

VIETNAM NATIONAL UNIVERSITY OF HO CHI MINH CITY

THE INTERNATIONAL UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING



**Web Application Development - IT093IU**

Lab 6: AUTHENTICATION & SESSION MANAGEMENT

Nguyễn Vũ Thành Tính - ITCSIU23039

**Lab Instructor:** N.T.Nghia

<b>Test Login Flow.....</b>	<b>3</b>
1. Access homepage.....	3
2. Logging in as Admin (admin / password123).....	3
3. Viewing the Student List.....	4
4. Logging Out.....	5
5. Logging in as Regular User (john / password123).....	6
6. Attempting to Access Admin-Only Function.....	6
<b>Test URLs.....</b>	<b>7</b>
1. Public URLs (no login required).....	7
2. Protected URLs (login required).....	9
3. Admin-only URLs (must be logged in and have admin role).....	11

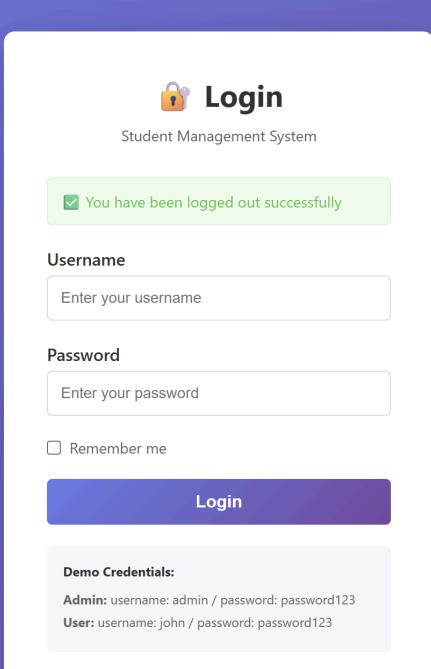
## Github Links:

[https://github.com/ngvuthinh/Web\\_Lab/tree/main/Lab\\_6/Student%20Management%20MVC%20Lab6](https://github.com/ngvuthinh/Web_Lab/tree/main/Lab_6/Student%20Management%20MVC%20Lab6)

## Test Login Flow

### 1. Access homepage

- User accesses: `http://localhost:8080/StudentManagementMVC/`
- The request passes through AuthFilter (file: `filter/AuthFilter.java`).
- AuthFilter checks whether a session exists:
  - If session is null, user is not logged in.
    - AuthFilter redirects the user to "login".
- The redirected request is handled by `LoginController.doGet()` (file: `controller/LoginController.java`).
- `LoginController` forwards the request to: `views/login.jsp`



### 2. Logging in as Admin (admin / password123)

- User submits login form in `login.jsp` with `action="login"` `method="post"`.
- Request is processed by `LoginController.doPost()`.
- Controller retrieves:

- `username = request.getParameter("username")`
  - `password = request.getParameter("password")`
- Controller calls `UserDAO.authenticate(username, password)`
- (File: `dao/UserDAO.java`)
- Inside `authenticate()`:
  - SQL executed: `SELECT * FROM users WHERE username = ? AND is_active = TRUE`
  - If a record exists:
    - Retrieve hashed password
    - Validate with `BCrypt.checkpw()`
  - If validation succeeds, `User` object is returned.
- LoginController creates a new session: `session.setAttribute("user", user)`
- If user role is "admin", LoginController redirects to "/dashboard".
- Request reaches DashboardController.`doGet()` → forwards to `views/dashboard.jsp`

The screenshot shows the 'Student Management System' dashboard. At the top, there is a header with the system name, a 'Logout' button, and a 'Logout' link. The main content area has a 'Welcome back, Admin User!' message and a sub-message 'Here's what's happening with your students today.' Below this, there is a summary card showing a student icon, the number '20', and the text 'Total Students'. At the bottom, there is a 'Quick Actions' section with three buttons: 'View All Students' (blue), 'Add New Student' (green), and 'Search Students' (orange).

