

# Portfolio.

---



# 목차

a table of contents

1 기획 의도

2 기본 개발 환경

3 요구사항 정의서

4 UI 설계

5 DB 설계

6 기술 상세



Part 1  
기획 의도

## 기획 의도

독립 서점의 수요가 전국적으로 증가하는 추세와 더불어 MZ세대의 ‘북캉스’라는 새로운 문화를 반영하여, 이를 웹사이트로 만들어서 오프라인 독립 서점의 제한적인 접근성을 보완하기 위해 기획

2022년 기준으로 운영 중인 전국 독립서점은 모두 815곳으로 나타났다. 전년보다 70곳 늘어난 것인데, 한 해 동안 일주일에 13곳씩 생겨난 셈이다.

동네책방 관련 정보를 제공하는 주식회사 ‘동네서점’은 30일 발행한 ‘2022년 동네서점 트렌드’ 통계분석 보고서에서 이렇게 밝혔다. 2016년부터 이 업체가 운영하는 ‘동네서점지도’(bookshopmap.com)를 중심으로 자료를 길게해 해마다 발행해오고 있는 보고서다.

### 전국 독립서점 증강추세

#### Nationwide Bookshops Trend

구분	운영 중 독립서점 수 (곳)	누적 휴폐점 수 (곳)	누적 등록 수 (곳)	누적 등록 대비 폐점 비율 (%)	지난해 대비 증감 수 (곳)
2015년	97	4	101	4.0	▲97
2016년	180	6	186	3.2	▲83
2017년	283	25	308	8.1	▲103
2018년	416	50	466	10.7	▲133
2019년	551	99	650	15.2	▲135
2020년	634	127	761	16.7	▲83
2021년	745	181	926	19.5	▲111
2022년	815	216	1,031	21.0	▲70

출처 : <https://www.hani.co.kr/arti/culture/book/1077473.html>

이런 가운데 새로운 물결이 있다. 2010년대 중반을 거쳐 탄생한 새로운 형태의 서점들 이야기다. 2014년 11월 시행한 도서정가제를 계기로 탄생한 독립서점이 그것. 2015년부터 독립서점 현황을 집계해온 주식회사 ‘동네서점’에 따르면, 2022년 기준으로 전국에서 운영 중인 독립서점은 815곳이나 된다. 2015년에 97곳밖에 되지 않았던 독립서점이 8년 만에 무려 8배 넘게 증가한 것이다. 어떤 사람들은 여행 코스의 하나로 그 지역의 독립서점을 방문하기도 한다. ‘서점의 반전’은 어떻게 가능했을까.

출처 : <https://www.hani.co.kr/arti/culture/book/1077473.html>

독립 서점의 운영이 전국적으로 늘어나는 추세

서점의 존폐 위기를 논하던 중, 독립 서점이라는 새로운 물결

독립서점을 운영하는 이들에게도 매장 이용객들 중 젊은 세대가 흔하다는 공통된 의견이 나온다. 제주도 제주시에서 독립서점을 운영하는 A씨는 “순수하게 읽을 거리를 찾아 나서는 젊은 손님들이 많이 찾아 온다”며 “전체 고객 중 40% 이상이 2030세대”라고 설명했다. 경남 거제시 한 독립서점에서 ‘책방지기’로 일하는 B씨도 “일부러 휴가를 맞아 서울에서 이곳까지 찾아오는 단골도 여럿 있다”고 말했다.

MZ세대는 독립서점을 찾는 이유로 ‘책 큐레이팅(우수한 책 소개)’을 꼽았다. 대형서점이나 온라인서점에선 판매 부수나 대중의 취향을 반영해 책을 진열하지만 독립서점은 점주가 자신만의 기준에 따라 책을 비치한다. 일반 온오프라인 서점이 줄 수 없는 개성 있는 책 경험을 독립서점에서 누릴 수 있다는 것이다.

방학마다 독립서점이 있는 곳으로 여행을 떠난다는 대학생 신민철(21) 씨는 “독립서점마다 책 방지기의 큐레이팅이 달리서 나오는 고유의 분위기가 있다”며 “새로운 곳에 갈 때마다 ‘여기엔 어떤 책들이 있을까’ 기대된다”고 말했다.

출처 : <https://news.heraldcorp.com/view.php?ud=20230814000696>

MZ세대의 새로운 휴가,  
‘북캉스’



# 개발 환경

- Java 1.8
- Spring
- Apache Tomcat 9.0
- Oracle
- HTML5
- CSS3
- JQuery
- Ajax

Software

- OS: Window10
- CPU: Intel(R) Core(TM) i3-6100H
- RAM: 8GB
- SSD: 250GB

Server  
Computer

- OS: Window10
- CPU: Intel(R) Core(TM) i3-6100H
- RAM: 8GB
- SSD: 250GB

Client  
Computer





# 요구사항 정의서

	이름	내용	우선순위	중요도
로그인	로그인 페이지	메인화면 상단의 Login 기능 클릭 시 로그인 페이지로 이동	필수	상
	로그인 기능	아이디와 비밀번호 기재 후 로그인 버튼 클릭, DB의 회원 정보 확인, 로그인 진행	필수	상
	로그인 확인	아이디나 비밀번호 입력 칸을 비워두거나 아이디는 이메일 양식이 아닌 상태에서 로그인 버튼 누르면 페이지 이동 불가	필수	상
회원가입	회원가입 페이지	메인화면 상단의 Sign Up 기능 클릭 시 회원가입 페이지로 이동	필수	상
	회원가입 기능	이름과 연락처, 아이디, 비밀번호 기재 후 회원가입 버튼 클릭, DB에 회원 정보 저장	필수	상
	회원가입 확인	이름과 연락처, 아이디, 비밀번호 입력 칸을 비워두거나 아이디는 이메일 양식이 아닌 상태에서 회원가입 버튼 누르면 페이지 이동 불가	필수	상
	로그인 이동	기존 회원일 경우 회원가입 페이지에서 하단의 Already-(생략) 링크 클릭하면 로그인 페이지로 이동	선택	중
헤더	로고	로고 클릭 시 메인 페이지로 이동	필수	하
	검색	검색어로 책 제목 입력 후 버튼 클릭 시 해당 내용의 결과물 출력	필수	상
	로그인 여부	로그인 성공 시 사용자의 이름과 인사말, 장바구니, 주문 목록, 로그아웃 버튼 생성	필수	중
	장바구니	장바구니 클릭 시 장바구니 페이지로 이동	필수	상
	주문 목록	주문 목록 클릭 시 주문 목록 페이지로 이동	필수	상

## 요구사항 정의서

	이름	내용	우선순위	중요도
메인	카테고리	국내 도서와 해외 도서 카테고리로 구성	필수	상
	서브 카테고리	국내 도서 – 소설, 시/에세이, 인문 해외 도서 – 문학, 예술/건축, 인문/사회로 하위 카테고리 구성	필수	상
	도서 목록	메인화면의 중앙에 홈페이지에서 등록된 도서 목록을 나열할 공간 구성	필수	상
	북마크 타이핑	메인화면의 하위에 인상깊은 도서 문구를 볼 수 있는 공간 구성	필수	중
	상품 광고	메인화면의 하위에 출간된 신간과 화제작을 홍보하는 배너 배치, 배너를 클릭하면 해당 도서 세부 페이지로 이동	필수	중
제품 상세 페이지	제품 정보	도서 목록에서 도서를 클릭하면 제품 상세 페이지로 이동, 구입에 필요한 제품 관련 정보가 나타남	필수	상
	제품 재고	제품의 구입 가능 재고가 나타남, 고객이 제품 구입 시 DB와 연동해서 재고의 수 자동으로 감소	필수	중
	제품 구입 수량	원하는 제품 구입 수량에 따라 +,-버튼으로 수량 입력, 고객이 구입한 제품의 수량만큼 제품 재고의 수 자동으로 감소	필수	상
	카트 담기	카트 담기 버튼 클릭 시 성공했다는 알림창을 생성	필수	중
	주문하기	장바구니 페이지에서 주문 정보 입력 후 주문 버튼 클릭	필수	상

# 요구사항 정의서

	이름	내용	우선순위	중요도
장바구니	제품 목록	메인화면에서 장바구니를 선택하면 장바구니에 담은 제품 목록 확인	필수	상
	제품 선택	여러 제품을 장바구니에 담았을 시 주문을 원하는 상품만 선택하거나 모두 선택, 모두 선택 해제	필수	중
	제품 삭제	장바구니에서 빼고 싶은 상품에 대해서만 삭제하거나 선택 삭제, 모두 삭제	필수	상
주문하기	주문 정보 입력	장바구니 하단의 주문 정보 입력 버튼 선택 시 주문 상자 발생	필수	상
	수령인 정보 기재	수령인과 수령인 연락처 기재 시 DB에도 정보 저장	필수	상
	수령 주소 기재	다음 주소 API를 이용해서 우편번호 찾기, 수령 주소 입력	필수	상
	주문, 취소, 총 합계	장바구니의 총 합계 금액 기재, 주문 버튼 클릭 시 DB에 주문 정보 저장	필수	상
	주문 목록	주문 정보에 입력된 정보와 각 주문에 생성된 고유 주문 번호로 주문 목록 생성, 주문 목록에서 주문 내역 확인	필수	상
상품 소감	소감 남기기	상품 상세 페이지 하단에 상품 소감 공간 배치	필수	상
	소감 등록	로그인 시에만 각 제품에 대한 소감 등록	필수	상
	소감 수정	소감 수정은 본인의 아이디로만 가능, 등록자 본인일 경우에 모달창으로 소감 수정	필수	상
	권한 확인	DB를 통해 소감 등록자의 아이디와 일치 여부 확인	필수	상
	소감 삭제	소감 삭제는 본인의 아이디로만 가능, 등록자 본인일 경우에 소감 삭제	필수	상

## 요구사항 정의서

	이름	내용	우선순위	중요도
관리자 모드	로그인 기능	관리자 로그인 시 관리자 화면 버튼 생성, 상단 로고와 이미지 변경	필수	상
	관리자 화면	카테고리 구성 변경 상품 등록, 상품 목록, 주문 목록, 상품 소감	필수	상
	상품 등록	상위/하위 카테고리 선택, 상품 명, 상품 가격, 상품 수량, 상품 소개, 상품 이미지 등록	필수	상
	상품 목록	사이트에 등록되어 있는 모든 상품을 목록으로 조회	필수	상
	상품 조회	상품 목록에서 조회를 원하는 상품 선택하면 상품 조회 페이지로 이동	필수	상
	상품 수정, 삭제	상품 조회 페이지의 제일 하단에서 해당 상품 수정과 삭제 가능 수정 버튼 클릭하면 상위/하위 카테고리, 상품 명, 상품 가격, 상품 수량, 상품 소개, 상품 이미지 수정 삭제 버튼 클릭 시 삭제 요청에 대한 확인을 위한 알림창 발생	필수	상
	주문 목록	사용자들의 주문에 대한 고유 주문 번호와 해당 주문 목록을 조회	필수	상
	주문 내역	주문 번호를 클릭 시 해당 주문 목록의 주문 정보와 주문 제품을 조회	필수	상
	배송 상태	모든 주문 내역의 배송 상태는 배송 준비중을 디폴트로 설정, 주문 내역에서 배송 상태를 배송 중 혹은 배송 완료로 변경	필수	상
	상품 소감	모든 상품에 달린 사용자들의 소감을 목록으로 조회, 삭제, 소감이 달린 해당 상품 페이지로 바로가기	필수	상

