

A background graphic featuring a complex network of interconnected nodes and lines. The nodes are represented by circles of various sizes and colors, including dark blue, light blue, and grey. The lines are thin and grey, creating a web-like structure across the entire image. Some nodes are highlighted with larger, concentric circles.

네트워크 프로그래밍 구현

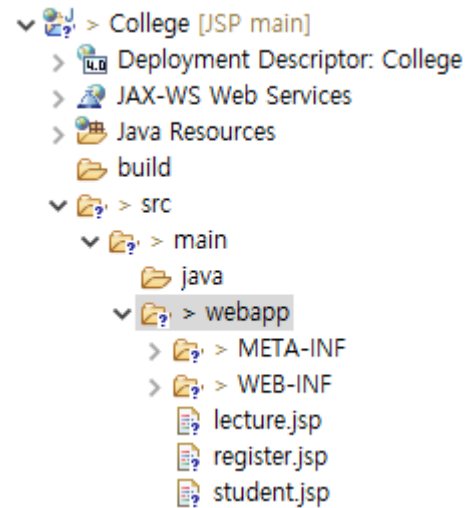
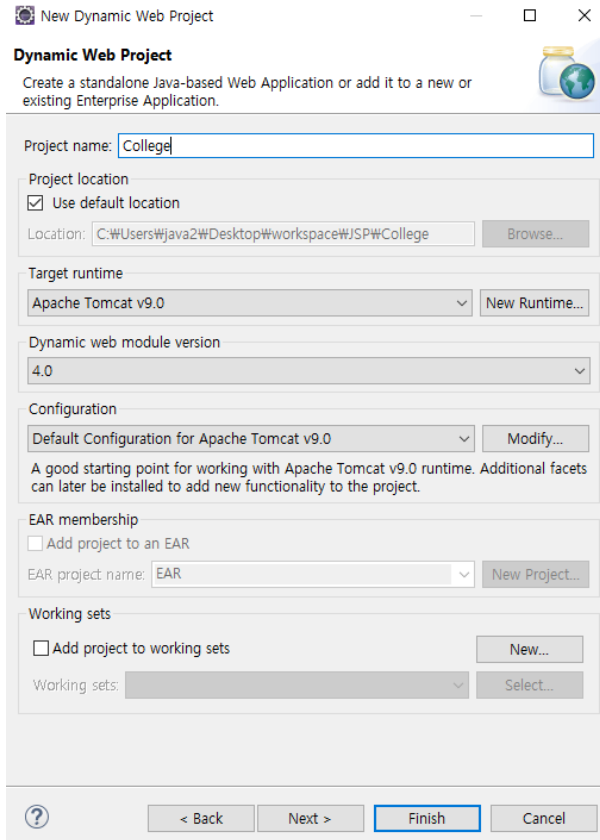
데이터베이스 & SQL

심규영

목차

- 1) 프로젝트 생성 및 구성
- 2) 화면 구현
- 3) 기능 구현
- 4) 실행

문제 1) 프로젝트 생성 및 구현



- 프로젝트명 : College
- WAS : Tomcat 9
- DB : Java2_college
- 개발도구 : Eclipse, Workbench

문제2) 화면 구현

강좌관리

[강좌관리](#) [수강관리](#) [학생관리](#)

강좌현황

등록

번호강좌명학점시간강의장

강좌등록

닫기

번호	
강좌명	
학점	23
시간	3131
강의장	
<div>추가</div>	

수강관리

[강좌관리](#) [수강관리](#) [학생관리](#)

수강현황

검색수강신청

학번이름강좌명강좌코드중간시험기말시험총점등급

수강신청

닫기

학번	
이름	
신청강좌	
<div>신청</div>	

학생관리

[강좌관리](#) [수강관리](#) [학생관리](#)

