fileName = upload(path, part);
            list.add(fileName);
        }
    }
}
```

```

        request.setAttribute("list", list);
        doGet(request, response);
    }
}

```

- Chạy lại trang upload/multi.html để xem kết quả

3.6. Template

- **Yêu cầu:** Xây dựng template dùng chung cho nhiều trang
- **Hướng dẫn sử dụng:**
 - Sử dụng tag để tạo template dùng chung
 - Sử dụng Tag Library descriptor để khai báo cho template
 - Khai báo taglib để sử dụng template đã được xây dựng trước.
- **Tóm tắt yêu cầu:**
 - Tạo file bg.tag trong content directory web/WEB-INF/tags với nội dung HTML như sau:

```

<%@ tag language="java" pageEncoding="UTF-8"%>
<%@attribute name="title" required="true" %>
<%@attribute name="content" required="true" fragment="true" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>${title}</title>
<link rel="stylesheet" type="text/css" ref="${pageContext.request.contextPath}/css/css.css">
</head>
<body>
    <div class="title">
        <div class="container">
            <div class="brand">
                <a href="${pageContext.request.contextPath}">
                    MODULE 3: LẬP TRÌNH WEB VỚI JAVA
                </a>
            </div>
        </div>
    </div>
</body>

```

```

        </div>
    </div>
</div>
<div class="sidebar">
    <ul>
        <li><a href="helloworld.html">Hello World</a></li>
        <li><a href="sumtwonumbers.html">Sum Two Numbers</a></li>
        <li><a href="summultiplenumbers.html">Sum Multiple Numbers</a></li>
        <li><a href="register.jsp">Register</a></li>
        <li><a href="login.jsp">Login</a></li>
        <li><a href="multiplicationtable.jsp">Multiplication Table</a></li>
        <li><a href="upload.html">Upload Image</a></li>
        <li><a href="upload/multi.html">Multiple Upload Image</a></li>
        <li><a href="template.jsp">Template Example</a></li>
        <li><a href="category.html">Category</a></li>
    </ul>
</div>
<div class="main">
    <jsp:invoke fragment="content" />
</div>
</body>
</html>

```

- Tạo file Tag Library Descriptors đặt tên là template.tld trong thư mục WEB-INF với nội dung như sau

```

<?xml version="1.0" encoding="UTF-8" ?>
<taglib version="2.1" xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-jsptaglibrary_2_1.xsd">
<tlib-version>1.0</tlib-version>
<short-name>template</short-name>
<uri>/WEB-INF/template</uri>
<tag-file>
    <name>bg</name>
    <path>/WEB-INF/tags/bg.tag</path>

```

```
</tag-file>
</taglib>
```

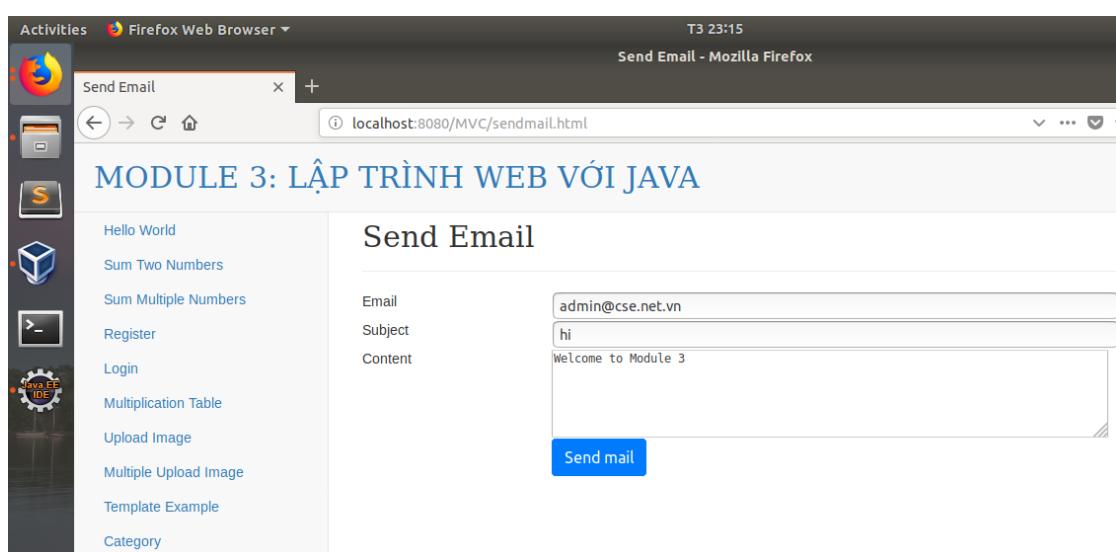
- Tạo trang template.jsp để sử dụng template đã được khai báo trước

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@taglib prefix="me" uri="/WEB-INF/template" %>
<me:bg title="Use Template">
    <jsp:attribute name="content">
        <div class="page-header">
            MODULE 3: LẬP TRÌNH WEB VỚI JAVA
        </div>
        <p>BÀI TẬP CHƯƠNG TRÌNH LẬP TRÌNH VIÊN CÔNG NGHỆ JAVA</p>
    </jsp:attribute>
</me:bg>
```

- Chạy thử template.jsp để xem kết quả

3.7. Send mail (kết hợp template, servlet, jsp)

- **Yêu cầu:** tạo trang gửi email theo mẫu sau



Trang sendmail.html

- **Hướng dẫn sử dụng:**

- Người dùng nhập email, subject, content => nhấn "Send" => Gửi email với chủ đề và nội dung vừa nhập.
- **Tóm tắt yêu cầu:**
 - Thiết kế giao diện: gồm có 1 form cho nhập email, subject, content.
 - Xử lý gửi mail với nội dung và chủ đề vừa nhập vào. Thông báo kết quả gửi thành công hay thất bại
- **Hướng dẫn:**
 - Tạo trang **mail.jsp** trong thư mục **views/mail/sendmail.jsp** với nội dung như sau

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="me" uri="/WEB-INF/template" %>
<me:bg title="Send Email">
    <jsp:attribute name="content">
        <h3 class="page-header">Send Email</h3>
        <form method="post" class="form">
            <p>
                <label>Email</label>
                <input type="email" name="email" class="col-7">
            </p>
            <p>
                <label>Subject</label>
                <input type="text" name="subject" class="col-7">
            </p>
            <p>
                <label>Content</label>
                <textarea rows="5" cols="20" name="content" class="col-7"></textarea>
            </p>
            <p>
                <button class="btn btn-primary">Send mail</button>
            </p>
        </form>
        <c:if test="${not empty msg }">
```

```
<div class="error">${msg}</div>
</c:if>
</jsp:attribute>
</me:bg>
```

- Tạo class **SendMailServlet** forward đến trang **sendmail.jsp** vừa được thiết kế đặt tên

```
@WebServlet("/sendmail.html")
public class SendMailServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        response.setContentType("text/html;charset=utf-8");
        request.getRequestDispatcher("/views/mail/sendmail.jsp").forward(request,
response);
    }
    private static boolean send(String to, String subject, String text) {
        try {
            Properties props = System.getProperties();
            props.setProperty("mail.smtp.host", "smtp.gmail.com");
            props.setProperty("mail.smtp.auth", "true");
            props.setProperty("mail.smtp.user", "admin@cse.net.vn");
            props.setProperty("mail.smtp.password", "cse");
            props.setProperty("mail.smtp.port", "465");
            props.setProperty("mail.smtp.starttls.enable", "true");
            props.setProperty("mail.smtp.socketFactory.port", "465");

            props.setProperty("mail.smtp.socketFactory.class","javax.net.ssl.SSLSocketFactory");
            Session session = Session.getDefaultInstance(props, new Authenticator()
{
    @Override
    protected PasswordAuthentication getPasswordAuthentication() {

        return new PasswordAuthentication("admin@cse.net.vn",
"cse");
    }
});
MimeMessage msg = new MimeMessage(session);
```

```
msg.setFrom(new InternetAddress("admin@cse.net.vn"));
msg.setRecipient(Message.RecipientType.TO, new InternetAddress(to));
msg.setSubject(subject);
msg.setText(text);
Transport.send(msg);
return true;
} catch (AddressException e) {
e.printStackTrace();
} catch (MessagingException e) {
e.printStackTrace();
}
return false;
}

protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
response.setContentType("text/html;charset=utf-8");
String msg = "Failed";
if(send(request.getParameter("email"), request.getParameter("subject"),
request.getParameter("content"))) {
msg = "Success";
}
request.setAttribute("msg", msg);
doGet(request, response);
}

}
```

- Chạy trang **sendmail.html** xem kết quả gửi email

BÀI 4: MVC



Kết hợp sức mạnh của Java, Servlet, JSP, JSTL, Tag Library để xây dựng theo mô hình MVC

4.1. Xây dựng mô hình MVC

- **Yêu cầu:** Xây dựng mô hình MVC cho project
- **Tóm tắt yêu cầu:**
 - Chuẩn bị thư viện kết nối cơ sở dữ liệu
 - Thiết kế model
 - Thiết kế controller
 - Thiết kế views
 - Qui tắc xử lý
 - Tạo class Entity tương ứng trong model.
 - Tạo class EntityRepository kế thừa từ abstract class Repository trong model
 - Tạo servlet EntityController forward đến trang jsp
 - Tạo trang jsp trong content directory Web/views/entity
- **Hướng dẫn:**
 - Chép thư viện **mysql driver** vào thư mục WEB-INF/lib:
 - Tạo package **model** lưu trữ class java tương tác với cơ sở dữ liệu
 - Tạo package **controller** chứa các **serverlet**
 - Tạo thư mục **views/publisher** trong content directory Web
 - Tạo abstract class **Repository** trong package **model**

```
public abstract class Repository {
    protected Connection connection;
    protected Statement stmt;
    protected ResultSet rs;
    protected PreparedStatement pstmt;
    static {
```

