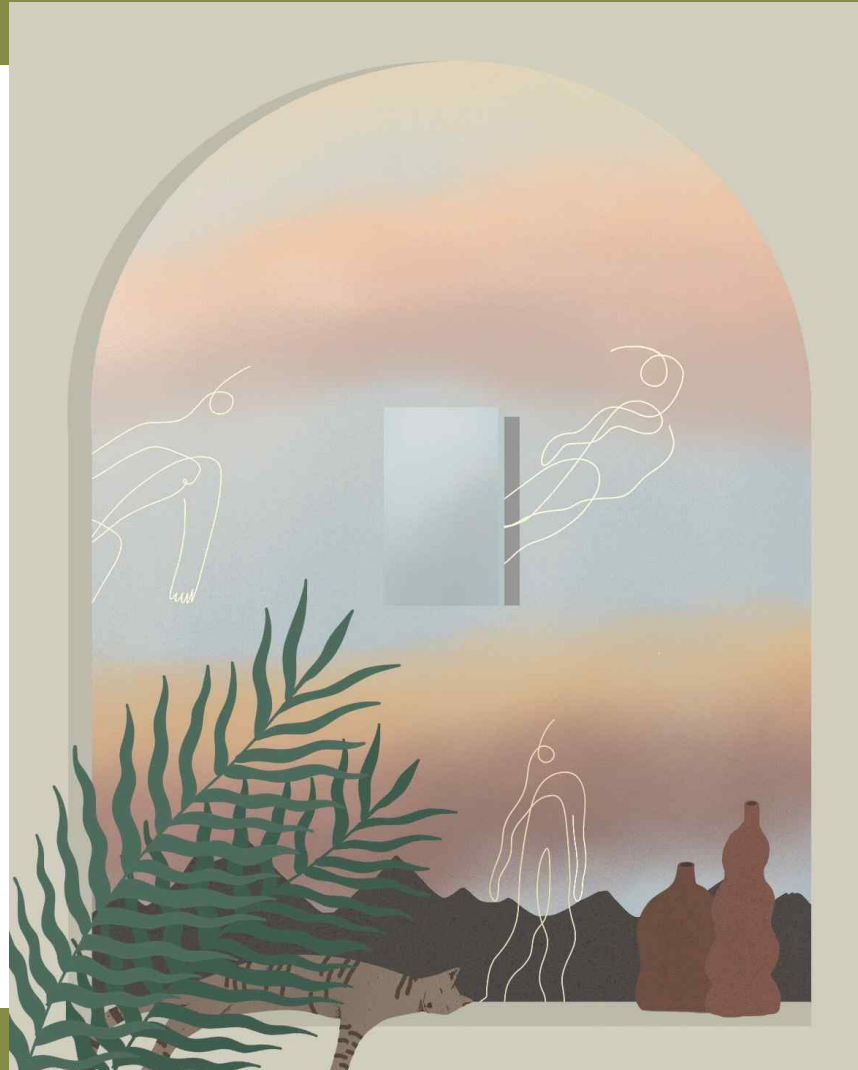


포트폴리오

## 강의평가 웹 사이트 구축

이희성



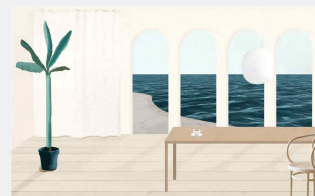
# INDEX



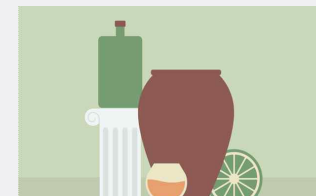
01. 제작동기



02. 프로그램 설명



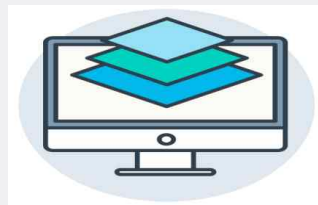
03. 개발환경



04. 제작기간



05. 설계구조



06. Front-End



07. Back-End



08. 보안 및 후기

# 01.

## 제작 동기

제작 동기 :

강의 수강 전에 수강평가를 보고 싶어하는 학생들의 요구가 많습니다.

- 웹 프로그래머로 진출하기 위한 필요한 학습을 했으며
- 이를 바탕으로 모든 웹 페이지에 필수적으로 사용되는 기능,
- SMS연결 보안 기능의 게시판 등을 사용하여
- 웹 사이트를 구축해 봄으로써
- 취업하여 회사업무를 하기 위하여 제작하게 되었습니다.

# 02.

## 프로그램 차별점

---

---

---

1. 게시판 부분 : XSS 해킹 방어를 위한 게시판 작성
2. 신고글
  - 모달 창으로 구현함으로써 게시판과 같이 볼 수 있어 편리합니다.
  - 신고 되었다면 매니저의 메일로 신고 접수 메일이 발송됩니다.
3. SMS
  - 실제 Cafe24 호스팅 업체와 연결하여 유료인 메시지 전송이 가능하도록 구현 하였습니다.
4. API
  - Google API 중 mail 관련 API를 사용하여 Gmail을 통한 메일 전송이 가능하도록 구현 하였습니다.

# 03.

## 개발환경

Front-End

BootStrap 3

JavaScript

Tomcat v9.0

Back-End

JDK v1.8

Mysql 5.1

웹 디자인에 대해 어렵지 않게 접근 가능한 부트스트랩은 처음 웹 프로그램을 개발함에 있어 필수적인 프레임워크라고 생각했습니다.

학원을 통해 학습한 JSP를 이용하여 웹 개발을 하였습니다.

# 04.

## 제작기간

---

---

---

총 개발 기간	1개월 2020.06.29 ~ 2020.07.31 (보완중에 있습니다.)	
작업	강의 평가 웹 사이트 개발	
작업 별 개발 기간	작업	개발 기간
	계획	6월 29일 ~ 7월 01일
	요구 분석	7월 02일 ~ 7월 04일
	설계	7월 05일 ~ 7월 15일
	구현	7월 16일 ~ 7월 25일
	시험 및 유지보수	7월 26일 ~ 현재
개발 순서	계획 → 요구 분석 → 설계 → 구현 → 시험 및 유지 보수	
필요 자원	개발용 PC, 개발 공간	