학생목록

등록

학번이름휴대폰학년주소

학생등록

닫기

학번	
이름	
휴대폰	
학년	1학년
주소	
<div>등록</div>	

- 강좌관리 화면 구현
- 수강관리 화면 구현
- 학생관리 화면 구현

문제3) 기능 구현

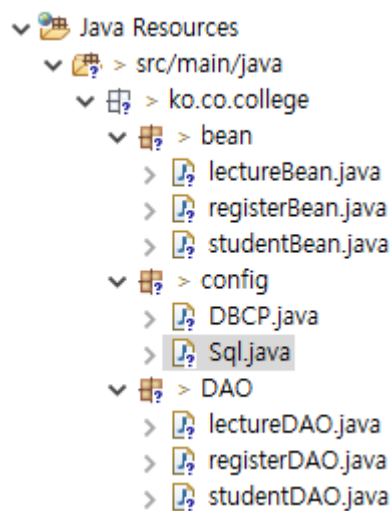
```
16 public class DBCP {
17     protected Connection conn;
18     protected PreparedStatement psmt;
19     protected Statement stmt;
20     protected ResultSet rs;
21
22     protected Logger logger = LoggerFactory.getLogger(this.getClass());
23
24     private static DataSource ds = null;
25     public static Connection getConnection() throws NamingException, SQLException {
26         if (ds == null) {
27             ds = (DataSource) new InitialContext()
28                 .lookup("java:comp/env/dbcp_java2_college");
29         }
30         return ds.getConnection();
31     }
32
33     /**
34      * 클래스 종료
35      */
36     public void close() {
37         try {
38             if(conn!=null) conn.close();
39             if(stmt!=null) stmt.close();
40             if(psmt!=null) psmt.close();
41             if(rs!=null) rs.close();
42         } catch (Exception e) {
43             e.printStackTrace();
44             logger.error("클래스 닫기 오류");
45             logger.error(e.getMessage());
46         }
47     }
48 }
```

```
<!-- java2_college 커넥션 풀 전역 설정 -->
<Resource auth="Container"
driverClassName="com.mysql.cj.jdbc.Driver"
maxIdle="16" maxTotal="16" maxWaitMillis="3000"
name="dbcp_java2_college" password="1234"
type="javax.sql.DataSource" |
url="jdbc:mysql://127.0.0.1:3306/java2_college"
username="root"/>

<!-- java2_college 커넥션 풀 자원 설정 -->
<ResourceLink
    global="dbcp_java2_college"
    name="dbcp_java2_college"
    type="javax.sql.DataSource"
/>
```

- 커넥션 풀 설정
- DBCP.class 생성

문제3) 기능 구현



```
1 package ko.co.college.bean;
2
3 public class lectureBean {
4     private int lecNo;
5     private String lecName;
6     private int lecCredit;
7     private int lecTime;
8     private String lecClass;
9     public int getLecNo() {
10         return lecNo;
11     }
12     public void setLecNo(int lecNo) {
13         this.lecNo = lecNo;
14     }
15     public String getLecName() {
16         return lecName;
17     }
18     public void setLecName(String lecName) {
```

```
1 package ko.co.college.DAO;
2
3 import ko.co.college.config.DBCP;
4
5 public class lectureDAO extends DBCP {
6     public lectureDAO () {
7         try {
8             conn = getConnection();
9         } catch (Exception e) {
10             e.printStackTrace();
11             logger.error("lecture 연결 오류");
12             logger.error(e.getMessage());
13         }
14     }
15
16     // Create
17
18     // read
19
20     // upload
21
22     // delete
23 }
```

- 각각의 Bean 과 DAO 생성
- Sql.java 생성

```
3 public class Sql {
4     // lecture
5     public static final String SELECT_LECTURE_LIST =
6         "";
7
8     // register
9
10    // student
11 }
```

