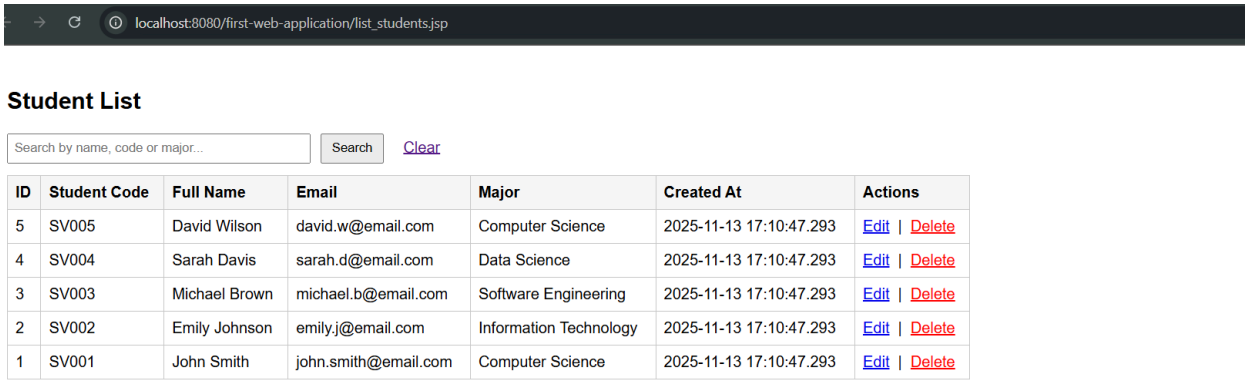
Student Name: Vũ Đình Đức
Student ID: ITCSIU23004
Course: Web-application Development
Semester: I 2025-2026
Instructor: Nguyễn Văn Sinh

Lab 04

PART B: HOMEWORK EXERCISES
Exercise 5:



ID	Student Code	Full Name	Email	Major	Created At	Actions
5	SV005	David Wilson	david.w@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete
4	SV004	Sarah Davis	sarah.d@email.com	Data Science	2025-11-13 17:10:47.293	Edit | Delete
3	SV003	Michael Brown	michael.b@email.com	Software Engineering	2025-11-13 17:10:47.293	Edit | Delete
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	2025-11-13 17:10:47.293	Edit | Delete
1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete

We have a total of 1 input for searching: keyword.

We set the search form to use GET so the search term appears in the URL:

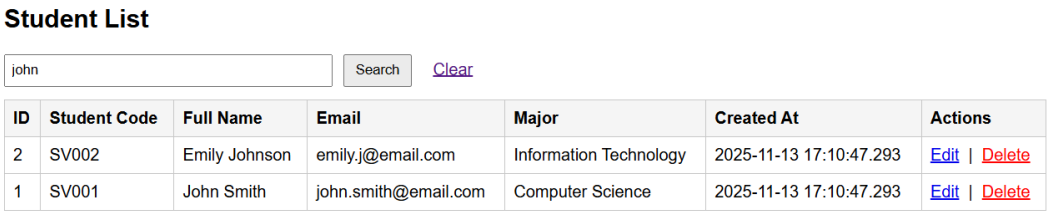
For the search input:
+The form is <form action="list_students.jsp" method="GET"> with <input type="text" name="keyword" ...> and a Search button.
+When the user types a keyword and clicks Search -> the browser requests list_students.jsp?keyword=....
+If the user clicks **Clear** -> the link goes to list_students.jsp (no keyword) and shows all students.

When the page receives the request:
+At the top we read keyword with request.getParameter("keyword") and trim it.
+If keyword is not empty -> we set hasKeyword = true and prepare a search SQL:
->SELECT ... FROM dbo.students WHERE full_name LIKE ? OR student_code LIKE ? OR major LIKE ? ORDER BY id DESC

->We use PreparedStatement and bind three parameters with "%"+keyword+"%".
->This ensures the LIKE operator is used correctly and searches name, code, and major.
+If keyword is empty -> we run the normal query:
->SELECT ... FROM dbo.students ORDER BY id DESC

When the prepared statement runs:
+We use try-with-resources for Connection, PreparedStatement, and ResultSet.
+We loop with while (rs.next()) and print each row (ID, student_code, full_name, email, major, created_at).
+For each row we show **Edit** and **Delete** links:
->Delete link: delete_student.jsp?id=[id] with onclick="return confirm(...)" so the confirmation dialog works.
->Delete link has CSS color:red (class delete-link) to be visually red.

Search behavior / test cases:
+If keyword = "John" -> rows with "John" in full_name appear.