# 05.

## 설계 구조

---

---

---

### DB 구조

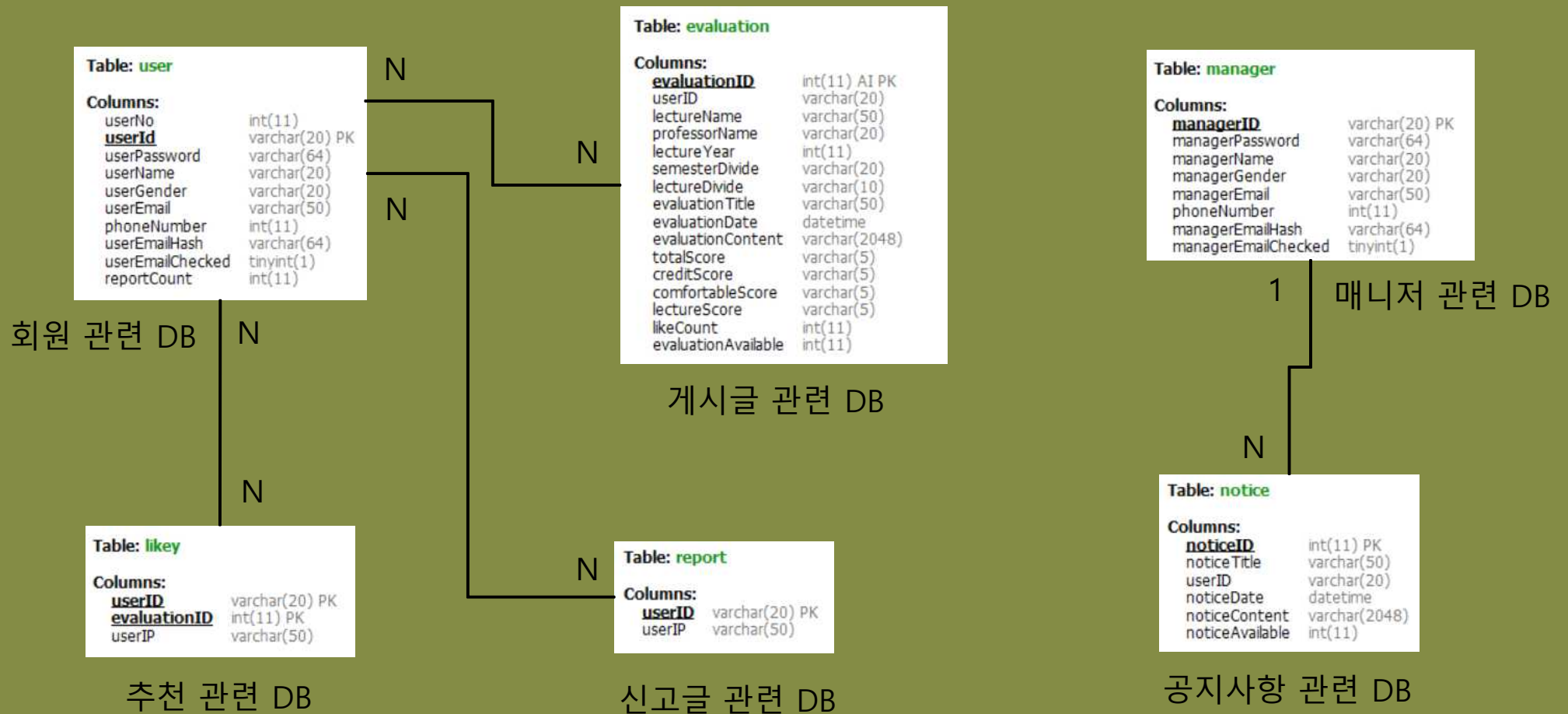
---

### 패키지 구조

---

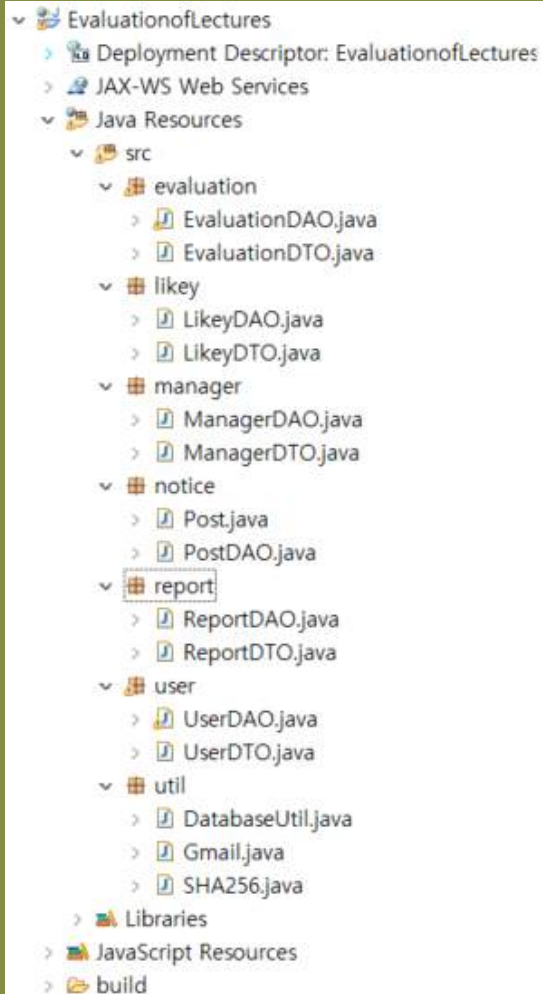
---

# DB 모델링



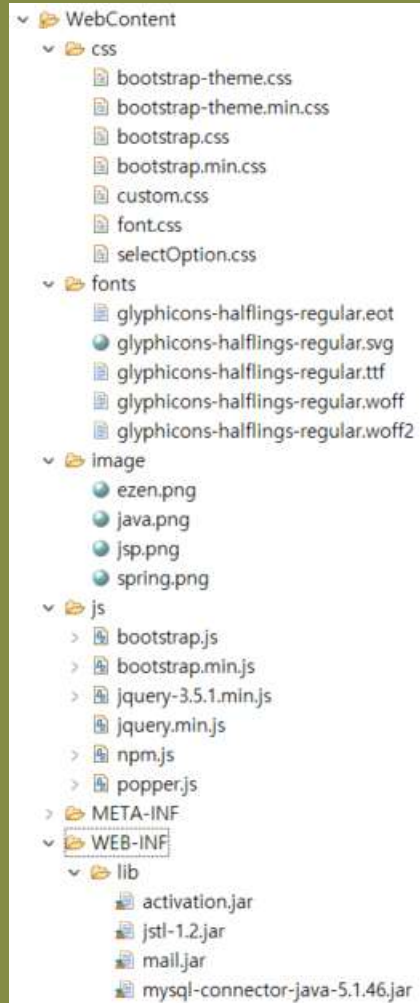


# 패키지 구조



1. 자바 클래스 중 DTO 클래스는 로직을 갖고 있지 않는 순수 데이터 객체입니다.  
즉, 속성과 그 속성에 접근하기 위한 getter, setter 메소드로만 구성되어 있습니다.
2. 자바 클래스 중 DAO 클래스는 데이터베이스의 data에 접근하기 위한 객체입니다.  
웹 서버가 DB와 연결하는데 매번 Connection 객체를 생성하는 것을 해결하기 위하여 전용 객체를 생성한 것입니다.  
즉, DB를 사용해 데이터를 조회 및 조작하는 기능을 전담하는 객체입니다.

# 패키지 구조



1. 웹 구현에 필요한 css, js, image 파일들을 넣어 놓은 폴더 입니다.
2. 프로그램 구현에 사용한 jar파일 또한 lib 폴더에 넣어두었습니다.

# 패키지 구조

1. 보여지는 화면에 대한 JSP 파일 입니다.
2. JSP 페이지 중 ~Action.jsp 파일은 이벤트 발생 시 해당 관련 이벤트를 처리하는 페이지입니다.

- colljason.jsp
- deleteAction\_index.jsp
- deleteAction.jsp
- deleteUser.jsp
- emailCheckAction.jsp
- emailSendAction.jsp
- emailSendConfirm.jsp
- emailSendReAction.jsp
- footer.jsp
- index.jsp
- indexAction.jsp
- indexBoard.jsp
- likeAction.jsp
- logoutAction.jsp
- main\_manager.jsp
- main.jsp
- managerEmailCheckAction.jsp
- managerEmailSendAction.jsp
- managerEmailSendConfirm.jsp
- managerEmailSendReAction.jsp
- managerLogin.jsp
- managerLoginAction.jsp
- managerModifyingAction.jsp
- managerModifyingInformation.jsp
- managerRegister.jsp
- managerRegisterAction.jsp
- managerSQL.sql
- memberManagement.jsp
- modifyingAction.jsp
- modifyingInformation.jsp
- noticeBoard.jsp
- reportAction\_doubleCheck.jsp
- sms.jsp
- smssend.jsp
- update\_index.jsp
- update.jsp
- updateAction\_index.jsp
- updateAction.jsp
- userLogin.jsp
- userLoginAction.jsp
- userRegister.jsp
- userRegisterAction.jsp
- view\_index.jsp
- view.jsp
- write.jsp
- writeAction.jsp