문제3) 기능 구현

- 강좌현황 보기 기능 구현
- 강좌 등록 숨기기
- 강좌 등록 DAO 기능 추가

```
/**
 * 강좌 리스트 불러오기
 * @return List<lectureBean>
 */
public List<lectureBean> readLectureList() {
    List<lectureBean> lecturelist = new ArrayList<lectureBean>();
    try {
        stmt = conn.createStatement();
        rs = stmt.executeQuery(Sql.SELECT_LECTURE_LIST);
        while(rs.next()) {
            lectureBean lb = new lectureBean();
            lb.setLecNo(rs.getInt("lecNo"));
            lb.setLecName(rs.getString("lecName"));
            lb.setLecCredit(rs.getInt("lecCredit"));
            lb.setLecTime(rs.getInt("lecTime"));
            lb.setLecClass(rs.getString("lecClass"));
            lecturelist.add(lb);
        }
    } catch (Exception e) {
        e.printStackTrace();
        logger.error("lecture list 불러오기 오류");
        logger.error(e.getMessage());
    }
    return lecturelist;
}

// lecture
public static final String SELECT_LECTURE_LIST =
    "select "
    + "`lecNo`, "
    + "`lecName`, "
    + "`lecCredit`, "
    + "`lecTime`, "
    + "`lecClass`"
    + "from `lecture`";
```

```
<section style="display:none;">
    <h4>강좌등록</h4>
    <input type="button" value="닫기">
    <table border="1">
        <tr>
            <td>
                /**
                 * 강좌 등록
                 * @param lb
                 * @return either (1) the row count for SQL Data Manipulation
                 */
                public int insertLecture(lectureBean lb) {
                    int result = 0;
                    try {
                        psmt = conn.prepareStatement(Sql.INSERT_LECTURE);
                        psmt.setInt(1, lb.getLecNo());
                        psmt.setString(2, lb.getLecName());
                        psmt.setInt(1, lb.getLecCredit());
                        psmt.setInt(1, lb.getLecTime());
                        psmt.setString(1, lb.getLecClass());
                        result = psmt.executeUpdate();
                    } catch (Exception e) {
                        e.printStackTrace();
                        logger.error("lecture 등록 오류");
                        logger.error(e.getMessage());
                    }
                    return result;
                }
            </td>
        </tr>
    </table>
</section>
```

문제3) 기능 구현

```
6 request.setCharacterEncoding("UTF-8");
7 String lecNo = request.getParameter("lecNo");
8 String lecName = request.getParameter("lecName");
9 String lecCredit = request.getParameter("lecCredit");
10 String lecTime = request.getParameter("lecTime");
11 String lecClass = request.getParameter("lecClass");
12
13 lectureBean lb = new lectureBean();
14 lb.setLecNo(Integer.parseInt(lecNo));
15 lb.setLecName(lecName);
16 lb.setLecCredit(Integer.parseInt(lecCredit));
17 lb.setLecTime(Integer.parseInt(lecTime));
18 lb.setLecClass(lecClass);
19
20 lectureDAO ldao = new lectureDAO();
21 int result = ldao.insertLecture(lb);
22 ldao.close();
23
24 JSONObject json = new JSONObject();
25 json.addProperty("result", result);
26 json.addProperty("lecNo", lecNo);
27 json.addProperty("lecName", lecName);
28 json.addProperty("lecCredit", lecCredit);
29 json.addProperty("lecTime", lecTime);
30 json.addProperty("lecClass", lecClass);
31
32 String jsonData = json.toString();
33
34 out.print(jsonData);
35
36 $(''.btnLectureOpen').click(function(){
37     $('section').show();
38 });
39
40 $(''.btnLectureClose').click(function(){
41     $('section').hide();
42 });
```

```
43 $('input[type=submit]').click(function(){
44     const lecNo = $('input[name=lecNo']").val();
45     const lecName = $('input[name=lecName']").val();
46     const lecCredit = $('input[name=lecCredit']").val();
47     const lecTime = $('input[name=lecTime']").val();
48     const lecClass = $('input[name=lecClass']").val();
49
50     const jsonData = {
51         "lecNo" : lecNo,
52         "lecName" : lecName,
53         "lecCredit" : lecCredit,
54         "lecTime" : lecTime,
55         "lecClass" : lecClass
56     }
57
58     $.ajax({
59         url: './proc/lectureInsert.jsp',
60         type: 'POST',
61         data: jsonData,
62         dataType: 'json',
63         success: function(data){
64             if(data.result > 0) {
65                 alert('강좌등록성공!');
66                 let tr = "<tr>"
67                     + "<td>"+data.lecNo+"</td>"
68                     + "<td>"+data.lecName+"</td>"
69                     + "<td>"+data.lecCredit+"</td>"
70                     + "<td>"+data.lecTime+"</td>"
71                     + "<td>"+data.lecClass+"</td>"
72                     + "</tr>";
73
74                 $('section').hide();
75                 $(''.lectureList').append(tr);
76             } else {
77                 alert("강좌등록실패!");
78             }
79         }
80     });
81 });
```

