

1. 프로젝트 개요

본 프로젝트는 “한성대학교 종합정보시스템” 기능 중 몇 가지 기능을 Spring Framework를 통해 구현한다. 웹페이지는 Spring MVC 패턴을 사용하였고, 데이터베이스의 연동, Spring web Form, Data Binding, Data Validation, Data Buffering 그리고 Spring Security의 기능을 사용하여 구현하도록 했다.

2. 프로젝트 구조.

프로젝트의 구조는 다음과 같다.

- finalproject
 - src/main/java
 - kr.ac.hansung.controller : Controller
 - CourseController.java
 - HomeController.java
 - LoginController.java
 - MenuController.java
 - kr.ac.hansung.dao : dao
 - CourseDao.java
 - kr.ac.hansung.filter : filter
 - TestFilter.java
 - kr.ac.hansung.model : Model
 - Course.java
 - kr.ac.hansung.service : Service
 - CourseService.java
 - src/main/resources
 - src/test/java
 - src/test/resources
 - Maven Dependencies
 - JRE System Library [JavaSE-11]
 - src
 - main
 - webapp
 - resources : resource 파일
 - css
 - main.css
 - img
 - course.png
 - WEB-INF
 - classes
 - props
 - jdbc.properties
 - spring
 - appServlet : context 파일
 - dao-context.xml
 - security-context.xml
 - service-context.xml
 - servlet-context.xml
 - views : View
 - courses.jsp
 - home.jsp
 - login.jsp
 - menu.jsp
 - semester.jsp
 - sign.jsp
 - signLookup.jsp
 - signSuccess.jsp
 - web.xml
 - test
 - target
 - pom.xml

3. 단계별 수행 과정.

* 인증 관련 파일

security-context.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/security"
  xmlns:beans="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.springframework.org/schema/beans
    https://www.springframework.org/schema/beans/spring-beans-3.0.xsd
    http://www.springframework.org/schema/security
    https://www.springframework.org/schema/security/spring-security.xsd">

  <authentication-manager>
    <authentication-provider>
      <jdbc-user-service data-source-ref="dataSource"
        users-by-username-query="select username,password, enabled from users where username = ?"
        authorities-by-username-query="select username, authority from authorities where username =?" />
    </authentication-provider>
  </authentication-manager>

  <http auto-config="true" use-expressions="true">
    <intercept-url pattern="/"
      access="permitAll" />
    <intercept-url pattern="/menu"
      access="isAuthenticated()" />
    <intercept-url pattern="/courses"
      access="isAuthenticated()" />
    <intercept-url pattern="/semester"
      access="isAuthenticated()" />
    <intercept-url pattern="/sign"
      access="isAuthenticated()" />
    <intercept-url pattern="/signSuccess"
      access="isAuthenticated()" />
    <intercept-url pattern="/signLookup"
      access="isAuthenticated()" />

    <form-login login-page="/Login"
      authentication-failure-url="/Login?error" />
    <logout />
  </http>
</beans:beans>
```

1) 첫페이지 home.jsp

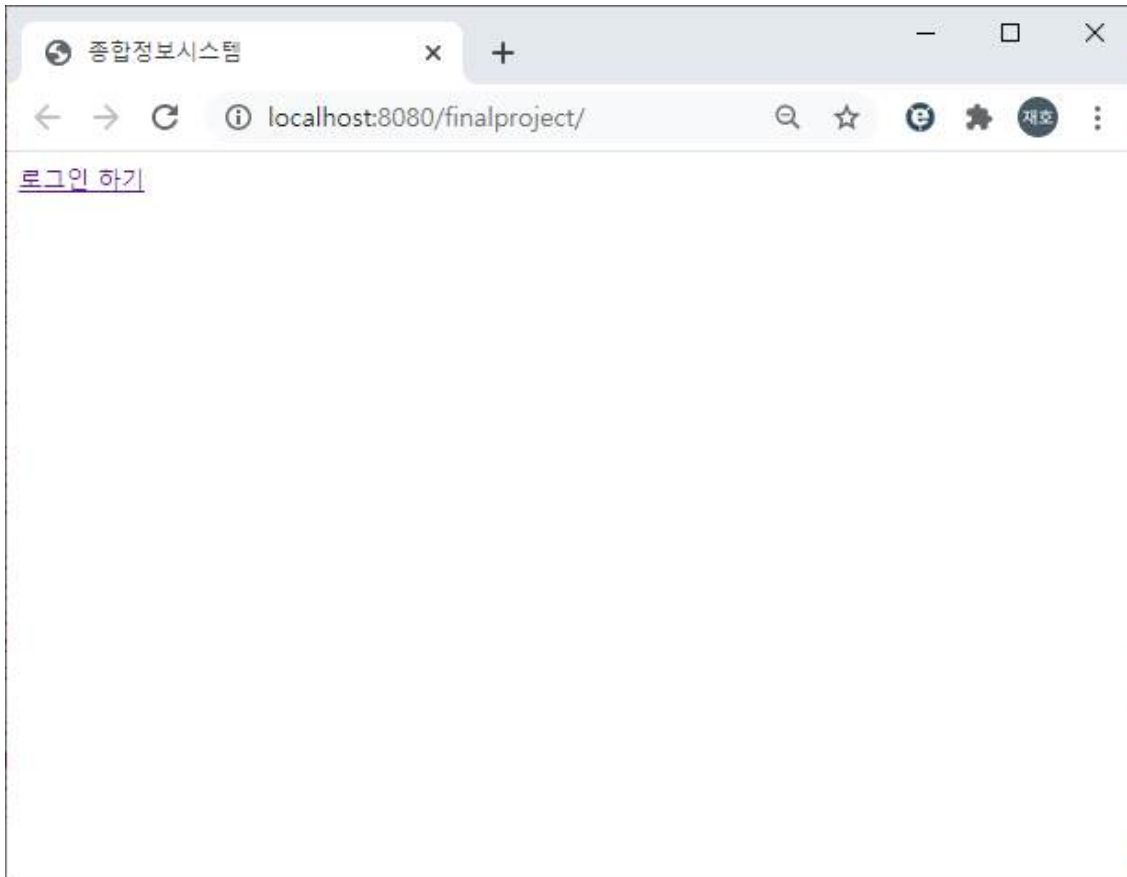
URL: <http://localhost:8080/finalproject/>

* Controller 매핑 정보

