BÀI TẬP MÔN LẬP TRÌNH WWW JAVA

(ADVANCED WEB PROGRAMMING WITH JAVA)

HỆ: ĐẠI HỌC

(DÀNH CHO CHUYÊN NGÀNH: SE)

BÀI TẬP TUẦN 01-02 MÔN LẬP TRÌNH WEB NÂNG CAO (JAVA)

Chương 1: Application Model - Web Application Architecture

Chương 2: Java Servlets

Mục tiêu:

- *(Review) Trình bày được mô hình ứng dụng Web các khái niệm liên quan.*
- *(Review) Thực hiện được layout của trang Web dùng HTML/CSS và thực hiện kiểm tra dữ liệu nhập phía Client dùng JavaScript.*
- *Hiểu được cấu trúc HTTP Request, HTTP Response.*
- *Phân tích yêu cầu theo đề tài bài tập lớn. Thực hiện các mô hình UML với yêu cầu tối thiểu.*
- *Cài đặt và cấu hình được Project với Tomcat 11 hoặc Glassfish 7.0*
- *Hiểu được một cấu trúc ứng dụng Web Back-End với Java Servlets.*
- *Thực hiện các bài tập FormData, Session Tracking, Send Mail, Upload Files với Java Servlets.*

Yêu cầu:

- *Tất cả các bài tập lưu trong thư mục: **T:\MSSV_HoTen_Tuan01***
- *Tạo Project **MSSV_HoTen_Tuan01** trong thư mục vừa tạo trong IDE IntelliJ Jakarta EE 11 (Jakarta Platform Enterprise Edition 11). Mỗi bài tập có thể lưu trong từng package riêng biệt.*
- *Cuối mỗi buổi thực hành, SV phải nén (.rar hoặc .zip) thư mục làm bài và nộp lại bài tập đã thực hiện trong buổi đó.*

Bài 1. Phân tích yêu cầu của Bài tập lớn

Yêu cầu của các đề tài 01 - 50, chức năng tối thiểu:

- Website bao gồm 3 loại người dùng tương tác: người dùng không có tài khoản (guest), người dùng có tài khoản (customer), người quản trị hệ thống (admin).
- Người dùng không có tài khoản (guest) có các chức năng:
- Xem danh sách sản phẩm (thiết bị máy tính, mỹ phẩm, quần áo ... tùy theo đề tài, danh sách này lấy từ CSDL)
- Xem chi tiết của từng sản phẩm từ danh sách sản phẩm.
- Chọn mua từng sản phẩm (có thể chọn mua từ trang Web danh sách sản phẩm hay từ trang Web chi tiết của từng sản phẩm), sản phẩm sau khi chọn mua sẽ được đưa vào trong giỏ hàng. o Xem giỏ hàng (danh sách sản phẩm đã chọn mua, thông tin này lưu trong biến Session, không cần cập nhật CSDL).
- Khi xem giỏ hàng, có thể chỉnh sửa số lượng của từng sản phẩm trong giỏ hàng (nếu chỉnh sửa số lượng là 0 □ bỏ sản phẩm đó ra khỏi giỏ hàng)
- Có thể đăng ký tài khoản của website với các thông tin cần thiết (email không trùng với tài khoản khác), sau khi đăng ký thành công với thông tin hợp lệ, lưu trữ CSDL + gửi email + thông báo về tài khoản.
- Người dùng có tài khoản (customer) có thể thực hiện các chức năng của Người dùng không có tài khoản (guest), ngoài ra người dùng có tài khoản (customer) còn có thể: o Xử lý thanh toán (chức năng này thực hiện khi giỏ hàng đã có sản phẩm và người dùng đăng nhập thành công vào hệ thống): cập nhật thông tin vào CSDL + gửi email + thông báo đăng ký đặt hàng thành công với các thông tin kèm theo. Sau khi xử lý thành công, Session được xóa về null.
- Người quản trị hệ thống (admin) có thể thực hiện được chức năng như một người dùng có tài khoản (customer). Ngoài ra, chức năng khác dành cho người quản trị hệ thống (admin) - Phần Back-End: o Tìm kiếm thông tin về sản phẩm/loại sản phẩm, tài khoản người dùng, các đơn đặt sản phẩm.
- Quản lý thông tin sản phẩm/loại sản phẩm:
- Xem danh sách sản phẩm/loại sản phẩm.
- Xem chi tiết từng sản phẩm/loại sản phẩm.
- Xóa sản phẩm/loại sản phẩm trong trường hợp sản phẩm chưa có trong đơn hàng nào hoặc loại sản phẩm chưa có sản phẩm nào.
- Thêm mới, cập nhật thông tin sản phẩm/loại sản phẩm.
- Quản lý thông tin tài khoản người dùng:
    - Xem danh sách các tài khoản người dùng đã đăng ký.
    - Xem chi tiết từng tài khoản người dùng, không xem được password của người dùng.
    - Xóa tài khoản người dùng nếu người dùng chưa thực hiện đặt hàng online lần nào.
    - Cập nhật thông tin tài khoản người dùng.
- Quản lý thông tin đơn hàng trực tuyến:
    - Xem danh sách các đơn hàng (sắp xếp theo ngày mua)
    - Xem chi tiết đơn hàng.
    - Cập nhật số lượng của mặt hàng trong đơn hàng trực tuyến o Lưu ý cho các chức năng quản lý thông tin:
    - Ràng buộc khi xóa dữ liệu