- 강좌등록 DAO 기능 사용

- 강좌등록 화면 끄기 켜기 기능

- 강좌등록 기능 구현 및 동적 목록 출력

문제3) 기능 구현

```
// read
/**
 * 학생 테이블 리스트 읽기
 * @return List<studentBean>
 */
public List<studentBean> studentList() {
    List<studentBean> sbs = new ArrayList<>();
    try {
        stmt = conn.createStatement();
        rs = stmt.executeQuery(Sql.SELECT_STUDENT_LIST);
        while(rs.next()) {
            studentBean sb = new studentBean();
            sb.setStdno(rs.getString(1));
            sb.setStdName(rs.getString(2));
            sb.setStdHp(rs.getString(3));
            sb.setStdYear(rs.getInt(4));
            sb.setStdAddress(rs.getString(5));
            sbs.add(sb);
        }
    } catch (Exception e) {
        e.printStackTrace();
        logger.error("student 리스트 읽기 오류");
        logger.error(e.getMessage());
    }
    return sbs;
}
```

```
// student
public static final String SELECT_STUDENT_LIST =
    "select * from `student`";

studentDAO sdao = new studentDAO();
List<studentBean> sbs = sdao.studentList();
sdao.close();

<% for (studentBean sb : sbs) { %>
<tr>
    <td><%= sb.getStdno() %></td>
    <td><%= sb.getStdName() %></td>
    <td><%= sb.getStdHp() %></td>
    <td><%= sb.getStdYear() %></td>
    <td><%= sb.getStdAddress() %></td>
</tr>
<% } %>
```

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
<script>
    $(function(){
        $('.btnOpen').click(function(){
            $('#section').show();
        });
        $('.btnClose').click(function(){
            $('#section').hide();
        });
    });
</script>
```

-학생목록 리스트 출력

- 등록 버튼 클릭시

학생 등록 출력 및 닫기
시 제거 기능

문제3) 기능 구현

-학생 등록 기능 구현

- ajax 이용

- 동적 처리

```
// Create
/**
 * 학생 등록
 * @param sb
 * @return
 */
public int studentInsert(studentBean sb) {
    int result = 0;
    try {
        pstmt = conn.prepareStatement(Sql.INSERT_STUDENT);
        pstmt.setString(1, sb.getStdno());
        pstmt.setString(2, sb.getStdName());
        pstmt.setString(3, sb.getStdHp());
        pstmt.setInt(4, sb.getStdYear());
        pstmt.setString(5, sb.getStdAddress());
        result = pstmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
        logger.error("student 등록 오류!");
        logger.error(e.getMessage());
    }
    return result;
}

public static final String INSERT_STUDENT =
    "insert into `student`"
    + " (`stdNo`,`stdName`,`stdHp`,`stdYear`,`stdAddress`)"
    + " values (?,?,?,?,?)";
```

```
request.setCharacterEncoding("UTF-8");
String studentId = request.getParameter("studentId");
String studentName = request.getParameter("studentName");
String studentHp = request.getParameter("studentHp");
String studentYear = request.getParameter("studentYear");
String studentAddr = request.getParameter("studentAddr");
```

