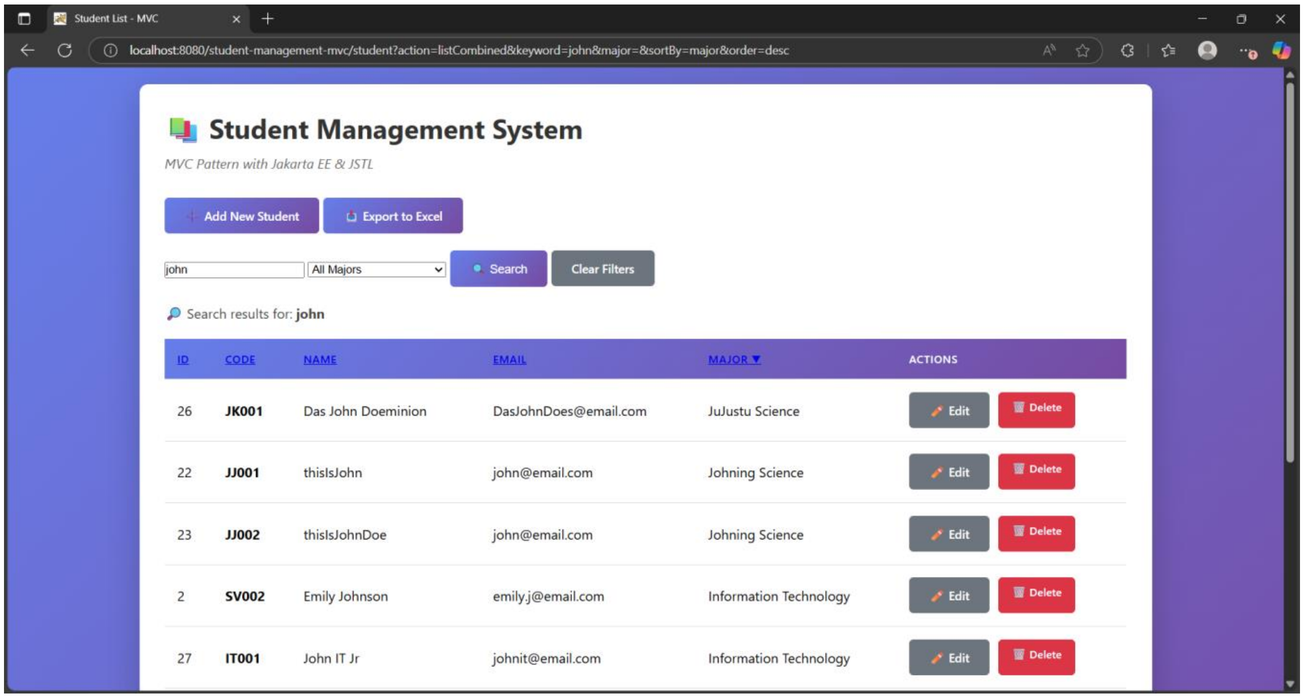
Web Application Development Lab 05
SERVLET & MVC PATTERN
Part B - HOMEWORK

I. SEARCH FUNCTIONALITY

1. Search results

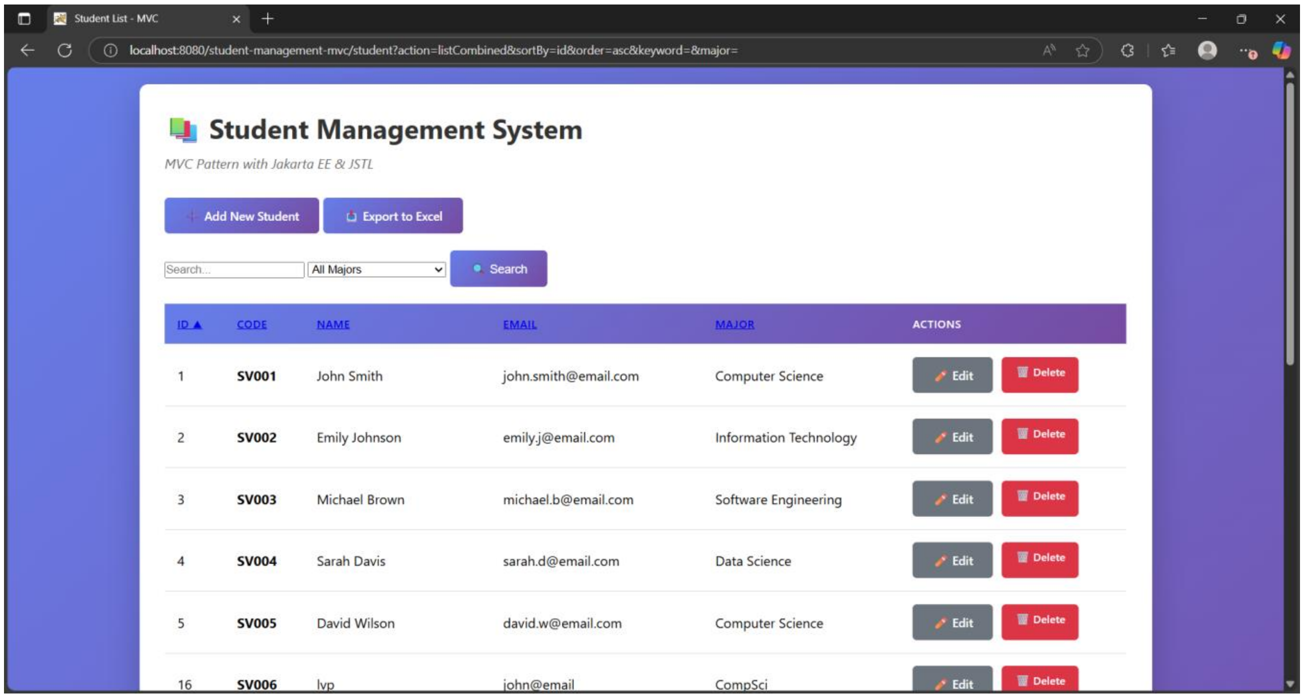
***Screenshots taken after the completion of most of the exercises**


Student List - MVC
localhost:8080/student-management-mvc/student?action=listCombined&keyword=john&major=&sortBy=major&order=desc
Student Management System
MVC Pattern with Jakarta EE & JSTL
Add New Student
Export to Excel
john
All Majors
Search
Clear Filters

Search results for: **john**

ID	CODE	NAME	EMAIL	MAJOR ▼	ACTIONS
26	JK001	Das John Doeminion	DasJohnDoes@email.com	JuJustu Science	Edit Delete
22	JJ001	thisIsJohn	john@email.com	Johning Science	Edit Delete
23	JJ002	thisIsJohnDoe	john@email.com	Johning Science	Edit Delete
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	Edit Delete
27	IT001	John IT Jr	johnit@email.com	Information Technology	Edit Delete

Searching for 'john'