- Trường hợp thêm hay cập nhật dữ liệu có thể kiểm tra phía Client bằng JavaScript/jQuery hoặc kiểm tra bằng Model phía Server, không dùng Functions/Check constraints/Stored Procedures trong hệ quản trị CSDL.

Yêu cầu của các đề tài 51 □54, chức năng tối thiểu:

- Website bao gồm 3 loại người dùng tương tác: người dùng không có tài khoản (guest), người dùng có tài khoản (customer), người quản trị hệ thống (admin).
- Người dùng không có tài khoản (guest) có các chức năng:
  - Xem danh sách các báo giấy/tour/phòng khách sạn/dịch vụ bảo hiểm (tùy theo đề tài, danh sách này lấy từ CSDL) ○ Xem chi tiết của từng báo giấy/tour/phòng khách sạn/dịch vụ bảo hiểm từ danh sách.
  - Chọn từng tờ báo giấy/tour/phòng khách sạn/dịch vụ bảo hiểm (có thể chọn từ trang Web danh sách hay từ trang Web chi tiết), các tờ báo giấy/tour/phòng khách sạn/dịch vụ bảo hiểm sản phẩm sau khi chọn đặt sẽ được đưa vào trong giỏ hàng.
  - Xem giỏ hàng (danh sách đã chọn, thông tin này lưu trong biến Session, không cần cập nhật CSDL). ○ Khi xem giỏ hàng, có thể chỉnh sửa số lượng của từng thành phần trong giỏ hàng (nếu chỉnh sửa số lượng là 0 □ bỏ sản phẩm đó ra khỏi giỏ hàng)
  - Có thể đăng ký tài khoản của website với các thông tin cần thiết (email không trùng với tài khoản khác), sau khi đăng ký thành công với thông tin hợp lệ, lưu trữ CSDL + *gửi email* + thông báo về tài khoản.
- Người dùng có tài khoản (customer) có thể thực hiện các chức năng của Người dùng không có tài khoản (guest), ngoài ra người dùng có tài khoản (customer) còn có thể:
  - Xử lý thanh toán (chức năng này thực hiện khi giỏ hàng đã có thông tin và người dùng đăng nhập thành công vào hệ thống): cập nhật thông tin vào CSDL + *gửi email* + thông báo đăng ký đặt báo giấy/tour/phòng khách sạn/dịch vụ bảo hiểm thành công với các thông tin kèm theo. Sau khi xử lý thành công, Session được xóa về null.
- Người quản trị hệ thống (admin) có thể thực hiện được chức năng như một người dùng có tài khoản (customer). Ngoài ra, chức năng khác dành cho người quản trị hệ thống (admin) - Phần Back-End:
  - Tìm kiếm thông tin về báo giấy/tour/phòng khách sạn/dịch vụ bảo hiểm và các loại dịch vụ, tài khoản người dùng, các đơn đặt sản phẩm.
  - Quản lý thông tin báo giấy/tour/phòng khách sạn/dịch vụ bảo hiểm/loại:
    - Xem danh sách báo giấy/tour/phòng khách sạn/dịch vụ bảo hiểm/loại.
    - Xem chi tiết từng báo giấy/tour/phòng khách sạn/dịch vụ bảo hiểm/loại.
    - Xóa báo giấy/tour/phòng khách sạn/dịch vụ bảo hiểm/loại trong trường hợp thông tin cần xóa có trong đơn hàng nào hoặc loại sản phẩm chưa có sản phẩm nào.
    - Thêm mới, cập nhật thông tin sản phẩm/loại sản phẩm.
  - Quản lý thông tin tài khoản người dùng:
    - Xem danh sách các tài khoản người dùng đã đăng ký.
    - Xem chi tiết từng tài khoản người dùng, không xem được password của người dùng.
    - Xóa tài khoản người dùng nếu người dùng chưa thực hiện đặt hàng online lần nào. □ Cập nhật thông tin tài khoản người dùng.
  - Quản lý thông tin đơn hàng trực tuyến:

- Xem danh sách các đơn hàng (sắp xếp theo ngày mua) Xem chi tiết đơn hàng.
- Cập nhật số lượng của mặt hàng trong đơn hàng trực tuyến o Lưu ý cho các chức năng quản lý thông tin:
- Ràng buộc khi xóa dữ liệu
- Trường hợp thêm hay cập nhật dữ liệu có thể kiểm tra phía Client bằng
- JavaScript/jQuery hoặc kiểm tra bằng Model phía Server, không dùng Functions/Check constraints/Stored Procedures trong hệ quản trị CSDL.

Bài 2. Layout Bài tập lớn - Bài tập cá nhân

- Layout trang web dùng HTML/CSS dùng chung cho Project bài tập lớn (dùng cho nhóm)

Yêu cầu cho layout bài tập cá nhân hàng tuần:

<table>
<tr><td colspan="3"><header></td></tr>
<tr><td colspan="3"><menu></td></tr>
<tr><td><nav></td><td>Nội dung</td><td><section></td></tr>
<tr><td colspan="3"><footer></td></tr>
</table>

Hoặc tương tự:

Bài 3. Cài đặt 7.0 Server

Cài đặt GlassFish và thiết lập JDK 21 trong IDE IntelliJ Ultimate

1. IDE IntelliJ Ultimate
2. Glassfish 7.0
3. JDK 21

I. Tải và cài đặt GlassFish

Tải GlassFish 7 (hỗ trợ Jakarta EE 11)

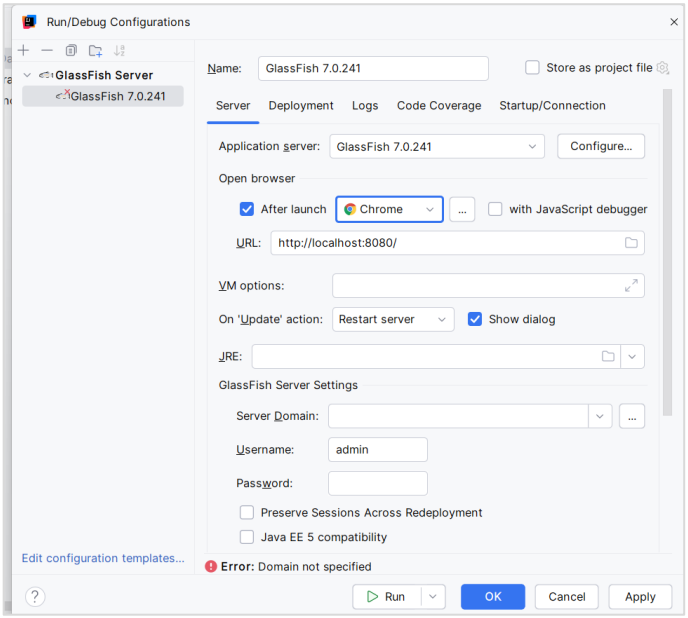
- Truy cập: https://projects.eclipse.org/projects/ee4j.glassfish/downloads
- Tải bản ZIP hoặc TAR.GZ phù hợp hệ điều hành
- Giải nén vào thư mục ví dụ: C:\glassfish7 hoặc /opt/glassfish7