### 3. Viewing the Student List

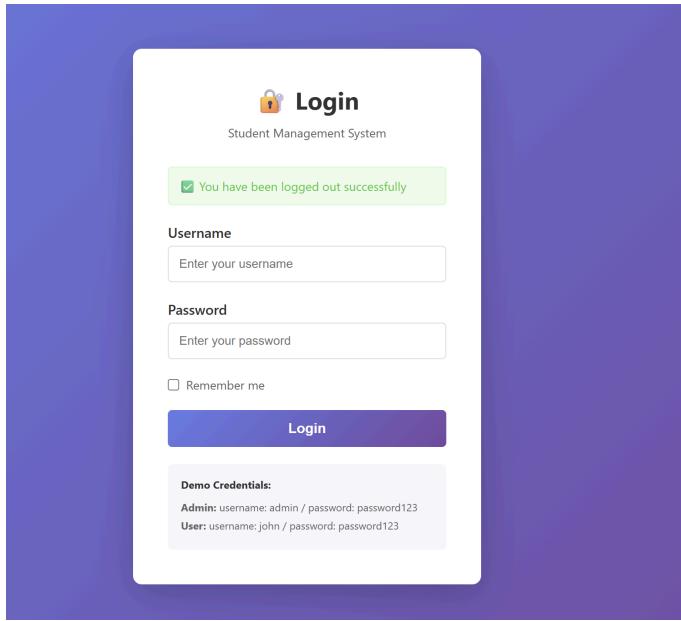
- Admin clicks "View All Students" which sends request to:  
`/student?action=list`
- Request passes through `AuthFilter` (session exists, so allowed).
- Request is handled by `StudentController.doGet()` (file: `controller/StudentController.java`).
- Controller checks action parameter:  
`action = "list"`
- Controller calls:  
`studentDAO.getAllStudents()`  
 (File: `dao/StudentDAO.java`)
- Inside `getAllStudents()`:
  - SQL executed: `SELECT * FROM students`
  - ResultSet is mapped into a `List<Student>`

- Controller forwards request to:  
views/student-list.jsp\*
- Because user role is admin:
  - Edit and Delete buttons are displayed.

ID	CODE	NAME	EMAIL	MAJOR	ACTIONS
1	SV010	Hoang Thuy Linh	linh.hoang@example.com	Computer Science	<a href="#">Edit</a> <a href="#">Delete</a>
2	SV011	Dang Van Lam	lam.dang@example.com	Information Technology	<a href="#">Edit</a> <a href="#">Delete</a>
3	SV012	Doan Van Hau	hau.doan@example.com	Software Engineering	<a href="#">Edit</a> <a href="#">Delete</a>
4	SV013	Nguyen Quang Hai	hai.nguyen@example.com	Business Administration	<a href="#">Edit</a> <a href="#">Delete</a>
5	SV014	Phan Van Duc	duc.phan@example.com	Computer Science	<a href="#">Edit</a> <a href="#">Delete</a>

## 4. Logging Out

- User clicks logout which sends request to:  
`/logout`
- Request hits LogoutController.doGet() (file: controller/LogoutController.java)
- Controller invalidates session using:  
`session.invalidate()`
- User is redirected back to `/login`
- AuthFilter again forces display of `login.jsp` because session is gone.



## 5. Logging in as Regular User (john / password123)

- User submits login through login.jsp
- LoginController.doPost() → UserDAO.authenticate() → BCrypt password check
- A "user" role is returned instead of "admin"
- LoginController redirects regular users to:  
/student?action=list
- StudentController loads student list and forwards to student-list.jsp
- In JSP, role is checked and:
  - Edit and Delete buttons are hidden

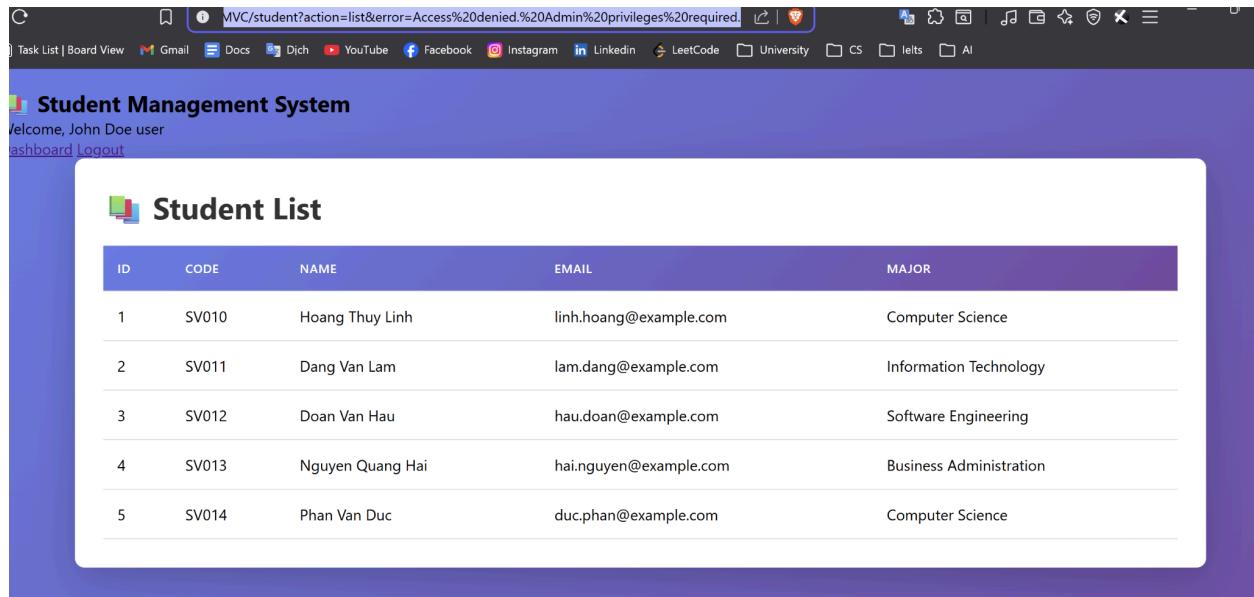
ID	CODE	NAME	EMAIL	MAJOR
1	SV010	Hoang Thuy Linh	linh.hoang@example.com	Computer Science
2	SV011	Dang Van Lam	lam.dang@example.com	Information Technology
3	SV012	Doan Van Hau	hau.doan@example.com	Software Engineering
4	SV013	Nguyen Quang Hai	hai.nguyen@example.com	Business Administration
5	SV014	Phan Van Duc	duc.phan@example.com	Computer Science