# 06.

## Front-End

---

---

---

메인화면

회원관리 toggle

회원가입

로그인

매니저 승인 대기

매니저 로그인

메인화면\_매니저

매니저\_회원관리

매니저\_SMS

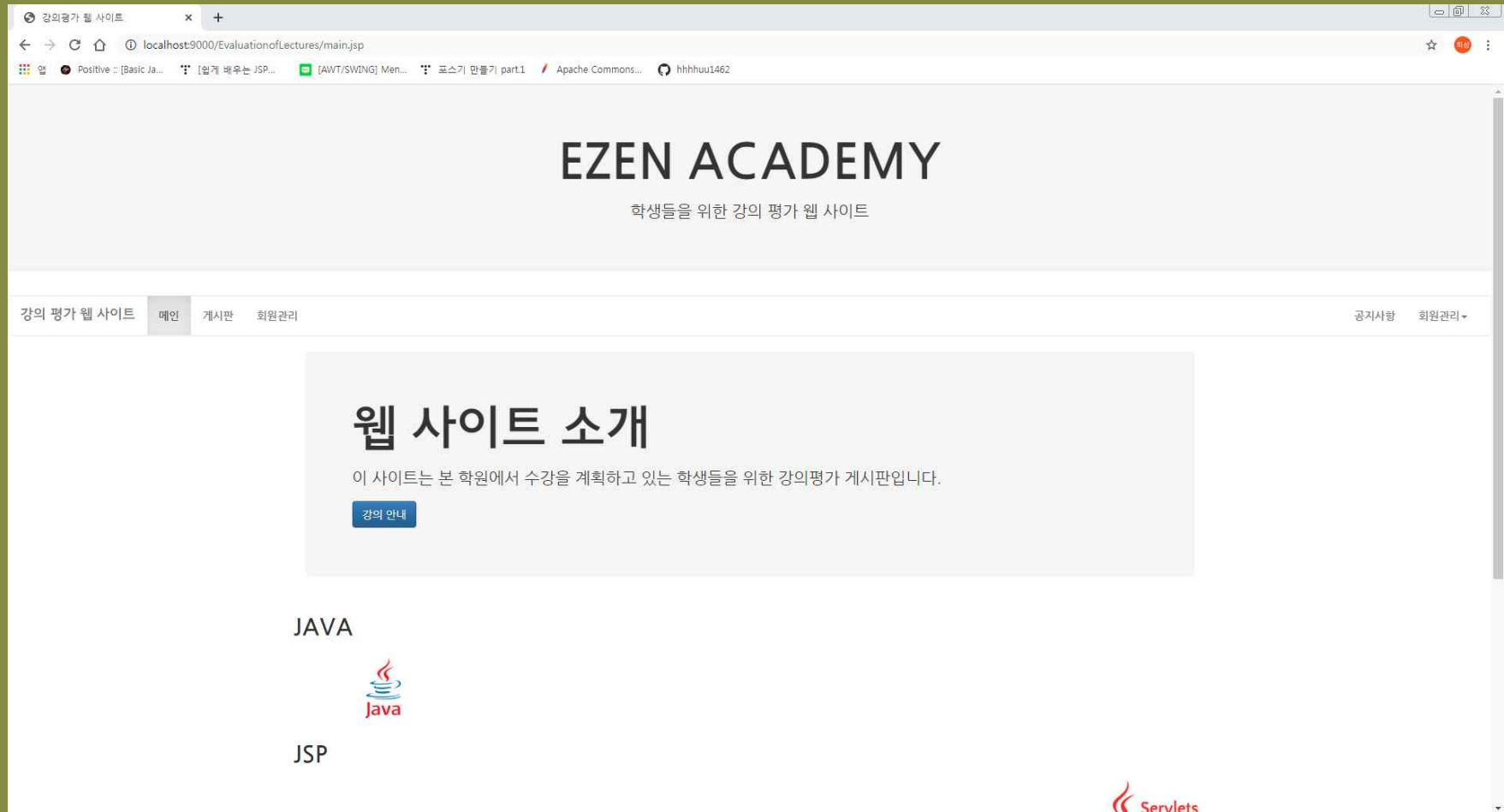
공지사항 작성

게시판 글 작성

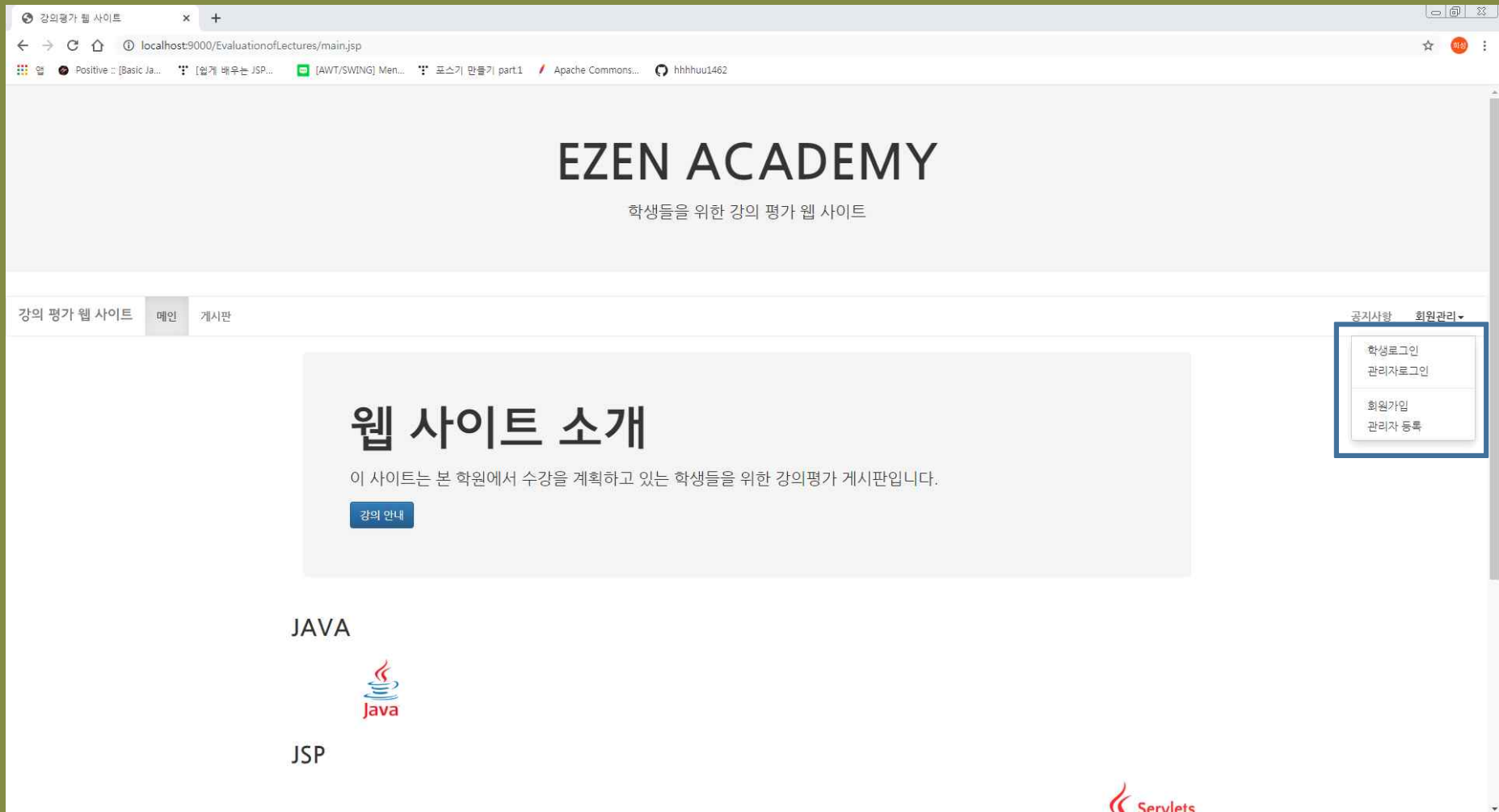
게시판 글 열람

게시판 조건 검색

# 메인 화면 – 프로그램 시작 시 보여지는 페이지



# 회원관리 toggle



# 회원가입

강의평가 웹 사이트

localhost:9000/EvaluationofLectures/userRegister.jsp

강의평가 웹 사이트

공지사항

회원관리

회원가입

아이디

비밀번호

이름

남자

여자

이메일

휴대폰 번호('-' 빼고 입력)

휴대폰번호를 입력해 주세요

회원가입

Copyright © 2020 HeeSung All Rights Reserved.

Address. 서울특별시 마포구 망원동

Tel. 010 - 2282 - 4338

# 로그인

강의평가 웹 사이트

java/Pos 관리 프로그램.pptx at /

+

localhost9000/EvaluationofLectures/userLogin.jsp

열Positive :: [Basic Ja...[입계 배우는 JSP...[AWT/SWING] Men...포스기 만들기 part1Apache Commons...hhhhhu1462

강의 평가 웹 사이트

메인

게시판

내용을 입력하세요.

검색

공지사항

회원관리

EZEN

이젠아카데미컴퓨터학원

로그인 화면

아이디

비밀번호

로그인

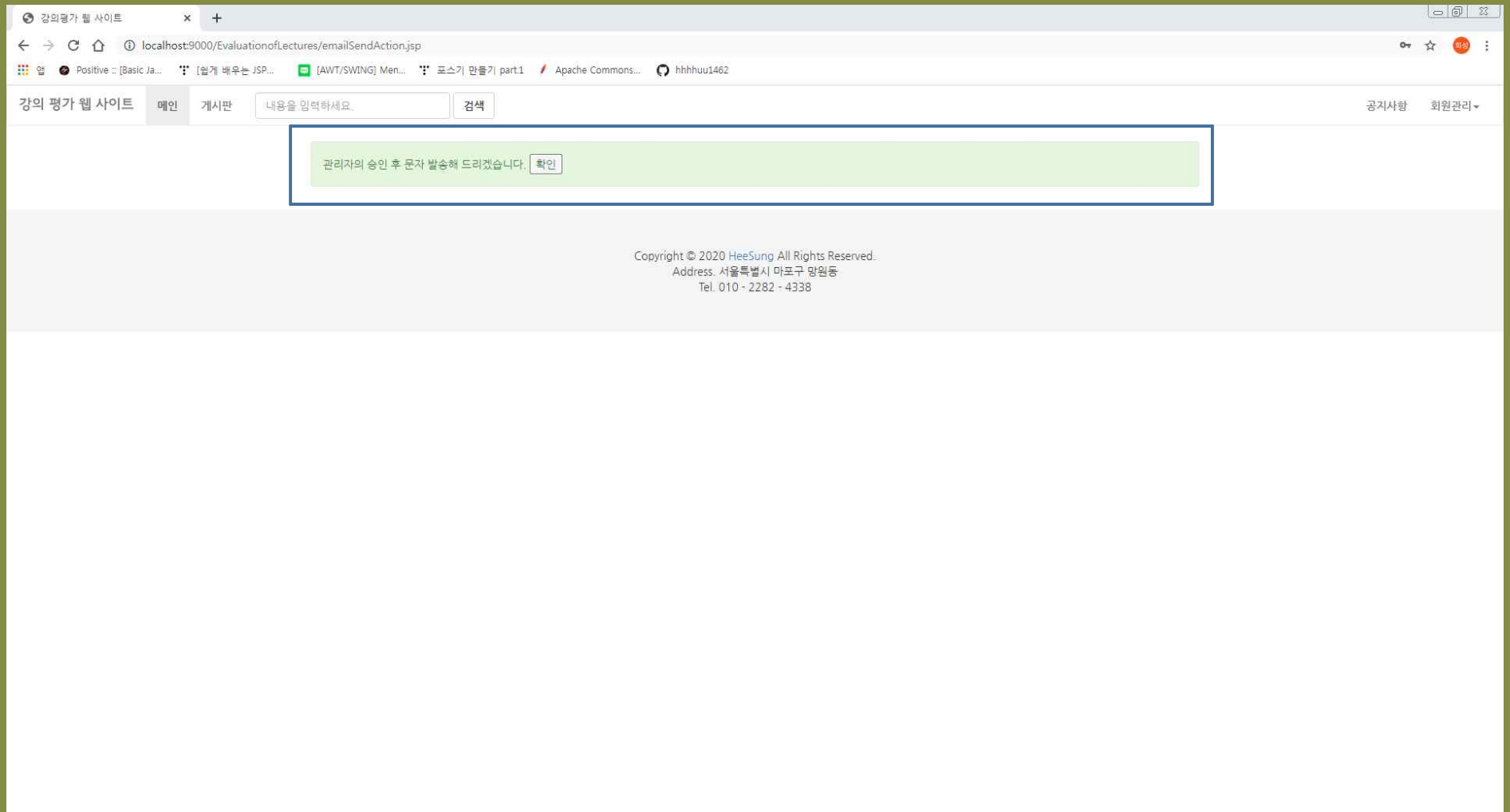
Copyright © 2020 HeeSung All Rights Reserved.

