


## List students

 **Student Management System**  
MVC Pattern with Jakarta EE & JSTL

[+ Add New Student](#)

ID	STUDENT CODE	FULL NAME	EMAIL	MAJOR	ACTIONS
27	SV021	Truong Huy Hoang	Hoang@email.com	CS	<a href="#">Edit</a> <a href="#">Delete</a>
26	SV020	ABC	john@email.com	CS	<a href="#">Edit</a> <a href="#">Delete</a>
24	SV013	QWERT	john@email.com	CS	<a href="#">Edit</a> <a href="#">Delete</a>
23	SV012	QWER	john@email.com	CS	<a href="#">Edit</a> <a href="#">Delete</a>
22	SV011	QWE	john@email.com	CS	<a href="#">Edit</a> <a href="#">Delete</a>
21	SV010	QW	john@email.com	CS	<a href="#">Edit</a> <a href="#">Delete</a>

## Add student

### + Add New Student

**Student Code \***

  
Format: 2 letters + 3+ digits

**Full Name \***

**Email \***

**Major \***

-- Select Major --

[+ Add Student](#)

[X Cancel](#)

## Edit student

### Edit Student

Student Code \*


Format: 2 letters + 3+ digits


Full Name \*

Email \*


Major \*

-- Select Major --

 Update Student


 Cancel













## Messages display



## Student Management

MVC Pattern with Jakarta EE & JSTL

 Add New Student

ID	STUDENT CODE	FULL NAME	EMAIL	MAJOR	ACTIONS
27	SV021	Truong Huy Hoang	Hoang@email.com	CS	<div> Edit  Delete</div>
26	SV020	ABC	john@email.com	CS	<div> Edit  Delete</div>
24	SV013	QWERT	john@email.com	CS	<div> Edit  Delete</div>
23	SV012	QWER	john@email.com	CS	<div> Edit  Delete</div>
22	SV011	QWE	john@email.com	CS	<div> Edit  Delete</div>
21	SV010	QW	john@email.com	CS	<div> Edit  Delete</div>

localhost:8080 says

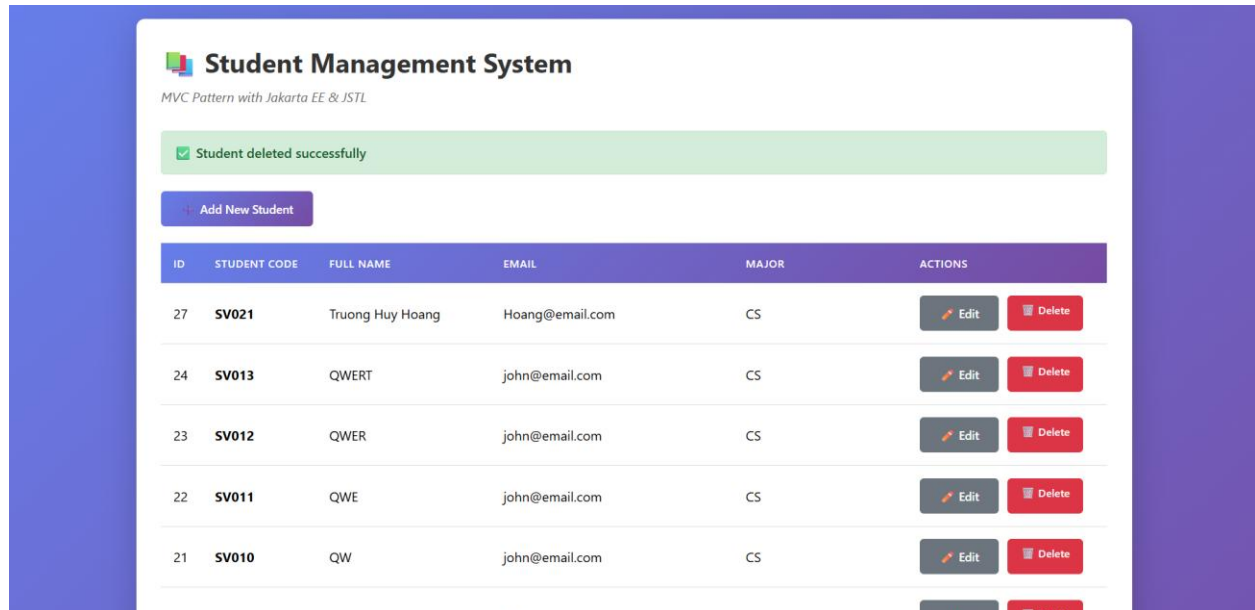
Are you sure you want to delete this student?

OK

Cancel

localhost:8080/student-management-mvc/student?action=delete&amp;id=26

## Delete student



## Code flow

**Model:** Student.java (the data object) and StudentDAO.java (the data access logic).

**View:** student-list.jsp and student-form.jsp (the user interface).

**Controller:** StudentController.java (the servlet that manages all traffic).

## Scenario: Viewing the Student List (The Default Action)

1. Browser → GET /student?action=list (<http://localhost:8080/student-management-mvc/student?action=list>)
2. Controller (StudentController.java)
  - Server Tomcat routes this request to StudentController servlet ( StudentController.java ) @WebServlet("/student")
  - The doGet() method is executed.
  - It reads the action parameter. Since it's the first visit, action is null.
  - The if (action == null) check sets action = "list".
  - The switch statement calls the listStudents() method.
3. Controller calls DAO (listStudents() method)
  - The controller calls studentDAO.getAllStudents().
4. Model/DAO (StudentDAO.java) executes SQL
  - The getAllStudents() method connects to MySQL database.

- It executes the SQL query: `SELECT * FROM students ORDER BY id DESC`.
  - It loops through the `ResultSet`, creates a `List<Student>` (a list of Model objects), and returns it.
5. DAO returns to Controller
    - The `listStudents()` method receives the `List<Student>`.
    - It stores this list in the request object: `request.setAttribute("students", students);`.
  6. Controller forwards to View sets data as request attribute
    - The controller gets a `RequestDispatcher` for `/views/student-list.jsp`.
    - It uses `dispatcher.forward(request, response)`. This is a server-side forward
  7. View (`student-list.jsp`) displays using JSTL
    - The JSTL tag `<c:forEach var="student" items="${students}">` finds the students attribute in the request.
  8. Browser receives HTML response