```
try {
    Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e) {
    e.printStackTrace();
}

protected void open() throws SQLException {
    connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/MiniShop?useUnicode=true&characterEncoding=UTF-8", "cse.net.vn", "cse");
}

protected void close() throws SQLException {
    if (rs != null) {
        rs.close();
    }
    if (stmt != null) {
        stmt.close();
    }
    if(pstmt != null) {
        pstmt.close();
    }
    if (connection != null) {
        connection.close();
    }
}
```

- Để xem được kết quả ta qua các bài tập tiếp theo sẽ rõ hơn về mô hình MVC

4.2. CRUD Publisher by JDBC

- **Yêu cầu:** Xây dựng chức năng xem danh sách, thêm, xóa, sửa trong bảng **publisher**

MODULE 3: LẬP TRÌNH WEB VỚI JAVA

Manage Publisher

	ID	Name	Edit	Delete
<input type="checkbox"/>	1	Appress		
<input type="checkbox"/>	2	O'Reilly		
<input type="checkbox"/>	3	Thien Long		
<input type="checkbox"/>	4	Ky thuat Ha Noi		
<input type="checkbox"/>	5	DHQG HCM		
<input type="checkbox"/>	6	Dong Nai		
<input type="checkbox"/>	7	7 Up		

Trang admin/publisher.html

Add New Publisher

Name	<input type="text" value="Nha Xuat Ban Ky Thuat"/>
<input type="button" value="Save"/>	

Trang admin/publisher/add.html

Edit Publisher

Name	<input type="text" value="Dong Nai"/>
<input type="button" value="Save"/>	

Trang admin/publisher/edit.html

- Hướng dẫn sử dụng:

- Người dùng truy cập vào link **admin/publisher.html** để xem xem thông tin Publisher
- Truy cập vào link **Add New Publisher** để thêm 1 publisher
- Truy cập vào icon **edit** để cập nhật một publisher
- Truy cập vào icon **delete** để xóa một publisher
- Check chọn để **delete** nhiều publisher

- Hướng dẫn:

- Tạo class **Publisher** trong package **model**

```
package model;
public class Publisher {
    private int id;
    private String name;
    public Publisher(int id, String name) {
        this.id = id;
        this.name = name;
    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

- Xây dựng trang **admin/publisher.html**
- Tạo class **PublisherRepository** trong package **model** kế thừa class **Repository** và viết phương thức **getPublishers**

```
package model
public class PublisherRepository extends Repository {
    public List<Publisher> getPublishers() throws SQLException{
        try {
            open();
            stmt = connection.createStatement();
            rs = stmt.executeQuery("SELECT * FROM Publisher");
            List<Publisher> list = new LinkedList<Publisher>();
        }
```

```

        while(rs.next()) {
            list.add(new Publisher(rs.getInt("PublisherId"),
rs.getString("PublisherName")));
        }
        return list;
    }finally {
        close();
    }
}
}

```

- Tạo servlet **PublisherController** forward **/views/publisher/index.jsp** trong package **controller**

```

@WebServlet("/admin/publisher.html")
public class PublisherController extends HttpServlet {
    private PublisherRepository repository = new PublisherRepository();
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        try {
            request.setAttribute("list", repository.getPublishers());
            request.getRequestDispatcher("/views/publisher/index.jsp").forward(request, response);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

- Tạo trang **/views/publisher/index.jsp**

```

<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@taglib prefix="me" uri="/WEB-INF/template" %>
<me:bg title="Publisher">
    <jsp:attribute name="content">
        <div class="page-header">Manage Publisher</div>
        <p><a href="${pageContext.request.contextPath}/publisher/add.html">Add

```

New Publisher</p>

```
<form method="post"
action="${pageContext.request.contextPath}/publisher/dels.html">
    <table class="table">
        <tr>
            <td><button class="btn btn-info">Delete</button> </td>
            <th>Id</th>
            <th>Name</th>
            <th>Edit</th>
            <th>Delete</th>
        </tr>
        <c:forEach items="${list}" var="o">
            <tr>
                <td>
                    <input type="checkbox" value="${o.id}"
name="ids">
                </td>
                <td>${o.id}</td>
                <td>${o.name}</td>
                <td>
                    <a
href="${pageContext.request.contextPath}/publisher/edit.html?id=${o.id}">
                        
                    </a>
                </td>
                <td>
                    <a
href="${pageContext.request.contextPath}/publisher/del.html?id=${o.id}">
                        
                    </a>
                </td>
            </tr>
        </c:forEach>
    </table>
</form>
```

```
</jsp:attribute>
</me:bg>
```

- Truy cập **admin/publisher.html** xem kết quả
- Xây dựng trang **/publisher/add.html**
- Viết thêm phương thức **add** vào class **PublisherRepository** trong package model

package model

```
public class PublisherRepository extends Repository {
    public int add(Publisher obj) throws SQLException {
        try {
            open();
            pstmt = connection.prepareStatement("INSERT INTO Publisher
(PublisherName) VALUES(?)");
            pstmt.setString(1, obj.getName());
            return pstmt.executeUpdate();
        }finally {
            close();
        }
    }
}
```

- Tạo Servlet **PublisherAddController** forward đến **/views/publisher/add.jsp** trong package **controller**

```
@WebServlet("/publisher/add.html")
public class PublisherAddController extends HttpServlet {
    private PublisherRepository repository = new PublisherRepository();
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        request.getRequestDispatcher("/views/publisher/add.jsp").forward(request,
response);
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        if(!request.getParameter("name").equals("")) {
            try {
                if(repository.add(new Publisher(0, request.getParameter("name"))))
```

```

> 0) {
    response.sendRedirect(request.getContextPath() +
    "/admin/publisher.html");
}
else{
    request.setAttribute("msg", "Inserted Failed");
    doGet(request, response);
}

} catch (SQLException e) {
    e.printStackTrace();
}
}
}
}
}

```

- Tạo trang **views/publisher/add.jsp** với nội dung như sau

```

<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@taglib prefix="me" uri="/WEB-INF/template" %>
<me:bg title="Publsher">
    <jsp:attribute name="content">
        <div class="page-header">Edit Publisher</div>
        <form method="post" class="form">
            <input type="hidden" name="id" value="${o.id}">
            <div>
                <label>Name</label>
                <input type="text" name="name" value="${o.name}">
            </div>
            <div>
                <button class="btn btn-primary">Save</button>
            </div>
        </form>
        <c:if test="${not empty msg}">
            <p class="error">${msg}</p>
        </c:if>
    </jsp:attribute>
</me:bg>

```

```
</jsp:attribute>
</me:bg>
```

- Truy cập vào liên kết **Add New Publisher** để xem kết quả
- Tạo trang **/publisher/add.html**
- Viết thêm 2 phương thức **getPublisher** và **edit** vào class **PublisherRepository**

package model

```
public class PublisherRepository extends Repository {
    public Publisher getPublisher(int id) throws SQLException{
        try {
            open();
            pstmt = connection.prepareStatement("SELECT * FROM Publisher WHERE
PublisherId = ?");
            pstmt.setInt(1, id);
            rs = pstmt.executeQuery();
            if(rs.next()) {
                return new Publisher(rs.getInt("PublisherId"),
rs.getString("PublisherName"));
            }
            return null;
        }finally {
            close();
        }
    }
    public int edit(Publisher obj) throws SQLException {
        try {
            open();
            pstmt = connection.prepareStatement("UPDATE Publisher SET
PublisherName = ? WHERE PublisherId = ?");
            pstmt.setString(1, obj.getName());
            pstmt.setInt(2, obj.getId());
            return pstmt.executeUpdate();
        }finally {
            close();
        }
    }
}
```

```
}
```

- Tạo servlet **PublisherEditController** trong package **controller** forward đến /views/publisher/edit.jsp

```
@WebServlet("/publisher/edit.html")
public class PublisherEditController extends HttpServlet {
    private PublisherRepository repository = new PublisherRepository();
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        try {
            int id = Integer.parseInt(request.getParameter("id"));
            request.setAttribute("o", repository.getPublisher(id));

            request.getRequestDispatcher("/views/publisher/edit.jsp").forward(request,
response);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        if(!request.getParameter("name").equals("")) {
            try {
                int id = Integer.parseInt(request.getParameter("id"));
                if(repository.edit(new Publisher(id,
request.getParameter("name")))) > 0) {
                    response.sendRedirect(request.getContextPath() +
"/admin/publisher.html");
                }else{
                    request.setAttribute("msg", "Edit Failed");
                    doGet(request, response);
                }
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

```
}
```

- Tạo **/views/publisher/edit.jsp** với nội dung như sau

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@taglib prefix="me" uri="/WEB-INF/template" %>
<me:bg title="Publsher">
    <jsp:attribute name="content">
        <div class="page-header">Edit Publisher</div>
        <form method="post" class="form">
            <input type="hidden" name="id" value="${o.id}">
            <div>
                <label>Name</label>
                <input type="text" name="name" value="${o.name}">
            </div>
            <div>
                <button class="btn btn-primary">Save</button>
            </div>
        </form>
        <c:if test="${not empty msg}">
            <p class="error">${msg}</p>
        </c:if>
    </jsp:attribute>
</me:bg>
```

- Truy cập vào icon **edit** để xem kết quả
- Tạo trang **/admin/publisher/del.html**
- Tạo thêm phương thức **delete** vào class **PublisherRepository**

```
package model
public class PublisherRepository extends Repository {
    public int delete(int id) throws SQLException {
        try {
            open();
            pstmt = connection.prepareStatement("DELETE FROM Publisher WHERE
```

```

PublisherId = "?");
        pstmt.setInt(1, id);
        return pstmt.executeUpdate();
    }finally {
        close();
    }
}
}

```

- Tạo servlet **PublisherEditController** trong package **controller**

```

@WebServlet("/admin/publisher/del.html")
public class PublisherDelController extends HttpServlet {
    private PublisherRepository repository = new PublisherRepository();
    protected void service(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        if(request.getParameter("id") != null) {
            try {
                int id = Integer.parseInt(request.getParameter("id"));
                repository.delete(id);
                response.sendRedirect(request.getContextPath() +
"/publisher.html");
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}

```

- Truy cập vào icon **delete** để xem kết quả
- Thực hiện **multiple delete** trên trang **admin/publisher.html**
- Viết thêm phương thức **delete** vào class **PublisherRepository**

```

package model
public class PublisherRepository extends Repository {
    public int[] delete(List<Integer> list) throws SQLException {
        try {
            open();
            pstmt = connection.prepareStatement("DELETE FROM Publisher WHERE
PublisherId = ?");
            for (Integer id : list) {
                pstmt.setInt(1, id);
                pstmt.addBatch();
            }
            return pstmt.executeBatch();
        }finally {
            close();
        }
    }
}

```

- Viết thêm phương thức **doPost** vào servlet **PublisherController**

```

@WebServlet("/admin/publisher.html")
public class PublisherController extends HttpServlet {
    private PublisherRepository repository = new PublisherRepository();
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        List<Integer> list = new LinkedList<>();
        for (String id : request.getParameterValues("ids")) {
            list.add(Integer.parseInt(id));
        }
        try {
            repository.delete(list);
            doGet(request, response);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

{}

- Quay lại link **/admin/publisher.html** check chọn và nhấn delete xem kết quả

4.3. Home.html

- **Yêu cầu:** Xây dựng chức năng hiển thị sản phẩm trên trang Home.html
 - Xử lý phần phân trang.

