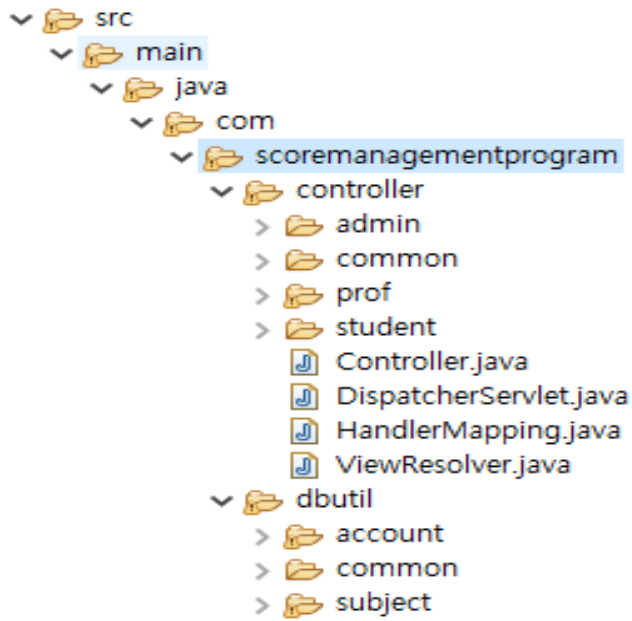
학생성적관리프로그램

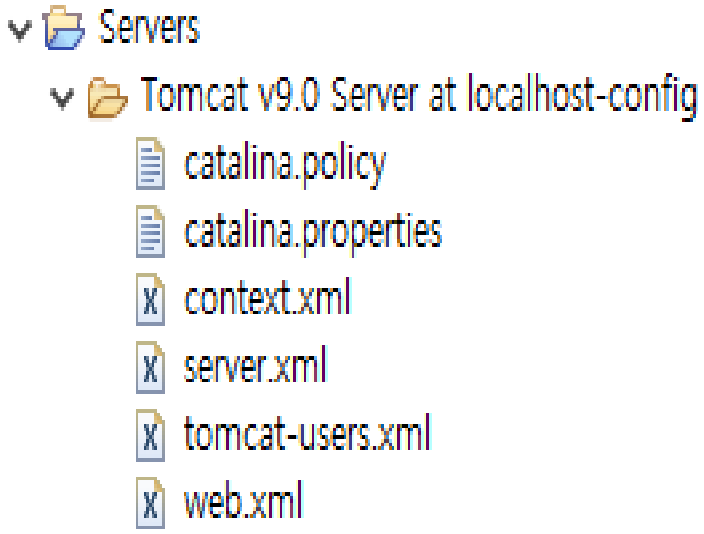
백앤드_27회차
김명찬
2024/10/16

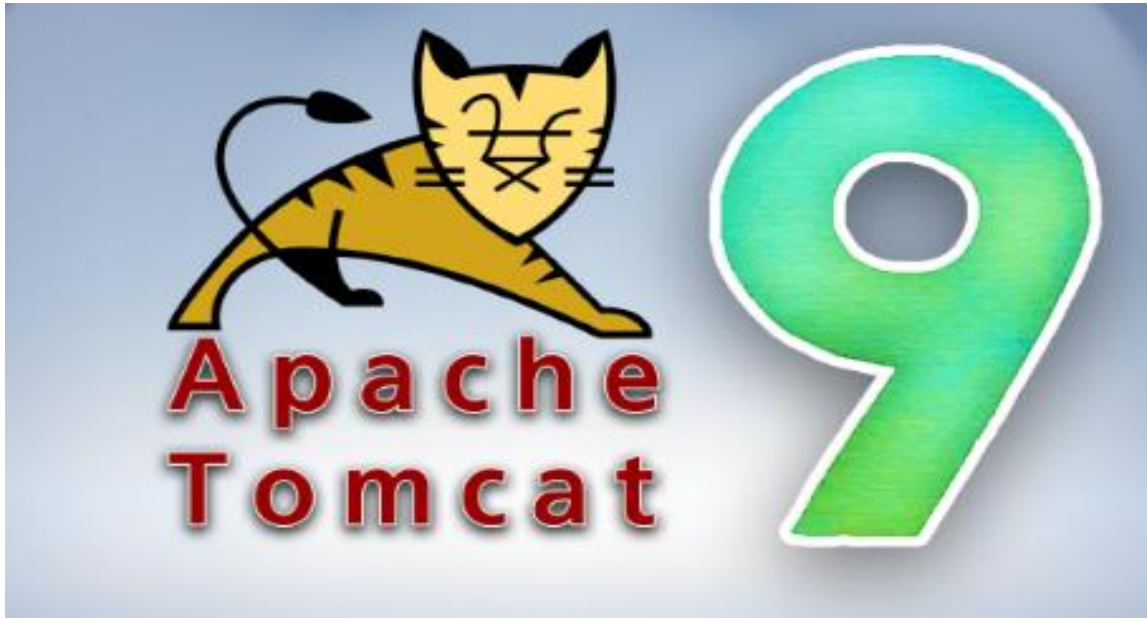
- **목차**

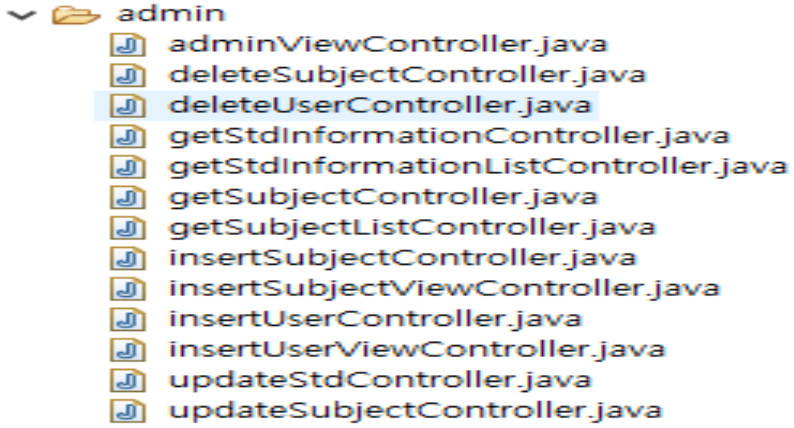
- **프로젝트 구조 설명**


src
main
java
com
scoremanagementprogram
controller
admin
common
prof
student
Controller.java
DispatcherServlet.java
HandlerMapping.java
ViewResolver.java
dbutil
account
common
subject


webapp
css
image
js
META-INF
WEB-INF
layout
lib
user
web.xml
index.jsp


Servers
Tomcat v9.0 Server at localhost-config
catalina.policy
catalina.properties
context.xml
server.xml
tomcat-users.xml
web.xml


Apache
Tomcat
9



- **Controller (브라우저를 통해 클라이언트가 보내는 요청을 수신하고, 해당 요청을 처리하는 중심적인 역할을 담당)**
- **admin : 관리자 요청 처리에 대한 디렉토리**

- **adminViewController : admin.jsp(관리자 전용 페이지) 로 이동요청에 대한 처리 담당**

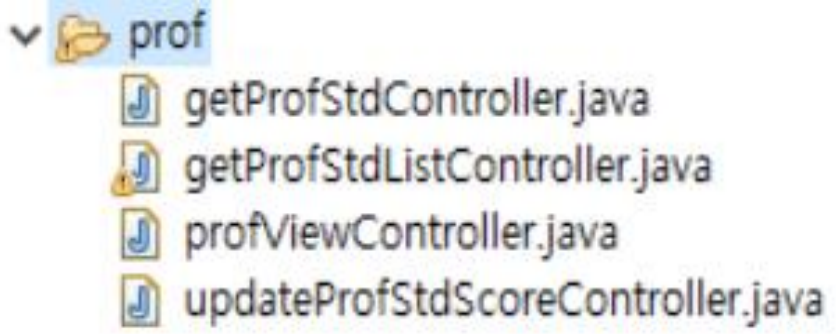
- **getSubjectListController : 전체 학생 성적정보 요청을 처리하고 getSubjectList.jsp 로 정보를 전달하는 역할 담당**
- **getSubjectController : 한 학생에 대한 성적정보 요청을 처리하고 getSubject.jsp 로 정보를 전달하는 역할 담당**