II. Cấu hình GlassFish trong IntelliJ IDEA Ultimate

1. Cấu hình Application Server
   - Vào menu: File → Settings (hoặc Ctrl + Alt + S)
   - Chọn: Build, Execution, Deployment → Application Servers
   - Click +, chọn GlassFish Server
   - Chỉ đường dẫn tới thư mục glassfish7

IntelliJ sẽ tự nhận thư mục glassfish chứa bin, lib, v.v...

2. Tạo cấu hình chạy (Run Configuration)
   - Vào menu: Run → Edit Configurations
   - Click +, chọn GlassFish Server → Local
   - Chọn cấu hình (Application Server: GlassFish bạn đã thêm ở trên)
   - Ở tab Deployment, Click + → chọn Artifact hoặc WAR exploded

3. 

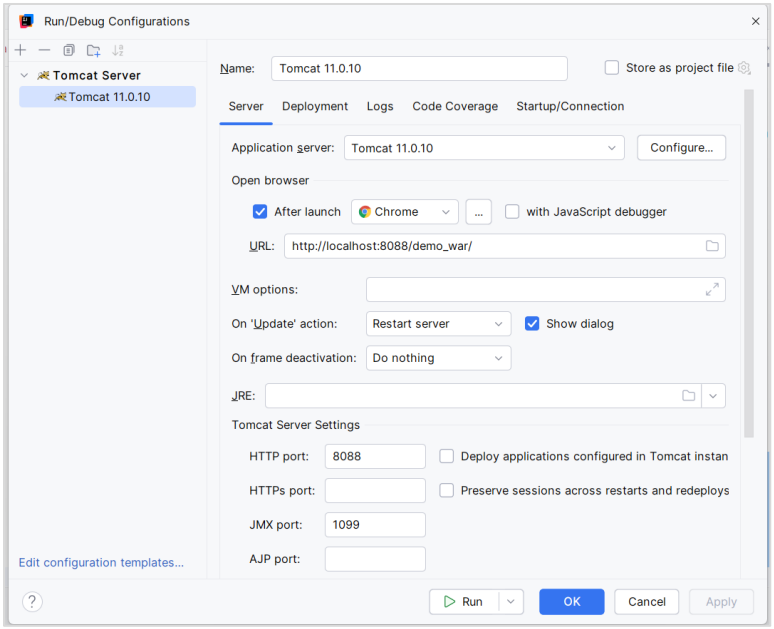
   - Tạo project kiểu Java Enterprise
   - Chọn Jakarta EE 11
   - Module: chọn Web Application, bật các framework như JPA, Servlet, CDI nếu cần

III. Chạy ứng dụng

1. Click Run hoặc Debug
2. IntelliJ sẽ khởi động GlassFish, triển khai ứng dụng của bạn
3. Truy cập ở http://localhost:8080/tên-ứng-dụng

