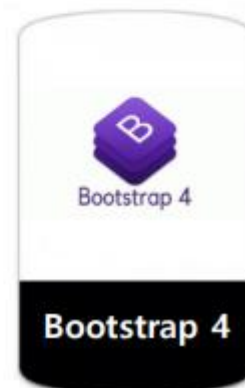


PORTFOLIO



PROJECT

Theme : 신발 쇼핑몰

1. 개요

PERIOD 2022.03.14 ~ 2022.04.15

OBJECT 회원관리- 로그인, 회원가입, 아이디/비밀번호 찾기, 마이페이지, 회원정보 수정, 회원관리 페이지
상품관리- 상품목록, 장바구니, 결제페이지, 관리자 페이지, 상품 상세페이지, 쇼핑몰 메인 페이지
게시판- 1:1문의, 자주 묻는 질문, 매장 위치, 공지사항, 이벤트

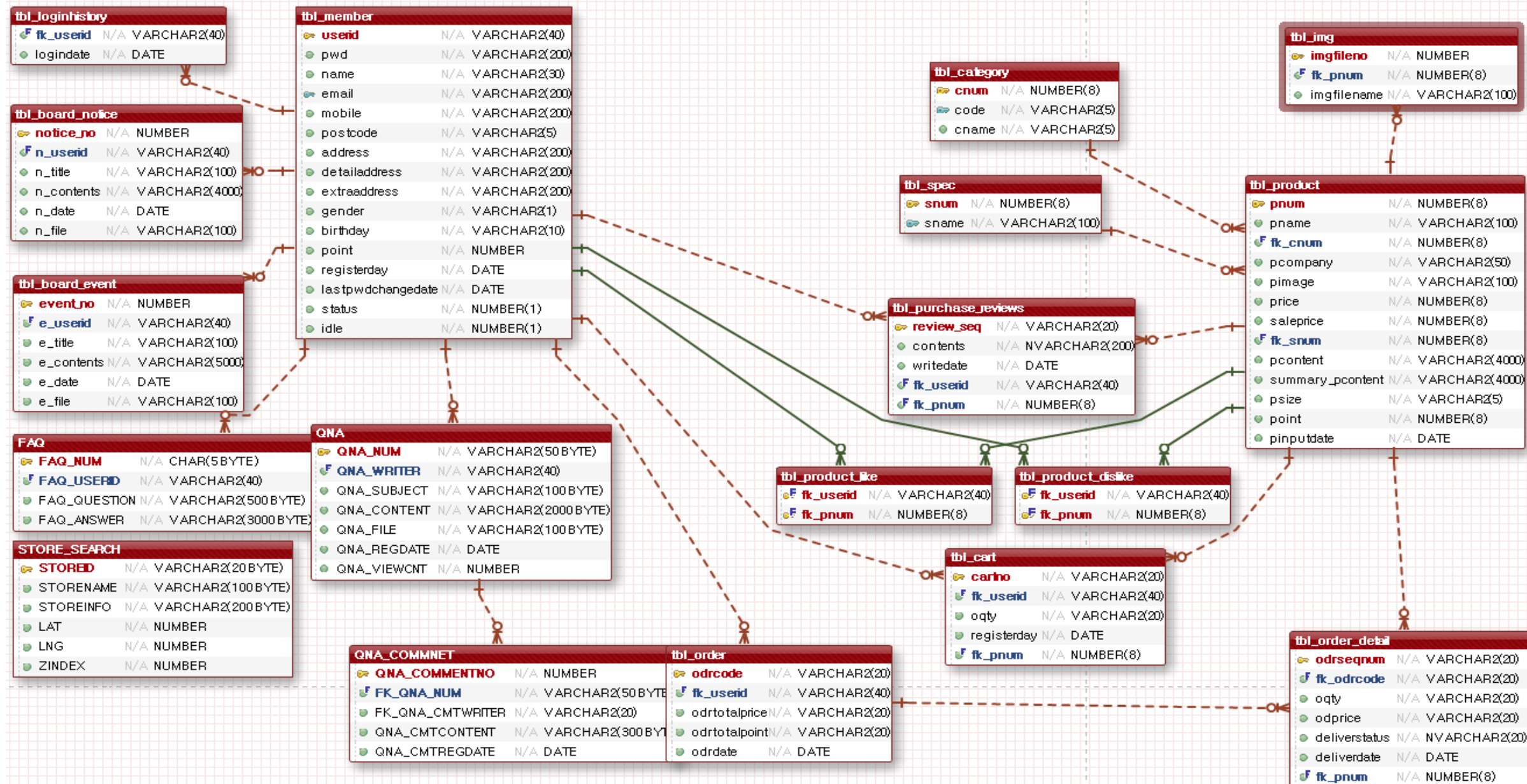
ROLE - 로그인
- 회원가입
- 아이디/비밀번호 찾기

CONTRIBUTION 20% 팀(6명)

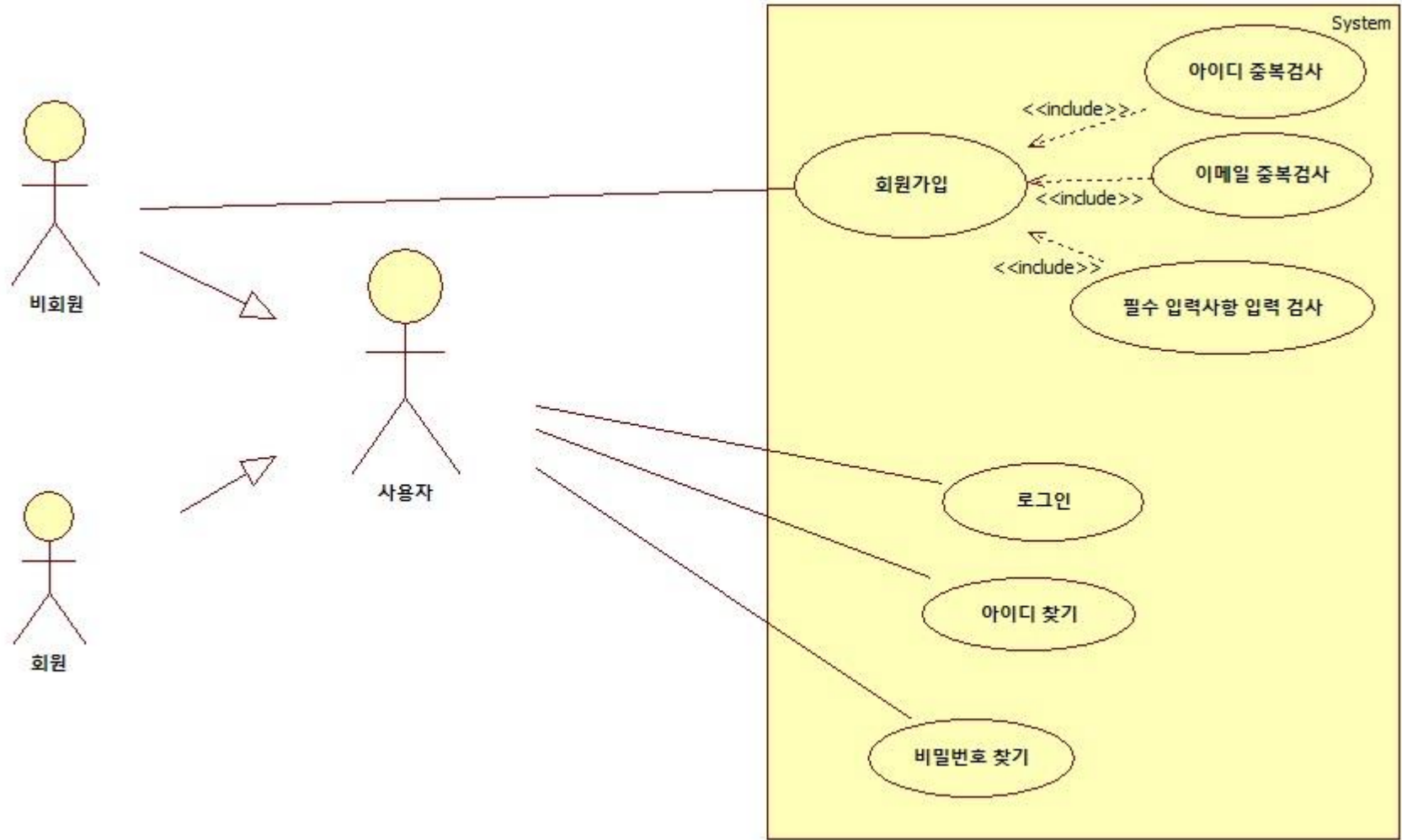
GitHub https://github.com/leejengjun/semi_prj_shoes.git