- **getStdInformationListController : 전체 학생에 대한 계정정보 요청을 처리하고 getStdInformationList.jsp 로 정보전달을 담당**
- **getStdInformationController : 한 학생에 대한 계정정보 요청을 처리하고 getStdInformation.jsp로 정보전달을 담당**
- **insertSubjectViewController : 학생 성적정보를 추가하기 위한 페이지 (insertSubject.jsp) 로 이동요청을 처리하는 역할을 담당**
- **insertSubjectController : insertSubject.jsp 에서 정보를 입력했을 때 DB에 정보를 저장하는 역할을 담당**

- **insertUserViewController : 회원의 계정를 추가하기 위한 페이지 (insertUser.jsp) 로 이동요청을 처리하는 역할을 담당**
- **insertUserController : 계정정보를 입력하면 DB에 정보를 저장하는 역할을 담당**

- **updateSubjectController : DB에 저장되어 있는 학생 성적정보를 수정하는 역할을 담당**
- **updateStdController : DB에 저장되어 있는 회원의 계정정보를 수정하는 역할을 담당**

- **deleteSubjectController : DB에 저장되어 있는 학생 성적정보를 삭제하는 역할을 담당**
- **deleteUserController : DB에 저장되어 있는 회원의 계정정보를 삭제하는 역할을 담당**

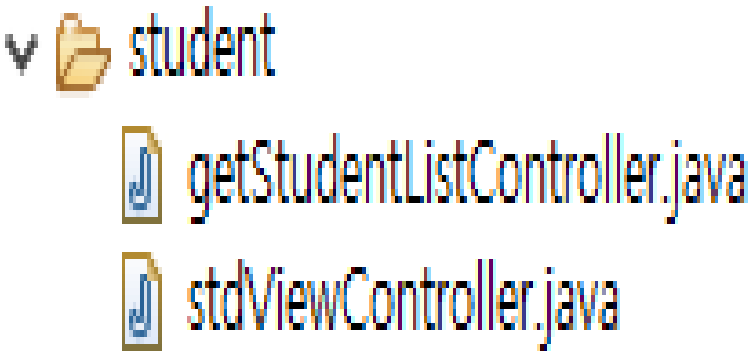


- **Controller (브라우저를 통해 클라이언트가 보내는 요청을 수신하고, 해당 요청을 처리하는 중심적인 역할을 담당)**
- **prof : 교수 요청 처리 디렉토리**

- **profViewController: prof.jsp(교수 전용 페이지)로 이동 요청에 대한 처리 담당**

- **getProfStdController : 교수가 담당하는 학생 성적 정보에 대한 요청을 처리하고 성적 정보를 getProfStdSubject.jsp로 전달**
- **getProfStdListController : 교수가 담당하는 전체 학생 성적 정보에 대한 요청을 처리 성적정보를 getStdInformationList.jsp로 전달**

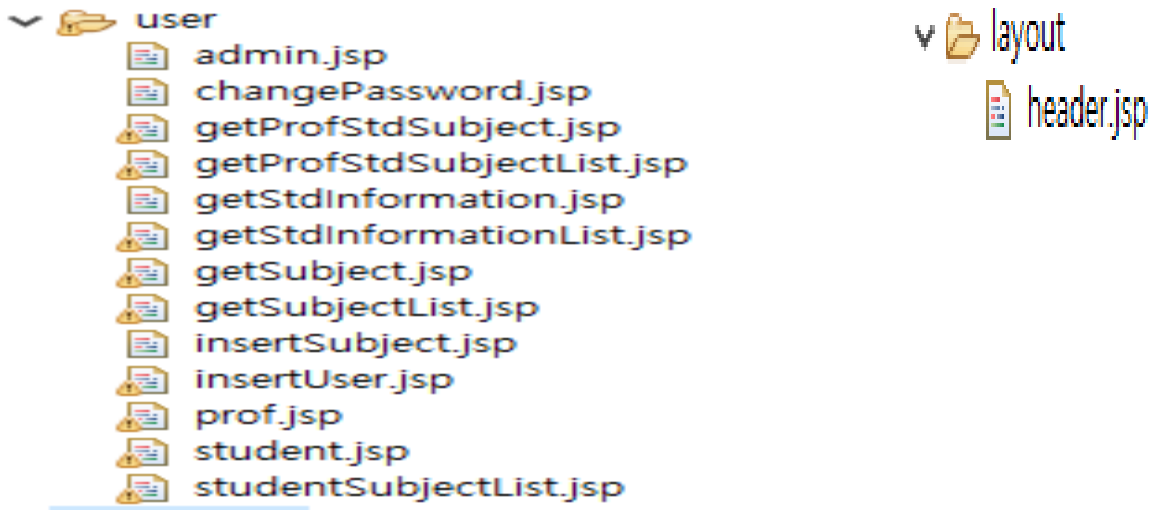
- **updateProfStdScoreController : DB에 저장된 학생에 대한 성적을 수정하는 요청 처리를 담당**



- **Controller (브라우저를 통해 클라이언트가 보내는 요청을 수신하고, 해당 요청을 처리하는 중심적인 역할을 담당)**
- **Student : 학생 요청 처리를 위한 디렉토리**

- **stdViewController : student.jsp (학생 전용 페이지) 로 이동 요청에 대한 처리 담당**

- **getStudentListController : 학생이 신청되어 있는 수업에 대한 정보를 studentSubjectList.jsp로 전달**



- View : 사용자가 보는 화면을 담당 jsp가 주로 사용되며, 컨트롤러가 넘겨준 데이터를 바탕으로 html을 생성하여 사용자에게 보여줌

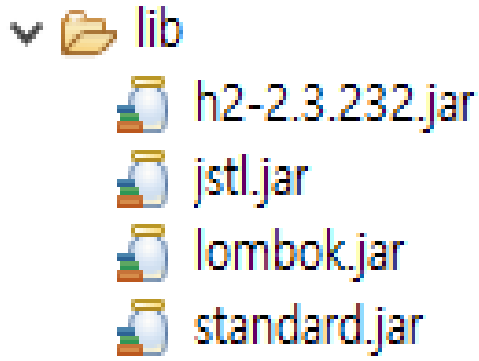
- admin.jsp : 관리자 권한으로 로그인할 때 처음 보여지는 페이지
- prof : 교수 권한으로 로그인할 때 처음 보여지는 페이지
- student : 학생 권한으로 로그인할 때 처음 보여지는 페이지

- getSubjectList.jsp : 관리자 전용 전체 학생 성적정보를 보여주는 페이지
- getSubject.jsp : 관리자 전용 List.jsp 페이지에서 학생의 학번을 클릭할 시 해당학생의 성적정보를 보여주는 페이지
- getStdInformationList.jsp : 관리자 전용 전체 학생의 계정정보를 보여주는 페이지
- getStdInformation.jsp : List.jsp에서 학번을 클릭 시 해당학생의 계정정보를 보여주는 페이지
- insertSubject.jsp : 관리자 전용 학생 성적정보를 등록하기 위한 페이지
- insertUser.jsp : 관리자 전용 학생정보를 등록하기 위한 페이지

