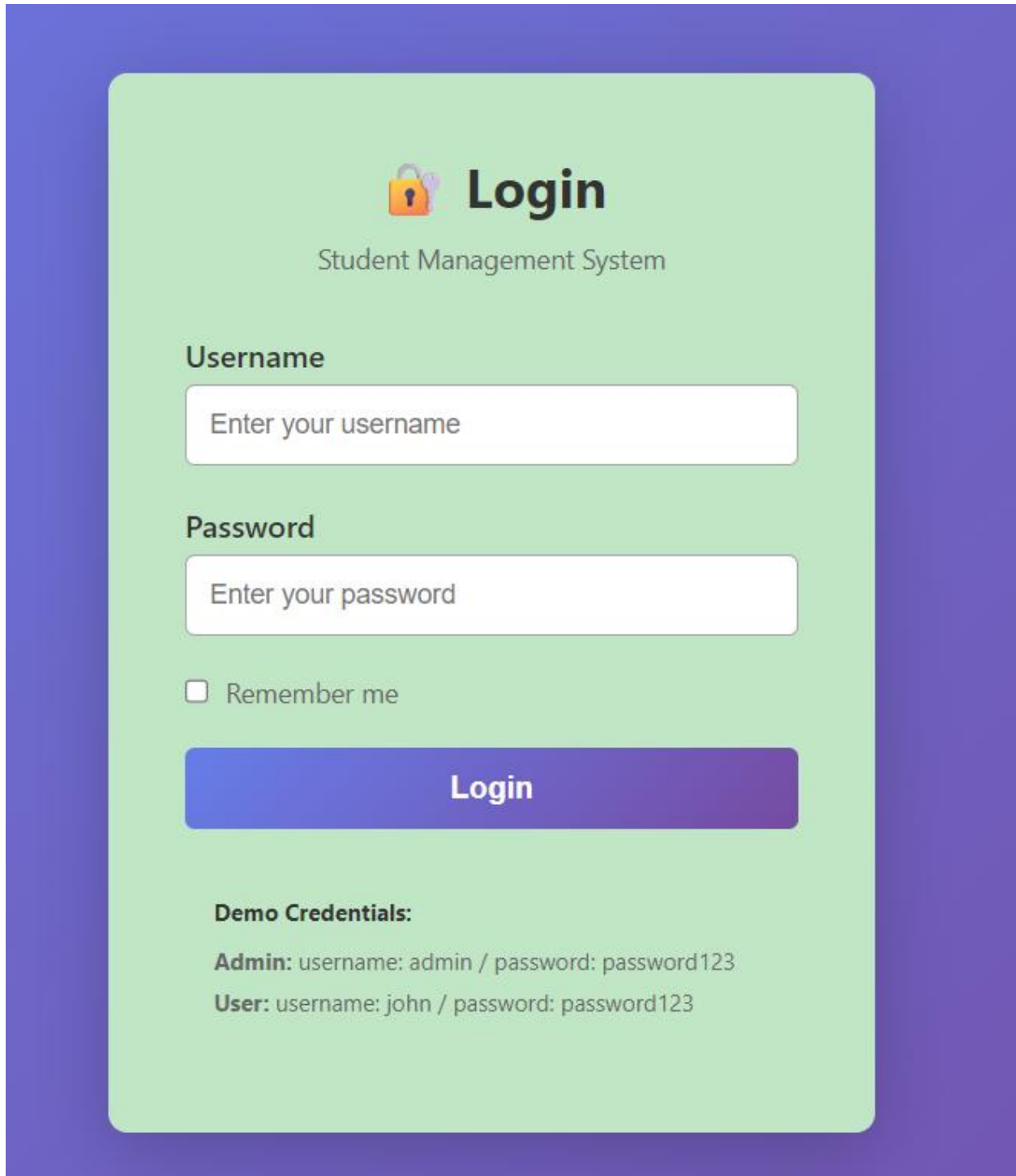



Login page



The login page features a light green rounded rectangle centered on a purple gradient background. At the top of the green area is a yellow padlock icon with a keyhole, followed by the word "Login" in a large, bold, black font. Below this is the text "Student Management System" in a smaller, gray font. The form includes two input fields: "Username" and "Password", each with a placeholder text "Enter your username" and "Enter your password" respectively. Below the password field is a checkbox labeled "Remember me". A large, blue-to-purple gradient button labeled "Login" is positioned below the checkbox. At the bottom of the green area, the text "Demo Credentials:" is followed by two lines of text: "Admin: username: admin / password: password123" and "User: username: john / password: password123".