```
1 package kr.ac.hansung.controller;
2
3 import org.springframework.stereotype.Controller;
4
5
6
7 @Controller
8 public class HomeController {
9
10     @RequestMapping(value = "/", method = RequestMethod.GET)
11     public String home() {
12         return "home";
13     }
14 }
15
```

* `<intercept-url pattern="/" access="permitAll" />`

* 페이지 화면



여기서 로그인 하기 버튼을 누르면 다음 url로 매핑이된다.

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>종합정보시스템</title>
</head>
<body>
    <a href="{pageContext.request.contextPath}/menu">로그인 하기</a>
</body>
</html>
```

2) 두 번째 페이지 : login.jsp

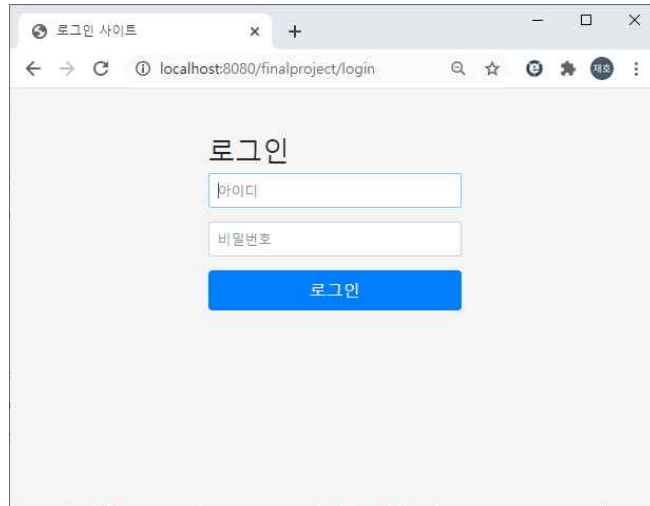
URL : <http://localhost:8080/finalproject/login>

* 관련 소스

```
<form-login login-page="/login" authentication-failure-url="/login?error" /> <logout />
```

```
* <intercept-url pattern="/" access="permitAll" />
```

* 페이지 화면



소스

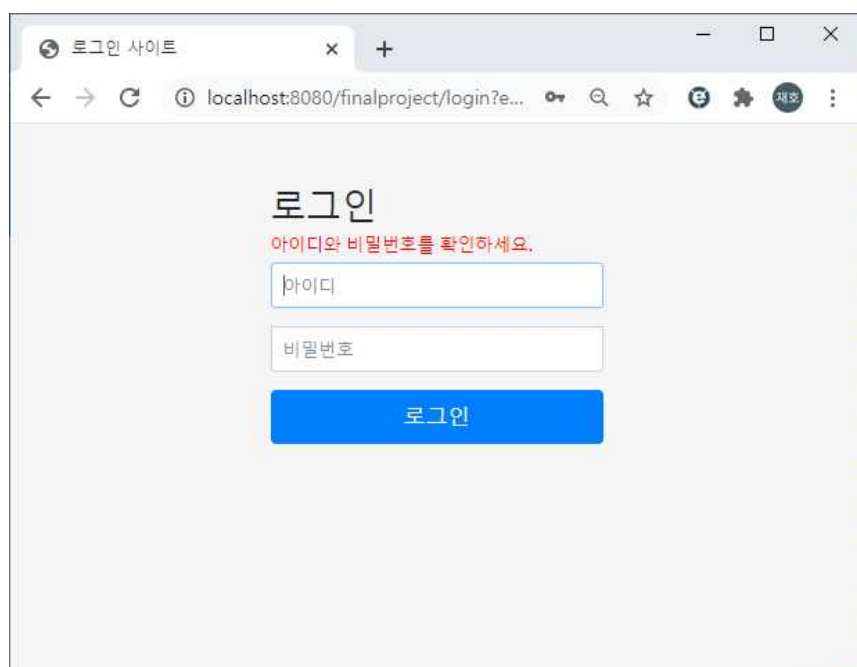
```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<!DOCTYPE html>
<html><head>
<meta charset="utf-8"><meta name="viewport"
    content="width=device-width, initial-scale=1, shrink-to-fit=no">
<meta name="description" content=""><meta name="author" content="">
<title>로그인 사이트</title>
<link
    href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta/css/bootstrap.min.css"
    rel="stylesheet"
    integrity="sha384-/Y6pD6FV/Vv2HJnA6t+vsIU6fwYXjCFtcEpHbNJ0lyAFsXTsjBbfaDjzALeQsN6M"
    crossorigin="anonymous">
<link
    href="https://getbootstrap.com/docs/4.0/examples/signin/signin.css"
    rel="stylesheet" crossorigin="anonymous" />
</head>
<body>
<div class="container">
<form class="form-signin" method="post" action="<c:url value="login"/>">
<h2 class="form-signin-heading">로그인</h2>
<c:if test="${not empty errorMsg}">
<div style="color:red"><h6>${errorMsg}</h6></div>
</c:if>
<c:if test="${not empty logoutMsg}">
<div style="color:blue"><h6>${logoutMsg}</h6></div>
</c:if>
<p>
<label for="username" class="sr-only">아이디 </label> <input
    type="text" id="username" name="username" class="form-control"
    placeholder="아이디" required autofocus>
</p>
<p>
<label for="password" class="sr-only">비밀번호</label> <input
    type="password" id="password" name="password" class="form-control"
    placeholder="비밀번호" required>
</p>
<input type="hidden" name="${_csrf.parameterName}"
    value="${_csrf.token}" />
<button class="btn btn-lg btn-primary btn-block" type="submit">로그인</button>
</form>
</div>
</body></html>
```

로그인 페이지에서는 Spring security를 활용하여 작성하였으며, 아이디를 제대로 출력하지 않았을 경우에 다시 작성하도록 오류 메시지를 나오게 하였다.
또한 로그아웃을 기능을 만들어서, 로그아웃을 할 경우 다시 login페이지로 매핑되도록 구현하였다.

