Lab6 Exercise ITCSIU22266

Exercise 1:

	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation
1	id	int	no	4	10	0	no	(n/a)	(n/a)	NULL
2	username	varchar	no	50			no	no	no	SQL_Latin1_General_CP1_CI_AS
3	password	varchar	no	255			no	no	no	SQL_Latin1_General_CP1_CI_AS
4	full_name	varchar	no	100			no	no	no	SQL_Latin1_General_CP1_CI_AS
5	role	varchar	no	10			yes	no	yes	SQL_Latin1_General_CP1_CI_AS
6	is_active	bit	no	1			yes	(n/a)	(n/a)	NULL
7	created_at	datetime	no	8			yes	(n/a)	(n/a)	NULL
8	last_login	datetime	no	8			yes	(n/a)	(n/a)	NULL

	id	username	full_name	role	is_active
1	1	admin	Admin User	admin	1
2	2	john	John Doe	user	1
3	3	jane	Jane Smith	user	1

```
Plain: password123
Hashed: $2a$10$aGLB9rs/nXpSJGaSiqh97.3aQIlDzJFdAVmBe8NBrrnSJoWpnsgHe
Verification: true
```

Excerise 2:

```
// SQL Queries
private static final String SQL_AUTHENTICATE =
    "SELECT * FROM users WHERE username = ? AND is_active = 1";
```

Check if that user is active or not

```java
public User authenticate(String username, String password) {
    User user = null;

    try (Connection conn = getConnection();
         PreparedStatement pstmt = conn.prepareStatement(SQL_AUTHENTICATE)) {

        pstmt.setString(1, username);

        try (ResultSet rs = pstmt.executeQuery()) {
            if (rs.next()) {
                String hashedPassword = rs.getString("password");

                // Verify password with BCrypt
                if (BCrypt.checkpw(password, hashedPassword)) {
                    user = mapResultSetToUser(rs);

                    // Update last login time
                    updateLastLogin(user.getId());
                }
            }
        }

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

    return user;
}
```

After checking if that user is active, check password. Password in database is hashed. The password input by user will be hashed and checked if it is the same as password in db of that user.

```java
private User mapResultSetToUser(ResultSet rs) throws SQLException {
    User user = new User();
    user.setId(rs.getInt("id"));
    user.setUsername(rs.getString("username"));
    user.setPassword(rs.getString("password"));
    user.setFullName(rs.getString("full_name"));
    user.setRole(rs.getString("role"));
    user.setActive(rs.getBoolean("is_active"));
    user.setCreatedAt(rs.getTimestamp("created_at"));
    user.setLastLogin(rs.getTimestamp("last_login"));
    return user;
}
```

If yes, map the result with user id. And return user information. Then update the lastest login of that user.

```
private void updateLastLogin(int userId) {
    try (Connection conn = getConnection();
         PreparedStatement pstmt = conn.prepareStatement(SQL_UPDATE_LAST_LOGIN)) {

        pstmt.setInt(1, userId);
        pstmt.executeUpdate();

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

```
private static final String SQL_UPDATE_LAST_LOGIN =
    "UPDATE users SET last_login = GETDATE() WHERE id = ?";
```

```
--- exec:3.1.0:exec (default-cli) @ student-management-mvc ---
Authentication successful!
User{id=1, username='admin', fullName='Admin User', role='admin', isActive=true}
Invalid auth: Correctly rejected
------------------------------------------------------------------------
```

Exercise 3:

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

    // If already logged in, redirect to dashboard
    HttpSession session = request.getSession(false);
    if (session != null && session.getAttribute("user") != null) {
        response.sendRedirect("dashboard");
        return;
    }

    // Show login page
    request.getRequestDispatcher("login.jsp").forward(request, response);
}
```

If user has already login with session still available, redirect to dashboard page.

Else, show login page.

```java
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

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

    // Validate input
    if (username == null || username.trim().isEmpty() ||
        password == null || password.trim().isEmpty()) {

        request.setAttribute("error", "Username and password are required");
        request.getRequestDispatcher("login.jsp").forward(request, response);
        return;
    }

    // Authenticate user
    User user = userDAO.authenticate(username, password);

    if (user != null) {
        // Authentication successful

        // Invalidate old session (prevent session fixation)
        HttpSession oldSession = request.getSession(false);
        if (oldSession != null) {
            oldSession.invalidate();
        }
```

Get params from login page and validate the input.