## 6. Attempting to Access Admin-Only Function

- When a regular user manually enters the URL:  
/student?action=new
- The request passes through AuthFilter (session exists) but then reaches AdminFilter.
- In AdminFilter.doFilter(), the system checks the user role:  
if (!user.isAdmin()) { block }
- Since the logged-in user has role "user", AdminFilter blocks the request.
- Depending on the implementation, the blocked request typically results in:
  - A forced redirect (e.g., back to login or dashboard), or
  - An empty/blank response, or
  - No access to the target JSP.
- Because no custom error page is provided, the browser URL will still display /student?action=new, but the content will not appear.

URL shows access denied:

http://localhost:8080/Student\_Management\_MVC/student?action=list&error=Access%20denied.%20Admin%20privileges%20required.



ID	CODE	NAME	EMAIL	MAJOR
1	SV010	Hoang Thuy Linh	linh.hoang@example.com	Computer Science
2	SV011	Dang Van Lam	lam.dang@example.com	Information Technology
3	SV012	Doan Van Hau	hau.doan@example.com	Software Engineering
4	SV013	Nguyen Quang Hai	hai.nguyen@example.com	Business Administration
5	SV014	Phan Van Duc	duc.phan@example.com	Computer Science

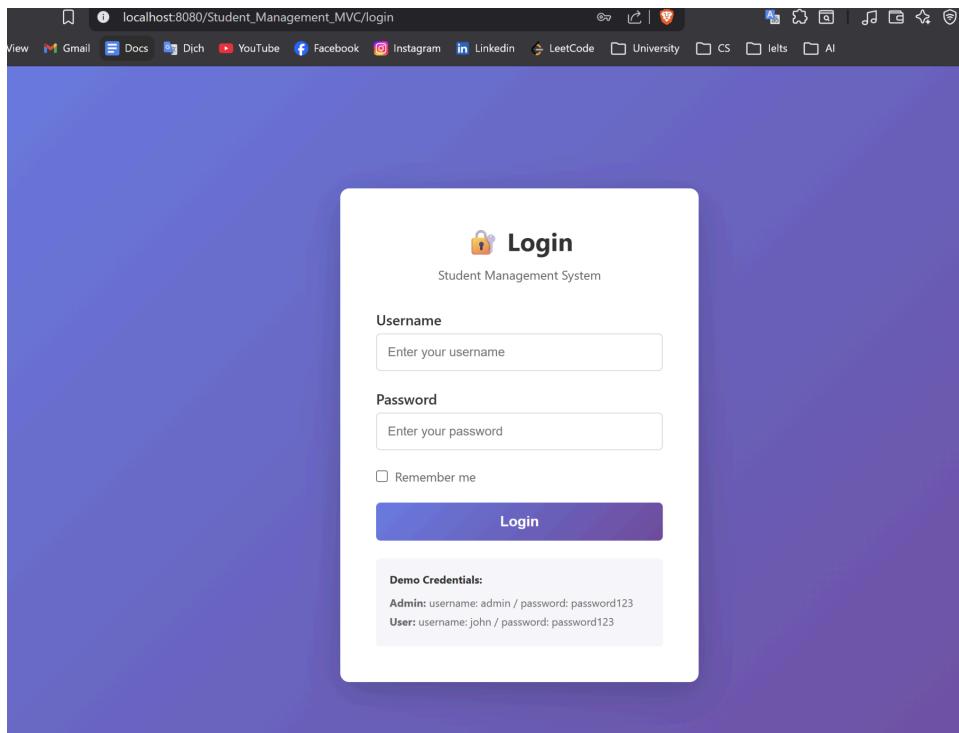
## Test URLs

### 1. Public URLs (no login required)

#### 1) URL: /login

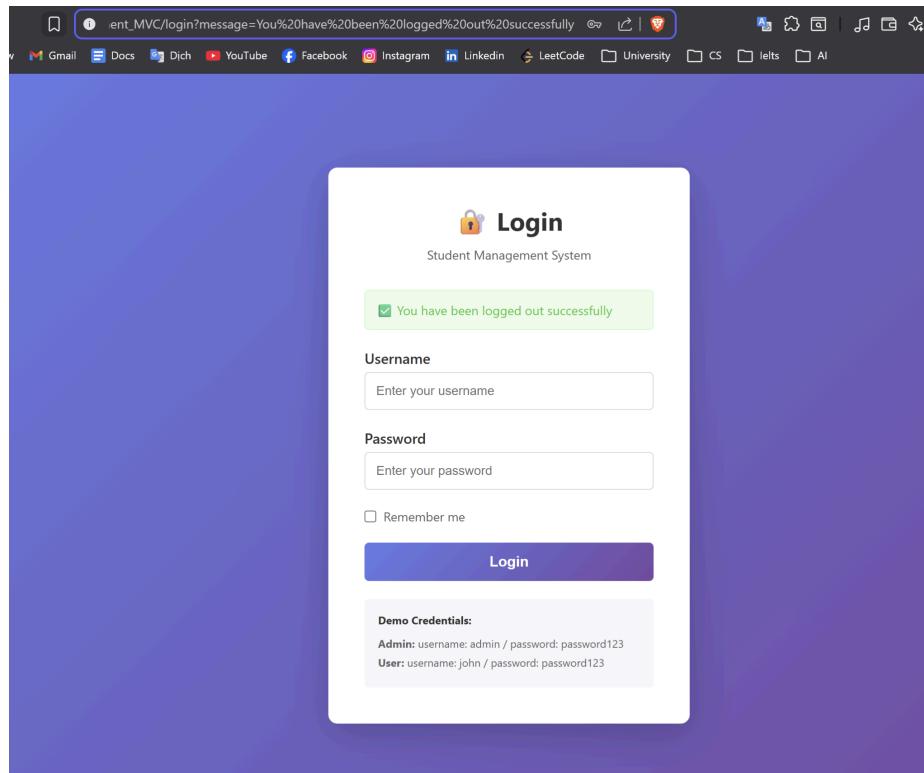
- This URL is mapped to LoginController (file: controller/LoginController.java).
- For a GET request:
  - LoginController.doGet() is executed.
  - It checks if there is already a logged-in user in session:

- If a session with attribute "user" exists, it redirects to "dashboard".
- Otherwise, it forwards the request to views/login.jsp to display the login form.
- For a POST request:
  - LoginController.doPost() handles the login form submission.
  - It reads the username and password parameters, then calls UserDAO.authenticate().
  - Based on the result, it either:
    - Creates a session and redirects to the appropriate page (dashboard or student list), or
    - Returns to login.jsp with an error message.