- Logincontroller.java

```
@Controller
public class LoginController {
    @RequestMapping("/login")
    public String showLogin(@RequestParam (value="error", required=false) String error,
        @RequestParam (value="logout", required=false) String logout,
        Model model) {
        if(error!= null) {
            model.addAttribute("errorMsg","아이디와 비밀번호를 확인하세요.");
        }
        if(logout!= null) {
            model.addAttribute("logoutMsg","로그아웃 되었습니다.");
        }
        return "login";
    }
}
```

2-1) 로그인이 실패한 경우 페이지



로그인이 성공한 경우에는 다음 페이지로 이동한다.

로그인을 할 수 있는 멤버의 경우는 DB에서 조회하여 Spring에서 인증을 해준다. DB는 users 테이블과 authorities 테이블을 이용하였다. Spring security 사용하였다.

관련 DB 테이블에 대해서 보자면 이렇다.

	username	password	enabled
	choi	{noop}1234	1
▶*	NULL	NULL	NULL

	username	authority
▶	choi	ROLE_ADMIN
*	NULL	NULL

```
<authentication-manager>
  <authentication-provider>
    <jdbc-user-service data-source-ref="dataSource"
      users-by-username-query="select username,password, enabled from users where username = ?"
      authorities-by-username-query="select username, authority from authorities where username =?" />
  </authentication-provider>
</authentication-manager>
```

3) 세 번째 페이지(login성공 이후) : menu.jsp

URL :http://localhost:8080/finalproject/menu

- MenuController.java

인증정보 :

```
<intercept-url pattern="/menu" access="isAuthenticated()" />
```

:인증을 받은 사용자만이 접근이 가능하다.

@Controller

```
public class MenuController {
```

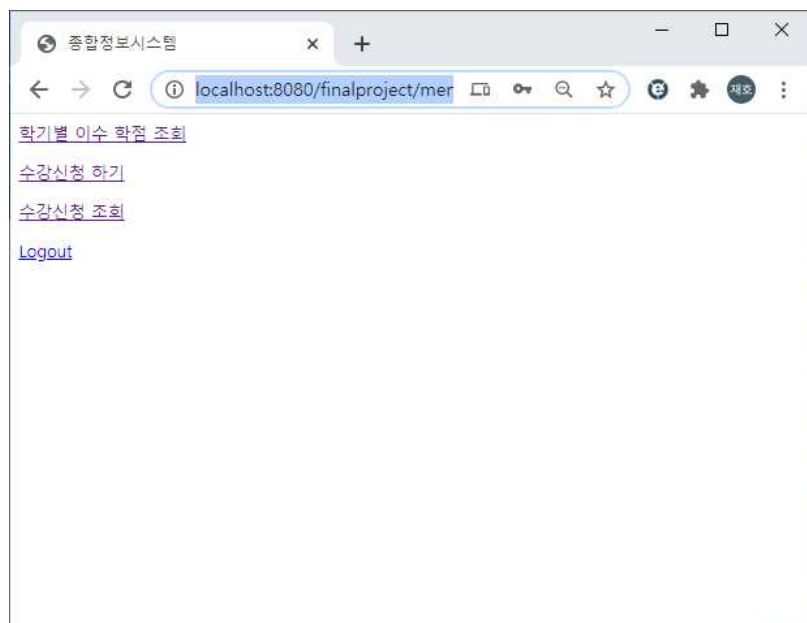
```
    @RequestMapping(value = "/menu", method = RequestMethod.GET)
```

```
    public String showMenu() {
```

```
        return "menu";
```

```
    }
```

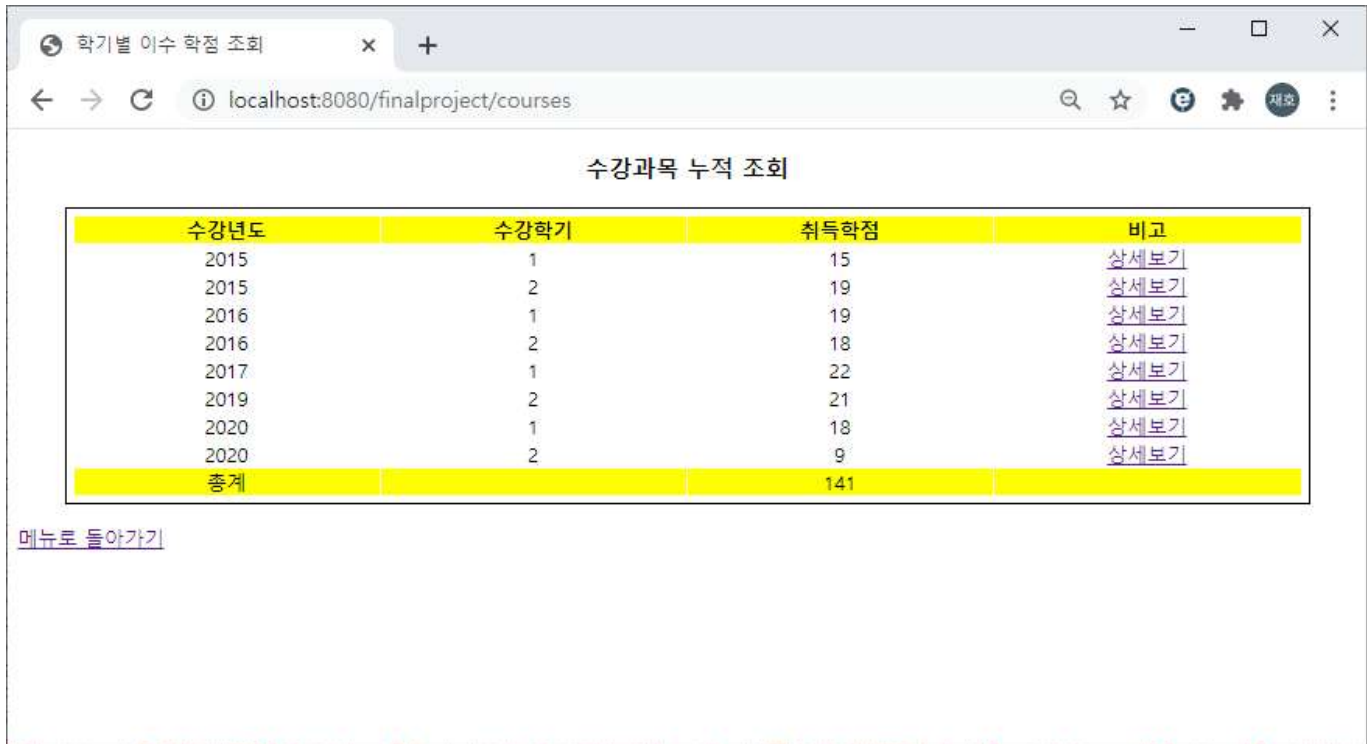
```
}
```



이후 웹페이지는 모두 인증된 사용자가만이 접근이 가능하다.