```
studentBean sb = new studentBean();
sb.setStdno(studentId);
sb.setStdName(studentName);
sb.setStdHp(studentHp);
sb.setStdYear(Integer.parseInt(studentYear));
sb.setStdAddress(studentAddr);
```

```
studentDAO sdao = new studentDAO();
int result = sdao.studentInsert(sb);
sdao.close();
```

```
JsonObject json = new JsonObject();
json.addProperty("result", result);
json.addProperty("studentId", studentId);
json.addProperty("studentName", studentName);
json.addProperty("studentHp", studentHp);
json.addProperty("studentYear", studentYear);
json.addProperty("studentAddr", studentAddr);
```

```
String jsonData = json.toString();
```

```
out.print(jsonData);
```

```
$('#input[type=submit]').click(function(){
    const studentId = $('#input[name=studentId']).val();
    const studentName = $('#input[name=studentName]').val();
    const studentHp = $('#input[name=studentHp]').val();
    const studentYear = $('#studentYear option:selected').val();
    const studentAddr = $('#input[name=studentAddr]').val();
```

```
const jsonData = {
    "studentId" : studentId,
    "studentName" : studentName,
    "studentHp" : studentHp,
    "studentYear" : studentYear,
    "studentAddr" : studentAddr
}
```

```
$.ajax({
    url: './proc/studentInsert.jsp',
    type: 'POST',
    data: jsonData,
    dataType: 'json',
    success: function(data){
        if(data.result > 0) {
            alert('학생등록성공!');
            let tr = "<tr>"
                + "<td>" + data.studentId + "</td>"
                + "<td>" + data.studentName + "</td>"
                + "<td>" + data.studentHp + "</td>"
                + "<td>" + data.studentYear + "</td>"
                + "<td>" + data.studentAddr + "</td>"
                + "</tr>";
```

```
$('#section').hide();
$('#.studentList').append(tr);
} else {
    alert("학생등록실패!");
}
```

```
});
});
```

문제3) 기능 구현

```
// read
/**
 * 수강테이블 리스트 읽기
 * @return
 */
public List<registerBean> readRegisterList() {
    List<registerBean> rbs = new ArrayList<registerBean>();
    try {
        stmt = conn.createStatement();
        rs = stmt.executeQuery(Sql.SELECT_REGISTER_LIST);
        while(rs.next()) {
            registerBean rb = new registerBean();
            rb.setRegStdNo(rs.getString(1));
            rb.setStdName(rs.getString(2));
            rb.setLecName(rs.getString(3));
            rb.setRegLecNo(rs.getInt(4));
            rb.setRegMidScore(rs.getInt(5));
            rb.setRegFinalScore(rs.getInt(6));
            rb.setRegTotalScore(rs.getInt(7));
            rb.setRegGrade(rs.getString(8));
            rbs.add(rb);
        }
    } catch (Exception e) {
        e.printStackTrace();
        logger.error("register 리스트 읽기 오류");
        logger.error(e.getMessage());
    }
    return rbs;
}
```

```
// register
public static final String SELECT_REGISTER_LIST =
    "select "
    + " r.`regStdNo`, "
    + " s.`stdName`, "
    + " l.`lecName`, "
    + " r.`regLecNo`, "
    + " r.`regMidScore`, "
    + " r.`regFinalScore`, "
    + " r.`regTotalScore`, "
    + " r.`regGrade` "
    + " from `register` as r "
    + " join `student` as s ON r.regStdNo = s.stdNo "
    + " join `lecture` as l ON r.regLecNo = l.lecNo";

<%
    registerDAO rdao = new registerDAO();
    List<registerBean> rbs = rdao.readRegisterList();
    rdao.close();
%>
<% for (registerBean rb : rbs) { %>
<tr>
    <td><%= rb.getRegStdNo() %></td>
    <td><%= rb.getStdName() %></td>
    <td><%= rb.getLecName() %></td>
    <td><%= rb.getRegLecNo() %></td>
    <td><%= rb.getRegMidScore() %></td>
    <td><%= rb.getRegFinalScore() %></td>
    <td><%= rb.getRegTotalScore() %></td>
    <td><%= rb.getRegGrade() %></td>
</tr>
<% } %>
```

-수강현황 register 테이블 수강 데이터 출력

문제3) 기능 구현

-수강현황 register 테이블 수강 데이터 출력

- 학번 검색시 테이블 변경 (동적)

