TUTOTIAL 02

01) import java.io.IOException;

import java.io.PrintWriter;

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(name = "HelloWorldServlet", urlPatterns = {"/hello"})

public class HelloWorldServlet extends HttpServlet {

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

try (PrintWriter out = response.getWriter()) {

out.println("<!DOCTYPE html>");

out.println("<html>");

out.println("<head>");

out.println("<title>Hello, World!</title>");

out.println("</head>");

out.println("<body>");

out.println("<h1>Hello, World!</h1>");

out.println("</body>");

out.println("</html>");

}

}

}

<!DOCTYPE html>

<html>

<head>

<title>Welcome</title>

</head>

<body>

<h1>Welcome to My Java Web Application!</h1>

<p>This is a simple Java web application created using NetBeans.</p>

</body>

</html>

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

<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>HelloWorldServlet</servlet-name>

<servlet-class>HelloWorldServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>HelloWorldServlet</servlet-name>

<url-pattern>/hello</url-pattern>

</servlet-mapping>

<welcome-file-list>

<welcome-file>index.jsp</welcome-file>

</welcome-file-list>

</web-app>

2)

\*\*\* Adding a Dependency for Servlet API:

The servlet container (such as Apache Tomcat) provides Servlet API. If you are using Apache Tomcat as your server in NetBeans, the Servlet API is already included in the server's libraries, so you don't need to add it explicitly. Nevertheless, take note of these actions if you're using a different server or wish to manually add the Servlet API:

Select your project with a right-click in the "Projects" pane.

Go to the context menu and choose "Properties".

Navigate to the "Libraries" tab in the project properties box.

The "Add Library..." button should be clicked.

Select the relevant library (such as "Apache Tomcat") by choosing "Server Library".

To exit the properties box, click "OK" after selecting "Add Library".

\*\*

Adding JSP API Dependency:

Usually, the servlet container also offers JSP API. As with the Servlet API, you can add the JSP API implicitly if you're using Apache Tomcat. Nevertheless, take note of these actions if you're using a different server or wish to manually add the JSP API:

Select your project with a right-click in the "Projects" pane.

Go to the context menu and choose "Properties".

Navigate to the "Libraries" tab in the project properties box.

The "Add Library..." button should be clicked.

Select the library named "Java EE Web API".

To exit the properties box, click "OK" after selecting "Add Library".

3) <!DOCTYPE html>

<html>

<head>

<title>Login</title>

<style>

/\* Simple CSS for styling the form \*/

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

background-color: #f2f2f2;

}

form {

background-color: white;

padding: 20px;

border-radius: 5px;

box-shadow: 0px 0px 10px 0px rgba(0,0,0,0.1);

}

input[type="text"],

input[type="password"],

input[type="submit"] {

width: 100%;

padding: 10px;

margin-bottom: 10px;

border: 1px solid #ccc;

border-radius: 3px;

box-sizing: border-box;

}

input[type="submit"] {

background-color: #007bff;

color: white;

cursor: pointer;

}

</style>

</head>

<body>

<form action="loginServlet" method="post"> <!-- Assuming "loginServlet" is the servlet handling the login -->

<h2>Login</h2>

<label for="username">Username:</label>

<input type="text" id="username" name="username" required><br>

<label for="password">Password:</label>

<input type="password" id="password" name="password" required><br>

<input type="submit" value="Login">

</form>

</body>

</html>

04) 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(name = "LoginServlet", urlPatterns = {"/loginServlet"})

public class LoginServlet extends HttpServlet {

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Retrieve the username and password from the request parameters

String username = request.getParameter("username");

String password = request.getParameter("password");

// Implement your login logic here

// For example, you can check if the username and password are valid

// and redirect the user to a different page based on the result

// For demonstration purposes, let's just print the username and password

System.out.println("Username: " + username);

System.out.println("Password: " + password);

// You can then forward or redirect the user to another page

// For example:

// request.getRequestDispatcher("welcome.jsp").forward(request, response);

// response.sendRedirect("welcome.jsp");

}

}

05) 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(name = "LoginServlet", urlPatterns = {"/loginServlet"})

public class LoginServlet extends HttpServlet {

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Retrieve parameters from the request

String username = request.getParameter("username");

// Set attributes to be displayed on the Index.jsp page

request.setAttribute("username", username);

// Forward the request to the Index.jsp page

request.getRequestDispatcher("Index.jsp").forward(request, response);

}

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Retrieve the username and password from the request parameters

String username = request.getParameter("username");

String password = request.getParameter("password");