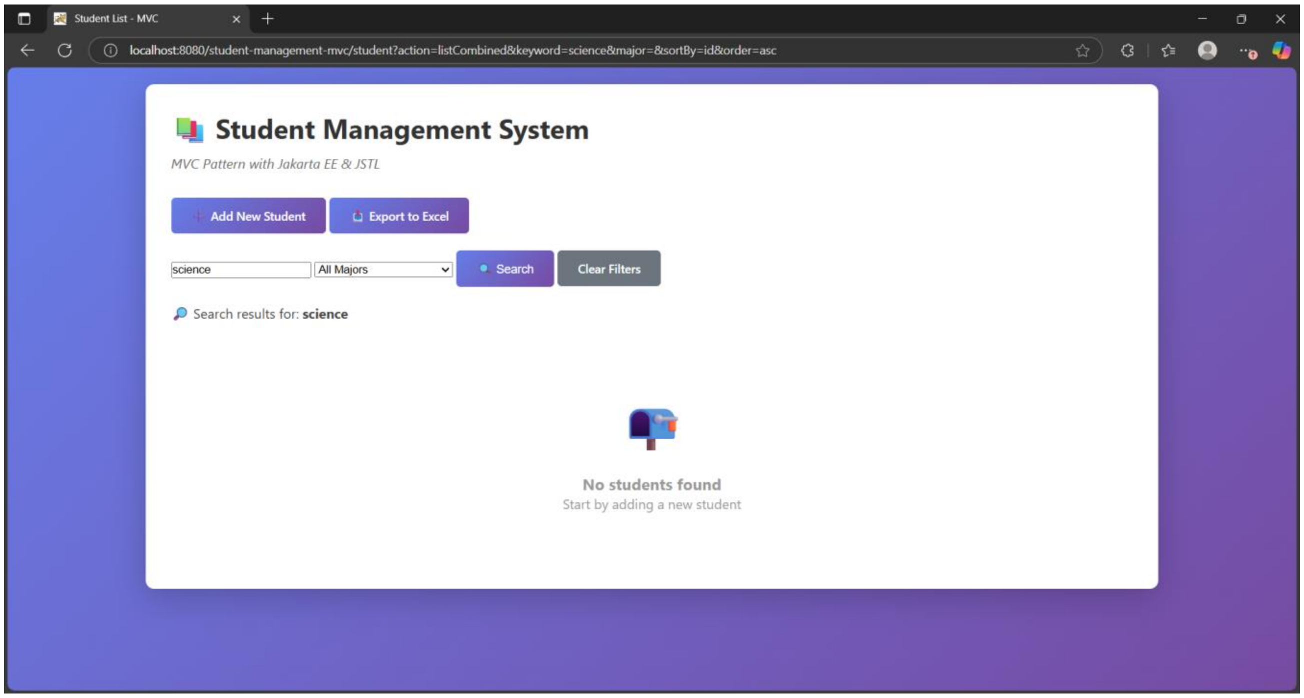

localhost:8080/student-management-mvc/student?action=listCombined&sortBy=id&order=asc&keyword=&major=

Student Management System

MVC Pattern with Jakarta EE & JSTL

ID ▲	CODE	NAME	EMAIL	MAJOR	ACTIONS
1	SV001	John Smith	john.smith@email.com	Computer Science	Edit Delete
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	Edit Delete
3	SV003	Michael Brown	michael.b@email.com	Software Engineering	Edit Delete
4	SV004	Sarah Davis	sarah.d@email.com	Data Science	Edit Delete
5	SV005	David Wilson	david.w@email.com	Computer Science	Edit Delete
16	SV006	lvp	john@email	CompSci	Edit Delete

Handles empty searches


localhost:8080/student-management-mvc/student?action=listCombined&keyword=science&major=&sortBy=id&order=asc

Student Management System

MVC Pattern with Jakarta EE & JSTL

Search results for: **science**

No students found

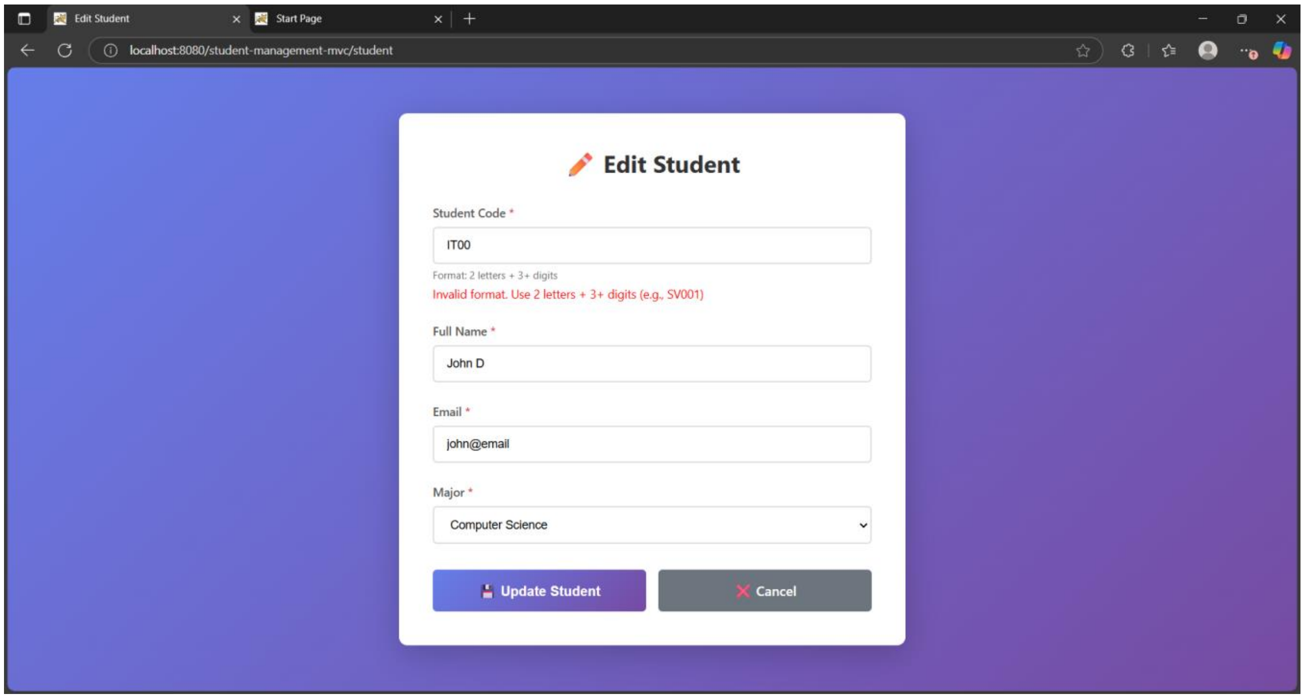
Start by adding a new student

Handles 'no students found'

*Issue: - 'Major' is used as a filter, as such, it cannot be used for searches.

II. SERVER-SIDE VALIDATION

1. Display Validation Errors in Form



Edit Student

Student Code *

IT00

Format: 2 letters + 3+ digits

Invalid format. Use 2 letters + 3+ digits (e.g., SV001)

Full Name *

John D

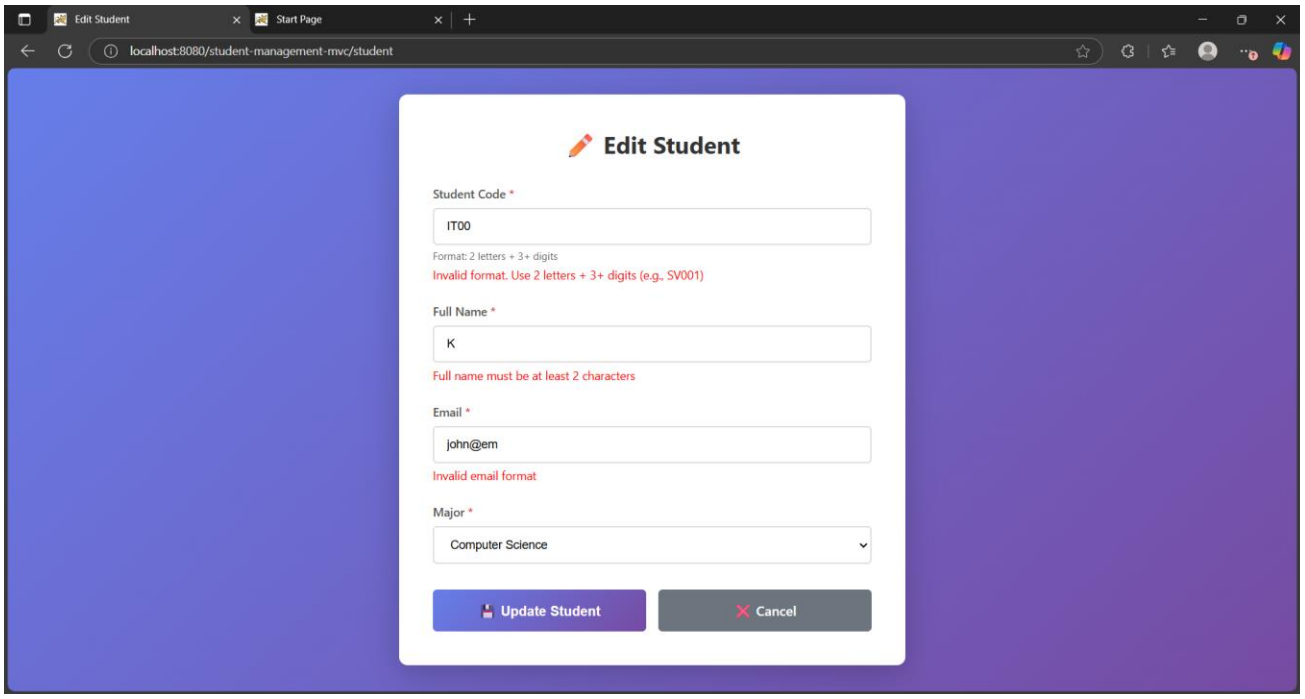
Email *

john@email

Major *

Computer Science

2. Errors appear in appropriate fields



Edit Student

Student Code *

IT00

Format: 2 letters + 3+ digits

Invalid format. Use 2 letters + 3+ digits (e.g., SV001)