3_1) 학기별 이수 학점 조회 페이지 : courses.jsp

URL : <http://localhost:8080/finalproject/courses>



학기별 이수 학점 조회 페이지의 경우는 DB에 있는 수강정보를 통해 출력하였다. 수강년도와 수강학기를 group으로 묶어서 출력하도록하였고, 비고란의 상세보기를 누르면 각 수강 학기의 상세 과목에 대한 정보를 볼 수 있다. 또한 취득학점의 총계도 확인할 수 있다.

```
//첫화면, 그룹 단위로 볼수 있는 화면
@RequestMapping("/courses")
public String showSemesterCourses(Model model) {
    List <Course> courses = courseService.getGroupCurrent();
    model.addAttribute("courses", courses);
    return "courses"; //view를 만들어준다.
}
```

(1) Model : Course.java

```
@Getter
@Setter
@NoArgsConstructor
@ToString
public class Course {

    private int year;
    private int semester;

    // size작동함.
    @Size(min = 2, max = 30, message = "필수 항목입니다.")
    private String courseName;

    @Size(min = 2, max = 10, message = "필수 항목입니다.")
    private String classification;

    @Size(min = 2, max = 10, message = "필수 항목입니다. ")
    private String professor;

    @Min(value = 1, message = "최소학점 1학점입니다.")
    @Max(value = 3, message = "최대학점 3학점입니다.")
    private int credit;
}
```

(2) DAO : CourseDao.java

```
@Repository
public class CourseDao {
    private JdbcTemplate jdbcTemplate;
    @Autowired
    public void setDataSource(DataSource dataSource) {
        this.jdbcTemplate = new JdbcTemplate(dataSource);
    }
    public int getRowCount() {
        String sqlStatement = "select count(*) from course";
        return jdbcTemplate.queryForObject(sqlStatement, Integer.class);
    }
    // 연도와 학기에 따른 출력.
    public List<Course> getSemesterCourses(String year, String semester) {
        String sqlStatement = "select * from course WHERE year=" + year+ " and semester =" + semester;

        return jdbcTemplate.query(sqlStatement, new RowMapper<Course>() {

            @Override
            public Course mapRow(ResultSet rs, int rowNum) throws SQLException {

                Course course = new Course();
                course.setYear(rs.getInt("year"));
                course.setSemester(rs.getInt("semester"));
                course.setCourseName(rs.getString("courseName"));
                course.setClassification(rs.getString("classification"));
                course.setProfessor(rs.getString("professor"));
                course.setCredit(rs.getInt("credit"));

                return course;
            }
        });
    }
    //그냥 그룹 단위로 출력해주는 함수
    public List<Course> getGroupCourses() {
        String sqlStatement = "select year, semester, sum(credit) from course group by year, semester order by year";
        return jdbcTemplate.query(sqlStatement, new RowMapper<Course>() {

            @Override
            public Course mapRow(ResultSet rs, int rowNum) throws SQLException {
                Course course = new Course();
                course.setYear(rs.getInt("year"));
                course.setSemester(rs.getInt("semester"));
                course.setCredit(rs.getInt("sum(credit)"));
                return course;
            }
        });
    }
    // 전체 출력해주는 함수.
    public List<Course> getCourses() {
        String sqlStatement = "select * from Course";
        return jdbcTemplate.query(sqlStatement, new RowMapper<Course>() {

            @Override
            public Course mapRow(ResultSet rs, int rowNum) throws SQLException {

                Course course = new Course();
                course.setYear(rs.getInt("year"));
                course.setSemester(rs.getInt("semester"));
                course.setCourseName(rs.getString("courseName"));
                course.setClassification(rs.getString("classification"));
                course.setProfessor(rs.getString("professor"));
                course.setCredit(rs.getInt("credit"));

                return course;
            }
        });
    }
    // 수강 신청 연산자. - 2021년 1월
    // Crud method
    public boolean insert(Course course) {

        int year = course.getYear();
        int semester = course.getSemester();
        String courseName = course.getCourseName();
        String classification = course.getClassification();
        String professor = course.getProfessor();
        int credit = course.getCredit();
        String sqlStatement = "INSERT INTO `course` (`year`, `semester`, `courseName`, `classification`, `professor`, `credit`) "
            + " VALUES (?, ?, ?, ?, ?, ?)";
        return (jdbcTemplate.update(sqlStatement, new Object[] {year, semester, courseName, classification, professor, credit }) == 1);
    }
}
```


(3) Service: CourseService.java

```
@Service
public class CourseService {

    @Autowired
    private CourseDao courseDao;

    public List <Course> getCurrent(){
        return courseDao.getCourses();
    }

    //그룹별로 보는 함수.
    public List<Course> getGroupCurrent() {
        return courseDao.getGroupCourses();
    }

    public List<Course> getSemesterCurrent(String year, String semester) {
        return courseDao.getSemesterCourses(year, semester);
    }

    public void insert(Course course) {
        courseDao.insert(course);
    }
}
```