MODULE 3: LẬP TRÌNH WEB VỚI JAVA

Hello World
 Sum Two Numbers
 Sum Multiple Numbers
 Register
 Login
 Multiplication Table
 Upload Image
 Multiple Upload Image
 Template Example
 Category



1 2 3

- Hướng dẫn sử dụng:

- Người dùng truy cập vào link **home.html** để xem danh sách sản phẩm.
- Người dùng có thể chọn link page để xem các sản phẩm ở trang khác

- Hướng dẫn:

- Tạo class **Product** trong package **model**

```

package model;
public class Product {
    private int id;
    private String title;
    private String isbn;
    private int price;
}
  
```

```
private String pages;
private String imageUrl;
public Product(int id, String title, String isbn, int price, String pages, String imageUrl) {
    this.id = id;
    this.title = title;
    this.isbn = isbn;
    this.price = price;
    this.pages = pages;
    this.imageUrl = imageUrl;
}
public int getId() {
    return id;
}
public void setId(int id) {
    this.id = id;
}
public String getTitle() {
    return title;
}
public void setTitle(String title) {
    this.title = title;
}
public String getIsbn() {
    return isbn;
}
public void setIsbn(String isbn) {
    this.isbn = isbn;
}
public int getPrice() {
    return price;
}
public void setPrice(int price) {
    this.price = price;
}
public String getPages()
```

```

        return pages;
    }

    public void setPages(String pages) {
        this.pages = pages;
    }

    public String getImageUrl() {
        return imageUrl;
    }

    public void setImageUrl(String imageUrl) {
        this.imageUrl = imageUrl;
    }

}

```

- Tạo class **ProductRepository** trong package **model** kế thừa class **Repository** và viết lệnh như sau

```

public class ProductRepository extends Repository{
    public int count() throws SQLException {
        try {
            open();
            stmt = connection.createStatement();
            rs = stmt.executeQuery("SELECT COUNT(*) AS Total FROM Product");
            if (rs.next()) {
                return rs.getInt("Total");
            }
            return 0;
        }finally {
            close();
        }
    }

    public List<Product> getProducts(int index, int size) throws SQLException{
        try {
            open();
            pstmt = connection.prepareStatement("SELECT * FROM Product LIMIT ?, ?");
            pstmt.setInt(1, (index - 1) * size );

```

```

        pstmt.setInt(2, size);
        rs = pstmt.executeQuery();
        List<Product> list = new LinkedList<>();
        while(rs.next()) {
            Product obj = new Product(
                rs.getInt("ProductId"),
                rs.getString("Title"),
                rs.getString("ISBN"),
                rs.getInt("Price"),
                rs.getString("Pages"),
                rs.getString("ImageUrl"));
            list.add(obj);
        }
        return list;
    }finally {
        close();
    }
}
}

```

- Tạo Servlet **HomeController** trong package **controller** forward đến **/views/home/index.jsp** như sau

```

@WebServlet("/home.html")
public class HomeController extends GenericServlet {
    private int size = 8;
    ProductRepository repository = new ProductRepository();
    private int getPage(int total) {
        return (int) Math.ceil(total/(float)size);
    }
    @Override
    public void service(ServletRequest req, ServletResponse res) throws ServletException,
    IOException {
        int p = 1;
        if(req.getParameter("p") != null) {
            p = Integer.parseInt(req.getParameter("p"));
        }
        list = repository.getPage(p, size);
        RequestDispatcher dispatcher = req.getRequestDispatcher("index.jsp");
        dispatcher.forward(req, res);
    }
}

```

```

        }
        try {
            req.setAttribute("n", getPage(repository.count()));
            req.setAttribute("list", repository.getProducts(p, size));
            req.getRequestDispatcher("/views/home/index.jsp").forward(req, res);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

- Tạo trang jsp **/views/home/index.jsp** sử dụng template **bg**

```

<%@taglib prefix="me" uri="/WEB-INF/template" %>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<me:bg title="Mini Shop">
    <jsp:attribute name="content">
        <form class="form-search" method="get"
action="${pageContext.request.contextPath}/home/search.html">
            <input type="text" placeholder="Search..." name="q" >
            <button class="btn btn-primary">Search</button>
        </form>
        <div class="page-header">
            Products
        </div>
        <div class="products">
            <c:forEach var="o" items="${list}">
                <div class="col-3">
                    <div class="item">
                        
                        <div class="info">
                            <a
href="${pageContext.request.contextPath}/home/detail.html?id=${o.id}">${o.title}</a>
                        </div>
                    </div>
                </div>
            </c:forEach>
        </div>
    </jsp:attribute>
</me:bg>

```

```

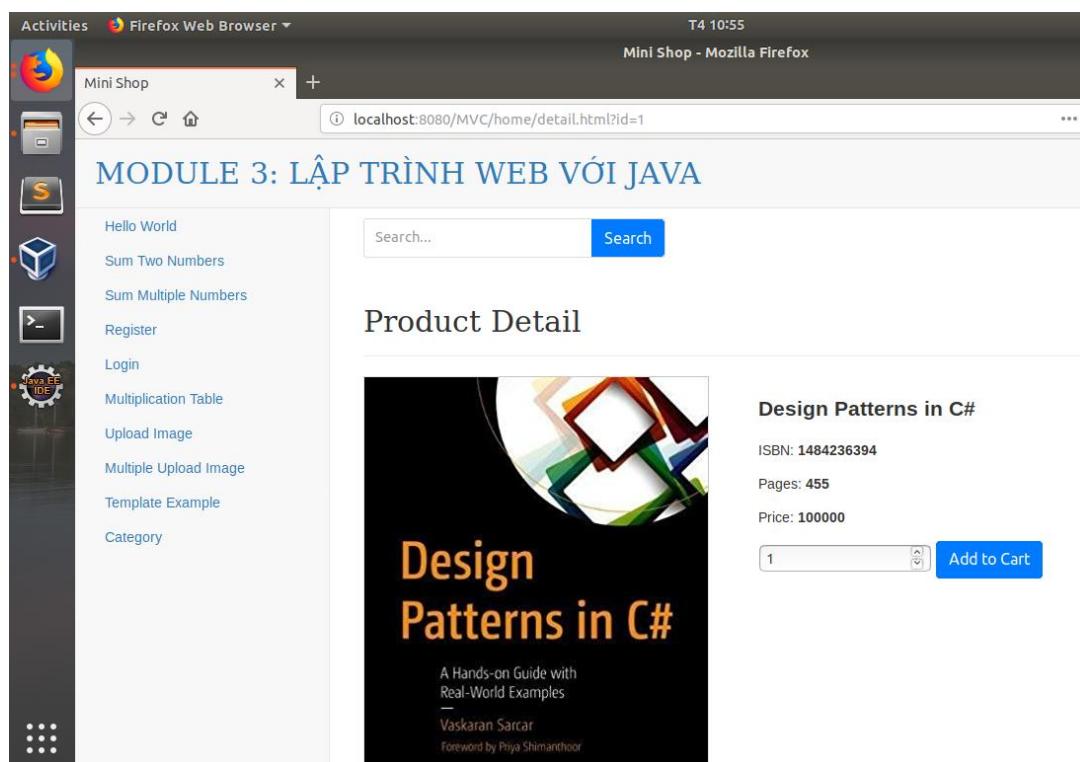
        </c:forEach>
        <div class="clear"></div>
    </div>
    <ul class="pagination">
        <c:forEach begin="1" end="${n}" step="1" var="i">
            <li class="page-item">
                <a class="page-link"
                href="${pageContext.request.contextPath}/home.html?p=${i}">${i}</a>
            </li>
        </c:forEach>
    </ul>
</jsp:attribute>
</me:bg>

```

- Truy cập link **home.html** để xem kết quả

4.4. home/detail.html

- **Yêu cầu:** Xây dựng trang xem chi tiết sản phẩm
 - Xử lý phần xem chi tiết sản phẩm khi khách hàng click vào sản phẩm trên trang **home.html**.



Trang chi tiết sản phẩm **detail.html**

- Hướng dẫn sử dụng:

- Người dùng chọn 1 sản phẩm từ trang **home.html** để xem chi tiết sản phẩm.

- Hướng dẫn:

- Thêm phương thức **getProduct** vào class **ProductRepository** trong model đã được xây dựng ở bài trước

```
public class ProductRepository extends Repository{
    public Product getProduct(int id) throws SQLException {
        try {
            open();
            pstmt = connection.prepareStatement("SELECT * FROM Product WHERE
ProductId = ?");
            pstmt.setInt(1, id);
            rs = pstmt.executeQuery();
            if(rs.next()) {
                return new Product(
                    rs.getInt("ProductId"),
                    rs.getString("Title"),
                    rs.getString("ISBN"),
                    rs.getInt("Price"),
                    rs.getString("Pages"),
                    rs.getString("ImageUrl"));
            }
            return null;
        }finally {
            close();
        }
    }
}
```

- Tạo Servlet **HomeDetailController** forward đến **/views/home/detail.jsp** như kết quả sau

```
@WebServlet("/home/detail.html")
public class HomeDetailController extends GenericServlet {
    ProductRepository repository = new ProductRepository();
```

```

@Override
public void service(ServletRequest request, ServletResponse response) throws
ServletException, IOException {
    try {
        int id = Integer.parseInt(request.getParameter("id"));
        request.setAttribute("o", repository.getProduct(id));
        request.getRequestDispatcher("/views/home/detail.jsp").forward(request,
response);
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}

```

- Tạo trang **/views/home/detail.jsp** với nội dung như sau.