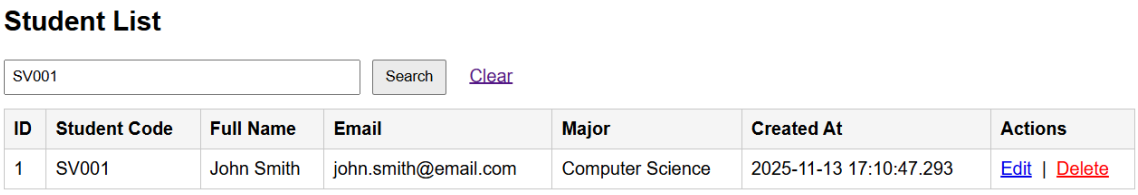


Student List

john Search Clear

ID	Student Code	Full Name	Email	Major	Created At	Actions
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	2025-11-13 17:10:47.293	Edit | Delete
1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete

+If keyword = "SV001" -> row with student_code SV001 appears.



Student List

SV001 Search Clear

ID	Student Code	Full Name	Email	Major	Created At	Actions
1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete

+If keyword = "science" -> rows with major containing "science" appear (Computer Science, Data Science).

Student List

science Search Clear

ID	Student Code	Full Name	Email	Major	Created At	Actions
5	SV005	David Wilson	david.w@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete
4	SV004	Sarah Davis	sarah.d@email.com	Data Science	2025-11-13 17:10:47.293	Edit | Delete
1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete

+If keyword is empty -> page shows all students (Clear link removes the query).

Student List

Search by name, code or major... Search Clear

ID	Student Code	Full Name	Email	Major	Created At	Actions
5	SV005	David Wilson	david.w@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete
4	SV004	Sarah Davis	sarah.d@email.com	Data Science	2025-11-13 17:10:47.293	Edit | Delete
3	SV003	Michael Brown	michael.b@email.com	Software Engineering	2025-11-13 17:10:47.293	Edit | Delete
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	2025-11-13 17:10:47.293	Edit | Delete
1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete

Exercise 6:

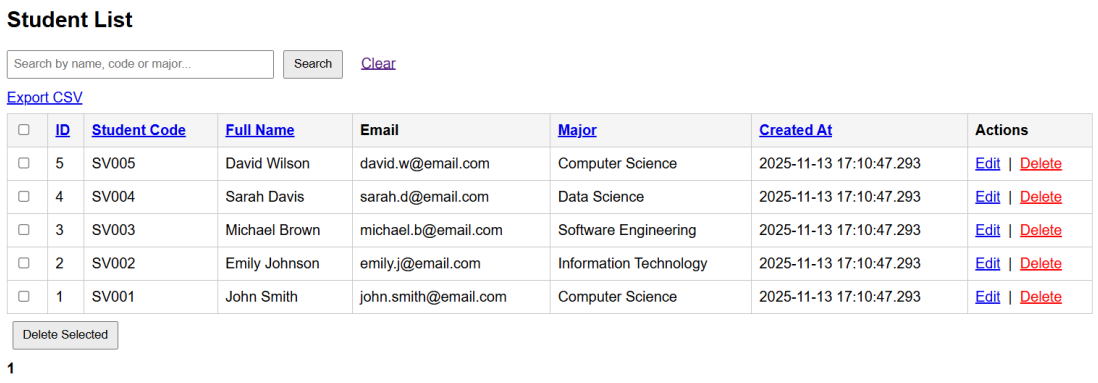
We added two validation patterns:
+codePattern = "^[A-Z]{2}[0-9]{3,}$" — ensures Student Code is 2 uppercase letters + at least 3 digits (SV001, IT123).
+emailPattern = "^[A-Za-z0-9+_.-]+@(.+)$" — ensures basic email format.

In process_add.jsp:
+When the form posts, we trim inputs and check required fields.
+We check student_code.matches(codePattern):

->If not match -> show clear error "Invalid Student Code format" and link back to add form.
+We check email if provided using the emailPattern:
->If invalid -> show "Invalid email format" and link back to add form.
+If all validations pass -> proceed to insert using PreparedStatement.

In process_edit.jsp:
+We validate id and full_name as before.
+We also validate student_code pattern (even though it is readonly) to be safe.
+We validate email if present.
+If validations pass -> perform UPDATE with PreparedStatement and redirect on success.