## 2) URL: /logout

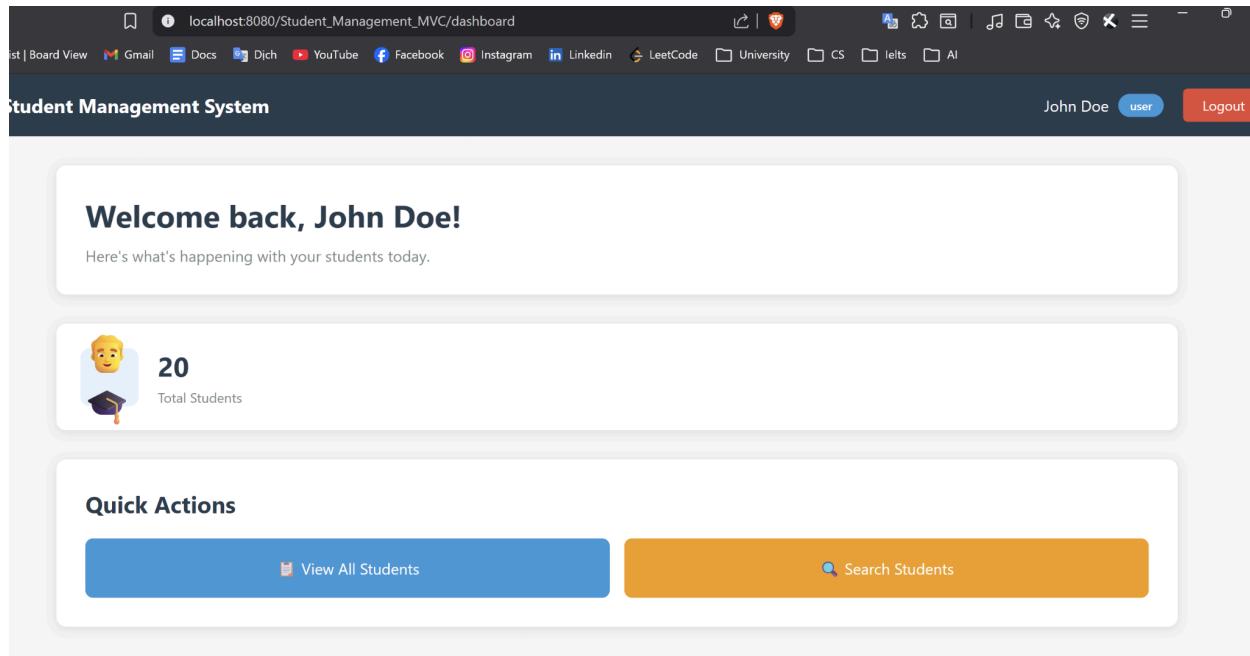
- This URL is mapped to LogoutController (file: controller/LogoutController.java).
- LogoutController.doGet() is called when /logout is requested.
- The controller does:
  - Retrieves the current session if it exists.
  - Invalidates the session using session.invalidate().
  - Redirects the user back to /login.
- After this, any protected URL will again be blocked by AuthFilter until the user logs in again.



## 2. Protected URLs (login required)

### 1) URL: /dashboard

- When user requests /dashboard, the request first passes through AuthFilter.
- AuthFilter:
  - Checks for an active session and the presence of session.getAttribute("user").
  - If no user in session, it redirects the request to /login.
  - If a user is present, it allows the request to continue.
- The request is then handled by DashboardController (file: controller/DashboardController.java).
- DashboardController.doGet():