(4) Controller : CourseController.java

```
@Controller
public class CourseController {

    @Autowired
    private CourseService courseService;

    @RequestMapping("/courses/semester")
    public String showSemesterCourses(@RequestParam String year,
        @RequestParam String semester, Model model) {
        List <Course> courses = courseService.getSemesterCurrent(year, semester);
        model.addAttribute("courses", courses);

        return "semester"; //view를 만들어준다.
    }

    //첫화면, 그룹 단위로 볼수 있는 화면
    @RequestMapping("/courses")
    public String showSemesterCourses(Model model) {
        List <Course> courses = courseService.getGroupCurrent();
        model.addAttribute("courses", courses);
        return "courses"; //view를 만들어준다.
    }

    //2021학년도 1학기 수강 신청조회.
    @RequestMapping("/signLookup")
    public String showSign(@RequestParam String year,
        @RequestParam String semester, Model model) {
        List <Course> courses = courseService.getSemesterCurrent(year, semester);
        model.addAttribute("courses", courses);

        return "signLookup"; //view를 만들어준다.
    }

    //수강신청 웹페이지들어가기
    @RequestMapping("/sign")
    public String sign(Model model) {
        model.addAttribute("course", new Course()); //data buffering
        return "sign"; //view를 만들어준다. 웹폼을 만든다.
    }

    //수강신청 Db에 저장. data validation 필요
    @RequestMapping("/docreate")
    public String docreate(Model model, @Valid Course course, BindingResult result) { //DB에 저장해야함
        if(result.hasErrors()) {
            System.out.println("== Form data does not validated ==");
            List<ObjectError> errors = result.getAllErrors();
            for(ObjectError error:errors) {
                System.out.println(error.getDefaultMessage());
            }
            return "sign"; //다시 입력을 받게 보냄
        }

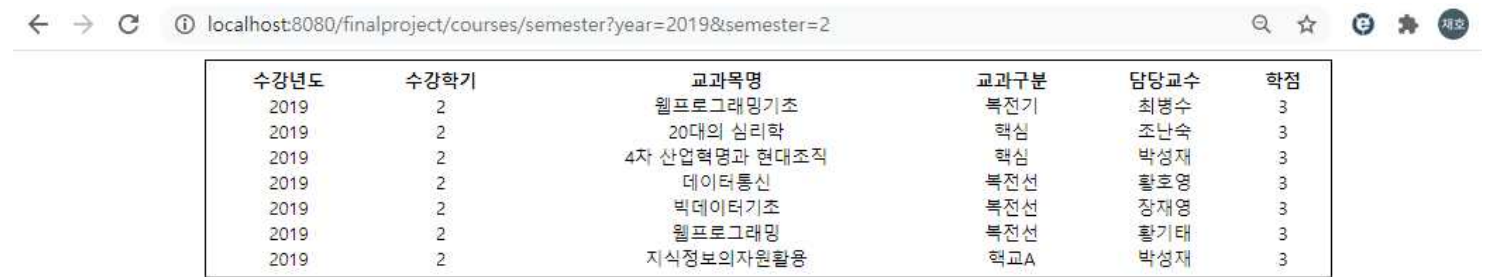
        courseService.insert(course); //수강신청하기.
        return "signSuccess"; //view를 만들어준다. 웹폼을 만든다.
    }
}
```

이렇게 관련된 model, controller, service, dao 가 있다.

3_1_1) 상세과목 조회 페이지 : semester.jsp

URL : 파라미터 값에 따라 다르게 나오도록 구현하였다.

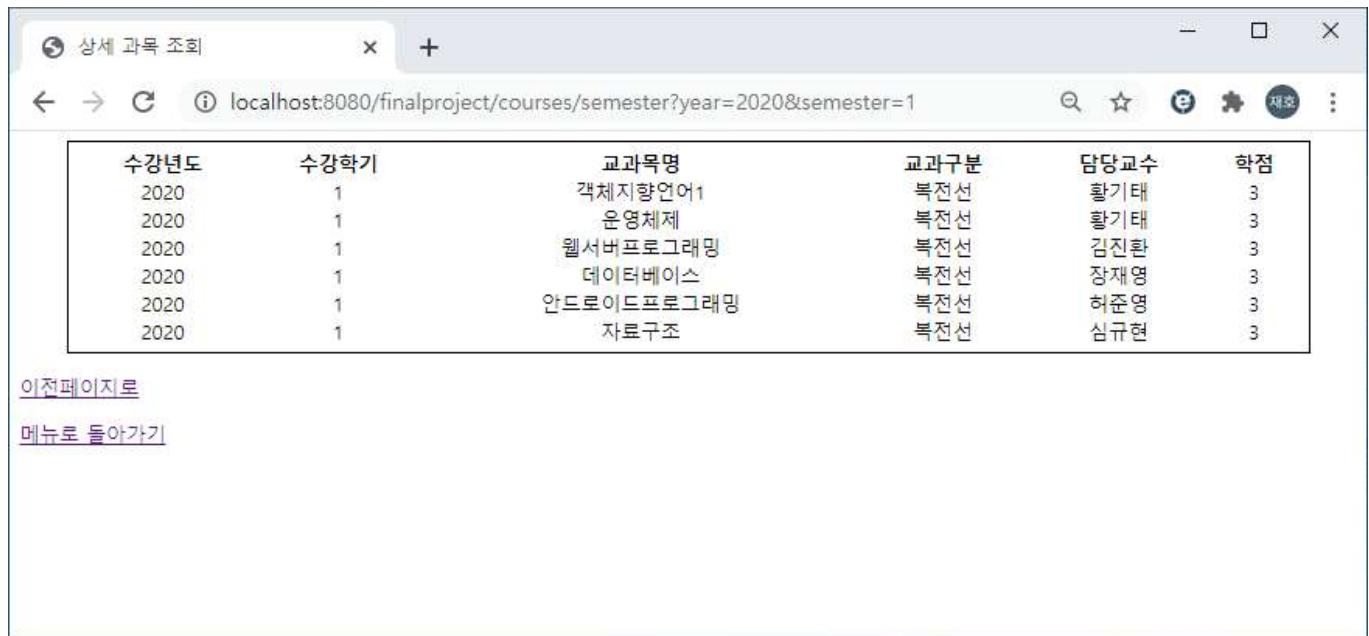
http://localhost:8080/finalproject/courses/semester?year=?&semester=?