```

<%@taglib prefix="me" uri="/WEB-INF/template" %>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<me:bg title="Mini Shop">
    <jsp:attribute name="content">
        <form class="form-search" method="get"
action="${pageContext.request.contextPath}/home/search.html">
            <input type="text" placeholder="Search..." name="q" >
            <button class="btn btn-primary">Search</button>
        </form>
        <div class="page-header">
            Product Detail
        </div>
        <div class="products">
            <div class="col-5">
                
            </div>
            <div class="col-7">
                <h2>${o.title}</h2>
                <p>ISBN: <b>${o.isbn}</b></p>
                <p>Pages: <b>${o.pages}</b></p>
            </div>
        </div>
    </jsp:attribute>
</me:bg>

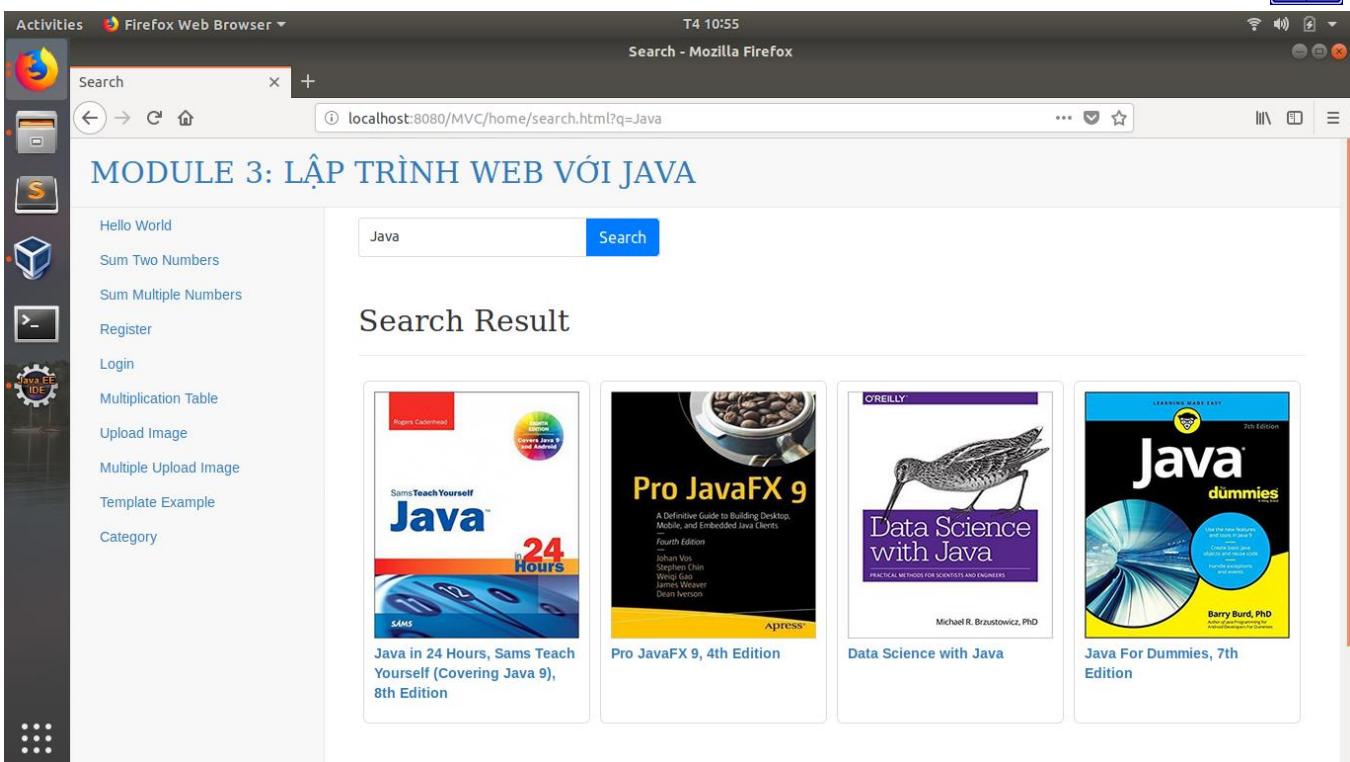
```

```
<p>Price: <b>${o.price}</b></p>
<form method="post"
action="${pageContext.request.contextPath}/cart.html">
    <input type="hidden" value="${o.id}" name="id" >
    <input type="number" name="qty" value="1">
    <button class="btn btn-primary">Add to Cart</button>
</form>
</div>
</div>
</jsp:attribute>
</me:bg>
```

- Quay về link **home.html** click vào link tiêu đề (home/detail.html) xem chi tiết sản phẩm

4.5. home/search.html Tạo trang tìm kiếm sản phẩm

- **Yêu cầu:** Thực hiện tìm kiếm cho form tìm kiếm đã thiết kế trong trang **home.html** và **detail.html**
- **Hướng dẫn sử dụng:**
 - Người dùng truy cập vào link **home.html** hoặc **detail.html** nhập từ khóa cần tìm và nhấn chọn **search**.
- **Tóm tắt yêu cầu:**
 - Thiết kế giao diện



- Qui tắc xử lý
- Từ trang **home.html** hoặc trang **detail.html** người dùng nhập từ khóa cần tìm sau đó chọn search sẽ chuyển qua trang tìm kiếm **search.html**
- Xử lý tìm kiếm với từ khóa vừa nhập

Hướng dẫn:

- Thêm phương thức **search** vào class **ProductRepository**

```
public class ProductRepository extends Repository{
    public List<Product> search(String q) throws SQLException{
        try {
            open();
            pstmt = connection.prepareStatement("SELECT * FROM Product WHERE
Title LIKE ?");
            pstmt.setString(1, "%" + q + "%");
            rs = pstmt.executeQuery();
            List<Product> list = new LinkedList<>();
            while(rs.next()) {
                Product obj = new Product(
                    rs.getInt("ProductId"),
                    rs.getString("Title"),
                    rs.getDouble("Price"),
                    rs.getString("Description"),
                    rs.getString("Image"),
                    rs.getString("Category")
                );
                list.add(obj);
            }
            return list;
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```

        rs.getString("ISBN"),
        rs.getInt("Price"),
        rs.getString("Pages"),
        rs.getString("ImageUrl"));

    list.add(obj);
}

return list;
}finally {
    close();
}
}

```

- Tạo Servelet **HomeSearchController** forward đến **/views/home/search.jsp**

```

@WebServlet("/home/search.html")
public class HomeSearchController extends HttpServlet {
    ProductRepository repository = new ProductRepository();
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    try {
        request.setAttribute("list", repository.search(request.getParameter("q")));
        request.getRequestDispatcher("/views/home/search.jsp").forward(request,
response);
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}

```

- Tạo trang **/views/home/search.jsp** với nội dung như sau

```

<%@taglib prefix="me" uri="/WEB-INF/template" %>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<me:bg title="Search">
    <jsp:attribute name="content">
        <form class="form-search" method="get"
action="${pageContext.request.contextPath}/home/search.html">

```

```

        <input type="text" placeholder="Search..." name="q" >
        <button class="btn btn-primary">Search</button>
    </form>
    <div class="page-header">
        Search Result
    </div>
    <div class="products">
        <c:forEach var="o" items="${list}">
            <div class="col-3">
                <div class="item">
                    
                    <div class="info">
                        <a href="${pageContext.request.contextPath}/home/detail.html?id=${o.id}">${o.title}</a>
                    </div>
                </div>
            </c:forEach>
            <div class="clear"></div>
        </div>
    </jsp:attribute>
</me:bg>

```

- Quay lại trang **home.html** để thực hiện việc tìm kiếm

4.6. Phân trang cho trang search.html(Bài làm thêm)

- **Yêu cầu:** Thực hiện việc xử lý phân trang cho trang search.html
- **Hướng dẫn sử dụng:**
- **Tóm tắt yêu cầu:**
 - Qui tắc xử lý
 - Tiếp tục sử dụng phần xử lý tìm kiếm của bài trước
 - Thực hiện xử lý phân trang trên kết quả tìm kiếm
- **Hướng dẫn:**

- Viết hàm count(String q) trong class ProductRepository đếm số lượng record với tham số truyền vào là từ khóa từ kiểm
- Viết lại hàm **search(String q, int index, int size)** trong class ProductRepository
- Tạo liên kết phân trang trong **/views/home/search.jsp** (xem lại phân trang trong trang home.html)

4.7. Tạo trang đăng ký thành viên

- **Yêu cầu:** Thực hiện chức năng đăng ký thành viên.
- **Tóm tắt yêu cầu:**
 - Thiết kế giao diện

Trang đăng ký thành viên auth/register.html

- Qui tắc xử lý
 - Tạo form để người dùng nhập thông tin Full Name, Username, Password, Email, Gender
 - Password phải được hash trước khi lưu vào database
 - MemberId sử dụng hàm random để tránh trường hợp truy vết thông tin
 - Xử lý lưu trữ thông tin đăng nhập đã được nhập vào
- **Hướng dẫn:**
 - Tạo Class **Member** trong package **model**

```
public class Member {
```

```
private long id;
private String username;
private String password;
private String email;
private String fullname;
private byte gender;
private Date addedDate;

public Member(long id, String username) {
    this(id, username, null, null, null, Byte.MIN_VALUE);
}

public Member(long id, String username, String password, String email, String fullname, byte
gender) {
    this(id, username, password, email, fullname, gender, null);
}

public Member(long id, String username, String password, String email, String fullname, byte
gender,
            Date addedDate) {
    this.id = id;
    this.username = username;
    this.password = password;
    this.email = email;
    this.fullname = fullname;
    this.gender = gender;
    this.addedDate = addedDate;
}

public long getId() {
    return id;
}

public void setId(long id) {
    this.id = id;
}

public String getUsername() {
    return username;
}

public void setUsername(String username) {
    this.username = username;
}
```

```
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getFullscreen() {
        return fullname;
    }
    public void setFullscreen(String fullname) {
        this.fullname = fullname;
    }
    public byte getGender() {
        return gender;
    }
    public void setGender(byte gender) {
        this.gender = gender;
    }
    public Date getAddedDate() {
        return addedDate;
    }
    public void setAddedDate(Date addedDate) {
        this.addedDate = addedDate;
    }
}
```

- Tạo class **MemberRepository** trong package **model**

```
public class MemberRepository extends Repository {
    private static byte[] sha256(String plaintext){
        try{
```

```

        MessageDigest digest = MessageDigest.getInstance("SHA-256");
        return digest.digest(plaintext.getBytes("UTF-8"));
    } catch (NoSuchAlgorithmException | UnsupportedEncodingException ex) {
        return null;
    }
}

private static long random() {
    Random rand = new Random();
    long a = rand.nextInt();
    long b = rand.nextInt();
    return a * b;
}

public int add(Member obj) throws SQLException {
    try {
        open();
        pstmt = connection.prepareStatement("INSERT INTO Member (MemberId,
Username, Password, FullName, Email, Gender) VALUES(?, ?, ?, ?, ?, ?)");
        pstmt.setLong(1, random());
        pstmt.setString(2, obj.getUsername());
        pstmt.setBytes(3, sha256(obj.getPassword()));
        pstmt.setString(4, obj.getFullname());
        pstmt.setString(5, obj.getEmail());
        pstmt.setByte(6, obj.getGender());
        return pstmt.executeUpdate();
    } finally {
        close();
    }
}
}

```

- Tạo servlet **AuthRegisterRepository** trong package controller forward đến **/views/auth/register.jsp**

```

@WebServlet("/auth/register.html")
public class AuthRegisterController extends HttpServlet {

```