```
/**
 * 수강현황 검색 결과 읽기
 * @param searchRegister
 * @return
 */
public List<registerBean> searchRegisterList(String searchRegister) {
    List<registerBean> rbs = new ArrayList<registerBean>();
    try {
        psmt = conn.prepareStatement(Sql.SEARCH_REGISTER_LIST);
        psmt.setString(1, searchRegister);
        rs = psmt.executeQuery();
        while(rs.next()) {
            registerBean rb = new registerBean();
            rb.setRegStdNo(rs.getString(1));
            rb.setStdName(rs.getString(2));
            rb.setLecName(rs.getString(3));
            rb.setRegLecNo(rs.getInt(4));
            rb.setRegMidScore(rs.getInt(5));
            rb.setRegFinalScore(rs.getInt(6));
            rb.setRegTotalScore(rs.getInt(7));
            rb.setRegGrade(rs.getString(8));
            rbs.add(rb);
        }
    } catch (Exception e) {
        e.printStackTrace();
        logger.error("register 검색 리스트 읽기 오류");
        logger.error(e.getMessage());
    }
    return rbs;
}
```

```
public static final String SEARCH_REGISTER_LIST =
    "select "
    + " r.`regStdNo`, "
    + " s.`stdName`, "
    + " l.`lecName`, "
    + " r.`regLecNo`, "
    + " r.`regMidScore`, "
    + " r.`regFinalScore`, "
    + " r.`regTotalScore`, "
    + " r.`regGrade` "
    + " from `register` as r "
    + " join `student` as s ON r.regStdNo = s.stdNo "
    + " join `lecture` as l ON r.regLecNo = l.lecNo "
    + " WHERE r.`regStdNo`=?";

<%
    request.setCharacterEncoding("UTF-8");
    String searchRegister = request.getParameter("searchRegister");

    registerDAO rdao = new registerDAO();
    List<registerBean> rbs = rdao.searchRegisterList(searchRegister);
    rdao.close();

    Gson gson = new Gson();
    String jsondata = gson.toJson(rbs);

    out.print(jsondata);
%>
```

문제3) 기능 구현

- 학번 검색시 테이블 변경 (동적)
- 수강신청 버튼 클릭시 동적으로 수강목록 불러오는 기능

```
$('#btnSearchRegister').click(function(){
    const searchRegister = $('#searchRegister').val();

    const jsonData = {
        "searchRegister" : searchRegister
    }

    $.ajax({
        url: './proc/searchRegister.jsp',
        type: 'POST',
        data: jsonData,
        dataType: 'json',
        success: function(data){
            $('#section').hide();
            if(true) {
                $('#registerList > tbody').empty();
                for(const rbs of data){
                    const tags = "<tr>"
                        + "<td>"+rbs.regStdNo+"</td>"
                        + "<td>"+rbs.stdName+"</td>"
                        + "<td>"+rbs.lecName+"</td>"
                        + "<td>"+rbs.regLecNo+"</td>"
                        + "<td>"+rbs.regMidScore+"</td>"
                        + "<td>"+rbs.regFinalScore+"</td>"
                        + "<td>"+rbs.regTotalScore+"</td>"
                        + "<td>"+rbs.regGrade+"</td>"
                        + "</tr>";
                    $('#registerList').append(tags);
                }
            } else {
                alert("일지하는 학번이 없습니다.");
            }
        }
    });
});
```

```
<%
    lectureDAO ldao = new lectureDAO();
    List<lectureBean> lbs = ldao.readLecNameList();
    ldao.close();