Address, 서울특별시 마포구 망원동

Tel, 010 - 2282 - 4338



# 매니저 승인 대기




# 매니저 로그인

강의평가 웹 사이트

localhost:9000/EvaluationofLectures/managerLogin.jsp

강의 평가 웹 사이트 메인 게시판

공지사항 회원관리



이젠아카데미컴퓨터학원

매니저 로그인 화면

manager

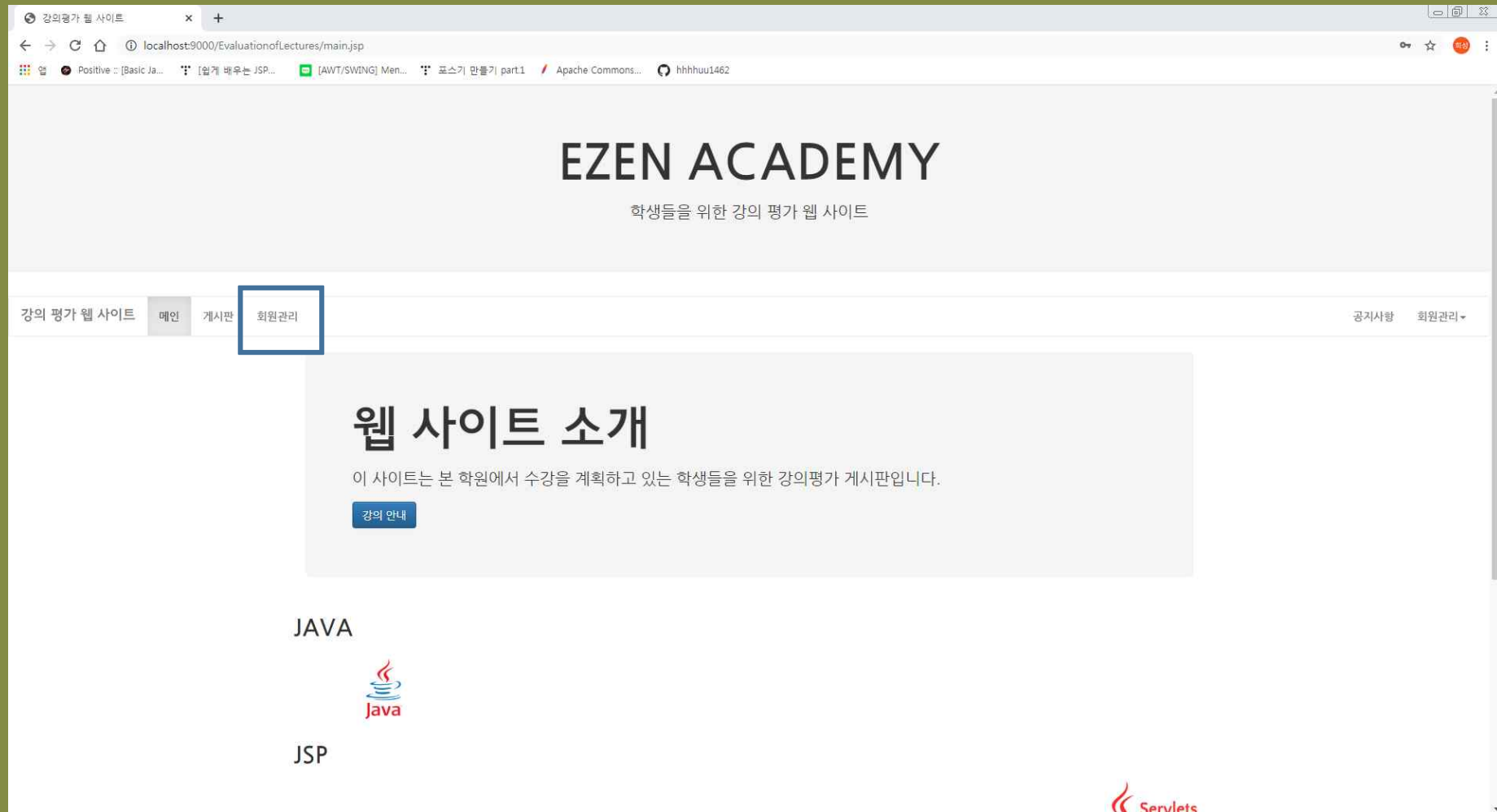
비밀번호

로그인

매니저 ID는 manager로 고정

Copyright © 2020 HeeSung All Rights Reserved.  
Address. 서울특별시 마포구 망원동  
Tel. 010 - 2282 - 4338

# 메인 화면\_매니저



# 매니저 회원관리 창

강의평가 웹 사이트

localhost:9000/EvaluationofLectures/memberManagement.jsp

☆ 44%

열

Positive :: [Basic Ja...

[쉽게 배우는 JSP...

[AWT/SWING] Men...

포스기 만들기 part1

Apache Commons...

hnhhuu1462

# EZEN ACADEMY

학생들을 위한 강의 평가 웹 사이트

강의 평가 웹 사이트

메인

게시판

회원관리

공지사항

회원관리

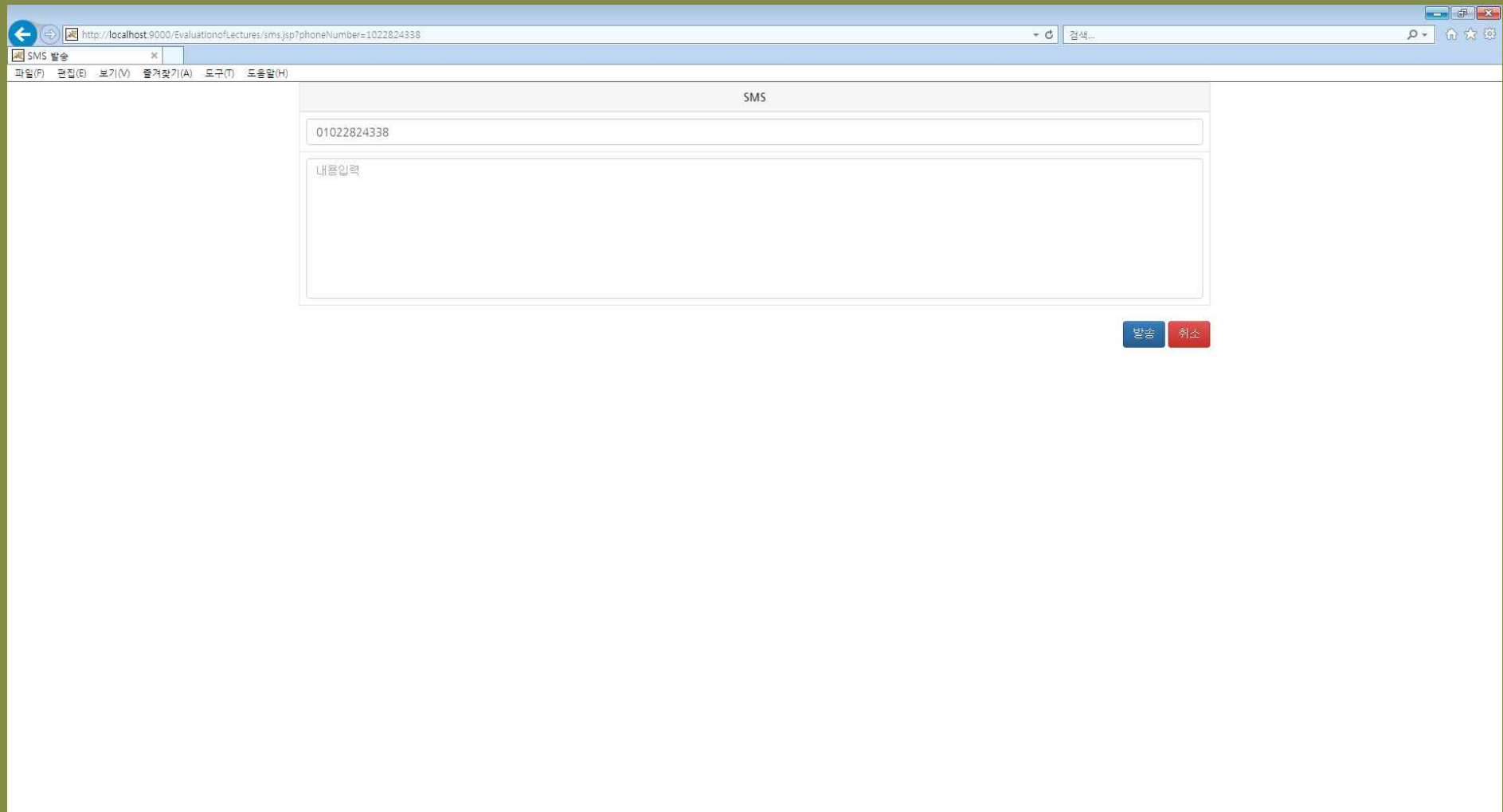
회원 수	총 게시물 수	총 공지사항 수
2	1	1

No.	회원명	회원ID	게시글 수	좋아요 개수	신고	가입요청	SMS	정보
1	aa	aa	1	0	0건	승인완료	SMS	삭제
2	이희성	hnhhuu1462	0	0	0건	요청	SMS	삭제

Copyright © 2020 HeeSung All Rights Reserved.  
Address. 서울특별시 마포구 망원동  
Tel. 010 - 2282 - 4338

가입 승인 요청  
문자 메시지 전송  
회원정보 삭제의 기능 담당

# 매니저 SMS 전송



Browser address bar: <http://localhost:9000/Evaluationoflectures/sms.jsp?phoneNumber=1022824338>

Page title: SMS 발송

Menu: 파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)

SMS

01022824338

내용입력

발송 취소

# 공지사항 관리

JSP 게시판 웹 사이트

localhost:9000/Evaluationoflectures/noticeBoard.jsp

강의 평가 웹 사이트 메인 게시판 회원관리

공지사항 회원관리

번호	제목	작성자	작성일
1	지나친 욕설은 피해주세요	manager	2020-07-23 18시12분

글쓰기

Copyright © 2020 HeeSung All Rights Reserved.  
Address: 서울특별시 마포구 망원동  
Tel. 010 - 2282 - 4338

매니저에 의해서만 글 작성 가능  
일반 회원은 열람만 가능하도록 구현

# 게시판 글 작성

JavaScript :: 자바스크립트 확인 × | jsp sms 전송 - Google 검색 × | (27) 서블릿/JSP 강의 53 - 자세한 × | JSP 게시판 웹 사이트 × +

localhost:9000/EvaluationofLectures/indexBoard.jsp ☆ EX ⓘ ? ↻ ⚙️ ⚠️

앱 Google NAVER hhhhuu1462/hhhh... GitHub - JinYongH... God of Java 2nd 인프런 - 자바 프로... 꾸슬 공부하자~!!!!1 B 콤포넌트 · 부트스... Bootstrap :: 부트스... 명칠 일지 :: [Bootst...