Part 4  
UI 설계

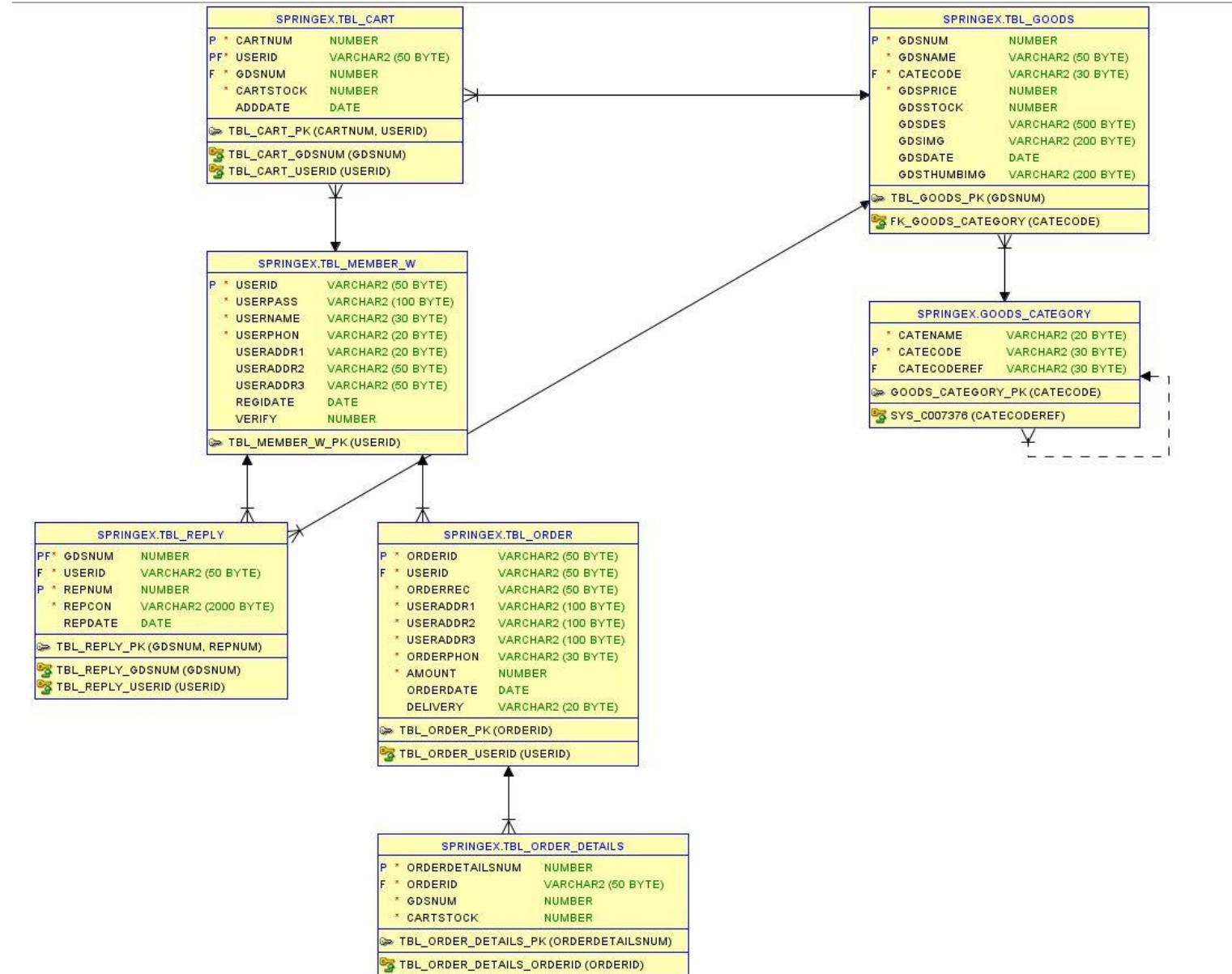
## UI 설계



Part 5  
DB 설계

## DB 설계 - ERD

테이블 명	테이블 설명
tbl_member_w	회원 테이블
tbl_goods	제품 테이블
goods_category	제품 카테고리 테이블
tbl_reply	상품 소감 테이블
tbl_cart	장바구니 테이블
tbl_order	주문 테이블
tbl_order_details	주문 상세 테이블



## tbl\_member\_w 테이블

Field	Data Type	Key	Not Null	Data Default	설명
userId	varchar2(50)	PK	O		회원 아이디
userPass	varchar2(100)		O		회원 비밀번호
userName	varchar2(30)		O		회원 이름
userPhon	varchar2(20)		O		회원 전화번호
userAddr1	varchar2(20)				우편번호
userAddr2	varchar2(50)				회원 주소
userAddr3	varchar2(50)				회원 상세 주소
regiDate	date			Sysdate	회원 등록일
verify	number			0	회원 인증 (시큐리티)

## tbl\_goods 테이블

Field	Data Type	Key	Not Null	Data Default	설명
gdsNum	number	PK	O		상품 번호
gdsName	varchar2(50)		O		상품 이름
cateCode	varchar2(30)		O		카테고리 번호
gdsPrice	number		O		상품 가격
gdsStock	number				상품 수량
gdsDes	varchar(500)				상품 설명
gdslmg	varchar(200)				상품 이미지
gdsDate	date			Sysdate	상품 등록일
gdsThumblmg	varchar(200)				상품 썸네일

## goods\_category 테이블

Field	Data Type	Key	Not Null	설명
cateName	varchar2(20)		O	카테고리 이름
cateCode	varchar2(30)	PK	O	상위 카테고리
cateCodeRef	varchar2(30)	Foreign Key		하위 카테고리

## tbl\_reply 테이블

Field	Data Type	Key	Not Null	Data Default	설명
gdsNum	number	PK	O		상품 번호
userId	varchar2(50)		O		회원 아이디
repNum	number	PK	O		소감 번호
repCon	varchar2(2000)		O		소감 내용
repDate	date			Sysdate	소감 등록일

## tbl\_cart 테이블

Field	Data Type	Key	Not Null	Data Default	설명
cartNum	number	PK	O		장바구니 번호
userId	varchar2(50)	PK	O		회원 아이디
gdsNum	number		O		상품 번호
cartStock	number		O		장바구니 수량
addDate	date			Sysdate	장바구니 등록일

## tbl\_order 테이블

Field	Data Type	Key	Not Null	Data Default	설명
orderId	varchar2(50)	PK	O		주문 고유 번호
userId	varchar2(50)		O		회원 아이디
orderRec	varchar2(50)		O		수신자
userAddr1	varchar2(30)		O		우편번호
userAddr2	varchar2(50)		O		회원 주소
userAddr3	varchar2(50)		O		회원 상세 주소
orderPhon	varchar2(20)		O		주문자 전화번호
amount	number		O		주문 합계 가격
orderDate	Date			Sysdate	주문 날짜
Delivery				‘배송 준비’	배송 상태

## tbl\_order\_details 테이블

Field	Data Type	Key	Not Null	설명
orderDetailsNum	number	PK	O	주문 상세 번호
orderId	varchar2(50)		O	주문 고유 번호
gdsNum	number		O	상품 번호
cartStock	number		O	장바구니 수량

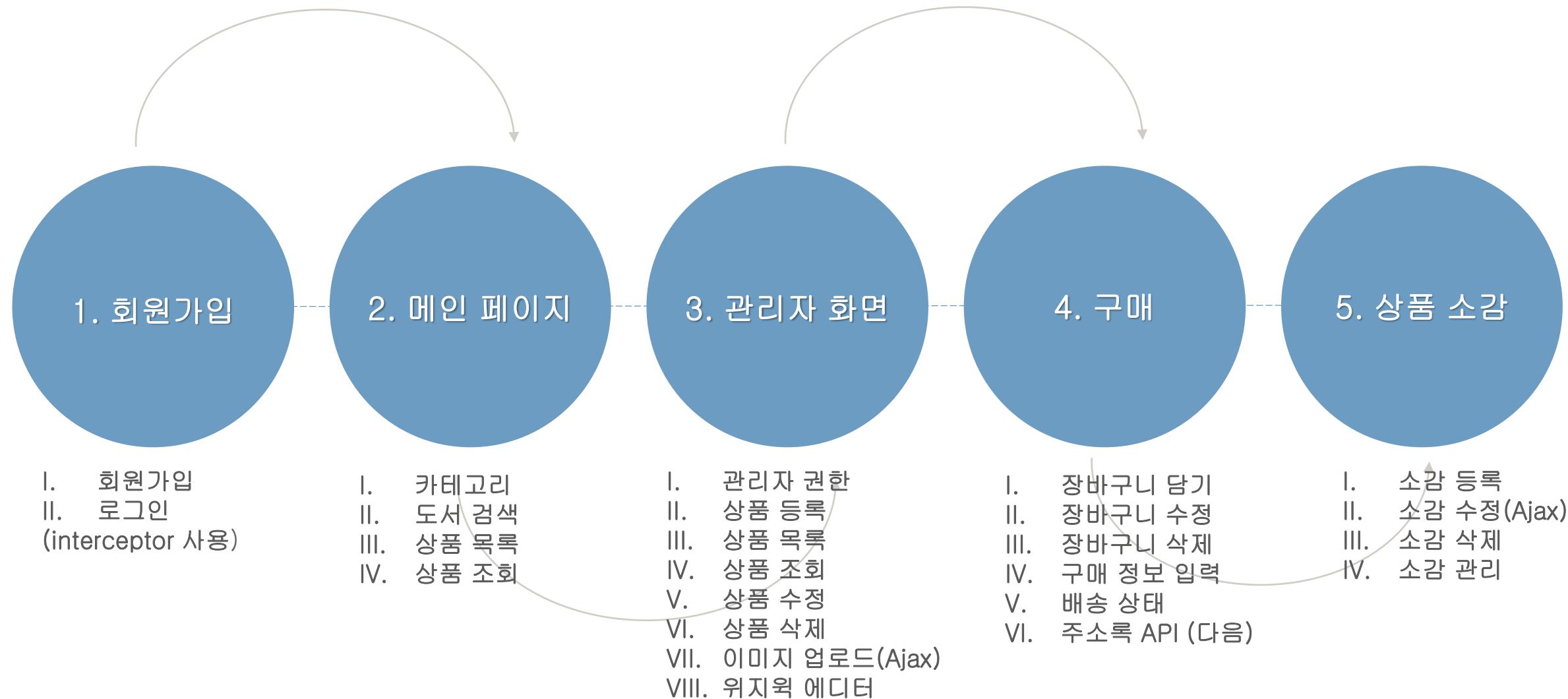




**GitHub 주소**

**<https://github.com/wooqppu/Personal-Project>**

## 기술 상세 - 기술 목록



## 1. 회원가입 - 회원가입

: Mapper -> VO -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 회원가입 -->
<insert id="signup">
    insert into tbl_member_w(userId, userPass, userName, userPhon)
        values(#{userId}, #{userPass}, #{userName}, #{userPhon})
</insert>
```

MemberMapper.xml

```
public interface MemberDAO {
    // 회원가입
    public void signup(MemberVO vo) throws Exception;
}

@Repository
public class MemberDAOImpl implements MemberDAO {
    @Inject
    private SqlSession sql;

    // mapper
    private static String namespace = "org.green.mapper.MemberMapper";

    // 회원가입
    @Override
    public void signup(MemberVO vo) throws Exception {
        sql.insert(namespace + ".signup", vo);
    }
}
```

MemberDAO&amp;Impl

```
@Controller
@RequestMapping("/member/*")
public class MemberController {
    private static final Logger logger = LoggerFactory.getLogger(MemberController.class);

    @Inject
    MemberService service;

    @Autowired
    BCryptPasswordEncoder passEncoder;

    // 회원가입 get
    @RequestMapping(value = "/signup", method = RequestMethod.GET)
    public void getSignup() throws Exception {
        logger.warn("get signup");
    }

    // 회원가입 post
    @RequestMapping(value = "/signup", method = RequestMethod.POST)
    public String postSignup(MemberVO vo) throws Exception {
        logger.warn("post signup");

        String inputPass = vo.getUserPass();
        String pass = passEncoder.encode(inputPass);
        vo.setUserPass(pass);

        service.signup(vo);
        logger.warn("vo : " + vo);
        logger.warn("-----");
        return "redirect:/";
    }
}
```

MemberController

```
private String userAddr3;
private Date regiDate;
private int verify;

public String getUserId() {
    return userId;
}
public void setUserId(String userId) {
    this.userId = userId;
}
public String getUserPass() {
    return userPass;
}
```

MemberVO

```
public interface MemberService {
    // 회원가입
    public void signup(MemberVO vo) throws Exception;
}

@Service
public class MemberServiceImpl implements MemberService {
    @Inject
    private MemberDAO dao;

    private static final Logger logger = LoggerFactory.getLogger(MemberController.class);

    // 회원가입
    @Override
    public void signup(MemberVO vo) throws Exception {
        logger.warn("-----");
        logger.warn("vo : " + vo);
        logger.warn("-----");

        dao.signup(vo);
    }
}
```

MemberService&amp;Impl

```
<div class="text-center">
    <div class="id text-gray-900 mb-4">Create an Account!</div>
</div>

