Ex No: 6
Date:

SESSION TRACKING & LOGIN PAGE

AIM:

To demonstrate session tracking using HttpSession in a Java Servlet by implementing a secure login system that maintains user state across multiple requests.

ALGORITHM:

1. **Design the User Interface**
    - Create a login page with username and password fields
    - Design a welcome page to display session information
    - Include a logout mechanism
2. **Session Creation Phase**
    - Receive user credentials via form submission
    - Validate credentials against stored values
    - For successful authentication:
        - Initialize a new session
        - Store user identity in session attributes
        - Record session start time
    - For failed authentication:
        - Return to login page with error message
3. **Session Maintenance Phase**
    - Verify session existence for protected pages
    - Retrieve stored attributes for personalized content
    - Calculate and display session duration
    - Track last activity timestamp
4. **Session Termination Phase**
    - Process logout requests
    - Invalidate the session object
    - Clear all session attributes
    - Redirect to login page with confirmation
5. **Security Implementation**
    - Enforce session checks for all restricted pages
    - Implement session timeout handling
    - Prevent session fixation attacks
    - Secure session cookie attributes

CODE:

LoginServlet.java

```
package com.example.session;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class LoginServlet extends HttpServlet {
    private static final String USERNAME = "admin";
    private static final String PASSWORD = "password";

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

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

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

        if(USERNAME.equals(username) && PASSWORD.equals(password)) {
            HttpSession session = request.getSession(true);
            session.setAttribute("username", username);
            response.sendRedirect("WelcomeServlet");
        } else {
            out.println("<h3>Invalid credentials. Please try again.</h3>");
            out.println("<a href='login.html'>Back to Login</a>");
        }
    }
}
```

LogoutServlet.java

```
package com.example.session;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class LogoutServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        HttpSession session = request.getSession(false);

        if(session != null) {
            String username = (String) session.getAttribute("username");
```

```
            session.invalidate();
            out.println("<h3>" + username + ", you have been logged
out.</h3>");
        } else {
            out.println("<h3>You were not logged in!</h3>");
        }
        out.println("<a href='login.html'>Login again</a>");
    }
}
```

WelcomeServlet.java

```
package com.example.session;
import java.io.*;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.servlet.*;
import javax.servlet.http.*;

public class WelcomeServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        HttpSession session = request.getSession(false);

        if(session == null || session.getAttribute("username") == null) {
            response.sendRedirect("login.html");
            return;
        }

        String username = (String) session.getAttribute("username");
        SimpleDateFormat sdf = new SimpleDateFormat("MMM dd, yyyy HH:mm:ss");

        out.println("<html><head><title>Welcome</title></head><body>");
        out.println("<h2>Welcome, " + username + "!</h2>");
        out.println("<p>Session ID: " + session.getId() + "</p>");
        out.println("<p>Last accessed: " + sdf.format(new
Date(session.getLastAccessedTime())) + "</p>");
        out.println("<form action='LogoutServlet' method='post'>");
        out.println("<input type='submit' value='Logout'>");
        out.println("</form></body></html>");
    }
}
```

Login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Neon Login | Session Demo</title>
    <style>
        :root {
            --neon-blue: #0ff0fc;
            --neon-pink: #ff00ff;
            --neon-purple: #8a2be2;
            --dark-bg: #0a0a1a;
        }

        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
            font-family: 'Rajdhani', 'Arial Narrow', sans-serif;
        }

        body {
            background: var(--dark-bg);
            min-height: 100vh;
            display: flex;
            justify-content: center;
            align-items: center;
            overflow: hidden;
            color: white;
        }

        /* Animated background grid */
        .grid {
            position: fixed;
            top: 0;
            left: 0;
            width: 100%;
            height: 100%;
            background:
                linear-gradient(rgba(10, 10, 26, 0.9),
                rgba(10, 10, 26, 0.9)),
                url('data:image/svg+xml;utf8,<svg
xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100"
preserveAspectRatio="none"><path d="M0,0 L100,0 L100,100 L0,100 Z" fill="none"
stroke="%231e1e3a" stroke-width="0.5"/></svg>');
            background-size: 50px 50px;
            z-index: -1;
            animation: gridMove 20s linear infinite;
        }

        @keyframes gridMove {
            0% { background-position: 0 0; }
            100% { background-position: 50px 50px; }
        }

        .login-container {
```

```
            background: rgba(20, 20, 40, 0.8);
            border-radius: 10px;
            width: 90%;
            max-width: 400px;
            padding: 40px;
            box-shadow: 0 0 20px rgba(0, 255, 252, 0.2);
            position: relative;
            overflow: hidden;
            animation: glowPulse 4s infinite alternate;
        }

        @keyframes glowPulse {
            0% { box-shadow: 0 0 20px rgba(0, 255, 252, 0.2); }
            100% { box-shadow: 0 0 30px rgba(0, 255, 252, 0.4),
                               0 0 60px rgba(255, 0, 255, 0.2); }
        }

        .login-container::before {
            content: '';
            position: absolute;
            top: -2px;
            left: -2px;
            right: -2px;
            bottom: -2px;
            background: linear-gradient(45deg,
                var(--neon-blue),
                var(--neon-pink),
                var(--neon-purple));
            z-index: -1;
            border-radius: 12px;
            animation: borderRotate 8s linear infinite;
        }