```

MemberRepository repository = new MemberRepository();

protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    request.getRequestDispatcher("/views/auth/register.jsp").forward(request,
response);
}

protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
try {
    if(repository.add(new Member(0,
        request.getParameter("usr"),
        request.getParameter("pwd"),
        request.getParameter("email"),
        request.getParameter("fullName"),
        Byte.parseByte(request.getParameter("gender") ))) > 0) {
        response.sendRedirect(request.getContextPath() +
"/auth/logon.html");
    }else {
        request.setAttribute("msg", "Register Failed");
        doGet(request, response);
    }
} catch (NumberFormatException | SQLException e) {
    e.printStackTrace();
}
}
}

```

- Tạo trang **/views/auth/register.jsp** form nhập thông tin đăng ký người dùng

```

<%@taglib prefix="me" uri="/WEB-INF/template" %>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<me:bgpanel title="Register">
    <jsp:attribute name="content">
        <form method="post">
            <p>
                <input type="text" class="form-control" placeholder="Full Name"
name="fullName">

```

```
</p>
<p>
    <input type="email" class="form-control" placeholder="Email"
name="email">
</p>
<p>
    <input class="form-control" placeholder="Username" type="text"
name="usr">
</p>
<p>
    <input class="form-control" placeholder="Password"
type="password" name="pwd">
</p>
<p>
    <select name="gender">
        <option value="0">Male</option>
        <option value="1">Female</option>
        <option value="2">Undefined</option>
    </select>
</p>
<p>
    <button class="btn btn-primary btn-lg">Register</button>
</p>
<p class="center"><a
href="${pageContext.request.contextPath}/auth/logon.html">Log On</a></p>
<p class="center"><a
href="${pageContext.request.contextPath}/auth/forgot.html">Forgot Password?</a></p>
</form>
<c:if test="${not empty msg}">
    <div class="error">${msg}</div>
</c:if>
</jsp:attribute>
</me:bgpanel>
```

- Vào link **auth/register.html** để thực hiện kết quả

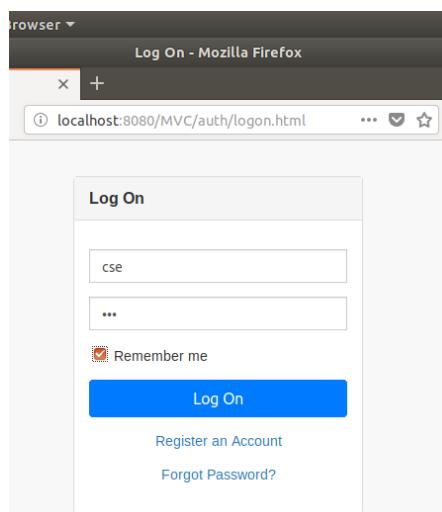
BÀI 5: Cookie, Session, Filter



- Sử dụng cookies, session để tạo trang đăng nhập
- Sử dụng filter để xác thực người dùng đã đăng nhập chưa

5.1. Tạo trang đăng nhập

- **Yêu cầu:** Viết trang đăng nhập vào hệ thống
- **Tóm tắt yêu cầu:**
 - Thiết kế giao diện:



- Quy tắc xử lý
 - Nhập vào Username và Password
 - Password phải được hash trước khi truy vấn database
 - Nếu kiểm tra thành công sẽ lưu MemberId, Username vào Session
 - Nếu người dùng chọn remember sẽ lưu SessionId vào cookie
- **Hướng dẫn:**
 - Viết thêm phương thức logOn thêm vào class **MemberRepository** trong package **model** đã tạo ra ở bài trước

```
public class MemberRepository extends Repository {
    public Member logOn(String username, String password) throws SQLException {
        try {
            open();
```

```

        pstmt = connection.prepareStatement("SELECT MemberId, Username
FROM Member WHERE Username = ? AND Password = ?");
        pstmt.setString(1, username);
        pstmt.setBytes(2, sha256(password));
        rs = pstmt.executeQuery();
        if (rs.next()) {
            return new Member(rs.getLong("MemberId"),
rs.getString("Username"));
        }
        return null;
    }finally {
        close();
    }
}
}

```

- Tạo servlet **AuthLogOncontroller** trong package **controller** forward đến /views/auth/logon.jsp

```

@WebServlet("/auth/logon.html")
public class AuthLogOnController extends HttpServlet {
    MemberRepository repository = new MemberRepository();
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        request.getRequestDispatcher("/views/auth/logon.jsp").forward(request,
response);
    }
    private static void savedCookie(HttpServletRequest request, HttpServletResponse
response) {
        for (Cookie cookie : request.getCookies()) {
            if (cookie.getName().equals("JSESSIONID")) {
                cookie.setMaxAge(30 * 24 * 3600);
                cookie.setPath(request.getContextPath());
                response.addCookie(cookie);
            }
        }
    }
    private static void savedSession(Member obj, HttpServletRequest request,

```

```

HttpServletResponse response) {
    HttpSession session = request.getSession();
    session.setAttribute("member", obj);
    if (request.getParameter("remember") != null) {
        savedCookie(request, response);
    }
}

protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    try {
        Member ret = repository.logOn(request.getParameter("usr"),
request.getParameter("pwd"));
        if(ret != null) {
            savedSession(ret, request, response);
            response.sendRedirect(request.getContextPath() + "/home.html");
        }else {
            request.setAttribute("msg", "Log On Failed");
            doGet(request, response);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}

```

- Tạo **/views/auth/logon.jsp** thiết kế form đăng nhập

```

<%@taglib prefix="me" uri="/WEB-INF/template" %>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<me:bgpanel title="Log On">
    <jsp:attribute name="content">
        <form method="post">
            <p>
                <input class="form-control" placeholder="Username" type="text"
name="usr">
            </p>
            <p>

```

```

        <input class="form-control" placeholder="Password"
type="password" name="pwd">
    </p>
    <p>
        <label>
            <input type="checkbox" name="remember" value="1">
Remember me
        </label>
    </p>
    <p>
        <button class="btn btn-primary btn-lg">Log On</button>
    </p>
    <p class="center"><a
href="${pageContext.request.contextPath}/auth/register.html">Register an
Account</a></p>
    <p class="center"><a
href="${pageContext.request.contextPath}/auth/forgot.html">Forgot Password?</a></p>
    </form>
    <c:if test="${not empty msg}">
        <div class="error">${msg}</div>
    </c:if>
</jsp:attribute>
</me:bgpanel>

```

- Truy cập link **auth/logon.html** để xem kết quả

5.2. Tạo Filter authentication

- **Yêu cầu:** Tạo filter để xác thực đã đăng nhập vào hệ thống web site
- **Tóm tắt yêu cầu:**
 - Quy tắc xử lý
 - Kế thừa interface **Filter**
 - Dựa trên urlPattern để xác thực đăng nhập
 - Kiểm tra Session LogIn nếu không tồn tại thì chuyển hướng đến trang LogOn.html
- **Hướng dẫn:**
 - Tạo package **filter**

- Trong package filter tạo class với tên là AuthFilter implement Filter

```
@WebFilter(urlPatterns = "/admin/*")
public class AuthFilter implements Filter {
    public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws IOException, ServletException {
        HttpServletRequest req = (HttpServletRequest)request;
        HttpServletResponse res = (HttpServletResponse)response;
        HttpSession session = req.getSession();
        if(session.getAttribute("member") == null) {
            res.sendRedirect(req.getContextPath() + "/auth/logon.html");
        }else {
            chain.doFilter(request, response);
        }
    }
}
```

- Truy cập vào link **admin/publisher.html** để kiểm tra kết quả

5.3. Giỏ hàng

- **Yêu cầu:** Tạo trang lưu thông tin giỏ hàng
- **Hướng dẫn sử dụng:**
 - Người dùng truy cập vào trang **detail.html**, nhấn chọn “Add to Cart”
- **Tóm tắt yêu cầu:**
 - Thiết kế giao diện

Title	Quantity	Price	Image	Delete
Pro C# 7-8th Edition	2	100000		

Trang thông tin giỏ hàng

- Qui tắc xử lý
 - Thông tin giỏ hàng được lưu trữ vào session
 - Khi người dùng click chọn sản phẩm có trong giỏ hàng thì phải kiểm tra nếu mã sản phẩm này đã tồn tại trong giỏ hàng thì không thêm vào giỏ hàng mà chỉ tăng số lượng sản phẩm đó
- **Hướng dẫn:**
 - Tạo class **Cart** trong package **model**

```
package model;
public class Cart {
    private int productId;
    private int price;
    private short quantity;
    private String imageUrl;
    private String title;
    public Cart(int productId, int price, short quantity, String imageUrl, String title) {
        this.productId = productId;
        this.price = price;
        this.quantity = quantity;
        this.imageUrl = imageUrl;
        this.title = title;
    }
    public int getProductId() {
        return productId;
    }
    public void setProductId(int productId) {
        this.productId = productId;
    }
    public int getPrice() {
        return price;
    }
    public void setPrice(int price) {
        this.price = price;
    }
}
```

```

public short getQuantity() {
    return quantity;
}

public void increaseQuantity(short quantity) {
    this.quantity += quantity;
}

public void setQuantity(short quantity) {
    this.quantity = quantity;
}

public String getImageUrl() {
    return imageUrl;
}

public void setImageUrl(String imageUrl) {
    this.imageUrl = imageUrl;
}

public String getTitle() {
    return title;
}

public void setTitle(String title) {
    this.title = title;
}

}

```

- Tạo Servlet **CartController** forward tới trang /views/cart/index.jsp

```

@WebServlet("/cart.html")
public class CartController extends HttpServlet {
    ProductRepository repository = new ProductRepository();
    private static Map<Integer, Cart> getCarts(HttpServletRequest request,
HttpServletResponse response){
        HttpSession session = request.getSession();
        Object obj = session.getAttribute("cart");
        Map<Integer, Cart> carts = null;
        if(obj != null) {
            carts = (Map<Integer, Cart>)obj;
        }else {

```

```
        carts = new HashMap();
        session.setAttribute("cart", carts);
    }
    return carts;
}

private void saveCarts(Map<Integer, Cart> carts, HttpServletRequest request,
HttpServletResponse response) {
    HttpSession session = request.getSession();
    session.setAttribute("cart", carts);
}

protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    request.setAttribute("list", getCart(request, response).values());
    request.getRequestDispatcher("/views/cart/index.jsp").forward(request,
response);
}

protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    try {
        Map<Integer, Cart> carts = getCart(request, response);
        int id = Integer.parseInt(request.getParameter("id"));
        short qty = Short.parseShort(request.getParameter("qty"));
        if (carts.containsKey(id)) {
            carts.get(id).increaseQuantity(qty);
        }else {
            Product obj = repository.getProduct(id);
            Cart cart = new Cart(
                obj.getId(),
                obj.getPrice(),
                qty,
                obj.getImageUrl(),
                obj.getTitle());
            carts.put(id, cart);
        }
    }
}
```