강의 평가 웹 사이트 메인 **게시판** 공지사항 회원관리

평가 등록

강의명

교수명

수강년도

해당학기

전공선택

제목

내용입력

종합

성적

분위기

강의

등록

취소

# 게시판 글 열람

강의 평가 웹 사이트 메인 게시판

전체 최신순 내용을 입력하세요. 검색

번호	제목	작성자	작성일	추천
1	너무 재밌어요	aa	2020-07-27 16시03분	0

글쓰기

Copyright © 2020 HeeSung All Rights Reserved.  
Address. 서울특별시 마포구 망원동  
Tel. 010 - 2282 - 4338

회원에게만 글 작성 가능  
매니저는 열람만 가능하도록 구현



# 게시판 글 열람

JSP 게시판 웹 사이트
Java/Pos 관리 프로그램.pptx at
localhost:9000/EvaluationofLectures/view\_index.jsp?evaluationID=1
강의 평가 웹 사이트
메인
게시판
공지사항
회원관리

게시판

글 제목	너무 재밌어요
작성자	aa
작성일자	2020-07-27 16시03분
강의명	너무 재밌어요
교수명	정현희
수강년도	2020
해당학기	1학기
전공	전공
총합	A
성적	A
분위기	B
강의	A
내용	수업이 너무 재미있고 지루하지 않았어요!

목록

수정

삭제

좋아요

Copyright © 2020 HeeSung All Rights Reserved.  
Address. 서울특별시 마포구 망원동  
Tel. 010 - 2282 - 4338

- 좋아요 버튼은 한 게시글 당 한 번만 누를 수 있도록 구현


- 본인의 게시글은 수정 및 삭제가 가능



# 게시판 글 열람

JSP 게시판 웹 사이트

java/Pos 관리 프로그램.pptx at / x | +

localhost:9000/EvaluationofLectures/view\_index.jsp?evaluationID=1

☆ 

  Positive [Basic Ja... [읽게 배우는 JSP... [AWT/SWING] Men... 포스기 만들기 part1 Apache Commons... hhhhuu1462

강의 평가 웹 사이트

메인

게시판

공지사항

회원관리

게시판

글 제목	너무 재밌어요
작성자	aa
작성일자	2020-07-27 16시03분
강의명	너무 재밌어요
교수명	정현희
수강년도	2020
해당학기	1학기
전공	전공
종합	A
성적	A
분위기	B
강의	A
내용	수업이 너무 재미있고 지루하지 않았어요!

목록

좋아요

신고

Copyright © 2020 HeeSung All Rights Reserved.  
Address. 서울특별시 마포구 망원동  
Tel. 010 - 2282 - 4338

# 게시판 조건 검색

전체 ▾

최신순 ▾

내용을 입력하세요.

검색

번호	제목	작성자	작성일	추천
3	Spring 스프링은 어려워요ㅠㅠ	aa	2020-07-27 23시56분	0
2	jsp 너무 꿀잼~	aa	2020-07-27 23시56분	0
1	자바 수업 너무 재미있어요~	aa	2020-07-27 23시55분	0

글쓰기

전체 ▾

최신순 ▾

자바

검색

번호	제목	작성자	작성일	추천
1	자바 수업 너무 재미있어요~	aa	2020-07-27 23시55분	0

글쓰기

# 게시판 조건 검색

전체 ▾

추천순 ▾

내용을 입력하세요.

검색

번호	제목	작성자	작성일	추천
2	jsp 너무 골쟁~	aa	2020-07-27 23시56분	2
1	자바 수업 너무 재미있어요~	aa	2020-07-27 23시55분	1
3	Spring 스프링은 어려워요 ㅠㅠ	aa	2020-07-27 23시56분	0

글쓰기

# 07.

## Back-End

---

---

---

회원가입

회원가입\_매니저 승인

로그인

페이징 처리

게시판

게시판 글 삭제 및 수정

게시판 글 검색

게시판 글 추천

신고 글 작성

SMS 전송

회원 정보 삭제

# 회원가입

### 회원가입

남자

여자

휴대폰 번호('-' 빼고 입력)

회원가입

```
int userNo = 0;
String userID = null;
String userPassword = null;
String userEmail = null;
String userName = null;
String userGender = null;
String phoneNumber = null;

if (request.getParameter("userNo") != null) {
    userNo = Integer.parseInt(request.getParameter("userNo"));
}
if (request.getParameter("userID") != null) {
    userID = (String) request.getParameter("userID");
}
if (request.getParameter("userPassword") != null) {
    userPassword = (String) request.getParameter("userPassword");
}
if (request.getParameter("userName") != null) {
    userName = (String) request.getParameter("userName");
}
if (request.getParameter("userEmail") != null) {
    userEmail = (String) request.getParameter("userEmail");
}
if (request.getParameter("phoneNumber") != null) {
    phoneNumber = (String) request.getParameter("phoneNumber");
}
userGender = (String) request.getParameter("userGender");
```

회원정보 입력 후 회원가입 버튼을 누르게 되면 Action 페이지로 넘어가 입력값에 대한 파라미터를 얻는다.

# 회원가입

// 사용자의 정보를 입력받아 회원가입

```
public int join(String userID, String userPassword, String userName, String userGender, String userEmail, String phoneNumber, String userEmailHash) {  
    String sql = "insert ignore into user values (?, ?, ?, ?, ?, ?, ?, ?, ?)";
```

```
    Connection conn = null;  
    PreparedStatement pstmt = null;
```

```
    try {  
        conn = DatabaseUtil.getConnection();  
        pstmt = conn.prepareStatement(sql);  
        pstmt.setInt(1, getNext());  
        pstmt.setString(2, userID);  
        pstmt.setString(3, userPassword);  
        pstmt.setString(4, userName);  
        pstmt.setString(5, userGender);  
        pstmt.setString(6, userEmail);  
        pstmt.setString(7, phoneNumber);  
        pstmt.setString(8, userEmailHash);  
        pstmt.setBoolean(9, false);  
        pstmt.setInt(10, 0);  
        return pstmt.executeUpdate();  
    } catch (Exception e) {  
        e.printStackTrace();  
    } finally {  
        try {if(conn != null) conn.close();} catch (Exception e) { e.printStackTrace(); }  
        try {if(pstmt != null) pstmt.close();} catch (Exception e) { e.printStackTrace(); }  
    }  
    return -1; // 회원가입 실패  
}
```

입력값이 하나라도 비었을 경우  
alert()로 오류 메시지 출력 후  
전 페이지로 이동

```
    if (userID == null || userPassword == null || userEmail == null || userName == null || phoneNumber == null || userID.equals("") || userPassword.equals("")  
        || userEmail.equals("") || userName.equals("") || phoneNumber.equals("")) {  
        PrintWriter script = response.getWriter();  
        script.println("<script>");  
        script.println("alert('입력사항을 다시 한 번 확인해 주세요.');"");  
        script.println("history.back();"");  
        script.println("</script>");  
        script.close();  
    } else {
```

```
        UserDao userDao = new UserDao();  
        int result = userDao.join(userID, userPassword, userName, userGender, userEmail, phoneNumber, SHA256.getSHA256(userEmail));
```


```
        if (result == -1) {  
            PrintWriter script = response.getWriter();  
            script.println("<script>");  
            script.println("alert('이미 존재하는 아이디입니다.');"");  
            script.println("history.back();"");  
            script.println("</script>");  
            script.close();  
        } else {  
            session.setAttribute("userID", userID);  
            PrintWriter script = response.getWriter();  
            script.println("<script>");  
            script.println("location.href='emailSendAction.jsp'");  
            script.println("</script>");  
            script.close();  
        }  
    }
```

모든 파라미터를 얻은 경우  
join() 함수를 통해 DB에 회원  
정보 저장 후 매니저 승인요청  
창으로 전달

# 회원가입\_매니저 승인

관리자의 승인 후 문자 발송해 드리겠습니다.

확인



3	이희성	hhhhuu1462	0	0	0건	요청	SMS	삭제
---	-----	------------	---	---	----	----	-----	----



3	이희성	hhhhuu1462	0	0	0건	승인완료	SMS	삭제
---	-----	------------	---	---	----	------	-----	----



# 로그인

로그인 화면

아이디

비밀번호

로그인

로그인 시 ID와 PW를 입력  
후 버튼을 누르게 되면  
Action페이지로 이동

```
UserDAO userDAO = new UserDAO();  
int result = userDAO.login(userID, userPassword);
```

```
if (result == 1) {  
    session.setAttribute("userID", userID);  
    PrintWriter script = response.getWriter();  
    script.println("<script>");  
    script.println("location.href = 'main.jsp'");  
    script.println("</script>");  
    script.close();  
    return;  
}  
else if (result == 0) {  
    PrintWriter script = response.getWriter();  
    script.println("<script>");  
    script.println("alert('비밀번호가 틀립니다.');"");  
    script.println("history.back();"");  
    script.println("</script>");  
    script.close();  
    return;  
}  
else if (result == -1) {  
    PrintWriter script = response.getWriter();  
    script.println("<script>");  
    script.println("alert('존재하지 않는 아이디입니다.');"");  
    script.println("history.back();"");  
    script.println("</script>");  
    script.close();  
    return;  
}  
else if (result == -2) {  
    PrintWriter script = response.getWriter();  
    script.println("<script>");  
    script.println("alert('데이터베이스 오류가 발생했습니다.');"");  
    script.println("history.back();"");  
    script.println("</script>");  
    script.close();  
    return;  
}
```

// 아이디와 비밀번호를 받아 로그인 시도

```
public int login(String userID, String userPassword) {  
    String sql = "select userPassword from user where userID = ?";  
  
    Connection conn = null;  
    PreparedStatement pstmt = null;  
    ResultSet rs = null;  
  
    try {  
        conn = DatabaseUtil.getConnection();  
        pstmt = conn.prepareStatement(sql);  
        pstmt.setString(1, userID);  
        rs = pstmt.executeQuery();  
        if(rs.next()) {  
            if(rs.getString(1).equals(userPassword)) {  
                return 1; // 로그인 성공  
            } else {  
                return 0; // 비밀번호 틀림  
            }  
        }  
        return -1; // 아이디 없음  
    } catch (Exception e) {  
        e.printStackTrace();  
    } finally {  
        try {if(conn != null) conn.close();} catch (Exception e) { e.printStackTrace(); }  
        try {if(pstmt != null) pstmt.close();} catch (Exception e) { e.printStackTrace(); }  
        try {if(rs != null) rs.close();} catch (Exception e) { e.printStackTrace(); }  
    }  
    return -2; //db 오류  
}
```

회원가입 시 등록 한 ID를 통해  
PASSWORD를 검색하고 입력한 정  
보와 일치하는지 판단합니다.