Cách add Tomcat 11 vào IntelliJ Ultimate

1. Chuẩn bị
   - Đảm bảo Tomcat 11 đã cài và chạy được (Java 17+).
   - Ghi nhớ đường dẫn thư mục cài Tomcat, ví dụ:
2. Mở cấu hình Application Server trong IntelliJ
   1. Mở IntelliJ → Vào menu File → Settings (hoặc Ctrl+Alt+S).
   2. Chọn mục Build, Execution, Deployment → Application Servers.
   3. Bấm dấu + → chọn Tomcat Server → Local.
3. Chỉ định đường dẫn Tomcat
   - Ở mục Tomcat Home → chọn thư mục cài Tomcat 11.
   - IntelliJ sẽ tự nhận version, ví dụ Apache Tomcat 11.0.0.
   - Ấn OK để lưu.
4. Tạo cấu hình chạy (Run/Debug Configuration)
   1. Vào menu Run → Edit Configurations…
   2. Nhấn + → chọn Tomcat Server → Local.
   3. Ở tab Server:
      - Application server: chọn Tomcat 11 bạn vừa add ở bước 3.
      - HTTP port: mặc định 8080 (bạn có thể đổi nếu port này bị chiếm).
   4. Ở tab Deployment:
      - Nhấn + → chọn Artifact hoặc External Source (tùy project của bạn).
      - Nếu là Maven/Gradle → chọn artifact dạng war exploded hoặc war.
   5. Ấn OK để lưu cấu hình.



5. Chạy thử

- Chọn cấu hình Tomcat vừa tạo ở góc trên phải IntelliJ.
- Bấm Run (Shift+F10).
- Truy cập: http://localhost:8080 để xem Tomcat chạy.

Bài 4. Java Servlet - Thao tác với doGet(), doPost()

Bài tập dùng Servlet truyền dữ liệu thông qua Form Data. Tạo form đăng ký thông tin bao gồm các thành phần TextBox, CheckBox, ComboBox, TextArea.

HTML Form Example with File Upload

Name:

Password:

Gender: Male Female

Hobbies: Reading Sports Music

Country: Vietnam

Birth Date: dd/mm/yyyy

Profile Picture: Chọn tệp Không có tệp nào được chọn

Submit

Hướng dẫn:

Bước 1: Tạo trang .jsp chứa HTML form.

```
<!-- enctype bắt buộc khi upload file -->
<form action="${pageContext.request.contextPath}/processFormUpload"
      method="post" enctype="multipart/form-data">
```

Bước 2: Tạo Servlet `@WebServlet("/processFormUpload")` *lấy dữ liệu từ form, xuất ra text.*

```
@WebServlet("/processFormUpload")
@MultipartConfig(
        fileSizeThreshold = 1024 * 1024,  // 1MB
        maxFileSize = 1024 * 1024 * 10,   // 10MB
        maxRequestSize = 1024 * 1024 * 15 // 15MB
)
public class FormUploadServlet extends HttpServlet {
```

Lấy data từ thành phần của form:
(Lưu ý các thành phần form text, radio, checkbox, Selection list,...)
req.getParameter(); → Trả về String
req.getParameterValues(); → Trả về mảng String

```
req.setCharacterEncoding("utf-8");
String name = req.getParameter( s: "name");
String password = req.getParameter( s: "password");
String gender = req.getParameter( s: "gender");
String[] hobbies = req.getParameterValues( s: "hobbies");
String country = req.getParameter( s: "country");
String birthDate = req.getParameter( s: "birthDate");
...
```

Lấy file và lưu file:

```
//lấy File
Part filePart = req.getPart( s: "profilePic");
String fileName = filePart.getSubmittedFileName();

// lưu vào thư mục uploads
String uploadPath = System.getProperty("user.home") + File.separator + "uploads";

File uploadDir = new File(uploadPath);
if (!uploadDir.exists()) {
    uploadDir.mkdir();
}

//lưu file vào thư mục
filePart.write( s: uploadPath + File.separator + fileName);
```

