7. Design a controller with servlet that provides the interaction with application developed in experiment 1 and the database created in experiment 5.

To design a controller with a servlet that provides a login page, you need to follow these steps:

1. **Set up your project**: Create a dynamic web project in your IDE (e.g., Eclipse).
2. **Create the servlet**: Write the servlet that will handle the login request.
3. **Create the login JSP page**: Write the JSP page that will be the login form.
4. **Configure the web.xml**: Define the servlet mapping in the `web.xml` file.

Step 1: Set up your project

Create a dynamic web project in your IDE and ensure you have the required libraries and server setup (e.g., Apache Tomcat).

https://www.eclipse.org/downloads/packages/release/2022-06/r/eclipse-ide-enterprise-java-and-web-developers

download Eclipse IDE for Enterprise Java and Web Developers

Step 2: Create the servlet

Create a servlet named `LoginServlet.java`.

```
package com.example;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;


@WebServlet("/login")

public class LoginServlet extends HttpServlet {

    private static final long serialVersionUID = 1L;

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

        // Forward to login JSP page

        request.getRequestDispatcher("login.jsp").forward(request, response);
```

```
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        // Handle login logic
        String username = request.getParameter("username");
        String password = request.getParameter("password");

        if ("admin".equals(username) && "password".equals(password)) {
            // Redirect to welcome page on successful login
            response.sendRedirect("welcome.jsp");
        } else {
            // Redirect back to login page with error message
            request.setAttribute("errorMessage", "Invalid username or password");
            request.getRequestDispatcher("login.jsp").forward(request, response);
        }
    }
}
```

Step 3: Create the login JSP page

Create a `login.jsp` file in the `WebContent` or `webapp` directory.

```
<!DOCTYPE html>
<html>
<head>
    <title>Login Page</title>
</head>
<body>
    <h2>Login</h2>
    <form action="login" method="post">
        <label for="username">Username:</label>
        <input type="text" id="username" name="username" required><br><br>
        <label for="password">Password:</label>
```

```
    <input type="password" id="password" name="password" required><br><br>
    <input type="submit" value="Login">
  </form>
  <c:if test="${not empty errorMessage}">
    <p style="color: red;">${errorMessage}</p>
  </c:if>
</body>
</html>
```

Step 4: Configure the web.xml

Ensure you have the servlet mapping in your `web.xml` file.

```
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" version="3.1">
  <servlet>
    <servlet-name>LoginServlet</servlet-name>
    <servlet-class>com.example.LoginServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>LoginServlet</servlet-name>
    <url-pattern>/login</url-pattern>
  </servlet-mapping>
</web-app>
```

Directory Structure

Your project directory should look like this:

```
MyWebApp/
│
├── src/
│   └── com/
│       └── example/
```

```
│       └── LoginServlet.java
│
├── WebContent/
│   └── login.jsp
│   └── WEB-INF/
│       └── web.xml
```