 **Login**

Student Management System

Username

Enter your username

Password

Enter your password

☐ Remember me

Login

Demo Credentials:

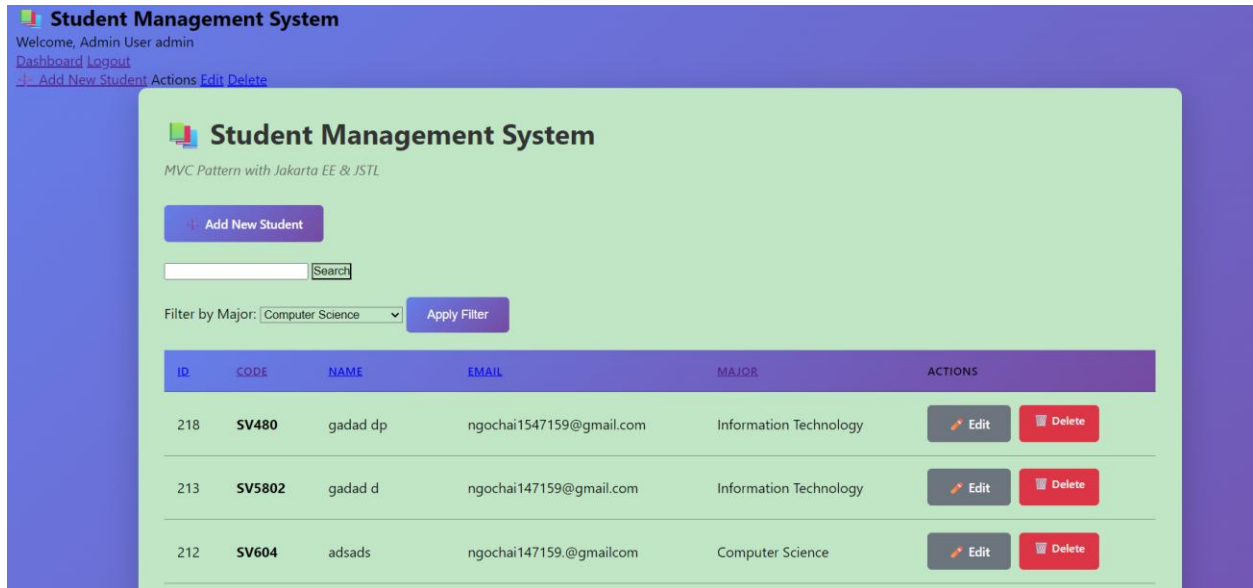
Admin: username: admin / password: password123

User: username: john / password: password123

Code can be seen In login controller, login UI in login.jsp

doGet checks the current session of the user, sends to the dashboard if the current session is logged in and the log in page if not.

doPost happens when the user submits their application, it checks whether the values are empty and uses authenticate() from userDAO to check if the hashed version(uses Bcrypt to hash) of the input matches the stored hashed password set to the user. If something is wrong, the page will inform the user and disallow the login. If not, the user logs in, whether as an admin or a normal user based on the role assigned.



The student list as an admin, with the edit and delete button viewable due to `<c:if test='${sessionScope.role eq 'admin'}'>`



Edit Student

Student Code *

SV480

Format: 2 letters + 3+ digits

Full Name *

gadad dp

Email *

ngochai1547159@gmail.com

Major *

Information Technology



Update Student



Cancel

You can only see this page as an admin due to it requiring the admin role to access



Student Management System

MVC Pattern with Jakarta EE & JSTL


[Add New Student](#)

Filter by Major: Computer Science

ID	CODE	NAME	EMAIL	MAJOR	ACTIONS
218	SV480	gadam dp	ngochai1547159@gmail.com	Information Technology	
213	SV5802	gadam d	ngochai147159@gmail.com	Information Technology	
212	SV604	adsads	ngochai147159@gmail.com	Computer Science	
210	SV580	gadam da	ngochai147159@gmail.com	Software Engineering	
209	SV024	Le Van W	w.le@example.com	Business Administration	
208	SV023	Tran Thi V	v.tran@example.com	Software Engineering	
207	SV022	Nguyen Van U	u.nguyen@example.com	Information Technology	


The student list for users, missing the edit and delete button due to requiring the role to be admin

localhost:8080/Student%20Management%20MVC/dashboard


Student Management System
John Doe user [Logout](#)

Welcome back, John Doe!

Here's what's happening with your students today.