**PRIMARY KEY : 회원ID**

# 페이징 처리

번호	제목	작성자	작성일	추천
6	오라클db	a	2020-07-28 00시27분	0
5	db 재밌어요~!!	a	2020-07-28 00시26분	0
4	자바관련	a	2020-07-28 00시25분	0
3	Spring 스프링은 어려워요 ㅠㅠ	aa	2020-07-27 23시56분	0
2	jsp 너무 꿀잼~	aa	2020-07-27 23시56분	2

[Next →](#)[글쓰기](#)

번호	제목	작성자	작성일	추천
1	자바수업 너무 재미있어요~	aa	2020-07-27 23시55분	1

[← Previous](#)[글쓰기](#)

# 페이징 처리

```
int pageNumber = 0;
if (request.getParameter("pageNumber") != null) {
    try {
        pageNumber = Integer.parseInt(request.getParameter("pageNumber"));
    } catch (Exception e) {
        System.out.println("검색 페이지 번호 오류");
    }
}
```

```
evaluationList = new EvaluationDAO().getSort(lectureDivide, searchType, search, pageNumber);
if(evaluationList != null)
    for(int i = 0; i < evaluationList.size(); i++) {
        if(i == 5) break;
        EvaluationDTO evaluation = evaluationList.get(i);
```

```
if(! (pageNumber == 0)) {
    <li><a style="background-color:AliceBlue" href= "/index.jsp?lectureDivide=<%=URLEncoder.encode(lectureDivide, "UTF-8")%>
    &searchType=<%=URLEncoder.encode(searchType, "UTF-8")%> &search=<%=URLEncoder.encode(search, "UTF-8")%> &pageNumber=<%=pageNumber - 1%> "> ← Previous</a></li>

    if(!(evaluationList.size() < 6)) {
        <li><a style="background-color:AliceBlue" href= "/index.jsp?lectureDivide=<%=URLEncoder.encode(lectureDivide, "UTF-8")%>
        &searchType=<%=URLEncoder.encode(searchType, "UTF-8")%> &search=<%=URLEncoder.encode(search, "UTF-8")%> &pageNumber=<%=pageNumber + 1%> "> Next →</a></li>
    }
}
```

```
if(searchType.equals("최신순")) {
    sql = "SELECT * FROM EVALUATION WHERE lectureDivide LIKE ? AND CONCAT(lectureName, professorName, evaluationTitle, evaluationContent) LIKE ? ORDER BY evaluationID DESC LIMIT " + pageNumber * 5 + ", " + 5;
} else if(searchType.equals("추천순")) {
    sql = "SELECT * FROM EVALUATION WHERE lectureDivide LIKE ? AND CONCAT(lectureName, professorName, evaluationTitle, evaluationContent) LIKE ? ORDER BY likeCount DESC LIMIT " + pageNumber * 5 + ", " + 5;
}
```

pageNumber가 0이 아닐 경우 <-Previous 버튼 생성

pageNumber가 6 이상일 경우 Next-> 버튼 생성

페이지 별 보이는 게시글 번호는 **limit(각 페이지\*5, 5) : (각 페이지\*5)+1 부터 5개의 게시글 표시**

# 게시판

평가 등록

강의명

교수명

수강년도

해당학기

전공선택

제목

내용입력

종합

성적

분위기

강의

등록

취소

```
int result = evaluationDAO.write(userID, evaluation.getLectureName(), evaluation.getProfessorName(), evaluation.getLectureYear(), evaluation.getSemesterDivide(), evaluation.getLectureDivide(),  
evaluation.getEvaluationTitle(), evaluation.getEvaluationContent(), evaluation.getTotalScore(), evaluation.getCreditScore(), evaluation.getComfortableScore(), evaluation.getLectureScore());
```

# 게시판

```
// 글 쓰기
public int write(String userID, String lectureName, String professorName, String lectureYear, String semesterDivide, String lectureDivide,
String evaluationTitle, String evaluationContent, String totalScore, String creditScore, String comfortableScore, String lectureScore) {
    String sql = "insert into evaluation values (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)";

    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;

    try {
        conn = DatabaseUtil.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setInt(1, getNext());
        pstmt.setString(2, userID.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(3, lectureName.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(4, professorName.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(5, lectureYear.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(6, semesterDivide.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(7, lectureDivide.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(8, evaluationTitle.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(9, getDate());
        pstmt.setString(10, evaluationContent.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(11, totalScore.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(12, creditScore.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(13, comfortableScore.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(14, lectureScore.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setInt(15, 0);
        pstmt.setInt(16, 1);
        return pstmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {if(conn != null) conn.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(pstmt != null) pstmt.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(rs != null) rs.close();} catch (Exception e) { e.printStackTrace(); }
    }
    return -1; //db 오류
}
```

XSS 해킹 방어를 위한  
replaceAll()함수 사용 :  
SQL문 삽입과 함께 웹 상에서  
가장 초보 단계의 공격방법으로  
데이터가 전달 될 Input 박스  
안에 스크립트를 작성하여 전달  
함으로서 의도치 않는 행동을  
실행시키거나 쿠키 같은 정보를  
탈취하게 됩니다.

그렇기 때문에 스크립트 요소  
를 일반 문자로 변환하는 코드  
를 작성하였습니다.

# 게시판 글 삭제 및 수정

```
public int update(int evaluationID, String lectureName, String professorName, String lectureYear, String semesterDivide, String lectureDivide,
String evaluationTitle, String evaluationContent, String totalScore, String creditScore, String comfortableScore, String lectureScore) {
    String SQL = "update evaluation set lectureName=?, professorName=?, lectureYear=?, semesterDivide=?, lectureDivide=?, evaluationTitle=?"
        + " , evaluationContent=?, totalScore=?, creditScore=?, comfortableScore=?, lectureScore=? where evaluationID=?";

    Connection conn = null;
    PreparedStatement pstmt = null;

    try {
        conn = DatabaseUtil.getConnection();
        pstmt = conn.prepareStatement(SQL);
        pstmt.setString(1, lectureName.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(2, professorName.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(3, lectureYear.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(4, semesterDivide.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(5, lectureDivide.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(6, evaluationTitle.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(7, evaluationContent.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(8, totalScore.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(9, creditScore.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(10, comfortableScore.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setString(11, lectureScore.replaceAll("<", "&lt;").replaceAll(">", "&gt;").replaceAll("WrWn", "<br>"));
        pstmt.setInt(12, evaluationID);
        return pstmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    }
    return -1;
}
```

게시글 삭제는 해당 게시글의 번호를 통해 삭제 하였습니다.

SQL : DELETE문 사용 (PRIMARY KEY : 게시글 번호)

게시글 수정도 마찬가지로 XSS 해킹 방어를 위해 스크립트 요소를 문자형으로 변경 시켰습니다.

SQL : UPDATE문 사용

```
// 게시글 삭제
public int delete(int evaluationID) {
    String sql = "delete from evaluation where evaluationID = ?";

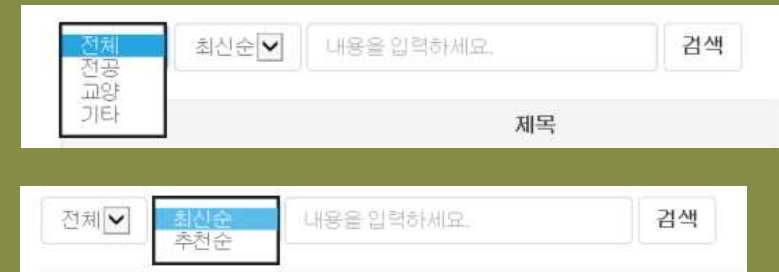
    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;

    try {
        conn = DatabaseUtil.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setInt(1, evaluationID);
        return pstmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {if(conn != null) conn.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(pstmt != null) pstmt.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(rs != null) rs.close();} catch (Exception e) { e.printStackTrace(); }
    }
    return -1; // db 오류
}
```



# 게시판 글 검색

```
<select name="lectureDivide" class="form-control mx-1 mt-2">
  <option value="전체">전체</option>
  <option value="전공"><%if (lectureDivide.equals("전공")) out.println("selected");%>전공</option>
  <option value="교양"><%if (lectureDivide.equals("교양")) out.println("selected");%>교양</option>
  <option value="기타"><%if (lectureDivide.equals("기타")) out.println("selected");%>기타</option>
</select>
<select name="searchType" class="form-control mx-1 mt-2">
  <option value="최신순">최신순</option>
  <option value="추천순"><%if (lectureDivide.equals("추천순")) out.println("selected");%>추천순</option>
</select>
```



```
if(searchType.equals("최신순")) {
  sql = "SELECT * FROM EVALUATION WHERE lectureDivide LIKE ? AND CONCAT(lectureName, professorName, evaluationTitle, evaluationContent) LIKE ? ORDER BY evaluationID DESC LIMIT " + pageNumber * 5 + ", " + 5;
} else if(searchType.equals("추천순")) {
  sql = "SELECT * FROM EVALUATION WHERE lectureDivide LIKE ? AND CONCAT(lectureName, professorName, evaluationTitle, evaluationContent) LIKE ? ORDER BY likeCount DESC LIMIT " + pageNumber * 5 + ", " + 5;
}
```

검색 방법 : 최신순 -> 선택한 강의구분과 (강의명, 교수명, 게시글 제목, 내용)을 연결한 문자열 중 입력한 값이 들어 있다면 게시글 번호를 내림차순으로 정렬한다.  
(한 페이지 당 5개의 게시글)