  - May load any summary/statistics needed for the dashboard.
  - Forwards to the JSP page: views/dashboard.jsp.



## 2) URL: /student?action=list

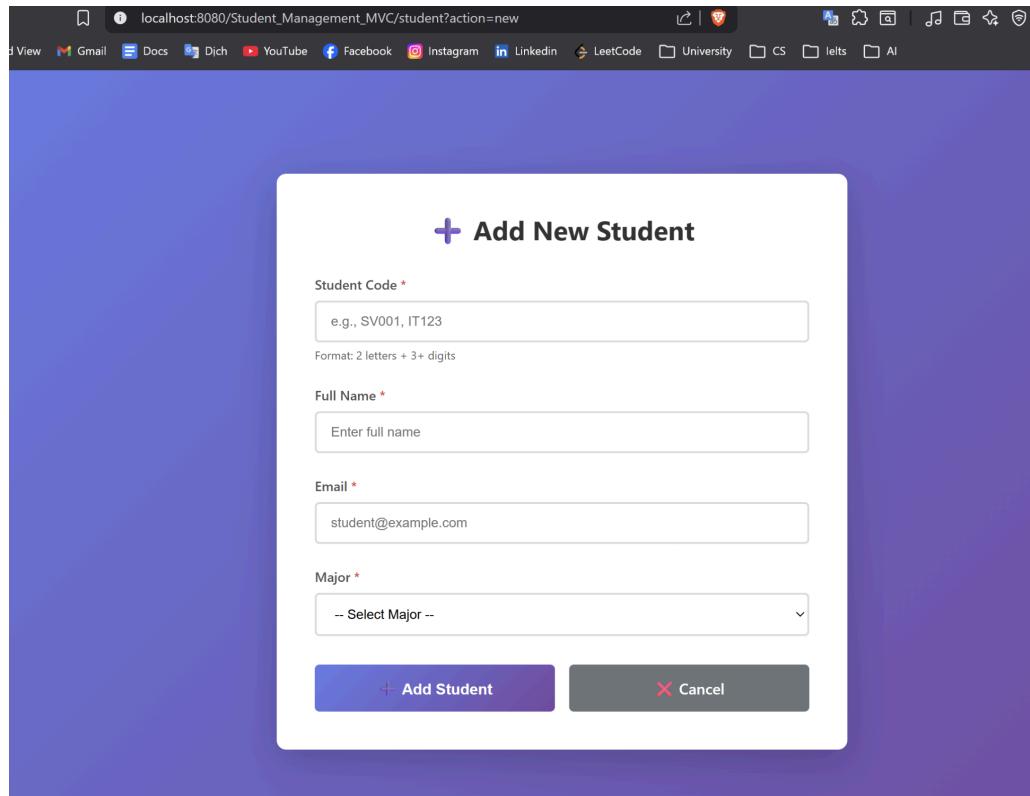
- This URL is also protected by AuthFilter.
  - If user is not logged in, AuthFilter redirects to /login.
  - If user is logged in (either admin or normal user), the request is allowed.
- The request is mapped to StudentController (file: controller/StudentController.java).
- StudentController doGet():
  - Reads action parameter from the request.
  - When action = "list":
    - Calls studentDAO.getAllStudents() in StudentDAO (file: dao/StudentDAO.java).
    - Retrieves the list of students from the students table in the database.
    - Attaches this list to the request as an attribute.
    - Forwards to views/student-list.jsp to render the data.
- On the JSP side:
  - The page renders the table of students.
  - It may show or hide admin actions (Edit/Delete) depending on the user role.