Exercise 7:



Student List

Search by name, code or major... Search Clear

Export CSV

	ID	Student Code	Full Name	Email	Major	Created At	Actions
☐	5	SV005	David Wilson	david.w@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete
☐	4	SV004	Sarah Davis	sarah.d@email.com	Data Science	2025-11-13 17:10:47.293	Edit | Delete
☐	3	SV003	Michael Brown	michael.b@email.com	Software Engineering	2025-11-13 17:10:47.293	Edit | Delete
☐	2	SV002	Emily Johnson	emily.j@email.com	Information Technology	2025-11-13 17:10:47.293	Edit | Delete
☐	1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete

Delete Selected

1

Student List

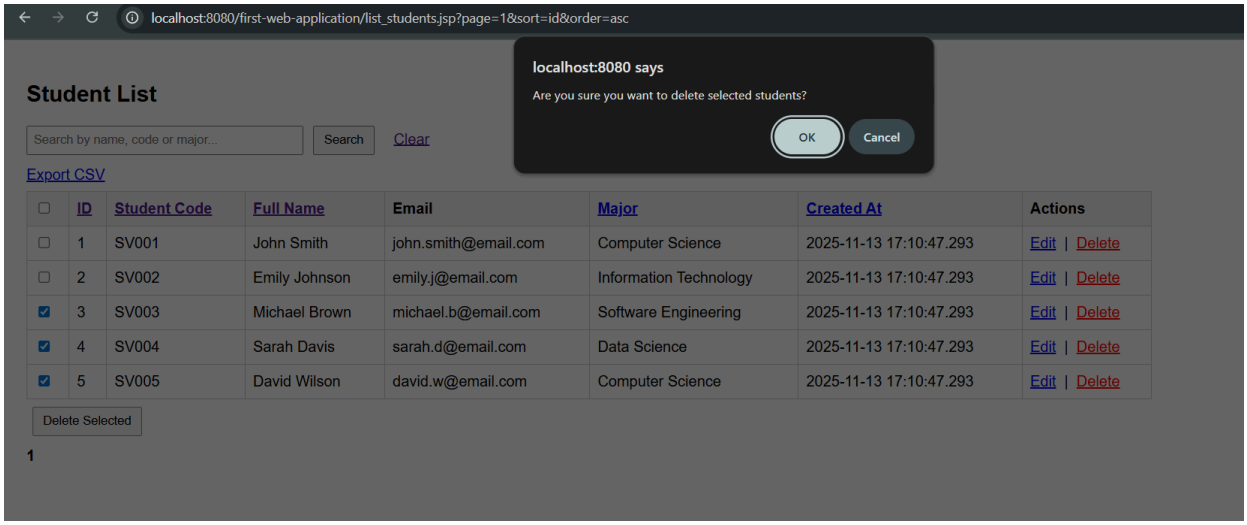
Search by name, code or major... Search Clear

Export CSV

	ID	Student Code	Full Name	Email	Major	Created At	Actions
☐	1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete
☐	2	SV002	Emily Johnson	emily.j@email.com	Information Technology	2025-11-13 17:10:47.293	Edit | Delete
☑	3	SV003	Michael Brown	michael.b@email.com	Software Engineering	2025-11-13 17:10:47.293	Edit | Delete
☑	4	SV004	Sarah Davis	sarah.d@email.com	Data Science	2025-11-13 17:10:47.293	Edit | Delete
☑	5	SV005	David Wilson	david.w@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete

Delete Selected

1



Student List

✓ 3 student(s) deleted

Search by name, code or major... Search Clear

Export CSV

	ID	Student Code	Full Name	Email	Major	Created At	Actions
☐	2	SV002	Emily Johnson	emily.j@email.com	Information Technology	2025-11-13 17:10:47.293	Edit | Delete
☐	1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-13 17:10:47.293	Edit | Delete

Delete Selected

1

We have a search input and several parameters in the URL: keyword, page, sort, order.

We read these params at the top:
+page -> converted to currentPage (default 1).

+recordsPerPage -> set to 10.
+We compute offset = (currentPage - 1) * recordsPerPage.
+keyword -> trimmed and checked. If present -> we run count & search queries using LIKE.
+sort & order -> sanitized (allowed columns only) to avoid SQL injection; default to id and desc.

To get total pages:
+We run a COUNT(*) query (same WHERE as search if keyword provided).
->totalPages = ceil(totalRecords / recordsPerPage).
->If 0 records -> totalPages = 1.

To get the current page records:
+We run a SELECT ... ORDER BY [sort] [order] OFFSET ? ROWS FETCH NEXT ? ROWS ONLY.
->If search -> WHERE full_name LIKE ? OR student_code LIKE ? OR major LIKE ?.
->We use PreparedStatement and set parameters in order (prevent injection).
->This returns exactly up to 10 rows for the page.

Pagination links:
+We show Previous link (if currentPage > 1).
+We loop i = 1..totalPages and show strong for current page and links for others.
+We show Next link (if currentPage < totalPages).
+All links preserve current keyword, sort, and order so the user stays in the same view.

Message styling & auto-hide:
+We display success messages in green with a ✓ icon and errors in red with a ✗ icon.
+We auto-hide messages after 3 seconds using setTimeout in JS.

Loading / submit prevention:
+Forms use onsubmit="return submitForm(this)" or submitForm is used for search; it disables submit button and sets text to Processing... to prevent double submits.