검색 방법 : 추천순 -> 선택한 강의구분과 (강의명, 교수명, 게시글 제목, 내용)을 연결한 문자열 중 입력한 값이 들어 있다면 추천순 기준 내림차순으로 정렬한다.  
(한 페이지 당 5개의 게시글)

# 게시판 글 추천

```
EvaluationDAO evaluationDAO = new EvaluationDAO();
LikeyDAO likeyDAO = new LikeyDAO();

int result = likeyDAO.like(userID, evaluationID, getClientIP(request));

if (result == 1) {

    result = evaluationDAO.like(evaluationID);

    if (result == 1) {
        PrintWriter script = response.getWriter();
        script.println("<script>");
        script.println("alert('추천이 완료되었습니다.');"");
        script.println("location.href='index.jsp'");
        script.println("</script>");
        script.close();
        return;
    } else {
        PrintWriter script = response.getWriter();
        script.println("<script>");
        script.println("alert('데이터베이스 오류가 발생했습니다.');"");
        script.println("history.back();"");
        script.println("</script>");
        script.close();
        return;
    }
} else {
    PrintWriter script = response.getWriter();
    script.println("<script>");
    script.println("alert('이미 추천을 누른 글입니다.');"");
    script.println("history.back();"");
    script.println("</script>");
    script.close();
    return;
}
```

좋아요 버튼을 누르게 되면 likeyDAO() 함수를 통해 이전에 회원이 해당 게시판에 좋아요 버튼을 눌렀는지 확인 후 result 값을 반환한다.  
(PRIMARY KEY : 회원ID, 게시글 번호)  
이후 게시글 함수에서 게시글 번호를 통해 좋아요 개수를 1씩 늘리는 함수를 실행한다.  
(PRIMARY KEY : 게시글 번호)

```
public int like(String userID, String evaluationID, String userIP) {
    String sql = "insert into likey values (?, ?, ?)";

    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;

    try {
        conn = DatabaseUtil.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, userID);
        pstmt.setString(2, evaluationID);
        pstmt.setString(3, userIP);
        return pstmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {if(conn != null) conn.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(pstmt != null) pstmt.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(rs != null) rs.close();} catch (Exception e) { e.printStackTrace(); }
    }
    return -1; // 추천 중복 오류
}
```

```
public static String getClientIP(HttpServletRequest request) {
    String ip = request.getHeader("X-FORWARDED-FOR");
    if (ip == null || ip.length() == 0) {
        ip = request.getHeader("Proxy-Client-IP");
    }
    if (ip == null || ip.length() == 0) {
        ip = request.getHeader("WL-Proxy-Client-IP");
    }
    if (ip == null || ip.length() == 0) {
        ip = request.getRemoteAddr();
    }
    return ip;
}
```

http server에 요청한 client의 IP를 식별하기 위한 표준

```
// 좋아요
public int like(String evaluationID) {
    String sql = "update evaluation set likeCount = likeCount + 1 where evaluationID = ?";

    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;

    try {
        conn = DatabaseUtil.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setInt(1, Integer.parseInt(evaluationID));
        return pstmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {if(conn != null) conn.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(pstmt != null) pstmt.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(rs != null) rs.close();} catch (Exception e) { e.printStackTrace(); }
    }
    return -1; // db 오류
}
```



# 신고 글 작성

신고하기

신고 대상  
aa

신고 제목

신고 내용

취소 신고하기

수업이 너무 재미있고 지루하지 않았어요!

좋아요 신고

Copyright © 2020 HeeSung All Rights Reserved.  
Address: 서울특별시 마포구 월드컵

신고 버튼을 클릭 시 신고 글 작성 모달 창이 띄워지게 구현하였습니다.

```
<div class="modal fade" id="reportModal" tabindex="-1" role="dialog" aria-labelledby="modal" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="modal"> 신고하기 </h5>
        <button type="button" class="close" data-dismiss="modal" aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
        <form method="post" action="/reportAction_doubleCheck.jsp">
          <div class="form-group">
            <label>신고 대상</label>
            <input type="text" name="reportTarget" class="form-control" maxlength="20" value="<%=evaluationDTO.getUserID()%>" readonly>
          </div>
          <div class="form-group">
            <label>신고 제목</label>
            <input type="text" name="reportTitle" class="form-control" maxlength="20">
          </div>
          <div class="form-group">
            <label>신고 내용</label>
            <textarea name="reportContent" class="form-control" maxlength="2048" style="height: 180px;"></textarea>
          </div>
          <div class="modal-footer">
            <button type="button" class="btn btn-secondary" data-dismiss="modal">취소</button>
            <button type="submit" onclick="userID=<%=userID %>" class="btn btn-danger">신고하기</button>
          </div>
        </form>
      </div>
    </div>
  </div>
</div>
```

# 신고 글 작성

```
UserDAO userDao = new UserDAO();
ReportDAO reportDAO = new ReportDAO();

int result = reportDAO.report(userID, getClientIP(request));

if (result == 1) {
    result = userDao.report(reportTarget);
    if (result == 1) {
        String host = "http://localhost:9000/Evaluationoflectures/";
        String from = userDao.getUserEmail(userID);
        String to = "hhuu1462@gmail.com";
        String subject = "강의평가 사이트에서 접수된 신고 메일입니다.";
        String content = "신고자 : " + userID + "<br> 신고대상 : " + reportTarget + "<br> 제목 : " + reportTitle + "<br> 내용 : " + reportContent;

        // SMTP에 접속하기 위한 정보를 기입합니다.
        Properties p = new Properties();
        p.put("mail.smtp.user", from);
        p.put("mail.smtp.host", "smtp.googlemail.com");
        p.put("mail.smtp.port", "465");
        p.put("mail.smtp.starttls.enable", "true");
        p.put("mail.smtp.auth", "true");
        p.put("mail.smtp.debug", "true");
        p.put("mail.smtp.socketFactory.port", "465");
        p.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");
        p.put("mail.smtp.socketFactory.fallback", "false");
```

- 신고하기 버튼 클릭 시 신고자ID와 해당 IP를 전달하여 이전에 신고한 적이 있는지 판단
- 신고가 되었다면 매니저의 메일로 신고 접수 메일 발송
- 신고 함수를 통해 신고 당한 회원의 신고 누적 횟수 증가

```
public int report(String reportTarget) {
    String sql = "update user set reportCount = reportCount + 1 where userID = ?";

    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;

    try {
        conn = DatabaseUtil.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, reportTarget);
        return pstmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {if(conn != null) conn.close();} catch (Exception e) {e.printStackTrace();}
        try {if(pstmt != null) pstmt.close();} catch (Exception e) {e.printStackTrace();}
        try {if(rs != null) rs.close();} catch (Exception e) {e.printStackTrace();}
    }
    return -1; // db 오류
}
```

```
public class ReportDAO {
    public int report(String userID, String userIP) {
        String sql = "insert into report values (?, ?)";

        Connection conn = null;
        PreparedStatement pstmt = null;
        ResultSet rs = null;

        try {
            conn = DatabaseUtil.getConnection();
            pstmt = conn.prepareStatement(sql);
            pstmt.setString(1, userID);
            pstmt.setString(2, userIP);
            return pstmt.executeUpdate();
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {if(conn != null) conn.close();} catch (Exception e) {e.printStackTrace();}
            try {if(pstmt != null) pstmt.close();} catch (Exception e) {e.printStackTrace();}
            try {if(rs != null) rs.close();} catch (Exception e) {e.printStackTrace();}
        }
        return -1;
    }
}
```

```
try {
    Authenticator auth = new Gmail();
    Session ses = Session.getInstance(p, auth);
    ses.setDebug(true);
    MimeMessage msg = new MimeMessage(ses);
    msg.setSubject(subject);
    Address fromAddr = new InternetAddress(from);
    msg.setFrom(fromAddr);
    Address toAddr = new InternetAddress(to);
    msg.addRecipient(Message.RecipientType.TO, toAddr);
    msg.setContent(content, "text/html;charset=UTF-8");
    Transport.send(msg);
} catch (Exception e) {
    e.printStackTrace();
    PrintWriter script = response.getWriter();
    script.println("<script>");
    script.println("alert('오류가 발생했습니다.');");
    script.println("history.back();");
    script.println("</script>");
    script.close();
    return;
}

PrintWriter script = response.getWriter();
script.println("<script>");
script.println("alert('정상적으로 신고되었습니다.');");
script.println("history.back();");
script.println("</script>");
script.close();
return;
} else {
    PrintWriter script = response.getWriter();
    script.println("<script>");
    script.println("alert('데이터베이스 오류가 발생했습니다.');");
    script.println("history.back();");
    script.println("</script>");
    script.close();
    return;
}

} else {
    PrintWriter script = response.getWriter();
    script.println("<script>");
    script.println("alert('이미 신고 한 글입니다.');");
    script.println("history.back();");
    script.println("</script>");
    script.close();
    return;
}
}
```

# 신고 글 작성

```
<%@page import="java.io.PrintWriter"%>
<%@page import="javax.mail.Transport"%>
<%@page import="javax.mail.Message"%>
<%@page import="javax.mail.Address"%>
<%@page import="javax.mail.internet.InternetAddress"%>
<%@page import="javax.mail.internet.MimeMessage"%>
<%@page import="javax.mail.Session"%>
<%@page import="javax.mail.Authenticator"%>
<%@page import="java.util.Properties"%>
<%@page import="util.SHA256"%>
<%@page import="util.Gmail"%>
```

**SMTP** : 간이 우편 전송 프로토콜로써 간이 우편 전송 프로토콜  
**Properties 객체** : SMTP에 접속하기 위한 정보를 담는 객체  
**Authenticator** : 구글의 모바일 애플리케이션 사용자들을 인증  
하기 위해 사용합니다.

