Lab 6 Excersise

Name: Nguyễn Hữu Hoàng Nam

StudentID: ITCSIU23028

EXERCISE 5:

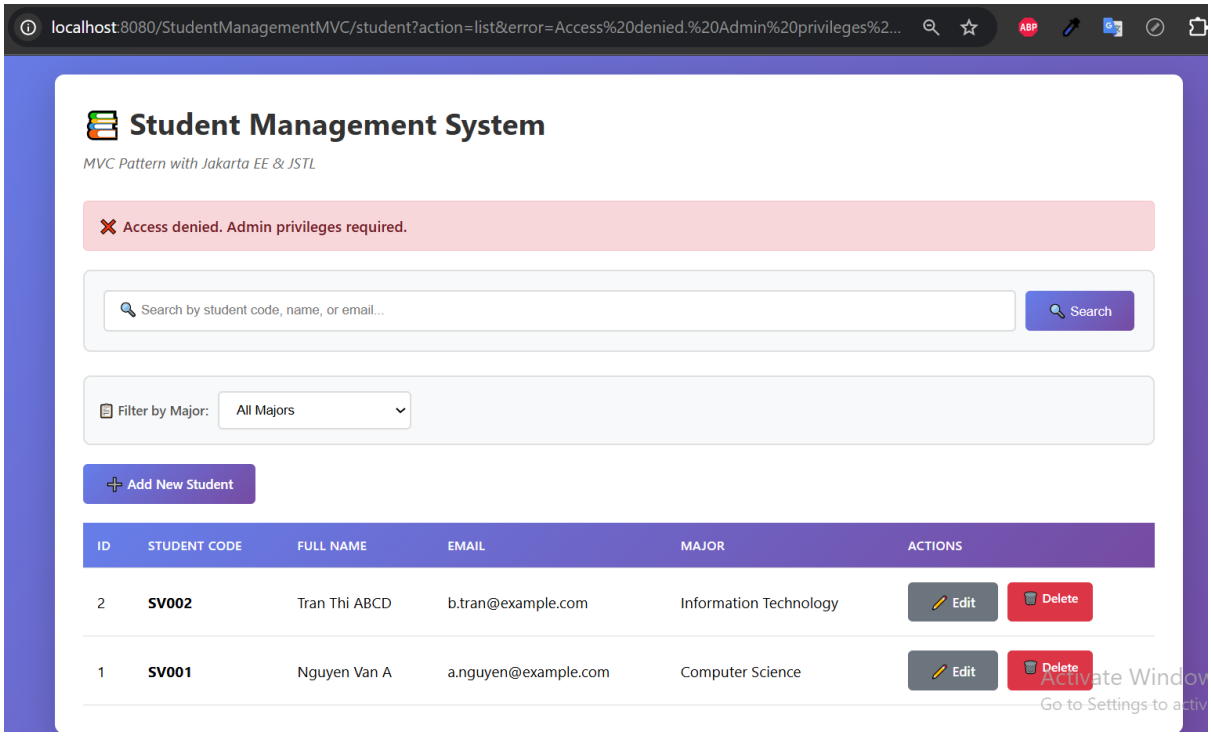
```
package com.student.filter;

import javax.servlet.*;
import javax.servlet.annotation.WebFilter;
import javax.servlet.http.*;
import java.io.IOException;
@WebFilter(filterName = "AuthFilter", urlPatterns = {"/*"})
public class AuthFilter implements Filter {
    private static final String[] PUBLIC_URLS = {
        "/login",
        "/logout",
        ".css",
        ".js",
        ".png",
        ".jpg",
        ".jpeg",
        ".gif",
        ".ico"
    };
    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
        System.out.println("AuthFilter initialized - protecting application routes");
    }
    @Override
    public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
            throws IOException, ServletException {
        HttpServletRequest httpRequest = (HttpServletRequest) request;
        HttpServletResponse httpResponse = (HttpServletResponse) response;
        String requestURI = httpRequest.getRequestURI();
        String contextPath = httpRequest.getContextPath();
        String path = requestURI.substring(contextPath.length());
        if (isPublicUrl(path)) {
            chain.doFilter(request, response);
            return;
        }
        HttpSession session = httpRequest.getSession(false);
        boolean isLoggedIn = (session != null && session.getAttribute("user") != null);
        if (isLoggedIn) {
            chain.doFilter(request, response);
        } else {
            httpResponse.sendRedirect(contextPath + "/login");
        }
```

```
    }
    @Override
    public void destroy() {
        System.out.println("AuthFilter destroyed");
    }
    private boolean isPublicUrl(String path) {
        if (path.equals("/") || path.isEmpty()) {
            return true;
        }
        if (path.equals("/login") || path.startsWith("/login")) {
            return true;
        }
        if (path.equals("/logout") || path.startsWith("/logout")) {
            return true;
        }
        for (String publicUrl : PUBLIC_URLS) {
            if (publicUrl.startsWith(".") && path.endsWith(publicUrl)) {
                return true;
            }
        }
        return false;
    }
}
```