ID	CODE	NAME	EMAIL	MAJOR
1	SV010	Hoang Thuy Linh	linh.hoang@example.com	Computer Science
2	SV011	Dang Van Lam	lam.dang@example.com	Information Technology
3	SV012	Doan Van Hau	hau.doan@example.com	Software Engineering
4	SV013	Nguyen Quang Hai	hai.nguyen@example.com	Business Administration
5	SV014	Phan Van Duc	duc.phan@example.com	Computer Science

### 3. Admin-only URLs (must be logged in and have admin role)

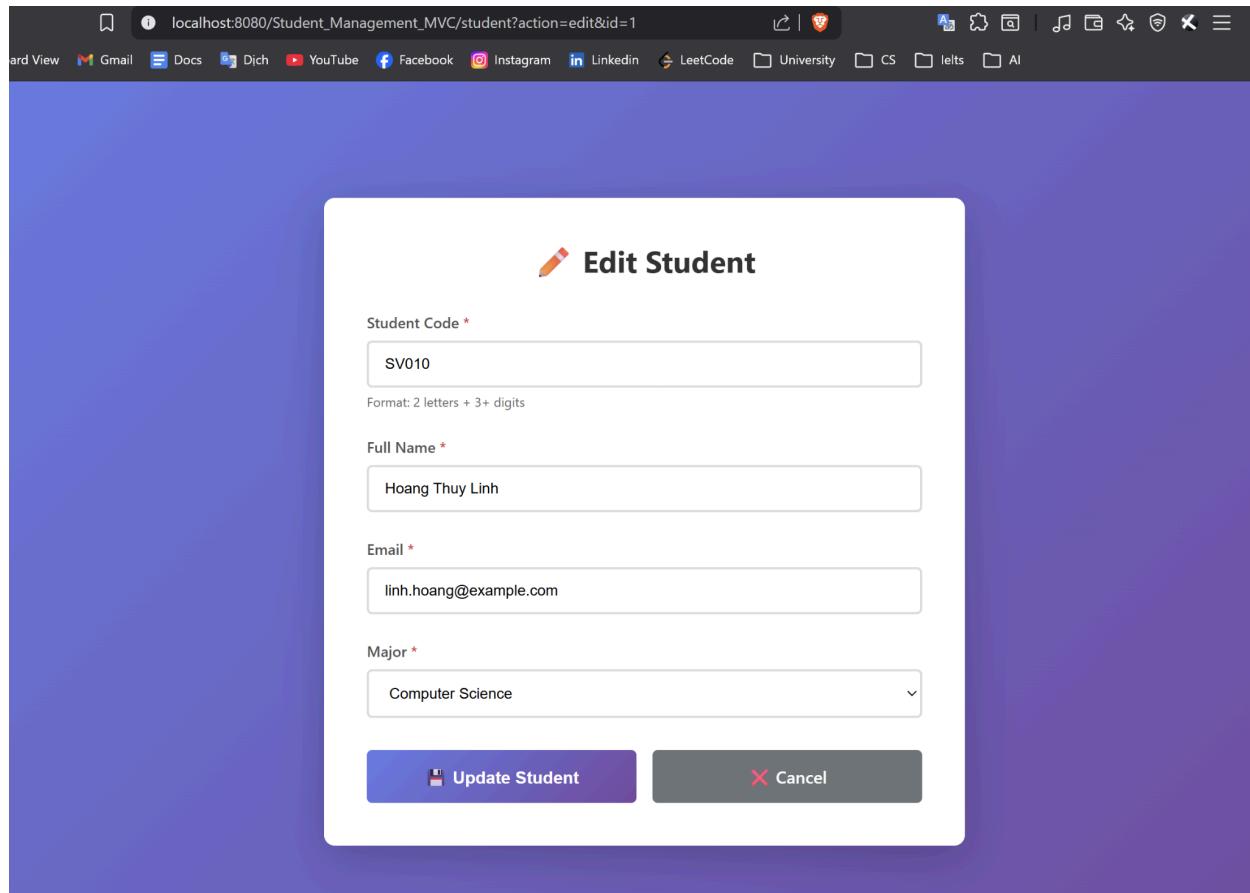
#### 1) URL: /student?action=new

- Request passes through AuthFilter:
  - If no session → redirected to /login.
  - If logged in → continues.
- Then request passes through AdminFilter (file: filter/AdminFilter.java):
  - Retrieves user from session.
  - If user role is not admin:
    - Access is blocked (redirect to another page or show error).
  - If user is admin:
    - Request continues to StudentController.
- In StudentController doGet():
  - If action = "new":