Impression 실제로 수업을 듣고 이해하는 것과 직접 구현하는 게 많이 다르다는 걸 느꼈습니다.
Github 사용과 기능을 구현하면서 이해가 가지 않는 에러가 발생해서 당황스러웠지만 그만큼 해결을 통해 에러에 대처하는 능력이 점점 발전함을 느껴 더욱 흥미를 느끼게 되었습니다. 또한 팀원들과 서로 도와가며 프로젝트를 만들어가는 과정에서 공부도 되었고 다시 한번 팀워크의 중요성을 깨달았습니다.
함께 열심히 하였던 팀장님과 팀원분들 모두 감사합니다.

2. 담당 기능-eXERD



2. 담당 기능- Usecase Diagram



2. 담당 기능- 회원가입



2. 담당 기능- 회원가입

아이디 * 아이디중복확인 abdd 사용가능합니다.

아이디 * 아이디중복확인 kimkimkim 은 중복된 ID
이므로 사용불가능합니다.

아이디 * 아이디중복확인
아이디는 필수입력 사항입니다.

[controller]

```
InterMemberDAO mdao = new MemberDAO();  
boolean isExist = mdao.idDuplicateCheck(userid);
```

```
JSONObject jsonObj = new JSONObject();  
jsonObj.put("isExist", isExist);
```

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

```
request.setAttribute("json", json);
```

[JSP]

```
if(userid == "") {
```

```
    // 입력하지 않거나 공백일 경우  
    $("table#tblMemberRegister :input").prop("disabled", true);  
    $target.prop("disabled", false);
```

```
    $target.parent().find(".error").show();
```

```
    $target.focus();
```

아이디를 입력하지 않았거나 공백인 경우
에러 메시지를 보여준다.

```
} else {
```

```
    $("table#tblMemberRegister :input").prop("disabled", false);  
    $target.parent().find(".error").hide();
```

[JSP 아이디중복검사 하기]

```
$.ajax({  
    url:"<%= ctxPath%>/member/idDuplicateCheck.shoes",  
    data:{"userid":$("input#userid").val()},  
    // /member/idDuplicateCheck.shoes 로 데이터 전송  
    type:"post",  
    async:true,    // 비동기 처리(기본값)  
    success:function(text){
```

```
        const json = JSON.parse(text);
```

```
        if(json.isExist) {  
            //입력한 $("input#userid").val() 값이 이미 사용중인 경우  
            $("span#idcheckResult").html($("input#userid").val()+" 은 중복된 ID 이므로 사용불가능합니다.").css("color","red");  
            $("input#userid").val("");
```

```
        } else{  
            // 입력한 $("input#userid").val() 값이 DB tbl_member 테이블에 존재하지 않는 경우  
            $("span#idcheckResult").html($("input#userid").val()+" 사용가능합니다.").css("color","green");  
        }
```

```
    },  
    error:function(request, status, error){ // 실패할 경우  
        alert("code: "+request.status+"\n"+"message: "+request.responseText+"\n"+"error: "+error);
```

[MemberDAO]

```
// ID 중복검사  
@Override  
public boolean idDuplicateCheck(String userid) throws SQLException {
```

```
    boolean isExist = false;
```

```
    try {  
        conn = ds.getConnection();
```

```
        String sql = " select * "  
            + " from tbl_member "  
            + " where userid = ? ";
```

```
        pstmt = conn.prepareStatement(sql);  
        pstmt.setString(1, userid);
```

```
        rs = pstmt.executeQuery();
```

```
        isExist = rs.next(); // 행이 있으면(중복된 userid) true,  
                            // 행이 없으면(사용가능한 userid) false
```

tbl_member 테이블에서 userid 가 존재하면
true(아이디 중복)를 return해주고, userid 가
존재하지 않으면 false(사용가능)를 return

2. 담당 기능- 회원가입

이메일 *	<input type="text" value="abc@def.com"/>	이메일중복확인 aaa@aaaa.aaa 은 중복된 email 이므로 사용불가합니다.
이메일 *	<input type="text" value="abab@naver.com"/>	이메일중복확인 abab@naver.com 사용가능합니다.
이메일 *	<input type="text" value="abc@def.com"/>	이메일 형식에 맞지 않습니다.