<form class="user" role="form" method="post" autocomplete="off" action="/member/signup">
    <div class="form-group row">
        <div class="col-sm-6 mb-3">
            <input type="text" class="form-control form-control-user" id="userName" name="userName" placeholder="이름을 입력하세요" required="required" />
        </div>
        <div class="col sm 6">
            <input type="text" class="form-control form-control-user" id="userPhon" name="userPhon" placeholder="전화번호를 입력하세요" required="required" />
        </div>
    </div>
    <div class="form-group">
        <input type="email" id="userId" name="userId" class="form-control form-control-user" placeholder="example@email.com" required="required" />
        <input type="password" class="form-control form-control-user" id="userPass" name="userPass" required="required" placeholder="Password" />
    </div>
    <button type="submit" id="signup btn" name="signup btn" class="btn btn-primary btn-block">Register Account</button>
</form>
```

Signup.jsp

## 1. 회원가입 - 로그인

: Mapper -> VO -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 로그인 -->
<select id="signin" resultType="org.green.domain.MemberVO">
    select userId, userName, userPass, userPhon, userAddr1, userAddr2, userAddr3, regiDate, verify
    from tbl_member_w where userId = #{userId}
</select>
```

MemberMapper.xml

```
// 로그인
public MemberVO signin(MemberVO vo) throws Exception;

// 로그인
@Override
public MemberVO signin(MemberVO vo) throws Exception {
    return sql.selectOne(namespace+ ".signin", vo);
}
```

MemberDAO&amp;Impl

```
// 로그인
@RequestMapping(value = "/signin", method = RequestMethod.GET)
public String postsignin(MemberVO vo, HttpServletRequest req, RedirectAttributes rta) throws Exception {
    logger.warn("get signin");
}

// 로그인
@RequestMapping(value = "/signin", method = RequestMethod.POST)
public String postsignin(MemberVO vo, HttpServletRequest req, RedirectAttributes rta) throws Exception {
    logger.warn("post signin");
    MemberVO login = service.signin(vo);
    HttpSession session = req.getSession();
    boolean passMatch = passEncoder.match(vo.getUserPass(), login.getUserPass());
    if(login != null && passMatch) {
        session.setAttribute("member", login);
        rta.addAttribute("msg", "true");
        rta.addFlashAttribute("msg", "false");
        return "redirect:/member/signin";
    }
    return "redirect:/";
}

// 로그아웃
@RequestMapping(value = "/signout", method = RequestMethod.GET)
public String signout(HttpServletRequest session) throws Exception {
    logger.warn("get logout");
    service.signout(session);
    return "redirect:/";
}
```

MemberController

```
private String userAddr3;
private Date regiDate;
private int verify;

public String getUserId() {
    return userId;
}
public void setUserId(String userId) {
    this.userId = userId;
}
public String getUserPass() {
    return userPass;
}
```

MemberVO

```
// 로그인
public MemberVO signin(MemberVO vo) throws Exception;

// 로그인
public MemberVO signin(MemberVO vo) throws Exception;

// 로그아웃
public void signout(HttpServletRequest session) throws Exception;
```

MemberService&amp;Impl

```
<div class="col-lg-6">
    <div class="p-5">
        <div class="text-center">
            <h1 class="h1 text-gray 900 mb-4">Welcome to Wave_Books!</h1>
        </div>
    </div>
</div>

<form class="user" role="form" method="post" autocomplete="off" action="/member/signin">
    <div class="form-group input_area">
        <input type="email" id="userid" name="userid"
               class="form-control form-control-user" aria-describedby="emailHelp" required="required"
               placeholder="Enter Email Address..."/>
    </div>
    <div class="form-group input_area">
        <input type="password" id="userPass" name="userPass"
               class="form-control form-control-user" required="required"
               placeholder="Password"/>
    </div>
    <button type="submit" id="signin btn" name="signin btn"
           class="btn btn-primary btn-user btn-block">Login</button>
</form>
<c:if test="#{msg == false}">
    <p style="color:#f00;">로그인에 실패했습니다.</p>
</c:if>
```

Signin.jsp

## 2. 메인 페이지 - 카테고리

: Mapper -> VO -> pom -> DAO&Impl -> Service&Impl -> JSP

```
<!-- 카테고리 호출 -->
<select id="category" resultType="org.green.domain.CategoryVO">
    select level, cateName, cateCode, cateCodeRef
    from goods_category
    start with cateCodeRef is null connect by prior cateCode = cateCodeRef
</select>
```

AdminMapper.xml

```
private String cateName;
private String cateCode;
private String cateCodeRef;
private int level;

public int getLevel() {
    return level;
}

public void setLevel(int level) {
    this.level = level;
}

public String getCateName() {
    return cateName;
}

public void setCateName(String cateName) {
    this.cateName = cateName;
}

public String getCateCode() {
```

CategoryVO

```
<dependency>
    <groupId>net.sf.ezmorph</groupId>
    <artifactId>ezmorph</artifactId>
    <version>1.0.6</version>
</dependency>
<dependency>
    <groupId>net.sf.json-lib</groupId>
    <artifactId>json-lib</artifactId>
    <version>2.4</version>
    <classifier>jdk15</classifier>
</dependency>
```

pom.xml

```
// 카테고리
public List<CategoryVO> category() throws Exception;

@Repository
public class AdminDAOImpl implements AdminDAO {
    @Inject
    private SqlSession sql;

    // 매퍼
    private static String namespace = "org.green.mapper.adminMapper";

    // 카테고리
    @Override
    public List<CategoryVO> category() throws Exception {
        return sql.selectList(namespace + ".category");
    }
}
```

AdminDAO&amp;Impl

```
// 상품 등록 get
@RequestMapping(value = "/goods/register", method = RequestMethod.GET)
public void getGoodsRegister(Model model) throws Exception {
    logger.warn("get goods register");

    List<CategoryVO> category = null;
    category = adminService.catregory();
    model.addAttribute("category", JSONArray.fromObject(category));
}
```

AdminController

```
// 카테고리
public List<CategoryVO> category() throws Exception;

@Service
public class AdminServiceImpl implements AdminService {
    @Inject
    private AdminDAO dao;

    // 카테고리
    @Override
    public List<CategoryVO> catregory() throws Exception {
        return dao.category();
    }
}
```

AdminService&amp;Impl

```
<form role="form" method="post" autocomplete="off" enctype="multipart/form-data">
    <div class="inputArea">
        <label id="firstCategory">1차분류</label>
        <select class="category1" name="cateCode1">
            <option value="">전체</option>
        </select>

        <label id="secondCategory">2차분류</label>
        <select class="category2" name="cateCode2">
            <option value="">전체</option>
        </select>
    </div>
```

register.jsp

## 2. 메인 페이지 – 도서 검색

: Mapper -> VO -> DAO&Impl -> Service&Impl -> Controller -> JSP

\* 상품 조회를 이용해서 구현했으므로 중복된 과정은 생략, 컨트롤러와 jsp만 기재

```
Logger.warn("get list");

List<GoodsViewVO> list = null;
list = service.list(101, 1);

model.addAttribute("list", list);

return "home";
}
```

HomeController

```
<!-- 검색창 -->



<form name="search-form" autocomplete="off"
      class="d-none d-sm-inline-block form-inline mr-auto ml-md-3 my-2 my-md-0 mr-100 navbar-search">

    <div class="input-group">
      <select name="type" class="form-control bg-light border-0 small" style="width: 60px!important;">
        <option selected value="goodsName">제목</option>
      </select>

      <input type="text" name="keyword" value=""
             class="form-control bg-light border-0 small" placeholder="Search for..." 
             aria-label="Search" aria-describedby="basic-addon2" style="width: 190px!important;"/>

    <div class="input-group-append">
      <input type="button" onClick="getSearchList()" class="btn btn-primary" value="검색" />
    </div>
  </div>


```

header.jsp

```
<script>
function getSearchList(){
  $('#searchresult').css({
    "display": "block"
  });

  $.ajax({
    type: 'GET',
    url: "/shop/getSearchList",
    data : $( "#search-form" ).serialize(),
    success : function(result){
      console.log(result);
      //데이터는 여기에
      if(result.length > 1){
        result.forEach(function(item){
          str='<tr>';
          str += '<td>' + item.goodsNum + "</td>";
          str += '<td>' + item.goodsName + "</td>";
          str += '<td>' + item.goodsPrice + "</td>";
          str += '<td>' + item.goodsStock + "</td>";
          str += "</tr>";
        })
        $('#boardtable').append(str);
      }
    }
  });
}
</script>
```

header.jsp – script

```
</form>

<div id="searchResult" class="card shadow mb-4" style="margin: 20px auto; max-width: 600px; display: none;">
  <div class="card-header py-3">
    <h6 class="m-0 font-weight-bold text-primary">검색 결과</h6>
  </div>
  <div class="card-body">
    <div class="table-responsive">

      <table id="boardtable" class="table table-bordered" style="width:100%; cellspacing: 0;">
        <tr>
          <td></td>
        </tr>
      </table>
    </div>
  </div>
</div>
```

header.jsp

## 2. 메인 페이지 - 상품 목록

: aside.jsp -> Mapper -> DAO&Impl -> Service&Impl -> Controller -> list.jsp

```
<li class="nav-item">
    <a href="/shop/list?c=10081" data="nav-link collapsed" data-toggle="collapse">
        <span style="padding-left: 50px; font-size: 20px;">상품 목록</span>
    </a>
    <div id="collapseTwo" class="collapse" aria-labelledby="headingTwo" data-parent="#accordionSidebar">
        <div class="card card-body">
            <div style="display: flex; justify-content: space-between; align-items: center; margin-bottom: 10px;">
                <span style="font-size: 14px; font-weight: bold;">카테고리별 상품 목록

```

aside.jsp

```
public interface ShopService {
    // 카테고리별 상품 리스트
    public List<GoodsViewVO> list(int cateCode, int level) throws Exception;

    @Service
    public class ShopServiceImpl implements ShopService {
        @Inject
        private ShopDAO dao;

        // 카테고리별 상품 리스트
        @Override
        public List<GoodsViewVO> list(int cateCode, int level) throws Exception {
            int cateCodeRef = 0; // 카테고리 첫 번째 코드. 엔드 코드
            if(level == 1) { // level 1 = 1차 목록
                cateCodeRef = cateCode;
                return dao.list(cateCode, cateCodeRef);
            } else { // level 2 = 2차 목록
                return dao.list(cateCode);
            }
        }
    }
}
```

ShopDAO&amp;Impl

```
@Controller
@RequestMapping("/shop/*")
public class ShopController {
    private static final Logger logger = LoggerFactory.getLogger(ShopController.class);

    @Inject
    ShopService service;

    // 카테고리별 상품 리스트
    @RequestMapping(value = "/list", method = RequestMethod.GET)
    public void getList(@RequestParam("c") int cateCode,
                        @RequestParam("l") int level, Model model) throws Exception {
        logger.warn("get list");
        List<GoodsViewVO> list = null;
        list = service.list(cateCode, level);
        model.addAttribute("list", list);
    }
}
```

ShopController

```
<!-- 카테고리별 상품 리스트 : 1차 분류 -->
<select id="list_1" resultType="org.green.domain.GoodsViewVO">
    select
        g.gdsNum, g.gdsName, g.cateCode, c.cateCodeRef, c.cateName,
        gdsPrice, gdsStock, gdsDes, gdsDate, g.gdsImg, g.gdsThumbImg
        from tbl_goods g
        inner join goods_category c
        on g.cateCode = c.cateCode
        where g.cateCode = #{cateCode}
        or c.cateCodeRef = #{cateCodeRef}
    </select>

<!-- 카테고리별 상품 리스트 : 2차 분류 -->
<select id="list_2" resultType="org.green.domain.GoodsViewVO">
    select
        g.gdsNum, g.gdsName, g.cateCode, c.cateCodeRef, c.cateName,
        gdsPrice, gdsStock, gdsDes, g.gdsImg, g.gdsThumbImg
        from tbl_goods g
        inner join goods_category c
        on g.cateCode = c.cateCode
        where g.cateCode = #{cateCode}
    </select>

```

ShopMapper

```
public interface ShopDAO {
    // 카테고리별 상품 리스트 : 1차 분류
    public List<GoodsViewVO> list(int cateCode, int cateCodeRef) throws Exception;

    @Service
    public class ShopDAOImpl implements ShopDAO {
        @Inject
        private SqlSession sql;
        // ...
        private static String namespace = "org.green.mapper.shopMapper";
        // 카테고리별 상품 리스트 : 1차 분류
        @Override
        public List<GoodsViewVO> list(int cateCode, int cateCodeRef) throws Exception {
            HashMap<String, Object> map = new HashMap<String, Object>();
            map.put("cateCode", cateCode);
            map.put("cateCodeRef", cateCodeRef);
            return sql.selectList(namespace + ".list 1", map);
        }
        // 카테고리별 상품 리스트 : 2차 분류
        @Override
        public List<GoodsViewVO> list(int cateCode) throws Exception {
            return sql.selectList(namespace + ".list 2", cateCode);
        }
    }
}
```

ShopService&amp;Impl

```
<section id="content">
    <ul>
        <c:forEach items="${list}" var="list">
            <li>
                <div class="goodsThumb">
                    
                </div>
                <div class="goodName">
                    <a href="/shop/view?n=${list.gdsNum}">${list.gdsName}</a>
                </div>
            </li>
        </c:forEach>
    </ul>
</section>
```

list.jsp

## 2. 메인 페이지 - 상품 조회

: Mapper -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 상품 조회 + 카테고리 조인 -->
<select id="goodsView" resultType="org.green.domain.GoodsViewVO">
    select
        g.gdsNum, g.gdsName, g.cateCode, c.cateCodeRef, c.cateName, gdsPrice,
        gdsStock, gdsDes, gdsImg, gdsDate, g.gdsImg, g.gdsThumbImg
        from tbl_goods g
        inner join goods_category c
        on g.cateCode = c.cateCode
        where g.gdsnum = #{gdsNum}
</select>
```

ShopMapper.xml

```
// 상품조회
public GoodsViewVO goodsView(int gdsNum) throws Exception;

// 상품 조회
@Override
public GoodsViewVO goodsView(int gdsNum) throws Exception {
    return sql.selectOne("org.green.mapper.adminMapper" + ".goodsView", gdsNum);
}
```

ShopDAO&amp;Impl

```
// 상품 조회 get
@RequestMapping(value = "/view", method = RequestMethod.GET)
public void getView(@RequestParam("n") int gdsNum, Model model) throws Exception {
    logger.warn("get view");

    GoodsViewVO view = service.goodsView(gdsNum);
    model.addAttribute("view", view);
```

ShopController

```
// 상품조회
public GoodsViewVO goodsView(int gdsNum) throws Exception;

// 상품 조회
@Override
public GoodsViewVO goodsView(int gdsNum) throws Exception {
    return dao.goodsView(gdsNum);
}
```

ShopService&amp;Impl

```
<form role="form" method="post">
    <input type="hidden" name="gdsNum" value="${view.gdsNum}" />
</form>

<div class="goods">
    <div class="goodsImage">
        
    </div>
    <div class="goodsInfo">
        <span>${view.gdsName}</span> <br> <span>${view.gdsDes}</span>
        <p> ${view.cateName} </p>
        <p> ${view.gdsPrice}</p>
        <span>${view.gdsStock}</span> <br> <span>${view.gdsStock}</span>
        <span>${view.gdsThumbImg}</span> <br> <span>${view.gdsThumbImg}</span>
    </div>
</div>

<div class="cartStock">
    <span>${view.gdsStock}</span>
    <button type="button" class="plus"></button>
    <input type="number" class="numBox" min="2" max=" ${view.gdsStock}" value="1" readonly="readonly"/>
    <button type="button" class="minus"></button>
</div>

<script>
    $(document).ready(function(){
        var num = ${view.gdsStock};
        var plusNum = Number(num) + 1;
        var minusNum = Number(num) - 1;
    });
</script>
```

view.jsp

### 3. 관리자 화면 - 관리자 권한 분리

: AdminInterceptor -> servlet-context.xml -> JSP

```
public class AdminInterceptor extends HandlerInterceptorAdapter {  
  
    @Override  
    public boolean preHandle(HttpServletRequest req, HttpServletResponse res, Object obj)  
        throws Exception {  
  
        HttpSession session = req.getSession();  
        MemberVO member = (MemberVO)session.getAttribute("member");  
  
        if(member == null) {  
            res.sendRedirect("/member/signin");  
            return false;  
        }  
  
        if(member.getVerify() != 9) {  
            res.sendRedirect("/");  
            return false;  
        }  
  
        return true;  
    }  
}
```

# AdminInterceptor

```
<beans:bean id="AdminInterceptor" class="org.green.interceptor.AdminInterceptor"></beans:bean>

<interceptors>
    <interceptor>
        <mapping path="/admin/**" />
        <beans:ref bean="AdminInterceptor" />
    </interceptor>
</interceptors>
```

## servlet-context.xml

```
<c:if test="${member != null}">
    <li class="nav-item"><a class="nav-link active" aria-current="page" href="/">Home</a></li>
    <li class="nav-item"><a class="nav-link" href="/">일반 회원</a></li>
    <li class="nav-item"><a class="nav-link" href="/member/signout">Logout</a></li>
</c:if>
```

nav.jsp

```
<title>Movie_Books Admin</title>
</head>
</body>
<div id="main">
    <div id="header">
        <div id="header_box">
            <img alt="Movie Books logo" style="width: 100px; height: 30px;" data-bbox="111 188 188 218"/>
        </div>
    </div>
    <div id="nav_bar">
        <div id="nav_bar_box">
            <%@include file="include/nav.jsp" %>
        </div>
    </div>
    <div id="maincontent">
        <div id="maincontent_box">
            <%@include file="include/main.jsp" %>
        </div>
    </div>
    <div id="aside">
        <div id="aside_box">
            <%@include file="include/aside.jsp" %>
        </div>
    </div>
    <div id="bottom">
        <div id="bottom_box">
            <%@include file="include/footer.jsp" %>
        </div>
    </div>
</div>
```

index.jsp

### 3. 관리자 화면 - 상품 등록

: Mapper -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 상품 등록 -->
<insert id="register">
    insert into tbl_goods (gdsNum, gdsName, cateCode, gdsPrice, gdsStock, gdsDes,
        gdsImg, gdsThumbImg)
    values (tbl_goods_seq.nextval, #{gdsName}, #{cateCode}, #{gdsPrice}, #{gdsStock}, #{gdsDes},
        #{gdsImg}, #{gdsThumbImg})
</insert>
```

AdminMapper.xml

```
// 상품 등록
public void register(GoodsVO vo) throws Exception;

// 상품 등록
@Override
public void register(GoodsVO vo) throws Exception {
    sql.insert(namespace + ".register", vo);
}
```

AdminDAO&amp;Impl

```
// 상품 등록
@RequiresRoles(value = "ROLE_ADMIN", method = RequestMethod.GET)
public void register(@ModelAttribute("model") Model model) throws Exception {
    logger.info("get goods register");
    List<CategoryVO> category = null;
    category = adminService.category();
    model.addAttribute("category", category);
}

// 상품 등록
@RequiresRoles(value = "ROLE_ADMIN", method = RequestMethod.POST)
public void register(@ModelAttribute("model") Model model) throws Exception {
    String categoryName = model.getParameter("category");
    String imgPath = UploadUtil.uploadThumbnailImage(); // 이미지 업로드 경로
    String titleName = null; // 상품 제목
    String getImgName() {
        if (file.getOriginalFilename() != null && file.getOriginalFilename() != "") {
            return file.getOriginalFilename();
        }
        return UploadUtil.getFileOriginalName(file.getOriginalFilename(), file.getBytes(), imgPath);
    }
    File uploadImage = new File(imgPath + File.separator + "original");
    uploadImage.mkdirs();
    File imgFile = new File(uploadImage, getImgName());
    vo.setGdsImgPath(uploadImage.getAbsolutePath() + File.separator + file.getOriginalFilename());
    vo.setGdsThumbImg(uploadImage.getAbsolutePath() + File.separator + file.getOriginalFilename() + "t" + File.separator + file.getOriginalFilename());
    if (titleName == null) {
        titleName = "images" + File.separator + "none.png";
    }
    vo.setGdsThumbImg(titleName);
    model.addAttribute("registerSuccess");
    return "redirect:/admin/index";
}
```

AdminController

```
// 상품 등록
public void register(GoodsVO vo) throws Exception;

// 상품 등록
@Override
public void register(GoodsVO vo) throws Exception {
    dao.register(vo);
}
```

AdminService&amp;Impl

```
<div id="container_box">
    <h2>상품 등록</h2>
    <form role="form" method="post" autocomplete="off" enctype="multipart/form-data">
        <div class="inputArea">
            <label id="firstCategory">제1차분류</label>
            <select class="category1" name="cateCode">
                <option value="--선택--">선택</option>
            </select>
            <label id="secondCategory">제2차분류</label>
            <select class="category2" name="cateCode">
                <option value="--선택--">선택</option>
            </select>
        </div>
        <div class="inputArea">
            <label for="gdsName">상품명</label>
            <input type="text" id="gdsName" name="gdsName" />
        </div>
        <div class="inputArea">
            <label for="gdsPrice">판매가격</label>
            <input type="text" id="gdsPrice" name="gdsPrice" />
        </div>
        <div class="inputArea">
            <label for="gdsStock">재고량</label>
            <input type="text" id="gdsStock" name="gdsStock" />
        </div>
        <div class="inputArea">
            <label for="gdsDes">상세설명</label>
            <textArea rows="5" cols="50" id="gdsDes" name="gdsDes"></textArea>
        </div>
    </form>

```

register.jsp

### 3. 관리자 화면 - 상품 목록

: Mapper -> DAO&Impl -> Service&Impl -> Controller -> list.jsp -> aside.jsp

```
<select id="goodslist" resultType="org.green.domain.GoodsViewVO">
    select
        g.gdsNum, g.gdsName, g.cateCode, c.cateCodeRef, c.cateName, gdsPrice, gdsStock,
        gdsDes, gdsImg, gdsDate, g.gdsImg, g.gdsThumbImg
    from tbl_goods g
        inner join goods_category c
            on g.cateCode = c.cateCode
</select>
```

AdminMapper.xml

```
// 상품 등록
public void register(GoodsVO vo) throws Exception;

// 상품 목록
@Override
public List<GoodsViewVO> goodslist() throws Exception {
    System.out.println("서비스");
    return dao.goodslist();
}
```

AdminService&amp;Impl

```
<table>
    <thead>
        <tr>
            <th>번호</th>
            <th>상품명</th>
            <th>카테고리</th>
            <th>가격</th>
            <th>재고</th>
        </tr>
    </thead>
    <tbody>
        <c:forEach items="${list}" var="list">
            <tr>
                <td></td>
                <td><a href="/admin/goods/view?n=${list.gdsNum}&n=${list.gdsName}">${list.gdsName}</a></td>
                <td>${list.cateName} -->  
${list.cateCode}</td>
                <td><fmt:formatNumber value="${list.gdsPrice}" pattern="#,###,###" /></td>
                <td><fmt:formatNumber value="${list.gdsStock}" pattern="#,###,###" /></td>
            </tr>
        </c:forEach>
    </tbody>
</table>
```

list.jsp

```
// 상품 등록
public void register(GoodsVO vo) throws Exception;

// 상품 목록
@Override
public List<GoodsViewVO> goodslist() throws Exception {
    return sql.selectList(namespace + ".goodslist");
}
```

AdminDAO&amp;Impl

```
// 상품 목록
@RequestMapping(value = "/goods/list", method = RequestMethod.GET)
public void getGoodsList(Model model) throws Exception {
    logger.warn("get goods list");

    List<GoodsViewVO> list = adminService.goodslist();
    model.addAttribute("list", list);
}
```

AdminController

```
<li class="nav-item">
    <a class="nav-link" href="/admin/goods/list">
        <span style="padding-left: 50px; font-size: 18px;">상품 목록</span>
    </a>
</li>
```

aside.jsp

### 3. 관리자 화면 - 상품 조회

: Mapper -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 상품 조회 + 카테고리 조인 -->
<select id="goodsView" resultType="org.green.domain.GoodsViewVO">
    select
        g.gdsNum, g.gdsName, g.cateCode, c.cateCodeRef, c.cateName, gdsPrice,
        gdsStock, gdsDes, gdsImg, gdsDate, g.gdsImg, g.gdsThumbImg
        from tbl_goods g
        inner join goods_category c
        on g.cateCode = c.cateCode
        where g.gdsnum = #{gdsNum}
</select>
```

AdminMapper.xml

```
// 상품 조회 + 카테고리 조인
public GoodsViewVO goodsView(int gdsNum) throws Exception;

// 상품 조회 + 카테고리 조인
@Override
public GoodsViewVO goodsView(int gdsNum) throws Exception {
    return sql.selectOne(namespace + ".goodsView", gdsNum);
}
```

AdminDAO&amp;Impl

```
// 상품 조회 - 상품 링크 클릭 시 조회 화면으로 이동
@RequestMapping(value = "/goods/view", method = RequestMethod.GET)
public void getGoodsView(@RequestParam("n") int gdsNum, Model model) throws Exception {

    logger.warn("get goods view");

    GoodsViewVO goods = adminService.goodsView(gdsNum);
    model.addAttribute("goods", goods);
}
```

AdminController

```
// 상품 조회 + 카테고리 조인
public GoodsViewVO goodsView(int gdsNum) throws Exception;

// 상품 조회 + 카테고리 조인
@Override
public GoodsViewVO goodsView(int gdsNum) throws Exception {
    return dao.goodsView(gdsNum);
}
```

AdminService&amp;Impl

```
<div id="container_box">
    <h2>상품 조회</h2>
    <form role="form" method="post" autocomplete="off">
        <input type="hidden" name="n" value="${goods.gdsNum}" />
        <div class="inputArea">
            <label>1차분류</label>
            <span class="category1">${goods.cateCode}</span>
        </div>
        <div class="inputArea">
            <label>2차분류</label>
            <span class="category2">${goods.cateCode}</span>
        </div>
        <div class="inputArea">
            <label for="gdsName">상품명</label>
            <span>${goods.gdsName}</span>
        </div>
    </form>

```

view.jsp

### 3. 관리자 화면 - 상품 수정

: Mapper -> VO -> DAO&Impl -> Service&Impl -> modify.jsp -> view.jsp

```
<!-- 상품 수정 -->
<update id="goodsModify">
    update tbl_goods
    set
        gdsName = #{gdsName},
        cateCode = #{cateCode},
        gdsPrice = #{gdsPrice},
        gdsStock = #{gdsStock},
        gdsDes = #{gdsDes},
        gdsImg = #{gdsImg},
        gdsThumbImg = #{gdsThumbImg}
    where gdsNum = ${gdsNum}
</update>
```

AdminMapper.xml

```
// 상품 수정
public void goodsModify(GoodsVO vo) throws Exception;

// 상품 수정
@Override
public void goodsModify(GoodsVO vo) throws Exception {
    dao.goodsModify(vo);
}
```

AdminService&amp;Impl

```
public class GoodsViewVO {
    private int gdsNum;
    private String gdsName;
    private String cateCode;
    private int gdsPrice;
    private int gdsStock;
    private String gdsDes;
    private String gdsImg;
    private Date gdsDate;

    private String cateCodeRef;
    private String cateName;
}
```

GoodsViewVO

```
// 상품 수정
public void goodsModify(GoodsVO vo) throws Exception;

// 상품 수정
@Override
public void goodsModify(GoodsVO vo) throws Exception {
    sql.update(namespace + ".goodsModify", vo);
}
```

AdminDAO&amp;Impl

```
<div id="container_box">
    <h2>상품 수정</h2>
    <form role="form" method="post" autocomplete="off" enctype="multipart/form-data">
        <input type="hidden" name="gdsNum" value="${goods.gdsNum}" />
        <div class="inputArea">
            <label>1차분류</label>
            <select class="category1">
                <option value="">>전체</option>
            </select>
        </div>
        <div class="inputArea">
            <label>2차분류</label>
            <select class="category2" name="cateCode">
                <option value="">>전체</option>
            </select>
        </div>
        <div class="inputArea">
            <label for="gdsName">상품명</label>
            <input type="text" id="gdsName" name="gdsName" value="${goods.gdsName}" />
        </div>
        <div class="inputArea">
            <label for="gdsPrice">상품가격</label>
            <input type="text" id="gdsPrice" name="gdsPrice" value="${goods.gdsPrice}" />
        </div>
```

modify.jsp

```
// 상품 수정 get
@RequestMapping(value = "/goods/modify", method = RequestMethod.GET)
public void getGoodsModify(@RequestParam("n") int gdsNum, Model model) throws Exception {
    logger.warn("get goods modify");

    GoodsViewVO goods = adminService.goodsView(gdsNum);
    model.addAttribute("goods", goods);

    List<CategoryVO> category = null;
    category = adminService.catregory();
    model.addAttribute("category", JSONArray.fromObject(category));
}
```

AdminController

```
<div class="inputArea">
    <button type="button" id="modify_Btn" class="btn btn-warning">수정</button>
    <button type="button" id="delete_Btn" class="btn btn-danger">삭제</button>
</div>

<script>
    var formObj = $("form[role='form']");
    $("#modify_Btn").click(function() {
        formObj.attr("action", "/admin/goods/modify");
        formObj.attr("method", "get");
        formObj.submit();
    });
</script>
```

view.jsp

## 3. 관리자 화면 - 상품 삭제

: Mapper -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 상품 삭제 -->
<delete id="goodsDelete">
    delete from tbl_goods where gdsNum = #{gdsNum}
</delete>
```

AdminMapper.xml

```
// 상품 삭제
public void goodsDelete(int gdsNum) throws Exception;

// 상품 삭제
@Override
public void goodsDelete(int gdsNum) throws Exception {
    sql.delete(namespace + ".goodsDelete", gdsNum);
}
```

AdminDAO&amp;Impl

```
// 상품 삭제
@RequestMapping(value = "/goods/delete", method = RequestMethod.POST)
public String postGoodsDelete(@RequestParam("n") int gdsNum) throws Exception {

    Logger.warn("post goods delete");

    adminService.goodsDelete(gdsNum);

    return "redirect:/admin/index";
}
```

AdminController

```
// 상품 조회 + 카테고리 조인
public GoodsViewVO goodsView(int gdsNum) throws Exception;

// 상품 삭제
@Override
public void goodsDelete(int gdsNum) throws Exception {
    dao.goodsDelete(gdsNum);
}
```

AdminService&amp;Impl

```
$("#delete_Btn").click(function() {
    var con = confirm("정말로 삭제하시겠습니까?");

    if(con) {
        formObj.attr("action", "/admin/goods/delete");
        formObj.attr("method", "post")
        formObj.submit();
    }
});
```

view.jsp

### 3. 관리자 화면 - 이미지 업로드(Ajax)

: Mapper -> VO -> servlet-context.xml -> pom.xml -> UploadFileUtils -> Controller -> JSP

```
<!-- 테이블 -->
<insert id="register">
    insert into tbl_goods (gdsNum, gdsName, cateCode, gdsPrice, gdsStock, gdsDes,
    gdsImg, gdsThumbImg)
    values (tbl_goods_seq.nextval, #{gdsName}, #{cateCode}, #{gdsPrice}, #{gdsStock}, #{gdsDes},
    #{gdsImg}, #{gdsThumbImg})
</insert>
```

AdminMapper.xml

```
private String gdsImg;
private Date gdsDate;

private String gdsThumbImg;

public String getGdsThumbImg() {
    return gdsThumbImg;
}
public void setGdsThumbImg(String gdsThumbImg) {
    this.gdsThumbImg = gdsThumbImg;
}
```

GoodsVO

```
<!-- 업로드 폴더 설정 -->
<beans:bean class="java.lang.String" id="uploadPath">
    <beans:constructor-arg value="D:\springworkspace\.metadata.plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps\wave-books\resources" />
</beans:bean>

<!-- 일반 파일 업로드 경로 -->
<resources mapping="/imgUpload/**" location="/resources/imgUpload/" />

<beans:bean class="org.springframework.web.multipart.commons.CommonsMultipartResolver"
id="multipartResolver">
    <beans:property name="maxUploadSize" value="10485760"/>
</beans:bean>
```

servlet-context.xml

```
<!-- https://mvnrepository.com/artifact/commons-fileupload/commons-fileupload -->
<dependency>
    <groupId>commons-fileupload</groupId>
    <artifactId>commons-fileupload</artifactId>
    <version>1.3.3</version>
</dependency>

<!-- https://mvnrepository.com/artifact/net.coobird/thumbnailator -->
<dependency>
    <groupId>net.coobird</groupId>
    <artifactId>thumbnailator</artifactId>
    <version>0.4.8</version>
</dependency>
```

pom.xml

```
public class UploadfileUtils {
    static final int THUMB_WIDTH = 300;
    static final int THUMB_HEIGHT = 300;

    public static String fileupload(String uploadPath, String fileName,
        byte[] fileData, String ymlPath) throws Exception {
        UUID uid = UUID.randomUUID();
        String newFileName = uid + "_" + fileName;
        String imgPath = uploadPath + ymlPath;
        File target = new File(imgPath, newFileName);
        FileCopyUtils.copy(fileData, target);

        String thumbFileName = "s_" + newFileName;
        File image = new File(imgPath + File.separator + newFileName);
        File thumbnail = new File(imgPath + File.separator + "s" + File.separator + thumbFileName);

        if (image.exists()) {
            thumbnail.createNewFile();
            Thumbnails.of(image).size(THUMB_WIDTH, THUMB_HEIGHT).toFile(thumbnail);
        }
        return newFileName;
    }

    public static String calcPath(String uploadPath) {
        Calendar cal = Calendar.getInstance();
    }
}
```

UploadFileUtils

```
// 등록 post
@RequestBody(value = "/goods/register", method = RequestMethod.POST)
public String postGoodsRegister(GoodsVO vo, MultipartFile file) throws Exception {
    String imgUploadPath = uploadPath + File.separator + "imgUpload"; // 이미지를 업로드할 폴더를 설정 - uploadPath/imgUpload
    String ymlPath = uploadPath + File.separator + "ymlPath"; // 파일 경로를 기준으로 업로드 폴더를 정함
    String fileName = null; // 기본 경로와 일치하는 경로 - 파일이름

    if(file.getOriginalFilename() != null && file.getOriginalFilename().length() != 0) {
        // 파일 업로드에 첨부된 파일이 있으면(첨부된 파일이 있을 경우)
        #fileName = UploadfileUtils.fileupload(uploadPath, file.getOriginalFilename(), file.getBytes(), ymlPath);
        // 업로드된 파일의 원본명을 파일이름으로 설정(첨부된 파일이 있을 경우)
        vo.setGdsImg(fileName);
        // gdsThumbImg 파일 경로 설정
        vo.setGdsThumbImg(ymlPath + File.separator + "imgUpload" + ymlPath + File.separator + "s" + File.separator + "s_" + fileName);
    } else { // 첨부된 파일이 없으면
        #fileName = File.separator + "images" + File.separator + "none.jpg";
        // 기본 경로와 none.jpg경로를 디렉토리 경로로 설정
        vo.setGdsImg(fileName);
        vo.setGdsThumbImg(fileName);
    }
    adminService.register(vo);
    return "redirect:/admin/index";
}
```

AdminController

```
<div class="inputArea">
    <label for="gdsImg">이미지</label>
    <input type="file" id="gdsImg" name="file" />
    <div class="select_img"><img src="" /></div>
<script>
    $("#gdsImg").change(function() {
        if(this.files && this.files[0]) {
            var reader = new FileReader();
            //console.log("파일업로드")
            reader.onload = function(data) {
                $(".select_img img").attr("src", data.target.result).width(500);
            }
            reader.readAsDataURL(this.files[0]);
        }
    });
</script>
<%=request.getRealPath("/") %>
</div>
```

register.jsp

## 3. 관리자 화면 - 위치 편집기

: servlet-context.xml -> register.jsp -> Controller -> modify.jsp

```
<!-- ck에디터 파일 업로드 경로 -->
<resources mapping="/ckUpload/**" location="/resources/ckUpload/" />
```

servlet-context.xml

```
<script>
    var ckeditor_config = {
        resize_enable : false,
        enterMode : CKEDITOR.ENTER_BR,
        shiftEnterMode : CKEDITOR.ENTER_P,
        filebrowserUploadUrl : "/admin/goods/ckUpload"
    };

    CKEDITOR.replace("gdsDes", ckeditor_config);
</script>
```

register.jsp

```
// ck 에디터 파일 업로드
@Name("CKEditorCustom")
@HandlesRequest("goods/ckUpload")
@Method(RequestMethod.POST)
public void postCKEditorImgUpload(HttpServletRequest req, HttpServletResponse res,
@RequestParam MultipartFile upload) throws Exception {
    logger.warn("post CKEditor img upload");
    // UUID 생성
    UUID uid = UUID.randomUUID();
    OutputStream out = null;
    PrintWriter printWriter = null;
    // UTF-8 설정
    res.setCharacterEncoding("utf-8");
    res.setContentType("text/html;charset utf-8");
    try {
        String fileName = upload.getOriginalFilename(); // 파일 이름 가져오기
        byte[] bytes = upload.getBytes();
        // 파일 저장
        String ckPath = goodsPath + File.separator + "ckUpload" + File.separator + uid + "_" + fileName;
        logger.info("file save path : " + ckPath);
        out = new FileOutputStream(new File(ckUploadPath));
        out.write(bytes);
        out.flush(); // 파일 저장 후 파일 크기 초기화
        String callback = req.getParameter("CKEditorFuncNum");
        logger.warn("callback : " + callback);
        printWriter = res.getWriter();
        String fileURL = "/resources/ckUpload/" + uid + "_" + fileName; // 파일 URL
        printWriter.println(fileURL);
    } catch (Exception e) {
        logger.error("file upload error : " + e.getMessage());
    }
}
```

AdminController

```
<div class="inputArea">
    <label for="gdsDes">상품소개</label>
    <textarea rows="5" cols="50" id="gdsDes" name="gdsDes">${goods.gdsDes}</textarea>
</div>

<script>
    var ckeditor_config = {
        resize_enable : false,
        enterMode : CKEDITOR.ENTER_BR,
        shiftEnterMode : CKEDITOR.ENTER_P,
        filebrowserUploadUrl : "/admin/goods/ckUpload"
    };

    CKEDITOR.replace("gdsDes", ckeditor_config);
</script>
</div>
```

modify.jsp

## 4. 구매 - 장바구니 담기

: Mapper -> VO -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 카트 담기 -->
<insert id="addCart">
    insert into tbl_cart (cartNum, userId, gdsNum, cartStock)
        values (tbl_cart_seq.nextval, #{userId}, #{gdsNum}, #{cartStock})
</insert>
```

ShopMapper.xml

```
private int cartNum;
private String userId;
private int gdsNum;
private int cartStock;
private Date addDate;

public int getCartNum() {
    return cartNum;
}
public void setCartNum(int cartNum) {
    this.cartNum = cartNum;
}
public String getUserId() {
    return userId;
}
public void setUserId(String userId) {
```

CartVO

```
// 카트 담기
public void addCart(CartListVO cart) throws Exception;

// 카트 담기
@Override
public void addCart(CartListVO cart) throws Exception {
    sql.insert(namespace + ".addCart", cart);
}
```

ShopDAO&amp;Impl

```
public interface ShopDAO {
    // 카테고리별 상품 리스트 : 1차 분류
    public List list(int cateCode, int cateCodeRef) throws Exception;

    // 카테고리별 상품 리스트 : 2차 분류
    public List list(int cateCode) throws Exception;

    // 카트 담기
    @Override
    public void addCart(CartListVO cart) throws Exception {
        dao.addCart(cart);
    }
}
```

ShopService&amp;Impl

```
// 카트 담기
@ResponseBody
@RequestMapping(value = "/view/addCart", method = RequestMethod.POST)
public int addCart(CartListVO cart, HttpSession session) throws Exception {
    int result = 0;
    MemberVO member = (MemberVO) session.getAttribute("member");
    if(member != null) {
        cart.setUserId(member.getUserId());
        service.addCart(cart);
        result = 1;
    }
    return result;
}
```

ShopController

```
<script>
    $('#addCart_btn').click(function(){
        var gdsNum = $('#gdsNum').val();
        var cartStock = $('#numBox').val();
        console.log("gdsNum : " + gdsNum);
        console.log("cartStock : " + cartStock);

        // ReplyVO 형태로 데이터 생성
        var data = {
            gdsNum : gdsNum,
            cartStock : cartStock
        };

        $.ajax({
            url : "/shop/view/addCart",
            type : "post",
            data : data,
            success : function(result){
                if(result == 1){
                    alert("장바구니 담기 완료");
                    $('#numBox').val('1');
                } else {
                    alert("로그인에 필요한 기능입니다.");
                    $('#numBox').val('1');
                }
            },
            error : function(){
                alert("장바구니 담기 실패");
            }
        });
    });
</script>
```

view.jsp

## 4. 구매 - 장바구니 목록

: Mapper -> VO -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 카트 리스트 -->
<select id="cartList" resultType="org.green.domain.CartListVO">
    select
        row_number() over(order by c.cartNum desc) as num,
        c.cartNum, c.userId, c.gdsNum, c.cartStock, c.addDate,
        g.gdsName, g.gdsPrice, g.gdsThumbImg
    from tbl_cart c
        inner join tbl_goods g
            on c.gdsNum = g.gdsNum
        where c.userId = #{userId}
</select>
```

ShopMapper.xml

```
// 카트 리스트
public List<CartListVO> cartList(String userId) throws Exception;

// 카트 리스트
@Override
public List<CartListVO> cartList(String userId) throws Exception {
    return sql.selectList(namespace + ".cartList", userId);
}
```

ShopDAO&amp;Impl

```
// 카트 목록
@RequestMapping(value = "/cartList", method = RequestMethod.GET)
public void getCartList(HttpServletRequest session, Model model) throws Exception {
    Logger.warn("get cart list");

    MemberVO member = (MemberVO)session.getAttribute("member");
    String userId = member.getUserId();

    List<CartListVO> cartList = service.cartList(userId);

    model.addAttribute("cartList", cartList);
}
```

ShopController

```
private int cartStock;
private Date addDate;

private int num;
private String gdsName;
private int gdsPrice;
private String gdsThumbImg;

public int getNum() {
    return num;
}
public void setNum(int num) {
    this.num = num;
}
public String getGdsName() {
    return gdsName;
}
```

CartListVO

```
// 카트 리스트
public List<CartListVO> cartList(String userId) throws Exception;

// 카트 리스트
@Override
public List<CartListVO> cartList(String userId) throws Exception {
    return dao.cartList(userId);
}
```

ShopService&amp;Impl

```
<c:set var="sum" value="0" />
<c:foreach items="${cartList}" var="cartList">
    <div class="checkbox">
        <input type="checkbox" name="chBox" class="chBox" data.cartNum="${cartList.cartNum}" data.cartPrice="${cartList.gdsPrice * cartList.cartStock}" />
    </div>
    <script>
        $(".chBox").click(function(){
            //만일 체크된다면
            if($(".chBox").is(":checked")){
                $("#allcheck").prop("checked", false);
            }
        });
    </script>
    <div class="thumb">
        
    </div>
    <div class="gdsInfo">
        <p>
            <span>상품명 : </span><span>${cartList.gdsName}<br /></span>가격 : </span>
            <input formNumber="###,###,###" value="${cartList.gdsPrice}" /> <br />
            <span>구입 수량 : </span><span>${cartList.cartStock}<br /></span>금액 : </span>
            <input formNumber="###,###,###" value="${cartList.gdsPrice * cartList.cartStock}" /> <br />
        </p>
    </div>
</c:foreach>
```

CartList.jsp

## 4. 구매 - 장바구니 삭제

: Mapper -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 카트 삭제 -->
<delete id="deleteCart">
    delete tbl_cart
        where cartNum = #{cartNum}
            and userId = #{userId}
</delete>
```

ShopMapper.xml

```
// 카트 삭제
public void deleteCart(CartVO cart) throws Exception;

// 카트 삭제
@Override
public void deleteCart(CartVO cart) throws Exception {
    sql.delete(namespace + ".deleteCart", cart);
}
```

ShopDAO&amp;Impl

```
<div class="delete">
<button type="button" class="delete ${cartList.cartNum} btn" data-cartNum="${cartList.cartNum}">삭제</button>
</div>
$(".delete_${cartList.cartNum}_btn").click(function(){
    var confirm_val = confirm("정말 삭제하시겠습니까?");
    if(confirm_val) {
        var checkArr = new Array();
        checkArr.push($(this).attr("data-cartNum"));

        $.ajax({
            url : "/shop/deleteCart",
            type : "post",
            data : {checkbox : checkArr},
            success : function(result){
                if(result == 1) {
                    location.href = "/shop/cartList";
                } else {
                    alert("삭제 실패");
                }
            }
        });
    }
});
```

ShopController

```
// 카트 삭제
public void deleteCart(CartVO cart) throws Exception;

// 카트 삭제
@Override
public void deleteCart(CartVO cart) throws Exception {
    dao.deleteCart(cart);
}
```

ShopService&amp;Impl

```
// 품목 삭제
@RequestBody
@RequestMapping(value = "/deleteCart", method = RequestMethod.POST)
public int deleteCart(HttpServletRequest session,
                      @RequestParam("checkbox[]") List<String> chArr, CartVO cart) throws Exception {
    logger.warn("delete cart");

    MemberVO member = (MemberVO)session.getAttribute("member");
    String userId = member.getUserId();

    int result = 0;
    int cartNum = 0;

    if(member != null) {
        cart.setUserId(userId);

        for(String i : chArr) {
            cartNum = Integer.parseInt(i);
            cart.setCartNum(cartNum);
            service.deleteCart(cart);
        }
        result = 1;
    }
    return result;
}
```

CartList.jsp

## 4. 구매 - 주문 정보 입력

: Mapper -> VO -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 주문 정보 -->
<insert id="orderInfo">
    insert into tbl_order(orderId, userId, orderRec, userAddr1, userAddr2, userAddr3, orderPhon, amount)
        values(#{orderId}, #{userId}, #{orderRec}, #{userAddr1}, #{userAddr2}, #{userAddr3}, #{userAddr4}, #{orderPhon}, #{amount})
</insert>

<!-- 주문 상세정보 -->
<insert id="orderInfo_Details">
    insert into tbl_order_details(orderDetailsNum, orderId, gdsNum, cartStock)
        select tbl_order.details_seq.nextval, #{orderId}, gdsNum, cartStock
        from tbl_cart
</insert>

<!-- 카트 비우기 -->
<delete id="cartAllDelete">
    delete from tbl_cart
        where userId = #{userId}
</delete>
```

ShopMapper.xml

```
// 주문 정보
public void orderInfo(OrderVO order) throws Exception;

// 주문 상세정보
public void orderInfo_Details(OrderDetailVO orderDetail) throws Exception;

// 카트 비우기
public void cartAllDelete(String userId) throws Exception;
@Override
public void orderInfo(OrderVO order) throws Exception {
    sql.insert(namespace + ".orderInfo", order);
}

// 주문 상세정보
@Override
public void orderInfo_Details(OrderDetailVO orderDetail) throws Exception {
    sql.insert(namespace + ".orderInfo_Details", orderDetail);
}

// 카트 비우기
@Override
public void cartAllDelete(String userId) throws Exception {
    sql.delete(namespace + ".cartAllDelete", userId);
}
```

ShopDAO&amp;Impl

```
// 주문 정보
@RequestMapping(value = "/cartList", method = RequestMethod.POST)
public String order(HttpServletRequest session, OrderVO order, OrderDetailVO orderDetail) throws Exception {
    logger.warn("order");
    MemberVO member = (MemberVO)session.getAttribute("member");
    String orderId = member.getId();
    Calendar cal = Calendar.getInstance();
    int year = cal.get(Calendar.YEAR);

    String ymd = year + new DecimalFormat("00").format(cal.get(Calendar.MONTH + 1));
    String ymd_ = ymd + new DecimalFormat("00").format(cal.get(Calendar.DATE));
    String subnum = "";

    for(int i = 1; i < 6; i++) {
        subnum += (int)(Math.random() * 10);
    }

    String orderId_ = ymd + "_" + subnum;
    order.setOrderId(orderId_);
    order.setUserId(member.getId());
    service.orderInfo(order);
    orderDetail.setOrderid(orderId_);
    orderDetail.setUserId(member.getId());
    service.cartAllDelete(userId);
    return "redirect:/shop/orderList";}
```

ShopController

```
private String orderId;
private String UserId;
private String orderRec;
private String userAddr1;
private String userAddr2;
private String userAddr3;
private String orderPhon;
private int amount;
private Date orderDate;

private String delivery;

public String getDelivery() {
    return delivery;
}

public void setDelivery(String delivery) {
    this.delivery = delivery;
}

public String getOrderId() {
    return orderId;
}

public void setOrderId(String orderId) {
    this.orderId = orderId;
}
```

OrderVO

```
// 주문 정보
public void orderInfo(OrderVO order) throws Exception;

// 주문 상세정보
public void orderInfo_Details(OrderDetailVO orderDetail) throws Exception;

// 카트 비우기
public void cartAllDelete(String userId) throws Exception;
@Override
public void orderInfo(OrderVO order) throws Exception {
    dao.orderInfo(order);
}

// 주문 상세정보
@Override
public void orderInfo_Details(OrderDetailVO orderDetail) throws Exception {
    dao.orderInfo_Details(orderDetail);
}

// 카트 비우기
@Override
public void cartAllDelete(String userId) throws Exception {
    dao.cartAllDelete(userId);
}
```

ShopService&amp;Impl

```
<div class="listResult">
    <div class="sum">
        <input type="text" pattern="###,###,###" value="$sum" />
    </div>
    <div class="orderOpen">
        <button type="button" class="orderOpen_bnt">주문 신청</button>
        <script>
            $(".orderOpen_bnt").click(function(){
                $(".orderInfo").slideDown();
                $(".orderOpen_bnt").slideUp();
            });
            $(document).ready(function(){
                $(".chBox").each((index,checkbox)>{
                    $(checkbox).on("change",function(){
                        //체크박스를 때 살때
                        let sumprice = 0;
                        $( ".chBox" ).each((index,checkbox2) =>{
                            if($(checkbox2).prop("checked")){
                                sumprice += $(checkbox2).data("cartprice");
                            }
                        });
                        $( ".sum" ).text("총금액 : " + sumprice + " 원");
                        $("inputprice").val(sumprice);
                    });
                });
            });
        </script>
    </div>

```

CartList.jsp

## 4. 구매 - 주문 목록

: Mapper -> DAO&Impl -> Service&Impl -> Controller -> OrderList.jsp -> nav.jsp

```
<!-- 특정 유저의 주문 목록 -->
<select id="orderList" resultType="org.green.domain.OrderVO">
    select
        orderId, userId, orderRec, userAddr1, userAddr2, userAddr3, orderPhon, amount, orderDate, delivery
    from tbl_order
    where userId = #{userId}
</select>
```

ShopMapper.xml

```
// 주문 목록
public List<OrderVO> orderList(OrderVO order) throws Exception;

// 주문목록
@Override
public List<OrderVO> orderList(OrderVO order) throws Exception {
    return dao.orderList(order);
}
```

ShopService&amp;Impl

```
<section id="content">
    <ul class="orderList">
        <c:foreach items="${orderList}" var="orderList">
            <li>
                <div>
                    <p><span>주문번호</span><a href="/shop/orderView?n=${orderList.orderId}">${orderList.orderId}</a></p>
                    <p><span>수령인</span> ${orderList.orderRec}</p>
                    <p><span>주소</span> ${orderList.userAddr1} ${orderList.userAddr2} ${orderList.userAddr3}</p>
                    <p><span>가격</span><fmt:formatNumber pattern="#,##,##" value="${orderList.amount}" /> 원</p>
                    <p><span>상태</span> ${orderList.delivery}</p>
                </div>
            </li>
        </c:foreach>
    </ul>
```

OrderList.jsp

```
// 주문 목록
public List<OrderVO> orderList(OrderVO order) throws Exception;

// 주문목록
@Override
public List<OrderVO> orderList(OrderVO order) throws Exception {
    return sql.selectList(namespace + ".orderList", order);
}
```

ShopDAO&amp;Impl

```
// 주문 목록
@RequestMapping(value = "/orderList", method = RequestMethod.GET)
public void getOrderList(HttpServletRequest session, OrderVO order, Model model) throws Exception {
    Logger.warn("get order list");

    MemberVO member = (MemberVO)session.getAttribute("member");
    String userId = member.getUserId();

    order.setUserId(userId);

    List<OrderVO> orderList = service.orderList(order);
    model.addAttribute("orderList", orderList);
}
```

ShopController

```
<li class="nav-item"><a class="nav-link" href="/shop/orderList">주문 목록</a></li>
```

nav.jsp

## 4. 구매 - 주문 상세 보기

: Mapper -> VO -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 특정 주문 목록 -->
<select id="orderView" resultType="org.green.domain.OrderListVO">
    select
        o.orderId, o.userId, o.orderRec, o.userAddr1, o.userAddr2, o.userAddr3, o.orderPhon, o.amount,
        o.orderDate, o.delivery,
        d.orderDetailsNum, d.gdsNum, d.cartStock,
        g.gdsName, g.gdsThumbImg, g.gdsPrice
    from tbl_order o
        inner join tbl_order_details d
            on o.orderId = d.orderId
        inner join tbl_goods g
            on d.gdsNum = g.gdsNum
    where o.userId = #{userId}
        and o.orderId = #{orderId}
</select>
```

ShopMapper.xml

```
// 특정 주문 목록
public List<OrderListVO> orderView(OrderVO order) throws Exception;

// 특정 주문목록
@Override
public List<OrderListVO> orderView(OrderVO order) throws Exception {
    return sql.selectList(namespace + ".orderView", order);
}
```

ShopDAO&amp;Impl

```
// 주문 상세 목록
@RequestMapping(value = "/orderView", method = RequestMethod.GET)
public void getOrderList(HttpServletRequest session,
    @RequestParam("n") String orderId,
    OrderVO order, Model model) throws Exception {
    logger.warn("get order view");

    MemberVO member = (MemberVO)session.getAttribute("member");
    String userId = member.getUserId();

    order.setUserId(userId);
    order.setOrderId(orderId);

    List<OrderListVO> orderView = service.orderView(order);
    model.addAttribute("orderView", orderView);
}
```

ShopController

```
private String orderId;
private String userId;
private String orderRec;
private String userAddr1;
private String userAddr2;
private String userAddr3;
private String orderPhon;
private int amount;
private Date orderDate;

private int orderDetailsNum;
private int gdsNum;
private int cartStock;

private String gdsName;
private String gdsThumbImg;
private int gdsPrice;
```

OrderListVO

```
// 특정 주문 목록
public List<OrderListVO> orderView(OrderVO order) throws Exception;

// 특정 주문목록
@Override
public List<OrderListVO> orderView(OrderVO order) throws Exception {
    return dao.orderView(order);
}
```

ShopService&amp;Impl

```
<ul class="orderView">
    <c:forEach items="${orderView}" var="orderView">
        <li>
            <div class="thumb">
                
            </div>
            <div class="gdsInfo">
                <p>
                    <span>상품명</span> ${orderView.gdsName}<br />
                    <span>가격</span><fmt:formatNumber pattern="#,###,###" value="${orderView.gdsPrice}" /> 원<br />
                    <span>구입 수량</span> ${orderView.cartStock} &lt;br />
                    <span>최종 가격</span><fmt:formatNumber pattern="#,###,###" value="${orderView.gdsPrice * orderView.cartStock}" /> 원
                </p>
            </div>
        </li>
    </c:forEach>
</ul>
```

OrderView.jsp

## 4. 구매 - 배송 상태

: Mapper -> DAO&Impl -> Service&Impl -> Controller -> OrderList.jsp -> OrderView.jsp

```
<!-- 배송 상태 -->
<update id="delivery">
    update tbl_order
    set delivery = #{delivery}
    where orderId = #{orderId}
</update>

<!-- 상품 수량 조절 -->
<update id="changeStock">
    update tbl_goods
    set gdsStock = gdsStock - #{gdsStock}
    where gdsNum = #{gdsNum}
</update>

<!-- 상품 수량 조절 보조 -->
<select id="changeStock_sub" resultType="org.green.domain.OrderDetailVO">
    select
        orderId, gdsNum, cartStock
    from tbl_order_details
    where orderId = #{orderId}
</select>
```

AdminMapper.xml

```
// 배송 상태
public void delivery(OrderVO order) throws Exception;

// 상품 수량 조절
public void changeStock(GoodsVO goods) throws Exception;

// 배송 상태
@Override
public void delivery(OrderVO order) throws Exception {
    dao.delivery(order);
}

// 상품 수량 조절
@Override
public void changeStock(GoodsVO goods) throws Exception {
    dao.changeStock(goods);
}
```

AdminService&amp;Impl

```
<ul class="orderList">
    <c:forEach items="${orderList}" var="orderList">
        <li>
            <div>
                <p><a href="/admin/shop/orderView?n=${orderList.orderId}">${orderList.orderId}</a></p>
                <p>주문번호</p>
                <p>주문자</p>
                <p>수령인</p>
                <p>주소</p>
                <p>가격</p>
                <fmt:formatNumber pattern="###,###,###" value="${orderList.amount}" /> 원</p>
                <p>상태</p>
                <p>${orderList.delivery}</p>
            </div>
        </li>
    </c:forEach>
</ul>
```

OrderList.jsp

```
// 배송 상태
public void delivery(OrderVO order) throws Exception;

// 상품 수량 조절
public void changeStock(GoodsVO goods) throws Exception;

// 배송 상태
@Override
public void delivery(OrderVO order) throws Exception {
    sql.update(namespace + ".delivery", order);
}

// 상품 수량 조절
@Override
public void changeStock(GoodsVO goods) throws Exception {
    sql.update(namespace + ".changeStock", goods);
}
```

AdminDAO&amp;Impl

```
// 주문 상세 목록 - 상태 변경
@RequestMapping(value = "/shop/orderView", method = RequestMethod.POST)
public String delivery(OrderVO order) throws Exception {
    Logger.warn("post order view");
    adminService.delivery(order);

    // 상품 수량 조절
    List<OrderListVO> orderView = adminService.orderView(order);
    GoodsVO goods = new GoodsVO();

    for(OrderListVO i : orderView) {
        goods.setGdsNum(i.getGdsNum());
        goods.setGdsStock(i.getCartStock());
        adminService.changeStock(goods);
    }

    return "redirect:/admin/shop/orderView?n=" + order.getOrderId();
}
```

AdminController

```
<p><span>상태</span>${orderView.delivery}</p>
<div class="deliveryChange">
    <form role="form" method="post" class="deliveryForm">
        <input type="hidden" name="orderId" value="${orderView.orderId}" />
        <input type="hidden" name="delivery" class="delivery" value="" />
        <button type="button" class="delivery1_btn">배송 완료</button>
        <button type="button" class="delivery2_btn">배송 미완료</button>
    </form>
    <script>
        $(".delivery1_btn").click(function(){
            console.log("실행");
            $(".delivery").val("배송 완료");
            run();
        });
        $(".delivery2_btn").click(function(){
            $(".delivery").val("배송 미완료");
            run();
        });
        function run(){
            $(".deliveryForm").submit();
        }
    </script>

```

OrderView.jsp

## 4. 구매 - 주소 API (다음)

: CartList.jsp

```

<div class="orderInfo">
    <form role="form" method="post" autocomplete="off">
        <input type="hidden" name="amount" id="inputprice" value="${sum} />
        <div class="inputArea">
            <label for="">수령인</label>
            <input type="text" name="orderRec" id="orderRec" required="required" />
        </div>
        <div class="inputArea">
            <label for="orderPhon">수령인 연락처</label>
            <input type="text" name="orderPhon" id="orderPhon" required="required" />
        </div>
        <!--
        <div class="inputArea">
            <label for="userAddr1">우편번호</label>
            <input type="text" name="userAddr1" id="userAddr1" required="required" />
        </div>
        <div class="inputArea">
            <label for="userAddr2">1차 주소</label>
            <input type="text" name="userAddr2" id="userAddr2" required="required" />
        </div>
        <div class="inputArea">
            <label for="userAddr3">2차 주소</label>
            <input type="text" name="userAddr3" id="userAddr3" required="required" />
        </div>
        -->
    </form>

```

CartList.jsp

```

<div class="inputArea">
    <p>
        <input type="text" id="sample2_postcode" placeholder="우편번호">
        <input type="button" onclick="sample2_execDaumPostcode()" value="우편번호 찾기"/>
    </p>
    <p>
        <input type="text" name="userAddr1" id="sample2_address" placeholder="도로">
        <input type="text" name="userAddr2" id="sample2_detailAddress" placeholder="상세주소">
        <input type="text" name="userAddr3" id="sample2_extraAddress" placeholder="영문주소">
    </p>
    <!-- iOS에서는 position:fixed 브가 있음, 적용하는 사이트에 따라 position:absolute 를 이용하여 top, left를 조정 필요 -->
    <div id="layer" style="display:none;position:fixed;overflow:hidden;z-index:1;webkit-overflow-scrolling:touch;">
        
    </div>
    <script src="//t1.daumcdn.net/mapjsapi/bundle/postcode/prod/postcode.v2.js"></script>
    <script>
        // 우편번호 찾기 화면을 넣을 element
        var element_layer = document.getElementById('layer');

        function closeDaumPostcode() {
            // iframe을 넣은 element를 제거하기 한다.
            element_layer.style.display = 'none';
        }

        function sample2_execDaumPostcode() {
            new daum.Postcode({
                onComplete: function(data) {
                    // 선택한 주소를 클릭했을 때 실행할 코드를 작성하는 부분.