- getProfStdSubjectList.jsp : 교수 전용 교수가 담당하는 학생 성적정보를 보여주는 페이지
- getProfStdSubject.jsp : 교수 전용 List.jsp 페이지에서 학생의 학번을 클릭할 시 해당학생의 성적정보를 보여주는 페이지

- studentSubjectList.jsp : 학생 전용 학생이 수강하는 전체 수업을 보여주는 페이지

- changePassword.jsp : 교수/학생 이 비밀번호를 수정할 수 있는 페이지
- header.jsp : 관리자/교수/학생 로그아웃 기능을 제공해주는 페이지

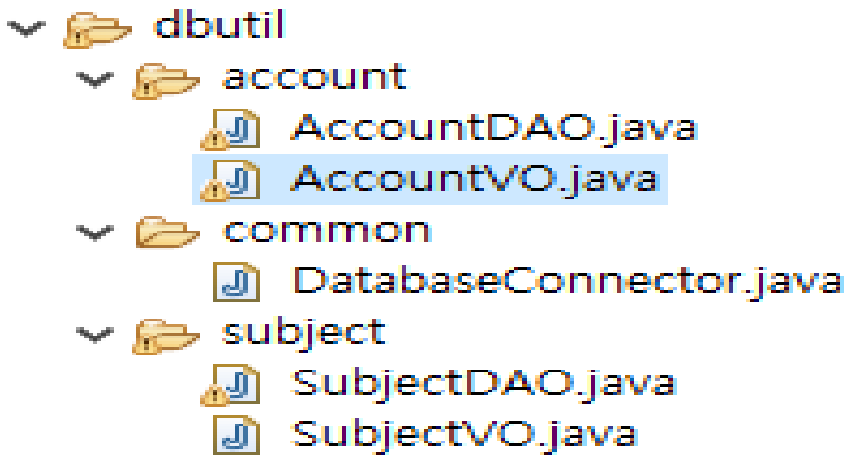


- **h2-2.3.232.jar : h2 데이터베이스의 (2.3.232 버전) JDBC 드라이버 이며, 자바 애플리케이션에서 쉽게 H2 데이터베이스와 상호작용할 수 있게 해준다.**

- **jstl.jar : JSTL은 JSP 페이지에서 사용할 수 있는 표준 태그 라이브러리로, JSP 코드에서 자바코드 삽입을 최소화하고 태그를 이용하여 로직을 처리할 수 있게 해준다. (이를 통해 JSP 페이지에서 비즈니스 로직과 뷰 로직의 분리가 보다 명확해짐)**

- **standard.jar : JSP에서 JSTL을 사용하기 위한 표준 구현 라이브러리**
- **JSTL 표준 태그 라이브러리를 JSP에서 사용하려면 필수적으로 추가되어야 하는 JAR 파일**
- **JSTL 태그를 처리하는 데 필요한 구현체들을 포함하고 있으며, JSP에서 조건문,반복문 등을 구현할 수 있게 해준다.**
- **jstl.jar와 함께 사용됨**

- **lombok.jar : Lombok은 자바에서 반복적인 코드를 줄이기 위한 라이브러리**
- **자바 코드에서 Getter/Setter나 equals, hashCode, toString 메서드와 같은 보일러플레이트 코드를 직접 작성할 필요 없이 어노테이션으로 대체 가능**



- **Model : 데이터와 비즈니스 로직을 담당하는 핵심 부분으로, 애플리케이션의 데이터 관리와 관련된 작업을 수행함**
- **(클래스의 메소드마다 어떤 메소드인지 설명해놓았습니다.)**

- **DatabaseConnector : 애플리케이션과 데이터베이스 사이의 연결을 설정하고, 데이터베이스 자원을 효율적으로 관리하기 위해 자원을 해제하는 (close) 메소드가 작성되어 있는 클래스**

- **AccountVO : 회원 계정에 관련된 필드들을 담고 있고 getter,setter 메소드를 통해 필드를 수정 또는 가져올 수 있게 만들어놓은 데이터 저장 클래스**
- **AccountDAO : 직접적으로 DB에 회원 계정에 대한 정보를 추가 또는 수정 등 작업을 할 수 있게 쿼리와 메소드를 담아놓은 데이터 접근 클래스**

- **SubjectVO : 학생 성적 정보에 관련된 변수들을 담고 있고 getter,setter 메소드를 통해 필드를 수정 또는 가져올 수 있게 만들어놓은 데이터 저장 클래스**
- **SubjectDAO : 직접적으로 DB에 학생 성적 정보에 대한 정보를 추가 또는 수정 등 작업을 할 수 있게 쿼리와 메소드를 담아놓은 데이터 접근 클래스**

- **2. 데이터베이스 설명**

show columns from account;

FIELD	TYPE	NULL	KEY	DEFAULT
ID	CHARACTER VARYING(20)	NO	PRI	NULL
PASSWORD	CHARACTER VARYING(20)	NO		NULL
ROLE	CHARACTER VARYING(5)	NO		NULL
NAME	CHARACTER VARYING(6)	NO		NULL

- **ACCOUNT – 성적프로그램을 사용하는 모든 회원의 계정정보를 저장하고 있는 테이블**

- **ID : 회원의 아이디를 저장하는 필드**
- **PASSWORD : 회원의 비밀번호를 저장하는 필드**
- **ROLE : 회원의 권한을 저장하는 필드**
- **NAME : 회원의 이름을 저장하는 필드**

```
show columns from subject;
```

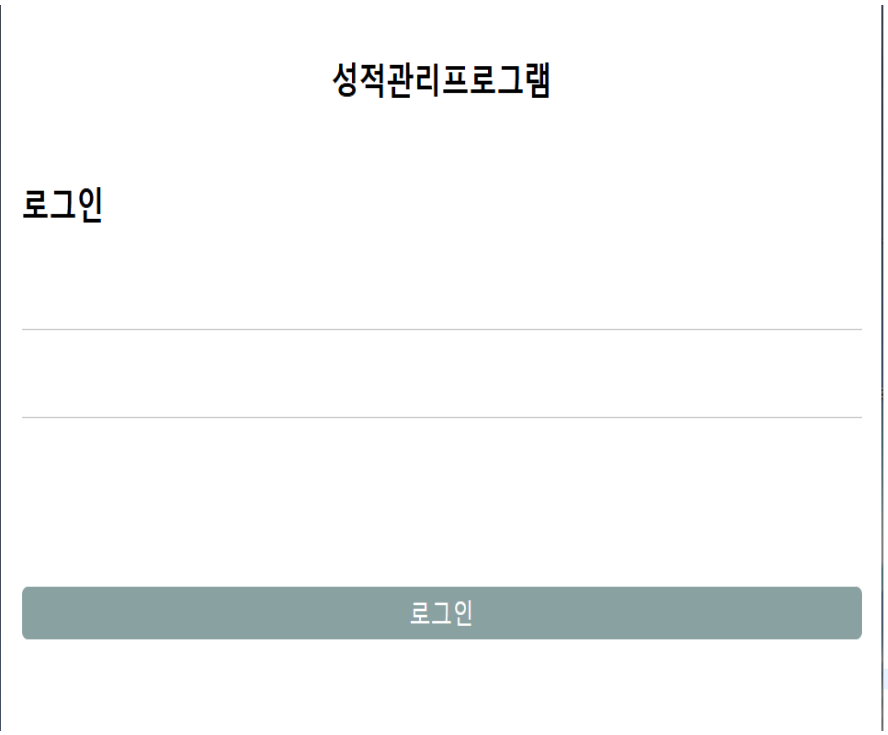
FIELD	TYPE	NULL	KEY	DEFAULT
ID	CHARACTER(8)	NO		NULL
UV_YEAR	CHARACTER(4)	NO		NULL
SEMESTER	CHARACTER(1)	NO		NULL
COURSE	CHARACTER VARYING(6)	NO		NULL
SUBJECT_NAME	CHARACTER VARYING(100)	NO		NULL
CREDIT	CHARACTER(3)	NO		NULL
PROFESSOR	CHARACTER VARYING(6)	NO		NULL
SCORE	CHARACTER VARYING(2)	NO		NULL