    - It forwards to views/student-form.jsp with an empty form for creating a new student.



## 2) URL: /student?action=edit&id=1

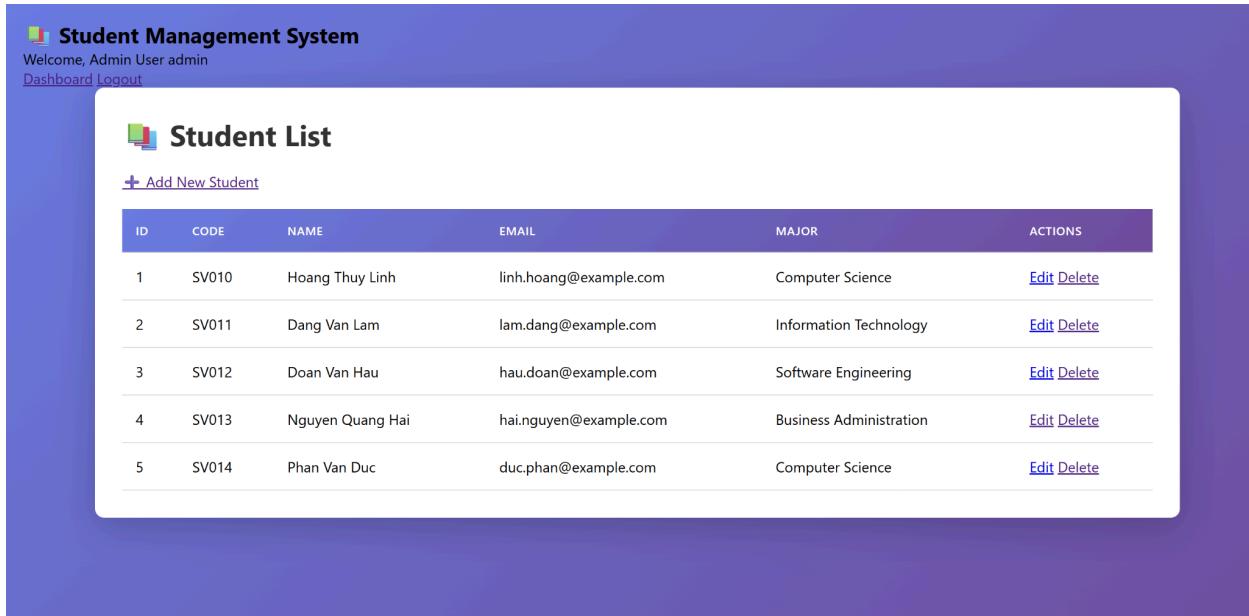
- Again, the request is filtered by AuthFilter → must be logged in.
- Then it is filtered by AdminFilter → must be admin.
- If checks pass, StudentController.doGet() is executed:
  - Reads parameters action="edit" and id.
  - Calls studentDAO.getStudentById(id) to fetch data from DB.
  - Sets the Student object as a request attribute.
  - Forwards to views/student-form.jsp, this time with fields pre-filled for editing.



### 3) URL: /student?action=delete&id=1

- Protected by both AuthFilter and AdminFilter:
  - User must be logged in.
  - User must have admin role.
- If allowed, StudentController.doGet() handles it:
  - Reads action="delete" and id.
  - Calls studentDAO.deleteStudent(id) to remove the record from DB.
  - Redirects back to /student?action=list to show updated student list.