[JSP]

```
$.ajax({
  url:"<%= ctxPath%>/member/emailDuplicateCheck.shoes",
  data:{"email":$("#input#email").val()},
  type:"post",
  dataType:"json",
```

```
  async:true,    // 비동기 처리(기본값)
  success:function(json){
```

```
    if(json.isExist) {
```

```
      //입력한 $("#input#email").val() 값이 이미 사용중인 경우
```

```
      $("#span#emailCheckResult").html($("#input#email").val()+" 은 중복된 email 이므로 사용불가합니다.").css("color","red");
      $("#input#email").val("");
    }
```

```
  }
  else{
```

```
    // 입력한 $("#input#email").val() 값이 DB tbl_member 테이블에 존재하지 않는 경우
```

```
    $("#span#emailCheckResult").html($("#input#email").val()+" 사용가능합니다.").css("color","green");
  }
```

```
},
```

```
error:function(request, status, error){ // 실패할 경우
```

```
  alert("code: "+request.status+"\n"+"message: "+request.responseText+"\n"+"error: "+error);
```

tbl_member 테이블에서 email이 존재하면 true(이메일 중복)를 return해주고, email이 존재하지 않으면 false(사용가능)를 return

[JSP]

```
// 이메일 정규표현식
```

```
const regExp = new RegExp(/^[0-9a-zA-Z]([-_\.]?[0-9a-zA-Z])*@[0-9a-zA-Z]([-_\.]?[0-9a-zA-Z])*\.[a-zA-Z]{2,3}$/i);
```

```
const bool = regExp.test($target.val());
```

```
if(!bool) { // 이메일이 정규표현식에 위배된 경우
```

```
  $("#table#tblMemberRegister :input").prop("disabled", true);
  $target.prop("disabled", false);
```

```
  $target.parent().find(".error").show();
  $target.parent().find(".emailChecking").hide();
  $target.focus();
}
```

```
else {
```

```
  $("#table#tblMemberRegister :input").prop("disabled", false);
```

```
  $target.parent().find(".error").hide();
  $target.parent().find(".emailChecking").show();
}
```

[controller]

```
InterMemberDAO mdao = new MemberDAO();
```

```
boolean isExist = mdao.emailDuplicateCheck(email);
```

```
JSONObject jsonObj = new JSONObject();
```

```
jsonObj.put("isExist", isExist);
```

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

[MemberDAO]

```
pstmt = conn.prepareStatement(sql);
pstmt.setString(1, aes.encrypt(email));
```

이메일은 암호화 되어진다.

2. 담당 기능- 회원가입

비밀번호 *

비밀번호는 8글자 이상 15글자 이하에 영문자,숫자,특수기호만 가능합니다.

비밀번호 *

비밀번호확인 *

비밀번호가 일치하지 않습니다.

연락처 - -

휴대폰 형식이 아닙니다.

연락처 - -

성별 ☐ 남성 ☒ 여성

생년월일

`const regExp = new RegExp(/^([1-9][0-9]{3})$/g);`

정규표현식 객체 생성
(첫번째는 숫자 1~9 까지만 가능)

`const regExp = new RegExp(/^\d{4}$/g);`

연락처에 숫자를 제외한 것을 입력할 경우
에러 메시지를 보여준다.

[JSP]

```
const regExp = new RegExp(/^(?=.*{8,15}$)(?=.*\d)(?=.*[a-zA-Z])(?=.*[^\a-zA-Z0-9]).*$/g);

if(pwd != pwdcheck ) {
    //비밀번호가 일치 하지 않은 경우
    $("table#tblMemberRegister :input").prop("disabled", true);
    $target.prop("disabled", false);
    $("input#pwd").prop("disabled", false);

    $target.parent().find(".error").show();

    $("input#pwd").focus();
}
```

숫자/문자/특수문자/ 포함 형태의 8~15자리 이내의
암호 정규표현식 객체 생성.
정규 표현식에 위배되는 경우나 비밀번호 확인 시
일치 하지 않는 경우 에러 메시지를 보여준다

[JSP]

```
////### 생년월일 ###////
let mm_html = "";
for(let i=1; i<=12; i++) {
    if(i<10) {
        mm_html += "<option value ='0'+i+'>0"+i+"</option>";
    }
    else {
        mm_html += "<option value ='"+i+"'>"+i+"</option>";
    }
}

$("select#birthmm").html(mm_html);

let dd_html = "";
for(let i=1; i<=31; i++) {
    if(i<10) {
        dd_html += "<option value ='0'+i+'>0"+i+"</option>";
    }
    else {
        dd_html += "<option value ='"+i+"'>"+i+"</option>";
    }
}

$("select#birthdd").html(dd_html);
```

2. 담당 기능- 회원가입

우편번호	<input type="text" value="13479"/>	<input type="button" value="우편번호찾기"/>
주소	경기 성남시 분당구 서판교로 32	
	기본주소	
	(판교동)	

Daum Postcode Service - Chrome

about:blank

|예) 판교역로 235, 분당 주공, 삼평동 681

tip

아래와 같은 조합으로 검색을 하시면 더욱 정확한 결과가 검색됩니다.

도로명 + 건물번호

예) 판교역로 235, 제주 첨단로 242

지역명(동/리) + 번지

예) 삼평동 681, 제주 영평동 2181

지역명(동/리) + 건물명(아파트명)

예) 분당 주공, 연수동 주공3차

사서함명 + 번호

예) 분당우체국사서함 1~100

Powered by kakao | 우편번호 서비스 안내

카카오(Daum) 주소 찾기

API 사용.

팝업창에서 검색 항목을 클릭

하였을 때 실행할 코드

변수가 값이 없는 경우 공백(' ')

////### 주소 ###////

[JSP]

```
$( "#zipcodeSearch" ).click(function(){
    new daum.Postcode({
        oncomplete: function(data) {

            let addr = ''; // 주소 변수
            let extraAddr = ''; // 참고항목 변수

            if (data.userSelectedType === 'R') { // 사용자가 도로명 주소를 선택했을 경우
                addr = data.roadAddress;
            } else { // 사용자가 지번 주소를 선택했을 경우(J)
                addr = data.jibunAddress;
            }

            // 사용자가 선택한 주소가 도로명 타입일때 참고항목을 조합한다.
            if(data.userSelectedType === 'R'){

                if(data.bname !== '' && /[동|로|가]$/g.test(data.bname)){
                    extraAddr += data.bname;
                }
                // 건물명이 있고, 공동주택일 경우
                if(data.buildingName !== '' && data.apartment === 'Y'){
                    extraAddr += (extraAddr !== '' ? ', ' + data.buildingName : data.buildingName);
                }
                // 표시할 참고항목이 있을 경우, 괄호까지 추가한다.
                if(extraAddr !== ''){
                    extraAddr = ' (' + extraAddr + ')';
                }

                document.getElementById("extraAddress").value = extraAddr;

            } else {
                document.getElementById("extraAddress").value = '';
            }

            // 우편번호와 주소 정보를 해당 필드에 넣는다.
            document.getElementById('postcode').value = data.zonecode;
            document.getElementById("address").value = addr;

            document.getElementById("detailAddress").focus();

        }
    }).open();
})
```

2. 담당 기능- 회원가입

```
<input type="checkbox" name="agree_box" id="agree_box1" /><label for="agree_box1"> (필수) 이용 약관에 대한 동의</label>
```

☑ 모든 약관 동의

아래 모든 약관(필수), 개인정보 수집 및 이용에 대한 동의(필수) 및
광고성 정보수신 동의(선택) 내용을 확인하고 전체 동의합니다.