- ***SUBJECT – 학생 성적 정보를 저장하는 테이블***

- ***ID : 학생의 학번을 저장하는 필드 ex : (2024XXXX) 8자리 고정***
- ***UV_YEAR : 해당 과목의 수강년도를 저장하는 필드***
- ***SEMESTER : 해당 과목의 수강학기를 저장하는 필드***
- ***COURSE : 해당 과목의 이수구분을 저장하는 필드***
- ***SUBJECT_NAME : 수강하는 과목의 이름을 저장하는 필드***
- ***CREDIT : 수강 과목의 학점을 저장하는 필드***
- ***PROFESSOR : 수강과목의 담당 교수의 이름을 저장하는 필드***
- ***SCORE : 수강과목의 성적을 저장하는 필드***

• 3. 기능 설명

- 로그인 (공통)




```
    String id = request.getParameter("id");  //아이디 입력값 가져오기
    String password = request.getParameter("password"); //비밀번호 입력값 가져오기

    AccountVO vo = new AccountVO();
    AccountDAO dao = new AccountDAO();
    HttpSession session = request.getSession();

    //로그인 창에서 입력받은 아이디를 vo 객체의 아이디 값에 넣음
    vo.setId(id);


    //db에서 입력받은 id가 있으면 user에 아이디 비밀번호 권한 값이 들어가고 아니면 user는 null
    AccountVO user = dao.getUser(vo);
    System.out.println(user.getPassword());


    //입력한 아이디가 db에 있고 입력한 아이디에 대한 비밀번호가 일치할 때
    if(user != null && user.getPassword() != null && user.getPassword().equals(password)) {
        session.setAttribute("user", user); //회원정보를 세션에 등록

        if(user.getRole().equals("ADMIN")) { // 회원의 권한이 관리자일 때
            return "/adminView.do"; //관리자 페이지로 이동
        }
        else if(user.getRole().equals("PROF")) { //회원의 권한이 교수일 때
            return "/profView.do"; //교수 페이지로 이동
        }
        else { //관리자 또는 교수가 아닐 때 (학생일 때)
            return "/stdView.do"; //학생 페이지로 이동
        }

    }
    else {
        return "index";
    }
}
```

로그아웃

관리자 님 환영합니다.

회원등록 | 회원성적조회 | 회원정보조회 | 학생성적정보추가

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>



<c:if test="${sessionScope.user != null}">
    <a href="logout.do">로그아웃</a>   
</c:if>
<hr>
<br>
```

로그인을 하게 되면 세션에 회원 정보가 담긴 객체(user)가 등록이 되는데 로그인 할 때만 로그아웃이 페이지에 뜨게 하기 위해서 설정함.

로그아웃

관리자 님 환영합니다.

회원등록	회원성적조회	회원정보조회	학생성적정보추가
			

- 관리자

- 회원등록/회원조회/회원수정/회원삭제/
- 회원성적추가/회원성적조회/회원성적수정/회원성적삭제


```
<header>
        <h2>회원정보 입력</h2>
    </header>

    <form action="insertUser.do" method="POST">

        <div class="input-box">
            아이디  <input id="username" type="text" name="id" placeholder="아이디">

        </div>

        <div class="input-box">
            비밀번호  <input id="password" type="password" name="password" placeholder="비밀번호">
        </div>

         <div class="input-box">
            이름  <input id="name" type="text" name="name" placeholder="이름">
        </div>

        <div > 권한
            <input type="radio" name="role" value="ADMIN">관리자
            <input type="radio" name="role" value="PROF">교수
            <input type="radio" name="role" value="STD">학생
        </div>

        <input type="submit" value="회원등록">
    </form>

    <br>
    <center>
        <a href="adminView.do"> 홈으로 </a>
    </center>
```

insertUserController.java
insertUserViewController.java

회원등록

관리자 페이지에 있는 회원등록 이미지 -> (insertUserView.do 요청이) --> dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → insertSubjectController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동

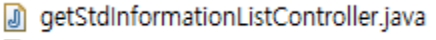
```
RequestDispatcher dispatcher = request.getRequestDispatcher(view);
dispatcher.forward(request, response);
```

전체학생정보

아이디(학번)	패스워드	이름	권한
admin	admin123	관리자	ADMIN
professor01	professor01	이교수	PROF
20242222	20242222	박구영	STD
20241111	20241111	안경원	STD
20191111	20191111	김태이	STD
professor02	professor02	박장그	PROF

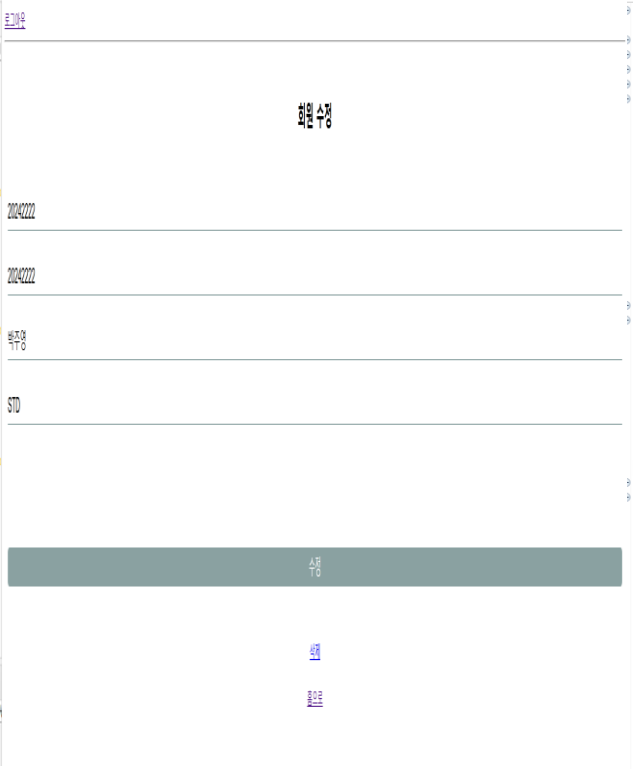
로그아웃

```
<center>
    <h1>전체학생정보</h1>
    <form action ="/getSubjectList.do" method="post" >
        <table border="1" cellpadding="0" cellspacing="0" width="700">
        <tr>
            <td align="right">
            <select name="searchCondition">
                <option value="stdId" <c:if test="${condition == '학번'}"> selected</c:if>>학번

            </select>

                <input name="searchKeyword" type="text" value="${keyword}"/>
                <input type="submit" value="검색"/>
            </td>
        </tr>
        </table>
    </form>



    <table border="1" cellpadding="0" cellspacing="0" width="700">
        <tr>
            <th bgcolor="orange" width="100">년도</th>
            <th bgcolor="orange" width="100">학기</th>
            <th bgcolor="orange" width="200">학번</th>
            <th bgcolor="orange" width="150">이수구분</th>
            <th bgcolor="orange" width="200">과목</th>
            <th bgcolor="orange" width="100">학점</th>
            <th bgcolor="orange" width="100">담당교수</th>
            <th bgcolor="orange" width="100">학점</th>
        </tr>
<!-- 리스트 불러오는 도중에 에러가 생김 확인 해야됨로 -->
    <c:forEach var="list" items="${subjectList}">
       <tr>
           <td>${list.year}</td> <!-- 년도 -->
           <td> ${list.semester} </td> <!-- 학기 -->
           <td> <a href="getSubject.do?id=${list.id}&subject_name=${list.subjectName}"> ${list.id} </td> </a> <!-- 학번 -->
           <td>${list.course}</td> <!-- 이수구분 -->
           <td>${list.subjectName}</td> <!-- 과목 -->
           <td>${list.credit}</td> <!-- 학점 -->
           <td>${list.professor}</td> <!-- 담당교수 -->
           <td>${list.score}</td> <!-- 학점 -->
       </tr>
    </c:forEach>