수강년도	수강학기	교과목명	교과구분	담당교수	학점
2019	2	웹프로그래밍기초	복전기	최병수	3
2019	2	20대의 심리학	핵심	조난숙	3
2019	2	4차 산업혁명과 현대조직	핵심	박성재	3
2019	2	데이터통신	복전선	황포영	3
2019	2	빅데이터기초	복전선	장재영	3
2019	2	웹프로그래밍	복전선	황기태	3
2019	2	지식정보의자원활용	핵교A	박성재	3

[이전페이지로](#)

[메뉴로 돌아가기](#)



수강년도	수강학기	교과목명	교과구분	담당교수	학점
2020	1	객체지향언어1	복전선	황기태	3
2020	1	운영체제	복전선	황기태	3
2020	1	웹서버프로그래밍	복전선	김진환	3
2020	1	데이터베이스	복전선	장재영	3
2020	1	안드로이드프로그래밍	복전선	허준영	3
2020	1	자료구조	복전선	심규현	3

[이전페이지로](#)

[메뉴로 돌아가기](#)

```
@RequestMapping("/courses/semester")
public String showSemesterCourses(@RequestParam String year,@RequestParam String semester,
Model model) {
    List <Course> courses = courseService.getSemesterCurrent(year, semester);
    model.addAttribute("courses", courses);

    return "semester"; //view를 만들어준다.
}
```

밑에 이전페이지를 누르면 학기별 이수학점 조회 페이지로 돌아가게 되고,

메뉴로 돌아가기를 누르면 menu 페이지로 돌아가게 된다.

3_2) 수강신청 하기 페이지 : sign.jsp

http://localhost:8080/finalproject/sign

- 2021학년도 1학기 수강신청을 할 수 있는 페이지이다.

- 수강신청을 누르면 `action = "${pageContext.request.contextPath}/docreate`을 통해 `docreate`을 `RequestMapping` 된다.

//수강신청 Db에 저장. data validation 필요

```
@RequestMapping("/docreate")
public String docreate(Model model, @Valid Course course, BindingResult result) { //DB에 저장해야함
    if(result.hasErrors()) {
        System.out.println("= Form data does not validated =");
        List<ObjectError> errors = result.getAllErrors();
        for(ObjectError error:errors) {
            System.out.println(error.getDefaultMessage());
        }
        return "sign"; //다시 입력을 받게 보냄
    }

    courseService.insert(course); //수강신청하기.
    return "signSuccess"; //view를 만들어준다. 웹폼을 만든다.
}
```

이 페이지에서 Data Binding , Data Validation, Data Buffering 기술을 사용하였다.

- 아무것도 작성하지 않을 경우에는 data validation을 통해 에러 메시지를 출력하도록 하였다.

- 에러메시지 관련 코드

```
@Getter
@Setter
@NoArgsConstructor
@ToString
public class Course {

    private int year;
    private int semester;

    // size작들함.
    @Size(min = 2, max = 30, message = "필수 항목입니다.")
    private String courseName;

    @Size(min = 2, max = 10, message = "필수 항목입니다.")
    private String classification;

    @Size(min = 2, max = 10, message = "필수 항목입니다. ")
    private String professor;

    @Min(value = 1, message = "최소학점 1학점입니다.")
    @Max(value = 3, message = "최대학점 3학점입니다.")
    private int credit;
}
```

수강신청을 제대로 안한 경우 위와같이 sign으로 매핑되고, 수강신청을 제대로 작성한 경우 signSuccess로 매핑된다.

3_2_1) 수강신청 성공 페이지 : signSuccess.jsp

http://localhost:8080/finalproject/docreate

수강신청 성공페이지이다. 수강신청 계속하기를 누르면 수강신청페이지로 돌아가고, 수강신청 조회하기를 누르면 수강신청 조회페이지로 가게 된다.

2021학년도 1학기 수강신청

순번	과목명	과목구분	학점	학년	교수
1	ios프로그래밍	전선	3	4	이재문
2	SW 통합구현(IPP)	전선	3	전학년	구동영, 이강훈
3	SW설계 및 테스트(IPP)	전선	3	전학년	정인상
4	가상현실 애니메이션 (웹스튜디오자입)	전선	3	전학년	김진모, 정지신
5	객체지향언어1	전선	3	2	계희원
6				2	유상미
7				2	안영아
8				2	황기태
9				2	김진모
10				2	김진모
11				2	유상미
12	데이터마이닝	전선	3	2	황기태
13				3	임종석
14				3	임종석
15				3	임종석
16	데이터베이스	전선	3	3	장재영
17				3	장재영
18				3	김영웅
19				3	김영웅
20				3	김영웅
21	디지털콘텐츠-가상현실 취창업	전선	3	4	이동희
22	디지털콘텐츠기획및제작	전선	3	4	조세훈, 김진모
23				4	조세훈, 김진모
24	모바일 소프트웨어 취창업	전선	3	4	이동희
25	빅데이터 취창업	전선	3	4	이동희
26	빅데이터프로그래밍	전선	3	4	장재영
27				4	장재영
28	사용자인터페이스와 서버구현(IPP)	전선	3	전학년	황기태, 황호영
29	소프트웨어공학	전선	3	3	정인상
30	안드로이드프로그래밍	전선	3	3	허준영
31				3	허준영
32				3	허준영
33	영상처리	전선	3	3	이항찬
34				3	이항찬
35				3	이항찬
36	운영체제	전선	3	3	김진환
37				3	황기태
38				3	김진환
39				3	황기태
40	웹공학 취창업	전선	3	4	이동희
41	웹서버프로그래밍	전선	3	3	김진환
42				3	이석기
43				3	이석기
44	웹프레임워크2	전선	3	4	김남윤

과목명 :

과목구분 :

교수명 :

학점 :

최소학점 1학점입니다.
수강신청

[메뉴로 돌아가기](#)
[수강신청 조회](#)

웹프레임워크2 과목의 수강신청을 성공하였습니다.