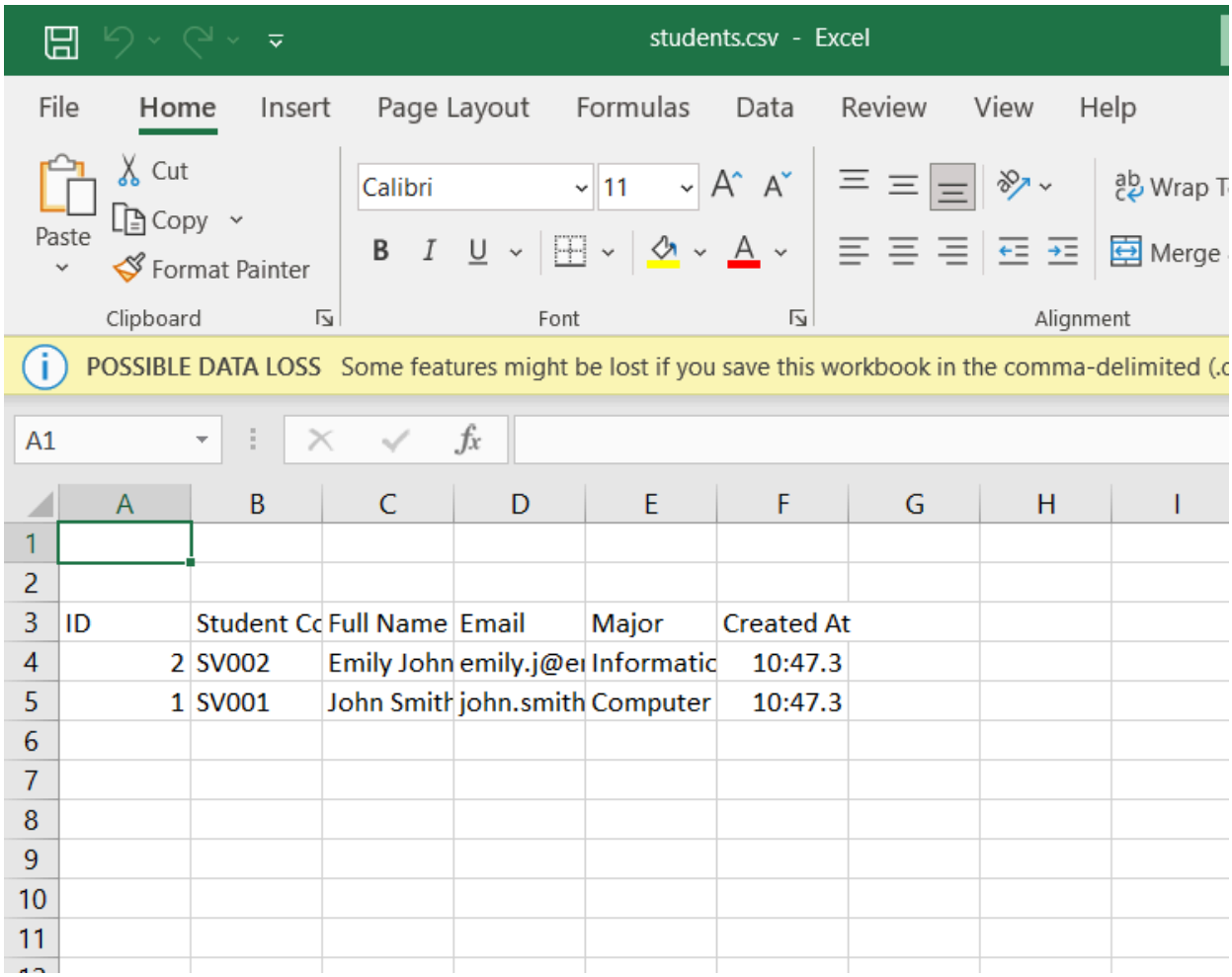
Responsive table:
+The table is wrapped in .table-responsive { overflow-x:auto; }.
+CSS media query reduces font/padding on smaller screens.

Delete links and bulk delete:
+Each row has individual Delete link with onclick="return confirm(...)" and is red.
+There is a checkbox for each row and a Select All checkbox in header that toggles all.
+Bulk Delete posts to delete_selected.jsp which validates ids, builds a safe DELETE ... WHERE id IN (?, ?, ...) using PreparedStatement, executes it, and redirects back with a message.

Export CSV:
+Export CSV link calls export_csv.jsp with same keyword, sort, and order when applicable.

+export_csv.jsp streams CSV with proper Content-Disposition so the browser downloads the file.

Security & resource handling:
+All DB operations use PreparedStatement (no concatenation of user input into SQL except for the sanitized sort column).
+Database resources are closed via try-with-resources.
+Sort column is whitelisted to prevent injection via sort parameter.



LINK GITHUB REPO: https://github.com/ducvu01/web_lab_04_exercise567.git