// Implement your login logic here

// For demonstration purposes, let's just print the username and password

System.out.println("Username: " + username);

System.out.println("Password: " + password);

// You can then forward or redirect the user to another page

// For example:

// request.getRequestDispatcher("welcome.jsp").forward(request, response);

// response.sendRedirect("welcome.jsp");

}

}

<!DOCTYPE html>

<html>

<head>

<title>Welcome</title>

</head>

<body>

<h1>Welcome, <%= request.getAttribute("username") %>!</h1>

<p>This is a simple Java web application created using NetBeans.</p>

</body>

</html>

06)\*\* URL Parameters Viewable in Browser History:

Form parameters are attached to the URL when utilising the GET technique. This implies that private data, including passwords and usernames, will be accessible through the address bar of the browser and might even be saved in the history. This presents a risk in the event that someone else shares the user's device or gains access to the history.

\*\*Information Leakage via Logs:

Web servers frequently record URLs and other HTTP requests. The server might unintentionally log sensitive data if it is sent using GET parameters. Attackers or unauthorised users may be able to view these records, which could expose confidential information.

\*\*URL Length Restrictions: The majority of web servers and browsers place restrictions on URL length. Sensitive data may be truncated as a result of this restriction, which may limit the amount of data that may be transmitted using GET parameters. This restriction could be used by attackers to truncate or modify parameters in order to get around security measures or insert malicious payloads.

\*\*Caching Issues: Browsers, proxies, and intermediary servers frequently cache GET requests. Sensitive URLs that have been cached may be kept on the client end or at different locations in the network path. Because of this caching behaviour, unauthorised persons with access to cached resources may be able to see sensitive data.

\*\*Referrer Headers: The URL of the referring page is usually included in the HTTP request headers by the browser when a user navigates from one page to another. This implies that the Referrer header of a future request may contain sensitive data that was sent via GET parameters. Sensitive information may unintentionally be made public on websites run by third parties if the destination URL is located on a different domain.

\*\*Vulnerabilities related to cross-site scripting (XSS):

GET parameters are susceptible to XSS attacks, which take advantage of the way malicious scripts are inserted into websites and run within the user's browser. Attackers may use XSS vulnerabilities to change an application's behaviour or steal confidential data sent via GET parameters.

07) <!DOCTYPE html>

<html>

<head>

<title>Login</title>

<style>

/\* Simple CSS for styling the form \*/

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

background-color: #f2f2f2;

}

form {

background-color: white;

padding: 20px;

border-radius: 5px;

box-shadow: 0px 0px 10px 0px rgba(0,0,0,0.1);

}

input[type="text"],

input[type="password"],

input[type="submit"] {

width: 100%;

padding: 10px;

margin-bottom: 10px;

border: 1px solid #ccc;

border-radius: 3px;

box-sizing: border-box;

}

input[type="submit"] {

background-color: #007bff;

color: white;

cursor: pointer;

}

</style>

</head>

<body>

<form action="loginServlet" method="post"> <!-- Changed method attribute to "post" -->

<h2>Login</h2>

<label for="username">Username:</label>

<input type="text" id="username" name="username" required><br>

<label for="password">Password:</label>

<input type="password" id="password" name="password" required><br>

<input type="submit" value="Login">

</form>

</body>

</html>

08) 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(name = "LoginServlet", urlPatterns = {"/loginServlet"})

public class LoginServlet extends HttpServlet {

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Retrieve the username and password from the request parameters

String username = request.getParameter("username");

String password = request.getParameter("password");

// Perform validation of the username and password

boolean isValidCredentials = validateCredentials(username, password);

// Set attributes indicating the result of the validation

if (isValidCredentials) {

request.setAttribute("message", "Login successful. Welcome, " + username + "!");

} else {

request.setAttribute("message", "Invalid username or password. Please try again.");

}

// Forward the request to the Index.jsp page

request.getRequestDispatcher("Index.jsp").forward(request, response);

}

private boolean validateCredentials(String username, String password) {

// Perform actual validation logic here

// For demonstration purposes, let's assume hardcoded valid credentials

return "admin".equals(username) && "password123".equals(password);

}

}

<!DOCTYPE html>

<html>

<head>

<title>Login</title>

</head>

<body>

<h1>Login</h1>

<%-- Display the message attribute set by the servlet --%>

<p><%= request.getAttribute("message") %></p>

<form action="loginServlet" method="post">

<label for="username">Username:</label>

<input type="text" id="username" name="username" required><br>

<label for="password">Password:</label>

<input type="password" id="password" name="password" required><br>

<input type="submit" value="Login">

</form>

</body>

</html>