- 필수 정보의 수집 및 이용에 관한 동의를 거부하실 수 있으나,
다만 이 경우 회원 가입 및 관련 서비스 이용은 불가합니다.
- 필수 정보의 수집 및 이용에 관한 동의를 거부하실 수 있으나,
다만 이 경우 회원 가입 및 관련 서비스 이용은 불가합니다.
- 만 14세 미만은 서비스 이용이 불가합니다.

공식 온라인 스토어 회원 약관에 대한 동의

☑(필수) 이용 약관에 대한 동의 **이용약관 보기**

광고성 정보 수신 동의

☑(선택) 이메일 수신 동의

☑(선택) 문자 수신동의

☐ 모든 약관 동의

아래 모든 약관(필수), 개인정보 수집 및 이용에 대한 동의(필수) 및
광고성 정보수신 동의(선택) 내용을 확인하고 전체 동의합니다.

- 필수 정보의 수집 및 이용에 관한 동의를 거부하실 수 있으나,
다만 이 경우 회원 가입 및 관련 서비스 이용은 불가합니다.
- 필수 정보의 수집 및 이용에 관한 동의를 거부하실 수 있으나,
다만 이 경우 회원 가입 및 관련 서비스 이용은 불가합니다.
- 만 14세 미만은 서비스 이용이 불가합니다.

공식 온라인 스토어 회원 약관에 대한 동의

☐(필수) 이용 약관에 대한 동의 **이용약관 보기**

광고성 정보 수신 동의

☑(선택) 이메일 수신 동의

☐(선택) 문자 수신동의

이용약관

전자상거래(인터넷사이버몰) 표준약관

표준약관 제10023호
(2015. 6. 26. 개정)

제1조(목적) 이 약관은 ○○ 회사(전자상거래 사업자)가 운영하는 ○○ 사이버 몰(이하 "몰"이라 한다)에서 제공하는 인터넷 관련 서비스(이하 "서비스"라 한
다)를 이용할 때 당사(이하 "몰"이라 한다)와 이용자의 권리, 의무 및 책임사항을 규정함을 목적으로 합니다.

※ 「PC통신, 무선 등을 이용하는 전자상거래에 대해서도 그 성질에 반하지 않는 한 이 약관을 준용합니다.」

제2조(정의)

○ "몰"이란 ○○ 회사가 제화 또는 운영(이하 "제화 등"이라 함)을 이용자에게 제공하기 위하여 컴퓨터 등 정보통신설비를 이용하여 제화 등을 거래할 수
있도록 설정한 가상상의 영업장을 말하며, 아울러 사이버몰을 운영하는 사업자의 의미로도 사용됩니다.

○ "이용자"란 "몰"에 접속하여 이 약관에 따라 "몰"이 제공하는 서비스를 받는 회원 및 비회원을 말합니다.

○ '회원'이라 함은 "몰"에 회원등록을 한 자로서, 계속적으로 "몰"이 제공하는 서비스를 이용할 수 있는 자를 말합니다.

○ '비회원'이라 함은 회원에 가입하지 않고 "몰"이 제공하는 서비스를 이용하는 자를 말합니다.

제3조 (약관 동의 명시 및 설정 및 개정)

○ "몰"은 이 약관의 내용과 상호 및 대표자 성명, 영업소 소재지 주소(소배자의 별명을 처리할 수 있는 곳의 주소 포함), 전화번호, 모사전송번호, 전자
우편주소, 사업자등록번호, 통신판매업 신고번호, 개인정보관리책임자 등을 이용자가 쉽게 알 수 있도록 ○○ 사이버몰의 초기 서비스화면(전면)에 게시합니다.
다만, 약관의 내용은 이용자가 연결화면을 통하여 볼 수 있도록 할 수 있습니다.

○ "몰"은 이용자가 약관에 동의하기에 앞서 약관에 정하여져 있는 내용 중 청약철회, 배송책임, 환불조건 등과 같은 중요한 내용을 이용자가 이해할 수 있도
록 별도의 연결화면 또는 팝업화면 등을 제공하여 이용자의 확인을 구하여야 합니다.

○ "몰"은 「전자상거래 등에서의 소비자보호에 관한 법률」, 「약관의 규제에 관한 법률」, 「전자문서 및 전자거래기본법」, 「전자금융거래법」, 「전자

localhost:9090 내용:

필수 이용약관에 동의하세요.

확인

모든 약관 동의의 체크박스를 누르면 하위 체크박스
전체선택 전체 해제가 가능하고 하위 체크박스에서
하나라도 체크를 해제하면 모든 약관동의
체크박스가 해제 되어진다.

//약관 동의 여부

```
const agreeCheckedLength = $("input:checkbox[id='agree_box1']:checked").length;  
  
if(agreeCheckedLength == 0) {  
    alert("필수 이용약관에 동의하세요.");  
    return;  
}
```

하위 체크박스 항목 중 (필수) 항목을 체크
안할 시 에러 메세지 팝업창이 발생한다.

////### 체크박스 전체선택,전체해제 ###////

```
const allCheck = document.getElementById("allCheck");  
const agree_checkBox_List = document.getElementsByName("agree_box");
```

```
allCheck.addEventListener('click', ()=>{  
    const bool = allCheck.checked; // true OR false;
```

```
    for(let agree_checkBox of agree_checkBox_List) {  
        agree_checkBox.checked = bool;  
    }  
});
```

```
for(let agree_checkBox of agree_checkBox_List) {
```

```
    agree_checkBox.addEventListener('click', ()=>{  
        const bool = agree_checkBox.checked;
```

```
        if(!bool) {  
            allCheck.checked = false;
```

```
        }  
        else {  
            let bFlag = false;  
            for(let chkbox of agree_checkBox_List) {  
                if(chkbox.checked) {
```

```
                    bFlag = true;  
                }  
                else {
```

```
                    bFlag = false;  
                    break;
```

```
                }  
            } // end of for-----  
            if(bFlag) {
```

```
                allCheck.checked = true;
```

2. 담당 기능- 회원가입

USERID	PWD	NAME	EMAIL	MOBILE
9 bakhakbak	9695b88a59a1610320897fa84cb7e144cc51f2984520efb77111d94b402a8382	박박박	WzbCDB120Z124+PPHzZLJqKtUX8A3JW6V1yzdiSP/UY=	gntES83qW8BM/5UKkd/eEA

[MemberDAO]

```
try {
    conn = ds.getConnection();

    String sql = " insert into tbl_member(userid, pwd, name, email, mobile, postcode, address, detailaddress, extraaddress, gender, birthday) "
        + " values(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?) ";

    pstmt = conn.prepareStatement(sql);

    pstmt.setString(1, member.getUserid());
    pstmt.setString(2, Sha256.encrypt(member.getPwd()));
    pstmt.setString(3, member.getName());
    pstmt.setString(4, aes.encrypt(member.getEmail())); // 양방향 암호화
    pstmt.setString(5, aes.encrypt(member.getMobile())); // 양방향 암호화
    pstmt.setString(6, member.getPostcode());
    pstmt.setString(7, member.getAddress());
    pstmt.setString(8, member.getDetailaddress());
    pstmt.setString(9, member.getExtraaddress());
    pstmt.setString(10, member.getGender());
    pstmt.setString(11, member.getBirthday());

    result = pstmt.executeUpdate();
```