Xuất dữ liệu:

```
resp.setContentType("text/html;charset=UTF-8");
resp.getWriter().println("<h2>Form Data Received:</h2>");
resp.getWriter().println("Name: " + name + "<br>");
resp.getWriter().println("Password: " + password + "<br>");
resp.getWriter().println("Gender: " + gender + "<br>");
resp.getWriter().println("Hobbies: " + (hobbies != null ? String.join( delimiter: ", ", hobbies) : "None") + "<br>");
resp.getWriter().println("Country: " + country + "<br>");
resp.getWriter().println("Birth Date: " + birthDate + "<br>");
resp.getWriter().println("Uploaded File: " + (fileName != null ? fileName : "No file") + "<br>");
resp.getWriter().println("Saved to: " + uploadPath + "<br>");
```

Bài 5. Java Servlet - Filter

Thực hiện Filter qua ứng dụng mô tả như sau: Filter trong Servlet kiểm tra đăng nhập: Nếu đúng thông tin đăng nhập thì thực hiện các chức năng của servlet trong thư mục /secure.

Các file cần thực hiện:

- AuthFilter.java

Filter lọc hết các request, chỉ chấp nhận các request trong thư mục /secure

```
@WebFilter("/secure/*")
public class AuthFilter implements Filter {
    @Override
    public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)

        HttpServletRequest req = (HttpServletRequest) request;
        HttpServletResponse res = (HttpServletResponse) response;
        HttpSession session = req.getSession( create: false);

        boolean loggedIn = (session != null && session.getAttribute( name: "user") != null);

        if (loggedIn) {
            chain.doFilter(request, response); // Cho phép đi tiếp
        } else {
            res.sendRedirect( location: req.getContextPath() + "/login.jsp");
        }
    }
}
```

- *login.jsp* tương ứng cho phần xử lý backend *loginServlet.java.*

login.jsp (SV tự thiết kế)

Đăng nhập

Tên đăng nhập:

Mật khẩu:

Đăng nhập

LoginServlet.java

```java
@WebServlet("/login")
public class LoginServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;  no usages
    public LoginServlet() { }  no usages

    @Override  1 usage
    protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws
//        super.doPost(req, resp);
        String username = req.getParameter( name: "username");
        String password = req.getParameter( name: "password");
        // username = admin, password=123
        if("admin".equals(username) && "123".equals(password)) {
            HttpSession session = req.getSession();
            session.setAttribute( name: "user", username);
            resp.sendRedirect( location: req.getContextPath() + "/home.jsp");
        }else
        {
            req.setAttribute( name: "error",  o: "Sai tài khoản");
            req.getRequestDispatcher( path: "/login.jsp").forward(req, resp);
        }
    }
}
```

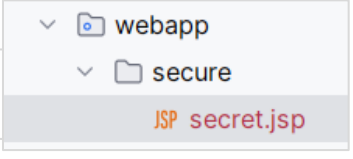
- LogoutServlet.java

```java
@WebServlet("/logout")
public class LogoutServlet extends HttpServlet {

    @Override  5 usages
    protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws
//        super.doGet(req, resp);
        HttpSession session = req.getSession( create: false);
        if (session != null) {
            session.invalidate();
        }
        resp.sendRedirect( location: req.getContextPath() + "/login.jsp");
    }


}
```

- home.jsp cho phép đăng nhập bị chặn theo AuthFilter.java

```jsp
<h2>Xin chào, ${sessionScope.user}</h2>
<a href="${pageContext.request.contextPath}/secure/secret.jsp">Trang bảo mật</a><br>
<a href="${pageContext.request.contextPath}/logout">Đăng xuất</a>
```

- /secure/secret.jsp: chỉ cho phép các Servlet trong thư mục /secure thực hiện theo Filter, cấu trúc thư mục /secure trên Project như sau:



secret.jsp

```jsp
<h2>Đây là nội dung bí mật chỉ user đã login mới thấy!</h2>
<a href="${pageContext.request.contextPath}/home.jsp">Về trang chủ</a>
```

Bài 6. Java Servlet - Upload files

Thực hiện upload nhiều file lưu trữ ở Server.

Chức năng **upload nhiều file** trên **Tomcat 11** bằng @MultipartConfig và Servlet.