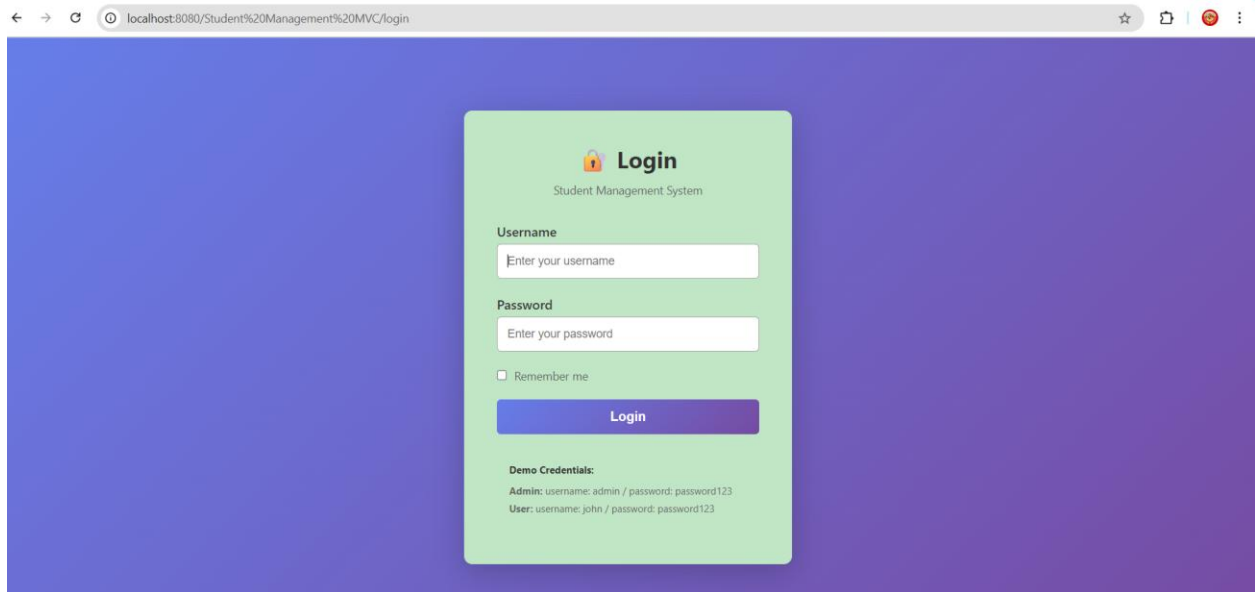
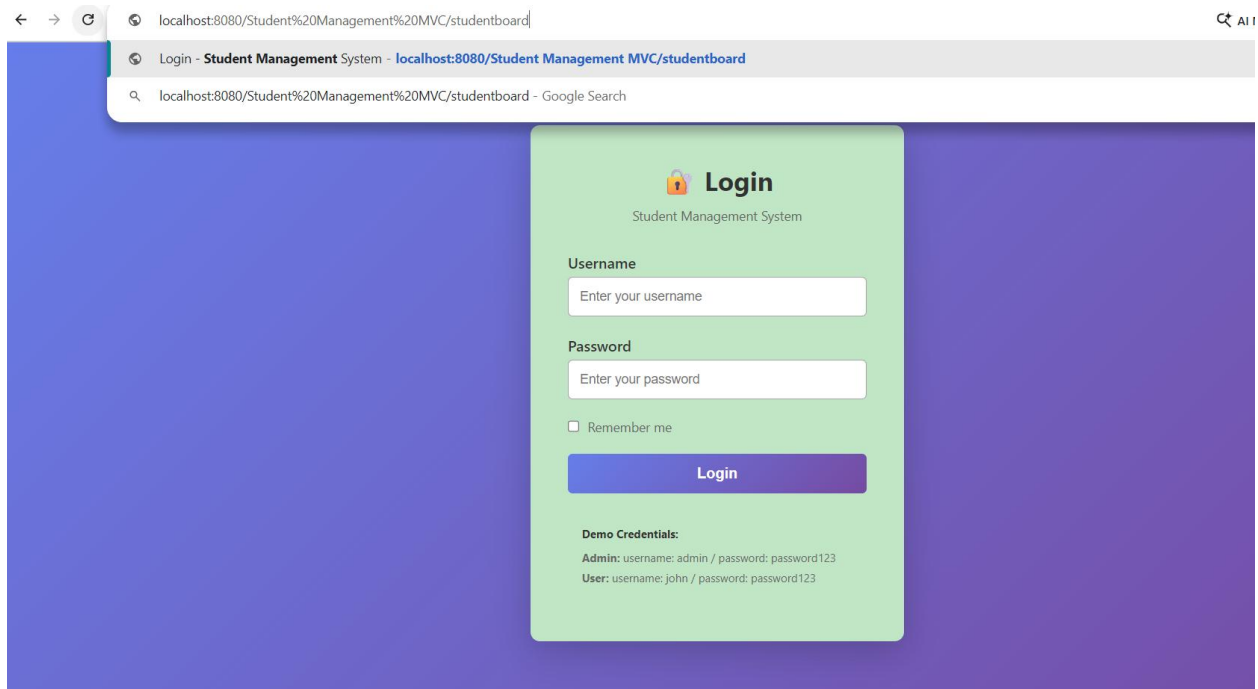