Flow:

1. User access resource
2. Activate AuthFilter
3. Take out request url
4. Check whether it is public url or not
5. If public url, allow the request go through to servlet
   If not, get user session
   If they have it, allow them to continue
   Else, redirect to /login page

EXERCISE 6:




```java
package com.student.filter;
import com.student.model.User;
import javax.servlet.*;
import javax.servlet.annotation.WebFilter;
import javax.servlet.http.*;
import java.io.IOException;
@WebFilter(filterName = "AdminFilter", urlPatterns = {"/student"})
public class AdminFilter implements Filter {
    private static final String[] ADMIN_ACTIONS = {
        "new",
        "insert",
        "edit",
        "update",
        "delete"
    };
    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
        System.out.println("AdminFilter initialized - protecting admin-only
actions");
    }
    @Override
    public void doFilter(ServletRequest request, ServletResponse response,
FilterChain chain)
            throws IOException, ServletException {
        HttpServletRequest httpRequest = (HttpServletRequest) request;
        HttpServletResponse httpResponse = (HttpServletResponse) response;
        String action = httpRequest.getParameter("action");
        if (isAdminAction(action)) {
            HttpSession session = httpRequest.getSession(false);
```

```
            User user = null;
            if (session != null) {
                user = (User) session.getAttribute("user");
            }
            if (user != null && user.isAdmin()) {
                chain.doFilter(request, response);
            } else {
                String contextPath = httpRequest.getContextPath();
                httpResponse.sendRedirect(contextPath +
"/student?action=list&error=Access denied. Admin privileges required.");
            }
        } else {
            chain.doFilter(request, response);
        }
    }
    @Override
    public void destroy() {
        System.out.println("AdminFilter destroyed");
    }
    private boolean isAdminAction(String action) {
        if (action == null || action.trim().isEmpty()) {
            return false;
        }
        for (String adminAction : ADMIN_ACTIONS) {
            if (adminAction.equalsIgnoreCase(action)) {
                return true;
            }
        }
        return false;
    }
}
```

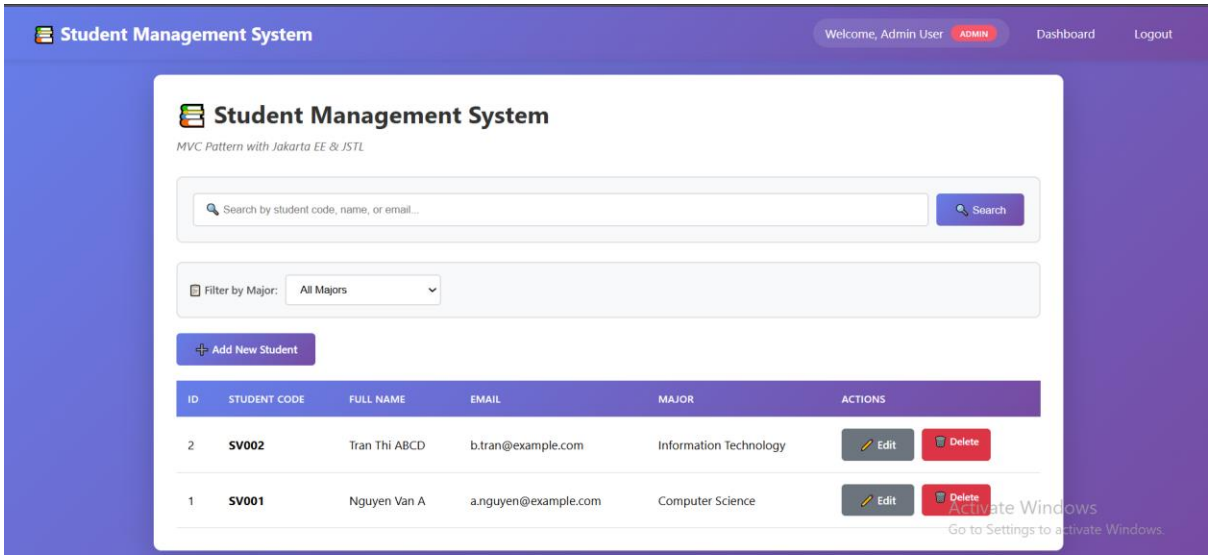
Flow:

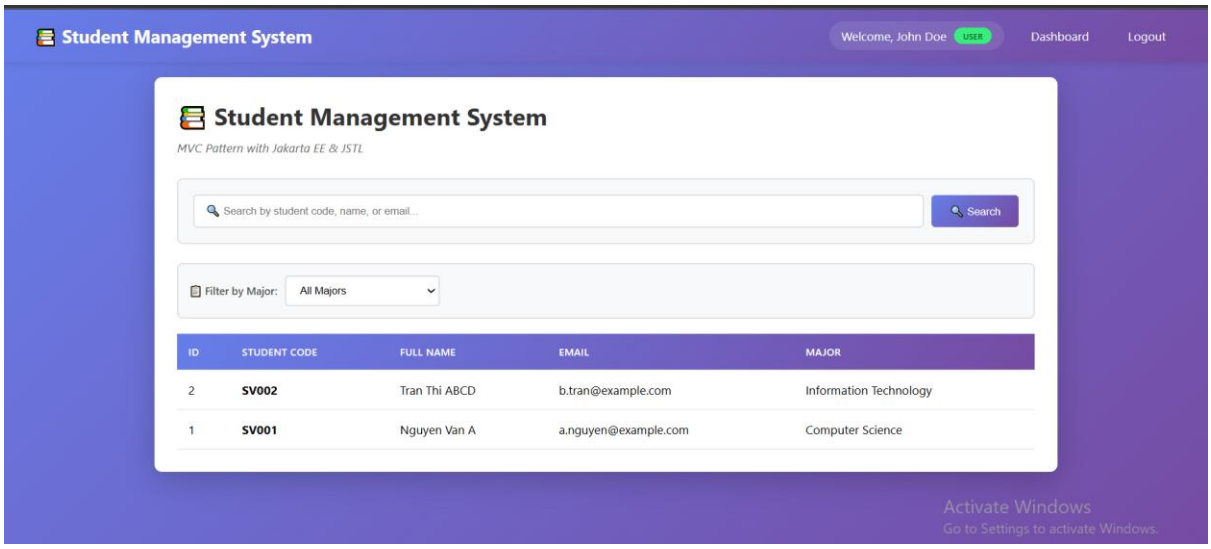
1. User take action through url
2. Filter check whether it is admin action or not
3. If not, continue to controller
   If admin action, get session and check what type of user