DB에 insert시 암호는 SHA256 알고리즘으로 단방향. 이메일,휴대폰
번호는 AES256 알고리즘으로 양방향 암호화

[controller]

```
MemberVO member = new MemberVO(userid, pwd, name, email, mobile, postcode, address, detailaddress, extraaddress, gender, birthday);

try {
    InterMemberDAO mdao = new MemberDAO();
    int n = mdao.registerMember(member);

    if(n==1) {
        request.setAttribute("userid", userid);
        request.setAttribute("pwd", pwd);

        //super.setRedirect(false);
        super.setViewPage("/WEB-INF/n01_wonhyejin/registerAfterAutoLogin.jsp");
    }
}
```

모든 제약조건에 위배되지 않고 약관 동의를 하게 되면 '회원가입을 축하드립니다!!!'

라는 팝업창이 뜨고 'OOO님 로그인 성공하셨습니다.'라는 팝업창이 뜬 후 자동 로그인이 되면서 시작페이지로 간다.

localhost:9090 내용:

회원가입을 축하드립니다!!!

확인

localhost:9090 내용:

김유영님 로그인 성공하셨습니다.

확인

2. 담당 기능- 로그인

로그인

아이디

아이디를 입력하세요

비밀번호

비밀번호를 입력하세요

아이디/비밀번호찾기

회원가입

☐아이디저장

로그인

로그인 관련하여 궁금한 사항이 있으신가요?

[자주 묻는 질문 바로가기]

```
//로그인처리
$(document).ready(function(){

    $("button#btnSubmit").click(function(){
        goLogin();
    });
    $("input#loginpw").bind("keydown",function(event){
        if(event.keyCode == 13) {
            goLogin();
        }
    });

    // == localStorage에 저장된 key가 "saveid"인 userid값을 불러와서 input태그 userid에 넣어주기 ==//
    const loginid = localStorage.getItem('saveid');

    if(loginid != null) {
        $("input#loginid").val(loginid);
        $("input:checkbox[id='saveid']").prop("checked",true);
    }
}); // end of $(document).ready(function(){})-----

//로그인 처리
function goLogin() {

//아이디 저장
if( $("input:checkbox[id='saveid']").prop("checked") ) {
    localStorage.setItem('saveid', $("input#loginid").val());
}
else {
    localStorage.removeItem('saveid');
}
}
```

[JSP]

```
String login_success = (String) session.getAttribute("login_ok");

if(login_success != null) {
    alert("<%= login_success%>");
    session.removeAttribute("login ok");
}
```

모든 조건에 맞게 로그인을 할 경우 'loginuser.getName() 님 로그인 성공하셨습니다.'라고 팝업창이 뜬 후 시작페이지로 이동한다.
실패할 경우 다시 로그인 페이지로 이동한다.

[controller]

```
String userid = request.getParameter("userid");
String pwd = request.getParameter("pwd");
```

```
Map<String, String> paramMap = new HashMap<>();
paramMap.put("userid", userid);
paramMap.put("pwd", pwd);
```

```
InterMemberDAO mdao = new MemberDAO();
```

```
MemberVO loginuser = mdao.selectOneMember(paramMap);
```

```
if( loginuser != null) {
```

```
    if(loginuser.getIdle() == 1) {
        String message = "1년 이상 로그인이 되지 않아 휴면 상태로 전환되었습니다. 관리자에게 문의 바랍니다.";
        String loc = request.getContextPath()+"/index.shoes";
```

```
    request.setAttribute("message", message);
    request.setAttribute("loc", loc);
```

```
    super.setRedirect(false);
    super.setViewPage("/WEB-INF/n01_wonhyejin/msg.jsp");
```

```
    return;
```

```
}

HttpSession session = request.getSession();
session.setAttribute("loginuser", loginuser);
```

```
session.setAttribute("login_ok", loginuser.getName()+"님 로그인 성공하셨습니다."); //로그인 했을 때 나타내기
```

```
if( loginuser.isRequirePwdChange() == true ) {
    String message = "비밀번호를 변경하신지 3개월이 지났습니다. 보안을 위해 비밀번호를 변경하시기 바랍니다.";
    String loc = request.getContextPath()+"/index.shoes";
```

```
    request.setAttribute("message", message);
    request.setAttribute("loc", loc);
```

```
    super.setRedirect(false);
    super.setViewPage("/WEB-INF/n01_wonhyejin/msg.jsp");
```

```
}
```

로그인 처리 함수 호출

아이디 저장 체크박스에 체크 시
localStorage.setItem('saveid',
\$("input#loginid").val())
으로 localStorage에 저장

localhost:9090 내용:

로그인 실패하였습니다. 다시 시도해주시기 바랍니다.

확인

localhost:9090 내용:

김김김님 로그인 성공하셨습니다.

확인

김김김님 환영합니다.

2. 담당 기능- 로그인

입력 받은 paramMap을 가지고 1명의 회원 정보를 return 시켜주는 메소드 생성

[MemberDAO]

```
try {
    conn = ds.getConnection();
    String sql = "SELECT userid, name, email, mobile, postcode, address, detailaddress, extraaddress, gender, "+
        "        birthyyy, birthmm, birthdd, point, registerday, pwdchangegap, "+
        "        nvl(lastloggingap, trunc(months_between(sysdate, registerday) ) ) AS lastloggingap "+
    "FROM "+
    "( "+
    "select userid, name, email, mobile, postcode, address, detailaddress, extraaddress, gender "+
    "    , substr(birthday,1,4) AS birthyyy, substr(birthday,6,2) AS birthmm, substr(birthday,9) AS birthdd "+
    "    , point, to_char(registerday, 'yyyy-mm-dd') AS registerday "+
    "    , trunc( months_between(sysdate, lastpwdchangedate) ) AS pwdchangegap "+
    "from tbl_member "+
    "where status = 1 and userid = ? and pwd = ? "+
    ") M "+
    "CROSS JOIN "+
    "( "+
    "select trunc(months_between(sysdate, max(logindate)) ) AS lastloggingap "+
    "from tbl_loginhistory "+
    "where fk_userid = ? "+
    ") H";

    pstmt = conn.prepareStatement(sql);

    pstmt.setString(1, paramMap.get("userid"));
    pstmt.setString(2, Sha256.encrypt(paramMap.get("pwd")) );
    pstmt.setString(3, paramMap.get("userid"));

    rs = pstmt.executeQuery();
    if(rs.next()) {
        member = new MemberVO();

        member.setUserid(rs.getString(1));
        member.setName(rs.getString(2));
        member.setEmail(aes.decrypt(rs.getString(3)) ); //복호화
        member.setMobile(aes.decrypt(rs.getString(4)) ); //복호화
        member.setPostcode(rs.getString(5));
        member.setAddress(rs.getString(6));
        member.setDetailaddress(rs.getString(7));
        member.setExtraaddress(rs.getString(8));
        member.setGender(rs.getString(9));
        member.setBirthday(rs.getString(10) + rs.getString(11) + rs.getString(12));
        member.setPoint(rs.getInt(13));
        member.setRegisterday(rs.getString(14));
    }
}
```

```
if(rs.getInt(15) >=3) {
    // 마지막으로 비밀번호를 변경한 날짜가 현재시각으로 부터 3개월 이내면 false 경과됐으면 true
    member.setRequirePwdChange(true);
}
if(rs.getInt(16) >= 12) {
    // 마지막으로 로그인한 날짜시간이 현재시각으로 부터 1년이 경과되면 휴면으로 처리
    member.setIdle(1);

    // == tbl_member 테이블의 idle 컬럼의 값을 1로 변경하기 == //
    sql = " update tbl_member set idle = 1 "
        + "where userid = ? ";

    pstmt = conn.prepareStatement(sql);
    pstmt.setString(1, paramMap.get("userid"));

    pstmt.executeUpdate();

    // == tbl_loginhistory(로그인기록) 테이블에 insert 하기 == //
    if(member.getIdle() != 1) {
        sql = " insert into tbl_loginhistory(fk_userid) "
            + " values(?) ";

        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, paramMap.get("userid"));
        pstmt.executeUpdate();
    }
}
```