    </table>

    <br>

    <a href="adminView.do"> 홈으로 </a>

</center>
```



- **전체 회원정보조회**

- **관리자 페이지에 있는 회원정보조회 이미지 -> (getStdInformationList.do 요청이) --> dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → getStdInformationListController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**




```jsp
<center>
    <h1>전체학생정보</h1>
    <form action ="/getSubjectList.do" method="post" >
        <table border="1" cellpadding="0" cellspacing="0" width="700">
        <tr>
            <td align="right">
            <select name="searchCondition">
                <option value="stdId" <c:if test="${condition == '학번'}"> selected</c:if>>학번

            </select>

                <input name="searchKeyword" type="text" value="${keyword}"/>
                <input type="submit" value="검색"/>
            </td>
        </tr>
        </table>
    </form>



    <table border="1" cellpadding="0" cellspacing="0" width="700">
        <tr>
            <th bgcolor="orange" width="100">년도</th>
            <th bgcolor="orange" width="100">학기</th>
            <th bgcolor="orange" width="200">학번</th>
            <th bgcolor="orange" width="150">이수구분</th>
            <th bgcolor="orange" width="200">과목</th>
            <th bgcolor="orange" width="100">학점</th>
            <th bgcolor="orange" width="100">담당교수</th>
            <th bgcolor="orange" width="100">학점</th>
        </tr>
<!-- 리스트 불러오는 도중에 에러가 생김 확인 해야될듯 -->
    <c:forEach var="list" items="${subjectList}">
       <tr>
            <td>${list.year}</td> <!-- 년도 -->
            <td> ${list.semester} </td> <!-- 학기 -->
            <td> <a href="getSubject.do?id=${list.id}&subject_name=${list.subjectName}"> ${list.id} </td> </a> <!-- 학번 -->
            <td>${list.course}</td> <!-- 이수구분 -->
            <td>${list.subjectName}</td> <!-- 과목 -->
            <td>${list.credit}</td> <!-- 학점 -->
            <td>${list.professor}</td> <!-- 담당교수 -->
            <td>${list.score}</td> <!-- 학점 -->
       </tr>
    </c:forEach>

    </table>

    <br>

    <a href="adminView.do"> 홈으로 </a>

</center>
```

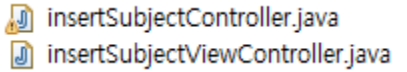
deleteUserController.java
getStdInformationController.java
updateStdController.java

- **회원정보조회/수정/삭제**

- **전체회원정보 페이지에서 학번을 누르면 --> (getStdInformation.do 요청) --> dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → getStdInformationController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**

- **정보를 수정하고 수정 버튼을 누르면 -> (*updateStd.do?stdid=*${account.id} 요청(아이디 값이 같이 넘어감)) --> dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → updateStdController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**

- **삭제를 누르면 → *deleteUser.do?stdid=*${account.id} → dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → deleteUserController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**



- **회원성적추가**

- **관리자 페이지에 있는 회원성적추가 이미지 -> (insertSubjectView.do 요청이) --> dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → insertSubjectController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**

전체학생정보

학번 검색

년도	학기	학번	이수구분	과목	학점	담당교수	학점
2023	1	20191111	전공	자바프로그래밍	3	이영표	B+
2023	1	20241111	전공	테스트용	3	이영표	A
2024	1	20242222	전공	테스트용	3	히딩크	B+
2024	2	20242222	전공	C++프로그래밍	3	이영표	A
2024	2	20242222	전공	C프로그래밍	3	이영표	B+
2024	2	20242222	전공	운영체제	3	히딩크	A+
2024	2	20242222	전공	임베디드시스템	3	히딩크	A+
2024	2	20241111	전공	자바프로그래밍	3	이영표	B+
2024	2	20242222	전공	자바프로그래밍	3	히딩크	C+
2024	2	20242222	전공	파이썬프로그래밍	3	이영표	A+

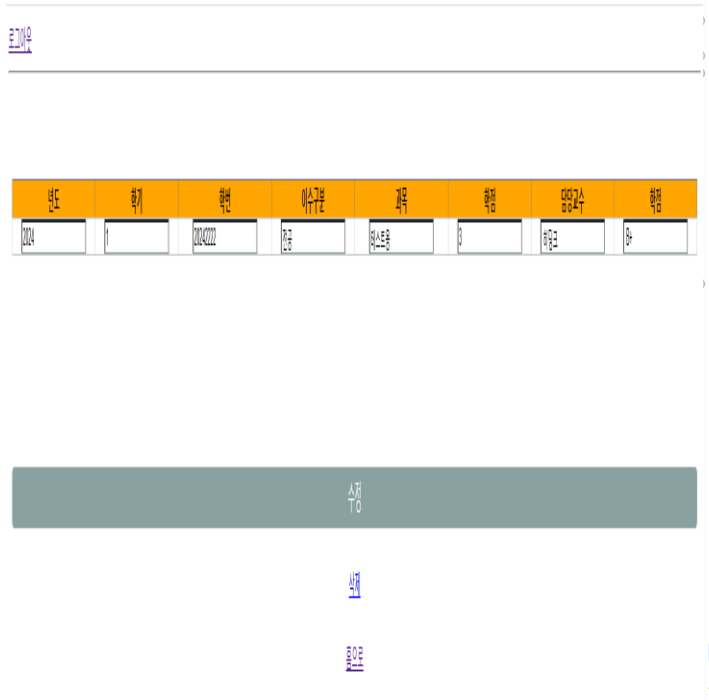
홈으로

getSubjectListController.java

```
<center>
    <h1>전체학생정보</h1>
    <form action ="/getSubjectList.do" method="post" >
        <table border="1" cellpadding="0" cellspacing="0" width="700">
        <tr>
            <td align="right">
            <select name="searchCondition">
                <option value="stdId" <c:if test="${condition == '학번'}"> selected</c:if>>학번

            </select>

                <input name="searchKeyword" type="text" value="${keyword}"/>
                <input type="submit" value="검색"/>
            </td>
        </tr>
        </table>
    </form>



    <table border="1" cellpadding="0" cellspacing="0" width="700">
        <tr>
            <th bgcolor="orange" width="100">년도</th>
            <th bgcolor="orange" width="100">학기</th>
            <th bgcolor="orange" width="200">학번</th>
            <th bgcolor="orange" width="150">이수구분</th>
            <th bgcolor="orange" width="200">과목</th>
            <th bgcolor="orange" width="100">학점</th>
            <th bgcolor="orange" width="100">담당교수</th>
            <th bgcolor="orange" width="100">학점</th>
        </tr>

    <c:forEach var="list" items="${subjectList}">
       <tr>
            <td>${list.year}</td> <!-- 년도 -->
            <td> ${list.semester} </td> <!-- 학기 -->
            <td> <a href="getSubject.do?id=${list.id}&subject_name=${list.subjectName}"> ${list.id} </td> </a> <!-- 학번 -->
            <td>${list.course}</td> <!-- 이수구분 -->
            <td>${list.subjectName}</td> <!-- 과목 -->
            <td>${list.credit}</td> <!-- 학점 -->
            <td>${list.professor}</td> <!-- 담당교수 -->
            <td>${list.score}</td> <!-- 학점 -->
       </tr>
    </c:forEach>