```

        saveCarts(carts, request, response);
    } catch (SQLException e) {
        e.printStackTrace();
    }
    doGet(request, response);
}
}

```

- Tạo trang **/views/cart/index.jsp** thiết kế màn hình hiển thị thông tin có trong giỏ hàng

```

<me:bg title="Cart">
    <jsp:attribute name="content">
        <div class="page-header">
            <div>Your Cart</div>
        </div>
        <table class="table table-bordered">

            <tr><th>Title</th><th>Quantity</th><th>Price</th><th>Image</th><th>Delete</th></tr>
            <c:forEach items="#${list}" var="o">
                <tr>
                    <td>${o.title}</td>
                    <td><input type="number" value="#${o.quantity}" name="qty" class="qty"></td>
                    <td>${o.price}</td>
                    <td></td>
                    <td></td>
                </tr>
            </c:forEach>
        </table>
        <p><a href="#${pageContext.request.contextPath}/cart/checkout.html" class="btn btn-primary">Check Out</a></p>
    </jsp:attribute>
</me:bg>

```

- Quay về trang **detail.html** click chọn “Add to Cart” để kiểm tra kết quả

5.4. Checkout

- **Yêu cầu:** Tạo trang lưu thông tin đặt hàng
- **Tóm tắt yêu cầu:**
 - Thiết kế giao diện

The screenshot shows a Firefox browser window with the title "Check out - Mozilla Firefox". The address bar shows "localhost:8080/MVC/cart/checkout.html". The main content area displays a "Check Out" form with fields for Email (admin@cse.net.vn), Tel (0933112233), and Address (hcm). Below the form is a table representing the shopping cart. The table has columns: Title, Quantity, Price, Image, and Delete. One item is listed: "Pro C# 7-8th Edition" with a quantity of 2, a price of 100000, and a thumbnail image. A blue "Order" button is at the bottom of the form.

*Trang đặt hàng **cart/checkout.html***

- **Quy tắc xử lý**
 - Yêu cầu người dùng nhập thông tin **email, tel, address**
 - Click chọn **Order** để tiến hành đặt hàng
 - Sau khi đặt hàng thành công sẽ chuyển qua trang **order/detail.html** để xem kết quả đặt hàng thành công và theo dõi tình trạng đơn hàng
- **Hướng dẫn:**
 - Tạo class **Invoice** trong package **model**

```
public class Invoice {  
    private Long id;  
    private Long memberId;  
    private String tel;  
    private String address;  
    private String email;  
    private Date date;  
    private byte statusId;  
    private String status;  
    private List<InvoiceDetail> details;  
    public Invoice(Long id, Long memberId, String tel, String address, String email) {  
        this(id, memberId, tel, address, email, Byte.MIN_VALUE, null, null);  
    }  
    public Invoice(Long id, Long memberId, String tel, String address, String email, byte  
statusId,  
        String status, Date date) {  
        this.id = id;  
        this.memberId = memberId;  
        this.tel = tel;  
        this.address = address;  
        this.email = email;  
        this.statusId = statusId;  
        this.status = status;  
        this.date = date;  
    }  
    public Long getId() {  
        return id;  
    }  
    public void setId(Long id) {  
        this.id = id;  
    }  
    public Long getMemberId() {  
        return memberId;  
    }  
    public void setMemberId(Long memberId) {
```

```
        this.memberId = memberId;
    }
    public Date getDate() {
        return date;
    }
    public void setDate(Date date) {
        this.date = date;
    }
    public String getTel() {
        return tel;
    }
    public void setTel(String tel) {
        this.tel = tel;
    }
    public String getAddress() {
        return address;
    }
    public void setAddress(String address) {
        this.address = address;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public byte getStatusId() {
        return statusId;
    }
    public void setStatusId(byte statusId) {
        this.statusId = statusId;
    }
    public String getStatus() {
        return status;
    }
}
```

```
public void setStatus(String status) {  
    this.status = status;  
}  
public List<InvoiceDetail> getDetails() {  
    return details;  
}  
public void setDetails(List<InvoiceDetail> details) {  
    this.details = details;  
}  
}
```

- Tạo tiếp class **InvoiceDetail** trong package model

```
public class InvoiceDetail {  
    private long id;  
    private int productId;  
    private short quantity;  
    private int price;  
    private String title;  
    private String imageUrl;  
    public InvoiceDetail(long id, int productId, short quantity, int price) {  
        this(id, productId, quantity, price, null, null);  
    }  
    public InvoiceDetail(long id, int productId, short quantity, int price, String title, String  
imageUrl) {  
        this.id = id;  
        this.productId = productId;  
        this.quantity = quantity;  
        this.price = price;  
        this.title = title;  
        this.imageUrl = imageUrl;  
    }  
    public long getId() {  
        return id;  
    }  
    public void setId(long id) {
```

```
        this.id = id;
    }
    public int getId() {
        return productId;
    }
    public void setId(int productId) {
        this.productId = productId;
    }
    public short getQuantity() {
        return quantity;
    }
    public void setQuantity(short quantity) {
        this.quantity = quantity;
    }
    public int getPrice() {
        return price;
    }
    public void setPrice(int price) {
        this.price = price;
    }
    public String getTitle() {
        return title;
    }
    public void setTitle(String title) {
        this.title = title;
    }
    public String getImageUrl() {
        return imageUrl;
    }
    public void setImageUrl(String imageUrl) {
        this.imageUrl = imageUrl;
    }
}
```

- Tạo class **InvoiceRepository** kế thừa class Repository trong package model

```
public class InvoiceRepository extends Repository{  
    private int[] add(List<InvoiceDetail> list) throws SQLException {  
        pstmt = connection.prepareStatement("INSERT INTO InvoiceDetail (InvoiceId,  
ProductId, Quantity, Price) VALUES(?, ?, ?, ?)");  
        for (InvoiceDetail detail : list) {  
            pstmt.setLong(1, detail.getId());  
            pstmt.setInt(2, detail.getProductId());  
            pstmt.setShort(3, detail.getQuantity());  
            pstmt.setInt(4, detail.getPrice());  
            pstmt.addBatch();  
        }  
        return pstmt.executeBatch();  
    }  
    public int add(Invoice obj) throws SQLException {  
        try {  
            open();  
            pstmt = connection.prepareStatement("INSERT INTO Invoice (InvoiceId,  
MemberId, Email, Tel, Address) VALUES(?, ?, ?, ?, ?)");  
            pstmt.setLong(1, obj.getId());  
            pstmt.setObject(2, obj.getMemberId());  
            pstmt.setString(3, obj.getEmail());  
            pstmt.setString(4, obj.getTel());  
            pstmt.setString(5, obj.getAddress());  
            int ret = pstmt.executeUpdate();  
            pstmt.close();  
            add(obj.getDetails());  
            return ret;  
        }finally {  
            close();  
        }  
    }  
}
```

- Tạo servlet CartCheckoutController trong package controller forward đến **/views/cart/checkout.jsp**

```
@WebServlet("/cart/checkout.html")
public class CartCheckoutController extends HttpServlet {
    private static long random() {
        Random rand = new Random();
        return Math.abs(rand.nextLong());
    }
    InvoiceRepository repository = new InvoiceRepository();
    private static Map<Integer, Cart> getCart(HttpServletRequest request,
HttpServletResponse response) {
        HttpSession session = request.getSession();
        Object obj = session.getAttribute("cart");
        Map<Integer, Cart> carts = null;
        if(obj != null) {
            carts = (Map<Integer, Cart>)obj;
        }else {
            carts = new HashMap();
            session.setAttribute("cart", carts);
        }
        return carts;
    }
    private static Member getMember(HttpServletRequest request, HttpServletResponse response) {
        HttpSession session = request.getSession();
        Object obj = session.getAttribute("member");
        if(obj != null) {
            return (Member)obj;
        }else {
            return null;
        }
    }
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        request.setAttribute("list", getCart(request, response).values());
        request.getRequestDispatcher("/views/cart/checkout.jsp").forward(request,
response);
    }
}
```

```
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    try {
        Invoice obj = new Invoice(random(), null, request.getParameter("tel"),
request.getParameter("address"), request.getParameter("email"));
        Member member = getMember(request, response);
        if(member != null) {
            obj.setMemberId(member.getId());
        }
        List<InvoiceDetail> details = new LinkedList<>();
        for (Cart cart : getCart(request, response).values()) {
            details.add(new InvoiceDetail(obj.getId(), cart.getProductId(),
cart.getQuantity(), cart.getPrice()));
        }
        obj.setDetails(details);
        if(repository.add(obj) > 0){
            response.sendRedirect(request.getContextPath() +
"/order/detail.html?id=" + obj.getId());
        }else {
            request.setAttribute("msg", "Ordered Failed");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

- Tạo trang **/views/cart/checkout.jsp** thiết kế giao diện cho trang **checkout.html**

```
<me:bg title="Check out">
    <jsp:attribute name="content">
        <div class="page-header">
            <div>Check Out</div>
        </div>
        <form method="post" class="form">
            <p>
                <label>Email</label>
                <input placeholder="Email" value="npthanhthai@yahoo.com.vn" type="email" name="email">
            </p>
            <p>
                <label>Tel</label>
                <input type="tel" name="tel" value="1234234" placeholder="Tel">
            </p>
            <p>
                <label>Address</label>
                <input type="text" name="address" value="hcm" placeholder="Address">
            </p>
            <table class="table table-bordered">
                <tr><th>Title</th><th>Quantity</th><th>Price</th><th>Image</th><th>Delete</th></tr>
                <c:forEach items="${list}" var="o">
                    <tr>
                        <td>${o.title}</td>
                        <td><input type="number" value="${o.quantity}" class="qty"></td>
                        <td>${o.price}</td>
                        <td></td>
                        <td></td>
                    </tr>
                </c:forEach>
            </table>
        </form>
    </jsp:attribute>
</me:bg>
```

```

        </table>
        <p><button class="btn btn-primary">Order</button>
    </form>
</jsp:attribute>
</me:bg>

```

- Quay lại link **cart/index.html** tiến hành checkout.