   If not admin, redirect with query param error

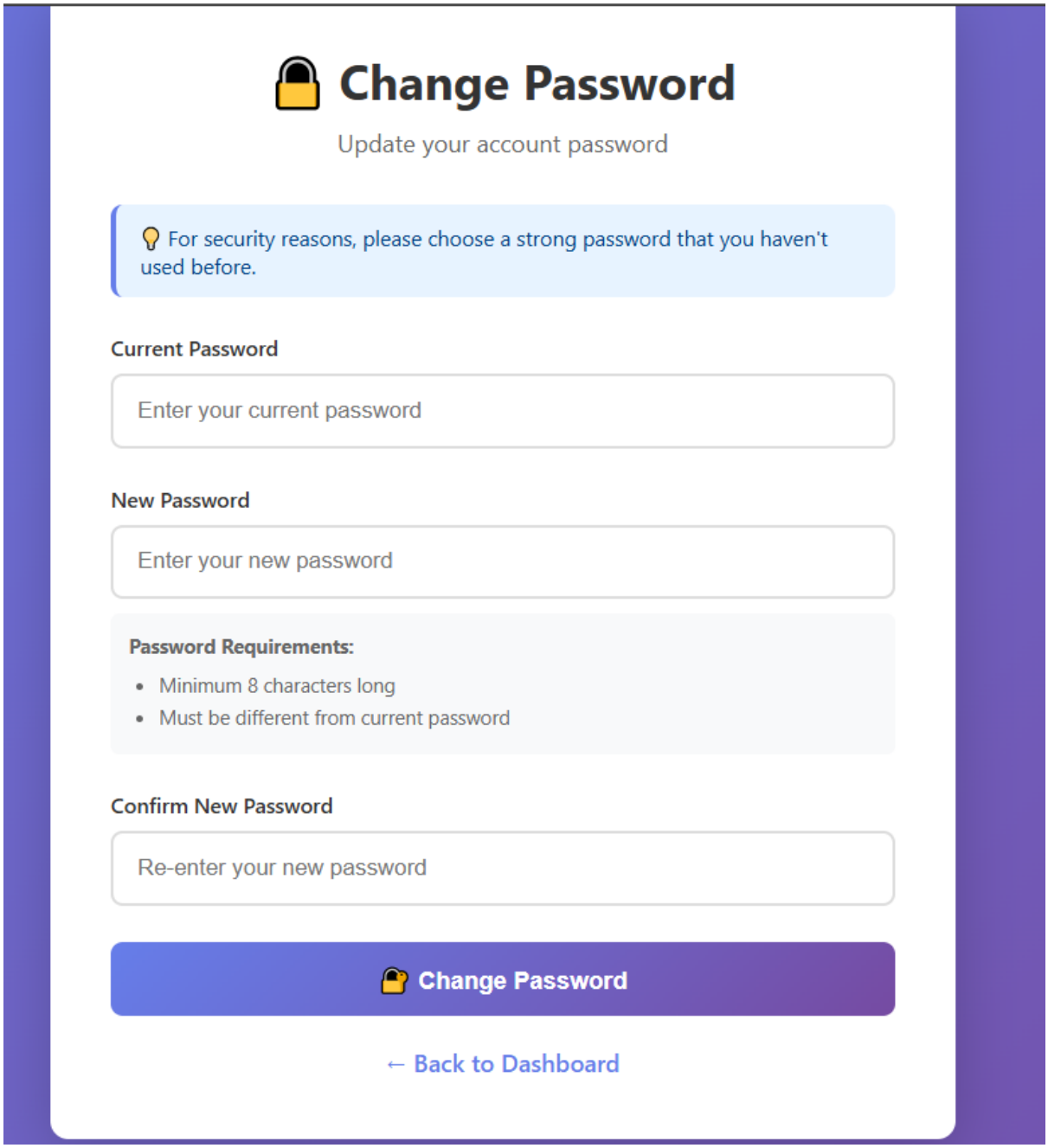
   If admin, allow to go through filter

Exercise 7:


Student Management System
Welcome, Admin User ADMIN
Dashboard
Logout
Student Management System
MVC Pattern with Jakarta EE & JSTL
Search by student code, name, or email...
Search
Filter by Major:
All Majors
Add New Student
ID
STUDENT CODE
FULL NAME
EMAIL
MAJOR
ACTIONS
2
SV002
Tran Thi ABCD
b.tran@example.com
Information Technology
Edit
Delete
1
SV001
Nguyen Van A
a.nguyen@example.com
Computer Science
Edit
Delete
Activate Windows
Go to Settings to activate Windows.


Student Management System
Welcome, John Doe USER
Dashboard
Logout
Student Management System
MVC Pattern with Jakarta EE & JSTL
Search by student code, name, or email...
Search
Filter by Major:
All Majors
ID
STUDENT CODE
FULL NAME
EMAIL
MAJOR
2
SV002
Tran Thi ABCD
b.tran@example.com
Information Technology
1
SV001
Nguyen Van A
a.nguyen@example.com
Computer Science
Activate Windows
Go to Settings to activate Windows.

Exercise 8:



Flow:

1. User clicks "Change Password"
2. User logged in? → NO → Redirect to /login
   ↓ YES → Continue
3. Show change-password.jsp form
4. User fills form and submits
5. Verify current password with BCrypt
6. Hash new password with BCrypt
7. Update database

8. Redirect to /dashboard?message=Success