    </table>

    <br>

    <a href="adminView.do"> 홈으로 </a>

</center>
```

- **전체학생성적조회**

- **관리자 페이지에 있는 회원성적조회 이미지 -> (getSubjectList.do 요청이) --> dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → getSubjectListController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**



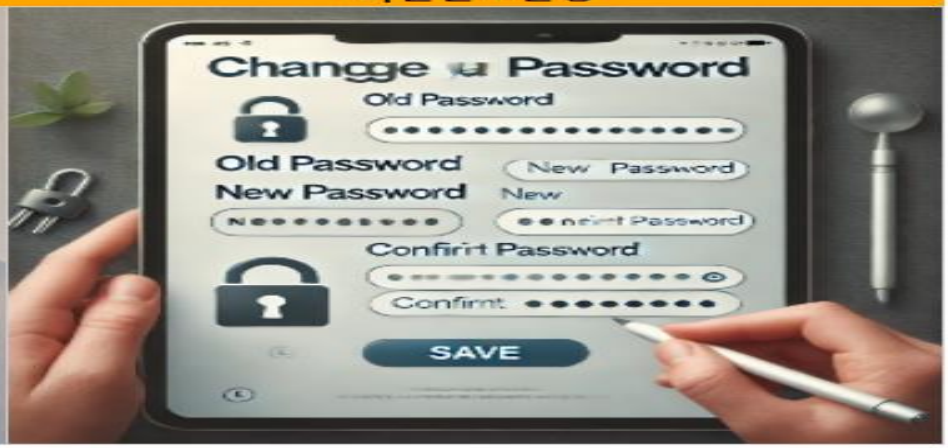

```
<form action="updateSubject.do?stdid=${subject.id} " method="post">

    <table border="1" cellpadding="0" cellspacing="0" width="700">
            <tr>
                <th bgcolor="orange" width="100">년도</th>
                <th bgcolor="orange" width="100">학기</th>
                <th bgcolor="orange" width="200">학번</th>
                <th bgcolor="orange" width="150">이수구분</th>
                <th bgcolor="orange" width="200">과목</th>
                <th bgcolor="orange" width="100">학점</th>
                <th bgcolor="orange" width="100">담당교수</th>
                <th bgcolor="orange" width="100">학점</th>
            </tr>
<!-- 리스트 불러오는 도중에 에러가 생김 확인 해야할듯 -->
            <!-- list 명 수정하고 getUser컨트롤러 request.setattribute 해야함. -->
            <tr>
                <td> <input type="text" name="year" value="${subject.year}" > </input> </td> <!-- 년도 -->
                <td> <input type="text" name="semester" value="${subject.semester}" > </input> </td> <!-- 학기 -->
                <td> <input type="text" name="id" value="${subject.id}" >  </td> </a> <!-- 학번 -->
                <td> <input type="text" name="course" value="${subject.course}" > </td> <!-- 이수구분 -->
                <td> <input type="text" name="subjectName" value="${subject.subjectName}" > </td> <!-- 과목 -->
                <td>  <input type="text" name="credit" value="${subject.credit}" > </td> <!-- 학점 -->
                <td>  <input type="text" name="professor" value=" ${subject.professor}" ></td> <!-- 담당교수 -->
                <td> <input type="text" name="score" value=" ${subject.score}" > </td> <!-- 학점 -->
            </tr>
    </table>

    <br>
    <input type="submit" value="수정"> </input>
    <br>
    <br>
    <a href="deleteSubject.do?stdid=${subject.id}&subjectName=${subject.subjectName} ">삭제</a>


    <br>
    <br>
    <a href="adminView.do"> 홈으로 </a>
</form>
</center>
```

getSubjectController.java
deleteSubjectController.java
updateSubjectController.java

- **학생성적조회/수정/삭제**
- **전체회원정보 페이지에서 학번을 누르면 --> (getSubject.do 요청) --> dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → getSubjectController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**
- **정보를 수정하고 수정 버튼을 누르면 -> (*updateSubject.do?stdid*=${subject.id} 요청(학번이 요청과 같이 넘어감)) --> dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → updateSubjectController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**
- **삭제를 누르면 → *deleteSubject.do?stdid*=${subject.id}*&subjectName*=${subject.subjectName} → dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → deleteSubjectController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**

로그아웃

이영표 교수님 환영합니다.

학생성적조회	비밀번호변경
	

- 교수

- 학생성적조회/성적수정
- 비밀번호 변경

로그아웃

전체학생정보

학번 검색

년도	학기	학번	이수구분	과목	학점	담당교수	학점
2023	1	20191111	전공	자바프로그래밍	3	이영표	B+
2023	1	20241111	전공	테스트용	3	이영표	A
2024	2	20242222	전공	C++프로그래밍	3	이영표	A
2024	2	20242222	전공	C프로그래밍	3	이영표	B+
2024	2	20241111	전공	자바프로그래밍	3	이영표	B+
2024	2	20242222	전공	파이썬프로그래밍	3	이영표	A+

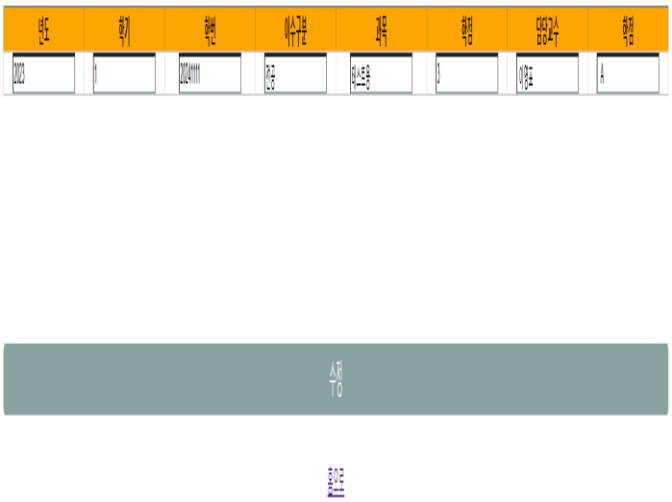
홈으로

getProfStdSubjectList.jsp

```
<h1>전체학생정보</h1>

<form action ="/getSubjectList.do" method="post" >
    <table border="1" cellpadding="0" cellspacing="0" width="700">
    <tr>
        <td align="right">
        <select name="searchCondition">
            <option value="stdId" <c:if test="${condition == '학번'}"> selected</c:if>>학번

        </select>

            <input name="searchKeyword" type="text" value="${keyword}"/>
            <input type="submit" value="검색"/>
        </td>
    </tr>
    </table>
</form>


<table border="1" cellpadding="0" cellspacing="0" width="700">
    <tr>
        <th bgcolor="orange" width="100">년도</th>
        <th bgcolor="orange" width="100">학기</th>
        <th bgcolor="orange" width="200">학번</th>
        <th bgcolor="orange" width="150">이수구분</th>
        <th bgcolor="orange" width="200">과목</th>
        <th bgcolor="orange" width="100">학점</th>
        <th bgcolor="orange" width="100">담당교수</th>
        <th bgcolor="orange" width="100">학점</th>
    </tr>