5.5. Chi tiết đơn hàng (Bài làm thêm)

- **Yêu cầu:** Tạo xem chi tiết đơn hàng
- **Tóm tắt yêu cầu:**
 - Xử lý tiếp qui trình đặt hàng sau khi tiến hàng checkout sẽ chuyển đến trang cart/detail.html
- **Hướng dẫn cài đặt**
 - Lấy thông tin mã đơn hàng từ url
 - Truy vấn lấy ra thông tin đơn hàng hiển thị trang chi tiết đơn hàng

5.6. Xem lịch sử đơn đặt hàng (Bài làm thêm)

- **Yêu cầu:** Tạo xem tất cả các lịch sử mà khách hàng đã đặt
- **Tóm tắt yêu cầu:**
 - Xử lý tiếp qui trình đặt hàng sau khi tiến hàng đơn hàng
- **Hướng dẫn cài đặt**
 - Lấy thông tin MemberId khi người dùng đăng nhập vào
 - Truy vấn lấy ra thông tin đơn hàng hiển thị tất cả các đơn hàng khách hàng đó đã thực hiện

BÀI 6: Hibernate



- Xử lý kết nối cơ sở dữ liệu với thư viện Hibernate

6.1. CRUD Author by Hibernate

- **Yêu cầu:** Thực hiện các thao tác Select, Insert, Update, Delete trên bảng Author bằng Hibernate
- **Tóm tắt yêu cầu:**
 - Thiết kế giao diện

		Name	Edit	Delete
1	Vaskaran Sarcar			
2	Andrew Troelsen			
3	Rogers Cadenhead			

Kết quả trang hiển thị danh sách Author từ link /admin/author.html

Kết quả truy cập trang /admin/author/add.html

Kết quả từ trang /admin/author/edit.html

- Qui tắc xử lý: Sử dụng thư viện hibernate để truy vấn lấy dữ liệu từ database
- **Hướng dẫn:**

- Import thư viện hibernate 5.3 download từ link <http://hibernate.org/orm/releases/>
- Tạo tập tin hibernate.cfg.xml trong src với nội dung cấu hình như sau

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD
3.0//EN" "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
        <property
            name="hibernate.connection.url">jdbc:mysql://localhost:3306/MiniShop?zeroDateTimeBehavior=convertToNull</property>
        <property name="hibernate.connection.username">cse</property>
        <property name="hibernate.connection.password">123</property>
        <mapping class="model.Author"/>
    </session-factory>
</hibernate-configuration>
```

- Tạo class **HibernateUtil** trong package **model** với nội dung như sau

```
public class HibernateUtil {
    private static final SessionFactory sessionFactory;
    static {
```

```
        sessionFactory = new Configuration().configure().buildSessionFactory();
    }
    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }
}
```

- Tạo class **Author** trong package **model** với nội dung như sau, lưu ý các dòng Anotation

```
@Entity
@Table(name="Author")
public class Author {
    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column(name="AuthorId")
    private Integer id;
    @Column(name="AuthorName")
    private String name;
    public Author() {
    }
    public Author(Integer id, String name) {
        this.id = id;
        this.name = name;
    }
    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;
    }
}
```

```
}
```

- Tạo class **AuthorRepository** viết method **getAuthors()** trong package model với nội dung như sau

```
public class AuthorRepository {
    public List<Author> getAuthors(){
        try(Session session = HibernateUtil.getSessionfactory().openSession()){
            return session.createQuery("From Author").list();
        }
    }
}
```

- Tạo servlet **AuthorController** trong package **controller** forward đến /views/author/index.jsp

```
@WebServlet("/admin/author.html")
public class AuthorController extends HttpServlet {
    AuthorRepository repository = new AuthorRepository();
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        request.setAttribute("list", repository.getAuthors());
        request.getRequestDispatcher("/views/author/index.jsp").forward(request,
response);
    }
}
```

- Tạo /views/author/index.jsp thiết kế giao diện cho trang /admin/author/html

```
<me:bg title="Manager Author">
    <jsp:attribute name="content">
        <form class="form-search" method="get"
action="${pageContext.request.contextPath}/author/search.html">
            <input type="search" name="q" placeholder="Searching...">
            <button class="btn btn-second">Search</button>
        </form>
        <div class="page-header">
            Manage Author
        </div>
    </jsp:attribute>
</me:bg>
```

```
</div>
<p>
    <a class="btn btn-info"
    href="#">
        <td>${o.id}</td>
        <td>${o.name}</td>
        <td>
            <a href="#">![Edit](#)
            </a>
        </td>
        <td>
            <a href="#">![Delete](#)
            </a>
        </td>
    
```

- Truy cập link /admin/author.html để xem kết quả
- Cài đặt cho trang **/admin/author/add.html**
- Thêm phương thức **add** vào class **AuthorRepository** với nội dung như sau

```
public class AuthorRepository {
    public boolean add(Author obj) {
        Transaction tran = null;
        try(Session session = HibernateUtil.getSessionfactory().openSession()){
            tran = session.beginTransaction();
            session.save(obj);
            tran.commit();
            return true;
        }catch (Exception e) {
            if(tran != null) {
                tran.rollback();
            }
            return false;
        }
    }
}
```

- Tạo servlet **AuthorAddController** forward đến **/views/author/add.jsp** với nội dung như sau

```
@WebServlet("/author/add.html")
public class AuthorAddController extends HttpServlet {
    AuthorRepository repository = new AuthorRepository();
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        request.getRequestDispatcher("/views/author/add.jsp").forward(request,
response);
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        Author obj = new Author(null, request.getParameter("name"));
        repository.add(obj);
    }
}
```

```

        response.sendRedirect(request.getContextPath() + "/admin/author.html");
    }
}

```

- Tạo **/views/author/add.jsp** với nội dung như sau

```

<me:bg title="Add Author">
    <jsp:attribute name="content">
        <div class="page-header">
            Add Author
        </div>
        <p><a class="btn btn-info"
href="#">

```

- Truy cập link **/admin/author/add.html**
- Cài đặt cho trang **/admin/author/edit.html**
- Thêm phương thức **edit** vào class **AuthorRepository** với nội dung như sau

```

public class AuthorRepository {
    public boolean edit(Author obj) {
        Transaction tran = null;
        try(Session session = HibernateUtil.getSessionfactory().openSession()){
            tran = session.beginTransaction();
            session.update(obj);
            tran.commit();
        }
    }
}

```

```
        return true;  
    }catch (Exception e) {  
        if(tran != null) {  
            tran.rollback();  
        }  
        return false;  
    }  
}
```

- Tạo servlet **AuthorEditController** forward đến **/views/author/edit.jsp** với nội dung như sau

```
@WebServlet("/admin/author/edit.html")
public class AuthorEditController extends HttpServlet {
    AuthorRepository repository = new AuthorRepository();

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        int id = Integer.parseInt(request.getParameter("id"));
        request.setAttribute("o", repository.getAuthor(id));
        request.getRequestDispatcher("/views/author/edit.jsp").forward(request,
response);
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        int id = Integer.parseInt(request.getParameter("id"));
        Author obj = new Author(id, request.getParameter("name"));
        repository.edit(obj);
        response.sendRedirect(request.getContextPath() + "/admin/author.html");
    }
}
```

- Tao **/views/author/edit.jsp** với nội dung như sau

```
<me:bg title="Edit Author">  
    <jsp:attribute name="content">  
        <div class="page-header">
```

```

Edit author

</div>
<p><a class="btn btn-info"
href="#">

```

- Quay về trang /admin/author.html click chọn cập link /admin/author/edit.html từ icon edit
- Cài đặt cho trang **/admin/author/del.html**
- Thêm phương thức **delete** vào class **AuthorRepository** với nội dung như sau

```

public class AuthorRepository {
    public boolean delete(int id) {
        Transaction tran = null;
        try(Session session = HibernateUtil.getSessionfactory().openSession()){
            tran = session.beginTransaction();
            session.delete(new Author(id, null));
            tran.commit();
            return true;
        }catch(Exception ex){
            if(tran != null) {
                tran.rollback();
            }
            return false;
        }
    }
}

```

```

        }
    }
}
```

- Tạo servlet **AuthorDeleteController**

```

@WebServlet("/admin/author/del.html")
public class AuthorDelController extends HttpServlet {
    AuthorRepository repository = new AuthorRepository();
    @Override
    protected void service(HttpServletRequest req, HttpServletResponse resp) throws
    ServletException, IOException {
        int id = Integer.parseInt(req.getParameter("id"));
        repository.delete(id);
        resp.sendRedirect(req.getContextPath() + "/admin/author.html");
    }
}
```

- Quay về trang /admin/author.html click chọn cập link /admin/author/del.html từ icon delete
- Triển khai

6.2. Search Author by Hibernate(Bài làm thêm)

- **Yêu cầu:** Viết trang tìm kiếm author
- **Hướng dẫn sử dụng:**
 - Truy cập link admin/author.html nhập từ khóa tìm kiếm
 - Xem kết quả tìm kiếm trên trang admin/author/search.html
- **Tóm tắt yêu cầu:**
 - Thiết kế giao diện

MODULE 3: LẬP TRÌNH WEB VỚI JAVA

Hello World

Sum Two Numbers

Sum Multiple Numbers

Register

Login

Multiplication Table

Upload Image

Multiple Upload Image

Template Example

Category

Search Result Author

Id	Name	Edit	Delete
1	Vaskaran Sarcar		
14	Phạm Văn Ất		
15	Pham Van Hung		

- Qui tắc xử lý:

- Từ giao diện /admin/author.html nhập từ khóa tìm kiếm action đến /admin/author/search.html bằng phương thức GET
- Lấy thông tin tìm kiếm từ phương thức get viết hàm truy cập đến database rồi hiển thị kết quả lên trang /admin/author/search.html

- **Hướng dẫn:**

- Thêm phương thức search vào class AuthorRepository được viết như sau

```
public class AuthorRepository {
    public List<Author> search(String q){
        try(Session session = HibernateUtil.getSessionfactory().openSession()){
            final Query query = session.createQuery("From Author WHERE
AuthorName LIKE :q");
            query.setParameter("q", '%' + q + '%');
            return query.list();
        }
    }
}
```

- Các thao tác khác học viên tự viết thêm

6.3. Multiple Delete Author by Hibernate (Tự tìm hiểu)

- **Yêu cầu:** Viết xử lý cho người dùng check chọn dòng dữ liệu trên danh sách Author sau đó chọn delete (Xem lại cách viết trong table Publisher)

BÀI 7: Custom Tag



Sử dụng custom tag nhằm tái sử dụng code trong views

7.1. Custom Tag Pagination

Yêu cầu: Sử dụng custom tag tạo tag phân trang cho ứng dụng:

- **Tóm tắt yêu cầu:**

- Thiết kế giao diện

MODULE 3: LẬP TRÌNH WEB VỚI JAVA

Hello World Sum Two Numbers Sum Multiple Numbers Register Login Multiplication Table Upload Image Multiple Upload Image Template Example Category	<p>Patterns in C# A Hands-on Guide with Real-World Examples Viktoran Sarcar Forward by Praya Stamatov</p>	<p>Pro C# 7-8th Edition With .NET and JET Core Eighth Edition Andreas Tackmann Philip Japikse</p>	<p>Java in 24 Hours, Sams Teach Yourself (Covering Java 9), 8th Edition 24 HOURS SAMS</p>	<p>Pro JavaFX 9, 4th Edition A Definitive Guide to Building Desktop, Mobile, and Embedded Java Clients Fourth Edition Johan Von Strijp-Chin Wenqi Gao James Weaver Dean Iverson</p>
Data Science with Java <p>Data Science with Java PRACTICAL METHODS FOR SCIENTISTS AND ENGINEERS Michael R. Brzustowicz, PhD</p>	<p>Advanced Data Analytics Using Python With Machine Learning, Deep Learning and NLP Examples Sayan Mukhopadhyay</p>	<p>Java For Dummies, 7th Edition LEARNING MADE EASY Barry Burd, PhD</p>	<p>Beginning SQL Queries From Novice to Professional Apply the right operations to the right problem to generate the right results, every time Second Edition Clare Churcher</p>	
1 2 3				

- Qui tắc xử lý
- Kế thừa từ class **SimpleTagSupport**

- **Hướng dẫn:**

- Tạo package **form**
- Tạo class **Pagination** trong package **form** kế thừa class **SimpleTagSupport** với nội dung như sau