Full Name *

K

Full name must be at least 2 characters

Email *

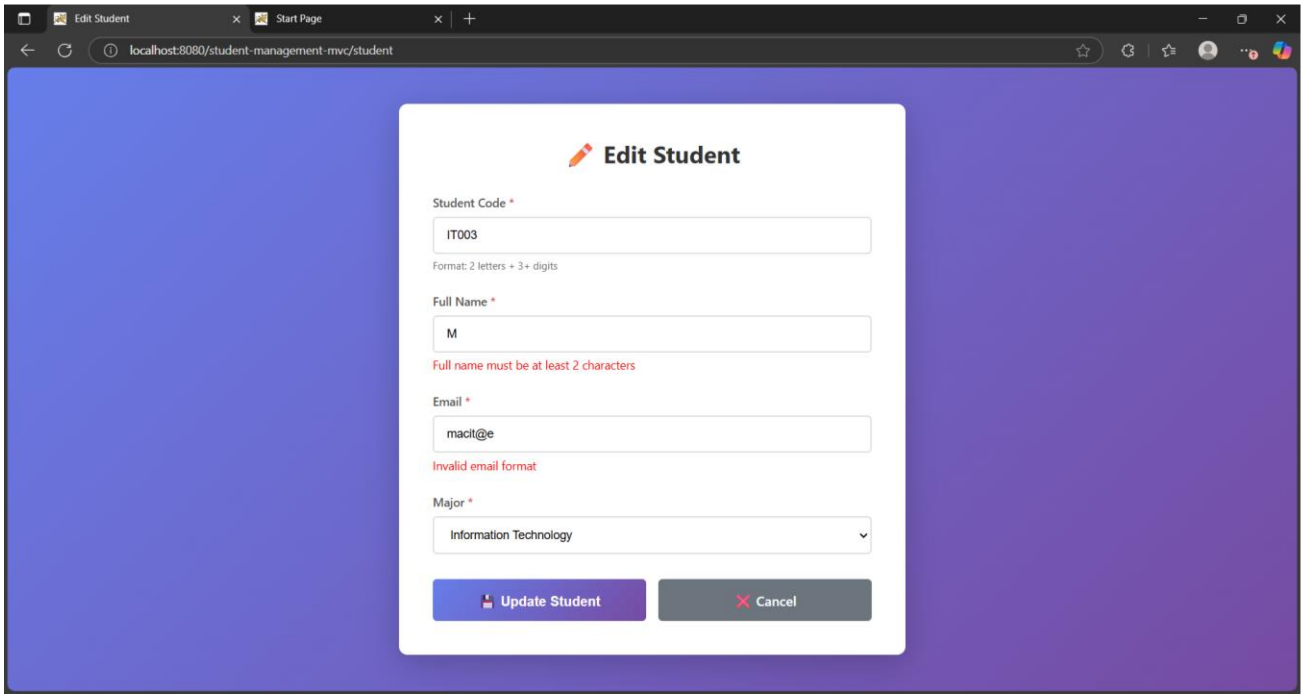
john@em

Invalid email format

Major *

Computer Science

3. For editing



Edit Student

Student Code *

IT003

Format: 2 letters + 3+ digits

Full Name *

M

Full name must be at least 2 characters

Email *

macit@e

Invalid email format

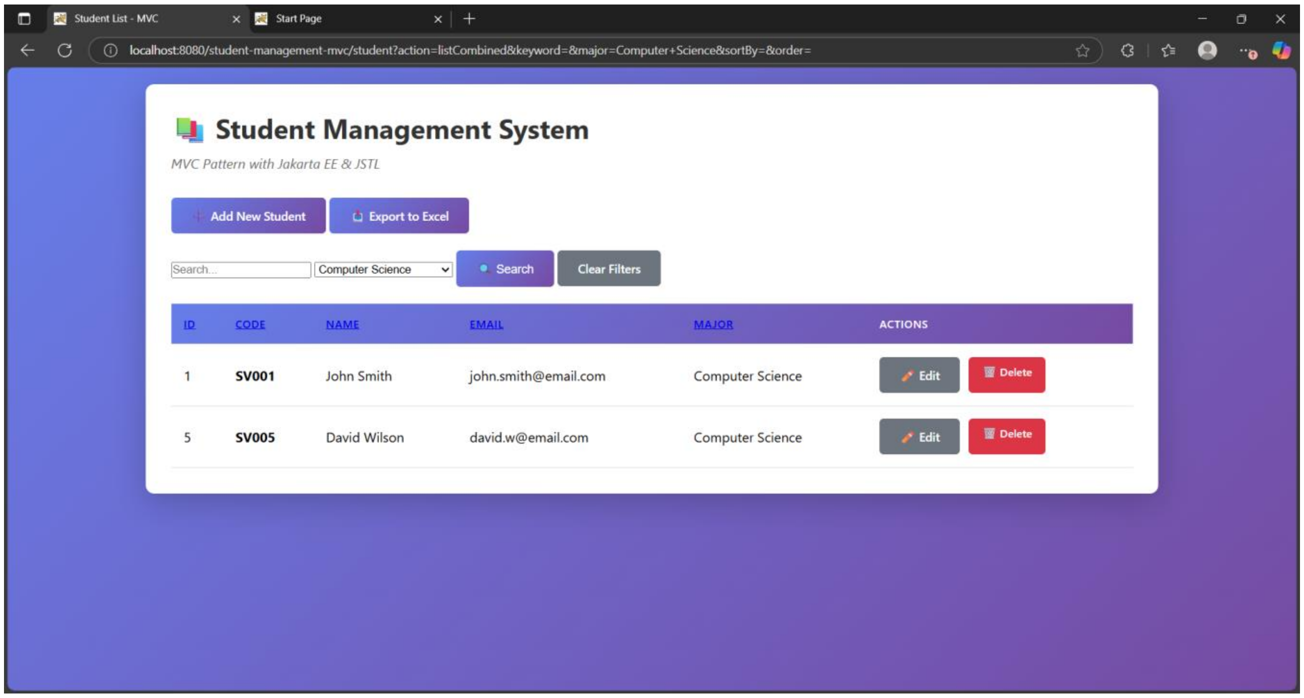
Major *

Information Technology

Student code set as ‘read-only’

III. SORTING & FILTERING

1. Filtering



localhost:8080/student-management-mvc/student?action=listCombined&keyword=&major=Computer+Science&sortBy=&order=