        @keyframes borderRotate {
            0% { filter: blur(5px); opacity: 0.7; transform: rotate(0deg); }
            100% { filter: blur(5px); opacity: 0.7; transform: rotate(360deg);
}
        }

        h1 {
            text-align: center;
            margin-bottom: 30px;
            font-size: 2.2rem;
            text-transform: uppercase;
            letter-spacing: 3px;
            color: var(--neon-blue);
            text-shadow: 0 0 10px var(--neon-blue),
                         0 0 20px var(--neon-blue);
            animation: textGlow 2s infinite alternate;
        }
```

```
        @keyframes textGlow {
            0% { text-shadow: 0 0 10px var(--neon-blue),
                              0 0 20px var(--neon-blue); }
            100% { text-shadow: 0 0 15px var(--neon-blue),
                                0 0 30px var(--neon-blue),
                                0 0 45px var(--neon-pink); }
        }

        .form-group {
            margin-bottom: 25px;
            position: relative;
        }

        label {
            display: block;
            margin-bottom: 8px;
            color: var(--neon-blue);
            font-weight: 500;
            letter-spacing: 1px;
        }

        input {
            width: 100%;
            padding: 15px;
            background: rgba(10, 10, 30, 0.7);
            border: 1px solid var(--neon-purple);
            border-radius: 5px;
            color: white;
            font-size: 16px;
            transition: all 0.3s;
        }

        input:focus {
            outline: none;
            border-color: var(--neon-blue);
            box-shadow: 0 0 10px rgba(0, 255, 252, 0.5);
        }

        button {
            width: 100%;
            padding: 15px;
            background: linear-gradient(45deg, var(--neon-purple), var(--neon-
pink));
            border: none;
            border-radius: 5px;
            color: white;
            font-size: 18px;
            font-weight: bold;
            letter-spacing: 1px;
            cursor: pointer;
            transition: all 0.3s;
```

```
        position: relative;
        overflow: hidden;
        z-index: 1;
    }

    button:hover {
        transform: translateY(-3px);
        box-shadow: 0 5px 15px rgba(255, 0, 255, 0.4);
    }

    button::before {
        content: '';
        position: absolute;
        top: 0;
        left: -100%;
        width: 100%;
        height: 100%;
        background: linear-gradient(90deg,
            transparent,
            rgba(255, 255, 255, 0.2),
            transparent);
        transition: 0.5s;
        z-index: -1;
    }

    button:hover::before {
        left: 100%;
    }

    .error-message {
        color: #ff3860;
        margin-top: 20px;
        padding: 10px;
        border-radius: 5px;
        background: rgba(255, 56, 96, 0.1);
        display: none;
        text-shadow: 0 0 5px #ff3860;
        animation: errorPulse 1.5s infinite;
    }

    @keyframes errorPulse {
        0% { opacity: 0.7; }
        50% { opacity: 1; }
        100% { opacity: 0.7; }
    }

    footer {
        margin-top: 30px;
        text-align: center;
        color: #6c757d;
        font-size: 14px;
```

```
        }

        /* Floating particles */
        .particle {
            position: absolute;
            background: var(--neon-blue);
            border-radius: 50%;
            filter: blur(1px);
            opacity: 0.7;
            animation: float linear infinite;
        }

        @keyframes float {
            0% { transform: translateY(0) rotate(0deg); }
            100% { transform: translateY(-100vh) rotate(360deg); }
        }
    </style>
</head>
<body>
    <div class="grid"></div>

    <!-- Generate floating particles -->
    <script>
        document.addEventListener('DOMContentLoaded', () => {
            for (let i = 0; i < 20; i++) {
                createParticle();
            }
        });

        function createParticle() {
            const particle = document.createElement('div');
            particle.classList.add('particle');

            // Random properties
            const size = Math.random() * 5 + 2;
            const posX = Math.random() * window.innerWidth;
            const duration = Math.random() * 20 + 10;
            const delay = Math.random() * 5;
            const color = `hsl(${Math.random() * 60 + 180}, 100%, 50%)`;

            particle.style.width = `${size}px`;
            particle.style.height = `${size}px`;
            particle.style.left = `${posX}px`;
            particle.style.bottom = `-10px`;
            particle.style.animationDuration = `${duration}s`;
            particle.style.animationDelay = `${delay}s`;
            particle.style.background = color;

            document.body.appendChild(particle);

            // Remove particle after animation completes
```

```
            setTimeout(() => {
                particle.remove();
                createParticle(); // Create new particle
            }, duration * 1000);
        }
    </script>

    <div class="login-container">
        <h1>Login</h1>

        <form action="LoginServlet" method="post">
            <div class="form-group">
                <label for="username">Username</label>
                <input type="text" id="username" name="username"
placeholder="Enter neon username" required>
            </div>

            <div class="form-group">
                <label for="password">Password</label>
                <input type="password" id="password" name="password"
placeholder="••••••••" required>
            </div>

            <button type="submit">ACCESS TERMINAL</button>
        </form>

        <div class="error-message" id="errorMessage">
            ACCESS DENIED: Invalid credentials
        </div>

        <footer>
            <p>NEON SESSION DEMO © 2023</p>
        </footer>
    </div>

    <script>
        // Show error message if URL has error parameter
        if (window.location.search.includes('error=true')) {
            document.getElementById('errorMessage').style.display = 'block';
        }

        // Add input focus effects
        const inputs = document.querySelectorAll('input');
        inputs.forEach(input => {
            input.addEventListener('focus', () => {
                input.style.boxShadow = `0 0 15px ${input.id === 'username' ?
'var(--neon-blue)' : 'var(--neon-pink)'}`;
            });
            input.addEventListener('blur', () => {
                input.style.boxShadow = 'none';
            });
```

```
        });
    </script>
</body>
</html>
```

OUTPUT: