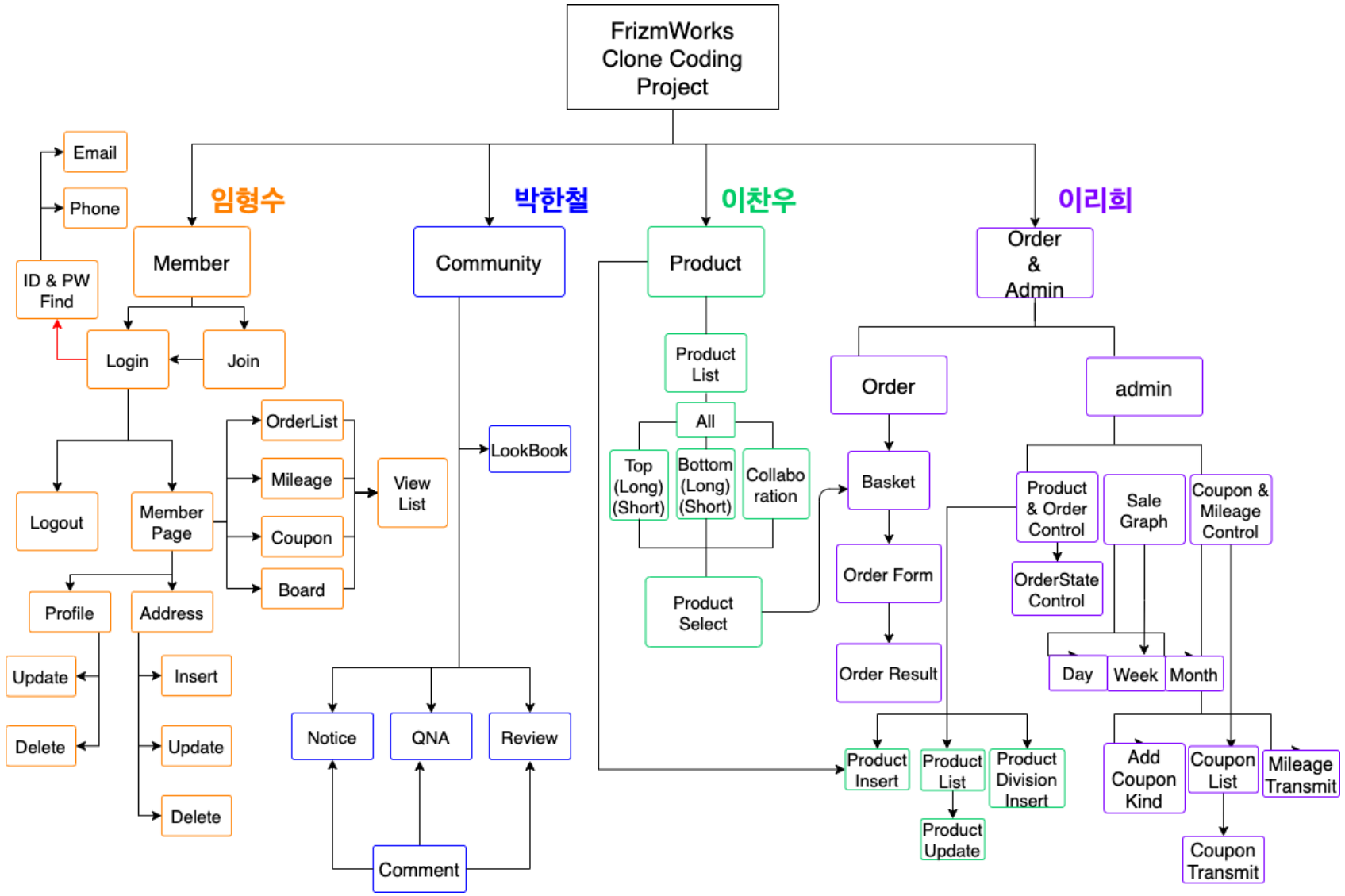
FrizmWorksCloneCoding

박한철,이리희,이찬우,임형수

개발 환경

OS	Windows10 / MacOS
DBMS	MariaDB
Server	Apache Tomcat 9.0
Language	Java
Front-End	HTML5 / CSS / JavaScript / jQuery / Ajax / JSP
Framework	Spring Boot / Mybatis / Bootstrap
Tools	Git / Maven
API	카카오 로컬 / day.js / Spring Validation / Spring WebSecurity


FrizmWorks Clone Coding Project
임형수
박한철
이찬우
이리희
Email
Phone
ID & PW Find
Member
Login
Join
Logout
Member Page
Profile
Address
Update
Delete
Insert
Update
Delete
OrderList
Mileage
Coupon
Board
View List
Community
LookBook
Notice
QNA
Review
Comment
Product
Product List
All
Top (Long) (Short)
Bottom (Long) (Short)
Collabo ration
Product Select
Order & Admin
Order
admin
Basket
Order Form
Order Result
Product Insert
Product List
Product Division Insert
Product Update
Product & Order Control
Sale Graph
Coupon & Mileage Control
OrderState Control
Day
Week
Month
Add Coupon Kind
Coupon List
Mileage Transmit
Coupon Transmit

productInfo

top bottom 상품정보

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			index	pInfoNum	BIGINT
			✔	사이즈	size	VARCHAR(10)
		✔	✔	상품번호	productNum	BIGINT
			✔	재고수량	stock	BIGINT

cart

장바구니

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			장바구니 번호	cartNum	BIGINT
		✔	✔	품목 번호	productNum	BIGINT
		✔	✔	품목 정보	pInfoNum	BIGINT
			✔	품목 개수	productCount	BIGINT
		✔	✔	회원id	username	VARCHAR(100)

purchase

구매목록(각 단일 항목에 대한 구매내역)

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			구매번호	purNum	BIGINT
		✔	✔	상품 정보	pInfoNum	BIGINT
		✔	✔	상품 번호	productNum	BIGINT
		✔	✔	주문번호	orderNum	VARCHAR(100)
			✔	상품개수	productCount	BIGINT
			✔	상품총가격	proPriceSum	BIGINT

orderlist

주문목록(전체 항목에 대한 구매내역)

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			주문번호	orderNum	VARCHAR(100)
			✔	총 가격	totPrice	BIGINT
			✔	지불 가격	spPrice	BIGINT
		✔	✔	쿠폰번호	cuNum	BIGINT
			✔	주문자id	username	VARCHAR(100)
			✔	주소지	destination	VARCHAR(450)
			✔	주문날짜	orderDate	DATETIME
			✔	주문 현재상황	orderState	BIGINT
			✔	배송 메세지	orderMassage	TEXT
			✔	배송이름	orderName	VARCHAR(200)

mileage

마일리지

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			마일리지 PK	mileNum	BIGINT
			✔	사용된 적립금	usedMile	BIGINT
			✔	변화된 마일리지수치	changeMile	BIGINT
		✔	✔	멤버 id FK	username	VARCHAR(100)
		✔	✔	주문번호 FK	orderNum	VARCHAR(100)
			✔	내용	mileContents	VARCHAR(100)
			✔	사용가능 적립금	enableMile	BIGINT

product

top bottom collaboration 상품

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			상품번호	productNum	BIGINT
			✔	상품이름	productTitle	VARCHAR(100)
			✔	가격	productPrice	INT
			✔	할인율	productDisRate	INT
			✔	상품소개	summary	LONGTEXT
			✔	설명	productContents	LONGTEXT
		✔	✔	상품 종류번호	productDivNum	BIGINT
			✔	할인적용가격	finalPrice	BIGINT
			✔	상품 마일리지	productMileage	BIGINT
			✔	상품판매여부	productSaleable	BIT

productDivision

product table의 상품 종류들의 종류를 모아둔 table

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			상품 종류번호	productDivNum	BIGINT
			✔	콜라보레이션이름	collab	VARCHAR(100)
			✔	상품 상세타입	productType	VARCHAR(100)

memberRole

memberRole

PK	AI	FK	Null	Logical Name	Name	Type
		✔		username	username	VARCHAR(100)
✔		✔		id	id	BIGINT

role

Table logical name

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			id	id	BIGINT
			✔	roleName	roleName	VARCHAR(100)

coupon

쿠폰

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			쿠폰발급번호	cuNum	BIGINT
		✔	✔	회원id	username	VARCHAR(100)
		✔	✔	쿠폰 종류번호	cuSpNum	BIGINT
			✔	발급일	pubDate	DATE
			✔	만료일	exDate	DATE
			✔	쿠폰사용여부	useCheck	BIT

couponsp

쿠폰 종류

PK	AI	FK	Null	Logical Name	Name	Type
✔				쿠폰종류번호	cuSpNum	BIGINT
			✔	쿠폰 이름	cuName	VARCHAR(100)
			✔	할인율	disRate	INT

productFiles

제품 사진

PK	AI	FK	Null	Logical Name	Name	Type
		✔	✔	상품 번호	productNum	BIGINT
✔	✔			파일번호	fileNum	BIGINT
			✔	파일이름	fileName	VARCHAR(100)
			✔	원본이름	oriName	VARCHAR(100)

admin

관리자용 테이블(통합 조회용)

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			관리자용 인덱스	adminIdx	BIGINT
			✔	정산 날짜	adminDate	DATE
		✔	✔	제품 번호	productNum	BIGINT
			✔	그날 총 판매개수	sellCount	BIGINT

qna

qna

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			글번호	num	BIGINT
		✔	✔	제품	productNum	BIGINT
			✔	제목	title	VARCHAR(100)
		✔	✔	쓴사람	writer	VARCHAR(100)
			✔	등록일	regDate	DATETIME
			✔	조회수	hit	BIGINT
			✔	내용	contents	LONGTEXT
		✔	✔	boardType	typeNum	BIGINT

member

멤버

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			id	username	VARCHAR(100)
			✔	pw	password	VARCHAR(100)
			✔	전화번호	phone	VARCHAR(100)
			✔	이름	name	VARCHAR(100)
			✔	email	email	VARCHAR(100)
			✔	생년월일	birth	VARCHAR(100)
			✔	enabled	enabled	BIT
			✔	sms 수신동의	smsAgree	BIT
			✔	email 수신동의	emailAgree	BIT

address

주소

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			주소번호 PK	addrNum	BIGINT
		✔	✔	회원id FK	username	VARCHAR(100)
			✔	배송지명	addrName	VARCHAR(100)
			✔	수령인	recipient	VARCHAR(100)
			✔	전화번호	addrPhone	VARCHAR(100)
			✔	우편번호	zipCode	VARCHAR(100)
			✔	기본주소	basicAddr	VARCHAR(200)
			✔	상세주소	detailAddr	VARCHAR(200)
			✔	주문자정보여부	orderAddr	BIT
			✔	최근 사용여부	recentUse	BIT

qnaFiles

Table logical name

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			파일번호	fileNum	BIGINT
		✔	✔	글번호	num	BIGINT
			✔	파일이름	fileName	VARCHAR(100)
			✔	오리진	oriName	VARCHAR(100)

qnaComment

qna댓글

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			댓글번호	commentNum	BIGINT
		✔	✔	글번호	num	BIGINT
			✔	등록일	regDate	DATETIME
		✔	✔	쓴사람	writer	VARCHAR(100)
			✔	내용	contents	LONGTEXT

boardType

게시판의 타입이 무엇이 있는지 확인하기 위함

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			typeNum	typeNum	BIGINT
			✔	typeName	typeName	VARCHAR(100)

review

리뷰테이블

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			글번호	num	BIGINT
		✔	✔	boardType	typeNum	BIGINT
			✔	제목	title	VARCHAR(100)
		✔	✔	쓴사람	writer	VARCHAR(100)
			✔	등록일	regDate	DATETIME
			✔	조회수	hit	BIGINT
			✔	내용	contents	LONGTEXT

notice

notice

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			글번호	num	BIGINT
		✔	✔	쓴사람	writer	VARCHAR(100)
			✔	등록일	regDate	DATETIME
			✔	내용	contents	LONGTEXT
			✔	조회수	hit	BIGINT
			✔	제목	title	VARCHAR(100)

reviewFiles

Table logical name

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			파일번호	fileNum	BIGINT
		✔	✔	글번호	num	BIGINT
			✔	파일이름	fileName	VARCHAR(100)
			✔	오리진	oriName	VARCHAR(100)

reviewComment

리뷰댓글

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			댓글번호	commentNum	BIGINT
		✔	✔	글번호	num	BIGINT
			✔	등록일	regDate	DATETIME
		✔	✔	쓴사람	writer	VARCHAR(100)
			✔	내용	contents	LONGTEXT

noticeFiles

Table logical name

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			파일번호	fileNum	BIGINT
		✔	✔	글번호	num	BIGINT
			✔	파일이름	fileName	VARCHAR(100)
			✔	오리진	oriName	VARCHAR(100)

noticeComment

notice댓글

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			댓글번호	commentNum	BIGINT
		✔	✔	글번호	num	BIGINT
		✔	✔	쓴사람	writer	VARCHAR(100)
			✔	등록일	regDate	DATETIME
			✔	내용	contents	LONGTEXT

lookbook

lookbook

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			글번호	lookbookNum	BIGINT
			✔	매체 종류	division	VARCHAR(10)
			✔	제목	title	varchar(100)

lookbookFiles

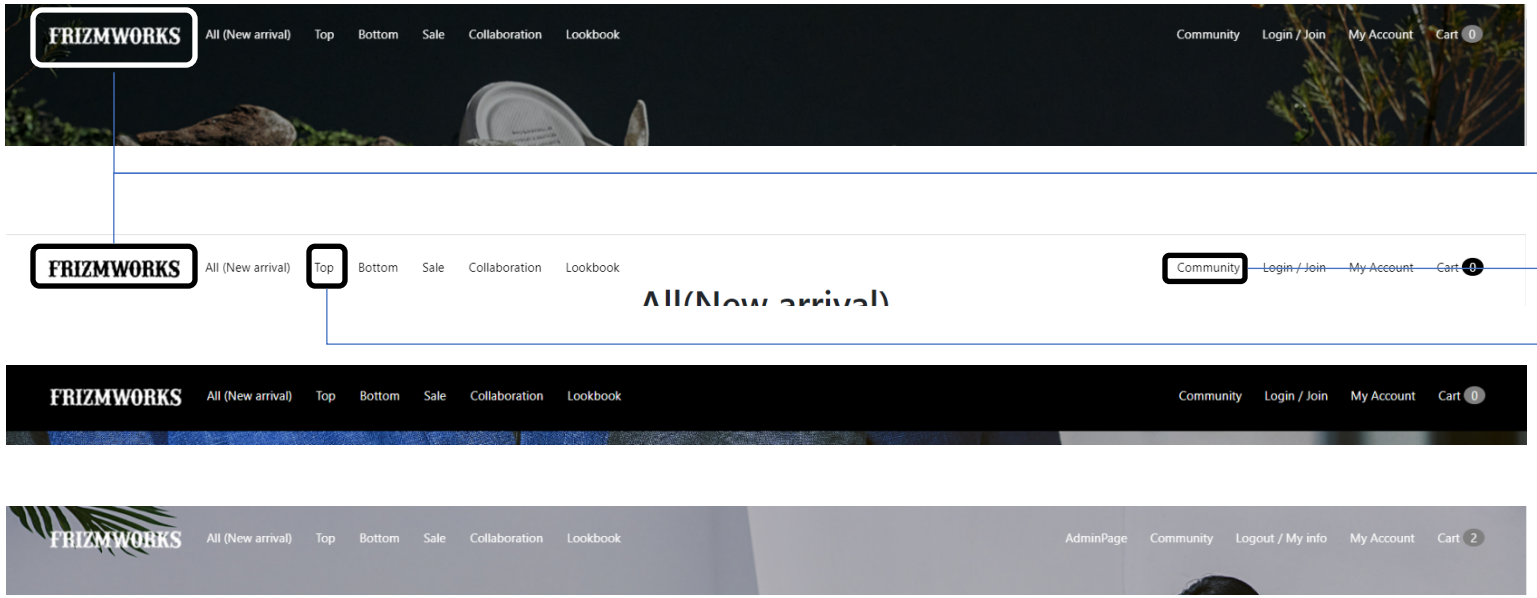
lookbook에 들어가는 파일 테이블

PK	AI	FK	Null	Logical Name	Name	Type
✔	✔			룩북파일번호	lookbookFileNum	BIGINT
			✔	파일이름	filename	VARCHAR(100)
		✔	✔	글번호	num	BIGINT
			✔	원본이름	oriNAme	VARCHAR(100)

Admin & Order

이리희

Public Form




```
<a href="${pageContext.request.contextPath}/">
    <c:if test='${param.isCommon eq "true"}'>
        <img src="/images/logo_b.png" alt="" />
    </c:if>
    <c:if test='${param.isCommon eq "false"}'>
        <img src="/images/logo_w.png" alt="" />
    </c:if>
</a>
```

Top Bottom

TOP (long)

TOP (short)

Community

Notice

Stockist

Q&A

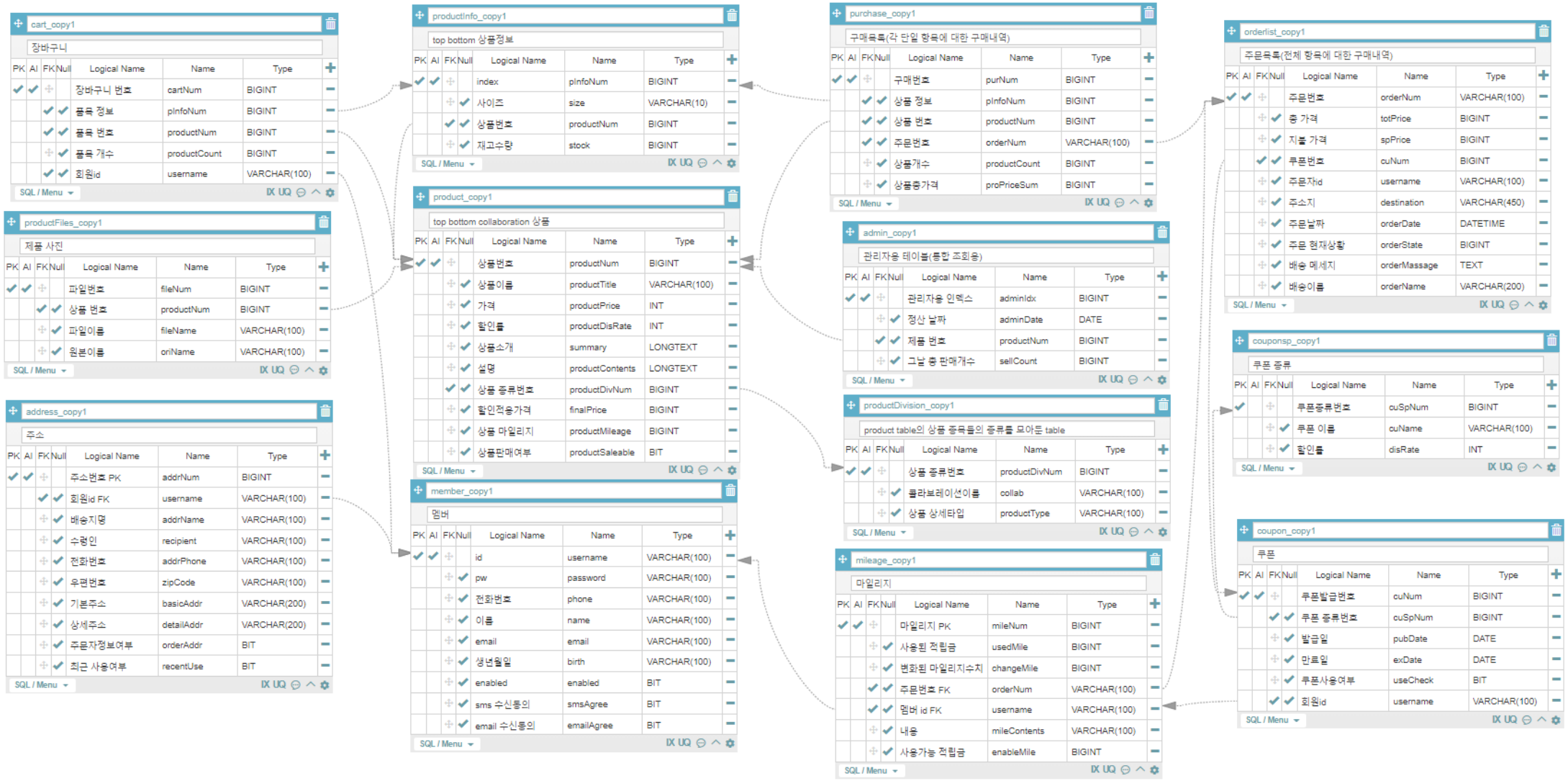
Review

Index page와 다른 form에서 글자 색과 마우스를 올렸을 때 나타나는 navbar색이 다르게 설정되어있다.
또한 top, bottom, collaboration, community, my Account의 경우 마우스를 올리면 하단 메뉴가 나오게 했으며,
Spring Security를 이용하였기에 로그인하면 navbar의 구성이 달라지게 된다.
또한 로그인 시 DB에서 조회하여 cart에 담긴 품목 개수를 띄우게 만들었다.

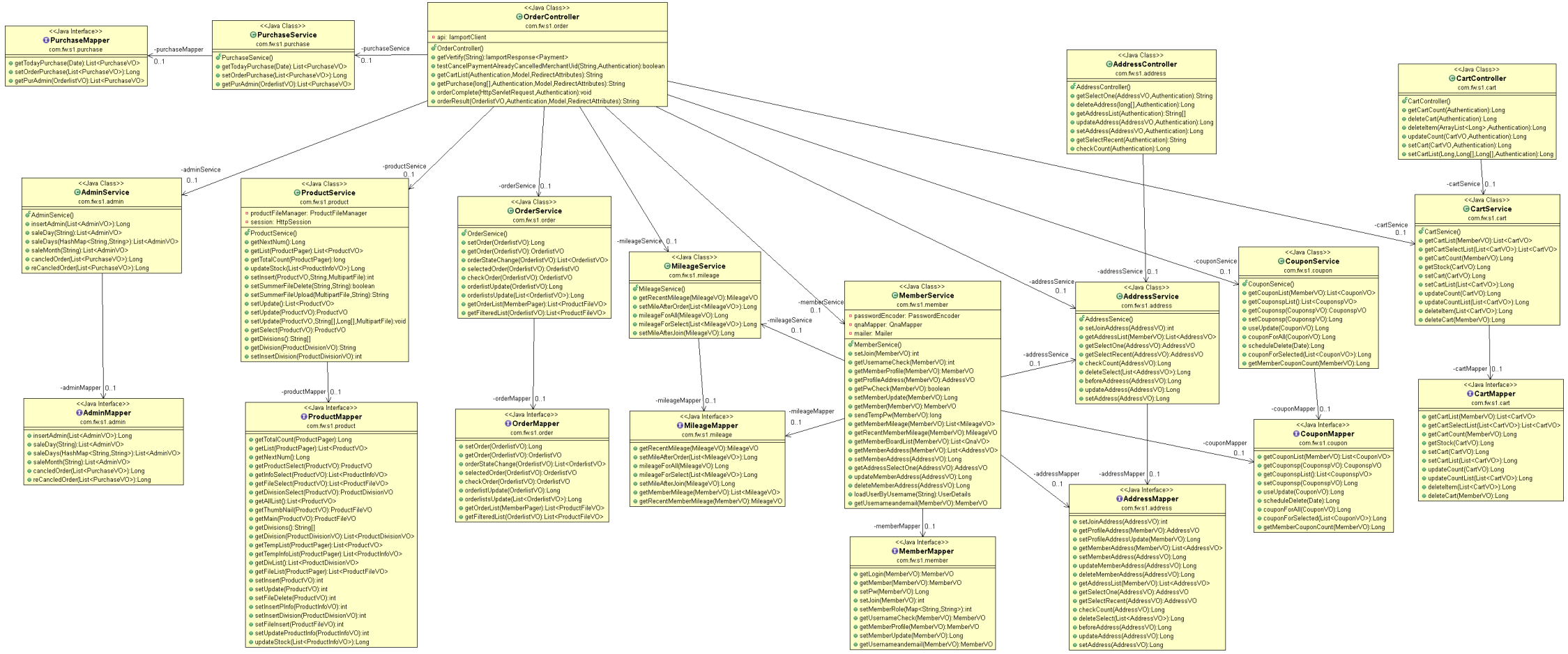
Index page는 bootstrap을 이용한 Carousel이 존재한다.

Footer의 경우 Up button을 넣어 어디서든 해당 버튼을 누르면 Page의 최 상단으로 이동할 수 있다.

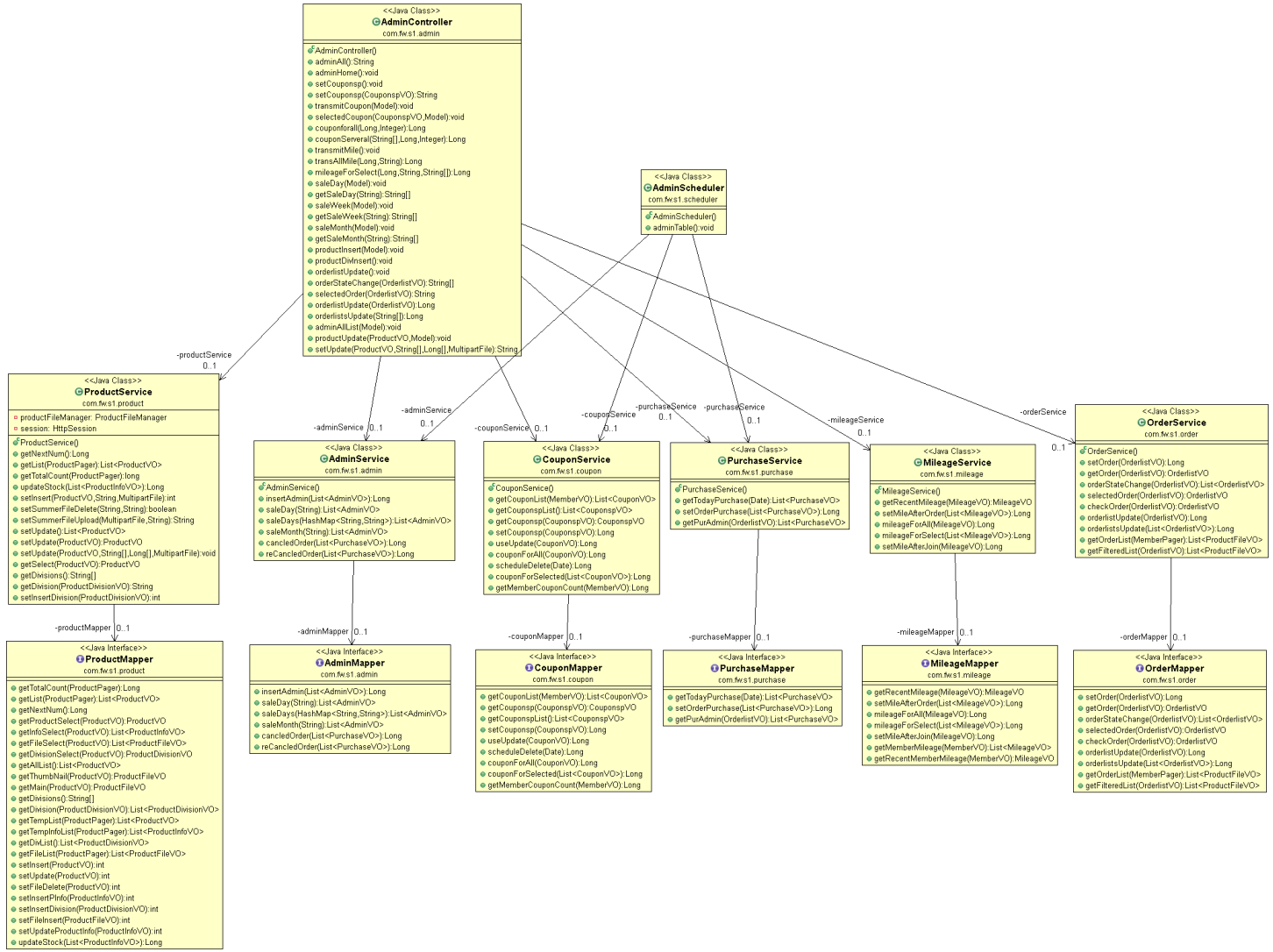
Admin & Order Database Table



Order Class Diagram



Admin Class Diagram



Order와 Admin 둘의 diagram을 보았을 때 다른 영역에 굉장히 영향을 많이 받는다는 것을 알 수 있다.
실제로도 테이블의 변경에 굉장히 영향을 많이 받았고, 처리하는 과정이 많기에 코드가 굉장히 복잡하고 난해해졌다.
해당 작업을 통해 spring에 대한 이해도와 클린코딩의 중요성을 깊게 체득하게 되었다.

CART – Set Cart

S_shirt19

KRW 45300

2265 (5%)

S_shirt19S_shirt19S_shirt19

CART(BASKET)

CART

일반상품(2)

	THUMB	PRODUCT	PRICE	QUANTITY	MILEAGE	DELIVERY	CHARGE	TOTAL	SELECT
☐		S_pants29 [옵션: M]	KRW 29300 [할인률 : 0%]	1 변경	1465원	기본배송	[무료]	KRW 29300	주문하기 삭제하기
☐		L_pants29 [옵션: M]	KRW 39000 [할인률 : 0%]	1 변경	1950원	기본배송	[무료]	KRW 39000	주문하기 삭제하기

[기본배송]

상품구매금액 **68300** + 배송비 0 (무료) = 합계: **KRW 68300**

할인 적용 금액은 주문서작성의 결제예정금액에서 확인가능합니다.

장바구니 비우기

R SELECT

GO SHOPPING

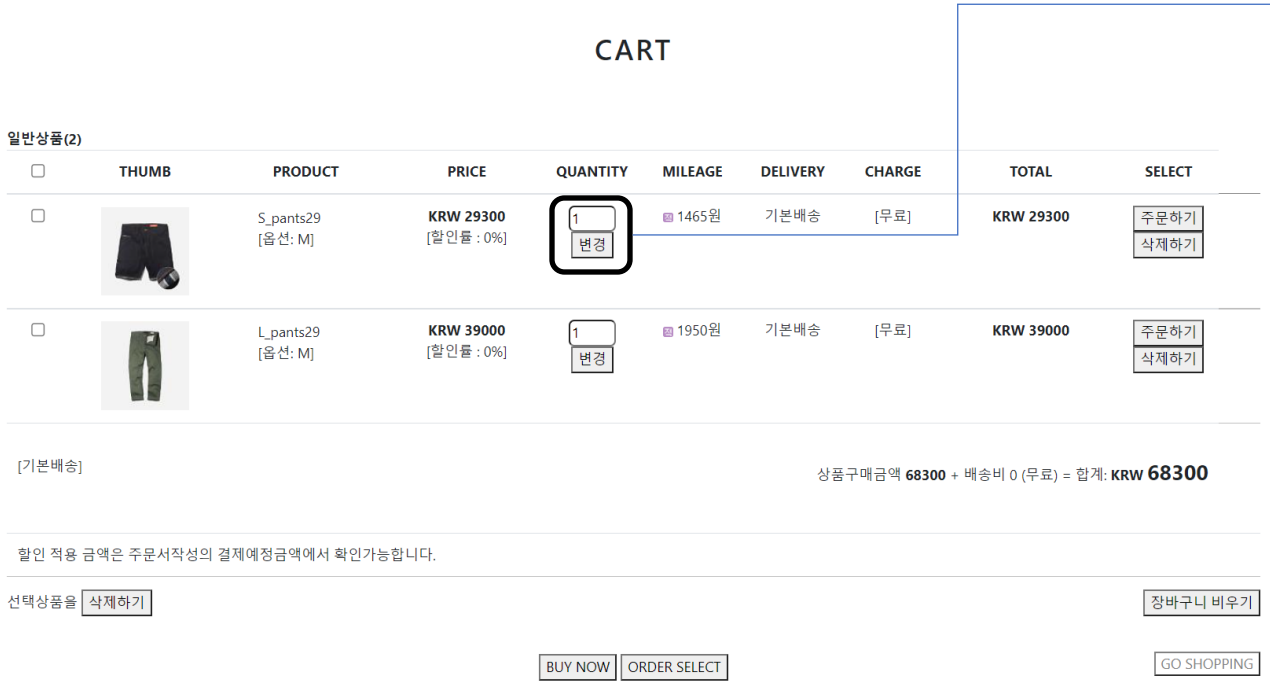
```
<resultMap id="cartfragments" type="CartVO">
    <id column="cartNum" property="cartNum"/>
    <result column="productNum" property="productNum"/>
    <result column="pInfoNum" property="pInfoNum"/>
    <result column="productCount" property="productCount"/>
    <result column="username" property="username"/>
    <association property="productVO" javaType="ProductVO">
        <id column="productNum" property="productNum"/>
        <result column="productTitle" property="productTitle"/>
        <result column="productPrice" property="productPrice"/>
        <result column="productDisRate" property="productDisRate"/>
        <result column="productDivNum" property="productDivNum"/>
        <result column="finalPrice" property="finalPrice"/>
        <result column="productMileage" property="productMileage"/>
        <result column="productSaleable" property="productSaleable"/>
    </association>
    <association property="productInfoVO" javaType="ProductInfoVO">
        <id column="pInfoNum" property="pInfoNum"/>
        <result column="productNum" property="productNum"/>
        <result column="size" property="size"/>
        <result column="stock" property="stock"/>
    </association>
    <association property="productFileVO" javaType="ProductFileVO">
        <id column="fileNum" property="fileNum"/>
        <result column="productNum" property="productNum"/>
        <result column="fileName" property="fileName"/>
        <result column="oriName" property="oriName"/>
    </association>
</resultMap>
```

```
<select id="getCartList" parameterType="MemberVO" resultMap="cartfragments">
  SELECT cf.*, p.*, i.*
      from (SELECT c.*, pf.fileName
          FROM cart c left JOIN (select * from productFiles where fileName LIKE CONCAT('T\_', '%')) pf
          ON c.productnum=pf.productNum WHERE c.username=#{username}) cf
      INNER JOIN product p ON cf.productNum=p.productNum
      INNER JOIN productinfo i on cf.pInfoNum=i.pInfoNum
</select>
```

해당 user id를 이용하여 장바구니 항목을 가져온다.
동시에 join문을 이용하여 다양한 table의 데이터를
resultMap을 이용하여 한번에 가져온다.

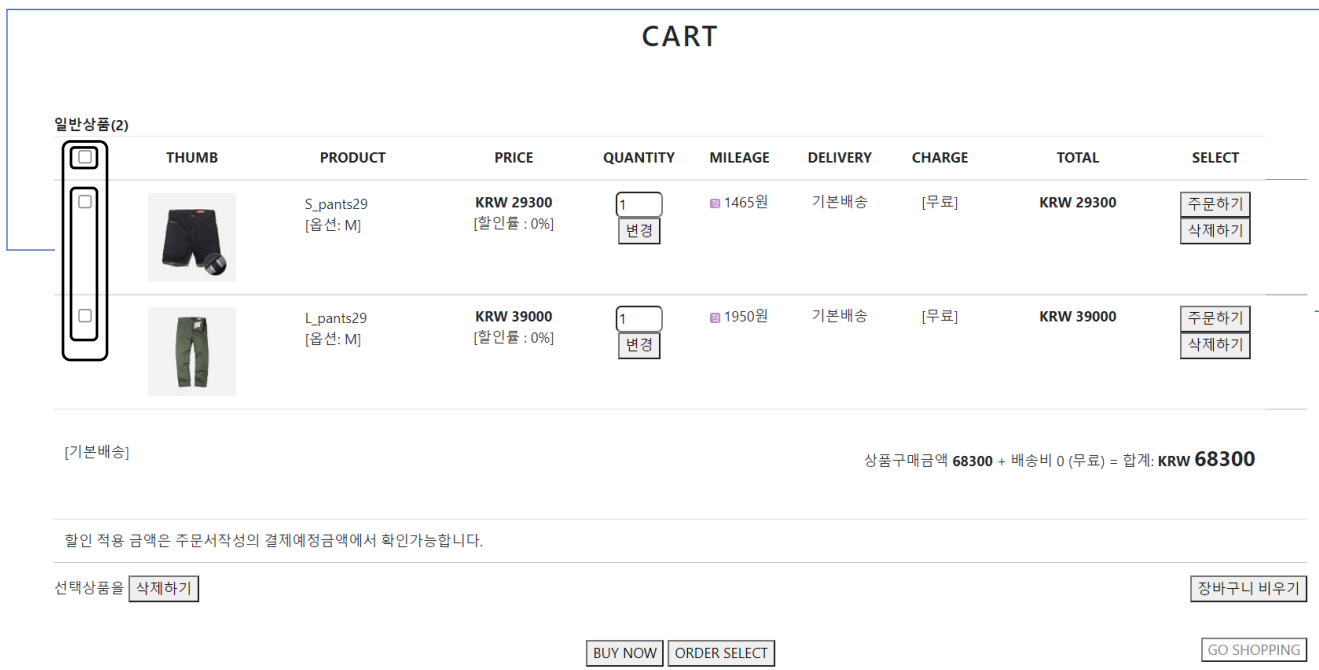
```
@GetMapping("basket")
public String getCartList(Authentication authentication, Model model, RedirectAttributes redirectAttributes) throws Exception{
    String msg = "";
    MemberVO memberVO = new MemberVO();
    memberVO.setUsername(((UserDetails)authentication.getPrincipal()).getUsername());
    List<CartVO> getlist = cartService.getCartList(memberVO);
    List<CartVO> deleteList = new ArrayList<>();
    List<CartVO> updatelist = new ArrayList<>();
    List<CartVO> list = new ArrayList<>();
    long cartCount = getlist.size();
    long totalprice = 0;
    if(updatelist.size()>0) {
        msg="몇몇 상품의 재고가 부족합니다.";
        if(cartService.updateCountList(list)==0) {
            msg = "서버와의 통신이 불안정합니다.";
            redirectAttributes.addFlashAttribute("msg", msg);
            return "redirect:/";
        }
    }

    if(deleteList.size()>0) {
        if(msg.equals("몇몇 상품의 재고가 부족합니다.")) {
            msg = "구입이 불가능 한 상품이 제거되고, 부족한 재고의 상품의 개수가 변경되었습니다.";
        }else {
            msg = "구입이 불가능한 상품이 자동으로 제거되었습니다.";
        }
        if(cartService.deleteItem(deleteList)==0) {
            msg = "서버와의 통신이 불안정합니다.";
            redirectAttributes.addFlashAttribute("msg", msg);
            return "redirect:/";
        }
    }

    model.addAttribute("msg", msg);
    model.addAttribute("cartCount", cartCount);
    model.addAttribute("items", list);
    model.addAttribute("totalprice", totalprice);
    return "/order/basket";
}
```

장바구니 항목 중
재고가 없는 것이
있다면 제거,
재고가 있으면
최대치로 변경.
실패시
메시지와 함께
인덱스로 이동




```javascript
//장바구니에 있는 상품의 구매 개수 조절
$(".countchange").click(function(event){
    event.preventDefault();

    const typenumber=$(this).siblings('input[type="number"]');
    const typehidden=$(this).siblings('input[type="hidden"]');
    const productCount = typenumber.val();
    const cartNum = typehidden.attr("data-cartNum");
    const pInfoNum = typehidden.attr("data-pInfoNum");

    if(productCount<1){
        swal({
            icon:"error",
            title:"실패",
            text:"0 이하의 숫자는 입력할 수 없습니다."
        });
        return;
    }
    $.get({
        url:"/cart/updateCount",
        data:{
            productCount:productCount,
            cartNum:cartNum,
            pInfoNum:pInfoNum
        },
        success:function(result){
            if(result>0){
                location.href="/order/basket";
            }else{
                swal({
                    icon:"error",
                    title:"실패",
                    text:"현재 재고보다 수량이 더 많습니다."
                });
                typenumber.val(typehidden.attr("data-productCount"));
            }
        }
    });
});
```

Cart에서 선택한 제품의 구매 개수를 수정한다.
선택한 개수가 재고보다 많거나, 0이하라면 update하지 않고 경고창을 띄운다.
그 외의 조건에서는 Cart테이블에서 해당 항목의 구매 예정 개수를 update한다.




```html
<thead class="bd100">
    <tr>
        <th>
            <input class="checkbox" type="checkbox" id="allcheck"/>
        </th>
<c:forEach items="${items}" var="item">
    <tr class="bd50">
        <td>
            <input type="checkbox" class="selectcheck checkbox"
                   data-cartNum='<c:out value="${item.cartNum}"></c:out>'
                   name="checklist">
        </td>
```

```jsp
//전체 체크박스 선택
$("#allcheck").click(function(){
    $(".selectcheck").each(function(){
        $(this).prop("checked", $("#allcheck").prop("checked"));
    });
});

//체크박스 선택
$(".selectcheck").click(function(){
    let check = true;
    $(".selectcheck").each(function(){
        if(!$(this).prop("checked"))    {
            check=false;
        }
    });
    $("#allcheck").prop("checked", check);
});
```

장바구니에서 상품을 선택하는 checkbox가 존재한다.
맨 위에 thead의 checkbox는 모든 항목을 선택하는 checkbox이고,
아래의 checkbox들은 단일 항목을 선택하는 checkbox이다.
이 단일 checkbox들은 각각의 상품에 대한 장바구니 번호를 가지고 있다.(data-cartNum)
이 번호 들은 장바구니에서 체크된 상품들의 제거나, 체크된 항목 구매에 사용된다.

CART

일반상품(2)

	THUMB	PRODUCT	PRICE	QUANTITY	MILEAGE	DELIVERY	CHARGE	TOTAL	SELECT
☐		S_pants29 [옵션: M]	KRW 29300 [할인률 : 0%]	1 변경	1465원	기본배송	[무료]	KRW 29300	주문하기 삭제하기
☐		L_pants29 [옵션: M]	KRW 39000 [할인률 : 0%]	1 변경	1950원	기본배송	[무료]	KRW 39000	주문하기 삭제하기

[기본배송]

상품구매금액 68300 + 배송비 0 (무료) = 합계: KRW 68300

할인 적용 금액은 주문서작성의 결제예정금액에서 확인가능합니다.

선택상품을 삭제하기

장바구니 비우기

BUY NOW ORDER SELECT

GO SHOPPING

```java
//장바구니에서 체크한 항목 제거
$("#selectcartremove").click(function(event){
    event.preventDefault();

    let count = 0;
    let cartNums = new Array();

    $(".selectcheck").each(function(){
        if($(this).prop("checked")){
            count++;
            cartNums.push($(this).attr("data-cartNum"));
        }
    });

    if(count==0){
        swal({
            icon:"info",
            title:"INFO",
            text:"선택된 항목이 없습니다."
        });
    }
```

```java
$.post({
    url:"/cart/deleteItems",
    traditional:true,
    data : {
        cartNums:cartNums
    },
    success:function(result){
        if(result>0){
            swal({
                icon:"success",
                title:"삭제 완료",
                text:"선택된 항목을 비웠습니다."
            });
            location.href="/order/basket";
        }else{
            swal({
                icon:"error",
                title:"에러발생",
                text:"장바구니를 비우는 동안 에러가 발생했습니다."
            });
        }
    }
});
});
```

```mybatis
<delete id="deleteItem" parameterType="java.util.List">
  delete from cart where
  <foreach collection="list" item="item" separator="or" open="(" close=")">
      cartNum=#{item.cartNum} and username = #{item.username}
  </foreach>
</delete>
```

```java
//장바구니에서 모든 항목 제거
$("#allcartremove").click(function(event){
    event.preventDefault();

    let count = 0;

    $(".selectcheck").each(function(){
        count++;
    });

    if(count==0){
        swal({
            icon:"info",
            title:"INFO",
            text:"장바구니가 이미 비어있습니다."
        });
    }
```

```java
$.get({
    url:"/cart/deleteCart",
    success:function(result){
        if(result>0){
            swal({
                icon:"success",
                title:"삭제 완료",
                text:"장바구니를 모두 비웠습니다."
            });
            location.href="/order/basket";
        }else{
            swal({
                icon:"error",
                title:"에러발생",
                text:"장바구니를 비우는 동안 에러가 발생했습니다."
            });
        }
    }
});
});
```

```mybatis
<delete id="deleteCart" parameterType="MemberVO">
  delete from cart where username=#{username}
</delete>
```

카트에 있는 항목 전체 삭제는 username을 이용해서
해당 유저의 모든 장바구니 항목을 지운다.
선택된 항목 제거의 경우 checkbox의 check된 항목의
Data-cartNum 속성을 이용해서 데이터를 모아 보낸 후
해당 항목을 제거한다.

CART

일반상품(2)

	THUMB	PRODUCT	PRICE	QUANTITY	MILEAGE	DELIVERY	CHARGE	TOTAL	SELECT
☐		S_pants29 [옵션: M]	KRW 29300 [할인률 : 0%]	1 변경	1465원	기본배송	[무료]	KRW 29300	주문하기 삭제하기
☐		L_pants29 [옵션: M]	KRW 39000 [할인률 : 0%]	1 변경	1950원	기본배송	[무료]	KRW 39000	주문하기 삭제하기

[기본배송]

상품구매금액 **68300** + 배송비 0 (무료) = 합계: **KRW 68300**

할인 적용 금액은 주문서작성의 결제예정금액에서 확인가능합니다.

선택상품을 삭제하기

장바구니 비우기

BUY NOW ORDER SELECT

GO SHOPPING

해당 항목은 하나만 주문, 삭제이지만
똑같이 리스트에 담아서 주문, 삭제한다.

```
//선택된 항목 하나만 주문
$(".oneOrder").click(function(event){
    event.preventDefault();

    let cartNums = new Array();

    cartNums.push($(this).attr("data-cartNum"));

    $("body").append('<form id="transferorderform" action="/order/orderform" method="post"></form>');
    for(let cartNum of cartNums){
        $("#transferorderform").append('<input type="hidden" name="cartNums" value="'+cartNum+'">');
    }
    $("#transferorderform").submit();
});
```

```
//선택된 상품을 구매
$("#buyselects").click(function(event){
    event.preventDefault();

    let cartNums = new Array();
    let count = 0;
    $(".selectcheck").each(function(){
        if($(this).prop("checked")){
            cartNums.push($(this).attr("data-cartNum"));
            count++;
        }
    });

    if(count==0){
        swal({
            icon:"error",
            title:"실패",
            text:"선택된 상품이 없습니다."
        });
        return;
    }

    $("body").append('<form id="transferorderform" action="/order/orderform" method="post"></form>');
    for(let cartNum of cartNums){
        $("#transferorderform").append('<input type="hidden" name="cartNums" value="'+cartNum+'">');
    }
    $("#transferorderform").submit();
});
```

```
//모든 상품 구매
$("#buyall").click(function(event){
    event.preventDefault();

    let count = 0;
    let cartNums = new Array();

    $(".selectcheck").each(function(){
        cartNums.push($(this).attr("data-cartNum"));
        count++;
    });

    if(count==0){
        swal({
            icon:"error",
            title:"실패",
            text:"구매할 상품이 없습니다."
        });
        return;
    }

    $("body").append('<form id="transferorderform" action="/order/orderform" method="post"></form>');
    for(let cartNum of cartNums){
        $("#transferorderform").append('<input type="hidden" name="cartNums" value="'+cartNum+'">');
    }
    $("#transferorderform").submit();
});
```

BUY NOW는 cart에 있는
모든 항목을 구매한다.
ORDER SELECT는 선택된 항목들을
보이지 않는 form안에 넣어
Cartnum들을 전송하여
주문페이지로 넘어간다.

둘다 똑같은 주소로 보내지만, 모두 구매, 선택된 항목들만
구매에 따라 checkbox 항목 모두 보내느냐, checked된
cartNum만 보내느냐가 달라진다.

Order Form

```java
//주소 추가작업 필요
@PostMapping("orderform")
public String getPurchase(long[] cartNums, Authentication authentication, Model model,
                                                                RedirectAttributes redirectAttributes)throws Exception {

    String msg = "";
    List<CartVO> clist = new ArrayList<>();
    List<CartVO> list = new ArrayList<>();
    List<CartVO> dlist = new ArrayList<>();
    List<CartVO> updatelist = new ArrayList<>();
    MemberVO memberVO = new MemberVO();
    memberVO.setUsername(((UserDetails)authentication.getPrincipal()).getUsername());
    memberVO = memberService.getUsernameandemail(memberVO);
    memberVO.emailSeperator();

    for(long cartNum : cartNums) {
        CartVO cartVO = new CartVO();
        cartVO.setCartNum(cartNum);
        cartVO.setUsername(((UserDetails)authentication.getPrincipal()).getUsername());
        clist.add(cartVO);
    }

    clist = cartService.getCartSelectList(clist);
    for(CartVO cartVO : clist) {
        ProductVO productVO = cartVO.getProductVO();
        ProductInfoVO productInfoVO = cartVO.getProductInfoVO();
        if(!productVO.getProductSaleable()||productInfoVO.getStock()<=0) {
            dlist.add(cartVO);
        }else {
            list.add(cartVO);
            if(cartVO.getProductCount()>productInfoVO.getStock()) {
                updatelist.add(cartVO);
            }
        }
    }

    if(updatelist.size()>0) {
        msg="몇몇 상품의 재고가 부족합니다.";
        if(cartService.updateCountList(list)==0) {
            msg = "서버와의 통신이 불안정합니다.";
            redirectAttributes.addFlashAttribute("msg", msg);
            return "redirect:/";
        }
    }
    if(dlist.size()>0) {
        if(msg.equals("몇몇 상품의 재고가 부족합니다.")) {
            msg = "구입이 불가능 한 상품이 제거되고, 부족한 재고의 상품의 개수가 변경되었습니다.";
        }else {
            msg = "구입이 불가능한 상품이 자동으로 제거되었습니다.";
        }
        if(cartService.deleteItem(dlist)==0) {
            msg = "서버와의 통신이 불안정합니다.";
            redirectAttributes.addFlashAttribute("msg", msg);
            return "redirect:/";
        }
    }

    long orderCount = list.size();

    long totalprice=0;
    long totalMileage=0;
    int count = -1;
    String name = "";
    name += list.get(0).getProductVO().getProductTitle();
```

재고와 구매 예정 개수를 비교하여 재고가 더 적으면 변경하는데, 해당 작업이 실패하면 홈으로 보내고 redirectAttricutes를 통해 전송된 메시지를 띄우게 한다.

```java
    for(CartVO cartVO : list) {
        totalprice+=cartVO.getProductCount()*cartVO.getProductVO().getProductPrice();
        ProductVO productVO = cartVO.getProductVO();
        if(productVO.getProductDisRate()>0) {
            long cal = (((100-productVO.getProductDisRate())*productVO.getProductPrice()/100)/100)*100
            cartVO.setFinalPrice(cal*cartVO.getProductCount());
            totalprice -= (productVO.getProductPrice()-cal)*cartVO.getProductCount();
        }
        totalMileage += productVO.getProductMileage();
        count++;
    }

    if(count>0) {
        name+=" 외 "+count+"벌";
    }

    List<AddressVO> addresslist = addressService.getAddressList(memberVO);
    List<AddressVO> addrList = new ArrayList<>();
    AddressVO orderAddr = new AddressVO();
    AddressVO recentAddr = new AddressVO();
    for(AddressVO addressVO : addresslist) {
        addressVO.phoneSeperator();
        addressVO.concatAddress();
        if(!addressVO.getOrderAddr()) {
            orderAddr = addressVO;
        }else if(addressVO.getRecentUse()){
            recentAddr = addressVO;
            addrList.add(recentAddr);
        }else {
            addrList.add(addressVO);
        }
    }

    List<CouponVO> cuList = couponService.getCouponList(memberVO)

    model.addAttribute("name", name);
    model.addAttribute("cuList", cuList);
    model.addAttribute("recentAddr", recentAddr);
    model.addAttribute("orderAddr", orderAddr);
    model.addAttribute("orderDetail", memberVO);
    model.addAttribute("addrList", addrList);
    model.addAttribute("totalMileage", totalMileage);
    model.addAttribute("orderCount", orderCount);
    model.addAttribute("items", list);
    model.addAttribute("totalprice", totalprice);
    model.addAttribute("msg", msg);

    return "/order/orderform";
}
```

장바구니에서 cartNum들을 보내왔기 때문에 해당 cartNum들을 가져와서 장바구니처럼 조회한다. 그것 뿐만 아니라 결제에 필요한 해당유저에 대한 정보들도 model로 뿌려주게 된다.
결제에 필요한 정보들은 각각 해당 유저의 addresslist, 마일리지, 쿠폰 리스트 등 db에서 가져와야 하는 정보 뿐만 아니라 계산이 필요한 정보들까지 계산하여 view page에 뿌려준다.

Orderform – Product Table

```
<div style="text-align: left;">
    <strong>일반상품(<c:out value="${orderCount}"></c:out>)</strong>
</div>
```

ORDER

Controller에서 해당 상품의 개수를 받아온다.

일반상품(2)

THUMB	PRODUCT	PRICE	QUANTITY	MILEAGE	DELIVERY	CHARGE	TOTAL
	S_pants29 [옵션: M]	KRW 29300 [할인률 : 0%]	1	1465원	기본배송	[무료]	KRW 29300
	L_pants29 [옵션: M]	KRW 39000 [할인률 : 0%]	1	1950원	기본배송	[무료]	KRW 39000

[기본배송]

상품구매금액 68300+배송비 0 (무료) = 합계: KRW 68300

할인 적용 금액은 주문서작성의 결제예정금액에서 확인가능합니다.

Controller에서 모든 상품들의 가격합을 계산하여 보내주었다.

```
<tbody style="text-align: center; border-bottom: 1px gray solid;">
    <c:forEach items="${items}" var="item">
        <tr class="productDataList"
            data-productNum='<c:out value="${item.productNum}"></c:out>'
            data-pInfoNum='<c:out value="${item.PInfoNum}"></c:out>'
            data-finalPrice='<c:out value="${item.productVO.finalPrice}"></c:out>'
            data-productCount='<c:out value="${item.productCount}"></c:out>'
            data-cartNum='<c:out value="${item.cartNum}"></c:out>'
            style="border-bottom: 1px rgba(128,128,128,0.5) solid;">
            <td style="padding:1%;">
                <img alt="" src="${pageContext.request.contextPath}/resources/images/product/
                                ${item.productVO.productNum}/${item.productFileVO.fileName}"
                    width="100px" height="100px">
            </td>
```

장바구니처럼 mapper에서 가져온 cart 데이터들로 Product의 list를 띄운다.
다만 checkbo가 없고, 해당 forEach문에서 tr태그에 product정보들을 넣어두었다.
결제시에 필요한 상품 정보들에 대한 것이 해당 태그에 전부 존재하기 때문에 간편하게 정보들을 가져올 수 있다.

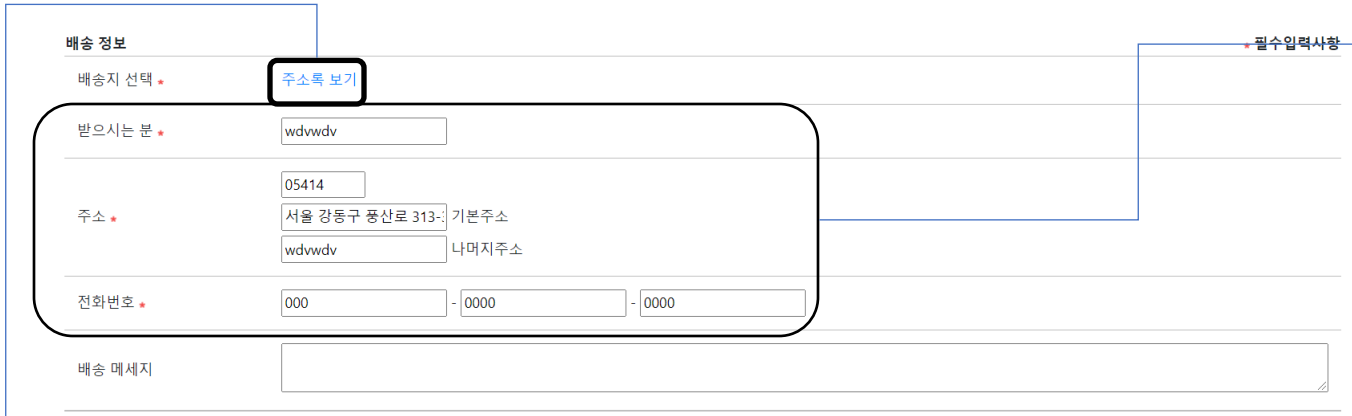
Order Form – Order Information

```java
List<AddressVO> addresslist = addressService.getAddressList(memberVO);
List<AddressVO> addrList = new ArrayList<>();
AddressVO orderAddr = new AddressVO();
AddressVO recentAddr = new AddressVO();
for(AddressVO addressVO : addresslist) {
    addressVO.phoneSeperator();
    addressVO.concatAddress();
    if(!addressVO.getOrderAddr()) {
        orderAddr = addressVO;
    }else if(addressVO.getRecentUse()){
        recentAddr = addressVO;
        addrList.add(recentAddr);
    }else {
        addrList.add(addressVO);
    }
}

List<CouponVO> cuList = couponService.getCouponList(memberVO);

model.addAttribute("name", name);
model.addAttribute("cuList", cuList);
model.addAttribute("recentAddr", recentAddr);
model.addAttribute("orderAddr", orderAddr);
model.addAttribute("orderDetail", memberVO);
model.addAttribute("addrList", addrList);
model.addAttribute("totalMileage", totalMileage);
model.addAttribute("orderCount", orderCount);
model.addAttribute("items", list);
model.addAttribute("totalprice", totalprice);
model.addAttribute("msg", msg);

return "/order/orderform";
}
```

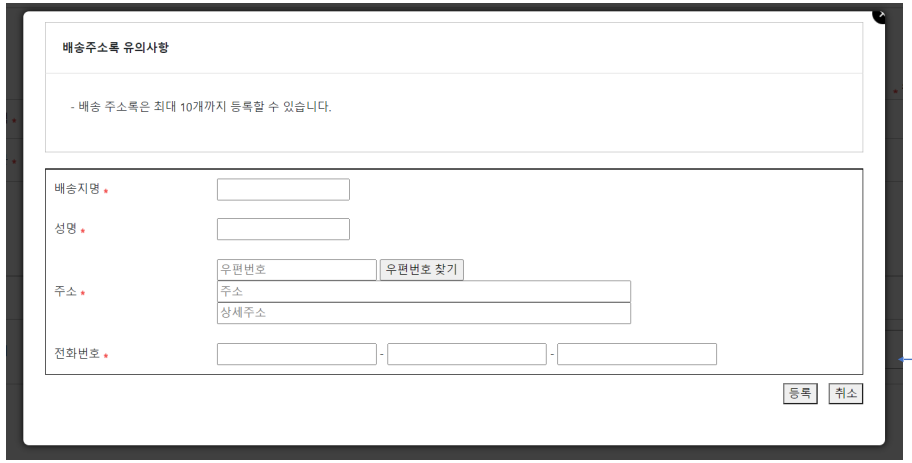
orderAddr이 0인 데이터는 주문자 자신의 정보로, 배송지로는 사용되지 않는다.
해당 user id(username)으로 모든 주소를 가져오는데 DB에 bit로 저장되어 있는
orderAddr과 recentUse라는 컬럼을 통해 해당 주소가 어떤 주소인지 알 수 있다.
orderAddr이 0이고, recentUse가 0인 경우 해당 유저의 정보로 유저가 member profile에서
수정하지 않는 이상 항상 주문자 정보에 들어가게 된다.
orderAddr이 1이라면 배송지 주소목록을 나타내며, recentUse가 1이면 최근에 해당 주소를
받는 곳으로 설정했다는 의미가 된다.

Order Form - destination

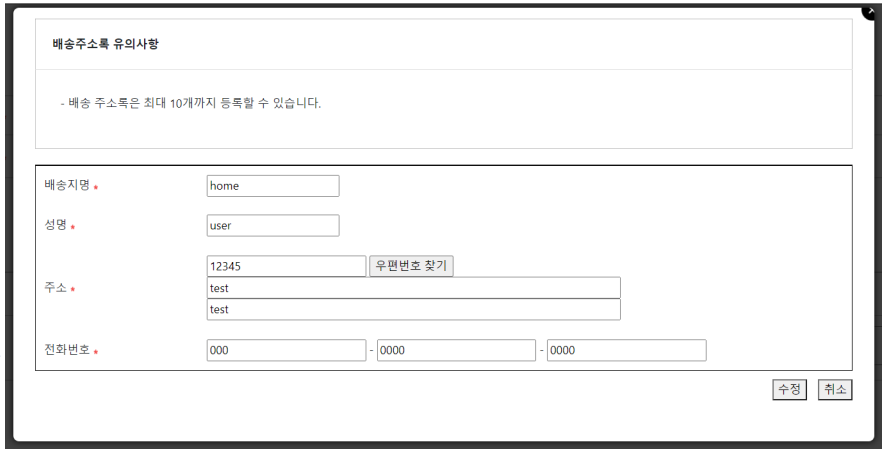
recentUse가 1인(true)인 항목이 자동으로 들어간다.
만약 없다면 공백으로 나오게 되고, 결제할 수 없다.



해당 버튼을 누르면 주소관련 모달창이 뜬다.
그곳에서 주소선택, update, 삭제, 추가 등을 할 수 있다.



둘은 같은 form으로
버튼과 입력된 값의
유무를 제외하고
다른 것이 없다.



Order Form - Address Modal - Address List

적용버튼을 누르면 modal창이 꺼지면서
배송 정보에 해당 정보들이 들어가게 된다.
또한 해당 항목을 제외한 주소들의
recentUse를 0으로 만들고,
선택된 주소의 recentUse를 1로 만든다.

```
<update id="recentUseUpdate" parameterType="AddressVO">
  update address set recentUse = case addrNum when #{addrNum} then 1 else 0 end
  where username=#{username}
</update>
```

수정이나 등록 버튼을 누르면
insert form으로 전환된다.
해당 모달창에 list와 insert div
두 개가 존재하는데, 서로 번갈아가며
Display:none과 display:block이 교체된다.

```
$("#addressAllCheck").click(function(){
    let value = $("#addressAllCheck").prop("checked");

    $(".addressSelect").each(function(){
        $(this).prop("checked", value);
    });
});

$(".addressSelect").click(function(){
    let check = true;

    $(".addressSelect").each(function(){
        if(!$(this).prop("checked")){
            check=false;
        }
    });

    $("#addressAllCheck").prop("checked", check);
});
```

주소를 선택하는
Checkbox를 컨트롤

선택된 주소들을 삭제하는 버튼,
Checkbox에 주소 번호가 들어 있어
선택된 번호들을 모아 Ajax로 보내 제거한 후
Ajax로 리스트를 가져와 다시 address를
띄워준다.

```
<update id="recentDeleteUpdate" parameterType="AddressVO">
  update address set recentUse = case addrNum when (select Max(addrNum) from address where username=#{username}) then 1 else 0 end
</update>
```

Order Form – Address Modal - selectedDelete

```java
$.get({
    traditional:true,
    url:"/address/deleteSelect",
    data:{
        addrNums:addrNums
    },
    success:function(result){
        if(result != ""){
            let parseJson = JSON.parse(result);
            $("#recPeople").val(parseJson.recipient);
            $("#recZipcode").val(parseJson.zipCode);
            $("#recBasic").val(parseJson.basicAddr);
            $("#recDetail").val(parseJson.detailAddr);
            $("#reccall1").val(parseJson.addrPhone1);
            $("#reccall2").val(parseJson.addrPhone2);
            $("#reccall3").val(parseJson.addrPhone3);
            $.get({
                url:"/address/getAddressList",
                success:function(result){
                    if(result[0]==""){
                        let nothingthere = "<tr><td colspan='6' style='text-align: center;'>등록된 주소가 없습니다.</td></tr>";
                        $("#settingAddressList").empty();
                        $("#settingAddressList").append(nothingthere);
                    }else{
                        $("#settingAddressList").empty();
                        for(let datas of result){
                            let data = JSON.parse(datas);

                            let trform = '<tr data-addrNum="'+data.addrNum+'"></tr>';
                            $("#settingAddressList").append(trform);
                            let target = $("#settingAddressList").find('tr[data-addrNum="'+data.addrNum+'"]');

                            let tdform = '<td><input type="checkbox" class="addressSelect" data-addrNum="'+data.addrNum+'"></td>';
                            target.append(tdform);
                            tdform = '<td>'+data.addrName+'</td>';
                            target.append(tdform);
                            tdform = '<td>'+data.recipient+'</td>';
                            target.append(tdform);
                            tdform = '<td>'+data.addrPhone+'</td>';
                            target.append(tdform);
                            tdform = '<td>'+data.fullAddress+'</td>';
                            target.append(tdform);
                            tdform = '<td><button class="addressAdapt" data-addrNum='+data.addrNum+'>적용</button>'
                                    +'<br><button class="addressRepareForm" data-addrNum='+data.addrNum+'>수정</button></td>';
                            target.append(tdform);
                        }
                    }
                }
            }
        }else{
            $("#recPeople").val('');
            $("#recZipcode").val('');
            $("#recBasic").val('');
            $("#recDetail").val('');
            $("#reccall1").val('');
            $("#reccall2").val('');
            $("#reccall3").val('');
            swal({
                icon:"error",
                title:"삭제 실패",
                text:"삭제하는 도중 서버에서 오류가 발생하였습니다."
            });
        }
    }
});
});
```

배열에 넣어둔 addrNums로 해당 목록을 지운 후, 번호가 제일 큰 주소에 recentUse를 1로 두고, 해당 항목을 가져와서 뿌려준다.

```xml
<delete id="deleteSelect" parameterType="java.util.List">
  delete from address where
  <foreach collection="list" item="item" separator="or" open="(" close=")">
      username=#{item.username} and addrNum=#{item.addrNum}
  </foreach>
</delete>
```

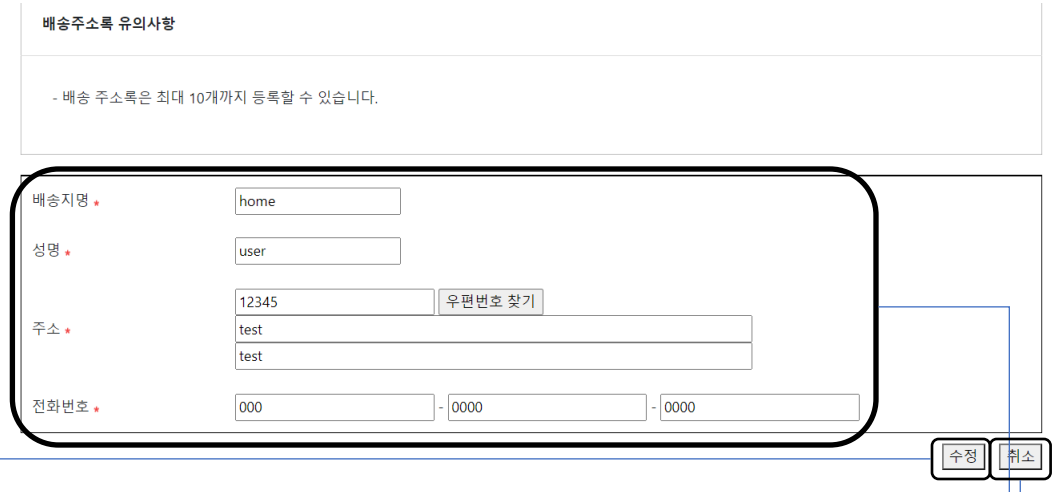
```xml
<update id="recentDeleteUpdate" parameterType="AddressVO">
  update address set recentUse = case addrNum when (select Max(addrNum) from address where username=#{username}) then 1 else 0 end
</update>
```

```xml
<select id="getSelectRecent" parameterType="AddressVO" resultType="AddressVO">
  select * from address where recentUse=1 and username=#{username}
</select>
```

```xml
<select id="getAddressList" parameterType="MemberVO" resultType="AddressVO">
  select * from address where username=#{username}
</select>
```

삭제를 진행하면 주소록이 변경되었기 때문에
새 주소록을 가져와 addressList div에 뿌려준다.
JSON데이터를 배열로 가지고 왔기 때문에
각각 하나씩 parsing 해주고 태그를 만들어 넣어준다.

Order Form – Address Modal – Address Insert




```
$("#addressChapter1").css("display","none");
$("#addThisAddress").attr("data-role", "add");
$("#addThisAddress").empty();
$("#addThisAddress").text("등록");
$("#addressChapter2").css("display","contents");
```

```
$("#addressChapter1").css("display","none");
$("#addThisAddress").attr("data-role", "edit");
$("#addThisAddress").empty();
$("#addThisAddress").text("수정");
$("#addThisAddress").attr("data-addrNum", addrNum);
$("#addressChapter2").css("display","contents");
```

등록 버튼과 수정버튼의 차이.
Data-role의 속성값 차이로 구분을 하기 때문에
보내는 주소나 메커니즘이 달라지게 된다.

다음 지도 api를 이용하여
주소를 찾아 해당 input태그에 넣어준다.

취소 버튼을 누르면 해당 div태그의
Css가 display:none이 되고 list의 css가
Display:block이 된다.

수정을 눌렀을 때 해당 구역 address의
addressNum을 ajax를 통해 보내 해당
주소를 가져온 후 상세정보를
저 공간에 뿌려준다.

Order Form – Address Modal – Insert & Update

```java
//주소 추가 혹은 수정용
$("#addThisAddress").click(function(event){
    event.preventDefault();

    let error = false;
    const addrNum = $(this).attr("data-addrNum");
    const addrName = $("#insertAddressTitle").val();
    const recipient = $("#insertAddrRecipient").val();
    const zipCode = $("#sample2_postcode").val();
    const basicAddr = $("#sample2_address").val();
    const detailAddr = $("#sample2_detailAddress").val();
    const addrPhone1 = $("#insertAddrPhone1").val();
    const addrPhone2 = $("#insertAddrPhone2").val();
    const addrPhone3 = $("#insertAddrPhone3").val();
    const addrPhone = addrPhone1+"-"+addrPhone2+"-"+addrPhone3;

    let reg = new RegExp("[^\\s]{1,10}");
    if(!reg.test(addrName)){
        swal({
            icon:"warning",
            title:"형식에러",
            text:"공백이 아닌 1~10글자를 입력해 주세요."
        });
        return;
    }
    if(!reg.test(recipient)){
        swal({
            icon:"warning",
            title:"형식에러",
            text:"공백이 아닌 1~10글자를 입력해 주세요."
        });
        return;
    }
```

주소를 등록하는데 필요한 변수들을 각각 변수로 담은 이후 정규식으로 패턴을 만족하는지 확인하고, ajax를 통해 필요한 데이터들을 전송한다.

Update

```java
if($(this).attr("data-role")=='edit'){
    $.get({
        url:"/address/updateAddress",
        async:false,
        data:{
            addrNum:addrNum,
            addrName:addrName,
            recipient: recipient,
            zipCode: zipCode,
            basicAddr:basicAddr,
            detailAddr:detailAddr,
            addrPhone:addrPhone
        },
```

Insert

```java
$.get({
    url:"/address/checkCount",
    async:false,
```

주소의 최대 개수를 10개로 맞추었기에 현재 저장된 주소의 개수를 가져온다.

```java
$.get({
    url:"/address/setAddress",
    async:false,
    data:{
        addrName:addrName,
        recipient: recipient,
        zipCode: zipCode,
        basicAddr:basicAddr,
        detailAddr:detailAddr,
        addrPhone:addrPhone
    },
```

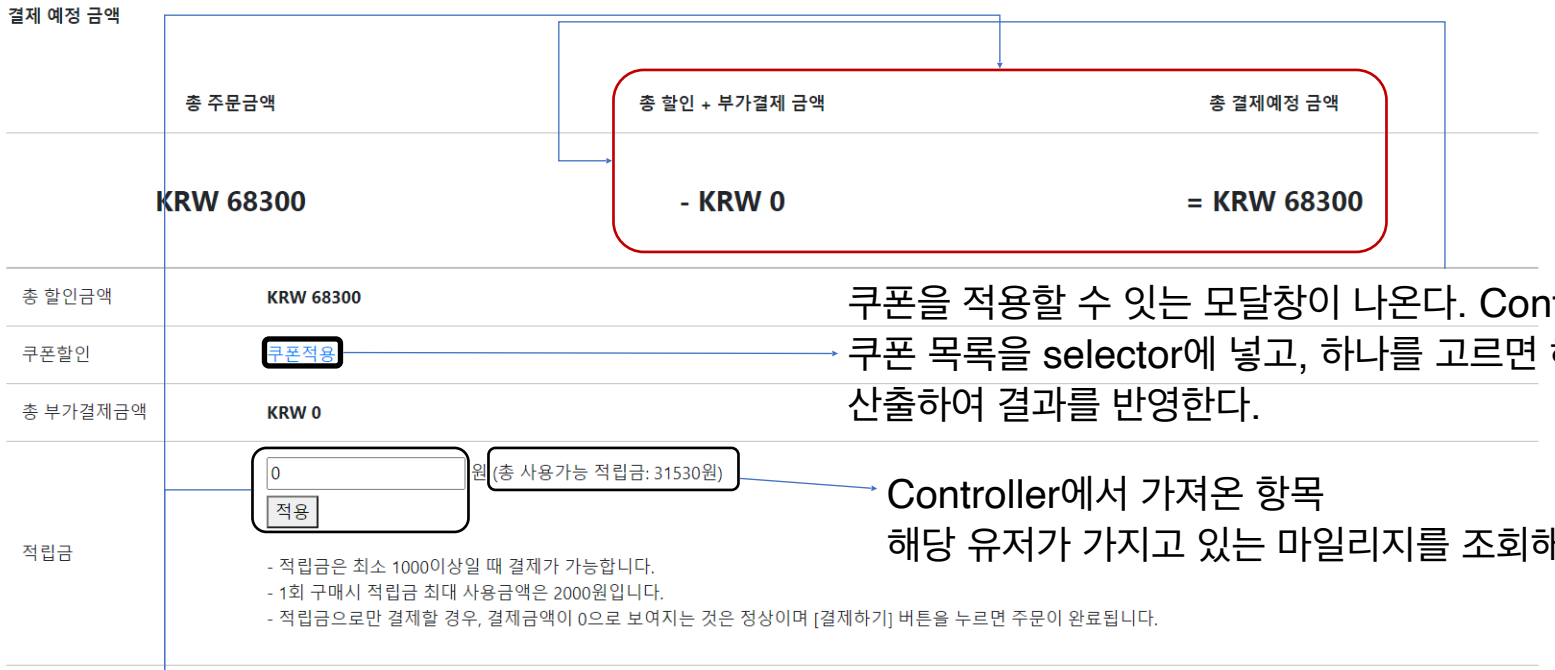
```java
$.get({
    url:"/address/getSelectRecent",
    async:false,
```

```java
$.get({
    url:"/address/getAddressList",
    async:false,
```

```java
    $("#addressChapter2").css("display","none");
    $("#addressChapter1").css("display","contents");
});
```

Update와 Insert과정을 거치면서 나온 결과를 가져 온 후, recentUse가 true인 주소와 Address List를 가져와서 배치한 후 Address List div가 보이게 css를 변경시킨다.

Order Form – Coupon & Mileage

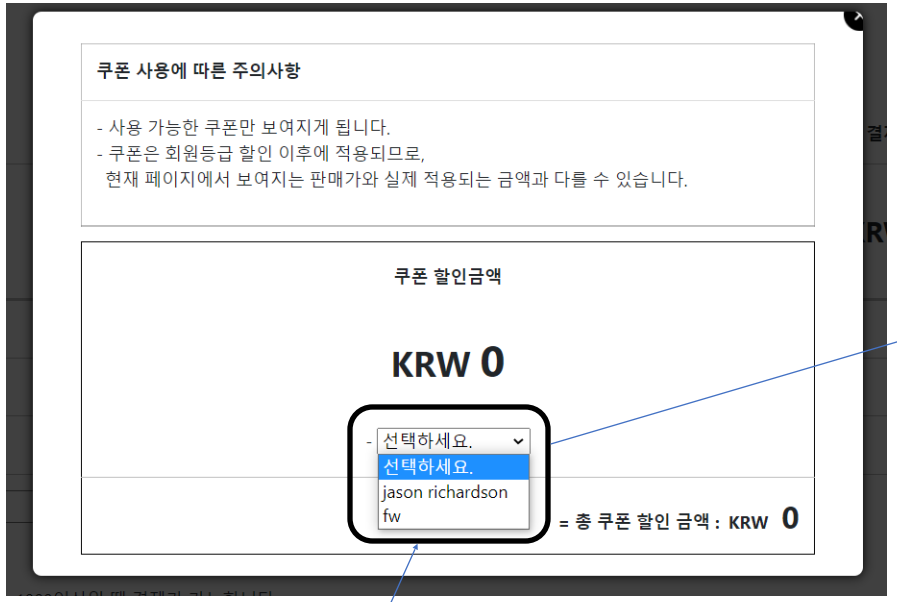


쿠폰을 적용할 수 잇는 모달창이 나온다. Controller에서 가져온 쿠폰 목록을 selector에 넣고, 하나를 고르면 해당 할인 가격을 산출하여 결과를 반영한다.

Controller에서 가져온 항목
해당 유저가 가지고 있는 마일리지를 조회해서 가져옴

마일리지를 사용할 수 있는 항목, 1000~2000원을 사용할 수 있고, 적용 버튼을 눌러야 적용된다.
총할인 + 부가결제 구역에 변화가 생기며, 총 결제 금액에 변화가 일어난다.
만약 적게 입력하거나 높게 입력한다면 적용되지 않는다.
쿠폰과 마일리지 할인은 기본 품목 할인가와 다르게 계산되고,
원래 총 결제 금액에서 비율을 산출해 그 둘을 합친 값을 결제하게끔 한다.

Order Form – Coupon Modal




```java
//쿠폰을 골랐다면 그만큼의 값이 변하도록 해주어야 한다.
$("#couponSelector").change(function(){
    const disRate = Number($("#couponSelector option:selected").attr("data-disRate"));
    const originTotPrice = Number($("#purchasebutton").attr("data-originTotPrice"));
    const mileSp = Number($("#purchasebutton").attr("data-mileSp"));
    const sale = Math.floor(originTotPrice*disRate/100/100)*100;
    const thisvalue = Number($("#couponSelector option:selected").val());

    $("#cuSalePrice1").text(sale);
    $("#cuSalePrice2").text(sale);
    $("#delPriceInsert").text(mileSp+sale)
    $("#purchasebutton").attr("data-cuSale", sale);
    $("#onsaleprice").text(mileSp+sale);
    $("#endprice").text(originTotPrice-(mileSp+sale));
    $("#forUserPrice").text(originTotPrice-(mileSp+sale));
    $("#purchasebutton").attr("data-cuNum", thisvalue);
});
```

Selected된 항목이 변경된다면 해당 function이 실행되게 된다.
쿠폰 할인률로 할인 가격을 구하고, 필요한 곳에 적용한다.
그리고 마일리지를 합쳐 총 할인가격으로 만든 후,
이를 결제버튼에 속성으로 넣어 쉽게 가져오고,
처리하기 좋게 만들었다.

```xml
<resultMap type="CouponVO" id="getCouponDetails">
    <id column="cuNum" property="cuNum"/>
    <result column="username" property="username"/>
    <result column="cuSpNum" property="cuSpNum"/>
    <result column="pubDate" property="pubDate"/>
    <result column="exDate" property="exDate"/>
    <result column="useCheck" property="useCheck"/>
    <association property="couponspVO" javaType="CouponspVO">
        <id column="cuSpNum" property="cuSpNum"/>
        <result column="cuName" property="cuName"/>
        <result column="disRate" property="disRate"/>
    </association>
</resultMap>

<select id="getCouponList" parameterType="MemberVO" resultMap="getCouponDetails">
    select c.*, cs.* from coupon c inner join couponsp cs on c.cuSpNum=cs.cuSpNum
    where c.username=#{username} and useCheck=0
</select>
```

Controller에서 Order Form으로 넘어 올 때
가져온 해당 유저의 쿠폰 목록이 담겨있다.
Result map으로 해당 coupon의 종류이름과 할인률 등이 담겨있다.

Order Form – Order Section

최종결제 금액

68300

☐ 결제정보를 확인하였으며, 구매진행에 동의합니다.

PAYMENT

적립예정금액 3415원

```
//구매버튼 누르기 전에 일단 form들에 자료가 전부 제대로 있는지 확인부터 해야한다.
$("#purchasebutton").click(function(event){
    event.preventDefault();

    if(!$("#agreepayment").prop("checked")){
        swal({
            icon:"warning",
            title:"필수 입력사항 누락",
            text:"약관동의에 체크해 주세요."
        });
        return;
    }
    let reg = new RegExp("[0-9]{4,5}");
    if(!reg.test($("#recZipcode").val())){
        swal({
            icon:"warning",
            title:"필수 입력사항 누락",
            text:"주소를 등록해주세요."
        });
        return;
    }
```

체크 혹은 배송 정보를
입력하지 않았다면
구매할 수 없다.

```
//시작하자마자 아임포트 설정이 될 수 있도록 처리
$(document).ready(function(){
    var IMP = window.IMP;
    IMP.init('imp27607634');
});
```

아임포트 api를 사용하기 위한 준비를 한 후
아임포트 결제 시스템을 실행시킨다.
아래는 결제를 완료하고 저장하기 위한 많은
매개변수들의 처리를 담고 있다.

```
IMP.request_pay({
    pg : 'html5_inicis',
    pay_method : 'card',
    merchant_uid : merchant_uid,
    name : name,
    amount : amount,
    buyer_email : buyeremail,
    buyer_name : buyername,
    buyer_tel : buyertel,
    buyer_addr : buyeraddr,
    buyer_postcode : buyerpostcode
}, function(rsp) {
    if(rsp.success){

      $.ajax({
          type:"post",
          url:"/order/vertify/"+rsp.imp_uid,
          success:function(data){
```

```
const date = new Date();
const merchant_uid = date.getTime() + '-' + date.getFullYear()+date.getMonth()+date.getDate()+date.getHours()
                                    +date.getMinutes()+date.getSeconds();

const totPrice = Number($(this).attr("data-originTotPrice"));
const amount = totPrice - Number($("#purchasebutton").attr("data-cuSale")) - Number($("#purchasebutton").attr("data-mileSp"));
const name = $(this).attr("data-name");
const buyeremail = $("#buyerEmail1").val()+"@"+$("#buyerEmail2").val()+"."+$("#buyerEmail3").val();
const buyername = $("#buyerName").val();
const buyertel = $("#buyerPhone1").val()+"-"+$("#buyerPhone2").val()+"-"+$("#buyerPhone3").val();
const buyeraddr = $("#buyerBasic").val()+' '+$("#buyerDetail").val();
const buyerpostcode = $("#buyerZipcode").val();
const cuNum = Number($("#purchasebutton").attr("data-cuNum"));
const destination = "(" + $("#recZipcod").val() + ")" + $("#recBasic").val() + " "+$("#recDetail").val();
const orderMessage = $("#orderMessage").val();
let changeMiles = new Array();
changeMiles.push(Number($("#purchasebutton").attr("data-mileSp")));
changeMiles.push(Number($("#purchasebutton").attr("data-changeMile")));
let productNums = new Array();
let pInfoNums = new Array();
let productCounts = new Array();
let finalPrices = new Array();
let cartNums = new Array();
$(".productDataList").each(function(){
    productNums.push(Number($(this).attr("data-productNum")));
    pInfoNums.push(Number($(this).attr("data-pInfoNum")));
    productCounts.push(Number($(this).attr("data-productCount")));
    finalPrices.push(Number($(this).attr("data-finalprice")));
    cartNums.push(Number($(this).attr("data-cartNum")));
});
```

```
$.post({
    url:"./orderComplete",
    traditional:true,
    data:{
        cuNum: cuNum,
        orderNum: merchant_uid,
        totPrice: totPrice,
        spPrice: amount,
        destination: destination,
        orderMessage: orderMessage,
        changeMiles: changeMiles,
        productNums: productNums,
        pInfoNums: pInfoNums,
        productCounts: productCounts,
        finalPrices: finalPrices,
        orderName: name,
        cartNums: cartNums
    },
    success:function(){
        $("body").append("<form id='sendOrderResult' action='./orderResult'></form>");
        $("#sendOrderResult").append("<input type='hidden' name='orderNum' value='"+merchant_uid+"'>");
        $("#sendOrderResult").submit();
    },
```

이후 결제 완료 처리를 ajax를 통해 진행한 후
완료된 결과를 조회하여 띄워주는 페이지로
이동시키기 위해 form 태그를 만들어 전송한다.

자바스크립트는 보안성이
떨어지기 때문에 api의
vertify 과정으로 java에서
결제 정보를 재 확인한다.

Order Complete

```java
//결제를 위한 메서드, 결제 검증을 위해 필요하다.
@ResponseBody
@PostMapping("vertify/{imp_uid}")
public IamportResponse<Payment> getVertify(@PathVariable(value="imp_uid")String imp_uid)
                                                                    throws IamportResponseException, IOException {

    return api.paymentByImpUid(imp_uid);
}
```

우선 결제를 진행하기 전, 확인하는 작업으로,
결제예정금액과, 결제된 금액을 비교하기 위해
아임포트 서버에서 결제 관련 내역을
가져오는 역할을 한다.
이후 database에 주문 내역을 저장한다.

```java
@PostMapping("orderComplete")
@Transactional(rollbackFor = Exception.class)
public void orderComplete(HttpServletRequest request, Authentication authentication)throws Exception{
    Long cuNum = Long.parseLong(request.getParameter("cuNum"));
    if(cuNum!=0) {
        CouponVO couponVO = new CouponVO();
        couponVO.setUsername(((UserDetails)authentication.getPrincipal()).getUsername());
        couponVO.setCuNum(cuNum);
        if(couponService.useUpdate(couponVO)==0) {
            throw new Exception();
        }
    }

    String destination = request.getParameter("destination");
    String orderMessage = request.getParameter("orderMessage");
    String orderNum = request.getParameter("orderNum");
    long spPrice = Long.parseLong(request.getParameter("spPrice"));
    long totPrice = Long.parseLong(request.getParameter("totPrice"));
    String orderName = request.getParameter("orderName");
    OrderlistVO orderlistVO = new OrderlistVO();
    orderlistVO.setCuNum(cuNum);
    orderlistVO.setDestination(destination);
    orderlistVO.setOrderMessage(orderMessage);
    orderlistVO.setOrderNum(orderNum);
    orderlistVO.setSpPrice(spPrice);
    orderlistVO.setTotPrice(totPrice);
    orderlistVO.setOrderName(orderName);
    orderlistVO.setUsername(((UserDetails)authentication.getPrincipal()).getUsername());
    if(orderService.setOrder(orderlistVO)==0) {
        throw new Exception();
    }
```

```java
String[] temppInfoNums = request.getParameterValues("pInfoNums");
int length = temppInfoNums.length;
long[] pInfoNums = new long[length];
for(int i = 0 ; i < length; i++) {
    pInfoNums[i] = Long.parseLong(temppInfoNums[i]);
}
String[] tempProductNums = request.getParameterValues("productNums");
long[] productNums = new long[length];
for(int i = 0 ; i < length; i++) {
    productNums[i] = Long.parseLong(tempProductNums[i]);
}
String[] tempProductCounts = request.getParameterValues("productCounts");
long[] productCounts = new long[length];
for(int i = 0 ; i < length; i++) {
    productCounts[i] = Long.parseLong(tempProductCounts[i]);
}
String[] tempFinalPrices = request.getParameterValues("finalPrices");
long[] finalPrices = new long[length];
for(int i = 0 ; i < length; i++) {
    finalPrices[i] = Long.parseLong(tempFinalPrices[i]);
}
List<PurchaseVO> purchaseVOs = new ArrayList<>();
for(int i = 0 ; i < length; i++) {
    PurchaseVO purchaseVO = new PurchaseVO();
    purchaseVO.setOrderNum(orderNum);
    purchaseVO.setPInfoNum(pInfoNums[i]);
    purchaseVO.setProductNum(productNums[i]);
    purchaseVO.setProductCount(productCounts[i]);
    purchaseVO.setProPriceSum(finalPrices[i]*productCounts[i]);
    purchaseVOs.add(purchaseVO);
}
```

워낙 데이터가 많아서 모든 매개변수 정확하게 넣어도
몇몇 중요 변수가 들어오지 않았기 때문에
Request를 이용해서 해당 내역들을 하나하나 받아
원래 데이터로 만들어 주는 작업을 진행하였고,
데이터베이스에 데이터 추가, 수정을 위해서
해당 데이터들을 종합하여 새로운 VO 객체를 만드는
작업을 진행했다.
우선 OrderVO를 만들어서 주문정보를 저장하는 작업을
진행하였다.

Throw new Exception()을
처리한 이유는 해당 조건이라면
DB에 제대로 적용이
되지 않았다는 것이기 때문이며,
지금까지 이루어졌던 작업을
되돌리는 Transaction의 조건을
Exception으로 두었기 때문이다.

```java
if(purchaseService.setOrderPurchase(purchaseVOs)==0) {
    throw new Exception();
}
```

Orderlist의 경우 주문 전체 정보로, 각각 어느 제품을 구매했는지 확인하기 위해 각각 품목들의 정보를 purchase 테이블에 넣는다.

```java
MileageVO mileageVO = new MileageVO();
mileageVO.setUsername(((UserDetails)authentication.getPrincipal()).getUsername());
mileageVO = mileageService.getRecentMileage(mileageVO);

if(mileageVO==null) {
    throw new Exception();
}

String[] tempChangeMiles = request.getParameterValues("changeMiles");
long[] changeMiles = new long[2];
for(int i = 0 ; i < 2; i++) {
    changeMiles[i] = Long.parseLong(tempChangeMiles[i]);
}
List<MileageVO> mileageVOs = new ArrayList<>();
if(changeMiles[0]!=0) {
    MileageVO mileageVO1 = new MileageVO();
    mileageVO.setOrderNum(orderNum);
    mileageVO.setChangeMile(-1*changeMiles[0]);
    mileageVO.setEnabledMile(mileageVO.getEnabledMile()-changeMiles[0]);
    mileageVO.setUsedMile(mileageVO.getUsedMile()+changeMiles[0]);
    mileageVO.setMileContents("구매시 사용한 마일리지");
    BeanUtils.copyProperties(mileageVO, mileageVO1);
    mileageVOs.add(mileageVO1);
}
MileageVO mileageVO2 = new MileageVO();
mileageVO.setOrderNum(orderNum);
mileageVO.setChangeMile(changeMiles[1]);
mileageVO.setEnabledMile(mileageVO.getEnabledMile()+changeMiles[1]);
mileageVO.setMileContents("구매 후 얻은 마일리지");
BeanUtils.copyProperties(mileageVO, mileageVO2);
mileageVOs.add(mileageVO2);

if(mileageService.setMileAfterOrder(mileageVOs)==0) {
    throw new Exception();
}
```

현재 가지고 있는 마일리지를 이용하여 mileage table을 추가하기에 우선 해당 username으로 최근 마일리지 내역을 가져온다.
사용한 마일리지, 얻은 마일리지 항목, 두개로 mileageVO를 만들어서 DB에 저장하는 작업을 진행한다.

```java
//productinfo 테이블에서 해당 재고들을 제거해주어야한다.
List<ProductInfoVO> productInfoVOs = new ArrayList<>();
for(int i = 0 ; i < length; i++) {
    ProductInfoVO productInfoVO = new ProductInfoVO();
    productInfoVO.setPInfoNum(pInfoNums[i]);
    productInfoVO.setProductNum(productNums[i]);
    productInfoVO.setStock(productCounts[i]);
    productInfoVOs.add(productInfoVO);
}
if(productService.updateStock(productInfoVOs)==0) {
    throw new Exception();
}
 String[] tempCartNums = request.getParameterValues("cartNums");
 long[] cartNums = new long[length];
 for(int i = 0 ; i < length; i++) {
     cartNums[i] = Long.parseLong(tempCartNums[i]);
 }
 List<CartVO> cartVOs = new ArrayList<>();
 for(int i = 0 ; i < length; i++) {
     CartVO cartVO = new CartVO();
     cartVO.setCartNum(cartNums[i]);
     cartVO.setUsername(((UserDetails)authentication.getPrincipal()).getUsername());
     cartVOs.add(cartVO);
 }
 cartService.deleteItem(cartVOs);
}
```

마지막으로 구매한 항목이 존재하기 때문에 productInfo 테이블에 Stock컬럼을 각각 구매한 량만큼 제거하는 작업이 필요하다.
이후 구매한 항목을 카트에서 제거하면 주문과정은 끝이 난다.
(카트에서 항목 제거하는 과정은 크게 중요하지 않기 때문에 Transaction에 추가하지 않았다.)

Order Result

주문이 성공적으로 완료되었습니다.
주문해주셔서 감사합니다.

주문번호	1625389530752-20216418530
주문내역	S_pants29 외 1벌
결제금액	32200 원
결제날짜	2021-07-04

홈으로 이동하기

주문 처리과정이 모두 끝나고
완료된 주문에 대한 정보를 간략히 다시
보여주는 작업이 진행된다.
가장 최근에 완료된 주문을 가져온다.

```java
@GetMapping("orderResult")
public String orderResult(OrderlistVO orderlistVO, Authentication authentication, Model model,
                                                                    RedirectAttributes redirectAttributes)throws Exception{
    orderlistVO.setUsername(((UserDetails)authentication.getPrincipal()).getUsername());
    orderlistVO = orderService.getOrder(orderlistVO);

    if(orderlistVO == null) {
        redirectAttributes.addFlashAttribute("msg", "결제정보를 조회할 수 없습니다.");
        return "redirect:/";
    }

    model.addAttribute("order", orderlistVO);
    return "/order/orderResult";
}
```

```xml
<resultMap type="OrderlistVO" id="orderAndMileage">
    <id column="orderNum" property="orderNum"/>
    <result column="totPrice" property="totPrice"/>
    <result column="spPrice" property="spPrice"/>
    <result column="cuNum" property="cuNum"/>
    <result column="username" property="username"/>
    <result column="destination" property="destination"/>
    <result column="orderDate" property="orderDate"/>
    <result column="orderState" property="orderState"/>
    <result column="orderMessage" property="orderMessage"/>
    <result column="orderName" property="orderName"/>
    <association property="mileageVO" javaType="MileageVO">
        <id column="mileNum" property="mileNum"/>
        <result column="usedMile" property="usedMile"/>
        <result column="changeMile" property="changeMile"/>
        <result column="username" property="username"/>
        <result column="orderNum" property="orderNum"/>
        <result column="mileContents" property="mileContents"/>
        <result column="enabledMile" property="enabledMile"/>
    </association>
</resultMap>
<select id="getOrder" parameterType="OrderlistVO" resultMap="orderAndMileage">
    select * from orderlist o inner join mileage m on o.orderNum = m.orderNum
    where o.username=#{username} and m.mileNum = (select max(mileNum) from mileage where orderNum=#{orderNum})
</select>
```

Order Cancle

```java
$(".orderCancle").click(function(){

    const orderNum = $(this).attr("data-orderNum");

    $.post({
        url:"/order/cancle",
        data:{
            merchant_uid:orderNum
        },
        success:function(result){
            if(result){
                alert("결제 취소에 성공하였습니다.");
                location.href="./orderList";
            }else{
                alert("결제 취소 도중 에러가 발생하였습니다.");
            }
        }
    });
});
```

```java
//삭제 작업을 위한 메서드, 후반에 완성시키자.
@ResponseBody
@PostMapping("cancle")
public boolean testCancelPaymentAlreadyCancelledMerchantUid(String merchant_uid, Authentication authentication) {
    String test_already_cancelled_merchant_uid = merchant_uid;
    CancelData cancel_data = new CancelData(test_already_cancelled_merchant_uid, false); //merchant_uid를 통한 전액취소
    cancel_data.setEscrowConfirmed(true); //에스크로 구매확정 후 취소인 경우 true설정
    OrderlistVO orderlistVO = new OrderlistVO();
    orderlistVO.setOrderNum(merchant_uid);
    Calendar calendar = Calendar.getInstance();
    Date tempdate = calendar.getTime();
    java.sql.Date date = new java.sql.Date(tempdate.getTime());

    try {
        orderlistVO = orderService.selectedOrder(orderlistVO);
        if(!date.toString().equals(orderlistVO.getOrderDate().toString())) {
            List<PurchaseVO> list = purchaseService.getPurAdmin(orderlistVO);
            Long result = adminService.cancledOrder(list);
            if(result==0) {
                return false;
            }
        }
    } catch (Exception e1) {
        // TODO Auto-generated catch block
        e1.printStackTrace();
        return false;
    }
```

```java
boolean check = true;
try {
    if(authentication==null) {
        throw new ResponseStatusException(HttpStatus.UNAUTHORIZED);
    }

    orderlistVO.setUsername(((UserDetails)authentication.getPrincipal()).getUsername());

    try {
        orderlistVO = orderService.checkOrder(orderlistVO);
    } catch (Exception e) {
        check = false;
        e.printStackTrace();
        return false;
    }

    if(orderlistVO == null) {
        throw new ResponseStatusException(HttpStatus.UNAUTHORIZED);
    }

    IamportResponse<Payment> payment_response = api.cancelPaymentByImpUid(cancel_data);
    if(payment_response.getResponse()==null) {
        check=false;
    }
    //assertNull(payment_response.getResponse()); // 이미 취소된 거래는 response가 null이다
    System.out.println(payment_response.getMessage());
    } catch (IamportResponseException e) {
        System.out.println(e.getMessage());

        switch(e.getHttpStatusCode()) {
        case 401 :  //401은 Unauthorized일때 혹은 비교를 했을때 아이디가 다른 경우에 가능하게 할 수 있다.
            check=false;
            break;
        case 404:   //해당 merhcant_uid가 존재하지 않는다.
            check=false;
            break;
        case 500 :  //서버와 응답할 수 없는 경우
            check=false;
            //TODO
            break;
        }
    }catch (ResponseStatusException e) {
        check = false;
        e.printStackTrace();
    }catch (IOException e) {
        // TODO Auto-generated catch block
        check=false;
        e.printStackTrace();
    }
    if(check) {
        orderlistVO.setOrderState(4L);
        try {
            orderService.orderlistUpdate(orderlistVO);
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            return false;
        }
    }

    return check;
}
```

주문 취소 버튼을 누르면
Ajax를 통해 주문 취소를 진행,
성공하면 페이지 재실행하여
변경된 데이터를 가져오게 한다.

우선 날짜를 비교해서 결제일이
현재 날짜와 다르다면
스케쥴러로 해당 내역이 admin에
저장되어 있으므로
해당 내역들을 조회하여 변경한다.

그 다음으로는 로그인 상태를 확인한다.
로그인 상태가 아니라면
error를 발생시킨다.

마지막으로 미리 지정되어있는
Api기능으로 결제를 취소하고,
orderState를 변경시킨 후 종료한다.

Admin Scheduler

해당 scheduler의 역할은 두가지가 있다.
하나는 전날 팔린 제품들의 개수를 파악해서 admin Table에 넣어 내역 조회의 어려움을 줄이기 위함이고,
다른 하나는 사용하지 않았는데, 만료기간이 지나 더 이상 사용할 수 없는 쿠폰을 제거하여
쓸모 없는 데이터를 정리하기 위함이다.

```java
//하루 판매 현황을 모두 조회한 다음에 상품 별로 나눠서 각 상품이 팔린량을 저장한다(안팔린건 그냥 무시)
//이후 유효기간이 끝난 쿠폰을 삭제한다.
@Scheduled(cron = "0 10 0 * * *")
public void adminTable()throws Exception{
    Calendar calendar = Calendar.getInstance();
    calendar.add(Calendar.DATE, -1);
    Date dbdate = new Date(calendar.getTimeInMillis());

    List<PurchaseVO> purchaseVOs = purchaseService.getTodayPurchase(dbdate);
    HashMap<Long, AdminVO> hashMap = new HashMap<>();
    for(PurchaseVO purchaseVO : purchaseVOs) {
        if(hashMap.get(purchaseVO.getProductNum()) == null) {
            AdminVO adminVO = new AdminVO();
            adminVO.setAdminDate(dbdate);
            adminVO.setProductNum(purchaseVO.getProductNum());
            adminVO.setSellCount(purchaseVO.getProductCount());
            hashMap.put(purchaseVO.getProductNum(), adminVO);
        }else {
            AdminVO adminVO = hashMap.get(purchaseVO.getProductNum());
            adminVO.setSellCount(adminVO.getSellCount()+purchaseVO.getProductCount());
            hashMap.replace(purchaseVO.getProductNum(), adminVO);
        }
    }

    List<AdminVO> list = new ArrayList<>();
    Iterator<Long> iterator = hashMap.keySet().iterator();
    while(iterator.hasNext()) {
        list.add(hashMap.get(iterator.next()));
    }

        if(list.size() > 0) {
            adminService.insertAdmin(list);
        }

        couponService.scheduleDelete(dbdate);
    }
}
```

Cron을 이용하여 매일 00시 10분에 해당 scheduler가 실행되게 만들었다.

해당 날짜의 purchase목록을 가지고 와서 팔린 상품의 primary키로
Hash맵을 만들어 상품별 판매량을 산정하고 있다.
이후 list size가 0보다 크다면 해당 테이블을 집어넣는다.

마지막으로 이전날을 기준으로 검색하여 만료기간이 지난 쿠폰을 제거한다.

```xml
<delete id="scheduleDelete" parameterType="java.sql.Date">
    <![CDATA[delete from coupon where useCheck = 0 and exDate <= #{param}]]>
</delete>
```

해당 sql문에 <가 들어가기 때문에
<![CDATA[처리를 하여 에러를 방지했다.

Admin Page Template

전체적으로 상단bar와 좌측 navbar를 가지고 있으며, footer를 가지고 있다.
해당 태그들은 grid로 처리가 되어있기 때문에, 특정한 규칙을 지키면서
Template들을 가져와서 사용하여야 한다.

좌측 상단의 로고를 클릭하면
Admin홈으로 이동하게 된다.
톱니를 누르면 메뉴가 생기는데,
이곳에서 로그아웃과
메인페이지로 이동할 수 있다.
다시 톱니를 누르면 메뉴가 닫힌다.

```html
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html style="height:100%; overflow: auto;">
<head>
<c:import url='${pageContext.request.contextPath}/WEB-INF/views/templates/head.jsp'>
    <c:param name="title" value="-ADMIN"></c:param>
</c:import>
<meta charset="UTF-8">
</head>
<body style="width: 100%; height:100%; display: grid; grid-template-columns: 15% 85%; grid-template-rows:11% auto;">
<c:import url="${pageContext.request.contextPath}/WEB-INF/views/templates/adminHeader.jsp"></c:import>
<c:import url="${pageContext.request.contextPath}/WEB-INF/views/templates/adminNav.jsp"></c:import>
<c:import url="${pageContext.request.contextPath}/WEB-INF/views/templates/adminFooter.jsp"></c:import>
<script type="text/javascript" src="${pageContext.request.contextPath}/js/admin/setCouponSp.js"></script>
</body>
</html>
```

해당되는 HTML은 admin templat를 사용하는 예시로,
Html heigh를 100%로 해두어 그리드가
제대로 적용될 수 있게 만들었고,
Overflow속성으로 그리드를 넘어가는 태그가
나오더라도 제대로 대처할 수 있게 만들었다.
Footer는 해당 페이지 상단으로 올리는 버튼과
톱니바퀴 버튼을 작동시키는 javascript파일로
이루어져 있다.
좌측navbar만 sticky옵션으로 되어있고,
Y축을 모두 가질 수 있게 만들어 놓았다.
다만 Head.jsp의 경우 공통적으로 필요한
Head요소들이 들어있기 때문에 그대로 사용하였다.
화살표 사이에 div태그를 넣은 후 그 안에
해당 페이지의 내용을 집어넣으면 된다.

Admin Page

Admin HomePage

상품관리

상품등록

상품분류등록

상품정보수정

배송상태수정

판매현황

일간판매현황

주간판매현황

월간판매현황

쿠폰 &
마일리지

쿠폰전송

쿠폰등록

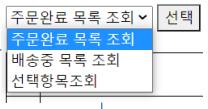
마일리지전송

Admin Home화면은 좌측 navbar의
내용과 별 다를 바가 없다.
상품과 주문을 관리하는 메뉴,
판매 현황을 차트로 보여주는 메뉴,
쿠폰과, 마일리지를 관리, 전송하는 메뉴로
크게 세가지로 나누어져 있고,
각각의 페이지로 이동할 수 있다.

상품 등록, 분류등록, 정보수정의 세가지는 product 쪽에서 담당해 주었고,
나머지 내용들은 모두 직접 구현하였다.

Admin Page – Order State Update

주문 상태는 총 4개의 단계로 구성되어있다.
1의 경우에는 주문완료,
2의 경우에는 배송중,
3의 경우에는 배송완료,
4의 경우에는 주문취소 로 이루어져 있다.
1과 2의 경우 상품을 준비해서 보내고, 완료신호가 온다면
빠르게 바꿔야 하기 때문에 선택 updat가 가능하게 만들었다.
하지만 updat를 잘못 했을 수 있고, 주문 취소를 문의사항에서 진행한다면
해당 과정을 Database 쿼리문으로 실행시키기 까다롭기 때문에
1과 2의 경우 각각 table로 만들어 두었고,
orderNum으로 조회하여 직접 단일 항목을 update시키는
세 개의 방안을 마련해 두었다.



Javascript를 통해 selector의 selected된 값이 달라진다면 ajax를 통해 데이터를 불러오거나,
Display:none 처리된 항목을 block으로 바꾸고, 원래 block인 상태를 none으로 바꾸어
단일 항목을 검색할 수 있는 div를 보이게 만들거나, 숨기게 만들었다.

배송 상태 수정

주문완료 목록 조회 선택

☐	주문번호	주문이름	주문현황
☐	1625389530752-20216418530	S_pants29 외 1벌	1
☐	1625133336531-202161185536	S_pants28 외 1벌	1

update

checkbox에 orderNum들이 들어있기 때문에 선택된 항목들을 배열에 담아 Controller로 보내 Update시킨다.

선택항목조회 선택

조회하기

수정하기

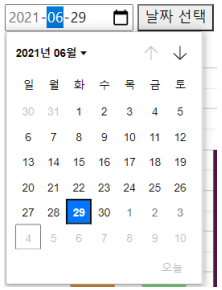
```
$("#stateChangeButton").click(function(event){
    event.preventDefault();

    let orderNums = new Array();
    $(".checklist").each(function(){
        if($(this).prop("checked")){
            orderNums.push($(this).attr("data-orderNum"));
        }
    });

    if(orderNums.length==0){
        swal({
            icon:"info",
            title:"체크요망",
            text:"상태정보를 업데이트할 주문번호를 선택해주세요."
        });
        return;
    }
   $.post({
       url:"./orderlistsUpdate",
       traditional:true,
       data:{
           orderNums:orderNums
       },
       success:function(result){
           if(result>0){
               alert("orderState가 변경되었습니다.");
               location.href="./orderlistUpdate";
           }else{
               swal({
                   icon:"error",
                   title:"ERROR",
                   text:"orderState 변경중 문제가 발생하였습니다."
               });
           }
       }
   });
});
```

```
@ResponseBody
@PostMapping("orderlistUpdate")
@Transactional(rollbackFor = Exception.class)
public Long orderlistUpdate(OrderlistVO orderlistVO)throws Exception{
    java.sql.Date date = new java.sql.Date(Calendar.getInstance().getTime().getTime());
    Long temp = orderlistVO.getOrderState();
    if(orderlistVO.getOrderState()==4) {
        orderlistVO = orderService.selectedOrder(orderlistVO);
        if(!date.toString().equals(orderlistVO.getOrderDate().toString())) {
            List<PurchaseVO> list = purchaseService.getPurAdmin(orderlistVO);
            if(adminService.cancledOrder(list) == 0) {
                throw new Exception();
            }
        }
    }else {
        orderlistVO = orderService.selectedOrder(orderlistVO);
        if(orderlistVO.getOrderState()==4) {
            List<PurchaseVO> list = purchaseService.getPurAdmin(orderlistVO);
            if(adminService.reCancledOrder(list) == 0) {
                throw new Exception();
            }
        }
    }
    orderlistVO.setOrderState(temp);
    return orderService.orderlistUpdate(orderlistVO);
}
```

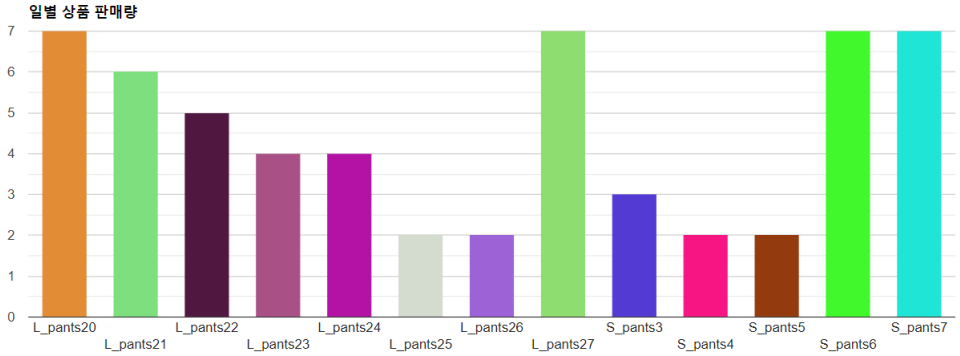
선택항목을 원하는 state로 바꾸는 과정에서 주문취소가 된다면 admin에서 이전에 구매한 항목을 제거해야 된다.
주문 취소를 복구한다면 다시 추가해주어야 하기에 state Update이외에도 날짜를 얻는 과정과 Admin Update구문이 들어가게 된다.

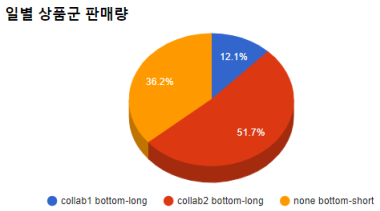
Admin Page – Sale Day

기본적으로 날짜 선택 input box와 두 가지의 chart로 이루어져 있다.



6월 1일부터 전날 까지 선택이 가능하고,
기본적으로 전날의 데이터를 가지고 와서 차트로 보여주게 된다.
Input type="date" 로 선택한 결과가 "yyyy-MM-dd" format으로 controller로 보내지기 때문에 그대로 sql문으로 검색하여 해당 날짜의 admin 데이터들을 가져오게 된다.





Sale Day에서의 테이블은 두개로, 그날 팔린 상품의 총 계수와 그날 팔린 상품들의 상품군을 기준으로한 총계수를 각각 column차트와 Pie 차트를 이용하여 표기하였다.

```javascript
var chart1;
var chart2;
$(document).ready(function(){
    google.charts.load("current", {packages:["corechart", "bar"]});
    let selectedDate = "";

    getData(selectedDate);
});
$("#ajaxDate").click(function(event){
    event.preventDefault();
    let selectedDate = $("#getDate").val();

    chart1.clearChart();
    chart2.clearChart();
    getData(selectedDate);
});
function getData(selectedDate){
    $.get({
        url:"./getSaleDay",
        data:{
            date:selectedDate
        },
        success:function(datas){
            let array1 = new Array();
            const hm = new Map();
            for(let data of datas){
                let obj = JSON.parse(data);
                let productTitle = obj.productVO.productTitle;
                let sellCount = obj.sellCount;
                let temp1 = new Array();
                temp1.push(productTitle);
                temp1.push(sellCount);
                array1.push(temp1);

                let temp2 = obj.productVO.productDivisionVO.collab+" "+obj.productVO.productDivisionVO.productType;

                if(hm.has(temp2)){
                    let tempdata = hm.get(temp2);
                    hm.set(temp2, tempdata + obj.sellCount);
                }else{
                    hm.set(temp2, obj.sellCount);
                }
            }
            let arr = Array.from(hm);
            let array2 = new Array();
            for(let data of array1){
                data.push(randomRGB());
                array2.push(data);
            }

            google.charts.setOnLoadCallback(
                function(){
                    drawChart1(arr);
                    drawChart2(array2);
                });
        }
    });
}
```

Input box의 value를 ajax로 보내고 결과를 json으로 가져와 이름, 상품군을 Key로 둔 후 map으로 판매량을 집계했다.

```javascript
function randomRGB(){
    const r = Math.floor(Math.random()*256).toString(16);
    const g = Math.floor(Math.random()*256).toString(16);
    const b = Math.floor(Math.random()*256).toString(16);
    let rgb = "color:#";
    if(r.length==1){
        rgb+="0"+r;
    }else{
        rgb+=r;
    }
    if(g.length==1){
        rgb+="0"+g;
    }else{
        rgb+=g;
    }
    if(b.length==1){
        rgb+="0"+b;
    }else{
        rgb+=b;
    }
    rgb+=";";
    return rgb;
}
```

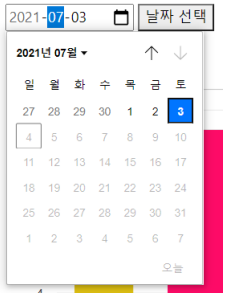
Column 그래프의 막대들은 기본적으로 색이 지정되어있지 않아 보기 편하게 랜덤으로 색을 만들어 주었다.

```javascript
function drawChart1(data1){
    let data = new google.visualization.DataTable();
    data.addColumn('string', 'product name');
    data.addColumn('number', 'sale count');
    data.addRows(data1);
    let options = {
        title: '일별 상품군 판매량',
        titleTextStyle: {
            fontSize:16
        },
        is3D: true,
        chartArea:{width:'100%', height:'80%', left:0},
        legend: {'position': 'bottom'}
    }

    chart2 = new google.visualization.PieChart(document.getElementById('chart2'));
    chart2.draw(data,options);
}
```

```javascript
function drawChart2(data2){
    let data = new google.visualization.DataTable();
    data.addColumn('string', 'product div name');
    data.addColumn('number', 'sale count');
    data.addColumn({type:"string",role: "style"});
    data.addRows(data2);
    var view = new google.visualization.DataView(data);
    let options = {
        title: '일별 상품 판매량',
        titleTextStyle: {
            fontSize:16
        },
        legend: {'position': 'none'},
        axes: {
            x:{
                0:{side:'top', label:'Percentage'}
            }
        },
        chartArea:{width:'90%', height:'80%'}
    }
    chart1 = new google.visualization.ColumnChart(document.getElementById('chart1'));
    chart1.draw(view,options);
}
```

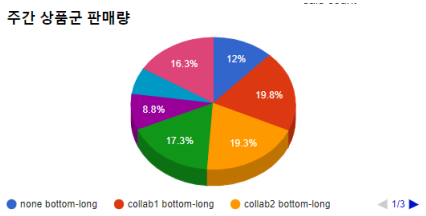
집계를 위해 사용한 Map을 Array로 바꾼 후 구글 차트를 이용하여 Column, pie 차트를 만들어 주었다.

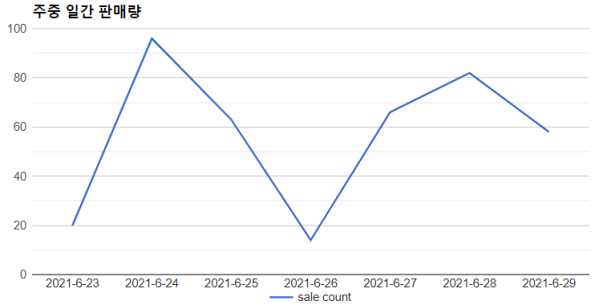
Admin Page - Sale Week

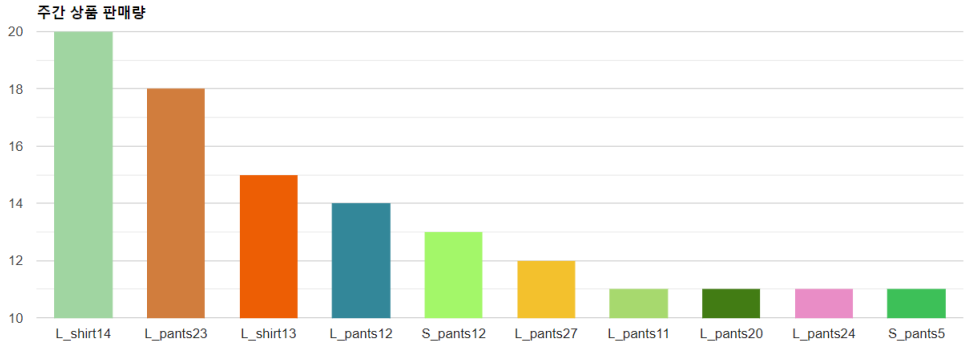


Sale Day와 전체적인 형식은 같지만
일별로 판매량 추이를 나타내는 barchart가 추가되었으며,
Column 차트의 데이터가 그 주 동안 판매량이 제일 높았던 항목을
최대 10개만을 보여주게 되었다.

원래대로라면 input Type을 Week로 했겠지만
input type을 Date로 설정한 이유는
Sale의 Week data가 Java의 week선정과
많이 달랐기 때문에 정확한 일자를 얻어낼 수
없었고, 주 선택이 Date를 고르는 것이
더 유동성이 좋았기 때문이다.
선택한 날짜가 그 주의 마지막 날로 계산이 된다.







```java
//일주일치의 판매량을 집계하여 보여주는 페이지의 일자를 변경하였을 때 해당 결과를 보내주는 페이지
@ResponseBody
@GetMapping("getSaleWeek")
public String[] getSaleWeek(String date)throws Exception{
    String date2 = "";
    if(date.equals("") {
        Calendar calendar = Calendar.getInstance();
        calendar.add(Calendar.DATE, -1);
        java.sql.Date tempdate = new java.sql.Date(calendar.getTimeInMillis());
        date = tempdate.toString();
        calendar.add(Calendar.DATE, -6);
        tempdate = new java.sql.Date(calendar.getTimeInMillis());
        date2 = tempdate.toString();
    }else {
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-MM-dd");
        Calendar calendar = Calendar.getInstance();
        calendar.setTimeInMillis(simpleDateFormat.parse(date).getTime());
        calendar.add(Calendar.DATE, -6);
        java.sql.Date tempdate = new java.sql.Date(calendar.getTimeInMillis());
        date2 = tempdate.toString();
    }
    HashMap<String, String> hm = new HashMap<>();
    hm.put("start", date2);
    hm.put("end", date);
    List<AdminVO> list = adminService.saleDays(hm);
    Gson gson = new Gson();
    int length = list.size();
    String[] result = new String[length];
    for(int i = 0 ; i < length; i++) {
        list.get(i).setParseDate(list.get(i).getAdminDate().toString());
        result[i] = gson.toJson(list.get(i));
    }
    return result;
}
```

Controller로 Date를 전송하면 그것을 받아 첫 날짜를 얻은 후 해당 시간 데이터들을 이용하여 mapper에서 데이터를 조회한 후 json으로 만들어 javascript로 전송한다.
이후 javascript에서 데이터를 종합한 후 chart로 만들어 이용자에게 보여주게 된다.

페이지를 열때 기본데이터를 “”로 두었기 때문에 해당 날짜의 전날을 마지막날로 두고 계산한다.
Calendar를 이용한 계산으로 첫날과 마지막날을 구한다.

```xml
<resultMap type="AdminVO" id="adminAndProduct">
    <id column="adminIdx" property="adminIdx"/>
    <result column="adminDate" property="adminDate"/>
    <result column="productNum" property="productNum"/>
    <result column="sellCount" property="sellCount"/>
    <association property="productVO" javaType="ProductVO">
        <id column="productNum" property="productNum"/>
        <result column="productTitle" property="productTitle"/>
        <result column="productPrice" property="productPrice"/>
        <result column="finalPrice" property="finalPrice"/>
        <result column="productDivNum" property="productDivNum"/>
        <association property="productDivisionVO" javaType="ProductDivisionVO">
            <id column="productDivNum" property="productDivNum"/>
            <result column="collab" property="collab"/>
            <result column="productType" property="productType"/>
        </association>
    </association>
</resultMap>
```

```xml
<select id="saleDays" parameterType="java.util.HashMap" resultMap="adminAndProduct">
    select a.*, p.*, pd.* from admin a inner join product p on a.productNum=p.productNum
    left join productdivision pd on p.productDivNum = pd.productDivNum
    where a.adminDate between #{start} and #{end}
</select>
```

Sale Day와 차이점이라면 차트가 하나 더 늘었고,
Column 차트의 데이터 수가 최대 10개라는 점이 있다.

```
function getData(){

    const date = $("#getDate").val();
    let first = new Date(date);
    first.setDate(first.getDate()-6);
    let count = 0;
    const arr2 = new Map();
    while(count<7){
        let tempdate = new Date(first);
        tempdate.setDate(first.getDate()+1);
        first.setDate(first.getDate()+1);
        tempdate = tempdate.getFullYear()+"-"+(tempdate.getMonth()+1)+"-"+tempdate.getDate();
        arr2.set(tempdate, 0);
        count++;
    }
    const productMap = new Map();

    const hm = new Map();
    $.get({
        url:"./getSaleWeek",
        data:{
            date:date
        },
        success:function(result){
            for(let datas of result){
                let data = JSON.parse(datas);

                //상품별로 판매한 수량
                if(productMap.has(data.productVO.productTitle)) {
                    productMap.set(data.productVO.productTitle, productMap.get(data.productVO.productTitle) + data.sellCount);
                }else {
                    productMap.set(data.productVO.productTitle, data.sellCount);
                }

                //일별로 판매한 수량
                let tempdate = new Date(data.parseDate);
                tempdate = tempdate.getFullYear()+"-"+(tempdate.getMonth()+1)+"-"+tempdate.getDate();
                arr2.set(tempdate, arr2.get(tempdate)+data.sellCount);

                //상품 종류별로 판매한 수량
                let temp2 = data.productVO.productDivisionVO.collab + " " + data.productVO.productDivisionVO.productType;
                if(hm.has(temp2)){
                    let tempdata = hm.get(temp2);
                    hm.set(temp2, tempdata + data.sellCount);
                }else{
                    hm.set(temp2, data.sellCount);
                }
            }
```

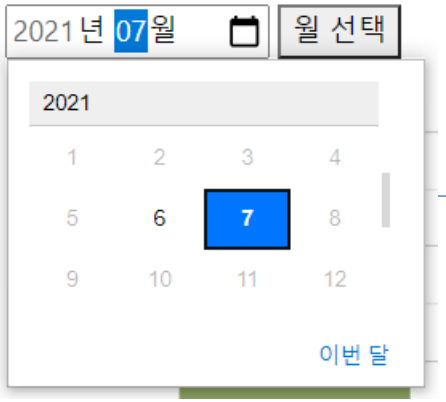
차트가 하나 더 늘었으므로, 일자별 데이터가 무엇이 있는지 알려주기 위해서(하나도 안 팔린 날이더라도 0으로 표기되어야 하기 때문) 해당 주의 모든 일을 map에 넣어 초기화 시킨다.

```
let array2 = Array.from(arr2);
let array1 = new Array();

let arr1 = Array.from(productMap);

arr1.forEach(function(value){
    value.push(randomRGB());
});

arr1.sort(function(a, b){
    return b[1]-a[1];
});
if(arr1.length<=10){
    array1 = arr1;
}else{
    for(let i = 0 ; i < 10; i++){
        array1.push(arr1[i]);
    }
}
google.charts.setOnLoadCallBack(
    function(){
        drawChart1(array1);
        drawChart2(array2);
        drawChart3(Array.from(hm));
    }
);
    }
});
}
```

상품별로 총 판매 계수를 구한 후 Sorting하여 최대 10개만 뽑아내거나 그대로 array로 만들어 사용하는 구문

각 날마다 얼마나 팔렸는지, 혹은 상품군에 따라서 얼마나 상품이 팔렸는지 해당 조건을 key로 지정한 map으로 통계를 내어 Chart의 data로 사용한다.

Admin Page – Sale Month



Sale Month또한 Sale Day와 Sale Week과 별다른 차이는 없다.
차이점이라면 날짜 선택이 Input type="month"라는 것이 있다.
Day표시가 안된 "2021-07"과 같은 형식으로 controller로 이동하기에
해당 월의 데이터를 모두 구하기 위해서 adminDate가 2021-07 로
시작하는 모든 데이터를 가져오게 만들었다.

```
<select id="saleMonth" parameterType="java.lang.String" resultMap="adminAndProduct">
    select a.*, p.*, pd.* from admin a inner join product p on a.productNum=p.productNum
    left join productdivision pd on p.productDivNum = pd.productDivNum
    where a.adminDate like concat(#{value},'%')
</select>
```

Admin Page - Sale - Testcase

```java
@Test
void test() throws Exception {
    List<AdminVO> list = new ArrayList<>();
    SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-MM-dd");
    for(int i = 1 ; i <= 30; i++) {
        String temp = String.valueOf(i);
        if(i < 10) {
            temp = 0 + temp;
        }
        Date middle = new Date(simpleDateFormat.parse("2021-06-"+temp).getTime());
        Random random = new Random(Calendar.getInstance().getTimeInMillis());
        int pCount = random.nextInt(30) + 1;
        List<AdminVO> templist = new ArrayList<>();
        for(int j = 0 ; j < pCount; j++) {
            AdminVO adminVO = new AdminVO();
            adminVO.setAdminDate(middle);
            random = new Random(Calendar.getInstance().getTimeInMillis());
            long pNum = (long)(random.nextInt(121) + 723);
            boolean check = false;
            for(int k = 0 ; k < templist.size(); k++) {
                if(pNum == templist.get(k).getProductNum()) {
                    check = true;
                    break;
                }
            }
            if(check) {
                j--;
                continue;
            }
            adminVO.setProductNum(pNum);
            random = new Random(Calendar.getInstance().getTimeInMillis());
            long count = (long)(random.nextInt(7) + 1);
            adminVO.setSellCount(count);
            templist.add(adminVO);
        }
        for(int j = 0 ; j < templist.size(); j++) {
            list.add(templist.get(j));
        }
    }
    assertNotEquals(0, adminService.insertAdmin(list));
}
```

판매량 chart를 테스트 하기 위해서
Test case를 많이 만들 필요가 존재했다.
한달 동안의 데이터를 만들기 위해서 simpleDateFormat으로
원하는 일자의 Date타입을 만들었고,
Random 함수를 이용하여 임의로 데이터를 만들었다.
(시드값이 겹치지 않게 계속 초기화 해주었다.)

들어가 있던 상품의 개수는 121개였으며, 해당번호가 723번부터
시작했으므로 Random 범위를 저렇게 잡아주었다.
또한 상품이 팔린 개수, 팔린 상품의 종류가 다양할 수 있기에
해당 데이터도 랜덤으로 잡아주었다.
또한 admin 테이블의 데이터는 그날 판매된 각 상품의 개수가
들어있기 때문에 adminDate와 productNumber 둘 모두 동일한 개체가
존재하면 안되기 때문에 해당 데이터들을 비교하여
같으면 되돌리는 작업을 진행하였다.

해당 반복문의 마지막에 adminDate와 productNum 둘 모두 겹치는 상황을 막기 위해 만들었던
List의 데이터들을 실제 저장에 사용할 list에 넣어두고 마지막에 list를 db에 넣는 작업을
진행하여 테스트 케이스를 집어 넣었다.

Admin Page – Coupon Transmit

쿠폰 전송

쿠폰 번호	쿠폰 이름	할인률
11	fw	50
10	jason richardson	30
9	회원가입감사쿠폰	90
8	couponinsert	80
6	test5	50
5	test4	40
4	test3	30
3	test2	20
2	test1	10
1	default	0

쿠폰을 전송하기 위해 우선적으로
쿠폰 종류의 테이블을 보여준다.
쿠폰 종류에는 쿠폰 종류, 이름, 할인률이 있으며,
해당 데이터가 삭제되면 다른 테이블들에도 큰
영향을 미치기 때문에 삭제는 만들지 않았다.
보내고자 하는 쿠폰을 하나 선택하면
Transmit form으로 이동하게 된다.

쿠폰 번호	쿠폰 이름	할인률
11	fw	50

유효기간(현재 일부터 개월수) : 1

지정전송 모두에게 전송

쿠폰의 사용 기한을 월 단위로 관리한다.
쿠폰을 보내게 되면 쿠폰을 받은 날짜와 만료일을 넣게 되는데, 해당 input box의 숫자를 이용해서 만료일을 계산해 넣어주게 된다.

전송은 지정전송과 모두에게 전송이 존재한다.

```
<insert id="couponForAll" parameterType="CouponVO">
    insert into coupon(username,cuSpNum, pubDate, exDate, useCheck)
    select m.username, #{cuSpNum}, #{pubDate}, #{exDate}, 0 from member m
    group by username
</insert>
```

모두에게 전송은 서브쿼리를 이용한다.
일반 데이터는 직접 넣어주고,
Username의 경우 member에서
조회한 후 각각을 넣어주게 된다.

지정전송

삭제하기
삭제하기
삭제하기

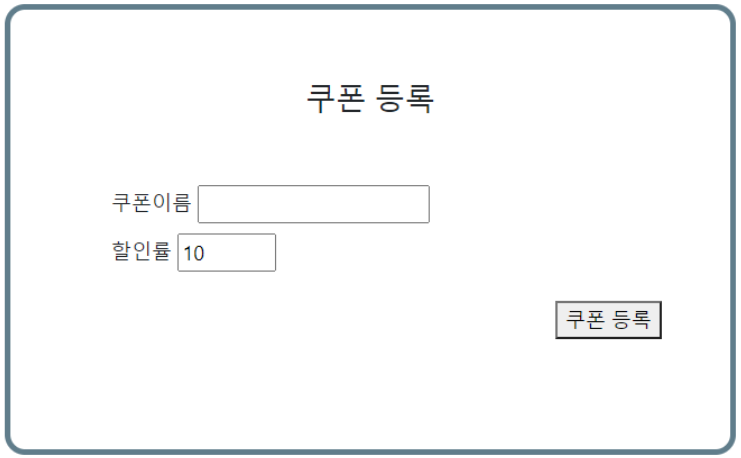
추가하기 전송하기

지정전송을 누르면 나타나게 만든 form이다.
추가하기를 누르면 username을 입력하는
Input box와 해당 username을 제거하는 삭제버튼
이 존재한다.

모두에게 전송과 비슷하나 username이
입력된 사람에게만 쿠폰을 전송한다는
차이가 존재한다.
해당 username들은 다른 데이터들과 합쳐
couponVO 객체들에 넣은 후
List형태로 mapper로 보내 insert 시키는
방식을 이용하였다.

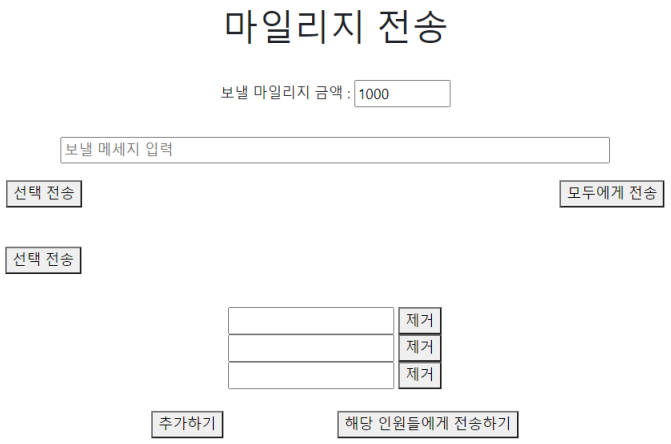
```
<insert id="mileageForSelect" parameterType="java.util.List">
    <foreach collection="list" item="item" separator=";">
        insert into mileage (usedMile, changeMile, username, orderNum, mileContents, enabledmile)
        (select usedMile, #{item.changeMile}, username, null, #{item.mileContents}, enabledMile+#{item.enabledMile}
        from mileage where mileNum = (select max(mileNum) from mileage where username=#{item.username}))
    </foreach>
</insert>
```

Admin Page – CouponSp Insert



쿠폰 종류를 추가하기 위해 만든 form이다.
쿠폰을 전송하기 쉽게 해당 종류를 넣어주는 것이고,
Couponsp table에서 데이터가 하나 삭제된다면
연관된 데이터들도 모두 영향을 받기 때문에
제거기능과 update기능은 만들어두지 않았다.
굳이 새로운 coupon을 위해서 update하지 않고
쿠폰종류 추가기능을 사용하면 된다.

Admin Page – Mileage Transmit




```
<insert id="mileageForAll" parameterType="MileageVO">
    INSERT INTO mileage (usedMile, changeMile, username, orderNum, mileContents, enabledmile)
    SELECT usedMile, changeMile+#{changeMile}, username, NULL, #{mileContents}, enabledMile+#{enabledMile}
    FROM mileage WHERE mileNum IN (SELECT MAX(mileNum) FROM mileage GROUP BY username);
</insert>
```

전체 구조는 쿠폰 전송과 매우 유사하다.
Mileage의 경우 어떠한 이유로 주어졌는지 조회할 수 있으므로 해당 메시지를 전송할 수 있게 하였고,
보내는 마일리지를 input type="number"를 통해 지정해 줄 수 있다.
똑같이 모두에게 전송과 지정전송이 존재하며,
Mileage 테이블 특성상 기존 데이터를 이용하여 Insert 해야되기 때문에 해당 유저의 맨 마지막 mileage 데이터를 조회한 후에 그 데이터를 이용하여 insert하는 식으로 진행하였다.

서브쿼리를 이용해 username을 그룹화 한 다음 max(MileNum)을 이용하여
유저들마다의 milenum 최대치(결과적으로 현재 가지고 있는 마일리지 데이터)를 얻고,
보내준 데이터들을 이용하여 다량으로 Mileage를 insert시켜주었다.

product

이찬우

List

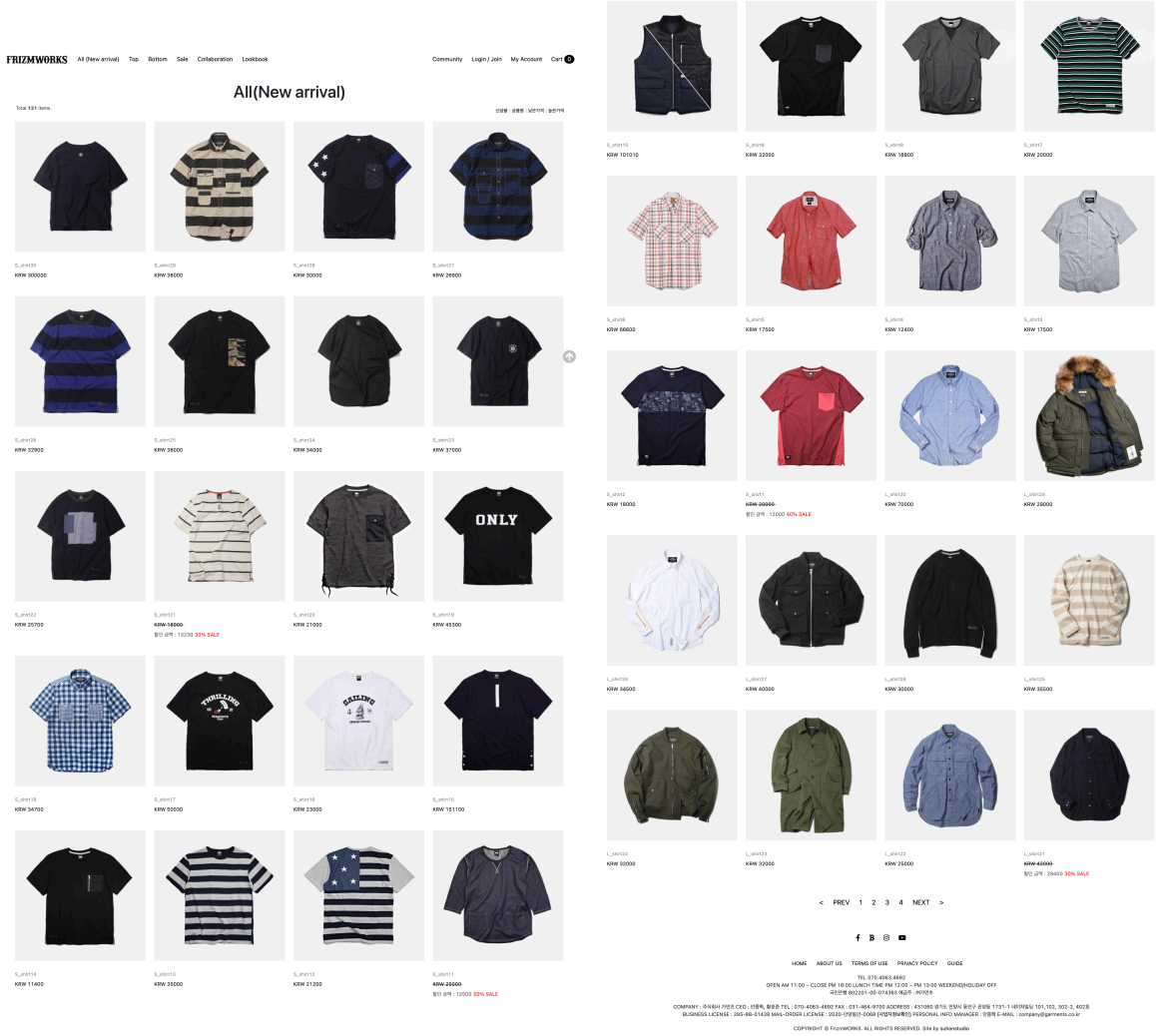
```xml
<select id="getList" parameterType="ProductPager"
    resultMap="productList">
    select * from
    (select row_number() over(
    order by
    <choose>
        <when test="sortStandard=='name'">
            productTitle asc
        </when>
        <when test="sortStandard=='lowPrice'">
            productPrice asc
        </when>
        <when test="sortStandard=='highPrice'">
            productPrice desc
        </when>
        <otherwise>
            productNum desc
        </otherwise>
    </choose>

    ) as R, P.*,PF.fileName,PF.oriName from
    (select * from product
    where
    productDivNum in (select productDivNum from productDivision where
    productType like concat('%',#{productType},'%'))
    and
    productDivNum in
    (select productDivNum from productDivision where collab like
    concat('%',#{collab},'%'))
    and
    productSaleable = 1
    <if test="sale">
        and productDisRate > 0
    </if>
    ) as P
    left join
    (select * from productFiles where fileName like "M\_%")
    as PF
    on P.productNum = PF.productNum
    ) as RP
    where R between #{startRow}
    and #{lastRow}
</select>

<resultMap type="ProductVO" id="productList">
    <id column="productNum" property="productNum" />
    <result column="productTitle" property="productTitle" />
    <result column="productPrice" property="productPrice" />
    <result column="productDisRate" property="productDisRate" />
    <result column="summary" property="summary" />
    <result column="productContents" property="productContents" />
    <result column="productDivNum" property="productDivNum" />
    <result column="finalPrice" property="finalPrice" />
    <result column="productMileage" property="productMileage" />
    <result column="productSaleable" property="productSaleable" />
    <association property="file" javaType="ProductFileVO">
        <id column="fileNum" property="fileNum" />
        <result column="fileName" property="fileName" />
        <result column="oriName" property="oriName" />
    </association>
</resultMap>
```

SQL

한 페이지에 제품 40개씩 노출되도록 설정

정렬 기준을 신제품, 제품명, 높은 가격, 낮은가격 순으로 구성

제품 하나당 메인이미지 하나씩을 받아와 화면에 표시



화면 구성

List

FRIZMWORKS All (New arrival) Top Bottom Sale Collaboration Lookbook Community Login / Join My Account Cart 0

Total **121** items.

All(New arrival)

신상품 | 상품명 | 낮은가격 | 높은가격

모든 제품

TOP

long short

BOTTOM

long short

collab12

TOP(long)

long short

TOP(short)

long short

BOTTOM(long)

long short

BOTTOM(short)

long short

상의

하의

콜라보레이션

```
<c:if test="${division ne ''}">
    <h1 style="text-align: center;">${name}(${division})</h1>
</c:if>
```

소분류 선택 시

```
<c:if test="${division eq ''}">
    <h1 style="text-align: center;">${name}</h1>
</c:if>
```

소분류 미선택 시

대분류가 상의, 하의 일 경우

```
<c:if test="${name eq 'TOP' or name eq 'BOTTOM'}">
    <div style="text-align: center;">
        <span><a class="selBtn"
            href="${pageContext.request.contextPath }/product/list?collab=&productType=${name}-long&name=${name}&division=long&sortStandard=&sale=${sale}">long</a><a
            class="selBtn"
            href="${pageContext.request.contextPath }/product/list?collab=&productType=${name}-short&name=${name}&division=short&sortStandard=&sale=${sale}">short</a></span>
    </div>
</c:if>
```

List

FRIZMWORKS All (New arrival) Top Bottom Sale Collaboration Lookbook Community Login / Join My Account Cart 0

All(New arrival)

Total **121** items.

신상품 | 상품명 | 낮은가격 | 높은가격

SQL

```
<select id="getTotalCount" parameterType="ProductPager"
    resultType="java.lang.Long">
    select count(productNum) from product
    where
    productDivNum in (select productDivNum from productDivision where
    productType like concat('%',#{productType},'%'))
    and
    productDivNum in (select productDivNum from productDivision where collab like concat('%',#{collab},'%'))
    and
    productSaleable = 1
    <if test="sale">
        and productDisRate > 0
    </if>
</select>
```

HTML

```
<p class="prdCount">
    Total <strong>${totalCount}</strong> items.
</p>
```

HTML

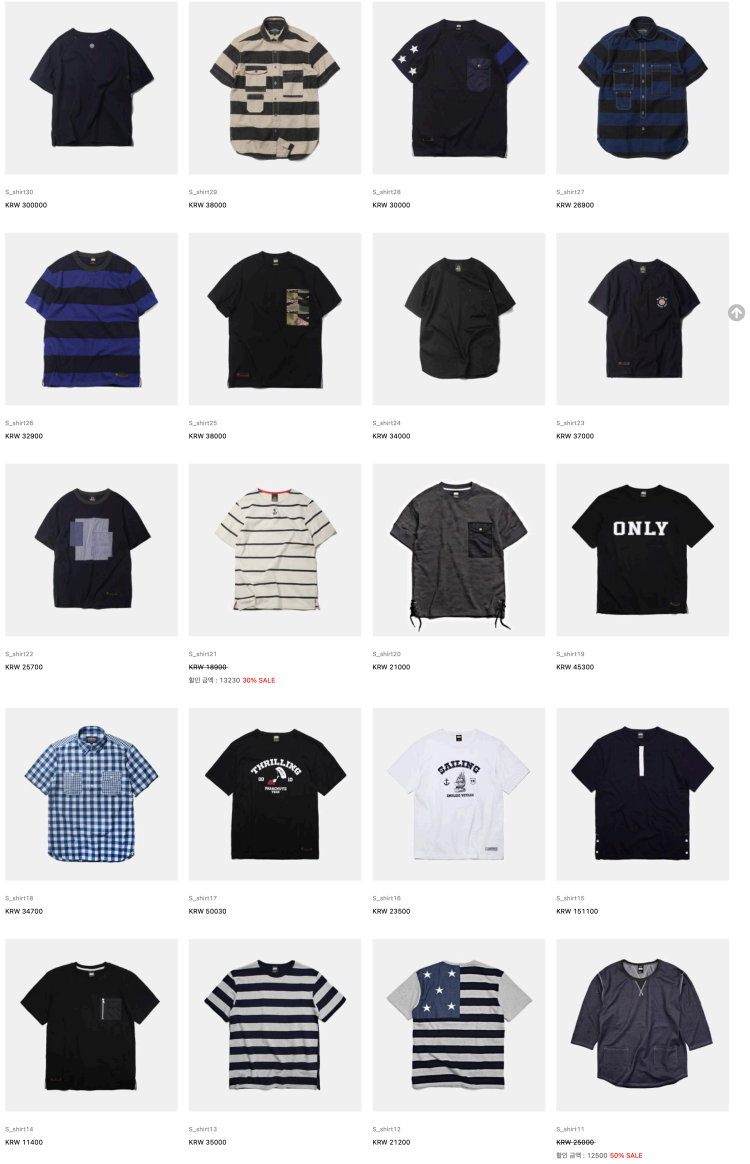
```
<div class="sort">
    <span><a class="sortName"
        href="${pageContext.request.contextPath }/product/list?collab=${collab}&productType=${name}-${division}&name=${name}&division=${division}&sortStandard=new&sale=${sale}">신상품</a></span>
    <span class="sortName"> | </span> <span><a class="sortName"
        href="${pageContext.request.contextPath }/product/list?collab=${collab}&productType=${name}-${division}&name=${name}&division=${division}&sortStandard=name&sale=${sale}">상품명</a></span>
    <span class="sortName"> | </span> <span><a class="sortName"
        href="${pageContext.request.contextPath }/product/list?collab=${collab}&productType=${name}-${division}&name=${name}&division=${division}&sortStandard=lowPrice&sale=${sale}">낮은가격</a></span>
    <span class="sortName"> | </span> <span><a class="sortName"
        href="${pageContext.request.contextPath }/product/list?collab=${collab}&productType=${name}-${division}&name=${name}&division=${division}&sortStandard=highPrice&sale=${sale}">높은가격</a></span>
</div>
```

List

```html
<div class="divMain">

    <c:forEach items="${productList}" var="product">
        <div class="cardList">
            <div id="thumb">
                <a
                    href="${pageContext.request.contextPath }/product/select?productNum=${product.productNum}">
                    <c:if test="${!empty product.file.fileName }">
                        <img id="cardImg" alt="${product.file.oriName}"
                            src="/resources/images/product/${product.productNum}/${product.file.fileName}">
                    </c:if> <c:if test="${empty product.file.fileName}">
                        <img id="cardImg" alt="not"
                            src="${pageContext.request.contextPath}/images/product/test/frizm_278.jpg">
                    </c:if>
                </a>
            </div>
            <div id="cardContents">
                <span><a
                    href="${pageContext.request.contextPath }/product/select?productNum=${product.productNum}"
                    style="font-size: 11px; color: #838383;">${product.productTitle}</a></span>
                <br>
                <c:if test="${product.productDisRate eq 0}">
                    <span style="font-size: 12px; color: #000000;">KRW
                        ${product.productPrice}</span>


                </c:if>
                <c:if test="${product.productDisRate ne 0}">
                    <span
                        style="font-size: 12px; color: #000000; text-decoration: line-through;">KRW
                        ${product.productPrice} </span>
                    <br>
                    <span style="font-size: 12px; color: #555555;">할인 금액 :
                        ${product.finalPrice}</span>
                    <span style="font-size: 12px; color: #ff0000;">
                        ${product.productDisRate}% SALE</span>
                </c:if>
            </div>
        </div>
    </c:forEach>

</div>
```

```css
.cardList {
    vertical-align: top;
    display: inline-block;
    width: 25%;
    padding: 0px;
    margin: 0 -2px 40px;
    border: 0px;
}

#thumb {
    margin: 0 10px 18px;
    text-align: center;
}

#cardImg {
    width: 100%;
    height: auto;
    box-sizing: border-box;
}
```

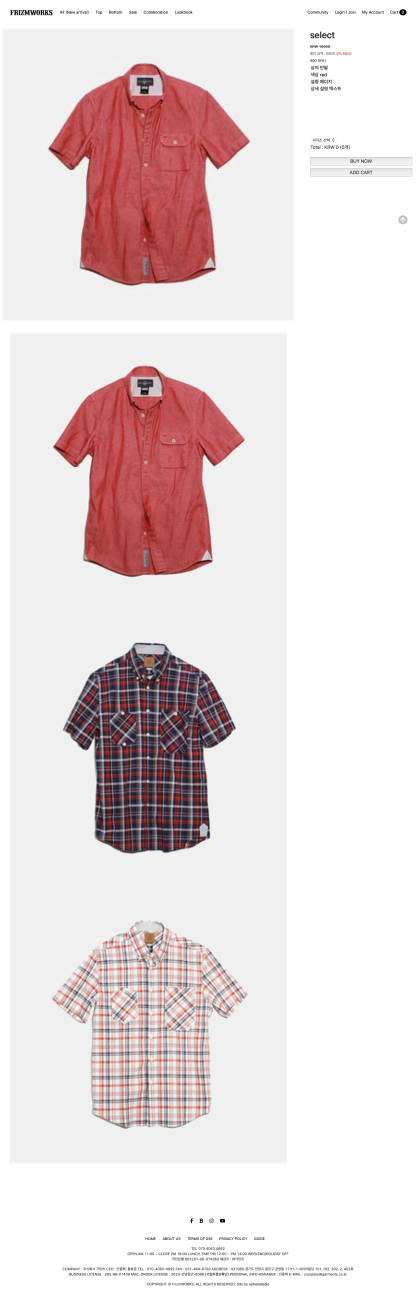
한 줄 당 4개의 제품을 배치
제품의 할인 여부를 판단하여 금액 표시 변경



Select

```
<select id="getProductSelect" parameterType="ProductVO"
    resultType="ProductVO">
    select * from product where productNum = #{productNum}
</select>
```

SQL



화면구성

Select

select

~~KRW 10000~~

할인 금액 : 9800 2% SALE

500 (5%)

상의 반팔
색상 red
설명 페이지
상세 설명 텍스트

사이즈 선택

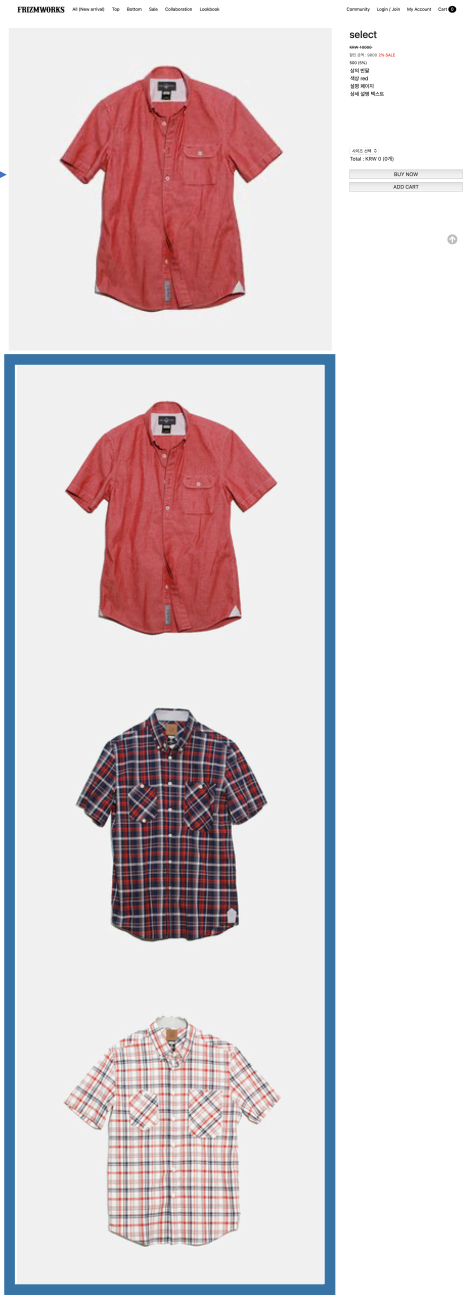
Total : KRW 0 (0개)

BUY NOW

ADD CART

```
private Long productNum;
private String productTitle;
private Long productPrice;
private Long productDisRate;
private String summary;
private String productContents;
private Long productDivNum;
private Long finalPrice;
private Long productMileage;

private String collab;
private String productType;
private Boolean productSaleable;
private ProductDivisionVO productDivisionVO;
private ProductFileVO file;
private List<ProductInfoVO> infos;
```

ProductVO



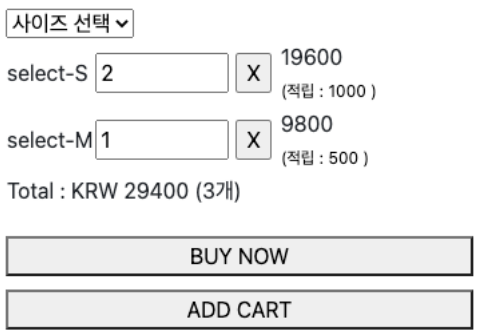

```
<select id="getProductSelect" parameterType="ProductVO"
    resultType="ProductVO">
    select * from product where productNum = #{productNum}
</select>
```

SQL

Select

```
<select id="sizes" onchange="changed(this.value)">
    <option value="default">사이즈 선택</option>
    <c:forEach items="${VO.infos}" var="info">
        <c:if test="${info.stock eq 0}">
            <option disabled>${info.size}[품절]</option>
        </c:if>
        <c:if test="${info.stock ne 0}">
            <option value="${info.size}">${info.size}</option>
        </c:if>
    </c:forEach>
</select>
<div id="showTest">
    <table>
        <thead></thead>
        <tbody>
            <c:forEach items="${VO.infos}" var="info">
                <tr style="display: none;" id="size_${info.size}" title="1">
                    <td>${VO.productTitle}-${info.size}</td>
                    <td><input style="width: 100px;" type="number" min="1"
                        value="1" data-value="${info.size}" class="noProduct"><input
                        type="button" class="delInfo" data-value="${info.size}"
                        data-infoNum="${info.PInfoNum}" value="X"
                        style="margin: 0 5px;"></td>
                    <td><span class="finalPrices" id="${info.size}FinalPrice"
                        title="${VO.finalPrice}">${VO.finalPrice}</span><br> <span
                        style="font-size: 12px; color: #000000;" id="${info.size}Mile"
                        title="${VO.productMileage}">(적립 : ${VO.productMileage})</span></td>
                </tr>
            </c:forEach>
            <tr>
                <td colspan="3">
                    <div id="tPrice" title="0"></div>
                    <p id="totalText">Total : KRW 0 (0개)</p>
                </td>
            </tr>
        </tbody>
    </table>
</div>
```

사이즈가 품절(재고 수량 : 0)인 상태일 경우
select에서 선택이 안되게 처리

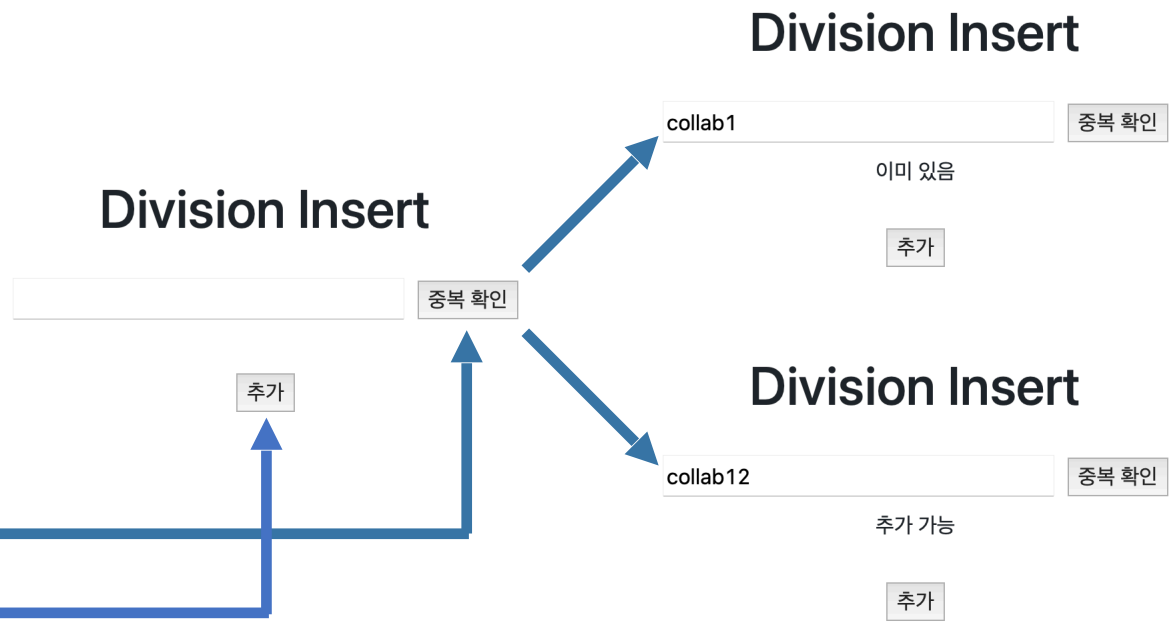



```
<script type="text/javascript">
    let tPrice = $("#tPrice").attr("title");

    countTotal = function() {
        let finalPrice = $("#finalPrice").attr("title");
        total = tPrice * finalPrice
        $("#totalText").text("Total : KRW " + total + " (" + tPrice + "개)")
    }

    function changed(str) {
        if ($("#size_" + str).attr("style").length != 0) {
            $("#size_" + str).children("td").children("input.noProduct")
                    .val("1")
            $("#size_" + str).attr("style", "");

            let count = $("#size_" + str).children("td").children(
                    "input.noProduct").val()
            let fPrice = $("#" + str + "FinalPrice").attr("title");
            let mile = $("#" + str + "Mile").attr("title");
            fPrice = count * fPrice;
            mile = count * mile;
            $("#" + str + "FinalPrice").text(fPrice);
            $("#" + str + "Mile").text("(적립 : " + mile + " )")
            tPrice++
            countTotal();
        }

        $("#sizes").val("default");

    }

    $(".delInfo").click(function() {
        let str = $(this).attr("data-value")
        $("#size_" + str).attr("style", "display:none;");
        tPrice = tPrice - Number($("#size_" + str).attr("title"))
        countTotal();
    })

    $(".noProduct").blur(function() {
        let t = 0;
        let count = $(this).val();
        let size = $(this).attr("data-value");
        let fPrice = $("#" + size + "FinalPrice").attr("title");
        let mile = $("#" + size + "Mile").attr("title");
        fPrice = count * fPrice;
        mile = count * mile;
        $("#" + size + "FinalPrice").text(fPrice);
        $("#" + size + "Mile").text("(적립 : " + mile + " )")

        $(".noProduct").each(function() {
            if ($(this).parent().parent().attr("style").length == 0) {
                t += Number($(this).val());
            }
        })
        $("#size_" + size).attr("title", count);
        tPrice = t;
        countTotal()
    })
</script>
```

총 수량, 금액 계산

Collabration Insert

```java
let dCheck = false;
let str="";

function duplCheck(checker){
    let collab=$("#collab").val();
    dCheck=false;
    if(collab.length>0){
        $.get("/product/divisionCheck?collab="+collab,function(result){
            result=result.trim();
            str = "이미 있음"
            if(result=="가능"){
                str = "추가 가능"
                dCheck=true;
            }
            $("#divCheckText").text(str);

            if(checker){
                alert(dCheck);
                if(dCheck){
                    $("#frm").submit();
                }
            }
        })
    }
}

$("#duplCheck").click(function(){
    duplCheck(false);
})

$("#addBtn").click(function(){
    duplCheck(true);
})
```




```java
public int setInsertDivision(ProductDivisionVO productDivisionVO)throws Exception{
    String[] types = {"top-long","top-short","bottom-long","bottom-short"};
    int result = 0;
    for(String str:types) {
        productDivisionVO.setProductType(str);
        result += productMapper.setInsertDivision(productDivisionVO);
    }
    return result;
}
```

productType으로 들어갈 항목을 list로 미리 만들어
신규 생성되는 division에 적용

```xml
<insert id="setInsertDivision" parameterType="ProductDivisionVO">
    insert into
    productDivision (collab,producttype)
    values(#{collab},#{productType})
</insert>
```

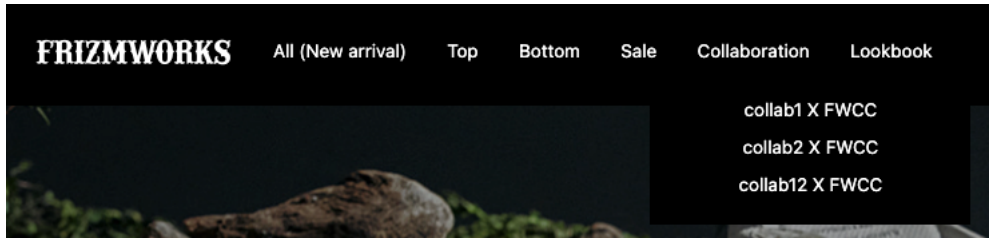
Collaboration List

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


    $.get({
        url:"/product/getDivisions",
        success:function(result){
            let hostIndex = location.href.indexOf(location.host)+location.host.length;
            let contextPath = location.href.substring(hostIndex,location.href.lastIndexOf('/',hostIndex+1))
            let step;
            for(step=0;step<result.length;step++){
                if(result[step]!="none"){
                    $("#navbar-colaul").append('<li><a href="'
                    +contextPath+'/product/list?collab='
                    +result[step]+'&productType=&name='
                    +result[step]+'&division=&sortStandard=&sale=false"> '
                    +result[step]+' X FWCC </a></li>')

                }
            }


        }
    })
});
```

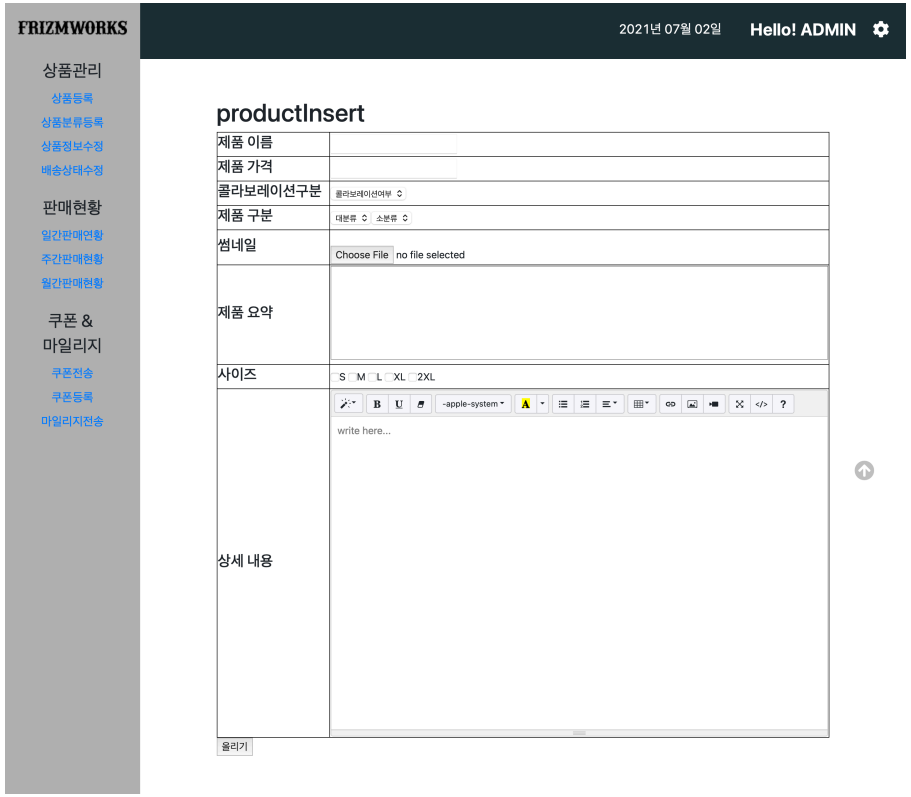
ajax를 이용하여 ContextPath 설정

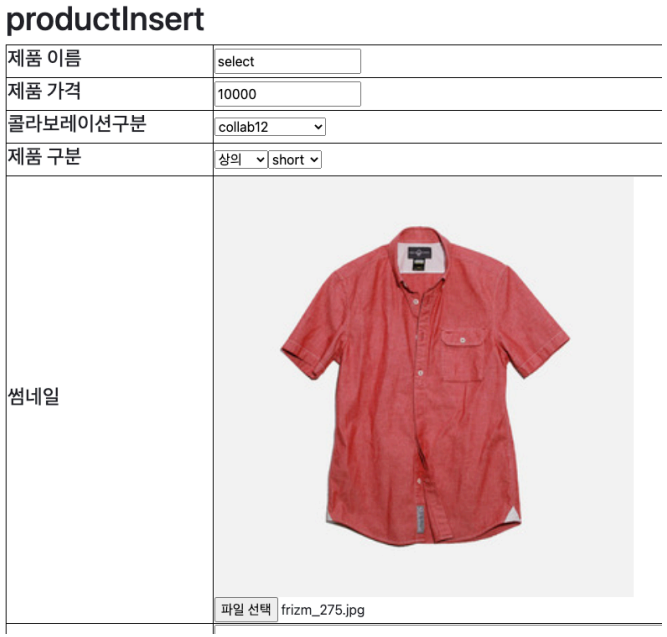
collaboration List 를 받아 navbar에 추가

```
<li id="navbar-columnlist"><a href="#">Collaboration</a>
    <ul id="navbar-colaul">

    </ul>
</li>
```



Insert






```
<div class="image-container">
    <img style="width: 500px;" id="preview-image"> <input
        style="display: block;" type="file" id="input-image"
        name="thumbnail">
</div>
```

```
function readImage(input) {

   if(input.files && input.files[0]) {


       const reader = new FileReader()

       reader.onload = e => {
           const previewImage = document.getElementById("preview-image")
           previewImage.src = e.target.result
       }

       reader.readAsDataURL(input.files[0])
   }
}

const inputImage = document.getElementById("input-image")
inputImage.addEventListener("change", e => {
   readImage(e.target)
})
```

썸네일 이미지 업로드 시 미리보기 표시

Insert

```java
public int setInsert(ProductVO productVO, String size, MultipartFile thumbNail) throws Exception{
    String type = productVO.getProductType();
    type = type.replace(",", "_");
    Long price = productVO.getProductPrice();
    Long disRate = productVO.getProductDisRate();
    if(disRate==null) {
        disRate = 0L;
    }
    Long finalPrice = price*(100-disRate)/100;
    Long mileage = ((price/100)*5);
    productVO.setFinalPrice(finalPrice);
    productVO.setProductMileage(mileage);
    productVO.setProductType(type);
    productVO.setProductSaleable(false);
    int result = productMapper.setInsert(productVO);

    String[] fileNames = productFileManager.thumbNailSave("/"+productVO.getProductNum(), thumbNail,session);
    for(String str:fileNames) {
        ProductFileVO pFileVO = new ProductFileVO();
        pFileVO.setProductNum(productVO.getProductNum());
        pFileVO.setFileName(str);
        pFileVO.setOriName(thumbNail.getOriginalFilename());
        result = productMapper.setFileInsert(pFileVO);
    }

    String[] sizeAssy = size.split(",");
    for(String str:sizeAssy) {
        ProductInfoVO pInfoVO = new ProductInfoVO();
        pInfoVO.setProductNum(productVO.getProductNum());
        pInfoVO.setSize(str);
        result = productMapper.setInsertPInfo(pInfoVO);
    }
    return result;
}
```

할인율 적용 최종 금액, 마일리지 계산후
ProductVO 수정
product table에 새로운 데이터 Insert

```xml
<insert id="setInsert" parameterType="ProductVO">
    insert into product
    (productTitle,summary,productPrice,productContents,
    finalPrice,productMileage,productSaleable,productDivNum)
    values
    (#{productTitle},#{summary},#{productPrice},#{productContents},
    #{finalPrice},#{productMileage},#{productSaleable},(
    select productDivNum from productDivision
    where collab=#{collab} and productType=#{productType}))
</insert>
```

productFiles table에 새로운 데이터 Insert

```xml
<insert id="setFileInsert" parameterType="ProductFileVO">
    insert into
    productFiles (productNum,fileName,oriName)
    values(#{productNum},#{fileName},#{oriName})
</insert>
```

productInfo table에 새로운 데이터 Insert

```xml
<insert id="setInsertPInfo" parameterType="ProductInfoVO">
    insert into
    productInfo (productNum,size)
    values(#{productNum},#{size})
</insert>
```

Insert

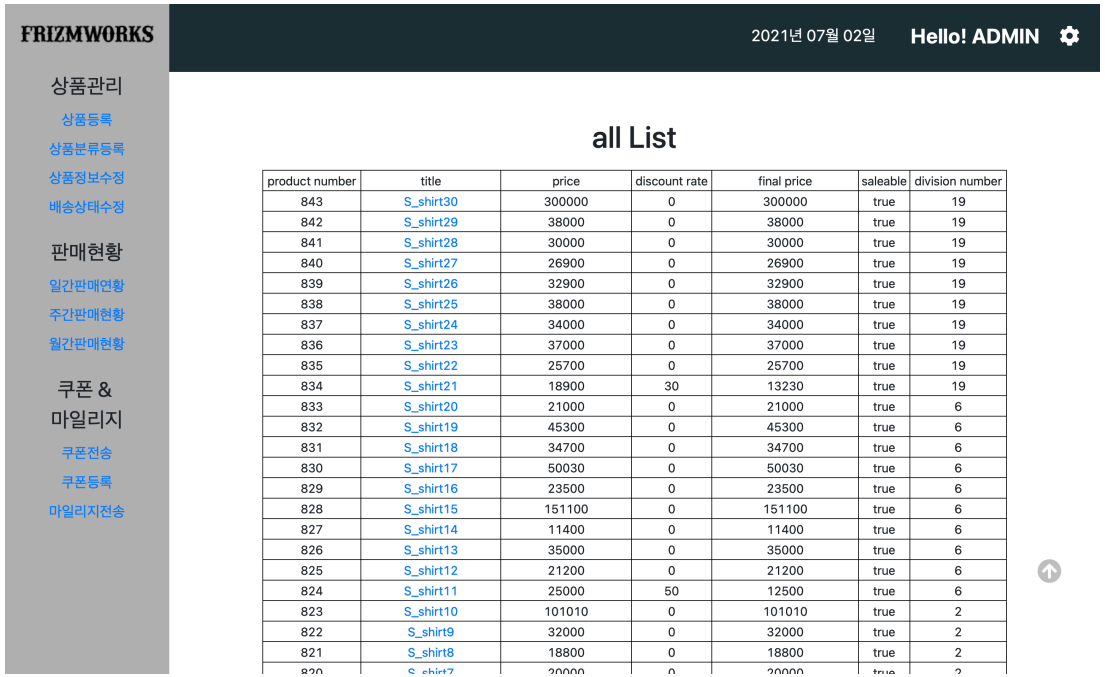
썸네일로 받아온 이미지를 임시 저장

임시저장된 이미지를 정방형 이미지로 편집(crop)후 저장(M_***)
crop된 이미지를 100px*100px로 resizing 후 저장(T_***)

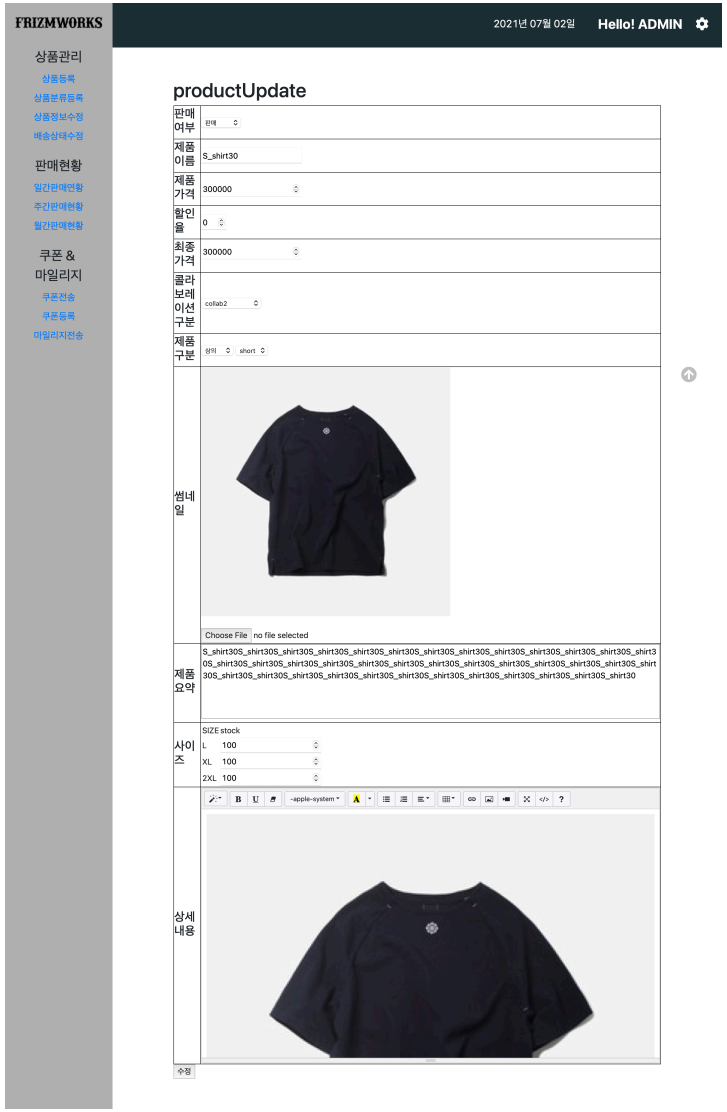
임시저장된 이미지 삭제

```java
public String[] thumbNailSave(String name, MultipartFile multipartFile,HttpSession session) throws Exception{
    // 임시파일 저장
    String[] array=new String[2];
    String path=session.getServletContext().getRealPath("resources/images/product"+name);
    File file = new File(path);

    System.out.println(file.getAbsolutePath());

    if(!file.exists()) {
        file.mkdirs();
    }

    String fileName="";

    fileName = UUID.randomUUID().toString()+"_"+multipartFile.getOriginalFilename();

    file = new File(file,fileName);


    multipartFile.transferTo(file);
    // 본파일 저장
    File file2 = new File(path,fileName);


    String ori=multipartFile.getOriginalFilename();
    String exp=ori.substring(ori.lastIndexOf(".")+1);
    String Oname=ori.substring(0,ori.lastIndexOf("."));

    BufferedImage originalImage = ImageIO.read(file.getAbsoluteFile());

    //        crop
    int ow = originalImage.getWidth();
    int oh = originalImage.getHeight();
    int nw = 0;
    int nh = 0;
    String mName="";
    if(ow>oh) {
        nw=oh;
        nh=oh;
        originalImage = Scalr.crop(originalImage, (ow-oh)/2, 0, nw, nh);
        mName = "M_"+UUID.randomUUID().toString()+"_"+ori;
        file = new File(path+"/"+mName);
        ImageIO.write(originalImage, exp, file);
    }else if(oh>ow) {
        nw=ow;
        nh=ow;
        originalImage = Scalr.crop(originalImage,0, (oh-ow)/2, nw, nh);
        mName = "M_"+UUID.randomUUID().toString()+"_"+ori;
        file = new File(path+"/"+mName);
        ImageIO.write(originalImage, exp, file);
    }else {
        mName = "M_"+UUID.randomUUID().toString()+"_"+ori;
        file = new File(path+"/"+mName);
        ImageIO.write(originalImage, exp, file);

    }

    array[0]=mName;


    MultiStepRescaleOp rescale = new MultiStepRescaleOp(100, 100);

    rescale.setUnsharpenMask(AdvancedResizeOp.UnsharpenMask.Soft);

    BufferedImage thumbImage = rescale.filter(originalImage, null);

    String finalName = "T_"+UUID.randomUUID().toString()+"_"+ori;
    file = new File(path+"/"+finalName);
    ImageIO.write(thumbImage, exp, file);
    array[1]=finalName;

    // 임시파일 삭제
    boolean check = file2.delete();
    System.out.println("delete : "+check);
    return array;
}
```

Update



product number	title	price	discount rate	final price	saleable	division number
843	S_shirt30	300000	0	300000	true	19
842	S_shirt29	38000	0	38000	true	19
841	S_shirt28	30000	0	30000	true	19
840	S_shirt27	26900	0	26900	true	19
839	S_shirt26	32900	0	32900	true	19
838	S_shirt25	38000	0	38000	true	19
837	S_shirt24	34000	0	34000	true	19
836	S_shirt23	37000	0	37000	true	19
835	S_shirt22	25700	0	25700	true	19
834	S_shirt21	18900	30	13230	true	19
833	S_shirt20	21000	0	21000	true	6
832	S_shirt19	45300	0	45300	true	6
831	S_shirt18	34700	0	34700	true	6
830	S_shirt17	50030	0	50030	true	6
829	S_shirt16	23500	0	23500	true	6
828	S_shirt15	151100	0	151100	true	6
827	S_shirt14	11400	0	11400	true	6
826	S_shirt13	35000	0	35000	true	6
825	S_shirt12	21200	0	21200	true	6
824	S_shirt11	25000	50	12500	true	6
823	S_shirt10	101010	0	101010	true	2
822	S_shirt9	32000	0	32000	true	2
821	S_shirt8	18800	0	18800	true	2
820	S_shirt7	20000	0	20000	true	2


FRIZMWORKS
2021년 07월 02일
Hello! ADMIN
상품관리
상품등록
상품분류등록
상품정보수정
배송상태수정
판매현황
일간판매연황
주간판매현황
월간판매현황
쿠폰 &
마일리지
쿠폰전송
쿠폰등록
마일리지전송
productUpdate
판매 여부
판매
제품 이름
S_shirt30
제품 가격
300000
할인율
0
최종 가격
300000
콜라보레이션 구분
collab2
제품 구분
상의
short
썸네일
Choose File
no file selected
제품 요약
S_shirt30S_shirt30S_shirt30

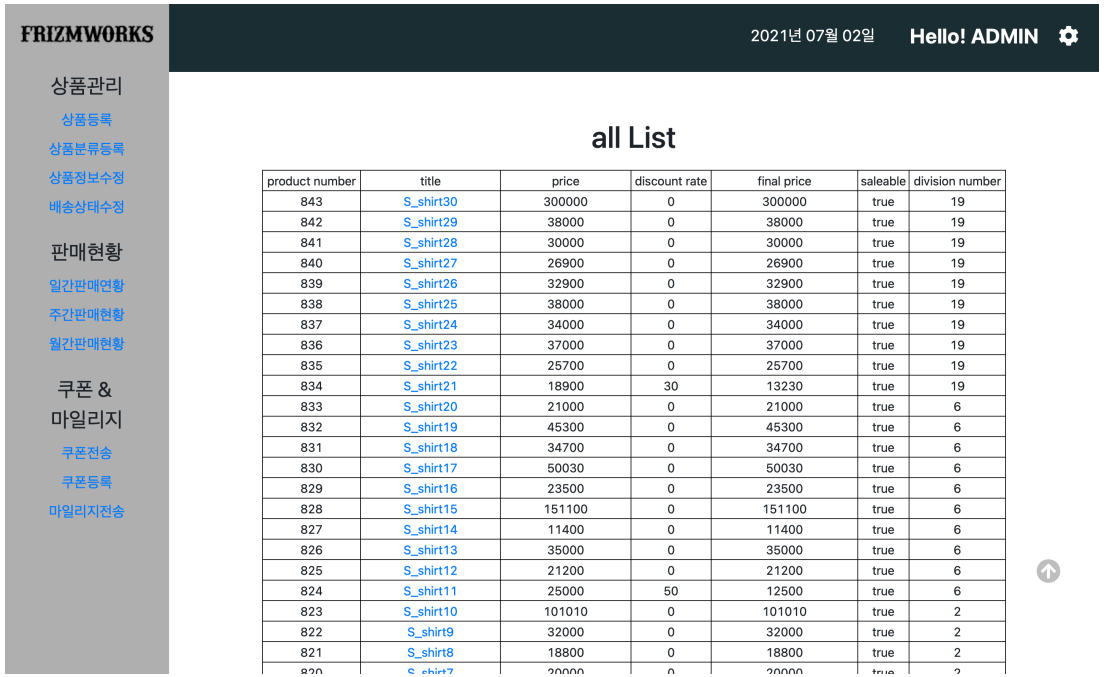
Update

```
<select id="getAllList" resultType="ProductVO">
    select * from product order
    by productNum desc
</select>
```

SQL

```
<div style="padding:5% 10%; text-align: center;">
    <h1>all List</h1>
    <table style="width: 100%; margin-top: 2%; border:1px solid black;">
        <thead>
            <tr>
                <td style="border:1px solid black;">product number</td>
                <td style="border:1px solid black;">title</td>
                <td style="border:1px solid black;">price</td>
                <td style="border:1px solid black;">discount rate</td>
                <td style="border:1px solid black;">final price</td>
                <td style="border:1px solid black;">saleable</td>
                <td style="border:1px solid black;">division number</td>
            </tr>
        </thead>
        <tbody>
            <c:forEach items="${allList}" var="product">
                <tr>
                    <td style="border:1px solid black;">${product.productNum}</td>
                    <td style="border:1px solid black;"><a href="./productUpdate?productNum=${product.productNum}">${product.productTitle}</a></td>
                    <td style="border:1px solid black;">${product.productPrice}</td>
                    <td style="border:1px solid black;">${product.productDisRate}</td>
                    <td style="border:1px solid black;">${product.finalPrice}</td>
                    <td style="border:1px solid black;">${product.productSaleable}</td>
                    <td style="border:1px solid black;">${product.productDivNum}</td>
                </tr>
            </c:forEach>
        </tbody>
    </table>
</div>
```

HTML

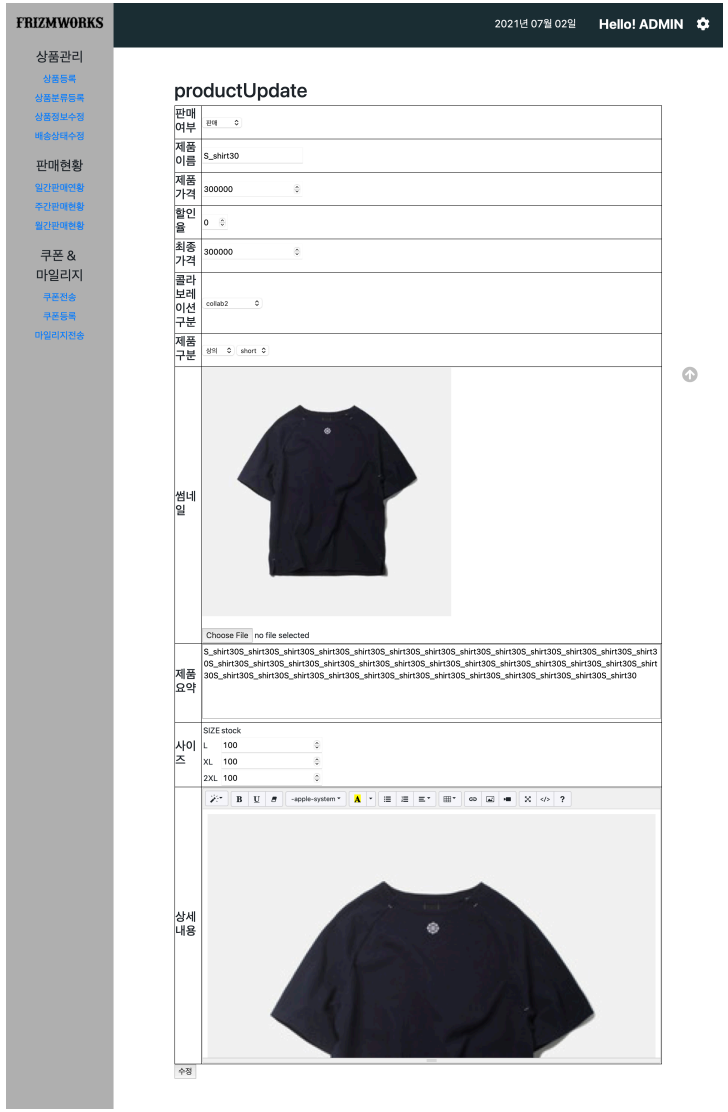


product number	title	price	discount rate	final price	saleable	division number
843	S_shirt30	300000	0	300000	true	19
842	S_shirt29	38000	0	38000	true	19
841	S_shirt28	30000	0	30000	true	19
840	S_shirt27	26900	0	26900	true	19
839	S_shirt26	32900	0	32900	true	19
838	S_shirt25	38000	0	38000	true	19
837	S_shirt24	34000	0	34000	true	19
836	S_shirt23	37000	0	37000	true	19
835	S_shirt22	25700	0	25700	true	19
834	S_shirt21	18900	30	13230	true	19
833	S_shirt20	21000	0	21000	true	6
832	S_shirt19	45300	0	45300	true	6
831	S_shirt18	34700	0	34700	true	6
830	S_shirt17	50030	0	50030	true	6
829	S_shirt16	23500	0	23500	true	6
828	S_shirt15	151100	0	151100	true	6
827	S_shirt14	11400	0	11400	true	6
826	S_shirt13	35000	0	35000	true	6
825	S_shirt12	21200	0	21200	true	6
824	S_shirt11	25000	50	12500	true	6
823	S_shirt10	101010	0	101010	true	2
822	S_shirt9	32000	0	32000	true	2
821	S_shirt8	18800	0	18800	true	2
820	S_shirt7	20000	0	20000	true	2

Update

```java
public ProductVO setUpdate(ProductVO productVO)throws Exception{
    ProductVO vo = productMapper.getProductSelect(productVO);
    vo.setFile(productMapper.getMain(vo));
    vo.setInfos(productMapper.getInfoSelect(vo));
    vo.setProductDivisionVO(productMapper.getDivisionSelect(vo));
    return vo;
}
```

```xml
<select id="getInfoSelect" parameterType="ProductVO"
    resultType="ProductInfoVO">
    select * from productInfo where productNum = #{productNum}
</select>
<select id="getFileSelect" parameterType="ProductVO"
    resultType="ProductFileVO">
    select * from productFiles where productNum = #{productNum}
</select>
<select id="getDivisionSelect" parameterType="ProductVO"
    resultType="ProductDivisionVO">
    select * from productDivision where productDivNum =
    #{productDivNum}
</select>
```



Update

제품 가격	300000
할인율	0
최종 가격	300000

제품 가격	300000
할인율	3
최종 가격	291000

ProductInsert에서 금액 설정 후 금액 직접 수정 불가
할인율을 변경하여 최종 금액 변경 가능

```
<tr>
    <td>
        <h4>제품 가격</h4>
    </td>
    <td><input type="number" readonly="readonly"
        name="productPrice" id="pPrice" value="${VO.productPrice}">
    </td>
</tr>
<tr>
    <td>
        <h4>할인율</h4>
    </td>
    <td><input type="number" name="productDisRate" id="pDisRate"
        value="${VO.productDisRate}" min="0" max="100"></td>
</tr>
<tr>
    <td>
        <h4>최종 가격</h4>
    </td>
    <td><input type="number" readonly="readonly" id="finalPrice"
        name="finalPrice" value="${VO.finalPrice}">
        <p id="finalPrice"></p></td>
</tr>
```

```
calFinalPrice = function(){
    let price = $("#pPrice").val();
    let disRate = $("#pDisRate").val();
    if(disRate>100){
        disRate=100;
        $("#pDisRate").val(disRate);
    }
    price = Math.floor(price*(100-disRate)/100);
    $("#finalPrice").val(price);
}
$("#pDisRate").blur(calFinalPrice);
```

Update

```java
@Transactional(rollbackFor = Exception.class)
public void setUpdate(ProductVO productVO,String[] sizeList, Long[] stockList, MultipartFile thumbNail) throws Exception{
    String type = productVO.getProductType();
    type = type.replace(",", "-");
    productVO.setProductType(type);
    int result = productMapper.setUpdate(productVO);
    if(sizeList.length==stockList.length) {
        for(int i=0;i<sizeList.length;i++) {
            ProductInfoVO piVO = new ProductInfoVO();
            piVO.setProductNum(productVO.getProductNum());
            piVO.setSize(sizeList[i]);
            piVO.setStock(stockList[i]);
            int result2 = productMapper.setUpdateProductInfo(piVO);
        }
    }
    if(thumbNail.getOriginalFilename().length()>0) {
        List<ProductFileVO> array = productMapper.getFileSelect(productVO);
        if(array.size()!=0) {
            productMapper.setFileDelete(productVO);
        }
        String[] fileNames = productFileManager.thumbNailSave("/"+productVO.getProductNum(), thumbNail, session);
        for(String str:fileNames) {
            ProductFileVO pFileVO = new ProductFileVO();
            pFileVO.setProductNum(productVO.getProductNum());
            pFileVO.setFileName(str);
            pFileVO.setOriName(thumbNail.getOriginalFilename());
            result = productMapper.setFileInsert(pFileVO);
        }
    }

}
```

product 테이블에 수정사항 update

productInfo 테이블에 수정사항 update

제품 정보 수정 중 썸네일 변경 시
productFiles 테이블에서 기존 데이터 삭제후
이미지 신규 저장 후
productFiles 테이블에 새로운 데이터 Insert

Board & lookbook

박한철

BoardList

Notice

NO	SUBJECT	WRITER	DATE	HIT
123	프리즘웍스 2021 SUMMER 컬렉션, 4번째 딜리버리	admin	2021-07-02	9
122	프리즘웍스 2021 SUMMER 컬렉션, 3번째 딜리버리	admin	2021-07-02	0
121	프리즘웍스 2021 SUMMER 컬렉션, 2번째 딜리버리	admin	2021-07-02	0
120	프리즘웍스 2021 SUMMER 컬렉션 첫번째 발매	admin	2021-07-02	0
119	TAW&TOE® X 프리즘웍스 협업 ZERO VITY™ FLIP-FLOP 발매	admin	2021-07-02	0
118	펜필드 X 프리즘웍스 2021 S/S 세일 안내	admin	2021-07-02	0
117	프리즘웍스 2021 S/S 미드 세일 안내	admin	2021-07-02	0
116	프리즘웍스 2021 S/S 컬렉션, 5번째 딜리버리	admin	2021-07-02	0
115	2021 S/S 펜필드 X 프리즘웍스 협업컬렉션 발매	admin	2021-07-02	0
114	프리즘웍스 2021 S/S 컬렉션, 3번째 딜리버리	admin	2021-07-02	0
113	프리즘웍스 2021 S/S 컬렉션, 두번째 딜리버리	admin	2021-07-02	1
112	프리즘웍스 2021 S/S 컬렉션, 3번째 딜리버리	admin	2021-07-02	1

글쓰기

< 1 2 3 4 5 >

제목 find

```xml
<mapper namespace="com.fw.s1.board.notice.NoticeMapper">
    <sql id="searcher">
    <!-- 공통으로 사용하는 sql문 -->
    <where>
            <choose>
                <when test="kind=='Writer'">
                    writer like concat('%', #{search}, '%')
                </when>

                <when test="kind=='contents'">
                    contents like concat('%', #{search}, '%')
                </when>

                <otherwise>
                    title like concat('%', #{search}, '%')
                </otherwise>
            </choose>
        </where>
    </sql>


    <select id="getList" resultType="NoticeVO" parameterType="Pager">
        select * from notice
        <include refid="searcher"></include>
        order by num desc
        limit #{startRow}, #{perPage}
        <!-- limit 시작index번호(0부터시작), 갯수 -->
    </select>

    <select id="getTotalCount" parameterType="Pager" resultType="java.lang.Long">
        select count(num) from notice
        <include refid="searcher"></include>
    </select>
```

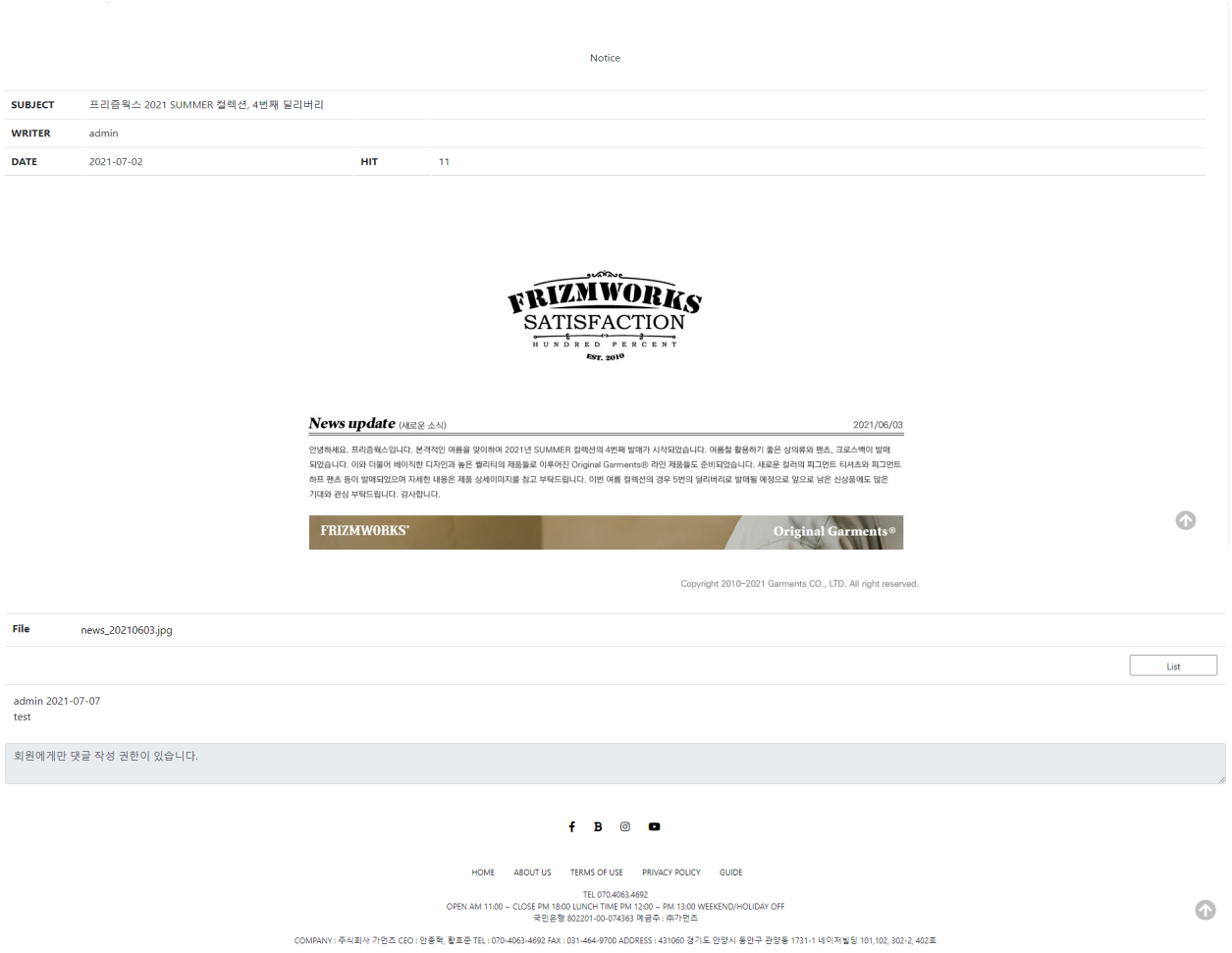
```xml
<sec:authorize access="isAuthenticated()">
    <sec:authentication property="principal.username" var="user_id" />
    <c:if test="${user_id == 'admin'}">
        <div style="text-align: right">
            <a href="./insert" class="btn " role="button"
                style="border: 1px solid gray; width: 100px; height: 30px; font-size: 14px;">글쓰기</a>
        </div>
    </c:if>

</sec:authorize>
```

12개씩 페이징 처리 및 검색기능 추가

Admin으로 로그인 했을 때 만 글쓰기 버튼 활성화

쎄큐리티에서 로그인한 아이디가 admin인지 비교

BoardSelect




```
<select id="getSelect" resultMap="selectResult" parameterType="BoardVO">
    SELECT N.*, NF.* from
    notice N left join noticeFiles NF
    on N.num = NF.num
    WHERE N.num=#{num}
</select>

<resultMap type="NoticeVO" id="selectResult">
    <id column="num" property="num"/>
    <result column="title" property="title"/>
    <result column="writer" property="writer"/>
    <result column="contents" property="contents"/>
    <result column="regDate" property="regDate"/>
    <result column="hit" property="hit"/>
    <collection property="files" javaType="java.util.List" ofType="BoardFileVO">
        <id column="filenum" property="fileNum"/>
        <result column="fileName" property="fileName"/>
        <result column="oriName" property="oriName"/>
    </collection>
</resultMap>
```

글 정보, 글 댓글과 글 번호에 해당하는 파일들을 불러옴

BoardComment

Notice

SUBJECT	프리즘웍스 2021 SUMMER 컬렉션, 4번째 딜리버리		
WRITER	admin		
DATE	2021-07-02	HIT	11

FRIZMWORKS
SATISFACTION
HUNDRED PERCENT
EST. 2010

News update (새로운 소식) 2021/06/03

안녕하세요. 프리즘웍스입니다. 본격적인 여름을 맞이하여 2021년 SUMMER 컬렉션의 4번째 발매가 시작되었습니다. 여름철 활용하기 좋은 상의류와 팬츠, 크로스백이 발매되었습니다. 이와 더불어 베이직한 디자인과 높은 퀄리티의 제품들로 이루어진 Original Garments® 라인 제품들도 준비되었습니다. 새로운 컬러의 피그먼트 티셔츠와 피그먼트 하프 팬츠 등이 발매되었으며 자세한 내용은 제품 상세이미지를 참고 부탁드립니다. 이번 여름 컬렉션의 경우 5번의 딜리버리로 발매될 예정으로 앞으로 남은 신상품에도 많은 기대와 관심 부탁드립니다. 감사합니다.

언제 어디서나 가볍게 활용하기 좋은 복조리 모양의 크로스백 입니다. 각기 다른 3개의 원단으로 제작되어 색다른 느낌을 주며, 깔끔하면서도 특색있는 실루엣으로 코디시 포인트 주기 좋은 제품입니다.

프리즘웍스 공식 홈페이지를 통해 5만원 이상의 2021 여름 신상품 구매하실 경우 SSL 핀버튼 세트(7종)를 증정합니다. 한정 수량으로 제공 되며 선착순 1,000 분에게 드립니다.

Copyright 2010-2021 Garments CO., LTD. All right reserved.

File	news_20210603.jpg

List

admin 2021-07-07 MODIFY DELETE
test

확인

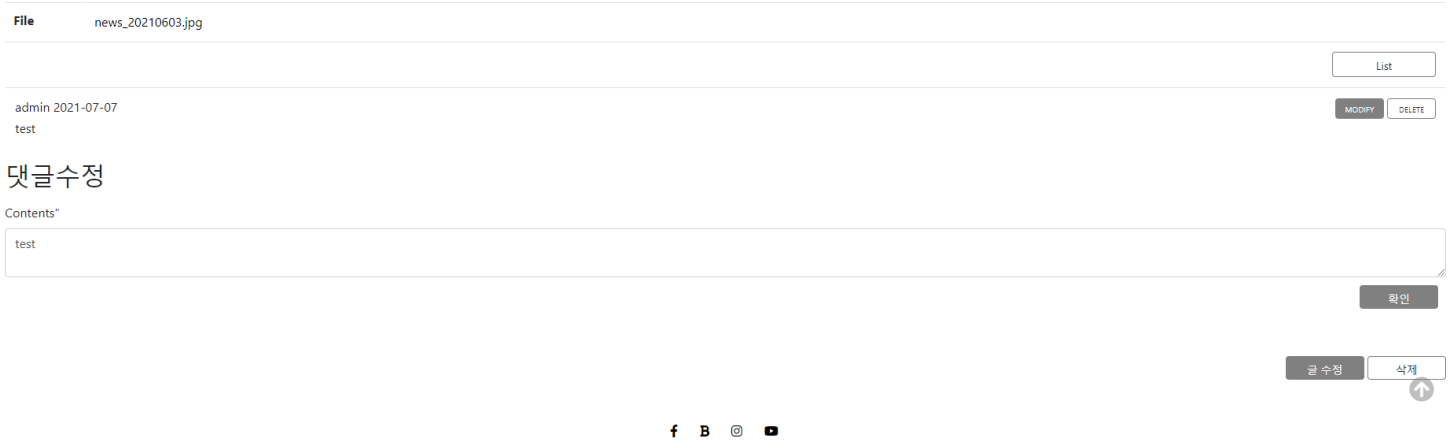
글 수정 삭제

```
<select id="commentList" parameterType="BoardCommentVO" resultType="BoardCommentVO">
    select * from noticeComment where num=#{num}
</select>

<select id="commentSelect" parameterType="BoardCommentVO" resultType="BoardCommentVO">
    select * from noticeComment where commentNum=#{commentNum}
</select>
```

```
<sec:authorize access="isAuthenticated()">
    <sec:authentication property="principal.username"
        var="user_id" />
    <c:if test="${com.writer == user_id }">
        <div class="col-sm-6" style="text-align: right;">
            <a
                href="./commentUpdate?commentNum=${com.commentNum}&num=${com.num}"
                class="btn"
                style="background: gray; color: white; width: 62px; height: 26px; font-size: 10px;">MODIFY</a>
            <a href="./commentDelete?commentNum=${com.commentNum}"
                id="del" class="btn"
                style="border: 1px solid gray; width: 62px; height: 26px; font-size: 10px;">DELETE</a>
        </div>

    </c:if>
</sec:authorize>
```

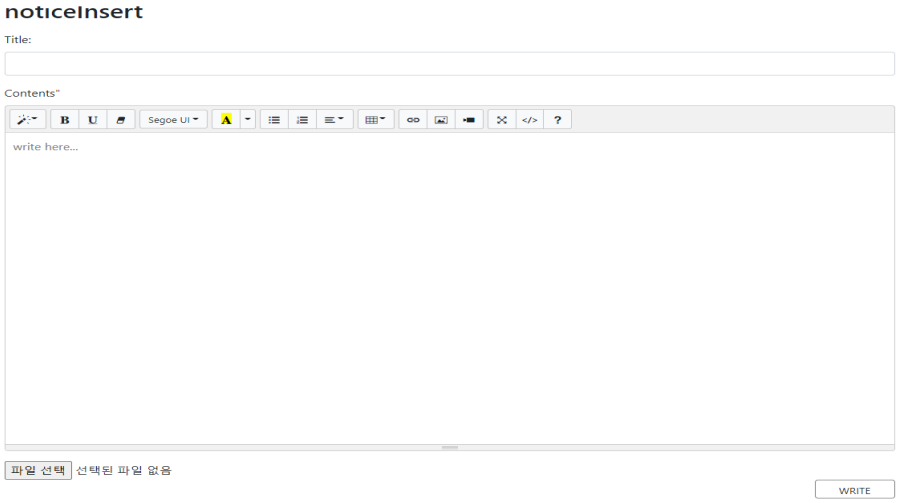
로그인할 경우 댓글창이 활성화 되고 해당아이디의 댓글을 수정, 삭제 할 수 있다.

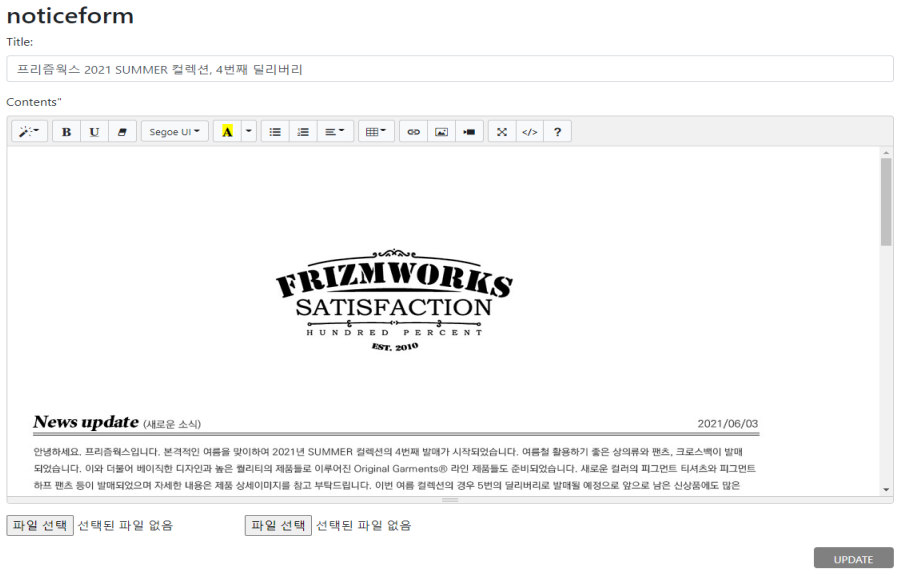
Comment Update,Delete




```
<update id="commentUpdate" parameterType="BoardCommentVO">
    update noticeComment set contents=#{contents}, regDate=now() where commentNum=#{commentNum}
</update>

<delete id="commentDelete" parameterType="BoardCommentVO">
    delete from noticeComment where commentNum=#{commentNum}
</delete>
```

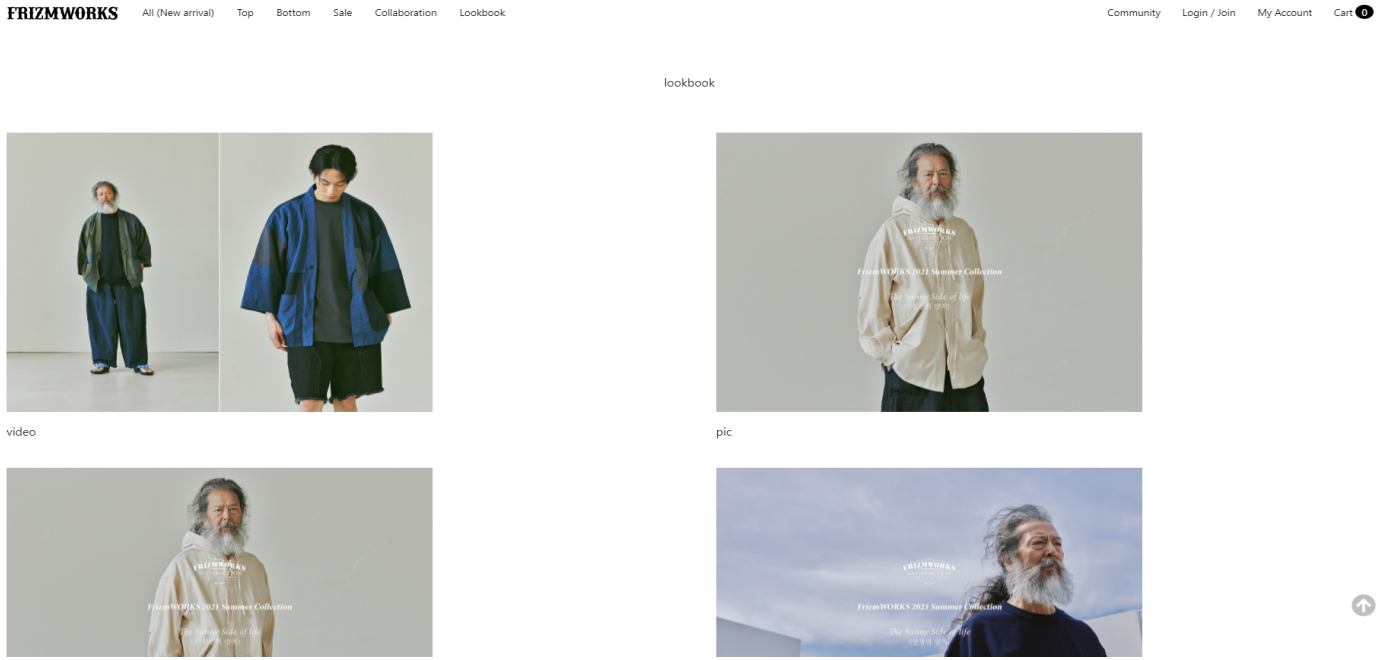
Board Update,Delete,Insert






```
<update id="setHitUpdate" parameterType="BoardVO">
    update notice set hit=hit+1 where num=#{num}
</update>

<insert id="setInsert" parameterType="BoardVO" useGeneratedKeys="true" keyProperty="num">
    insert into notice (title,writer,contents,regDate,hit)
    values(#{title}, #{writer},#{contents}, now(), 0)
</insert>

<insert id="setFileInsert" parameterType="BoardFileVO">
    insert into noticeFiles (num, filename, oriname)
    values (#{num}, #{fileName}, #{oriName})
</insert>

<update id="setUpdate" parameterType="BoardVO" useGeneratedKeys="true" keyProperty="num">
    update notice set title=#{title}, contents=#{contents} where num=#{num}
</update>

<delete id="setDelete" parameterType="BoardVO">
    SET foreign_key_checks = 0;
    delete from notice where num=#{num};
    delete from noticefiles where num=#{num};
    delete from noticecomment where num=#{num};
    SET foreign_key_checks = 1
</delete>
```

LookbookList

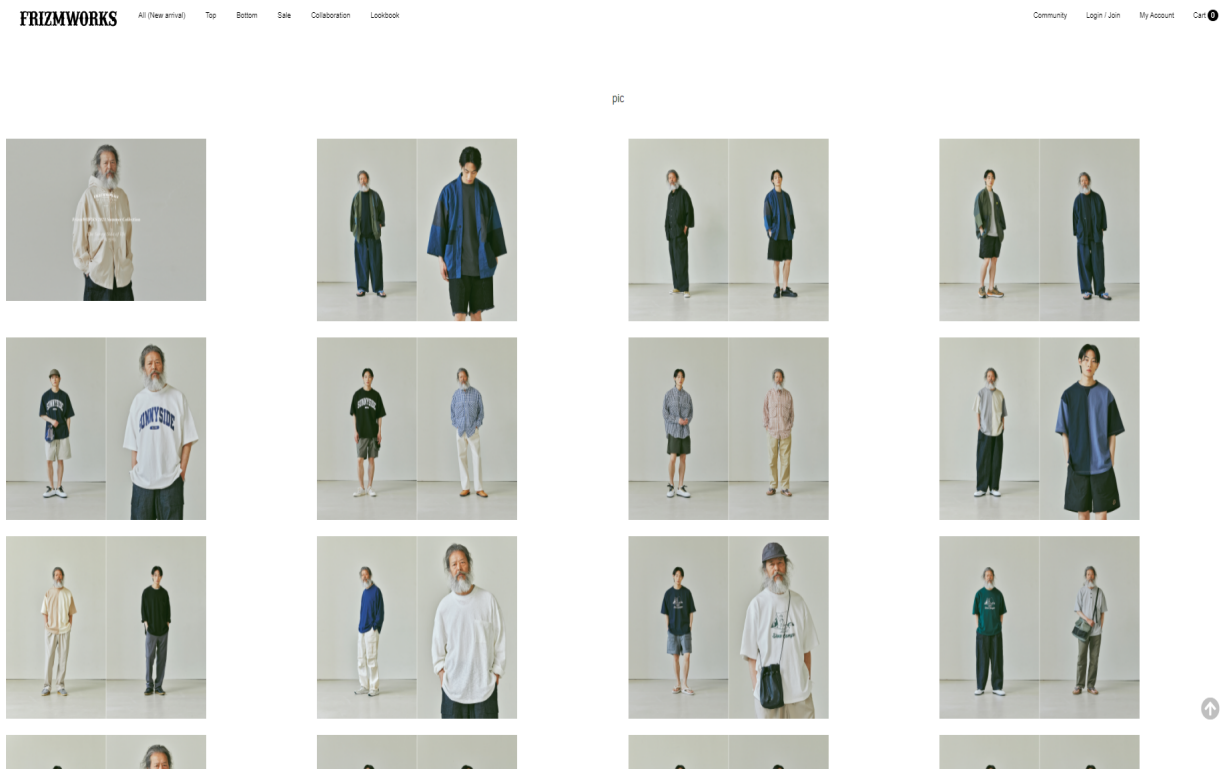


Lookbook에 해당하는 번호의 lookbookFile에서 첫번째 이미지를 가져와서 리스트의 메인 이미지로 가져옴

한 화면에 12개로 페이징 처리하고 6행 2열로 되게 정렬 시킴

```
<select id="List" resultType="LookBookVO" parameterType="Pager">
    SELECT  lookbook.lookbookNum,lookbook.division,lookbook.title,lookbookfiles.fileName from lookbook, lookbookfiles
    where lookbook.lookbookNum = lookbookfiles.num GROUP BY lookbooknum order by lookbookNum DESC
    limit #{startRow}, #{perPage}
</select>

<select id="getTotalCount" parameterType="Pager" resultType="java.lang.Long">
    SELECT count(DISTINCT num) from lookbookfiles

</select>
```

LookbookSelect(Picture)

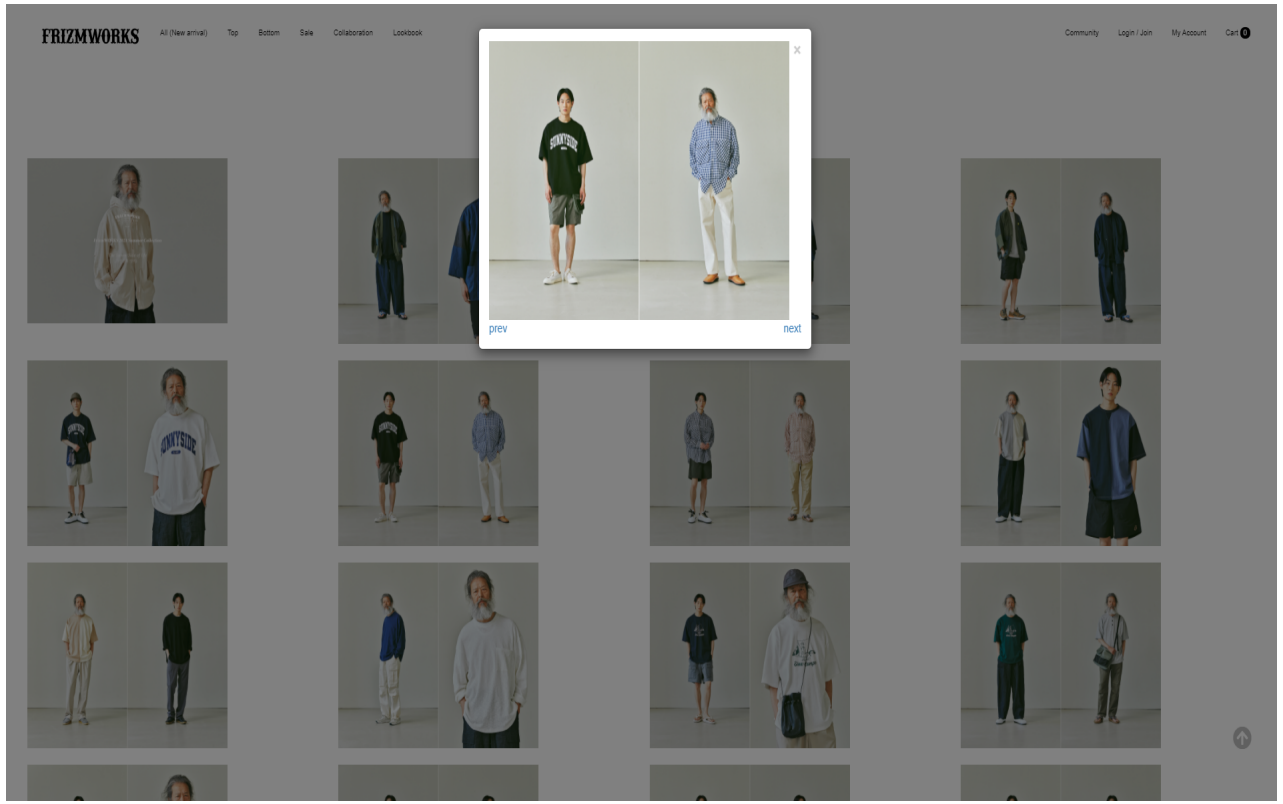



```xml
<select id="getSelect" resultMap="selectResult" parameterType="LBSelectVO">
    SELECT N.*, NF.* from
    lookbook N left join lookbookFiles NF
    on N.lookbookNum = NF.num
    WHERE N.lookbookNum=#{lookbookNum}
</select>

<resultMap type="LBSelectVO" id="selectResult">
    <id column="lookbookNum" property="lookbookNum"/>
    <result column="division" property="division"/>
    <result column="title" property="title"/>
    <collection property="files" javaType="java.util.List" ofType="LookBookFileVO">
        <id column="lookbookFileNum" property="lookbookFileNum"/>
        <result column="num" property="num"/>
        <result column="fileName" property="fileName"/>
        <result column="oriName" property="oriName"/>
    </collection>
</resultMap>
```

```xml
<c:if test="${vo.division == 'p'}">
    <c:forEach items="${vo.files}" var="fileVO">

    <div class="col-sm-3" style="padding-bottom: 20px;">

        <div>
        <a href="./modal?lookbookFileNum=${fileVO.lookbookFileNum}" data-toggle="modal" data-target="#myModal">
        <img alt="ss" style="width: 300px; " src="../upload/lookbook/${fileVO.fileName}">
        </a>
        </div>
    </div>

    </c:forEach>

</c:if>
```

Lookbook에서 그림인지 비디오인지 판단하는 칼럼값이 p일 경우 그림들을 4열로 나열 한다

LookbookModal

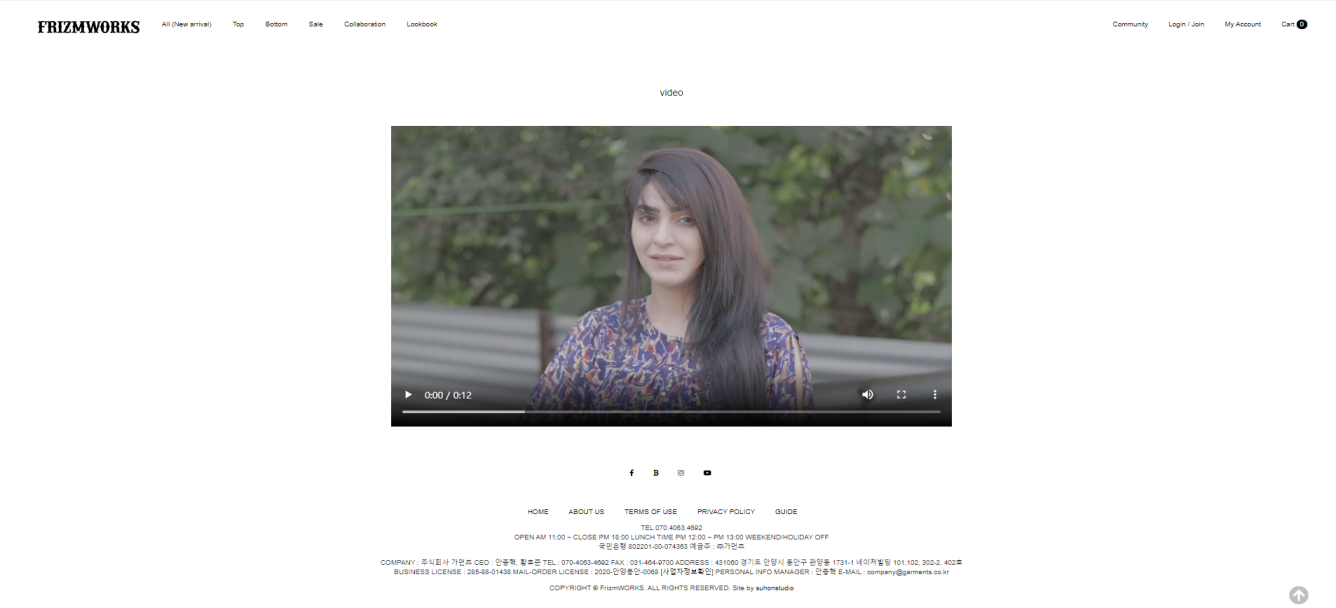



```
<!-- Modal HTML -->
<div id="myModal" class="modal fade">
    <div class="modal-dialog">
        <div class="modal-content">
            <!-- Content will be loaded here from "remote.php" file -->
        </div>
    </div>
</div>
```

```
<div class="modal-body">
<button class="close" type="button" data-dismiss="modal" aria-label="Close">
    <span aria-hidden="true">x</span>
  </button>

  <img alt="ss" style="width: 450px; " src="../upload/lookbook/${md.fileName}">

    <div class="row">
    <div class="col"> <a href="./modal?lookbookFileNum=${md.lookbookFileNum - 1}" data-toggle="modal" data-target="#myModal">prev</a></div>
    <div class="col" style="text-align: right"> <a href="./modal?lookbookFileNum=${md.lookbookFileNum + 1}" data-toggle="modal" data-target="#myModal">next</a></div>
    </div>
  </div>
```

해당 사진을 클릭하면 해당 그림의 번호를 모달jsp로 보내어서 해당이미지 하나만 모달jsp에 띄우는데 lookbookSelect에서 모달jsp를 modal형태로 불러옴

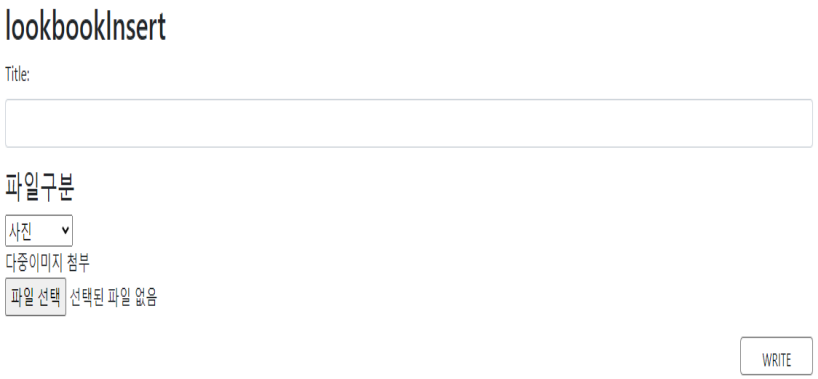
LookbookVideo

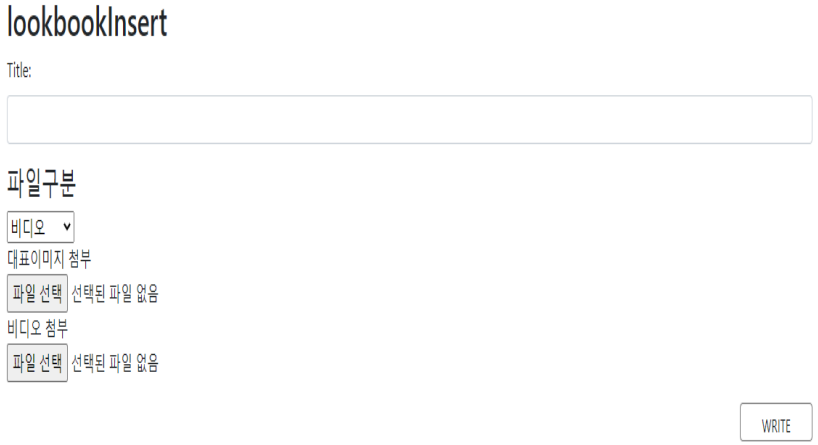


Lookbook에서 그림인지 비디오인지 판단하는 칼럼값이 v일 경우 두번째 저장된 파일을 파씽해서 마지막이 mp4일 경우 해당 파일의 경로를 video태그에 넣어서 동영상이 나오게 함

```
<c:if test="${vo.division == 'v'}">

    <c:forEach items="${vo.files}" var="fileVO">
    <c:set var="fileName" value="${fn:split(fileVO.fileName,('.'))}"></c:set>


    <c:if test="${fileName[fn:length(fileName)-1] == 'mp4'}">
    <video src="../upload/lookbook/${fileVO.fileName}" controls="controls" style="width: 800px;"></video>
    </c:if>
</c:forEach>
</c:if>
```

LookbookInsert





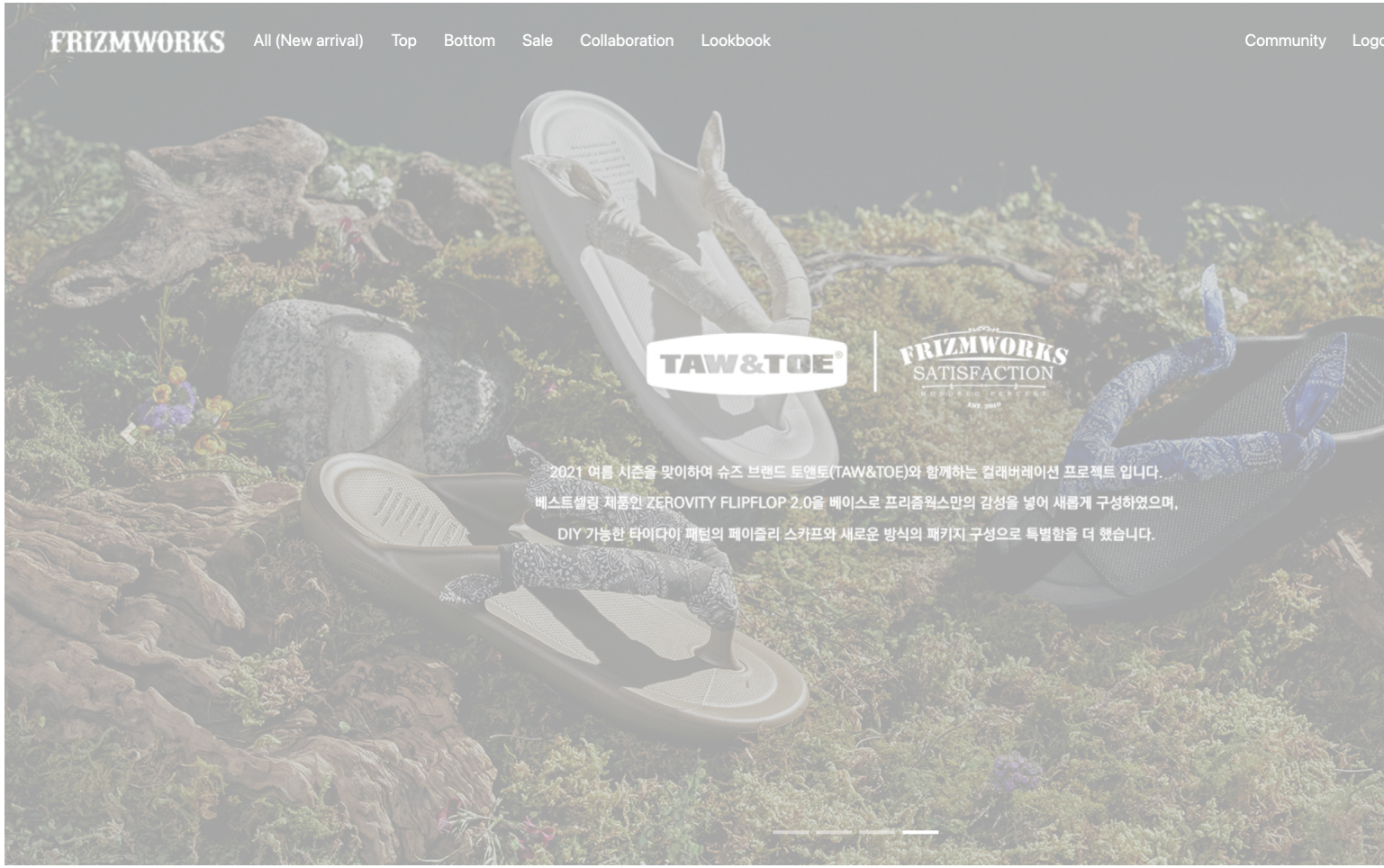

```
        <form id="frm" action="./insert" method="post" enctype="multipart/form-data">
            <div class="form-group">
                <label for="title">Title:</label> <input type="text"
                    class="form-control myCheck" id="title" name="title">
            </div>

            <h4>파일구분</h4>
            <select name="division" id="division" onchange="handleOnChange(this)">
            <option >종류 선택</option>
            <option value="p">사진</option>
            <option value="v">비디오</option>
            </select>
            <div id='result'></div>



            <div class="form-group" id="pic">
            <div>다중이미지 첨부</div>
                <input type="file" name="files" multiple="multiple">
            </div>


            <div class="form-group" id="vdo">
                <div>대표이미지 첨부</div>
                <input type="file" name="files">
                <div>비디오 첨부</div>
                <input type="file" name="files">
            </div>


            <div style="text-align: right">
        <input type="button" id="btn" value="WRITE" class="btn" style="border:1px solid gray;
                    width:100px;
                    height:30px;
                    font-size:14px;">
        </div>
        </form>
    </div>

</div>
</div>
<c:import url="${pageContext.request.contextPath}/WEB-INF/views/templates/footer.jsp"></c:import>

<script type="text/javascript">
$('#pic').hide();
$('#vdo').hide();
function handleOnChange(e) {
      // 선택된 데이터의 텍스트값 가져오기
      const text = e.options[e.selectedIndex].text;

      if(text == '사진'){
          $('#pic').show();
          $('#vdo').hide();
        }else{
          $('#pic').hide();
          $('#vdo').show();
        }
    }
</script>
```

Js를 이용하여 콤보박스에서 사진을 고르면 사진입력폼만 화면에 나타내고 동영상을 고르면 동영상폼만 화면에 나타낸다

Member

임형수

CONFIG


FRIZMWORKS
All (New arrival)
Top
Bottom
Sale
Collaboration
Lookbook
Community
TAW&TOE
FRIZMWORKS
SATISFACTION
2021 여름 시즌을 맞이하여 슈즈 브랜드 토앤토(TAW&TOE)와 함께하는 컬래버레이션 프로젝트 입니다.
베스트셀링 제품인 ZEROVITY FLIPFLOP 2.0을 베이스로 프리즘웍스만의 감성을 넣어 새롭게 구성하였으며,
DIY 가능한 타이다이 패턴의 페이즐리 스카프와 새로운 방식의 패키지 구성으로 특별함을 더 했습니다.

SPRING SECURITY

```java
@Override
protected void configure(HttpSecurity http) throws Exception {
    // URL에 따른 로그인, 권한 설정
    http
        .cors().and()
        .csrf().disable()
        .authorizeRequests()
//          .antMatchers("/**").permitAll()
            .antMatchers("/").permitAll()
            // --- community START ---
            .antMatchers("/notice/list", "/qna/list","/review/list").permitAll()
            .antMatchers("/notice/select", "/qna/select","/review/select").permitAll()
            .antMatchers("/lookbook/list", "/lookbook/select","/lookbook/modal","/stockist").permitAll()
            .antMatchers("/notice/**").hasAnyRole("MEMBER","ADMIN")
            // --- community END ---
            // --- product START ---
            .antMatchers("/product/**").permitAll()
//          .antMatchers("/product/list").permitAll()
//          .antMatchers("/product/select").permitAll()
//          .antMatchers("/product/**").hasRole("ADMIN")
            // --- product END ---
            // --- address START ---
            .antMatchers("/address/**").hasAnyRole("MEMBER","ADMIN")
            // --- address END ---
            // --- cart END ---
            // --- orderlist START
            .antMatchers("/order/**").hasAnyRole("MEMBER", "ADMIN")
            // --- orderlist END ---
            // --- admin START
            .antMatchers("/admin/**").hasRole("ADMIN")
            // --- admin END
            // --- member START ---
            .antMatchers("/member/memberJoin", "/member/memberLogin", "/member/usernameCheck","/member/finder/**").permitAll()
            .antMatchers("/member/memberPage/**").hasAnyRole("MEMBER", "ADMIN")
            // --- member END ---
            .anyRequest().authenticated()
        .and()
        .formLogin()
            .loginPage("/member/memberLogin")
            .defaultSuccessUrl("/member/memberLoginSuccess")
            .failureUrl("/member/memberLoginFail")
            .permitAll()
        .and()
        .logout()
            .logoutUrl("/member/memberLogout")
            .logoutSuccessUrl("/")
            .invalidateHttpSession(false)

            ;
}

}
```

CORS, CSRF disable을 통한 보안 강화

URL에 따른 접근 권한 부여

권한 필요한 URL 접근 시 로그인 페이지 이동

로그아웃 설정

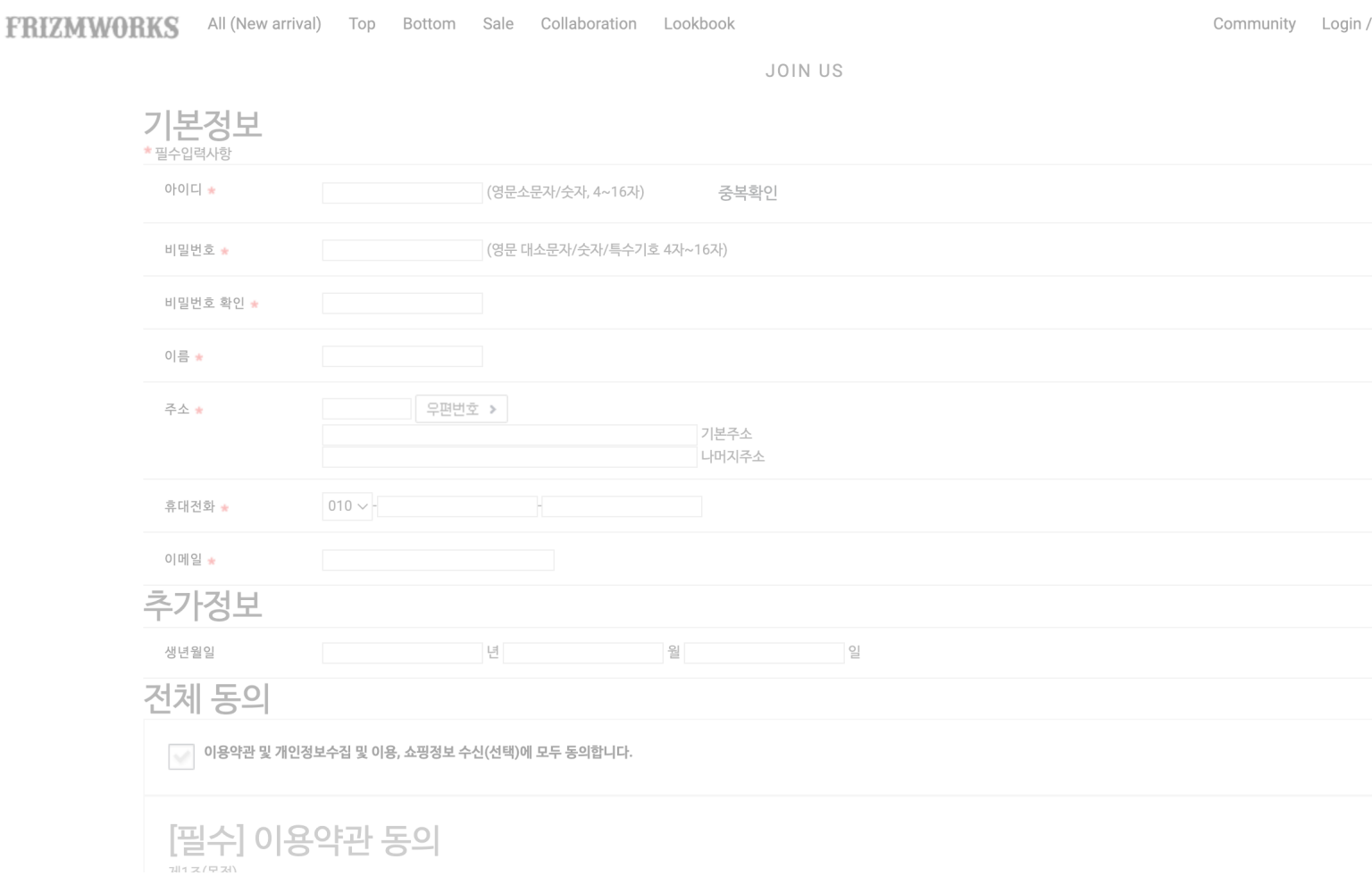
```java
@Bean
public PasswordEncoder passowrdEncoder() {
    return new BCryptPasswordEncoder();
}
```

PasswordEncoder를 통한 비밀번호 암호화

```java
@Override
public void configure(WebSecurity web) throws Exception {
    // Security 무시 경로 설정
    web.ignoring()
        .antMatchers("/resources/**")
        .antMatchers("/images/**")
        .antMatchers("/css/**")
        .antMatchers("/js/**")
        .antMatchers("vendor/**")
        .antMatchers("/favicon/**")
        .antMatchers("/upload/**")
        ;
}
```

URL 중 WebSecurity를 무시할 경로 지정

MEMBER JOIN


FRIZMWORKS
All (New arrival)
Top
Bottom
Sale
Collaboration
Lookbook
Community
Login /
JOIN US
기본정보
* 필수입력사항
아이디 *
(영문소문자/숫자, 4~16자)
중복확인
비밀번호 *
(영문 대소문자/숫자/특수기호 4자~16자)
비밀번호 확인 *
이름 *
주소 *
우편번호
기본주소
나머지주소
휴대전화 *
010
이메일 *
추가정보
생년월일
년
월
일
전체 동의
이용약관 및 개인정보수집 및 이용, 쇼핑정보 수신(선택)에 모두 동의합니다.
[필수] 이용약관 동의

MEMBER JOIN - Validation

```html
<div class="container">
    <form:form modelAttribute="memberVO" action="./memberJoin" method="POST">
        <div class="titleArea">
            <h2>JOIN US</h2>
        </div>
        <h3 class=" ">기본정보</h3>
        <p class="required ">
            <img src="//img.echosting.cafe24.com/skin/base/common/ico_required.gif" alt="필수" /> 필수입력사항
        </p>
        <div class="ec-base-table typeWrite">
            <table border="" summary="">
                <caption>회원 기본정보</caption>
                <colgroup>
                    <col style="width: 150px;" />
                    <col style="width: auto;" />
                </colgroup>
                <tbody>
                    <tr>
                        <th scope="row">아이디 <img
                                src="//img.echosting.cafe24.com/skin/base/common/ico_required.gif"
                                alt="필수" /></th>
                        <td>
                            <form:input id="usernameT" path="username" class="inputTypeText eraser"
                                placeholder="" type="text" value="" /> <span id="usernameError"
                                class="errorUsername error1"></span> <span id="usernameSuccess"
                                class="errorUsername"></span> <span id="usernameMsg"></span> (영문소문자/숫자,
                            4~16자) <a style="margin-left: 50px;" href="#" id="usernameCheck"
                                class="btn btn-sm">중복확인</a>
                        </td>
                    </tr>
                    <tr>
                        <th scope="row">비밀번호 <img
                                src="//img.echosting.cafe24.com/skin/base/common/ico_required.gif"
                                alt="필수" /></th>
                        <td>
                            <form:password id="passwordT" path="password" class="eraser"
                                autocomplete="off" minlength="4" maxlength="16" value="" /> <span
                                id="passwordError" class="errorPassword error1"></span> (영문 대소문자/숫자/특수기호
                            4자~16자)
                        </td>
                    </tr>
                    <tr>
                        <th scope="row">비밀번호 확인 <img
                                src="//img.echosting.cafe24.com/skin/base/common/ico_required.gif"
                                alt="필수" /></th>
                        <td>
                            <form:password id="passwordCheckT" path="passwordCheck" class="eraser"
                                autocomplete="off" minlength="4" maxlength="16" value="" /> <span
                                id="pwCheckError" class="errorPassword error1"></span> <span
                                id="pwCheckSuccess" class="errorPassword"></span> <span
                                id="passwordCheckMsg"></span>
                        </td>
                    </tr>
```

```javascript
// password 비어있으면 passwordCheck 비우기 ==================
$("#passwordT").blur(function(){
    password = $(this).val();
    let pwLength = password.length;
    if(pwLength == 0) {
        $("#passwordCheckT").val("");
    } else if(pwLength <4) {
        $("#passwordError").html("비밀번호는 4자리 이상이어야 합니다");
    } else if(pwLength > 4) {
        $("#passwordError").html("");
    } else if(pwLength > 16) {
        $("#passwordError").html("비밀번호는 16자리 이하이어야 합니다");
    }
});
```

```javascript
// passwordCheck ==========================================
$("#passwordCheckT").blur(function(){
    $(".errorPassword").each(function(){
        $(this).html("");
    });
    password = $("#passwordT").val();
    passwordCheck = $("#passwordCheckT").val();
    if(password == passwordCheck){
        $("#pwCheckSuccess").html("비밀번호가 일치합니다");
    }else {
        $("#pwCheckError").html("비밀번호가 일치하지 않습니다");
    }
});
```

```javascript
// input 커서 집어넣으면 값 초기화 ========================
$(".eraser").click(function(){
    $(this).val("");
    $(".errorUsername").html("");
});

$(".eraser").each(function(){
    $(this).blur(function(){
        let eraser = $(this).val();
        if(eraser.length == 0){
            $(this).next().html("필수입력사항 입니다");
        }else if (eraser.length >= 2){
            $(this).next().html("");
        }
    });
});
```

- JavaScript에서 1차적으로 input 검증
- 검증 후 재입력 방지를 위해 focuse 시, 데이터 초기화
- 검증 실패 종류에 따라 다른 오류 메세지 출력

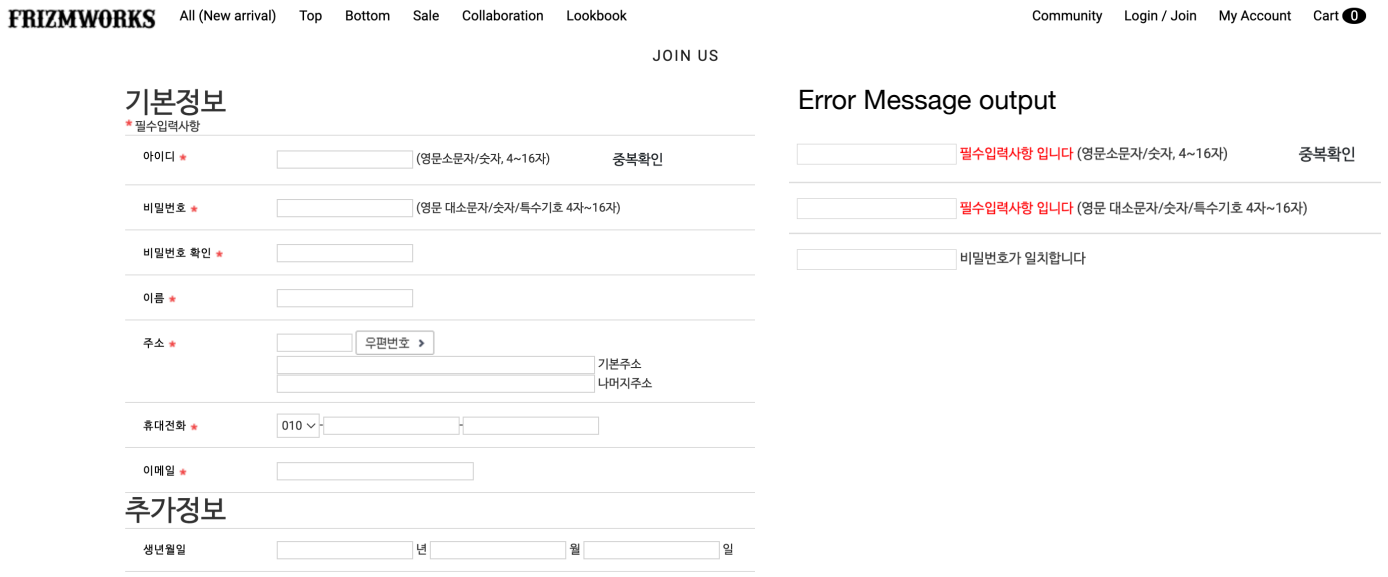
```javascript
// ID 중복 체크 ==========================================================
$("#usernameCheck").click(function(){
    $(".errorUsername").html("");
    username = $("#usernameT").val();
    console.log(username);
    $.ajax({
        type:"POST",
        url:"./usernameCheck",
        data:{username:username},
        success:function(data, status, xhr){
            console.log(data);
            console.log(status);
            console.log(xhr);
            if(data == 0){
                // 중복됐을때
                $("#usernameError").html("중복된 ID입니다");
            }else {
                // 사용가능할때
                $("#usernameSuccess").html("사용가능한 ID입니다");
            }
        }
    });
});
```

```java
    @PostMapping("usernameCheck")
    @ResponseBody
    public int getUsernameCheck(MemberVO memberVO) throws Exception {
//      ModelAndView mv = new ModelAndView();
        int result = memberService.getUsernameCheck(memberVO);
//      mv.addObject("result", result);
//      mv.setViewName("common/ajaxResult");
        return result;
    }
```

- ID 중복 여부 Ajax 통해 비동기 방식으로 처리

MEMBER JOIN - Validation

```html
<div class="container">
    <form:form modelAttribute="memberVO" action="./memberJoin" method="POST">
        <div class="titleArea">
            <h2>JOIN US</h2>
        </div>
        <h3 class=" ">기본정보</h3>
        <p class="required ">
            <img src="//img.echosting.cafe24.com/skin/base/common/ico_required.gif" alt="필수" /> 필수입력사항
        </p>
        <div class="ec-base-table typeWrite">
            <table border="" summary="">
                <caption>회원 기본정보</caption>
                <colgroup>
                    <col style="width: 150px;" />
                    <col style="width: auto;" />
                </colgroup>
                <tbody>
                    <tr>
                        <th scope="row">아이디 <img
                                src="//img.echosting.cafe24.com/skin/base/common/ico_required.gif"
                                alt="필수" /></th>
                        <td>
                            <form:input id="usernameT" path="username" class="inputTypeText eraser"
                                placeholder="" type="text" value="" /> <span id="usernameError"
                                class="errorUsername error1"></span> <span id="usernameSuccess"
                                class="errorUsername"></span> <span id="usernameMsg"></span> (영문소문자/숫자,
                            4~16자) <a style="margin-left: 50px;" href="#" id="usernameCheck"
                                class="btn btn-sm">중복확인</a>
                        </td>
                    </tr>
                    <tr>
                        <th scope="row">비밀번호 <img
                                src="//img.echosting.cafe24.com/skin/base/common/ico_required.gif"
                                alt="필수" /></th>
                        <td>
                            <form:password id="passwordT" path="password" class="eraser"
                                autocomplete="off" minlength="4" maxlength="16" value="" /> <span
                                id="passwordError" class="errorPassword error1"></span> (영문 대소문자/숫자/특수기호
                            4자~16자)
                        </td>
                    </tr>
                    <tr>
                        <th scope="row">비밀번호 확인 <img
                                src="//img.echosting.cafe24.com/skin/base/common/ico_required.gif"
                                alt="필수" /></th>
                        <td>
                            <form:password id="passwordCheckT" path="passwordCheck" class="eraser"
                                autocomplete="off" minlength="4" maxlength="16" value="" /> <span
                                id="pwCheckError" class="errorPassword error1"></span> <span
                                id="pwCheckSuccess" class="errorPassword"></span> <span
                                id="passwordCheckMsg"></span>
                        </td>
                    </tr>
```



Error Message output

필수입력사항 입니다 (영문소문자/숫자, 4~16자) 중복확인

필수입력사항 입니다 (영문 대소문자/숫자/특수기호 4자~16자)

비밀번호가 일치합니다

Controller

```java
@PostMapping("memberJoin")
public ModelAndView setJoin(@Valid MemberVO memberVO, Errors errors) throws Exception {
    ModelAndView mv = new ModelAndView();
    int result = memberService.setJoin(memberVO);
    if(errors.hasErrors()) {
        System.out.println("================================");
        System.out.println("errors : "+errors);
        System.out.println("================================");
    }

    mv.addObject("result", result);
    mv.setViewName("common/ajaxResult");
    return mv;
}
```

VO

```java
@Data
public class MemberVO implements UserDetails{

    @NotEmpty
    @Length(min = 4, max = 16)
    private String username;
    @NotEmpty
    @Length(min = 4, max = 16)
    private String password;
    private String passwordCheck;
    @NotEmpty
    private String name;
    @Length(min = 3)
    private String phone;

    private String phone0;
    private String phone1;
    private String phone2;

    @Email
    private String email;
    private String birth;
    private boolean smsAgree;
    private boolean emailAgree;
    private boolean enabled;
```

- Validation API를 활용한 form 검증
- Front에서 1차 검증 후 Back-End에서 2차적 검증 실행

MEMBER JOIN

```java
@Transactional(rollbackFor = Exception.class)
public int setJoin(MemberVO memberVO) throws Exception {
    // 1. password 암호화
    memberVO.setPassword(passwordEncoder.encode(memberVO.getPassword()));
    // 2. 사용자 활성화
    memberVO.setEnabled(true);
    // 3. member table 저장
    int result = memberMapper.setJoin(memberVO);
    // 4. memberRole table 저장
    Map<String, String> map = new HashMap<String, String>();
    map.put("username", memberVO.getUsername());
    map.put("roleName", "ROLE_MEMBER");
    result = memberMapper.setMemberRole(map);

    // 5. address table 저장
    AddressVO addressVO = new AddressVO();
    addressVO.setUsername(memberVO.getUsername());
    addressVO.setRecipient(memberVO.getName());
    addressVO.setAddrPhone(memberVO.getPhone());
    addressVO.setZipCode(memberVO.getZipCode());
    addressVO.setBasicAddr(memberVO.getBasicAddr());
    addressVO.setDetailAddr(memberVO.getDetailAddr());
    result = addressService.setJoinAddress(addressVO);

    // 6. mileage table 저장
    MileageVO mileageVO = new MileageVO();
    mileageVO.setUsedMile(0L);
    mileageVO.setChangeMile(2000L);
    mileageVO.setUsername(memberVO.getUsername());
    mileageVO.setMileContents("신규회원 적립금");
    mileageVO.setEnabledMile(2000L);
    result = (int)mileageService.setMileAfterJoin(mileageVO).longValue();

    // 7. coupon table 저장
    Date sdate = new Date(Calendar.getInstance().getTimeInMillis());
    Date edate = new Date(Calendar.getInstance().getTimeInMillis()+2592000000L);
    CouponVO couponVO = new CouponVO();
    List<CouponVO> ar = new ArrayList<CouponVO>();
    couponVO.setCuSpNum(9L);
    couponVO.setUsername(memberVO.getUsername());
    couponVO.setPubDate(sdate);
    couponVO.setExDate(edate);
    ar.add(couponVO);
    couponMapper.couponForSelected(ar);

    if(result<1) {
        throw new Exception();
    }

    return result;
}
```

```xml
<insert id="setJoin" parameterType="MemberVO">
    insert into member(username, password, name, phone, email, birth, smsAgree, emailAgree, enabled)
    values(#{username}, #{password}, #{name}, #{phone}, #{email}, #{birth}, #{smsAgree}, #{emailAgree}, #{enabled})
</insert>
```

```xml
<insert id="setMemberRole" parameterType="java.util.Map">
    insert into memberRole(username,id)
    values(
        #{username}, (
            select id from role where roleName=#{roleName}
        )
    )
</insert>
```

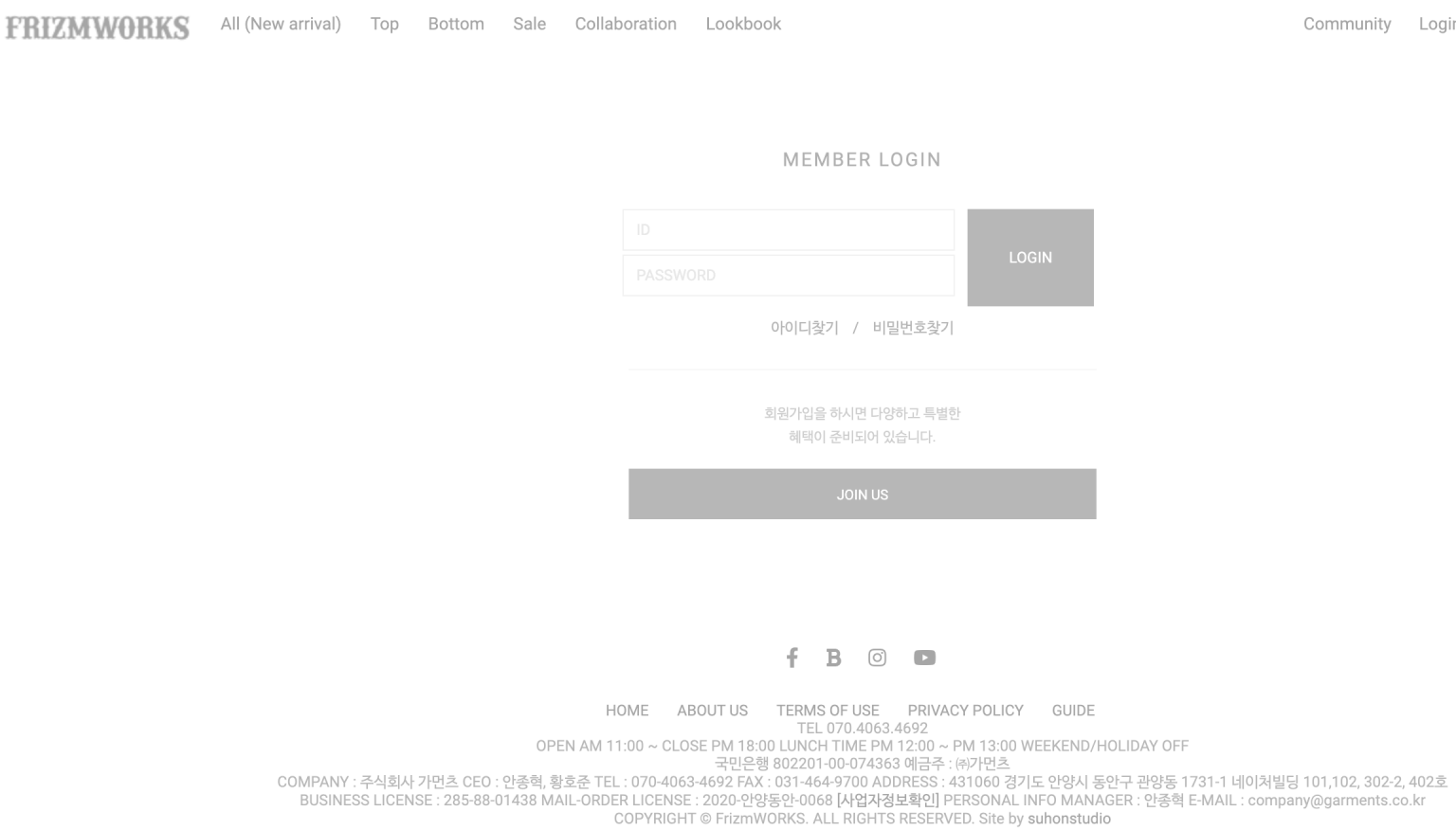
```xml
<insert id="setJoinAddress" parameterType="AddressVO">
  insert into address(addrNum, username, addrName, recipient, addrPhone, zipCode, basicAddr, detailAddr, orderAddr, recentUse)
  values(0, #{username}, "미지정", #{recipient}, #{addrPhone}, #{zipCode}, #{basicAddr}, #{detailAddr}, false, false)
</insert>
```

```xml
<insert id="setMileAfterJoin" parameterType="MileageVO">
    INSERT INTO mileage(mileNum, usedMile, changeMile, username, orderNum, mileContents, enabledMile)
    VALUES(0, #{usedMile}, #{changeMile}, #{username}, null, #{mileContents}, #{enabledMile})
</insert>
```

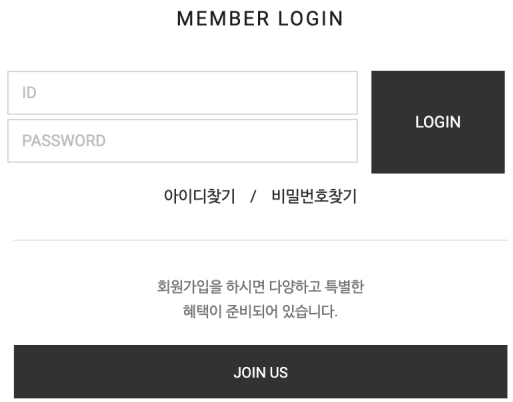
```xml
<insert id="couponForSelected" parameterType="java.util.List">
    insert into coupon(username,cuSpNum, pubDate, exDate, useCheck) values
    <foreach collection="list" item="item" separator=",">
        (#{item.username}, #{item.cuSpNum}, #{item.pubDate}, #{item.exDate}, 0)
    </foreach>
</insert>
```

- Service에서 해당 DB Table로 데이터 저장
- @Transactional 을 통해 query문 실행 중 하나라도 실패시 rollback 처리

MEMBER LOGIN



MEMBER LOGIN



WebSecurityConfig

```java
    @Override
    protected void configure(HttpSecurity http) throws Exception {
        // URL에 따른 로그인, 권한 설정
        http
            .cors().and()
            .csrf().disable()
            .authorizeRequests()
//              .antMatchers("/**").permitAll()
                .antMatchers("/").permitAll()
                // --- community START ---
                .antMatchers("/notice/list", "/qna/list","/review/list").permitAll()
                .antMatchers("/notice/select", "/qna/select","/review/select").permitAll()
                .antMatchers("/lookbook/list", "/lookbook/select","/lookbook/modal","/stockist").permitAll()
                .antMatchers("/notice/**").hasAnyRole("MEMBER","ADMIN")
                // --- community END ---
                // --- product START ---
                .antMatchers("/product/**").permitAll()
//              .antMatchers("/product/list").permitAll()
//              .antMatchers("/product/select").permitAll()
//              .antMatchers("/product/**").hasRole("ADMIN")
                // --- product END ---
                // --- address START ---
                .antMatchers("/address/**").hasAnyRole("MEMBER","ADMIN")
                // --- address END ---
                // --- cart END ---
                .antMatchers("/cart/counting").permitAll()
                .antMatchers("/cart/**").hasAnyRole("MEMBER","ADMIN")
                // --- cart END ---
```

Controller

```java
@GetMapping("memberLoginSuccess")
public ModelAndView loginSuccess(HttpSession session) throws Exception {
    ModelAndView mv = new ModelAndView();
    // 1. session의 속성명들 꺼내오기
    Enumeration<String> enumeration = session.getAttributeNames();

    mv.setViewName("index");
    return mv;
}
```

로그인 성공시 index 페이지로

```java
@GetMapping("memberLoginFail")
public ModelAndView loginFail() throws Exception {
    ModelAndView mv = new ModelAndView();
    mv.addObject("msg", "다시 로그인해주세요");
    mv.addObject("path", "/member/memberLogin");
    mv.setViewName("common/commonResult");
    return mv;
}
```

로그인 실패시 alert창
-> 다시 Login form으로

Service

```java
@Override
public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {
    MemberVO memberVO = new MemberVO();
    memberVO.setUsername(username);
    try {
        memberVO = memberMapper.getLogin(memberVO);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return memberVO;
}
```

Mapper

```xml
<select id="getLogin" parameterType="MemberVO" resultMap="result">
    select M.*, R.* from member M
    inner join memberRole MR
    on M.username = MR.username
    inner join role R
    on MR.id = R.id
    where M.username = #{username}
</select>
<resultMap type="MemberVO" id="result">
    <id column="username" property="username"/>
    <result column="password" property="password"/>
    <result column="name" property="name"/>
    <result column="phone" property="phone"/>
    <result column="email" property="email"/>
    <result column="birth" property="birth"/>
    <result column="smsAgree" property="smsAgree"/>
    <result column="emailAgree" property="emailAgree"/>
    <result column="enabled" property="enabled"/>
    <collection property="roles" javaType="java.util.List" ofType="RoleVO">
        <id column="id" property="id"/>
        <result column="roleName" property="roleName"/>
    </collection>
</resultMap>
```

- 로그인시 UserDetailsService Interface implement
- loadUserByUsername 메서드를 통해 로그인 ID parameter 가져옴
- query문 실행시 해당 ID의 ROLE 데이터 조회 -> WebSecurity에서 권한 식별

ID FIND

FIND ID

가입하신 방법에 따라 아이디 찾기가 가능합니다.

◉이메일 ○휴대폰번호

› 이름

› 이메일로 찾기

확인

○이메일 ◉휴대폰번호

› 이름

› 휴대폰 번호로 찾기 010 - -

확인

이메일 or 휴대폰 조회 → Controller로 parameter 전송 → parameter 가공 → email, phone 에 따라 다른 query문 실행 → 조회 결과 view로 전송, 후처리 → 결과 출력

```java
@PostMapping("finder/idFind")
public ModelAndView getMember(MemberVO memberVO) throws Exception {
    ModelAndView mv = new ModelAndView();
    memberVO = memberService.getMember(memberVO);
    String msg = "입력하신 정보로 가입된 회원 아이디는 존재하지 않습니다.";
    String path = "/member/finder/idFind";
    String viewName="common/commonResult";
    if(memberVO != null) {
        viewName = "/member/finder/idFindResult";
    }
    mv.addObject("msg", msg);
    mv.addObject("path", path);
    mv.addObject("VO", memberVO);
    mv.setViewName(viewName);
    return mv;
}
```

```java
public MemberVO getMember(MemberVO memberVO) throws Exception {
    if(memberVO.getEmail().equals("")) {
        memberVO.setPhone(memberVO.getPhone0()+"-"+memberVO.getPhone1()+"-"+memberVO.getPhone2());
    }
    return memberMapper.getMember(memberVO);
}
```

```sql
<select id="getMember" parameterType="MemberVO" resultType="MemberVO">
    <choose>
        <when test="email == ''">
            select username, name, phone from member where name=#{name} and phone=#{phone}
        </when>
        <otherwise>
            select username, name, email from member where name=#{name} and email=#{email}
        </otherwise>
    </choose>
</select>
```

```java
// ready ====================================
$(document).ready(function(){
    $(".encrypted_name").html($(".encrypted_name").attr("title"));
    $(".encrypted_searchInfo").html($(".encrypted_searchInfo").attr("title"));
    let username = $(".encrypted_username").attr("title");
    let parsed = username.substring(0, 3);
    let num = username.length - 3;
    let encrypt = "*";
    encrypt = encrypt.repeat(num);
    username = parsed.concat(encrypt);
    $(".encrypted_username").html(username);

});
```

고객님 아이디 찾기가 완료 되었습니다.

! 저희 쇼핑몰를 이용해주셔서 감사합니다.

이름	: test
이메일	: hyungsoolim@naver.com
ID	: tes*

고객님 즐거운 쇼핑 하세요!

고객님의 아이디 찾기가 성공적으로 이루어졌습니다. 항상 고객님의
즐겁고 편리한 쇼핑을 위해 최선의 노력을 다하는 쇼핑몰이 되도록 하겠습니다.

로그인 | 비밀번호 찾기

PW FIND

FIND PASSWORD

› 회원유형 ◉이메일 ◯휴대폰번호
› 아이디
› 이름
› 이메일로 찾기

확인

› 회원유형 ◯이메일 ◉휴대폰번호
› 아이디
› 이름
› 휴대폰 번호로 찾기 010 - -

확인

FIND PASSWORD

임시 비밀번호 전송

› 임시 비밀번호 ◉ 이메일
› 이메일 hyungsoolim@naver.com

임시 비밀번호 전송 취소

ID FIND 동일과정…

가입된 회원인지 조회 후
임시 비밀번호 전송

```java
@Transactional(rollbackFor = Exception.class)
public long sendTempPw(MemberVO memberVO) throws Exception {
    // 1. 임시비밀번호 생성
    Random random = new Random();
    String tempPw = random.nextInt()+"";
    System.out.println("Temporary Password : "+tempPw);
    // 2. 임시비밀번호 인코딩
    memberVO.setPassword(passwordEncoder.encode(tempPw));
    // 3. DB에 임시비밀번호로 바꾸기
    long result = memberMapper.setPw(memberVO);
    System.out.println("SERVICE ---------> Password Change Complete");
    // 4. Mail로 임시비밀번호 보내기
    result = mailer.sendMail(memberVO, tempPw);

    return result;
}
```

Random Class로 임시비밀번호 생성

임시 비밀번호 암호화

DB 해당 column 임시비밀번호로 update

MailSender로 임시비밀번호 전송

```java
@Component
public class Mailer {

    @Autowired
    JavaMailSender javaMailSender;

    // JavaMailSender configuration
    @Bean
    public JavaMailSender getJavaMailSender() {
        JavaMailSenderImpl mailSender = new JavaMailSenderImpl();
        mailSender.setHost("smtp.gmail.com");
        mailSender.setPort(587);

        // 자신의 gmail 주소 넣으면 됨
        mailSender.setUsername("이메일 주소");
        // PW 또는 생성한 앱 비밀번호
        mailSender.setPassword("비밀번호 / 앱 비밀번호");

        Properties props = mailSender.getJavaMailProperties();
        props.put("mail.transport.protocol", "smtp");
        props.put("mail.smtp.auth", "true");
        props.put("mail.smtp.starttls.enable", "true");
        props.put("mail.debug", "true");

        return mailSender;
    }

    public long sendMail(MemberVO memberVO, String tempPw) throws Exception {
        long result = 0L;
        if(memberVO != null) {
            MimeMessage mimeMessage = javaMailSender.createMimeMessage();
            mimeMessage.setSubject("[FrizmWORKS]"+memberVO.getName()+"임시 비밀번호가 생성되었습니다.");
            mimeMessage.setRecipients(Message.RecipientType.TO, memberVO.getEmail());
            mimeMessage.setText("ID : "+memberVO.getUsername()+"\r"+"임시 비밀번호 : "+tempPw);
            mimeMessage.setSentDate(new Date());
            javaMailSender.send(mimeMessage);
            result = 1L;
        }

        return result;
    }

}
```

보낼 이메일 주소

비밀번호 / 임시비밀번호

메일 제목, 내용, 수신자, 날짜 설정

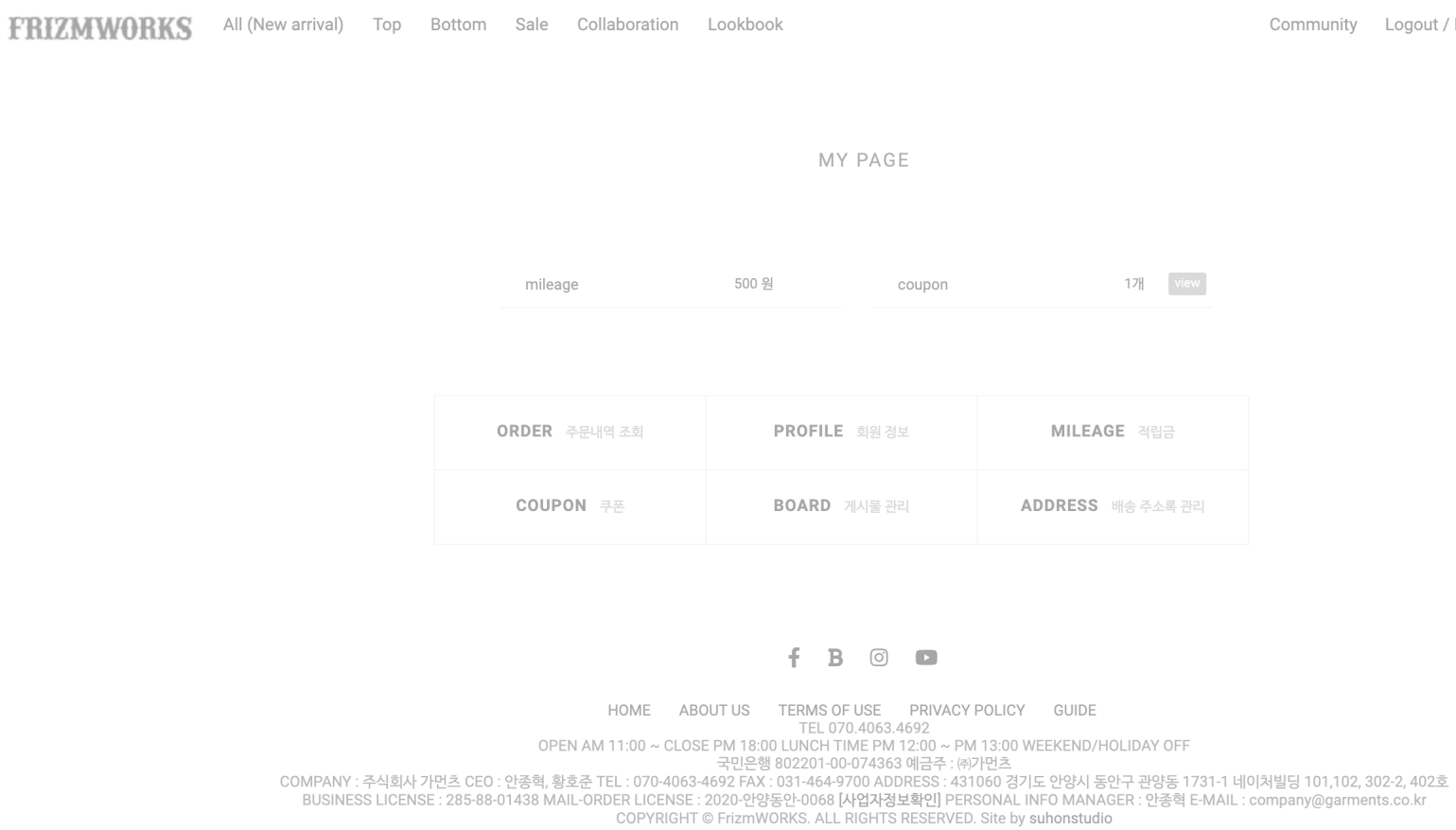
메일 전송

☆ [FrizmWORKS]test임시 비밀번호가 생성되었습니다.

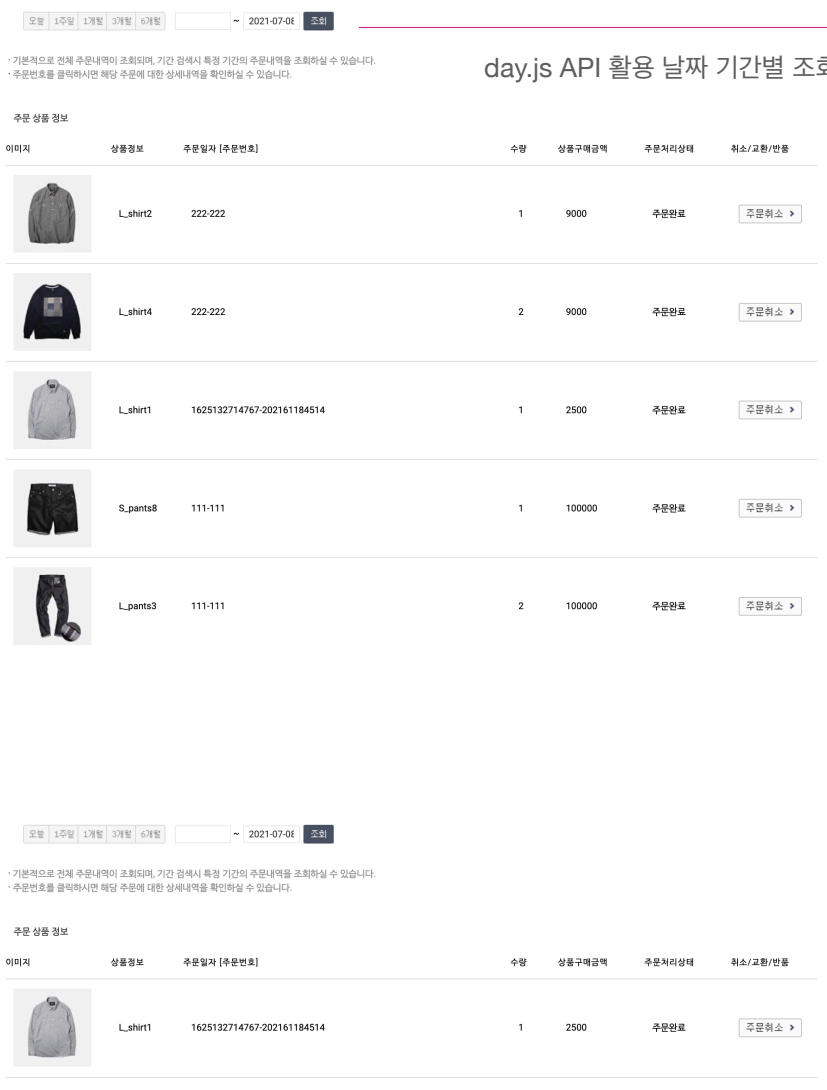
보낸사람 VIP <hyungsoolim0@gmail.com>
받는사람 <hyungsoolim@naver.com>

ID : test
임시 비밀번호 : -2054431774

MEMBER PAGE


FRIZMWORKS
All (New arrival)
Top
Bottom
Sale
Collaboration
Lookbook
Community
Logout /
MY PAGE
mileage
500 원
coupon
1개
view
ORDER 주문내역 조회
PROFILE 회원 정보
MILEAGE 적립금
COUPON 쿠폰
BOARD 게시물 관리
ADDRESS 배송 주소록 관리
HOME
ABOUT US
TERMS OF USE
PRIVACY POLICY
GUIDE
TEL 070.4063.4692
OPEN AM 11:00 ~ CLOSE PM 18:00 LUNCH TIME PM 12:00 ~ PM 13:00 WEEKEND/HOLIDAY OFF
국민은행 802201-00-074363 예금주 : ㈜가먼츠
COMPANY : 주식회사 가먼츠 CEO : 안종혁, 황호준 TEL : 070-4063-4692 FAX : 031-464-9700 ADDRESS : 431060 경기도 안양시 동안구 관양동 1731-1 네이처빌딩 101,102, 302-2, 402호
BUSINESS LICENSE : 285-88-01438 MAIL-ORDER LICENSE : 2020-안양동안-0068 [사업자정보확인] PERSONAL INFO MANAGER : 안종혁 E-MAIL : company@garments.co.kr
COPYRIGHT © FrizmWORKS. ALL RIGHTS RESERVED. Site by suhonstudio

ORDER LIST



day.js API 활용 날짜 기간별 조회

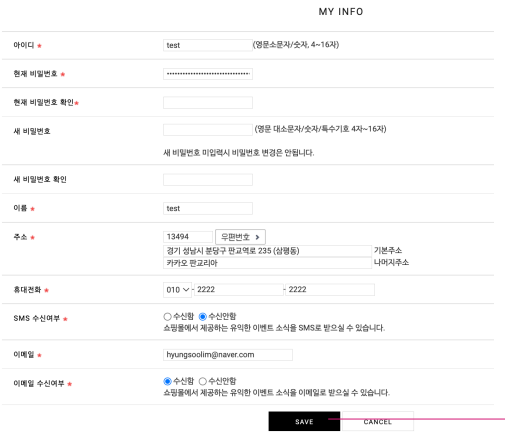
```
let today = dayjs().format("YYYY-MM-DD");

// document.ready ==========================================
// 페이지 로딩 시 기간 default 3개월 전
$(document).ready(function() {
    $("#history_end_date").val(today);
});

// 날짜 검색 버튼 ==========================================
// 오늘 버튼
$("#btn01").click(function() {
    $("#history_start_date").val(today);
});

// 1주일 버튼
$("#btn07").click(function() {
    let targetDate = dayjs().subtract(7, "day").format("YYYY-MM-DD");
    $("#history_start_date").val(targetDate);
});

// 1개월 버튼
$("#btn30").click(function(){
    let targetDate = dayjs().subtract(1, "month").format("YYYY-MM-DD");
    $("#history_start_date").val(targetDate);
});

// 3개월 버튼
$("#btn90").click(function(){
    let targetDate = dayjs().subtract(3, "month").format("YYYY-MM-DD");
    $("#history_start_date").val(targetDate);
});

// 6개월 버튼
$("#btn180").click(function(){
    let targetDate = dayjs().subtract(6, "month").format("YYYY-MM-DD");
    $("#history_start_date").val(targetDate);
});
```

- List 조회시 orderlist, purchase, product, productFiles Table join과 subquery 이용하여 필요한 데이터 조회

```
<resultMap type="ProductFileVO" id="getOrderProduct">
    <id column="fileNum" property="fileNum"/>
    <result column="productNum" property="productNum"/>
    <result column="fileName" property="fileName"/>
    <result column="oriName" property="oriName"/>
    <association property="productVO">
        <id column="productNum" property="productNum"/>
        <result column="productTitle" property="productTitle"/>
    </association>
    <association property="purchaseVO">
        <id column="purNum" property="purNum"/>
        <result column="productNum" property="productNum"/>
        <result column="pInfoNum" property="pInfoNum"/>
        <result column="orderNum" property="orderNum"/>
        <result column="productCount" property="productCount"/>
        <result column="proPriceSum" property="proPriceSum"/>
    </association>
    <association property="orderlistVO">
        <id column="orderNum" property="orderNum"/>
        <result column="totPrice" property="totPrice"/>
        <result column="spPrice" property="spPrice"/>
        <result column="cuNum" property="cuNum"/>
        <result column="username" property="username"/>
        <result column="destination" property="destination"/>
        <result column="orderDate" property="orderDate"/>
        <result column="orderState" property="orderState"/>
        <result column="orderMessage" property="orderMessage"/>
        <result column="orderName" property="orderName"/>
    </association>
</resultMap>
<select id="getOrderList" parameterType="MemberPager" resultMap="getOrderProduct">
    SELECT O.*, PU.*, P.productTitle, PF.fileName FROM orderlist O
    INNER JOIN purchase PU ON O.orderNum = PU.orderNum
    INNER JOIN product P ON PU.productNum = P.productNum
    LEFT JOIN (
        SELECT * FROM productfiles
        WHERE fileName LIKE CONCAT('T\_', '%')
    ) PF ON P.productNum = PF.productNum
    WHERE O.username = #{username}
    ORDER BY O.orderNum DESC
</select>
<select id="getFilteredList" parameterType="OrderlistVO" resultMap="getOrderProduct">
    SELECT O.*, PU.*, P.productTitle, PF.fileName FROM orderlist O
    INNER JOIN purchase PU ON O.orderNum = PU.orderNum
    INNER JOIN product P ON PU.productNum = P.productNum
    LEFT JOIN (
        SELECT * FROM productfiles
        WHERE fileName LIKE CONCAT('T\_', '%')
    ) PF ON P.productNum = PF.productNum
    WHERE O.username = #{username} and O.orderDate between #{startDate} and #{endDate}
    ORDER BY O.orderNum DESC
</select>
```

MY INFO

day.js API 활용 날짜 기간별 조회



- 회원정보 수정시, 현재 비밀번호와 비밀번호 확인 일치 여부 조회
- 불일치 일때 회원정보 수정 불가
- 새 비밀번호 미입력시, 기존 비밀번호 유지

```java
// 회원정보 수정 ==============================================
$("#updateBtn").click(function(){
    username = $("#usernameT").val();
    let oldPWCheck = $("#passwordCheckT").val();
    $.ajax({
        method:"POST",
        url:"./pwCheck",
        data:{
            username:username,
            passwordCheck:oldPWCheck
        },
        success:function(result){
            result = result.trim();
            console.log(result);
            if(result == "true"){
                username = $("#usernameT").val();
                username = username.toLowerCase();
                newPassword = $("#newPasswordT").val();
                if(newPassword == ""){
                    password = oldPWCheck;
                } else {
                    password = newPassword;
                }
                passwordCheck = $("#newPasswordCheckT").val();
                zipCode = $("#zipCodeT").val();
                basicAddr = $("#basicAddrT").val();
                detailAddr = $("#detailAddrT").val();
                phone0 = $("#phone0").val();
                phone1 = $("#phone1").val();
                phone2 = $("#phone2").val();
                phone = phone0+"-"+phone1+"-"+phone2;
                email = $("#emailT").val();
                smsAgree = $("#is_sms0").prop("checked");
                emailAgree = $("#is_news_mail0").prop("checked");
                $.ajax({
                    method:"POST",
                    url:"./memberUpdate",
                    data:{
                        username:username,
                        password:password,
                        phone:phone,
                        email:email,
                        smsAgree:smsAgree,
                        emailAgree:emailAgree,
                        zipCode:zipCode,
                        basicAddr:basicAddr,
                        detailAddr:detailAddr
                    },
                    success:function(result){
                        result = result.trim();
                        if(result != 0){
                            alert("회원정보 수정 성공");
                            location.href="/"
                        }
                    }
                }); // --- memberUpdate ajax END ---/
            }else {
                alert("비밀번호를 확인해주세요.");
            }
        } // --- pwCheck success END ---
    }); // --- pwCheck ajax END ---
```

MY INFO 페이지 접근시
로그인한 user 정보 출력

```java
// profile ==============================================
    @GetMapping("memberPage/memberProfile")
    public ModelAndView getMemberProfile(Authentication authentication) throws Exception {
        ModelAndView mv = new ModelAndView();
        MemberVO memberVO = new MemberVO();
        memberVO.setUsername(authentication.getName());
        memberVO = memberService.getMemberProfile(memberVO);
        AddressVO addressVO = memberService.getProfileAddress(memberVO);
        mv.addObject("VO", memberVO);
        mv.addObject("addressVO", addressVO);
        mv.setViewName("member/memberPage/memberProfile");
        return mv;
    }
```

기존 PW 검증

```java
    @PostMapping("memberPage/pwCheck")
    public ModelAndView getPwCheck(MemberVO memberVO) throws Exception {
        ModelAndView mv = new ModelAndView();
        boolean result = memberService.getPwCheck(memberVO);
        mv.addObject("result", result);
        mv.setViewName("common/ajaxResult");
        return mv;
    }
```

수정한 데이터
DB에 update

```java
    @PostMapping("memberPage/memberUpdate")
    public ModelAndView setMemberUpdate(MemberVO memberVO) throws Exception {
        ModelAndView mv = new ModelAndView();
        long result = 0L;
        result = memberService.setMemberUpdate(memberVO);
        mv.addObject("result", result);
        mv.setViewName("common/ajaxResult");
        return mv;
    }
```

1. 패스워드 암호화
2. member table update
3. address table update

```java
@Transactional(rollbackFor = Exception.class)
public Long setMemberUpdate(MemberVO memberVO) throws Exception {
    long result = 0L;
    // 1. password 암호화
    memberVO.setPassword(passwordEncoder.encode(memberVO.getPassword()));
    // 2. member Update
    result = memberMapper.setMemberUpdate(memberVO);
    // 3. address Update
    result = addressMapper.setProfileAddressUpdate(memberVO);

    return result;
}
```

MILEAGE & COUPON

Mileage

```java
// mileage ==============================================
    @GetMapping("memberPage/memberMileage")
    public ModelAndView getMemberMileage(Authentication authentication) throws Exception {
        ModelAndView mv = new ModelAndView();
        MemberVO memberVO = new MemberVO();
        memberVO.setUsername(authentication.getName());
        List<MileageVO> ar = memberService.getMemberMileage(memberVO);
        for(MileageVO VO:ar) {
            System.out.println(VO);
        }
        MileageVO mileageVO = memberService.getRecentMemberMileage(memberVO);
        mv.addObject("list", ar);
        mv.addObject("mileageVO", mileageVO);
        mv.setViewName("member/memberPage/memberMileage");
        return mv;
    }
```

```xml
<select id="getMemberMileage" parameterType="MemberVO" resultMap="memberMileage">
    select M.*, O.orderDate from mileage M
    left join orderlist O
    on M.orderNum = O.orderNum
    where M.username = #{username} order by M.mileNum desc
</select>
<resultMap type="MileageVO" id="memberMileage">
    <id column="mileNum" property="mileNum"/>
    <result column="usedMile" property="usedMile"/>
    <result column="changeMile" property="changeMile"/>
    <result column="username" property="username"/>
    <result column="orderNum" property="orderNum"/>
    <result column="mileContents" property="mileContents"/>
    <result column="enabledMile" property="enabledMile"/>
    <association property="orderlistVO" javaType="OrderlistVO" >
        <result column="orderDate" property="orderDate"/>
    </association>
</resultMap>
```

상품 주문, 마일리지 소비시, 해당 주문번호에 해당하는 마일리지 소비내역 조회

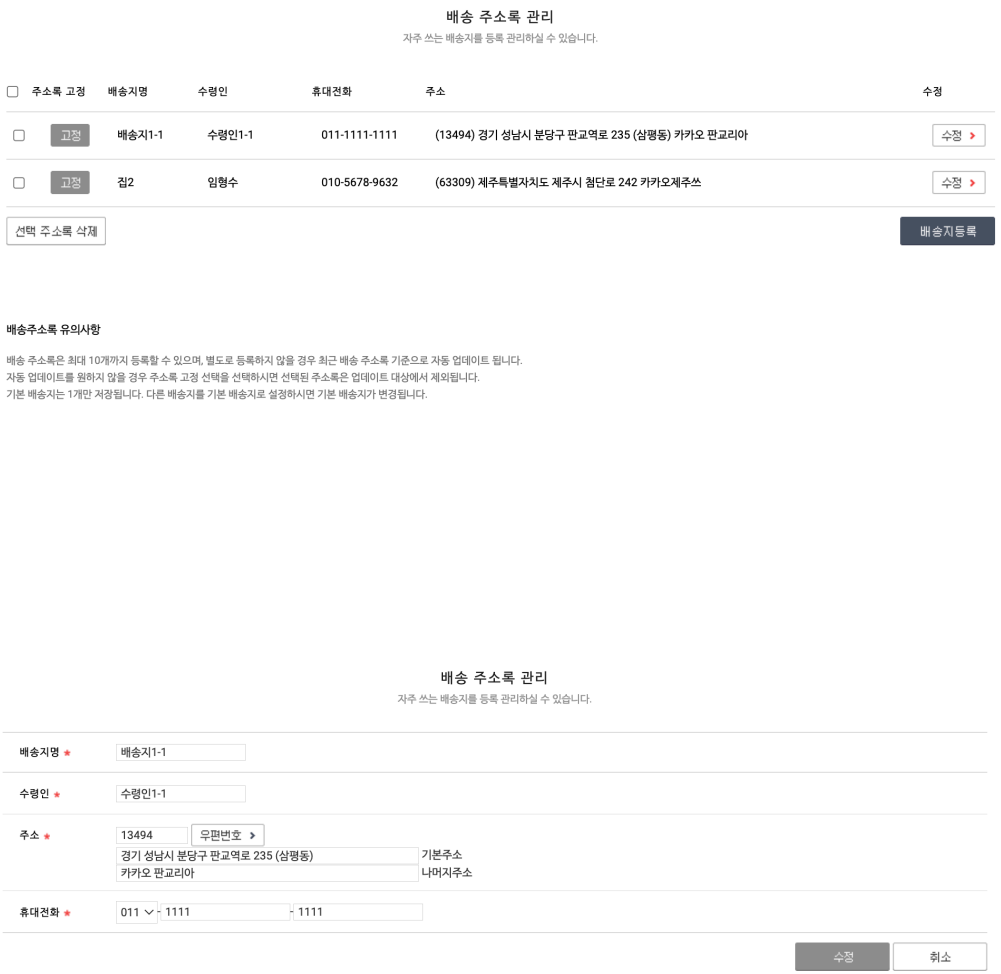
Coupon

```java
// coupon ==============================================
    @GetMapping("memberPage/memberCoupon")
    public ModelAndView getMemberCoupon(Authentication authentication) throws Exception {
        ModelAndView mv = new ModelAndView();
        MemberVO memberVO = new MemberVO();
        memberVO.setUsername(authentication.getName());
        List<CouponVO> ar = couponService.getCouponList(memberVO);

        long couponCount = couponService.getMemberCouponCount(memberVO);
        mv.addObject("list", ar);
        mv.addObject("couponCount", couponCount);
        mv.setViewName("member/memberPage/memberCoupon");
        return mv;
    }
```

```xml
<resultMap type="CouponVO" id="getCouponDetails">
    <id column="cuNum" property="cuNum"/>
    <result column="username" property="username"/>
    <result column="cuSpNum" property="cuSpNum"/>
    <result column="pubDate" property="pubDate"/>
    <result column="exDate" property="exDate"/>
    <result column="useCheck" property="useCheck"/>
    <association property="couponspVO" javaType="CouponspVO">
        <id column="cuSpNum" property="cuSpNum"/>
        <result column="cuName" property="cuName"/>
        <result column="disRate" property="disRate"/>
    </association>
</resultMap>

<select id="getCouponList" parameterType="MemberVO" resultMap="getCouponDetails">
    select c.*, cs.* from coupon c inner join couponsp cs on c.cuSpNum=cs.cuSpNum
    where c.username=#{username} and useCheck=0
</select>
```

coupon과 coupon 상세정보 join으로 조회

ADDRESS




```
<select id="getMemberAddress" parameterType="MemberVO" resultType="AddressVO">
  select * from address where username = #{username}
</select>
```

주소 조회

```
<insert id="setMemberAddress" parameterType="AddressVO">
  insert into address(username, addrName, recipient, zipCode, basicAddr, detailAddr, addrPhone, recentUse, orderAddr)
  values (#{username}, #{addrName}, #{recipient}, #{zipCode}, #{basicAddr}, #{detailAddr}, #{addrPhone}, 0, 0)
</insert>
```

주소 추가

```
<update id="updateMemberAddress" parameterType="AddressVO">
  update address set addrName=#{addrName}, recipient=#{recipient}, addrPhone=#{addrPhone},
  zipCode=#{zipCode}, basicAddr=#{basicAddr}, detailAddr=#{detailAddr}, recentUse=0
  where username=#{username} and addrNum=#{addrNum}
</update>
```

주소 변경

```
<delete id="deleteMemberAddress" parameterType="AddressVO">
    delete from address where username=#{username} and addrNum=#{addrNum}
</delete>
```

주소 삭제

```
// 주소 삭제 ==========================================
let ar = [];

$("#deleteAddr").click(function(){
    let confirmCheck = confirm("삭제하겠습니까?");
    if(confirmCheck) {
        $(".check").each(function(){
            let selected = $(this).prop("checked");
            if(selected){
                ar.push($(this).attr("title"));
            }
        });

        $.ajax({
            method:"POST",
            url:"./addressDelete",
            traditional:true,
            data:{ar:ar},
            success:function(result){
                result=result.trim();
                if(result>0){
                    alert("삭제 완료");
                    location.reload();
                }
            },
            error:function(result){
                alert("AJAX error : "+result);
            }
        });
    }

});
```

주소 여러개 삭제시 배열을 통해 Controller로 전송 후 처리

THANK YOU