Student Management System

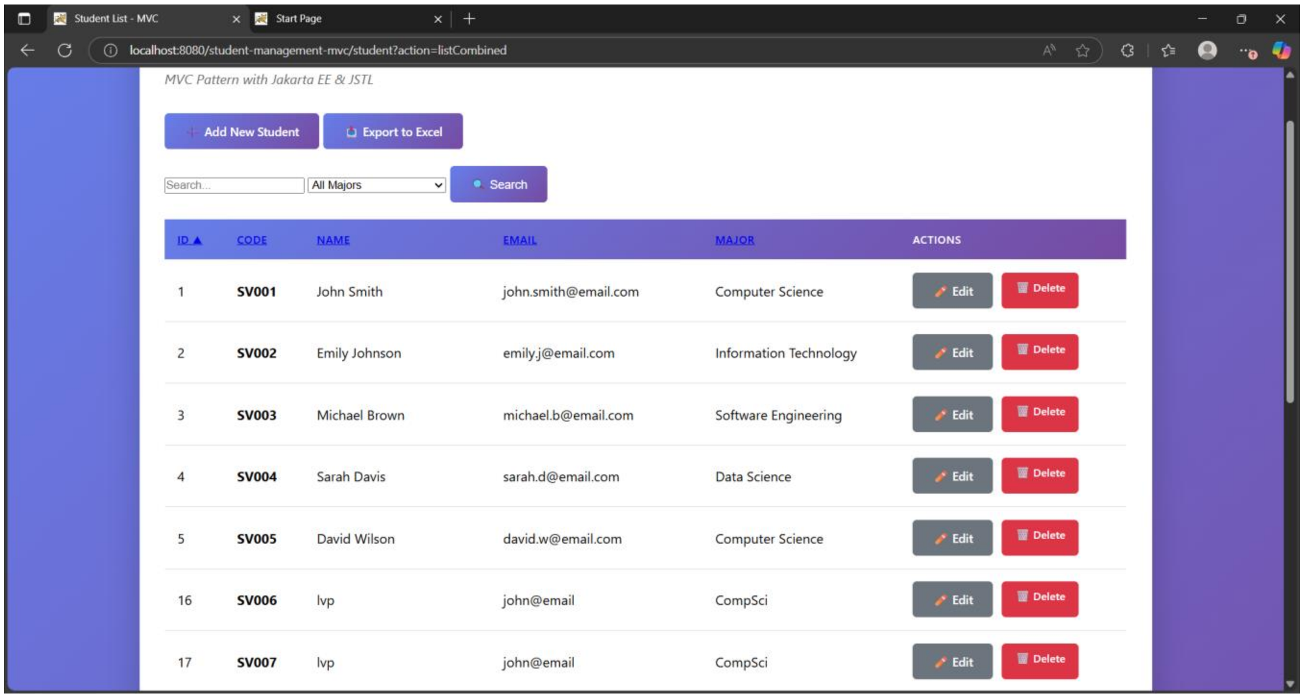
MVC Pattern with Jakarta EE & JSTL

Add New Student | Export to Excel

Search... | Computer Science | Search | Clear Filters

ID	CODE	NAME	EMAIL	MAJOR	ACTIONS
1	SV001	John Smith	john.smith@email.com	Computer Science	Edit Delete
5	SV005	David Wilson	david.w@email.com	Computer Science	Edit Delete

2. Sorting



localhost:8080/student-management-mvc/student?action=listCombined

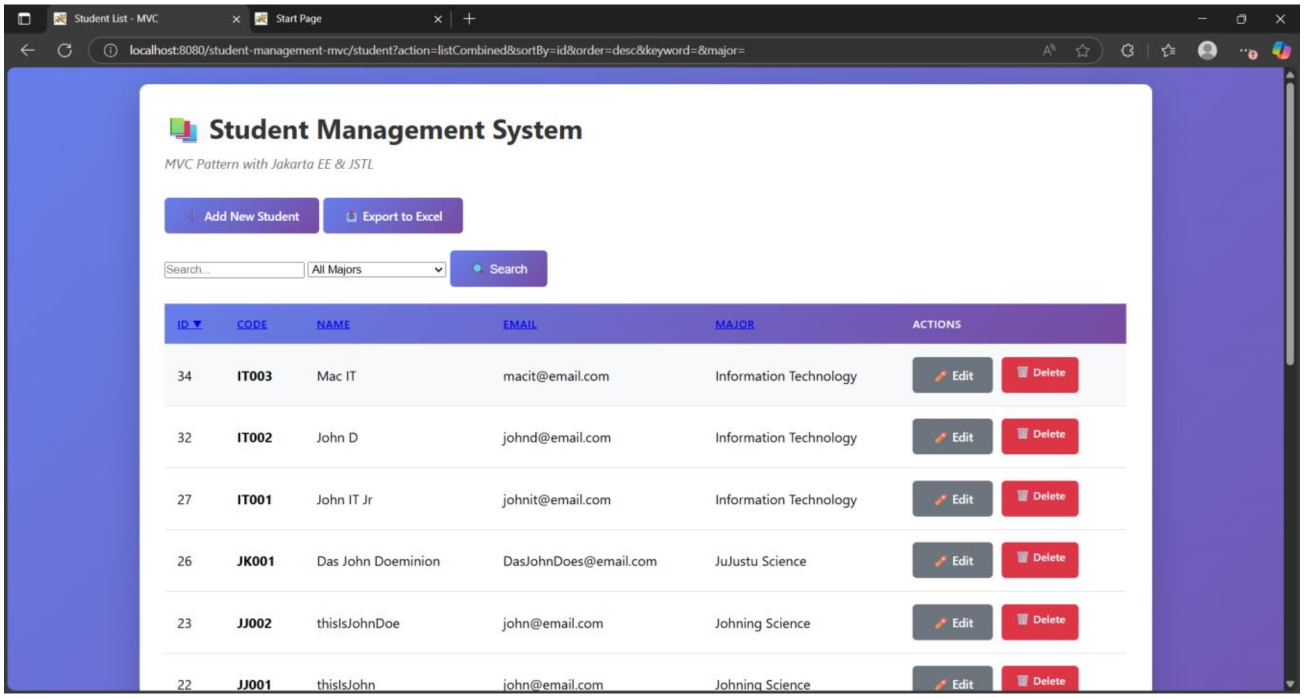
MVC Pattern with Jakarta EE & JSTL

Add New Student | Export to Excel

Search... | All Majors | Search

ID ▲	CODE	NAME	EMAIL	MAJOR	ACTIONS
1	SV001	John Smith	john.smith@email.com	Computer Science	Edit Delete
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	Edit Delete
3	SV003	Michael Brown	michael.b@email.com	Software Engineering	Edit Delete
4	SV004	Sarah Davis	sarah.d@email.com	Data Science	Edit Delete
5	SV005	David Wilson	david.w@email.com	Computer Science	Edit Delete
16	SV006	lvp	john@email	CompSci	Edit Delete
17	SV007	lvp	john@email	CompSci	Edit Delete

Sort by Id in Ascending order


localhost:8080/student-management-mvc/student?action=listCombined&sortBy=id&order=desc&keyword=&major=

Student Management System

MVC Pattern with Jakarta EE & JSTL

ID ▼	CODE	NAME	EMAIL	MAJOR	ACTIONS
34	IT003	Mac IT	macit@email.com	Information Technology	Edit Delete
32	IT002	John D	johnd@email.com	Information Technology	Edit Delete
27	IT001	John IT Jr	johnit@email.com	Information Technology	Edit Delete
26	JK001	Das John Doeminion	DasJohnDoes@email.com	JuJustu Science	Edit Delete
23	JJ002	thisIsJohnDoe	john@email.com	Johning Science	Edit Delete
22	JJ001	thisIsJohn	john@email.com	Johning Science	Edit Delete

Sort by Id in Descending order

*The current filter&sort is the same implementation as that of the bonus exercise: Search&Filter&Sort

Code submission:

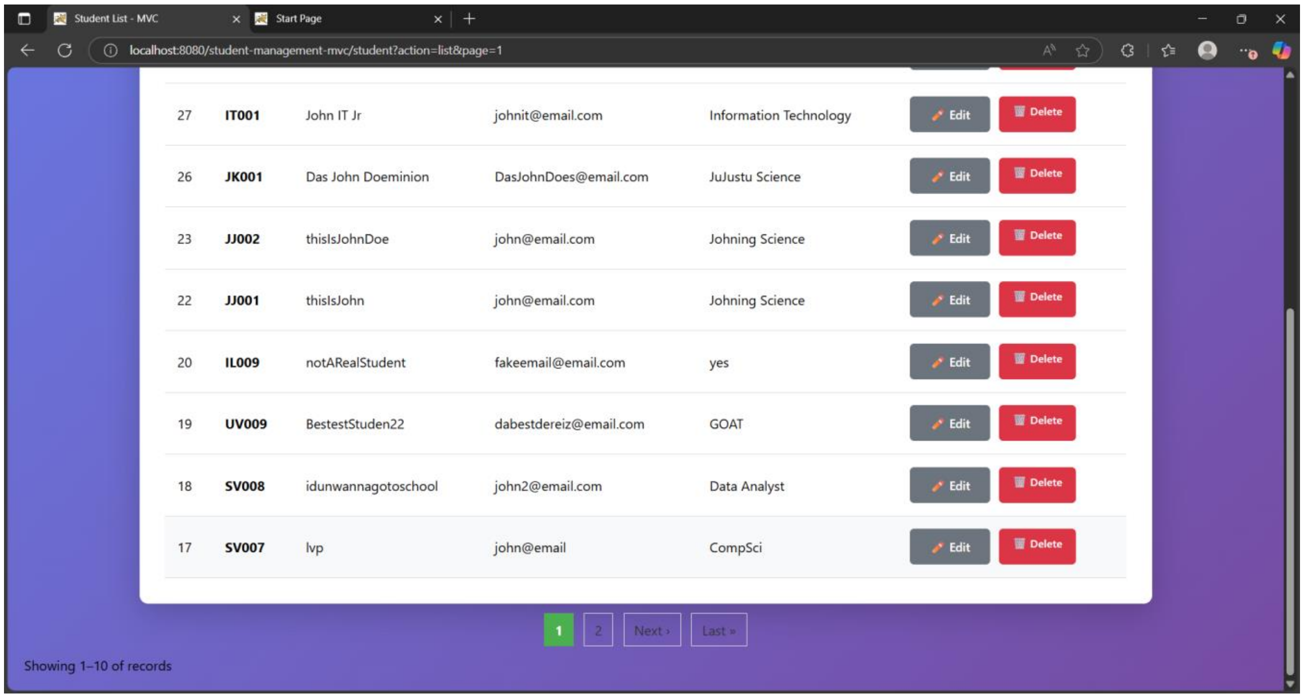
StudentDAO.java

```
public List<Student> getStudentsCombined(String keyword, String major, String sortBy, String order) {
    List<Student> students = new ArrayList<>();
    StringBuilder sql = new StringBuilder("SELECT * FROM students WHERE 1=1");

    if (keyword != null && !keyword.trim().isEmpty()) {
        sql.append(" AND (student_code LIKE ? OR full_name LIKE ? OR email LIKE ?)");
    }
    if (major != null && !major.trim().isEmpty()) {
        sql.append(" AND major = ?");
    }

    sql.append(" ORDER BY ").append(validateSortBy(sortBy)).append("
").append(validateOrder(order));

    try (Connection conn = getConnection();
         PreparedStatement pstmt = conn.prepareStatement(sql.toString())) {

        int index = 1;
        if (keyword != null && !keyword.trim().isEmpty()) {
            String pattern = "%" + keyword.trim() + "%";
            pstmt.setString(index++, pattern);
            pstmt.setString(index++, pattern);
            pstmt.setString(index++, pattern);
        }

        if (major != null && !major.trim().isEmpty()) {
            pstmt.setString(index++, major);
        }

        ResultSet rs = pstmt.executeQuery();
        while (rs.next()) {
            Student student = new Student();
            student.setId(rs.getInt("id"));
            student.setStudentCode(rs.getString("student_code"));
            student.setFullName(rs.getString("full_name"));
            student.setEmail(rs.getString("email"));
            student.setMajor(rs.getString("major"));
            student.setCreatedAt(rs.getTimestamp("created_at"));
            students.add(student);
        }

    } catch (SQLException e) {
        e.printStackTrace();
    }

    return students;

}
```

StudentController.java

```
private void listCombined(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {

    String keyword = request.getParameter("keyword");
    String major = request.getParameter("major");
    String sortBy = request.getParameter("sortBy");
    String order = request.getParameter("order");


    List<Student> students = studentDAO.getStudentsCombined(keyword, 
major, sortBy, order);


    request.setAttribute("students", students);
    request.setAttribute("keyword", keyword != null ? keyword : "");
    request.setAttribute("selectedMajor", major != null ? major : "");
    request.setAttribute("sortBy", sortBy != null ? sortBy : "id");
    request.setAttribute("order", order != null ? order : "asc");


    RequestDispatcher dispatcher = 
request.getRequestDispatcher("/views/student-list.jsp");
    dispatcher.forward(request, response);
}
```

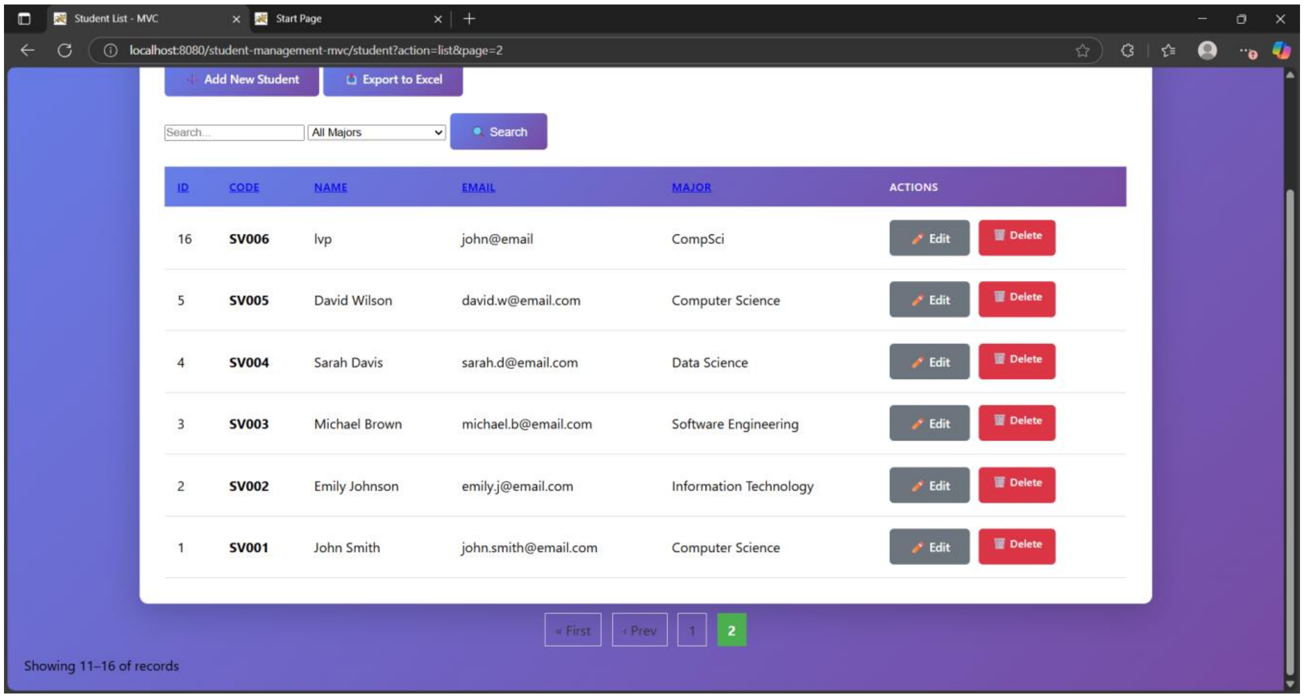
IV. PAGINATION


Student List - MVC
Start Page
localhost:8080/student-management-mvc/student?action=list&page=1

27	**IT001**	John IT Jr	johnit@email.com	Information Technology	Edit Delete
26	**JK001**	Das John Doeminion	DasJohnDoes@email.com	JuJustu Science	Edit Delete
23	**JJ002**	thisIsJohnDoe	john@email.com	Johning Science	Edit Delete
22	**JJ001**	thisIsJohn	john@email.com	Johning Science	Edit Delete
20	**IL009**	notARealStudent	fakeemail@email.com	yes	Edit Delete
19	**UV009**	BestestStuden22	dabestdereiz@email.com	GOAT	Edit Delete
18	**SV008**	idunwannagotoschool	john2@email.com	Data Analyst	Edit Delete
17	**SV007**	lvp	john@email	CompSci	Edit Delete

Showing 1–10 of records

First page



ID	CODE	NAME	EMAIL	MAJOR	ACTIONS
16	**SV006**	lvp	john@email	CompSci	Edit Delete
5	**SV005**	David Wilson	david.w@email.com	Computer Science	Edit Delete
4	**SV004**	Sarah Davis	sarah.d@email.com	Data Science	Edit Delete
3	**SV003**	Michael Brown	michael.b@email.com	Software Engineering	Edit Delete
2	**SV002**	Emily Johnson	emily.j@email.com	Information Technology	Edit Delete
1	**SV001**	John Smith	john.smith@email.com	Computer Science	Edit Delete