                    // 각 주소의 노출 규칙에 따른 주소를 조합한다.
                    // 내려오는 변수가 값이 있는 경우엔 괄호('()')값을 가지므로, 이를 참고하여 분기 한다.
                }
            });
        }
    </script>

```

CartList.jsp

## 5. 상품 소감 - 소감 등록

: Mapper -> VO -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 상품 소감(댓글) 작성 -->
<insert id="registReply">
    insert into tbl_reply (gdsNum, userId, repNum, repCon)
    values (#{gdsNum}, #{userId}, tbl_reply_seq.nextval, #{repCon})
</insert>

<!-- 상품 소감(댓글) 리스트 -->
<select id="replyList" resultType="org.green.domain ReplyListVO">
    select
        r.gdsNum, r.userId, r.repNum, r.repCon, r.repDate, m.userName
    from tbl_reply r
        inner join tbl_member_w m
            on r.userId = m.userId
        where gdsNum = #{gdsNum}
</select>
```

ShopMapper.xml

```
// 상품 소감(댓글) 작성
public void registReply(ReplyVO reply) throws Exception;

// 상품 소감(댓글) 리스트
public List<ReplyListVO> replyList(int gdsNum) throws Exception;

// 상품 소감(댓글) 작성
@Override
public void registReply(ReplyVO reply) throws Exception {
    sql.insert(namespace + ".registReply", reply);
}

// 상품 소감(댓글) 리스트
@Override
public List<ReplyListVO> replyList(int gdsNum) throws Exception {
    return sql.selectList(namespace + ".replyList", gdsNum);
}
```

ShopDAO&amp;Impl

```
// 상품 소감(댓글) 작성
@RequestBody
@RequestMapping(value = "/view/registReply", method = RequestMethod.POST)
public void registReply(ReplyVO reply, HttpSession session) throws Exception {
    Logger.warn("regist reply");
    MemberVO member = (MemberVO) session.getAttribute("member");
    reply.setUserId(member.getUserId());
    service.registReply(reply);
}

// 상품 소감(댓글) 목록
@RequestBody
@RequestMapping(value = "/view/replyList", method = RequestMethod.GET)
public List<ReplyListVO> getReplyList(@RequestParam("n") int gdsNum) throws Exception {
    Logger.warn("get reply list");
    List<ReplyListVO> reply = service.replyList(gdsNum);
    return reply;
}
```

ShopController

```
private int gdsNum;
private String userId;
private int repNum;
private String repCon;
private Date repDate;

public int getGdsNum() {
    return gdsNum;
}
public void setGdsNum(int gdsNum) {
    this.gdsNum = gdsNum;
}
public String getUserId() {
    return userId;
}
public void setUserId(String userId) {
    this.userId = userId;
}
```

ReplyVO

```
// 상품 소감(댓글) 작성
public void registReply(ReplyVO reply) throws Exception;

// 상품 소감(댓글) 리스트
public List<ReplyListVO> replyList(int gdsNum) throws Exception;

// 상품 소감(댓글) 작성
@Override
public void registReply(ReplyVO reply) throws Exception {
    dao.registReply(reply);
}

// 상품 소감(댓글) 리스트
@Override
public List<ReplyListVO> replyList(int gdsNum) throws Exception {
    return dao.replyList(gdsNum);
}
```

ShopService&amp;Impl

```
<%if(text == "member == null")%>
<a href="/member/signIn">로그인</a><%else%></a>
<%endif%>

<%if(text == "member != null")%>
<form class="replyForm" action="form" method="post" autocomplete="off">
    <input type="hidden" name="gdsNum" id="gdsNum" value="<${view.gdsNum}">
    <div class="input_area">
        <input type="text" name="repCon" id="repCon"/>
    </div>
    <div class="input_area">
        <button type="button" id="reply_btn">등록</button>
    </div>
    <script>
        $("#reply_btn").click(function(){
            var formObj = $(".replyForm form[name='form']");
            var gdsNum = $("#gdsNum").val();
            var repCon = $("#repCon").val();
            var data = {
                gdsNum : gdsNum,
                repCon : repCon
            };
            $.ajax({
                url : "/shop/view/registReply",
                type : "post",
                data : data,
                success : function(){
                    replyList();
                    $("#repCon").val("");
                }
            });
        });
    </script>
<%endif%>
```

view.jsp

## 5. 상품 소감 - 소감 수정

: Mapper -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 상품 소감(댓글) 수정 -->
<update id="modifyReply">
    update tbl_reply
    set
        repCon = #{repCon}
    where repNum = #{repNum}
        and userId = #{userId}
</update>
```

ShopMapper.xml

```
// 상품 소감(댓글) 수정
public void modifyReply(ReplyVO reply) throws Exception;

// 상품 소감(댓글) 수정
@Override
public void modifyReply(ReplyVO reply) throws Exception {
    dao.modifyReply(reply);
}
```

ShopService&amp;Impl

```
// 상품 소감(댓글) 수정
public void modifyReply(ReplyVO reply) throws Exception;

// 상품 소감(댓글) 수정
@Override
public void modifyReply(ReplyVO reply) throws Exception {
    sql.update(namespace + ".modifyReply", reply);
}
```

ShopDAO&amp;Impl

```
<script>
$(document).on("click", ".modify", function(){
    // $(".replyModal").attr("style", "display:block");
    $(".replyModal").fadeIn(200);

    var repNum = $(this).attr("data-repNum");
    var repCon = $(this).parent().parent().children(".replyContent").text();

    $(".modal_repCon").val(repCon);
    $(".modal_modify_btn").attr("data-repNum", repNum);
});

</script>
```

view.jsp

```
// 상품 소감(댓글) 수정
@RequestBody
@RequestMapping(value = "/view/modifyReply", method = RequestMethod.POST)
public int modifyReply(ReplyVO reply, HttpSession session) throws Exception {
    Logger.warn("modify reply");

    int result = 0;

    MemberVO member = (MemberVO)session.getAttribute("member");
    String userId = service.idCheck(reply.getRepNum());

    if(member.getUserId().equals(userId)) {
        reply.setUserId(member.getUserId());
        service.modifyReply(reply);
        result = 1;
    }

    return result;
}
```

ShopController

```
<script>
$(".modelCancel").click(function(){
    // $(".replyModal").attr("style", "display:none");
    $(".replyModal").fadeOut(200);
});

<script>
$(".model_modify_btn").click(function(){
    var modifyConfirm = confirm("정말로 수정하시겠습니까?");
    if(modifyConfirm) {
        var data = {
            repNum : $(this).attr("data-repNum"),
            repCon : $(this).parent().parent().children(".replyContent").val()
        };
        $.ajax({
            url : "shop/view/modifyReply",
            type : "post",
            data : data,
            success : function(result){
                // result는 1이면 수정 성공
                // replyListObj // 수정된 내용
                // $(".replyModal").fadeOut(200);
                alert("정말로 수정하시겠습니까?");
            },
            error : function(error){
                alert("수정에 실패하였습니다.");
            }
        });
    }
});
</script>
```

view.jsp

## 5. 상품 소감 - 소감 삭제

: Mapper -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 상품 삭제 -->
<delete id="goodsDelete">
    delete from tbl_goods where gdsNum = #{gdsNum}
</delete>
```

ShopMapper.xml

```
// 상품 삭제
public void goodsDelete(int gdsNum) throws Exception;

// 상품 삭제
@Override
public void goodsDelete(int gdsNum) throws Exception {
    sql.delete(namespace + ".goodsDelete", gdsNum);
}
```

ShopDAO&amp;Impl

```
// 상품 삭제
@RequestMapping(value = "/goods/delete", method = RequestMethod.POST)
public String postGoodsDelete(@RequestParam("h") int gdsNum) throws Exception {

    Logger.warn("post goods delete");

    adminService.goodsDelete(gdsNum);

    return "redirect:/admin/index";
}
```

ShopController

```
// 상품 삭제
public void goodsDelete(int gdsNum) throws Exception;

// 상품 삭제
@Override
public void goodsDelete(int gdsNum) throws Exception {
    dao.goodsDelete(gdsNum);
}
```

ShopService&amp;Impl

```
<script>
$(document).on("click", ".delete", function(){
    var deleteConfirm = confirm("정말로 삭제하시겠습니까?");
    if(deleteConfirm) {
        var data = {repNum : $(this).attr("data-repNum")};
        $.ajax({
            url : "/shop/view/deleteReply",
            type : "post",
            data : data,
            success : function(result){
                if(result == 1) {
                    replyList();
                } else {
                    alert("작성자 본인만 삭제할 수 있습니다.");
                }
            },
            error : function() {
                alert("로그인이 필요합니다.");
            }
        });
    }
});
</script>
```

view.jsp

## 5. 상품 소감 - 소감 관리

: Mapper -> DAO&Impl -> Service&Impl -> Controller -> JSP

```
<!-- 모든 소감(댓글) -->
<select id="allReply" resultType="org.green.domain.ReplyListVO">
    select
        r.gdsNum, r.userId, r.repNum, r.repCon, r.repDate,
        m.userName
    from tbl_reply
        inner join tbl_member_w m
        on r.userId = m.userId
</select>
```

AdminMapper.xml

```
// 모든 소감(댓글)
public List<ReplyListVO> allReply() throws Exception;

// 모든 소감(댓글)
@Override
public List<ReplyListVO> allReply() throws Exception {
    return sql.selectList(namespace + ".allReply");
}
```

AdminDAO&amp;Impl

```
// 모든 소감(댓글) get
@RequestMapping(value = "/shop/allReply", method = RequestMethod.GET)
public void getAllReply(Model model) throws Exception {
    Logger.warn("get all reply");
    List<ReplyListVO> reply = adminService.allReply();
    model.addAttribute("reply", reply);
}

// 모든 소감(댓글) post
@RequestMapping(value = "/shop/allReply", method = RequestMethod.POST)
public String postAllReply(ReplyVO reply) throws Exception {
    Logger.warn("post all reply");
    adminService.deleteReply(reply.getRepNum());
    return "redirect:/admin/shop/allReply";
}
```

AdminController

```
// 모든 소감(댓글)
public List<ReplyListVO> allReply() throws Exception;

// 모든 소감(댓글)
@Override
public List<ReplyListVO> allReply() throws Exception {
    return dao.allReply();
}
```

AdminService&amp;Impl

```
<div id="container_box">
    <c:forEach items="${reply}" var="reply">
        <div class="replyInfo">
            <p><span>작성자</span> ${reply.userName} (${reply.userId})</p>
            <p><span>작성일 상품</span> <a href="/shop/view?n=${reply.gdsNum}">${reply.repDate}</a></p>
        </div>
        <div class="replyContent">
            ${reply.repCon}
        </div>
        <div class="replyControl">
            <form role="form" method="post">
                <input type="hidden" name="repNum" value="${reply.repNum}" />
                <button type="submit" class="delete_${reply.repNum}_btn">삭제</button>
            </form>
        </div>
    </c:forEach>
</div>
```

allReply.jsp



THANK YOU