<c:forEach var="list" items="${subjectList}">
    <!-- fn:trim 으로 공백제거 하고 문자열 비교해야함. -->
    <c:if test="${fn:trim(list.professor) eq fn:trim(sessionScope.user.name)}">
       <tr>
            <td>${list.year}</td> <!-- 년도 -->
            <td> ${list.semester} </td> <!-- 학기 -->
            <td> <a href="getProfStd.do?id=${list.id}&subject_name=${list.subjectName}"> ${list.id} </td> </a> <!-- 학번 -->
            <td>${list.course}</td> <!-- 이수구분 -->
            <td>${list.subjectName}</td> <!-- 과목 -->
            <td>${list.credit}</td> <!-- 학점 -->
            <td>${list.professor}</td> <!-- 담당교수 -->
            <td>${list.score}</td> <!-- 학점 -->
       </tr>
    </c:if>
</c:forEach>

</table>

<br>

<a href="profView.do"> 홈으로 </a>
```

- **전체학생성적조회**

- **교수 페이지에 있는 학생성적조회 이미지 -> (getProfStdList.do 요청이) --> dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → getProfStdListController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**




```
<center>
    <br>

    <form action="updateProfStdScore.do?stdid=${subject.id}" method="post">

        <table border="1" cellpadding="0" cellspacing="0" width="700">
                <tr>
                    <th bgcolor="orange" width="100">년도</th>
                    <th bgcolor="orange" width="100">학기</th>
                    <th bgcolor="orange" width="200">학번</th>
                    <th bgcolor="orange" width="150">이수구분</th>
                    <th bgcolor="orange" width="200">과목</th>
                    <th bgcolor="orange" width="100">학점</th>
                    <th bgcolor="orange" width="100">담당교수</th>
                    <th bgcolor="orange" width="100">학점</th>
                </tr>
        <!-- 리스트 돌리므는 도중에 에러가 생김 확인 해야함ㅊ -->
                <!-- list 명 수정하고 getUser컨트롤러 request.setattribute 해야함. -->
                <tr>
                    <td> <input type="text" name="year" value="${subject.year}" readonly> </input> </td> <!-- 년도 -->
                    <td> <input type="text" name="semester" value="${subject.semester}" readonly> </input> </td> <!-- 학기 -->
                    <td> <input type="text" name="id" value="${subject.id}" readonly>  </td> </a> <!-- 학번 -->
                    <td> <input type="text" name="course" value="${subject.course}" readonly > </td> <!-- 이수구분 -->
                    <td> <input type="text" name="subjectName" value="${subject.subjectName}" readonly > </td> <!-- 과목 -->
                    <td>  <input type="text" name="credit" value="${subject.credit}" readonly > </td> <!-- 학점 -->
                    <td>  <input type="text" name="professor" value=" ${subject.professor}" readonly></td> <!-- 담당교수 -->
                    <td> <input type="text" name="score" value=" ${subject.score}" > </td> <!-- 학점 -->
                </tr>
        </table>

        <br>
        <input type="submit" value="수정"> </input>

        <br>
        <br>
        <a href="adminView.do"> 홈으로 </a>
    </form>
</center>
```

getProfStdController.java

updateProfStdScoreController.java

- **학생성적조회/수정/삭제**

- **전체 학생 성적 페이지에서 학번을 누르면 --> (getProfStd.do 요청) --> dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → getProfStdController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**

- **정보를 수정하고 수정 버튼을 누르면 -> (*updateProfStdScore.do?stdid=*${subject.id} 요청(학번이 요청과 같이 넘어감)) --> dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → updateProfStdScoreController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**

로그아웃

박주영 학생 환영합니다.

수강내역조회

비밀번호변경

- 학생

- 수강내역조회
- 비밀번호 변경

로그아웃

수강내역

년도	학기	학번	이수구분	과목	학점	담당교수	학점
2024	2	20242222	전공	자바프로그래밍	3	히딩크	C+
2024	2	20242222	전공	운영체제	3	히딩크	A+
2024	2	20242222	전공	C프로그래밍	3	이영표	B+
2024	2	20242222	전공	임베디드시스템	3	히딩크	A+
2024	2	20242222	전공	C++프로그래밍	3	이영표	A
2024	2	20242222	전공	파이썬프로그래밍	3	이영표	A+
2024	1	20242222	전공	테스트용	3	히딩크	B+

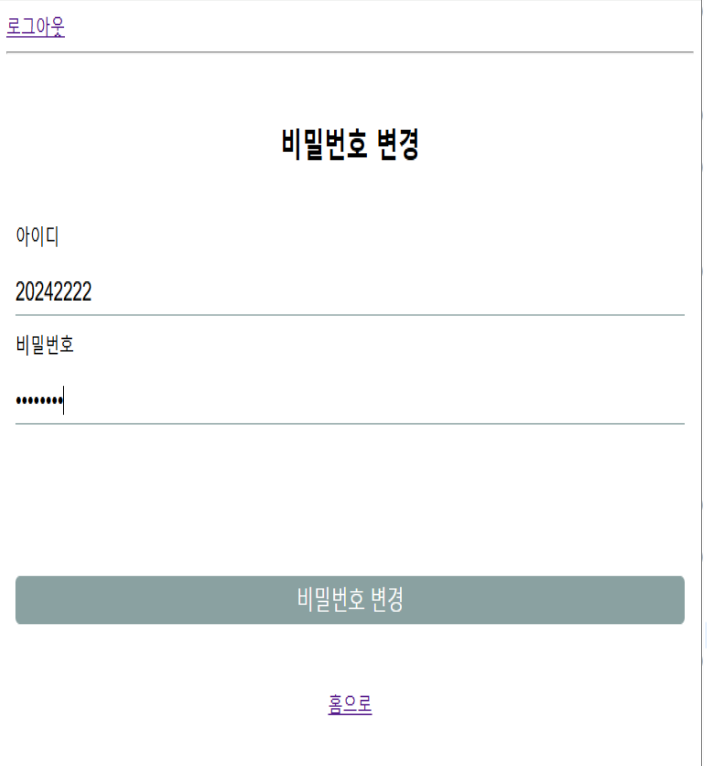
홈으로

getStudentListController.java

```
<center>
        <h1>수강내역</h1>



        <table border="1" cellpadding="0" cellspacing="0" width="700">
            <tr>
                <th bgcolor="orange" width="100">년도</th>
                <th bgcolor="orange" width="100">학기</th>
                <th bgcolor="orange" width="200">학번</th>
                <th bgcolor="orange" width="150">이수구분</th>
                <th bgcolor="orange" width="200">과목</th>
                <th bgcolor="orange" width="100">학점</th>
                <th bgcolor="orange" width="100">담당교수</th>
                <th bgcolor="orange" width="100">학점</th>
            </tr>


<!-- 리스트 불러오는 도중에 에러가 생김 확인 해야함로 -->
        <c:forEach var="list" items="${subject}">
            <tr>
                <td>${list.year}</td> <!-- 년도 -->
                <td> ${list.semester} </td> <!-- 학기 -->
                <td> <a href="getProfStd.do?id=${list.id}&subject_name=${list.subjectName}"> ${list.id} </td> </a> <!-- 학번 -->
                <td>${list.course}</td> <!-- 이수구분 -->
                <td>${list.subjectName}</td> <!-- 과목 -->
                <td>${list.credit}</td> <!-- 학점 -->
                <td>${list.professor}</td> <!-- 담당교수 -->
                <td>${list.score}</td> <!-- 학점 -->
            </tr>



        </c:forEach>

        </table>

        <br>

        <a href="stdView.do"> 홈으로 </a>

    </center>
```

- **수강내역조회**

- **학생 페이지에 있는 수강내역조회 이미지 -> (getStudentList.do 요청이) --> dispatcherServlet --> HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → getStudentListController 에서 작업 후 다음으로 넘어갈 view 이름 반환 --> dispatcherServlet 에서 확인 후 이동**




```
<header>
        <h2>비밀번호 변경</h2>
    </header>

    <form action="changePassword.do" method="POST">

        <div class="input-box">
            아이디 <input id="username" type="text" name="id" placeholder="아이디" value="${sessionScope.user.id}" readonly>
        </div>