Showing 11–16 of records

Second page

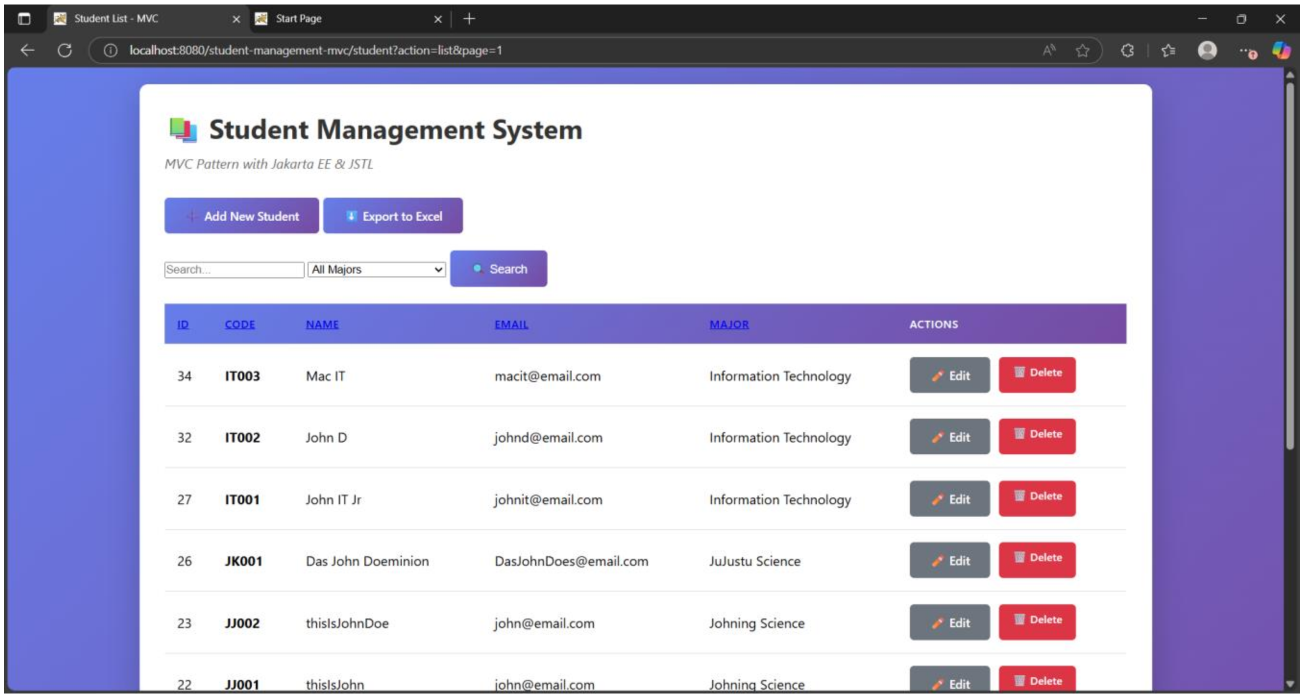
Code Implementation:

```
<c:if test="${totalPages > 1}">
        <div class="pagination">
            <c:if test="${currentPage > 1}">
                <a href="student?action=list&page=1">« First</a>
            </c:if>

            <c:if test="${currentPage > 1}">
                <a href="student?action=list&page=${currentPage - 1}">‹ Prev</a>
            </c:if>

            <c:set var="startPage" value="${currentPage - 2}" />
            <c:set var="endPage" value="${currentPage + 2}" />
            <c:if test="${startPage < 1}"><c:set var="startPage" value="1" /></c:if>
            <c:if test="${endPage > totalPages}"><c:set var="endPage"
value="${totalPages}" /></c:if>

            <c:forEach begin="${startPage}" end="${endPage}" var="i">
                <c:choose>
                    <c:when test="${i == currentPage}">
                        <strong>${i}</strong>
                    </c:when>
                    <c:otherwise>
                        <a href="student?action=list&page=${i}">${i}</a>
                    </c:otherwise>
                </c:choose>
            </c:forEach>


            <c:if test="${currentPage < totalPages}">
                <a href="student?action=list&page=${currentPage + 1}">Next ›</a>
            </c:if>


            <c:if test="${currentPage < totalPages}">
                <a href="student?action=list&page=${totalPages}">Last »</a>
            </c:if>
        </div>


        <c:set var="recordsPerPage" value="10" />
        <c:set var="startRecord" value="${(currentPage - 1) * recordsPerPage +
1}" />
        <c:set var="endRecord" value="${startRecord + students.size() - 1}" />
        <p>Showing ${startRecord}–${endRecord} of ${totalRecords}
records</p>
 </c:if>
```

*The pagination logic was taken from Lab04

V. BONUS EXERCISES

1. Export to Excel



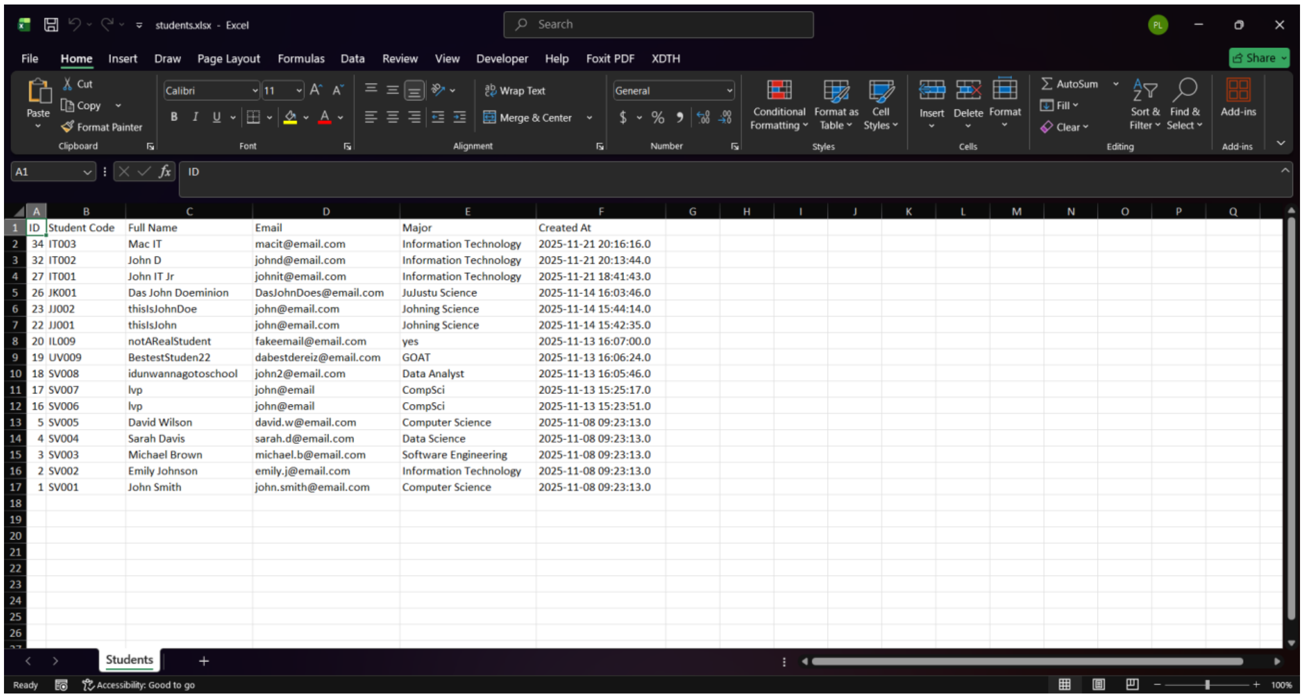
localhost:8080/student-management-mvc/student?action=list&page=1

Student Management System

MVC Pattern with Jakarta EE & JSTL

ID	CODE	NAME	EMAIL	MAJOR	ACTIONS
34	IT003	Mac IT	macit@email.com	Information Technology	Edit Delete
32	IT002	John D	johnd@email.com	Information Technology	Edit Delete
27	IT001	John IT Jr	johnit@email.com	Information Technology	Edit Delete
26	JK001	Das John Doeminion	DasJohnDoes@email.com	JuJustu Science	Edit Delete
23	JJ002	thisIsJohnDoe	john@email.com	Johning Science	Edit Delete
22	JJ001	thisIsJohn	john@email.com	Johning Science	Edit Delete