Then call authenticate function with username and password.

If User is authenticated, clear the old session, and create new session.

```java
if (user != null) {
    // Authentication successful

    // Invalidate old session (prevent session fixation)
    HttpSession oldSession = request.getSession(false);
    if (oldSession != null) {
        oldSession.invalidate();
    }

    // Create new session
    HttpSession session = request.getSession(true);
    session.setAttribute("user", user);
    session.setAttribute("role", user.getRole());
    session.setAttribute("fullName", user.getFullName());

    // Set session timeout (30 minutes)
    session.setMaxInactiveInterval(30 * 60);

    // Handle "Remember Me" (optional - cookie implementation)
    if ("on".equals(rememberMe)) {
        // TODO: Implement remember me functionality with cookie
    }

    // Redirect based on role
    if (user.isAdmin()) {
        response.sendRedirect("dashboard");
    } else {
        response.sendRedirect("student?action=list");
    }
```

If User role is admin, redirect to admin dashboard, else, redirect to student page.

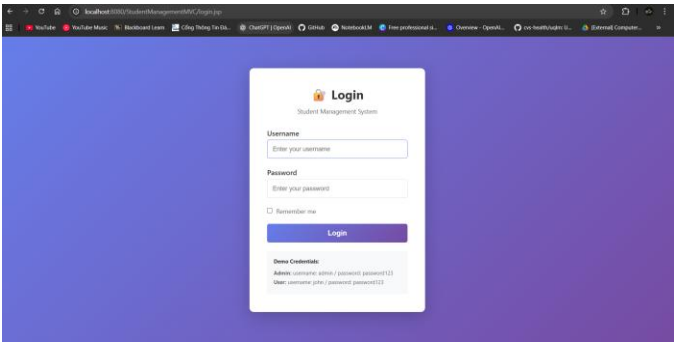
Else, show login page again with error message

```
} else {
    // Authentication failed
    request.setAttribute("error", "Invalid username or password");
    request.setAttribute("username", username); // Keep username in form
    request.getRequestDispatcher("login.jsp").forward(request, response);
}
```

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

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

        // Get current session
        HttpSession session = request.getSession(false);

        if (session != null) {
            // Invalidate session
            session.invalidate();
        }

        // Redirect to login page with message
        response.sendRedirect("login?message=You have been logged out successfully");
    }

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        doGet(request, response);
    }
}
```

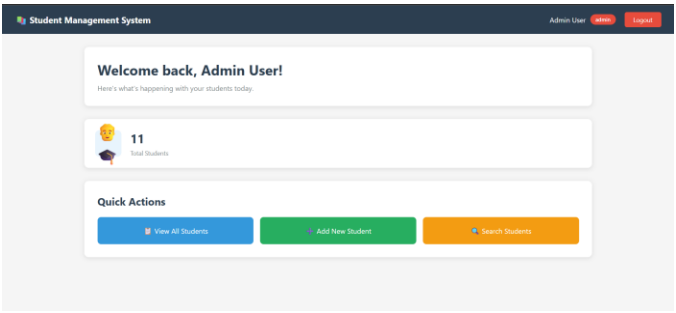
Login check the current session, then invalidate it, then redirect to login page.

Exercise 4:



User fills input with username and password. Check remember me to save the cookie for next session.

If user role is admin, navigate to dashboard.jsp



Dashboard get method check if the session is still available or not. If not, redirect to login page for a new session.

```java
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

    // Get user from session
    HttpSession session = request.getSession(false);
    if (session == null || session.getAttribute("user") == null) {
        response.sendRedirect("login");
        return;
    }

    User user = (User) session.getAttribute("user");

    // Get statistics
    int totalStudents = studentDAO.getTotalStudents();

    // Set attributes
    request.setAttribute("totalStudents", totalStudents);
    request.setAttribute("welcomeMessage", "Welcome back, " + user.getFullName() + "!");

    // Forward to dashboard
    request.getRequestDispatcher("dashboard.jsp").forward(request, response);
}
```

Else if user role is student, navigate to student page.