Hướng dẫn

Bước 1. Tạo trang .jsp chứa HTML form. Lưu ý thuộc tính của form enctype="multipart/form-data"

```
<form action="${pageContext.request.contextPath}/uploadmulti" method="post" enctype="multipart/form-data">
    File #1: <input type="file" name="file"/><br/><br/>
    File #2: <input type="file" name="file"/><br/><br/>
    File #3: <input type="file" name="file"/><br/><br/>
    File #4: <input type="file" name="file"/><br/><br/>
    File #5: <input type="file" name="file"/><br/><br/>
    <input type="submit" value="Upload"/>
    <input type="reset" value="Reset"/>
</form>
```

Bước 2: Tạo Servlet xử lý :

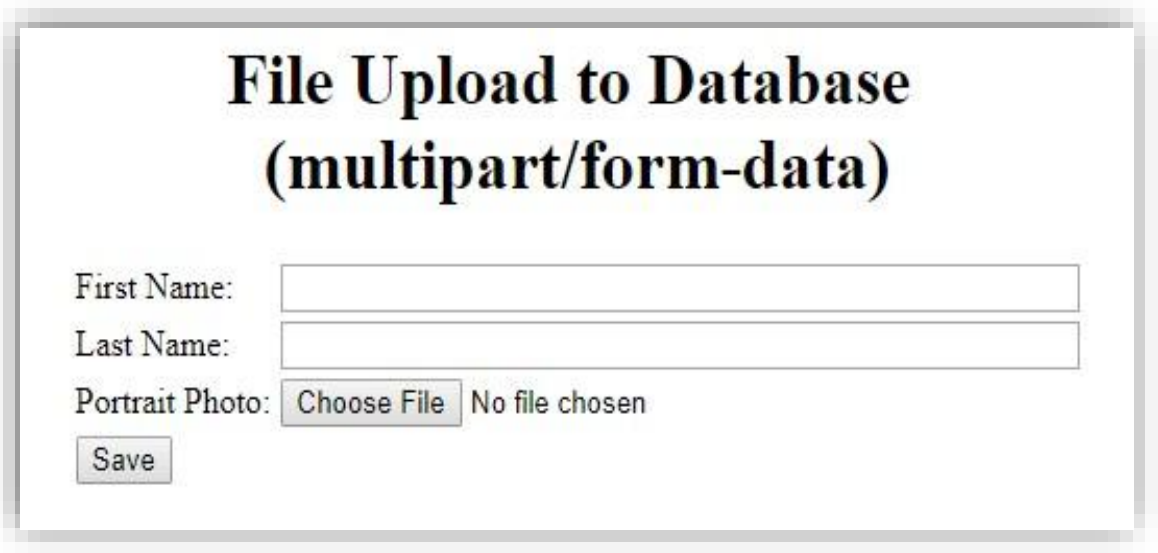
- *Đọc dữ liệu của form: dữ liệu text + dữ liệu file*

```
@WebServlet("/uploadmulti")
@MultipartConfig
        (
                fileSizeThreshold = 1024 * 1024,
                maxFileSize = 1024 * 1024 * 10,
                maxRequestSize = 1024 * 1024 * 50
        )
public class MultiFileUploadServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;  no usages
    public MultiFileUploadServlet() {}  no usages
    private static final String UPLOAD_DIR = "uploads";  1 usage
```

```java
@Override  1 usage
protected void doPost(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
    super.doPost(req, resp);
    // Thư mục lưu file trên server (trong thư mục webapp)
    String uploadPath = getServletContext().getRealPath( s: "") +  UPLOAD_DIR;
    //File.separator +

    File uploadDir = new File(uploadPath);
    if (!uploadDir.exists()) {
        uploadDir.mkdir();
    }

    // Lấy tất cả file từ request
    for (Part part : req.getParts()) {
        String fileName = getFileName(part);
        if (fileName != null && !fileName.isEmpty()) {
            part.write( s: uploadPath + File.separator + fileName);
        }
    }

    resp.setContentType("text/html;charset=UTF-8");
    resp.getWriter().println("<h3>Files uploaded successfully to " + uploadPath + "</h3>");
}
private String getFileName(Part part) {  1 usage
    String contentDisp = part.getHeader( s: "content-disposition");
    String[] tokens = contentDisp.split( regex: ";");
    for (String token : tokens) {
        if (token.trim().startsWith("filename")) {
            return token.substring(token.indexOf("=") + 2, token.length() - 1)
        }
    }
    return null;
}
}
```

Bài 7. Java Servlet - Upload hình, lưu CSDL

Thực hiện trang Web cho phép upload hình ảnh và lưu dạng Binary trong CSDL SQL Server.



<u>Hướng dẫn:</u> (Sinh viên cần tách các lớp thao tác với CSDL, không dùng chung)

- CSDL: (SQL Server)

```
CREATE DATABASE UploadFileServletDB; CREATE
TABLE contacts (
  contact_id int NOT NULL identity(1,1),
first_name nvarchar(45) DEFAULT NULL,
last_name nvarchar(45) DEFAULT NULL,
  photo  image,
  PRIMARY KEY (contact_id)

)
```

Hướng dẫn:

Bước 1: Tạo Servlet hoặc trang HTML form. Lưu ý thuộc tính của form enctype="multipart/form-data"

```
<form action="${pageContext.request.contextPath}/uploaddatabase" method="post" enctype="multipart/form-data">
    First Name: <input type="text" name="firstName"><br><br>
    Last Name: <input type="text" name="lastName"><br><br>
    Portrait Photo: <input type="file" name="photo"><br><br>
    <input type="submit" value="Save">
</form>
```

Bước 2: Tạo Servlet xử lý:

- *Đọc dữ liệu của form: dữ liệu text + dữ liệu file*
- *Kết nối CSDL : jdbc kết nối CSDL*

```
// Thay đổi kết nối theo SQL Server của bạn
private String dbURL = "jdbc:sqlserver://localhost\\MSSQLLocalDB;databaseName=QLThoiKhoaBieu;" + 1 usage
        "integratedSecurity=true;encrypt=false";
private String dbUser = "sa"; 1 usage
private String dbPass = "sapassword"; 1 usage
```

- *Insert dữ liệu vào bảng gồm cả dữ liệu Text và dữ liệu Image*

```
    protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
//        super.doPost(req, resp);
        String firstName = req.getParameter( s: "firstName");
        String lastName = req.getParameter( s: "lastName");

        InputStream inputStream = null;
        Part filePart = req.getPart( s: "photo");
        if (filePart != null) {
            inputStream = filePart.getInputStream();
        }

        Connection conn = null;
        String message = null;

    try {
        DriverManager.registerDriver(new com.microsoft.sqlserver.jdbc.SQLServerDriver());
        conn = DriverManager.getConnection(dbURL, dbUser, dbPass);

        String sql = "INSERT INTO StudentPhotos (first_name, last_name, photo) values (?, ?, ?)";
        PreparedStatement statement = conn.prepareStatement(sql);
        statement.setString( parameterIndex: 1, firstName);
        statement.setString( parameterIndex: 2, lastName);

        if (inputStream != null) {
            statement.setBinaryStream( parameterIndex: 3, inputStream, (int) filePart.getSize());
        } else {
            statement.setNull( parameterIndex: 3, Types.BLOB);
        }


            int row = statement.executeUpdate();
            if (row > 0) {
                message = "Upload and save into database successful!";
            }
        } catch (Exception ex) {
            ex.printStackTrace();
            message = "ERROR: " + ex.getMessage();
        } finally {
            if (conn != null) try { conn.close(); } catch (SQLException ignore) {}
        }

        resp.setContentType("text/html;charset=UTF-8");
        resp.getWriter().println("<h3>" + message + "</h3>");

    }
}
```

Bước 3: Trả về kết quả sau khi đã thực hiện xong.