'Export to Excel' button



ID	Student Code	Full Name	Email	Major	Created At
34	IT003	Mac IT	macit@email.com	Information Technology	2025-11-21 20:16:16.0
32	IT002	John D	johnd@email.com	Information Technology	2025-11-21 20:13:44.0
27	IT001	John IT Jr	johnit@email.com	Information Technology	2025-11-21 18:41:43.0
26	JK001	Das John Doeminion	DasJohnDoes@email.com	JuJustu Science	2025-11-14 16:03:46.0
23	JJ002	thisIsJohnDoe	john@email.com	Johning Science	2025-11-14 15:44:14.0
22	JJ001	thisIsJohn	john@email.com	Johning Science	2025-11-14 15:42:35.0
20	IL009	notARealStudent	fakeemail@email.com	yes	2025-11-13 16:07:00.0
19	UV009	BestestStuden22	dabestdereiz@email.com	GOAT	2025-11-13 16:06:24.0
18	SV008	idunwannagotoschool	john2@email.com	Data Analyst	2025-11-13 16:05:46.0
17	SV007	lvp	john@email	CompSci	2025-11-13 15:25:17.0
16	SV006	lvp	john@email	CompSci	2025-11-13 15:23:51.0
5	SV005	David Wilson	david.w@email.com	Computer Science	2025-11-08 09:23:13.0
4	SV004	Sarah Davis	sarah.d@email.com	Data Science	2025-11-08 09:23:13.0
3	SV003	Michael Brown	michael.b@email.com	Software Engineering	2025-11-08 09:23:13.0
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	2025-11-08 09:23:13.0
1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-08 09:23:13.0

Excel view

Dependency:

```
   Pom.xml
<dependency>
        <groupId>org.apache.poi</groupId>
        <artifactId>poi-ooxml</artifactId>
        <version>5.2.3</version>
</dependency>
```

Implementation:

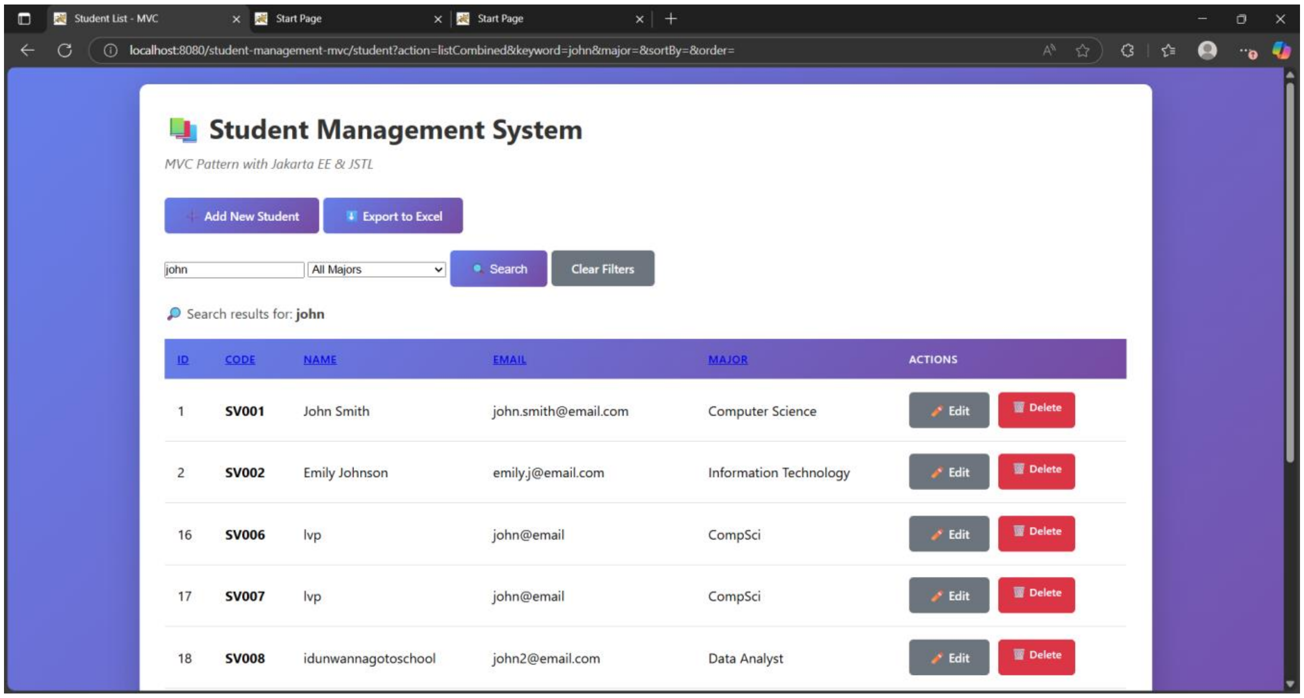
```
package com.student.controller;

import com.student.dao.StudentDAO;
import com.student.model.Student;

import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;

import org.apache.poi.ss.usermodel.*;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

import java.io.IOException;
import java.util.List;
```

```
@WebServlet("/export")
public class excelExport extends HttpServlet {

    private StudentDAO studentDAO;

    @Override
    public void init() {
        studentDAO = new StudentDAO();
    }

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

        // students
        List<Student> students = studentDAO.getAllStudents();

        try (
                Workbook workbook = new XSSFWorkbook()) {
            Sheet sheet = workbook.createSheet("Students");

            Row header = sheet.createRow(0);
            String[] columns = {"ID", "Student Code", "Full Name", "Email",
"Major", "Created At"};
            for (int i = 0; i < columns.length; i++) {
```

```
            Cell cell = header.createCell(i);
            cell.setCellValue(columns[i]);
        }

        int rowNum = 1;
        for (Student s : students) {
            Row row = sheet.createRow(rowNum++);
            row.createCell(0).setCellValue(s.getId());
            row.createCell(1).setCellValue(s.getStudentCode());
            row.createCell(2).setCellValue(s.getFullName());
            row.createCell(3).setCellValue(s.getEmail());
            row.createCell(4).setCellValue(s.getMajor());
            row.createCell(5).setCellValue(s.getCreatedAt().toString());
        }

        for (int i = 0; i < columns.length; i++) {
            sheet.autoSizeColumn(i);
        }

        response.setContentType("application/vnd.ms-excel");
        response.setHeader("Content-Disposition", "attachment;
filename=students.xlsx");

        workbook.write(response.getOutputStream());
    }
  }
```

```
}
```

2. Search&Filter&Sort



localhost:8080/student-management-mvc/student?action=listCombined&keyword=john&major=&sortBy=&order=

