

Exercise 5: AuthFilter

- Allow access to public urls:

The screenshot shows a browser window with the URL `localhost:8080/student-management-mvc/login`. The page has a purple header and a white content area. It features a yellow padlock icon and the word "Login". Below it is the text "Student Management System". There are two input fields: one for "Username" with placeholder text "Enter your username" and another for "Password" with placeholder text "Enter your password". A checkbox labeled "Remember me" is present. A large blue "Login" button is at the bottom. At the bottom of the page, there is a box containing "Demo Credentials": "Admin: username: admin / password: password123" and "User: username: john / password: password123".

- Redirect to login if url is not public:

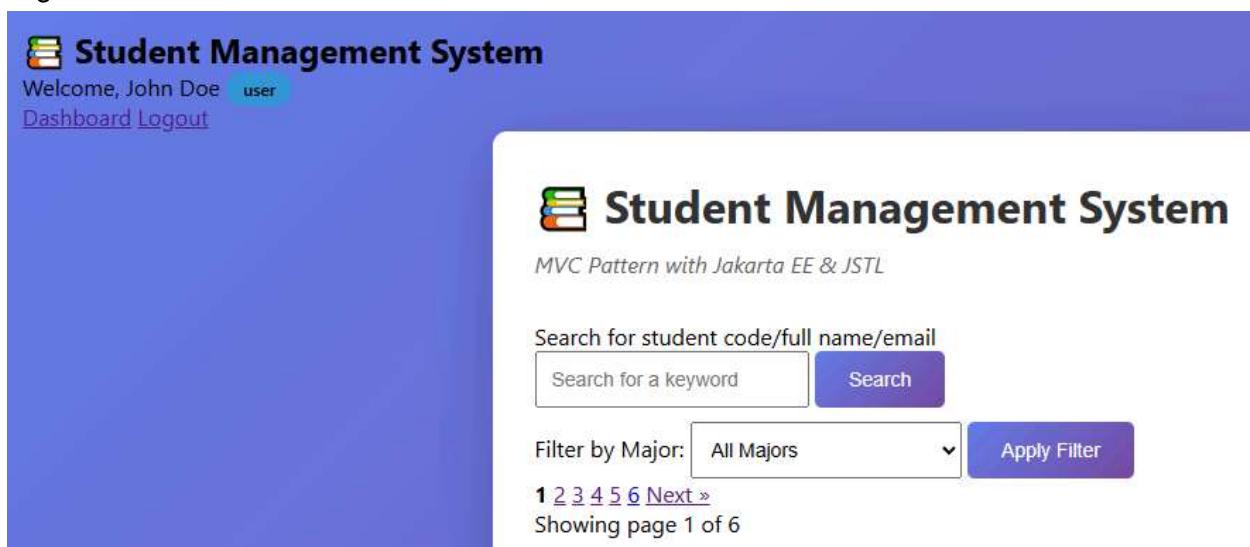
The screenshot shows a browser window with the URL `localhost:8080/student-management-mvc/dashboard`. The page displays a message: "You must be logged in to view this page. Please log in or register." Below this message is a link to "Log In". At the bottom of the page, there is a link to "Logout".

- How it works: When accessing public urls, the filter will allow the request to reach the servlet. When accessing non-public paths like /dashboard or /student, the filter will check first if a session exists, if there is a session that means the user is logged in and the filter will let the request go to the servlet.

Exercise 6: AdminFilter

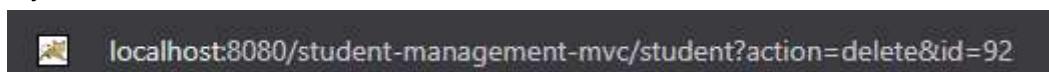
- How it works: Whenever the client sends a request with a url that begins with /student, the AdminFilter will intercept the request to check if the action parameter is admin action or not. Then if the action is an admin action, the filter will check if a session exists (logged in or not) and then check if the user of that session has an admin role or not. If the user is admin then the filter will let the request go to the servlet, otherwise the filter will redirect to list-student.jsp with an error message.
- Example:

Log in as normal user:



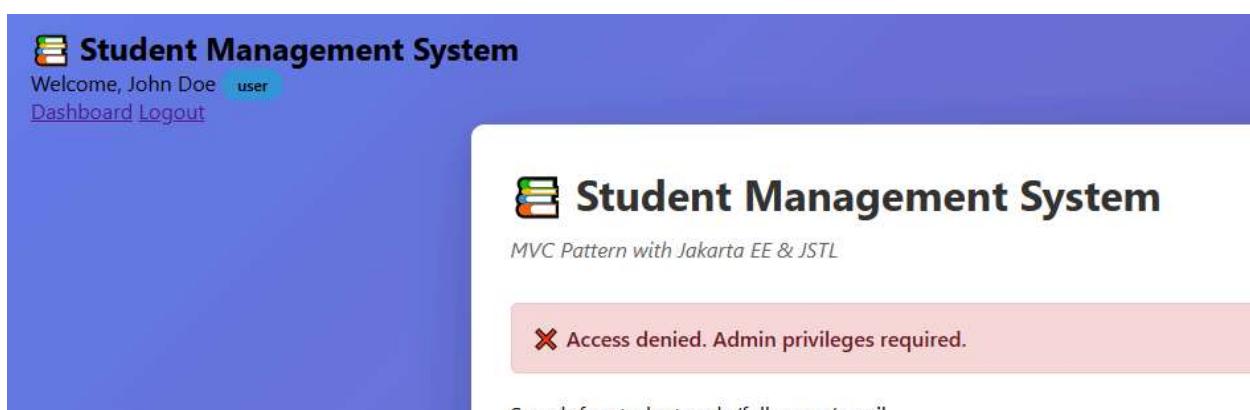
The screenshot shows the 'Student Management System' login page. The header includes the logo, the title 'Student Management System', and a welcome message 'Welcome, John Doe' followed by a 'user' status indicator. Below the header are links for 'Dashboard' and 'Logout'. The main content area displays the search interface with a search bar, a dropdown for 'Filter by Major', and a 'Search' button. Below the search area, there are navigation links (1, 2, 3, 4, 5, 6, Next ») and a message indicating 'Showing page 1 of 6'.

Try to do delete action in search bar:



The screenshot shows a browser address bar with the URL 'localhost:8080/student-management-mvc/student?action=delete&id=92'. This URL corresponds to a delete operation on a student record with ID 92, which is only accessible to admins.

Denied:



The screenshot shows the 'Student Management System' search interface. A red error message box at the top states 'Access denied. Admin privileges required.' Below the message, there is a search bar with the placeholder 'Search for student code/full name/email'.

Exercise 7: Role-based UI

- For normal users, no action column and add student button:

Student Management System

Welcome, John Doe user

Dashboard Logout

Student Management System

MVC Pattern with Jakarta EE & JSTL

Access denied. Admin privileges required.

Search for student code/full name/email

Search for a keyword

Filter by Major: All Majors

1 2 3 4 5 6 Next »

Showing page 1 of 6

ID	CODE	NAME	EMAIL	MAJOR
92	CR520	Tina Marsh	tina.marsh@example.com	Computer Science
91	CR519	Steven Cole	steven.cole@example.com	Computer Science

- For admin users, all functionalities:

Student Management System

Welcome, Admin User admin

Dashboard Logout

Student Management System

MVC Pattern with Jakarta EE & JSTL

Add New Student

Search for student code/full name/email

Search for a keyword

Filter by Major: All Majors

1 2 3 4 5 6 Next »

Showing page 1 of 6

ID	CODE	NAME	EMAIL	MAJOR	ACTIONS
92	CR520	Tina Marsh	tina.marsh@example.com	Computer Science	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
91	CR519	Steven Cole	steven.cole@example.com	Computer Science	<input type="button" value="Edit"/> <input type="button" value="Delete"/>

Exercise 8: Add change password functionality

- Add change password button to dashboard:

Admin User admin

Logout

Change Password

- Change password page:

Change Password

Current Password *

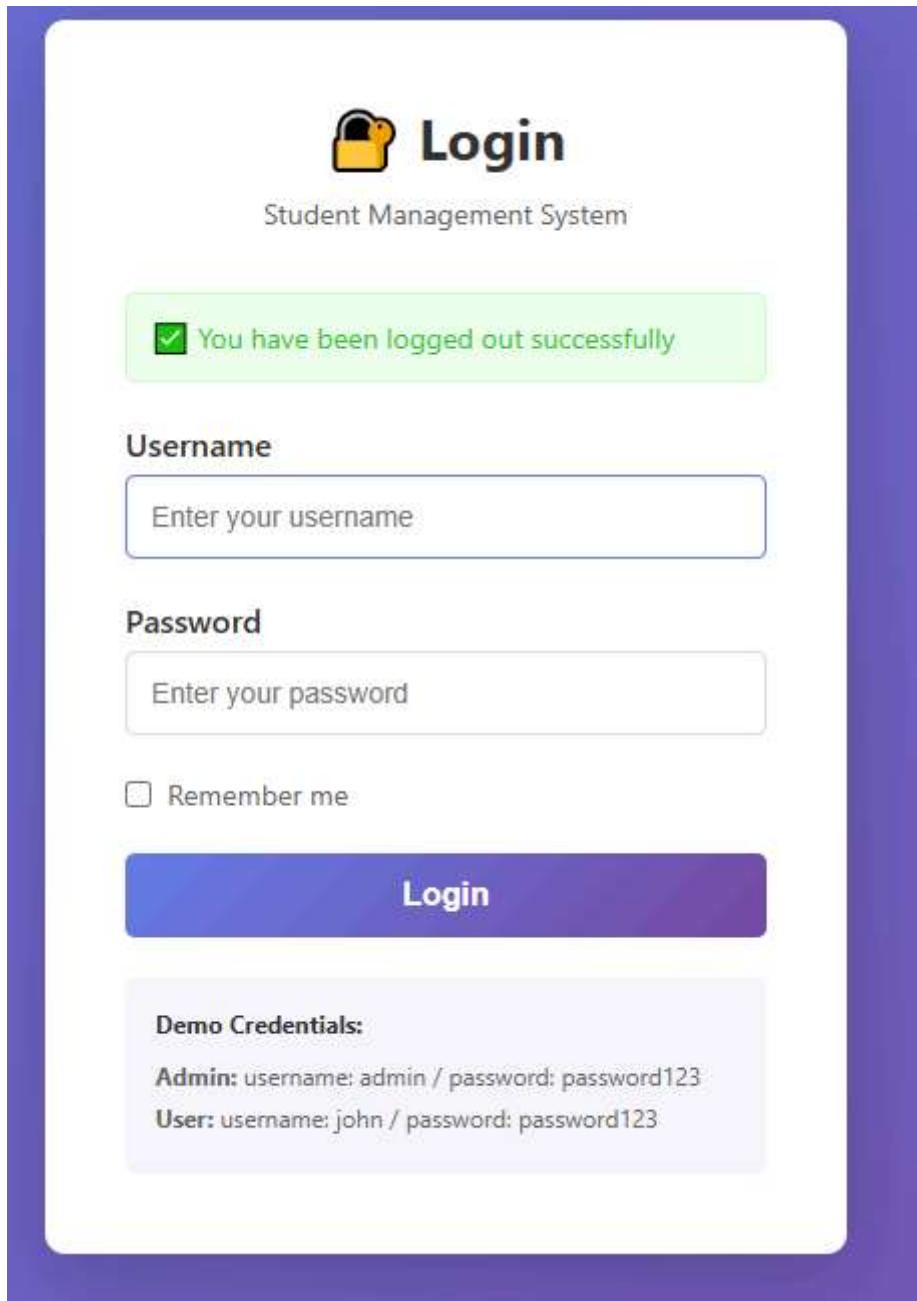
New Password *

Confirm Password *

Change Password

 Cancel

- Logout after changing password:



- How this works: When logged in, any user can click the change password button in the dashboard. That will send a GET request to the change password controller and the controller will forward to the change password page. After filling in the fields and pressing Change Password, a POST request will be sent to the controller, which will first check if the user is logged in, then validate all fields and finally invoke userDAO.updatePassword with the hashed value of the new password. If the method returns true, the controller will logout the user.