28
 Total Students

Quick Actions

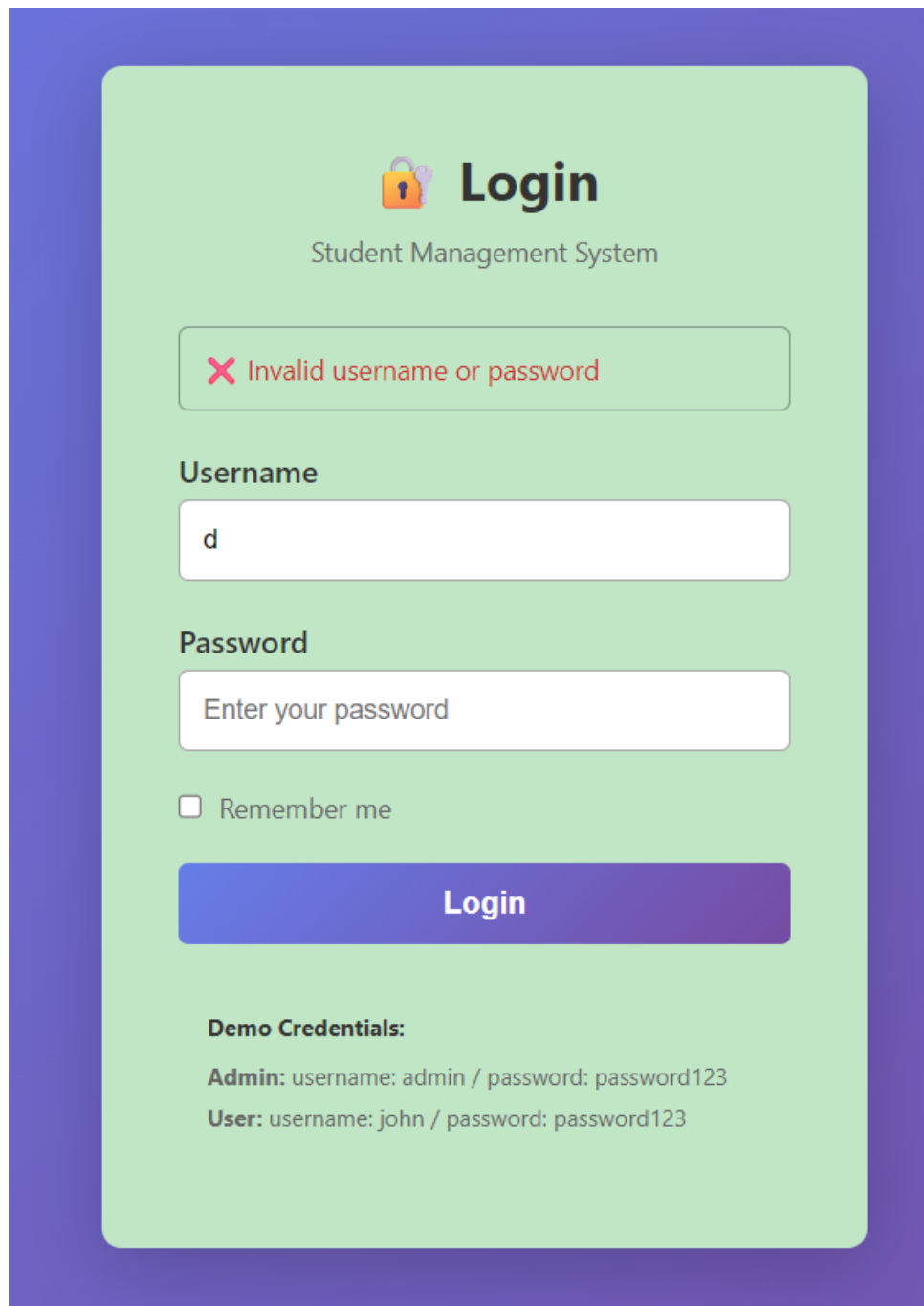
[View All Students](#)
[Search Students](#)

Dashboard after login, missing Add New Student due to not fulfilling the conditions of being an admin for that button


You also get redirected to this page as a user if you don't have the right for certain actions, adding new student in this case



Accessing a page that requires logging in without being logged in redirects to the login page, this is due to `response.sendRedirect("login")` in every page relending users to the login page if they aren't logged in.