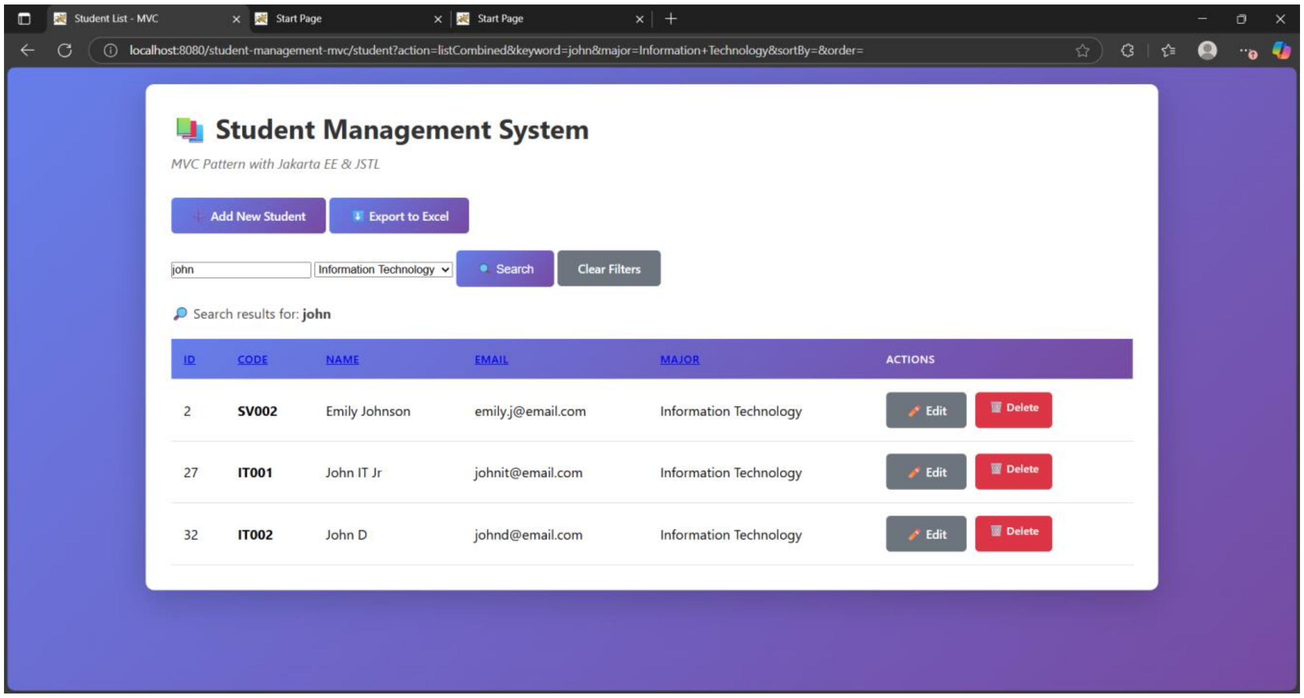
Student Management System

MVC Pattern with Jakarta EE & JSTL

Search results for: **john**

ID	CODE	NAME	EMAIL	MAJOR	ACTIONS
1	SV001	John Smith	john.smith@email.com	Computer Science	Edit Delete
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	Edit Delete
16	SV006	lvp	john@email	CompSci	Edit Delete
17	SV007	lvp	john@email	CompSci	Edit Delete
18	SV008	idunwannagotoschool	john2@email.com	Data Analyst	Edit Delete

Search results for 'John'



localhost:8080/student-management-mvc/student?action=listCombined&keyword=john&major=Information+Technology&sortBy=&order=

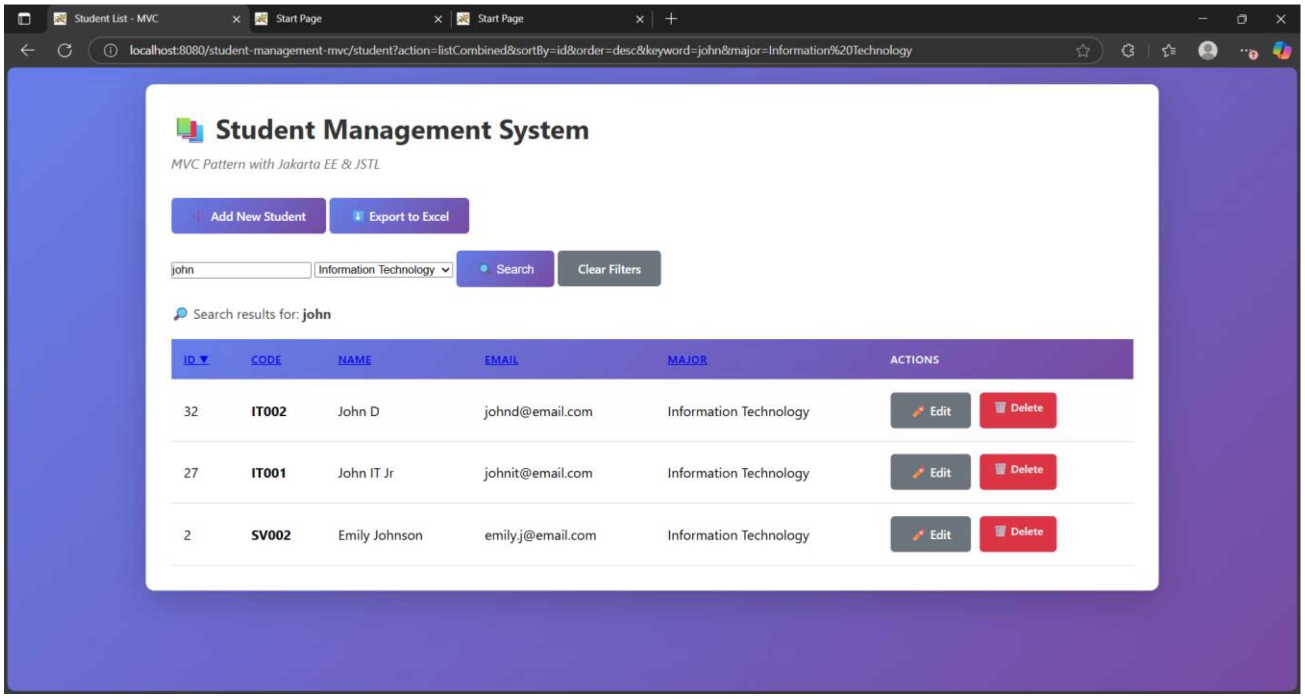
Student Management System

MVC Pattern with Jakarta EE & JSTL

Search results for: **john**

ID	CODE	NAME	EMAIL	MAJOR	ACTIONS
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	Edit Delete
27	IT001	John IT Jr	johnit@email.com	Information Technology	Edit Delete
32	IT002	John D	johnd@email.com	Information Technology	Edit Delete

Search results for 'john' & filtered by 'Information Technology'


localhost:8080/student-management-mvc/student?action=listCombined&sortBy=id&order=desc&keyword=john&major=Information%20Technology

Student Management System

MVC Pattern with Jakarta EE & JSTL

Search results for: **john**

ID ▼	CODE	NAME	EMAIL	MAJOR	ACTIONS
32	IT002	John D	johnd@email.com	Information Technology	Edit Delete
27	IT001	John IT Jr	johnit@email.com	Information Technology	Edit Delete
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	Edit Delete

Search results for 'john' & filtered by 'Information Technology' & in 'descending order'

*A similar problem was solved in Lab04, which I took as reference

Implementation

StudentController.java

```
private void listCombined(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

    String keyword = request.getParameter("keyword");
    String major = request.getParameter("major");
    String sortBy = request.getParameter("sortBy");
    String order = request.getParameter("order");
    List<Student> students = studentDAO.getStudentsCombined(keyword,
major, sortBy, order);


    request.setAttribute("students", students);
    request.setAttribute("keyword", keyword != null ? keyword : "");
    request.setAttribute("selectedMajor", major != null ? major : "");
    request.setAttribute("sortBy", sortBy != null ? sortBy : "id");
    request.setAttribute("order", order != null ? order : "asc");


    RequestDispatcher dispatcher =
request.getRequestDispatcher("/views/student-list.jsp");
    dispatcher.forward(request, response);
}
```

StudentDao.java

```
public List<Student> getStudentsCombined(String keyword, String major,
String sortBy, String order) {
    List<Student> students = new ArrayList<>();
    StringBuilder sql = new StringBuilder("SELECT * FROM students WHERE
1=1");


    if (keyword != null && !keyword.trim().isEmpty()) {
        sql.append(" AND (student_code LIKE ? OR full_name LIKE ? OR email
LIKE ?)");
    }


    if (major != null && !major.trim().isEmpty()) {
        sql.append(" AND major = ?");
    }

    sql.append(" ORDER BY ").append(validateSortBy(sortBy)).append("
").append(validateOrder(order));

    try (Connection conn = getConnection();
         PreparedStatement pstmt = conn.prepareStatement(sql.toString())) {

        int index = 1;
        if (keyword != null && !keyword.trim().isEmpty()) {
            String pattern = "%" + keyword.trim() + "%";
            pstmt.setString(index++, pattern);
            pstmt.setString(index++, pattern);
            pstmt.setString(index++, pattern);
        }

        if (major != null && !major.trim().isEmpty()) {
            pstmt.setString(index++, major);
        }

        ResultSet rs = pstmt.executeQuery();
        while (rs.next()) {
            Student student = new Student();
            student.setId(rs.getInt("id"));
            student.setStudentCode(rs.getString("student_code"));
```

```
          student.setFullName(rs.getString("full_name"));
          student.setEmail(rs.getString("email"));
          student.setMajor(rs.getString("major"));
          student.setCreatedAt(rs.getTimestamp("created_at"));
          students.add(student);
       }

   } catch (SQLException e) {
       e.printStackTrace();
   }

   return students;

}
```

References:

1. https://poi.apache.org/apidocs/dev/org/apache/poi/xssf/usermodel/XSSFWorkbook.html
2. https://stackoverflow.com/questions/5985318/apache-poi-xssf-reading-in-excel-files
3. https://viblo.asia/p/huong-dan-doc-va-ghi-file-excel-trong-java-su-dung-thu-vien-apache-poi-RQqKLENpZ7z
4. https://www.baeldung.com/java-dao-pattern
5. https://jakarta.ee/specifications/tags/
6. https://docs.oracle.com/javaee/7/tutorial/servlets.htm