tbl_loginhistory 조회시 userid와 로그인 날짜를 볼 수 있다.

	FK_USERID	LOGINDATE
1	admin	22/04/06

2. 담당 기능- 아이디 찾기/비밀번호 찾기

로그인

아이디
아이디를 입력하세요

비밀번호
비밀번호를 입력하세요

아이디/비밀번호찾기 회원가입

아이디서칭

로그인

로그인 관련하여 궁금한 사항이 있으신가요?
[자주 묻는 질문 바로가기]

로그인 창에서 아이디/비밀번호 찾기를
클릭시 아이디 찾기 창으로 이동한다.

아이디 찾기 창에서 로그인으로 돌아가기 버튼을
클릭시 로그인 창으로 이동한다.

아이디 찾기

계정에 연결된 정보를 입력하시면
아이디 찾기가 가능합니다.

이름
이름을 입력해 주세요

이메일
이메일을 입력해 주세요

비밀번호 찾기

찾기

로그인으로 돌아가기

아이디 찾기 창에서 비밀번호 찾기 버튼을
클릭시 비밀번호 찾기 창으로 이동한다.

비밀번호 찾기 창에서 아이디 찾기 버튼을
클릭시 아이디 찾기 창으로 이동한다.

비밀번호 찾기 창에서 로그인으로 돌아가기 버튼을
클릭시 로그인 창으로 이동한다.

비밀번호 찾기

계정에 연결된 정보를 입력하시면
비밀번호 찾기가 가능합니다.

아이디
아이디를 입력해 주세요

이메일
이메일을 입력해 주세요

아이디 찾기

찾기

로그인으로 돌아가기

2. 담당 기능- 아이디 찾기

이름

김김김

이메일

kimkimkim@gmail.com

비밀번호 찾기

찾기

회원 ID: kimkimkim

이름

가가가

이메일

gagaga@gmail.com

비밀번호 찾기

찾기

회원 ID: 존재하지 않는 회원 아이디입니다.

// 아이디 찾기 [MemberDAO]

```
@Override
public String findUserId(Map<String, String> paramMap) throws SQLException {

    String userid = null;

    try {
        conn = ds.getConnection();

        String sql = " select userid "
            + " from tbl_member"
            + " where status = 1 and name = ? and email = ? "; //1은 탈퇴한 회원

        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, paramMap.get("name") );
        pstmt.setString(2, aes.encrypt(paramMap.get("email")));
        rs = pstmt.executeQuery();

        if(rs.next()) {
            userid = rs.getString(1);
        }

    } catch (GeneralSecurityException | UnsupportedEncodingException e) {
        e.printStackTrace();
    } finally {
        close();
    }

    return userid;
}
```

이름과 이메일을 입력 받아서 사용자의
아이디를 알려준다

select 된 결과물을 userid에 넣어서 리턴

[controller]

```
if("POST".equalsIgnoreCase(method)) { //찾기 버튼을 클릭했을 경우
```

```
String name = request.getParameter("name");
String email = request.getParameter("email");
```

```
InterMemberDAO mdao = new MemberDAO();
```

```
Map<String, String> paramMap = new HashMap<>(); //name과 email보내기
paramMap.put("name", name);
paramMap.put("email", email);
```

이름과 이메일을 매핑

```
String userid = mdao.findUserId(paramMap);
```

```
if(userid != null) {
    request.setAttribute("userid", userid);
} else { // ID가 존재하지 않은 경우
    request.setAttribute("userid", "존재하지 않는 회원 아이디입니다.");
}
```

```
request.setAttribute("name", name ); //넘어온 키 값
request.setAttribute("email", email );
```

//찾기 [JSP]

```
$("#button#btnFind").click(function(){
    goFind();
});

$("#input#email").bind("keydown", function(event){

    if(event.keyCode == 13) {
        goFind();
    }

});
```

```
}); // end of $(document).ready(function()-----
```

Post 형식으로 form에 입력한 데이터 전송

```
// Function Declaration
function goFind() {
    const frm = document.idFindFrm;
    frm.action = "%<%= ctxPath%>/idfind.shoes";
    frm.method = "post";
    frm.submit();
}
```

2. 담당 기능- 비밀번호 찾기

비밀번호 찾기

계정에 연결된 정보를 입력하시면
비밀번호 찾기가 가능합니다.

아이디

아이디를 입력해 주세요

이메일

이메일을 입력해 주세요

아이디 찾기

찾기

로그인으로 돌아가기

이메일 g****11@gmail.com로 인증코드가 발송되었습니다.
인증코드를 입력해주세요. 0:04:46

연장

인증하기

[JSP]

```
else if(method == "POST") {
    $("div#div_findResult").show();

    $("input#userid").val("${requestScope.userid}");
    $("input#email").val("${requestScope.email}");

    if("${requestScope.sendMailSuccess} == true") {
        $("div#div_btnFind").hide(); //메일이 정상적으로 왔다면 찾기버튼을 숨긴다.
    }
}

// 찾기
$("button#btnFind").click(function(){
    goFind();
});

$("input#email").bind("keydown", function(event){
    if(event.keyCode == 13) {
        goFind();
    }
});

// 인증하기
$("button#btnConfirmCode").click(function(){ //인증키는 포스트 방식

    const frm = document.verifyCertificationFrm;
    frm.userCertificationCode.value = $("input#input_confirmCode").val(); // 사용자가 입력한 값을 넣어준다.
    frm.userid.value = $("input#userid").val();

    frm.action = "<%= ctxPath%>/verifyCertification.shoes";
    frm.method = "post";
    frm.submit();
}
```