[수강신청 계속하기](#)
[수강신청 조회하기](#)

3.3) 수강신청 조회 페이지 : signLookup.jsp

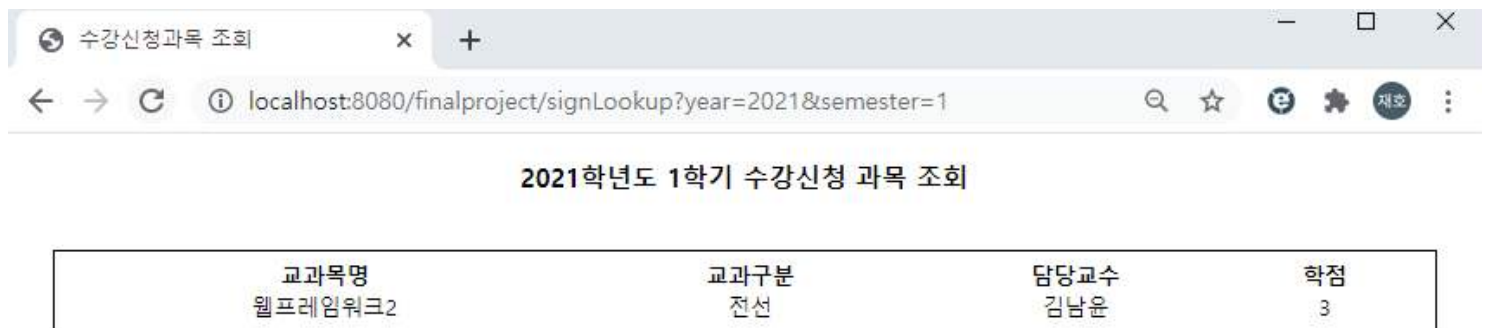
2021학년도 1학기 수강신청을 성공한 과목들을 출력해주는 페이지이다.

수강신청 계속하기를 누르면 수강신청을 할 수 있는 페이지로 돌아간다.

URL : <http://localhost:8080/finalproject/signLookup?year=2021&semester=1>

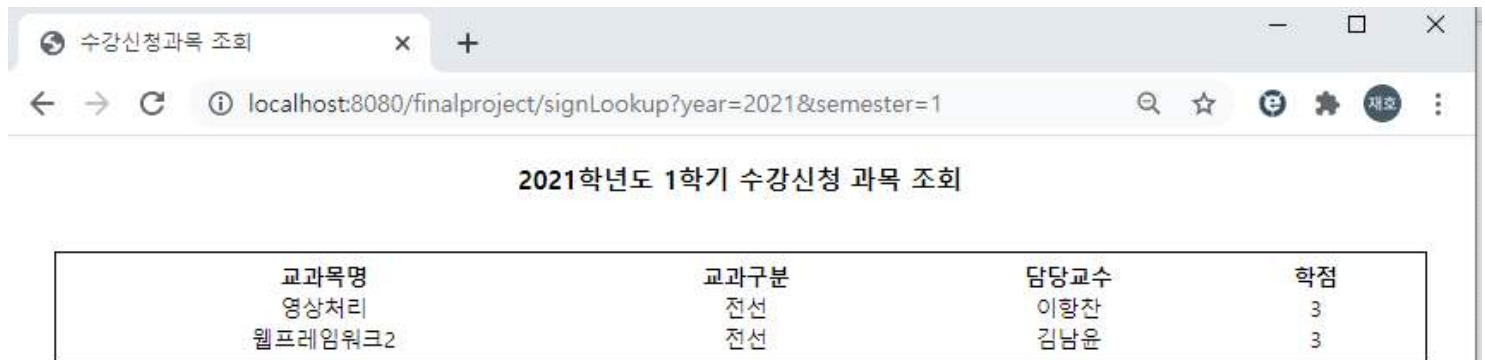
url에 직접 year값과 semester값을 주어서, 나오게 하였다.

```
//2021학년도 1학기 수강 신청조회.  
@RequestMapping("/signLookup")  
public String showSign(@RequestParam String year,  
    @RequestParam String semester, Model model) {  
    List <Course> courses = courseService.getSemesterCurrent(year, semester);  
    model.addAttribute("courses", courses);  
  
    return "signLookup"; //view를 만들어준다.  
}
```



교과목명	교과구분	담당교수	학점
웹프레임워크2	전선	김남운	3

[수강신청 계속하기.](#)



교과목명	교과구분	담당교수	학점
영상처리	전선	이항찬	3
웹프레임워크2	전선	김남운	3

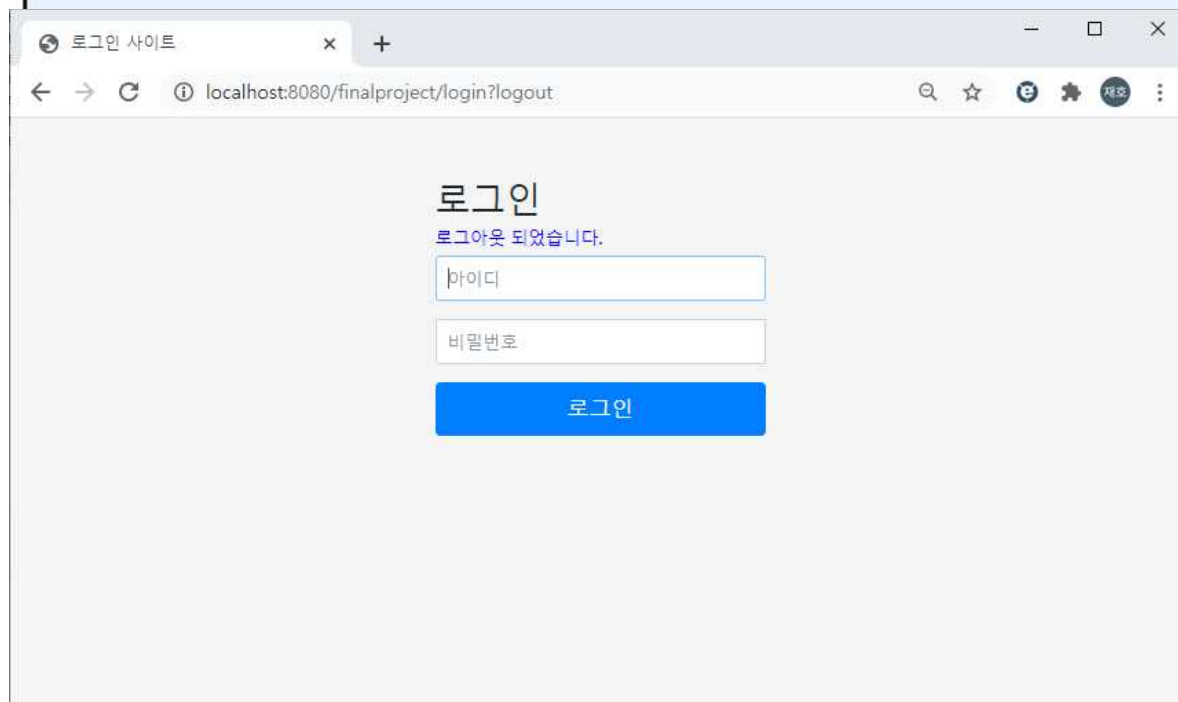
[수강신청 계속하기.](#)