```
public class Pagination extends SimpleTagSupport {
    private int size;
    private String url;
```

```

public void setSize(int size) {
    this.size = size;
}
public void setUrl(String url) {
    this.url = url;
}
@Override
public void doTag() throws JspException, IOException {
    PageContext pageContext = (PageContext) getJspContext();
    int p = 1;
    if(pageContext.getRequest().getParameter("p") != null) {
        p = Integer.parseInt(pageContext.getRequest().getParameter("p"));
    }
    JspWriter jw = getJspContext().getOut();
    jw.write("<ul class=\"pagination\">");
    for(int i = 1; i <= size; i++) {
        String href = String.format(url, i);
        if(p == i) {
            jw.write(String.format("<li class=\"page-item active\"><a class=\"page-link\" href=\"%s\">%d</a></li>", href, i));
        } else {
            jw.write(String.format("<li class=\"page-item\"><a class=\"page-link\" href=\"%s\">%d</a></li>", href, i));
        }
    }
    jw.write("</ul>");
}
}

```

- Tạo /WEB-INF/form.tld với nội dung như sau

```

<?xml version="1.0" encoding="UTF-8"?>

<taglib version="2.1" xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-jsptaglibrary_2_1.xsd">

```

```

<tlib-version>1.0</tlib-version>

<short-name>form</short-name>

<uri>/WEB-INF/form</uri>

<tag>

    <name>pagination</name>

    <tag-class>form.Pagination</tag-class>

    <body-content>empty</body-content>

    <attribute>

        <name>url</name>

        <required>true</required>

        <rtpvalue>true</rtpvalue>

    </attribute>

    <attribute>

        <name>size</name>

        <required>true</required>

        <rtpvalue>true</rtpvalue>

    </attribute>

</tag>

</taglib>

```

- Trong /views/home/index.jsp thêm nội dung vào phần hiển thị phân trang

```
<frm:pagination url="${pageContext.request.contextPath}/home.html?p=%s" size="${n}" />
```

- Truy cập trang /home.html xem kết quả mong đợi

7.2. Custom Tag Select (Tự thực hiện)

- **Yêu cầu:** Sử dụng Custom Tag Tạo tag Select
- **Tóm tắt yêu cầu:**
- **Cách dùng:**

```
<frm:select class="form-control" name="category" list="${category}" />
```

- **Quy tắc xử lý:** Tương tự bài 7.1

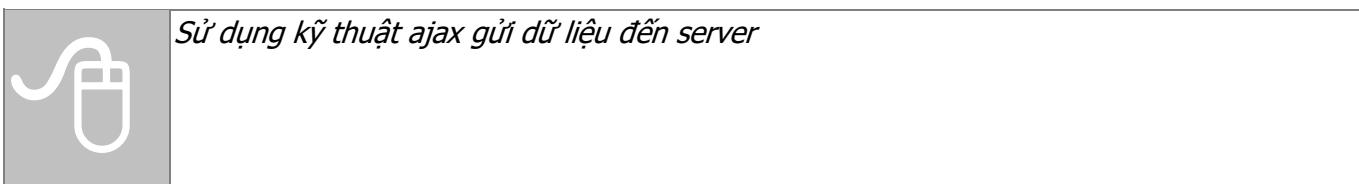
7.3. Custom Tag Menu Multiple Level (Tự thực hiện)

- **Yêu cầu:** Sử dụng custom tag tạo cây danh mục đa cấp
- **Tóm tắt yêu cầu:**
 - **Cách dùng:**

```
<frm:menu class="menu" list="${list}" />
```

- **Quy tắc xử lý:** tương tự bài 7.1

BÀI 8: Ajax



8.1. Update quantity in Shopping cart

- **Yêu cầu:** Cập nhật thay đổi số lượng trong giỏ hàng
- **Hướng dẫn sử dụng:**
 - Người sử dụng vào trang giỏ hàng, click chọn thay đổi số lượng sẽ được cập nhật tự động vào giỏ hàng mà không cần phải click chọn bất kỳ button nào
- **Tóm tắt yêu cầu:**
 - Thiết kế giao diện

Title	Quantity	Price	Image	Delete
Design Patterns in C#	3	100000		
Java in 24 Hours, Sams Teach Yourself (Covering Java 9), 8th Edition	2	100000		
Pro JavaFX 9, 4th Edition	2	100000		

- Qui tắc xử lý
 - Bắt sự kiện onchange khi người dùng thay đổi số lượng trên input number
 - Sử dụng ajax send dữ liệu về server
 - Tại server lấy dữ liệu thay đổi dữ liệu đang lưu
- **Hướng dẫn:**

- Tạo servlet **CartEditController** với nội dung như sau

```

@WebServlet("/cart/edit.html")
public class CartEditController extends HttpServlet {
    private static Map<Integer, Cart> getCart(HttpServletRequest request,
HttpServletResponse response){
        HttpSession session = request.getSession();
        Object obj = session.getAttribute("cart");
        Map<Integer, Cart> carts = null;
        if(obj != null) {
            carts = (Map<Integer, Cart>)obj;
        }else {
            carts = new HashMap();
            session.setAttribute("cart", carts);
        }
        return carts;
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        try(PrintWriter pw = response.getWriter()){
            short qty = Short.parseShort(request.getParameter("qty"));
            int key = Integer.parseInt(request.getParameter("id"));
            Map<Integer, Cart> carts = getCart(request, response);
            if(carts.containsKey(key)) {
                carts.get(key).setQuantity(qty);
            }
            pw.write("1");
        }
    }
}

```

- Mở trang **/views/cart/index.jsp** cập nhật thêm code **JavaScript** phía dưới

```

<script type="text/javascript">
var qties = document.getElementsByClassName('qty');
for(var i = qties.length - 1; i >= 0; i--){
    qties[i].onclick = function(){

```

```
var qty = this.value;
var id = this.getAttribute('data-id');
var xhr = new XMLHttpRequest();
xhr.onload = function(){
    alert(xhr.response);
}
xhr.open('POST', '${pageContext.request.contextPath}/cart/edit.html');
xhr.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
xhr.send('id=' + id +'&qty='+qty);
}
</script>
```

- Truy cập vào link /cart.html xem kết quả

BÀI 9: Phụ lục



Cơ sở dữ liệu trong ứng dụng MVC shop mini

9.1. Database Min shop

✓ Thiết kế CSDL Mini Shop:

- Bảng Category (Danh mục)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	CategoryId	int(11)			No	None		AUTO_INCREMENT
2	CategoryName	varchar(128)	utf8_general_ci		No	None		
3	ParentId	int(11)			Yes	NULL		

- Bảng Author (Tác giả)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	AuthorId	int(11)			No	None		AUTO_INCREMENT
2	AuthorName	varchar(128)	utf8_general_ci		No	None		

- Bảng Member (Thành viên)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	MemberId	bigint(20)			No	None		
2	Username	varchar(32)	utf8_general_ci		No	None		
3	Password	binary(32)			No	None		
4	Email	varchar(64)	utf8_general_ci		Yes	NULL		
5	FullName	varchar(128)	utf8_general_ci		Yes	NULL		
6	Gender	tinyint(4)			Yes	0		
7	AddedDate	datetime			No	CURRENT_TIMESTAMP		
8	ModifiedDate	datetime			Yes	NULL		

- Bảng Product (Sản phẩm)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	ProductId	int(11)			No	None		AUTO_INCREMENT
2	CategoryId	int(11)			No	None		
3	PublisherId	int(11)			No	None		
4	AuthorId	int(11)			No	None		
5	ISBN	varchar(16)	utf8_unicode_ci		No	None		
6	Title	varchar(128)	utf8_unicode_ci		No	None		
7	Pages	smallint(6)			No	None		
8	Year	smallint(6)			No	None		
9	Weight	varchar(32)	utf8_unicode_ci		Yes	NULL		
10	Size	varchar(16)	utf8_unicode_ci		Yes	NULL		
11	Description	varchar(1024)	utf8_unicode_ci		No	None		
12	Content	varchar(2048)	utf8_unicode_ci		Yes	NULL		
13	ImageUrl	varchar(128)	utf8_unicode_ci		Yes	NULL		
14	Price	int(11)			No	100000		

- Bảng Publisher (Nhà xuất bản)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	PublisherId	int(11)			No	None		AUTO_INCREMENT
2	PublisherName	varchar(128)	utf8_general_ci		No	None		

- Bảng Status (trạng thái đơn hàng)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	StatusId	tinyint(4)			No	None		
2	StatusName	varchar(32)	utf8_general_ci		No	None		

- Bảng Invoice (đơn hàng)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	Invoiced	bigint(20)			No	None		
2	MemberId	bigint(20)			Yes	NULL		
3	Email	varchar(64)	utf8_general_ci		No	None		
4	Tel	varchar(16)	utf8_general_ci		No	None		
5	Address	varchar(128)	utf8_general_ci		No	None		
6	StatusId	tinyint(4)			No	1		
7	AddedDate	datetime			No	CURRENT_TIMESTAMP		

- Bảng InvoiceDetail (chi tiết đơn hàng)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	Invoiced	bigint(20)			No	None		
2	Productid	int(11)			No	None		
3	Quantity	smallint(6)			No	None		
4	Price	int(11)			No	None		

- Bảng Role (Vai trò)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	RoleId	int(11)			No	None		
2	RoleName	varchar(32)	utf8_general_ci		No	None		

- Bảng MemberInRole (Bảng quan hệ giữa thành viên và vai trò)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	RoleId	int(11)			No	None		
2	MemberId	bigint(20)			No	None		

Relationship database Mini Shop (mối quan hệ giữa các bảng):