[controller]

```
@Override
public void execute(HttpServletRequest request, HttpServletResponse response) throws Exception {

    String method = request.getMethod();
    //"GET" 또는 "POST"

    if("POST".equalsIgnoreCase(method)) {

        String userid = request.getParameter("userid");
        String email = request.getParameter("email");

        InterMemberDAO mdao = new MemberDAO();
        Map<String, String> paramMap = new HashMap<>();
        paramMap.put("userid", userid);
        paramMap.put("email", email);

        boolean isUserExist = mdao.isUserExist(paramMap);
```

[MemberDAO]

```
try {
    conn = ds.getConnection();

    String sql = " select userid "
        + " from tbl_member"
        + " where status = 1 and userid = ? and email = ? "; //1은 탈퇴한 회원

    pstmt = conn.prepareStatement(sql);
    pstmt.setString(1, paramMap.get("userid"));
    pstmt.setString(2, aes.encrypt(paramMap.get("email")));

    rs = pstmt.executeQuery();

    isUserExist = rs.next();

} catch (GeneralSecurityException | UnsupportedEncodingException e) {
    e.printStackTrace();
} finally {
    close();
}

return isUserExist;
```

아이디, 이메일을 입력 받아서 해당사용자가
존재하는지 유무를 알려준다

Post 형식으로 form에 입력한 데이터 전송

입력한 정보를 가진 회원이 존재하는지 확인

2. 담당 기능- 비밀번호 찾기

Property에 SMTP 서버 정보 설정.

smtp는 이메일을 전송할 때 사용하는 프로토콜.

이메일 보내는 계정과 이메일을 형식에 관해 설정한다.

```
Properties prop = new Properties();  
prop.put("mail.smtp.user", "자신의 이메일 주소"); // 자신의 이메일 주소  
prop.put("mail.smtp.host", "smtp.gmail.com");  
  
prop.put("mail.smtp.port", "465");  
prop.put("mail.smtp.starttls.enable", "true");  
prop.put("mail.smtp.auth", "true");  
prop.put("mail.smtp.debug", "true");  
prop.put("mail.smtp.socketFactory.port", "465");  
prop.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");  
prop.put("mail.smtp.socketFactory.fallback", "false");  
  
prop.put("mail.smtp.ssl.enable", "true");  
prop.put("mail.smtp.ssl.trust", "smtp.gmail.com");
```

발신자의 이메일 주소를 입력한다.

```
Authenticator smtpAuth = new MySMTPAuthenticator();  
Session session = Session.getInstance(prop, smtpAuth); //메일 세션 생성
```

```
session.setDebug(true); //콘솔에 출력
```

```
// 메일의 내용을 담기 위한 객체생성  
MimeMessage msg = new MimeMessage(session);
```

```
//메일 제목  
String subject = "<semi_prj_shoes>회원님의 비밀번호를 찾기 위한 인증코드 발송";  
msg.setSubject(subject);
```

```
// 발신자의 메일주소  
String sender = "자신의 이메일 주소";  
Address fromAddr = new InternetAddress(sender);  
msg.setFrom(fromAddr);
```

발신자의 이메일 주소를 입력한다.

```
// 수신자의 메일주소  
Address toAddr = new InternetAddress(recipient);  
msg.addRecipient(Message.RecipientType.TO, toAddr);
```

```
// 내용  
msg.setContent("발송된 인증코드 : <span style='font-size:14pt; color:red;'>"+certificationCode+"</span>", "text/html;charset=UTF-8");
```

```
// 메일 보내기  
Transport.send(msg);
```

[controller]

랜덤한 인증코드 설정

```
boolean sendMailSuccess = false; // 메일이 정상적으로 전송되었는지 유무를 알아오기 위한 용도
```

```
if(isUserExist) {  
    // 회원으로 존재하는 경우  
    //인증키를 랜덤하게 생성하도록 한다.  
    Random rnd = new Random();  
  
    String certificationCode = "";  
    //인증키는 영문소문자 5글자 +숫자 7글자  
  
    char randchar = ' ';  
    for(int i=0; i<5; i++) {  
  
        randchar = (char) (rnd.nextInt('z' - 'a' + 1) + 'a');  
        certificationCode += randchar;  
    } //end of for-----
```

```
int randnum = 0;  
for(int i=0; i<7; i++) {  
    randnum = rnd.nextInt(9 - 0 + 1) + 0;  
    certificationCode += randnum;
```

```
} //end of for-----
```

```
//랜덤하게 생성한 인증코드(certificationCode)를 비밀번호 찾기를 하고자 하는 사용자의 email로 전송시킨다.  
GmailSend mail = new GmailSend();
```

```
try {  
    mail.sendmail(email, certificationCode);  
    sendMailSuccess = true; //메일전송이 성공했음을 기록함  
  
    //세션 불러오기  
    HttpSession session = request.getSession();  
    session.setAttribute("certificationCode", certificationCode);  
    //발급한 인증코드를 세션에 저장시킴  
  
} catch (Exception e) {  
    //메일전송이 실패한 경우  
    e.printStackTrace();  
    sendMailSuccess = false; //메일전송이 실패했음을 기록함  
}
```

```
} //end of if(isUserExist)-----
```

```
request.setAttribute("isUserExist", isUserExist);  
request.setAttribute("sendMailSuccess", sendMailSuccess);  
request.setAttribute("userid", userid);  
request.setAttribute("email", email);
```

```
} // end of if("POST".equalsIgnoreCase(method))-----
```

```
request.setAttribute("method", method);  
//super.setRedirect(false);  
super.setViewPage("/WEB-INF/n01_wonhyejin/pwdFind.jsp");
```

min 부터 max 사이의 값으로 랜덤한 정수를 얻으려면
int rndnum = rnd.nextInt(max - min + 1) + min;
영문 소문자 'a' 부터 'z' 까지 랜덤하게 5개를 만든다.

발송된 인증코드 : dnjuu5678548

2. 담당 기능- 비밀번호 찾기

비밀번호 찾기

계정에 연결된 정보를 입력하시면
비밀번호 찾기가 가능합니다.

아이디

gghh11

이메일

gghh11@gmail.com

아이디 찾기

이메일 gghh11@gmail.com로 인증코드가 발송되었습니다.
인증코드를 입력해주세요. 0:04:46 연장

인증하기

로그인으로 돌아가기

입력한 이메일로 인증코드가 보내지고
5분 제한시간이 주어진다.

5분이 지나기 전에 연장 버튼을 클릭시
다시 5분으로 제한시간이 주어진다.

5분이 지나면 비밀번호 찾기 창으로
이동되어 다시 인증코드를 받아야 한다.