3.4) 로그아웃 페이지 : login.jsp

로그아웃을 한 경우 login.jsp로 매핑하고, ?logout을 보내주고 로그아웃 메시지를 출력해준다.

컨트롤러에서, error! = null 의 경우에 메시지를 출력하도록 하였다.

```
@Controller
public class LoginController {
    @RequestMapping("/login")
    public String showLogin(@RequestParam (value="error", required=false) String error,
        @RequestParam (value="logout", required=false) String logout,
        Model model) {
        if(error!= null) {
            model.addAttribute("errorMsg", "아이디와 비밀번호를 확인하세요.");
        }
        if(logout!= null) {
            model.addAttribute("logoutMsg", "로그아웃 되었습니다.");
        }
        return "login";
    }
}
```



4. 데이터베이스 구조

* course 테이블

	year	semester	coursename	classification	professor	credit
▶	2020	1	객체지향언어1	복전선	황기태	3
	2020	1	운영체제	복전선	황기태	3
	2020	1	웹서버프로그래밍	복전선	김진환	3
	2019	2	웹프로그래밍기초	복전기	최병수	3
	2019	2	20대의 심리학	핵심	조난숙	3
	2019	2	4차 산업혁명과 현대조직	핵심	박성재	3
	2017	1	High-Success Point	자율	없음	1
	2020	2	객체지향언어2	복전선	황기태	3
	2015	2	경영학의 이해	핵교	최용식	2
	2016	2	공공도서관론	전선	박성재	3
	2015	1	기록과 정보	전선	이호신	3
	2016	1	기록관리학 개론	전선	강순애	3
	2016	2	기록평가및선별론	전선	강순애	3
	2016	2	기업경영과 비즈니스 전략	자율	시간강사	2
	2020	2	네트워크프로그래밍	복전선	정인환	3
	2015	1	대학과 지성	교필	김귀옥	3
	2020	1	데이터베이스	복전선	장재영	3
	2019	2	데이터통신	복전선	황호영	3
	2016	1	메타데이터의 이해	전선	박희진	3
	2015	1	문헌정보학의 이해	전기	서은경	2
	2016	2	미술사입문	자율	김효정	2
	2019	2	빅데이터기초	복전선	장재영	3
	2015	2	사고와 표현II	교필	권혁명	2
	2015	1	사고와 표현1	교필	나은미	2
	2015	1	상거래와 법	핵교A	박인갑	2
	2015	2	서지학	전기	강순애	3
	2017	1	설득전략과 글쓰기	인재	이상혁	3
	2015	2	세계화와 국제이해	핵교B	이태주	2
	2020	1	안드로이드프로그래밍	복전선	허준영	3
	2016	1	여가경영으로 배우는 삶...	자율	전형상	3
	2021	1	영상처리	전선	이항찬	3
	2015	1	영어커뮤니케이션 정취/...	교필	Niki Stam...	1
	2015	2	영어커뮤니케이션 정취/...	교필	Tato	1
	2020	2	웹프레임워크1	복전선	김남운	3
	2021	1	웹프레임워크2	전선	김남운	3
	2019	2	웹프로그래밍	복전선	황기태	3
	2015	2	인류문화의 이해	핵교B	문경덕	2
	2016	1	인테리어디자인과 풍수	자율	김미지자	2
	2020	1	자료구조	복전선	심규현	3
	2016	1	장서관리론	전선	김양우	3
	2016	2	전략적 의사결정과 문제...	자율	시간강사	3
	2016	2	정보자원의 기술과 접근	전선	박지영	3
	2017	1	정보자원의 분류	전선	박지영	3
	2015	2	정보학의 기초	전기	박희진	3
	2017	1	조사연구의 이해	전선	박성재	3
	2016	1	조직경영론	전선	박성재	3
	2015	2	지식정보와 진로탐색	전선	이호신	2
	2019	2	지식정보의 자원활용	핵교A	박성재	3
	2017	1	컴퓨터구조	복전선	황호영	3
	2016	1	컴퓨터와 인터넷개론	자율	조세홍	2
	2017	1	컴퓨터프로그래밍	복전기	황기태	3
	2015	2	패션코디네이션	핵교	유금화	2
	2016	2	한국문화생활사	자율	강성봉	2
	2017	1	한중일의 미의식	토대	지상현	3
	2015	1	현대인의 패션	핵교A	강운주	2
	2017	1	확률및통계	복전선	최병수	3

5. 결론 및 느낀점

Spring framework를 통해 처음으로 간단한 페이지를 만들어 봤는데, 이전에 JSP를 이용해서 만들때와 정말 다른점이 많다는 것을 느꼈다. 많은 부분이 Spring이 해주고 개발자가 할 일은 어느정도 정해져 있는 것이라고 생각이 들었다. 프레임워크를 정말 잘 다룰 수 있다면, 개발을 훨씬 효율적으로 할 수 있을 거라고 느꼈고 프레임워크에 대한 공부의 필요성을 정말 많이 느꼈다. 이번 과제를 통해 간단한 웹페이지를 만들었지만 정말 서비스가 되는 페이지를 만들기 위해 공부를 열심히 해야겠다고 다짐하는 좋은 경험이 되었다. 또한 이번 학기 동안 웹프레임 워크 1 과목을 들으면서 배웠던 지식들을 다시한번 적용 시키고 공부해야겠다는 생각이 들었다.