    Gson gson = new Gson();
    String jsondata = gson.toJson(lbs);

    out.print(jsondata);
%>
public List<lectureBean> readLecNameList() {
    List<lectureBean> lbs = new ArrayList<>();
    try {
        stmt = conn.createStatement();
        rs = stmt.executeQuery(Sql.SELECT_LECTURE_NAME_LIST);
        while(rs.next()) {
            lectureBean lb = new lectureBean();
            lb.setLecName(rs.getString(1));
            lbs.add(lb);
        }
    } catch (Exception e) {
        e.printStackTrace();
        logger.error("lecture name list 불러오기 오류");
        logger.error(e.getMessage());
    }
    return lbs;
}

public static final String SELECT_LECTURE_NAME_LIST =
    "select `lecName` from `lecture`";
```

```
$(function(){
    $('#btnOpen').click(function(){
        $('#section').show();

        $.ajax({
            url: './proc/lectureList.jsp',
            type: 'POST',
            dataType: 'json',
            success: function(data){
                for(const lbs of data) {
                    const option =
                        "<option value="+lbs.lecName+">"
                        + lbs.lecName + "</option>";
                    $('#select[name=lecName]').append(option);
                }
            }
        });
    });
});
```

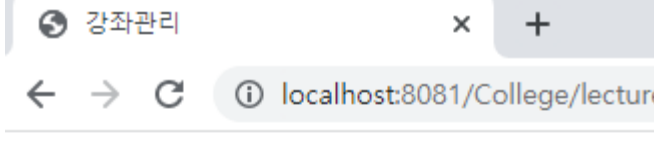
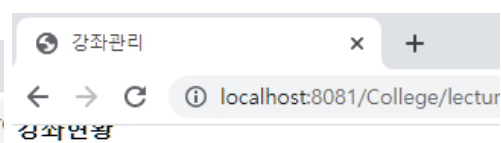
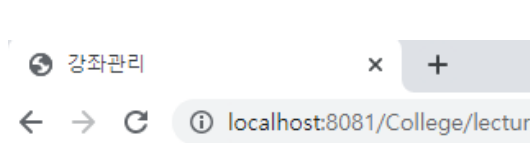
문제4) 실행

localhost:8081 내용:

강좌등록성공!

확인

- 강좌 목록
- 등록 기능 동작 화면



강좌관리

[강좌관리](#) [수강관리](#) [학생관리](#)

강좌현황

번호	강좌명	학점	시간	강의장
159	인지행동심리학	3	40	본304
167	운영체제론	3	25	본B05
234	중급 영문법	3	20	본201
239	세법개론	3	40	본204
248	빅데이터 개론	3	20	본B01
253	컴퓨팅사고와 코딩	2	10	본B02
349	사회복지 마케팅	2	50	본301

번호	강좌명	학점	시간	강의장
159	인지행동심리학	3	40	본304
167	운영체제론	3	25	본B05
234	중급 영문법	3	20	본201
239	세법개론	3	40	본204
248	빅데이터 개론	3	20	본B01
253	컴퓨팅사고와 코딩	2	10	본B02
349	사회복지 마케팅	2	50	본301

강좌등록

번호	501
강좌명	테스트강좌
학점	1
시간	50
강의장	본501
<input type="button" value="추가"/>	

강좌관리

[강좌관리](#) [수강관리](#) [학생관리](#)

강좌현황

번호	강좌명	학점	시간	강의장
159	인지행동심리학	3	40	본304
167	운영체제론	3	25	본B05
234	중급 영문법	3	20	본201
239	세법개론	3	40	본204
248	빅데이터 개론	3	20	본B01
253	컴퓨팅사고와 코딩	2	10	본B02
349	사회복지 마케팅	2	50	본301