        <div class="input-box">
            비밀번호 <input id="password" type="password" name="password" placeholder="비밀번호" value="${sessionScope.user.password}">
        </div>

        <input type="submit" value="비밀번호 변경">

    </form>

    <br>
    <center>

        <c:if test="${fn:trim(sessionScope.user.role) eq fn:trim('STD')}">
            <a href="stdView.do"> 홈으로 </a>
        </c:if>

        <c:if test="${fn:trim(user.role) eq fn:trim('PROF')}">
            <a href="profView.do"> 홈으로 </a>
        </c:if>
    </center>
```

changePasswordController.java

changePasswordViewController.java

- **교수/학생 공통**
- **비밀번호 변경**
- **교수/학생 페이지에서 비밀번호 변경 이미지 → changePasswordView.do → dispatcherServlet → HandlerMapping 클래스에서 해당 요청과 매핑되어 있는 클래스를 찾음 → changePasswordViewController 클래스에서 changePassword.jsp 로 이동 처리 →**
- **원하는 비밀번호 입력 후 비밀번호 변경을 누르게 되면 → 위와 동일하게 changePassword.do 이 dispatcherServlet에 처음으로 가게 되고 HandlerMapping에서 동일하게 요청과 매핑되어 있는 클래스를 찾아냄 → changePasswordController 에서 작업 후 → dispatcherServlet 을 통해 다음 페이지로 이동**

- # 4. 예외처리

```java
public static Connection getConnection() {
    Connection conn = null;


    try {
        //1단계 : 객체로딩
        DriverManager.registerDriver(new org.h2.Driver());

        //2단계 : 연결
        String jdbcUrl = "jdbc:h2:tcp://localhost/~/test";
        conn = DriverManager.getConnection(jdbcUrl, "sa", "");

    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

    return conn;
}
```

- 데이터베이스 연결에 실패할 때의 문제를 대비해서 try-catch 문에 감싸 코드를 작성했습니다.
- 만약 문제가 발생할 시에는 catch문에 있는 e.printStackTrace() 를 통해 오류를 출력

```
// 수강가능 학점 18점에 대한 예외처리를 위해서도 사용함.
private static String GET_STDIDSUBJECT2 ="select * from subject where id=? and UV_YEAR=? and SEMESTER=? ";

   //관리자
   //총 수강 가능한 학점 18점에 대한 예외처리를 위한 메소드
   public int getCountStdCredit(SubjectVO vo) {
       // TODO Auto-generated method stub
       int count = 0;
       try {
           conn = DatabaseConnector.getConnection();
           stmt = conn.prepareStatement(GET_STDIDSUBJECT2);

           stmt.setString(1, vo.getId());
           stmt.setString(2, vo.getYear());
           stmt.setString(3, vo.getSemester());

           rs = stmt.executeQuery();



           while(rs.next()) {
               String credit = rs.getString("credit");
               count +=  Integer.parseInt(credit.trim());
           }


       } catch (SQLException e) {
           // TODO Auto-generated catch block
           DatabaseConnector.close(rs, stmt, conn);
           e.printStackTrace();
       }

       return count;

   }
```

```
//관리자 학생 성적 검색 /중복된 수강신청에 대한 예외처리에도 사용됨.
private static String GET_STDIDSUBJECT ="select * from subject where id=?";


    //관리자 전용 추가하려는 수업이 db에 있는 지 확인하기 위한 메소드
    public int getSubjectName(SubjectVO vo) {
        // TODO Auto-generated method stub

        int flag=-1;
        try {
            conn = DatabaseConnector.getConnection();
            stmt = conn.prepareStatement(GET_STDIDSUBJECT);

            stmt.setString(1, vo.getId());

            rs = stmt.executeQuery();

            while(rs.next()) {
                if(vo.getSubjectName().equals(rs.getString("SUBJECT_NAME")) ) {
                    flag = 1; //추가하려는 과목은 기존에 저장되어 있는 과목임.
                }
            }


        } catch (SQLException e) {
            // TODO Auto-generated catch block
            DatabaseConnector.close(rs, stmt, conn);
            e.printStackTrace();
        }

        return flag;
    }
```

```java
        //insertSubjectController 클래스
        //jsp에서 입력한 정보 추출
        String id = request.getParameter("id");
        String year = request.getParameter("year");
        String semester = request.getParameter("semester");
        String course = request.getParameter("course");
        String subjectName = request.getParameter("subject_name");
        String credit = request.getParameter("credit");
        String professor = request.getParameter("professor");
        String score = request.getParameter("score");

        vo.setId(id);
        vo.setYear(year);
        vo.setSemester(semester);
        vo.setCourse(course);
        vo.setSubjectName(subjectName);
        vo.setCredit(credit);
        vo.setProfessor(professor);
        vo.setScore(score);

        SubjectVO vo2 = new SubjectVO();
        vo2.setId(id);
        vo2.setSubjectName(subjectName);
        vo2.setSemester(semester);
        vo2.setYear(year);

        int num = dao.getCountStdCredit(vo2); //추가하려는 학번의 학생이 수강하고 있는 학점
        int flag = dao.getSubjectName(vo2); //중복된 수업추가에 대한 예외처리 메소드

        String num2 = credit.trim();

        int num3 = Integer.parseInt(num2);


        if((num + num3) <= 18 && flag == -1) {  //수강학점을 초과하지 않고 수업이 중복되지 않을 때
            dao.insertSubject(vo);
            System.out.println("성적정보 입력 성공");
            return "admin";
        }

        else if((num + num3) > 18){ //수강학점을 초과할 때
            request.setAttribute("insertError", vo.getId() +"학생의 수강학점이 18점을 초과했습니다.");
            return "insertSubjectView.do";
        }

        else { //중복될 때
            request.setAttribute("insertError", vo.getSubjectName() + "은 기존에 수강했거나 수강중인 수업입니다.");
            return "insertSubjectView.do";
        }
    }

}
```

로그아웃

학생성적정보입력

20242222학생의 수강학점이 18점을 초과했습니다.

아이디

년도

학기

이수구분

과목

학점

담당교수

성적

로그아웃

학생성적정보입력

자바프로그래밍은 기존에 수강했거나 수강중인 수업입니다.

아이디

년도

학기

이수구분

과목

학점

담당교수

성적

```
<form action="insertSubject.do" method="POST">
    <h1>학생성적정보입력</h1>
    <br><br>
    <c:if test="${insertError != null}">
    <h3 style="color: red;"> ${insertError}</h3>
    <c:remove var ="insertError" scope="request"></c:remove>
    </c:if>
```

- 수강했던 과목 추가/최대 수강학점 18점 초과 시 예외처리

- 수강했던 과목 추가 시 예외처리
- getSubjectName 메소드를 통해 DB 에서 해당 학번의 학생이 추가하려는 과목을 수강했는 지 조회 (전에 수강했던 과목이면 flag =1 설정 후 반환) → insertError 라는 이름으로 요청범위 내에 데이터를 등록 → 위의 가운데 이미지와 같이 경고문 출력

- 최대 수강학점 초과 시 예외처리
- getCountStdCredit 메소드를 통해 수업을 추가하려고 하는 현재 년도 학기를 기준으로 추가된 수업의 학점을 계산하여 반환 → 추가하려고 하는 학점과 현재 수강신청되어 있는 학점을 계산 → 계산된 값이 18점보다 크면 insertError 라는 이름으로 요청범위 내에 데이터를 등록 → 왼쪽 상단의 이미지와 같이 경고문 출력

- (경고문 출력 후 다른 페이지 이동 후 다시 해당 페이지로 이동 시 경고문은 삭제되어 있음)