사용자 이름과 비밀번호 외에 지정해야 하는 6~8자리의 일회성  
비밀번호를 제공하며 이를 통해 구글 서비스와 다른 사이트에 로  
그인할 수 있습니다.

```
String host = "http://localhost:9000/EvaluationofLectures/";
String from = userDao.getUserEmail(userID);
String to = "hhhhuu1462@gmail.com";
String subject = "강의평가 사이트에서 접수된 신고 메일입니다.";
String content = "신고자 : " + userID + "<br> 신고대상 : " + reportTarget + "<br>제목 : " + reportTitle + "<br>내용 : " + reportContent;
```

```
// SMTP에 접속하기 위한 정보를 기입합니다.
Properties p = new Properties();
p.put("mail.smtp.user", from);
p.put("mail.smtp.host", "smtp.googlemail.com");
p.put("mail.smtp.port", "465");
p.put("mail.smtp.starttls.enable", "true");
p.put("mail.smtp.auth", "true");
p.put("mail.smtp.debug", "true");
p.put("mail.smtp.socketFactory.port", "465");
p.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");
p.put("mail.smtp.socketFactory.fallback", "false");
```

```
try {
    Authenticator auth = new Gmail();
    Session ses = Session.getInstance(p, auth);
    ses.setDebug(true);
    MimeMessage msg = new MimeMessage(ses);
    msg.setSubject(subject);
    Address fromAddr = new InternetAddress(from);
    msg.setFrom(fromAddr);
    Address toAddr = new InternetAddress(to);
    msg.addRecipient(Message.RecipientType.TO, toAddr);
    msg.setContent(content, "text/html;charset=UTF-8");
    Transport.send(msg);
} catch (Exception e) {
    e.printStackTrace();
    PrintWriter script = response.getWriter();
    script.println("<script>");
    script.println("alert('오류가 발생했습니다.');
```

강의평가 사이트에서 접수된 신고 메일입니다. > 받은편지함 X



hhhhuu1462@gmail.com

나에게 ▾

신고자 : hhhhhu1462

신고대상 : aa

제목 : 욕설이 심해요

내용 : 다른 사람들이 봤을때 욕설로 인한 거부감이 느껴지는 글입니다

← 답장

➡ 전달

# SMS 작성

4	이희성	hhhhuu1462	0	0	0건	승인완료	SMS	삭제
---	-----	------------	---	---	----	------	-----	----

SMS

01022824338

내용입력

발송 취소

```
<form method="post" name="smsForm" action="smsend.jsp">
  <input type="hidden" name="action" value="go">
  <div class="container">
    <div class="row">
      <table class="table table-striped" style="text-align: center; table-layout: fixed; border: 1px solid #dddddd;">
        <thead>
          <tr>
            <th colspan="4" style="background-color: #eeeeee; text-align: center;">SMS</th>
          </tr>
        </thead>
        <tbody>
          <tr>
            <td colspan="4">
              <input placeholder="받는 번호" type="text" name="rphone" class="form-control" maxlength="30" value="0<%=phoneNumber %>">
            </td>
          </tr>
          <tr>
            <td colspan="4">
              <textarea placeholder="내용입력" name="msg" class="form-control" maxlength="2048" style="height: 180px;"></textarea>
            </td>
          </tr>
        </tbody>
      </table>
      <div style="float: right">
        <input onclick="location='memberManagement.jsp'" type="button" class="btn btn-danger" value="취소" name="cancel" style="width: 50; float: right;">
        <input type="submit" class="btn btn-primary" value="발송" style="width: 50; float: right; margin-right: 4px;">
      </div>
    </div>
  </div>
  <input type="hidden" name="sphone1" value="010">
  <input type="hidden" name="sphone2" value="2282">
  <input type="hidden" name="sphone3" value="4338">
</form>
```

Hidden type의 경우 화면상으로는 보이지 않지만 해당 액션 페이지에 파라미터 값을 전달한다.



# SMS 작성

```
if (action.equals("go"))
```

```
String sms_url = "";  
sms_url = "https://sslsmc.cafe24.com/sms_sender.php"; // SMS 전송요청 URL  
String user_id = base64Encode("hhhuu1462"); // SMS아이디  
String secure = base64Encode("539756aaf0187a654dbc568e905e35f5"); // 인증키  
String msg = base64Encode(nullcheck(request.getParameter("msg"), ""));  
String rphone = base64Encode(nullcheck(request.getParameter("rphone"), ""));  
String sphone1 = base64Encode(nullcheck(request.getParameter("sphone1"), ""));  
String sphone2 = base64Encode(nullcheck(request.getParameter("sphone2"), ""));  
String sphone3 = base64Encode(nullcheck(request.getParameter("sphone3"), ""));  
String rdate = base64Encode(nullcheck(request.getParameter("rdate"), ""));  
String rtime = base64Encode(nullcheck(request.getParameter("rtime"), ""));
```

Action명이 go인 form의 정보가 전달이 되었다면  
SMS 전송요청 URL과 연결을 하고 해당 URL의 ID와 인증키를 전달한다.  
받아 온 파라미터 값들을 변수로 지정하여 데이터를 맵핑한다.

```
// 데이터 맵핑 변수 정의  
String arrKey[] = new String[] { "user_id", "secure", "msg", "rphone", "sphone1", "sphone2", "sphone3", "rdate",  
    "rtime", "mode", "testflag", "destination", "repeatFlag", "repeatNum", "repeatTime", "smsType", "subject" };  
String valKey[] = new String[arrKey.length];  
valKey[0] = user_id;  
valKey[1] = secure;  
valKey[2] = msg;  
valKey[3] = rphone;  
valKey[4] = sphone1;  
valKey[5] = sphone2;  
valKey[6] = sphone3;  
valKey[7] = rdate;  
valKey[8] = rtime;  
valKey[9] = mode;  
valKey[10] = testflag;  
valKey[11] = destination;  
valKey[12] = repeatFlag;  
valKey[13] = repeatNum;  
valKey[14] = repeatTime;  
valKey[15] = smsType;  
valKey[16] = subject;
```

[Web발신]  
관리자에 의해 회원 정보가 삭제되었  
습니다

# 회원정보 삭제

3	aaa	aaa	0	0	0건	승인완료	SMS	삭제
---	-----	-----	---	---	----	------	-----	----

<td><input type="button" value="삭제" onclick="location.href='deleteUser.jsp?userID=<%=list.get(i).getUserId()%>'"/></td>

```
EvaluationDAO evaluationDAO = new EvaluationDAO();  
LikeyDAO likeyDAO = new LikeyDAO();  
ReportDAO reportDAO = new ReportDAO();  
UserDAO userDAO = new UserDAO();
```

```
int result_evaluation = evaluationDAO.delete_user(userID);  
int result_likey = likeyDAO.delete_user(userID);  
int result_report = reportDAO.delete_user(userID);  
int result_user = userDAO.delete_user(userID);
```

```
if (result_evaluation == -1 && result_likey == -1 && result_report == -1 && result_user == -1 ) {  
    PrintWriter script = response.getWriter();  
    script.println("<script>");  
    script.println("alert('회원정보 삭제에 실패했습니다.');"");  
    script.println("history.back()");  
    script.println("</script>");  
} else {  
    PrintWriter script = response.getWriter();  
    script.println("<script>");  
    script.println("alert('회원정보가 삭제되었습니다.');"");  
    script.println("location.href = 'memberManagement.jsp'");  
    script.println("</script>");  
}
```

매니저의 권한으로 삭제 버튼을 누를 시 회원 삭제 함수 실행  
게시글, 추천, 신고, 회원 데이터베이스에 있는 해당 회원의 자료를 모두 삭제한다.

# 회원 정보 삭제

```
public int delete_user(String userID) {
    String sql = "delete from evaluation where userID = ?";

    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;

    try {
        conn = DatabaseUtil.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, userID);
        return pstmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {if(conn != null) conn.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(pstmt != null) pstmt.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(rs != null) rs.close();} catch (Exception e) { e.printStackTrace(); }
    }
    return -1; // db 오류
}
```

```
public int delete_user(String userID) {
    String sql = "delete from report where userID = ?";

    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;

    try {
        conn = DatabaseUtil.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, userID);
        return pstmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {if(conn != null) conn.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(pstmt != null) pstmt.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(rs != null) rs.close();} catch (Exception e) { e.printStackTrace(); }
    }
    return -1; // db 오류
}
```

```
public int delete_user(String userID) {
    String sql = "delete from likey where userID = ?";

    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;

    try {
        conn = DatabaseUtil.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, userID);
        return pstmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {if(conn != null) conn.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(pstmt != null) pstmt.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(rs != null) rs.close();} catch (Exception e) { e.printStackTrace(); }
    }
    return -1; // db 오류
}
```

```
public int delete_user(String userID) {
    String sql = "delete from user where userID = ?";

    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;

    try {
        conn = DatabaseUtil.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, userID);
        return pstmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {if(conn != null) conn.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(pstmt != null) pstmt.close();} catch (Exception e) { e.printStackTrace(); }
        try {if(rs != null) rs.close();} catch (Exception e) { e.printStackTrace(); }
    }
    return -1; // db 오류
}
```

# 08.

## 보완 및 후기

---

---

---

### [보완점]

먼저, MVC 모델링을 통한 방식으로 프로그램을 재 구현 해보고 싶습니다. Controller와 View 단이 섞여 있어 코드가 길어졌을 뿐만 아니라 지저분한 면이 상당히 많은 것 같습니다.

후에 Https 및 SSL 암호화 통신에 대해 공부하여 적용시켜 보고 싶습니다.

### [후기]

이번 강의 평가 사이트 프로젝트를 하면서 기존에 사용해보지 못했던 API(google API 등)들을 이용하여 구현 하면서 해당 API에 대한 기술을 이해하기에 어려움이 있었지만 제게 많은 도움이 되었던 것 같습니다.

개인 프로젝트를 통해서 스스로 문제에 대한 해결 능력을 기를 수 있는 능력을 향상 시킬 수 있었던 것 같습니다.