501	테스트강좌	1	50	본501

문제4) 실행

- 수강 목록 기능 동작 화면
- 학번 검색 기능
- 수강신청 버튼 클릭시 강좌목록 동적으로 불러오는 기능

수강관리

← → ↺ localhost:8081/College/register.jsp

수강관리

[강좌관리](#) [수강관리](#) [학생관리](#)

수강현황

20201016

검색

수강신청

학번	이름	강좌명	강좌코드	중간시험	기말시험	총점	등급
20201126	김춘추	중급 영문법	234	36	42	78	C
20201016	김유신	빅데이터 개론	248	24	62	86	B
20201016	김유신	컴퓨팅사고와 코딩	253	28	40	68	D
20201126	김춘추	세법개론	239	37	57	94	A
20210216	장보고	사회복지 마케팅	349	28	68	96	A
20210326	강감찬	사회복지 마케팅	349	16	65	81	B
20201016	김유신	운영체제론	167	18	38	56	F
20220416	이순신	사회복지 마케팅	349	25	58	83	B

수강관리

← → ↺ localhost:8081/College/register.jsp

수강관리

[강좌관리](#) [수강관리](#) [학생관리](#)

수강현황

20201016

검색

수강신청

학번	이름	강좌명	강좌코드	중간시험	기말시험	총점	등급
20201016	김유신	빅데이터 개론	248	24	62	86	B
20201016	김유신	컴퓨팅사고와 코딩	253	28	40	68	D
20201016	김유신	운영체제론	167	18	38	56	F

수강신청

닫기

학번

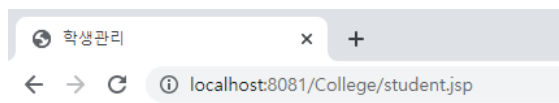
이름

신청강좌

세법개론 ▼

신청

문제4) 실행

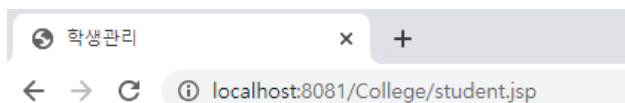


학생관리

[강좌관리](#) [수강관리](#) [학생관리](#)

학생목록

등록				
학번	이름	휴대폰	학년	주소
20201016	김유신	010-1234-1001	3	anywhere
20201126	김춘추	010-1234-1002	3	경상남도 경주시
20210216	장보고	010-1234-1003	2	전라남도 완도시
20210326	강감찬	010-1234-1004	2	서울시 영등포구
20220416	이순신	010-1234-1005	1	부산시 부산진구
20220521	송상현	010-1234-1006	1	부산시 동래구



학생목록

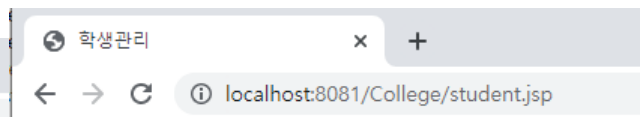
등록				
학번	이름	휴대폰	학년	주소
20201016	김유신	010-1234-1001	3	anywhere
20201126	김춘추	010-1234-1002	3	경상남도 경주시
20210216	장보고	010-1234-1003	2	전라남도 완도시
20210326	강감찬	010-1234-1004	2	서울시 영등포구
20220416	이순신	010-1234-1005	1	부산시 부산진구
20220521	송상현	010-1234-1006	1	부산시 동래구

학생등록

닫기	
학번	<input type="text" value="20221107"/>
이름	<input type="text" value="홍길동"/>
휴대폰	<input type="text" value="010-1234-2001"/>
학년	<input type="text" value="1학년"/>
주소	<input type="text" value="아무도 아무시"/>
<input type="button" value="등록"/>	

localhost:8081 내용:

학생등록성공!



학생관리

[강좌관리](#) [수강관리](#) [학생관리](#)

학생목록

등록				
학번	이름	휴대폰	학년	주소
20201016	김유신	010-1234-1001	3	anywhere
20201126	김춘추	010-1234-1002	3	경상남도 경주시
20210216	장보고	010-1234-1003	2	전라남도 완도시
20210326	강감찬	010-1234-1004	2	서울시 영등포구
20220416	이순신	010-1234-1005	1	부산시 부산진구
20220521	송상현	010-1234-1006	1	부산시 동래구
20221107	홍길동	010-1234-2001	1	아무도 아무시

- 학생 목록 기능 동작 화면

확인