The image shows a login interface for a 'Student Management System'. At the top, there is a lock icon and the word 'Login'. Below this is the system name. A red error message 'Invalid username or password' is displayed in a box. The 'Username' field contains the letter 'd'. The 'Password' field is empty with the placeholder text 'Enter your password'. There is a 'Remember me' checkbox and a 'Login' button. At the bottom, 'Demo Credentials' are listed: Admin (username: admin / password: password123) and User (username: john / password: password123).

 **Login**

Student Management System

✖ Invalid username or password

Username

d

Password

Enter your password

☐ Remember me

Login

Demo Credentials:

Admin: username: admin / password: password123

User: username: john / password: password123

Typing in an invalid username and password combination leads to this notification, added via `request.setAttribute("error", "Invalid username or password");` in `LoginController`



Login

Student Management System

☒ You have been logged out successfully

Username

Enter your username

Password

Enter your password

☐ Remember me

Login

Demo Credentials:

Admin: username: admin / password: password123

User: username: john / password: password123

Logging out leads to the login page with a confirmation at the top, added using
response.sendRedirect("login?message=You have been logged out successfully"); in
LogoutController

Changing Password

```
public class ChangePasswordController extends HttpServlet {

    private UserDao userDao = new UserDao();

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

        // User MUST be logged in to change password
        HttpSession session = request.getSession(false);
        if (session == null || session.getAttribute("currentUser") == null) {
            response.sendRedirect("login");
            return;
        }

        request.getRequestDispatcher("/views/change-password.jsp").forward(request,
response);
    }

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

```
HttpSession session = request.getSession(false);
```

```
if (session == null) {
```

```
    response.sendRedirect("login");
```

```
    return;
```

```
}
```

```
User currentUser = (User) session.getAttribute("currentUser");
```

```
// Read form values
```

```
String currentPassword = request.getParameter("currentPassword");
```

```
String newPassword = request.getParameter("newPassword");
```

```
String confirmPassword = request.getParameter("confirmPassword");
```

```
// Validate fields
```

```
if (currentPassword == null || newPassword == null || confirmPassword == null ||
```

```
    currentPassword.isBlank() || newPassword.isBlank() || confirmPassword.isBlank()) {
```

```
    request.setAttribute("error", "Please fill in all fields.");
```

```
    request.getRequestDispatcher("/views/change-password.jsp").forward(request,  
response);
```

```
    return;
```

```
}
```

```
// Validate current password
```

```
if (!currentPassword.equals(currentUser.getPassword())) {
```

```
        request.setAttribute("error", "Incorrect current password.");  
        request.getRequestDispatcher("/views/change-password.jsp").forward(request,  
response);  
        return;  
    }  
}
```

```
// Validate new password length  
if (newPassword.length() < 8) {  
    request.setAttribute("error", "New password must be at least 8 characters long.");  
    request.getRequestDispatcher("/views/change-password.jsp").forward(request,  
response);  
    return;  
}  
}
```

```
// Validate confirm password  
if (!newPassword.equals(confirmPassword)) {  
    request.setAttribute("error", "New password and confirm password do not match.");  
    request.getRequestDispatcher("/views/change-password.jsp").forward(request,  
response);  
    return;  
}  
}
```

```
// Hash new password  
String hashed = BCrypt.hashpw(newPassword, BCrypt.gensalt());  
  
// Update password in database  
userDAO.updatePassword(currentUser.getId(), hashed);
```

```
// Refresh session user

currentUser.setPassword(newPassword);

session.setAttribute("currentUser", currentUser);


request.setAttribute("success", "Password changed successfully.");

request.getRequestDispatcher("/views/change-password.jsp").forward(request,
response);

}

}
```

Checks if the passwords entered are empty, validate if the current password entered is actually the current password. Then checks if the confirm password is the same as the new password. If all goes well, the new password is hashed using Bcrypt and saved as the current password of the User