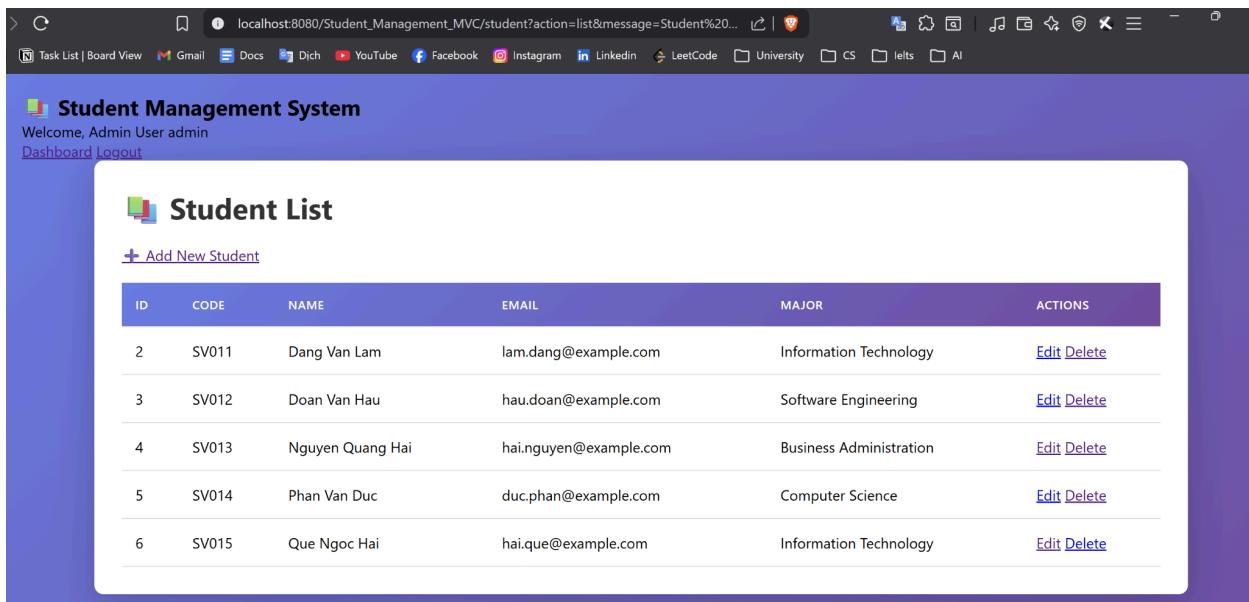
## Before



The screenshot shows the 'Student List' page of a web application. The header includes the logo 'Student Management System', a welcome message 'Welcome, Admin User admin', and navigation links 'Dashboard' and 'Logout'. The main content area is titled 'Student List' with a sub-link '+ Add New Student'. Below is a table with the following data:

ID	CODE	NAME	EMAIL	MAJOR	ACTIONS
1	SV010	Hoang Thuy Linh	linh.hoang@example.com	Computer Science	<a href="#">Edit</a> <a href="#">Delete</a>
2	SV011	Dang Van Lam	lam.dang@example.com	Information Technology	<a href="#">Edit</a> <a href="#">Delete</a>
3	SV012	Doan Van Hau	hau.doan@example.com	Software Engineering	<a href="#">Edit</a> <a href="#">Delete</a>
4	SV013	Nguyen Quang Hai	hai.nguyen@example.com	Business Administration	<a href="#">Edit</a> <a href="#">Delete</a>
5	SV014	Phan Van Duc	duc.phan@example.com	Computer Science	<a href="#">Edit</a> <a href="#">Delete</a>

## After



The screenshot shows the 'Student List' page of the web application. The browser's address bar indicates the URL is 'localhost:8080/Student\_Management\_MVC/student?action=list&message=Student%20...'. The page structure is identical to the 'Before' screenshot, with the 'Student Management System' header, 'Welcome, Admin User admin' message, and 'Dashboard' link. The 'Student List' table now contains the following data:

ID	CODE	NAME	EMAIL	MAJOR	ACTIONS
2	SV011	Dang Van Lam	lam.dang@example.com	Information Technology	<a href="#">Edit</a> <a href="#">Delete</a>
3	SV012	Doan Van Hau	hau.doan@example.com	Software Engineering	<a href="#">Edit</a> <a href="#">Delete</a>
4	SV013	Nguyen Quang Hai	hai.nguyen@example.com	Business Administration	<a href="#">Edit</a> <a href="#">Delete</a>
5	SV014	Phan Van Duc	duc.phan@example.com	Computer Science	<a href="#">Edit</a> <a href="#">Delete</a>
6	SV015	Que Ngoc Hai	hai.que@example.com	Information Technology	<a href="#">Edit</a> <a href="#">Delete</a>