[JSP]

```
//인증코드 발생 후 제한시간안에 인증(연장 가능)
counter_init();
var tid;
var cnt = parseInt(300); //초기값(초단위) 5분

function counter_init() { //카운트 실행
    tid = setInterval("counter_run()", 1000);
}

function counter_reset() {
    clearInterval(tid);
    cnt = parseInt(300);
    counter_init();
}

function counter_run() { //카운트
    document.getElementById("counter").innerText = time_format(cnt);
    cnt--;
    if(cnt < 0) {
        alert("인증 요청을 다시 하세요");
        clearInterval(tid);
        location.replace(location.href); //비밀번호 찾기 화면으로 이동.
    }
}

function time_format(s) {
    var nHour=0;
    var nMin=0;
    var nSec=0;
    if(s>0) {
        nMin = parseInt(s/60);
        nSec = s%60;

        if(nMin>60) {
            nHour = parseInt(nMin/60);
            nMin = nMin%60;
        }
    }
    if(nSec<10) nSec = "0"+nSec;
    if(nMin<10) nMin = "0"+nMin;

    return ""+nHour+":"+nMin+":"+nSec;
}
```


2. 담당 기능- 비밀번호 찾기(변경)

localhost:9090 내용:
인증이 완료되었습니다.

확인

비밀번호 변경

다른 사이트에서 사용하지 않는 보안 수준이 높은 비밀번호를 설정하는 것이 좋습니다.

새 비밀번호

새 비밀번호를 입력하세요.

새 비밀번호 확인

새 비밀번호를 한번 더 입력하세요.

비밀번호 변경

로그인으로 돌아가기

이메일로 발급된 인증코드를 올바르게 입력시 인증완료 팝업창이 뜬 후 비밀번호 변경 창으로 이동한다.

[controller]

```
@Override
public void execute(HttpServletRequest request, HttpServletResponse response) throws Exception {

    String userCertificationCode = request.getParameter("userCertificationCode"); //유저 인증키
    String userid = request.getParameter("userid");

    HttpSession session = request.getSession(); // 랜덤하게 발송되어진 인증키
    String certificationCode =(String)session.getAttribute("certificationCode"); //세션에 저장된 인증코드 가져오기

    String message = "";
    String loc = "";

    if(certificationCode.equals(userCertificationCode) ) {
        message = "인증이 완료되었습니다."; // 유저가 보내준 인증코드와 메일 받은 인증코드가 같다면 새로운 비밀번호 만들기
        loc = request.getContextPath()+"/pwdUpdate.shoes?userid="+userid; // 변경할 유저 아이디
    }
    else {
        message = "발급된 인증코드가 아닙니다. 인증코드를 다시 발급받으세요!!";
        loc = request.getContextPath()+"/pwdFind.shoes";
    }

    request.setAttribute("message", message);
    request.setAttribute("loc", loc);

    //super.setRedirect(false);
    super.setViewPage("/WEB-INF/n01_wonhyejin/msg.jsp");

    session.removeAttribute("certificationCode"); //세션에 저장된 인증코드 삭제
```

2. 담당 기능- 비밀번호 찾기(변경)

[controller]

비밀번호를 변경하려는 사용자의 ID와 비밀번호를 넘겨온다

```
@Override
public void execute(HttpServletRequest request, HttpServletResponse response) throws Exception {
```

```
String userid = request.getParameter("userid");
```

```
String method = request.getMethod();
```

```
if("POST".equalsIgnoreCase(method)) {
```

```
String pwd = request.getParameter("pwd");
```

```
Map<String, String> paramMap = new HashMap<>();
paramMap.put("pwd", pwd);
paramMap.put("userid", userid);
```

```
InterMemberDAO mdao = new MemberDAO();
```

```
int n = mdao.pwdUpdate(paramMap);
```

```
request.setAttribute("n", n);
```

} Map에 저장 후 비밀번호 변경 함수 호출

```
request.setAttribute("userid", userid);
request.setAttribute("method", method);
```

```
// super.setRedirect(false);
```

```
super.setViewPage("/WEB-INF/n01_wonhyejin/pwdUpdate.jsp");
```

비밀번호가 정규표현식에 위배되지 않고
입력한 비밀번호가 서로 일치한다면
비밀번호가 변경되어진다.

비밀번호 변경

다른 사이트에서 사용하지 않는 보안 수준이
높은 비밀번호를 설정하는 것이 좋습니다.

새 비밀번호

새 비밀번호를 입력하세요.

새 비밀번호 확인

새 비밀번호를 한번 더 입력하세요.

gghh11님의 비밀번호가 변경되었습니다.

로그인으로 돌아가기

```
<c:if test="${requestScope.method == 'POST' && requestScope.n == 1}">
  <div id="div_updateResult" align="center" style="color:red;">
    ${requestScope.userid}님의 비밀번호가 변경되었습니다.<br/>
  </div>
</c:if>
```

[MemberDAO]

```
try {
    conn = ds.getConnection();

    String sql = " update tbl_member set pwd = ? "
                + " , lastpwdchangedate = sysdate "
                + " where userid = ? ";

    pstmt = conn.prepareStatement(sql);

    pstmt.setString(1, Sha256.encrypt(paramMap.get("pwd")) ); // 암호를 SHA256 알고리즘으로 단방향 암호화 시킨다.
    pstmt.setString(2, paramMap.get("userid") );

    result = pstmt.executeUpdate();
} finally {
    close();
}
return result;
```

[JSP]

```
$(document).ready(function(){
```

```
$("#button#btnUpdate").click(function(){
```

```
const pwd = $("#input#pwd").val();
const pwd2 = $("#input#pwd2").val();
```

```
const regExp = new RegExp(/^(?=.*{8,15}$)(?=.*\d)(?=.*[a-zA-Z])(?=.*[a-zA-Z0-9]).*$/g);
```

```
const bool = regExp.test(pwd);
```

```
if(!bool) {
```

```
// 비밀번호가 정규표현식에 위배된 경우
```

```
alert("비밀번호는 8글자 이상 15글자 이하에 영문자,숫자,특수기호만 가능합니다.");
```

```
$("#input#pwd").val("");
```

```
$("#input#pwd2").val("");
```

```
return; //종료
```

```
} else if(bool && pwd != pwd2) {
```

```
// 비밀번호가 정규표현식에 위배된 경우
```

```
alert("비밀번호가 일치하지 않습니다.");
```

```
$("#input#pwd").val("");
```

```
$("#input#pwd2").val("");
```

```
return; //종료
```

```
} else {
```

```
const frm = document.pwdUpdateFrm;
```

```
frm.action = "<%= ctxPath%>/pwdUpdate.shoes";
```

```
frm.method = "post";
```

```
frm.submit();
```

숫자/문자/특수문자/ 포함 형태의 8~15자리 이내의
비밀번호 정규표현식 객체 생성

새 비밀번호(pwd)와 새 비밀번호 확인(pwd2)
일치 하는지 확인

THANK YOU