

Web-Programming



PORTFOLIO

Editor - YUN JAE PIL



01. 프로젝트 개요



02. 개발환경



03. 기획 및 설계



04. 와이어프레임
및 웹 구현



05. 개발구현
및 소스코드

포트폴리오 소개

중고마켓 프로젝트 (SECOND MARKET)

배포 웹사이트 주소 : <http://52.231.206.1:8080/second>

* 배포 웹 사이트는 시일 경과 후 종료될 수 있습니다.

주제 및 역할

주제

MVC 패턴 활용 '중고물품 거래소' & '커뮤니티' 웹사이트 개발

주요 기능

- 회원들 각자가 판매자 또는 구매자가 될 수 있는 시스템
- 신고시스템 및 신고게시판을 공개하여 거래 시 안전요인을 강화 함
- 게시판 및 채팅을 제공하여 회원 간의 커뮤니티 생태계 활성화 유도

역할 및 담당

- 프로젝트 팀장 & 일정 관리 및 문서화 & 깃허브 활용 버전 관리
- 메이븐 빌드 (pom.xml / web.xml / log4j) 세팅 적용
- 공지사항 게시판 & 내상점 (나의 주문 내역 / 판매 내역 / 짐 상품 List) 구현
- 상품 주문 - Daum 주소 검색 API 활용 입력 적용
- Azure 가상 머신 (Windows SERVER 2016) 및 AWS RDS (Oracle 11g) 활용 서버 배포

01. 프로젝트 개요



벤치마킹

당근마켓 <http://www.daangn.com>

당근마켓 인기 매물

제품 이미지	제품 이름	위치	가격	관심 및 채팅 수
	매직체어 새거	서울 노원구 상계2동	50,000원	관심 20 · 채팅 50
	캠핑화롯대	광주 광산구 수완동	15,000원	관심 9 · 채팅 42
	자전거	전북 전주시 완산구 중화...	70,000원	관심 3 · 채팅 30
	타이틀리스트 골프백	경기도 용인시 수지구 등...	50,000원	관심 13 · 채팅 29
	삼천리 자전거	경기도 김포시 고촌읍	15,000원	관심 7 · 채팅 33
	인조가죽 소파 2인용 ...	경남 창원시 의창구 팔용...	10,000원	관심 12 · 채팅 27
	LG전자 제습기 15L 판...	서울 송파구 가락1동	80,000원	관심 5 · 채팅 24
	자전거 구매후 동네세...	대구 달서구 죽전동	50,000원	관심 7 · 채팅 28

중고나라 www.joongna.com

중고나라 검색어를 입력해 주세요.

우리동네 미개봉 새상품 티켓/쿠폰

평화시장

AGK 스피너 스탠딩
에어 써큘레이터

평화시장가
33,900원

2 / 2

재고급차분 | 중고나라가 인증하는 새상품 특가전!

--	--	--	--	--

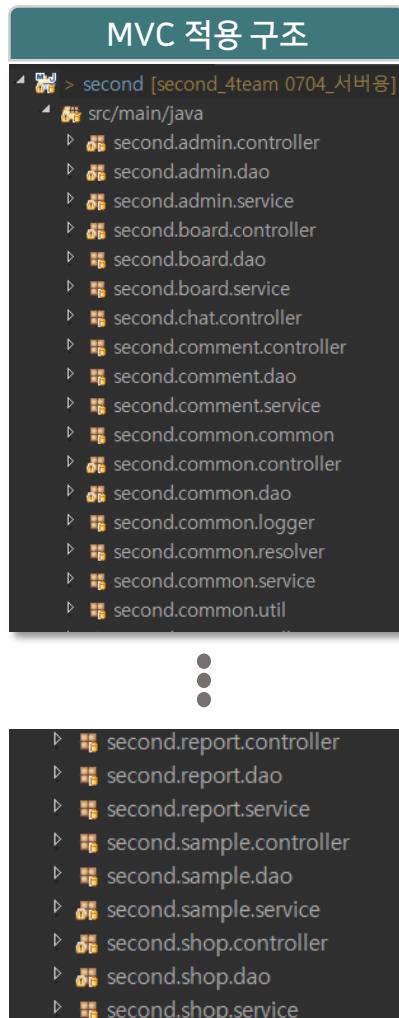
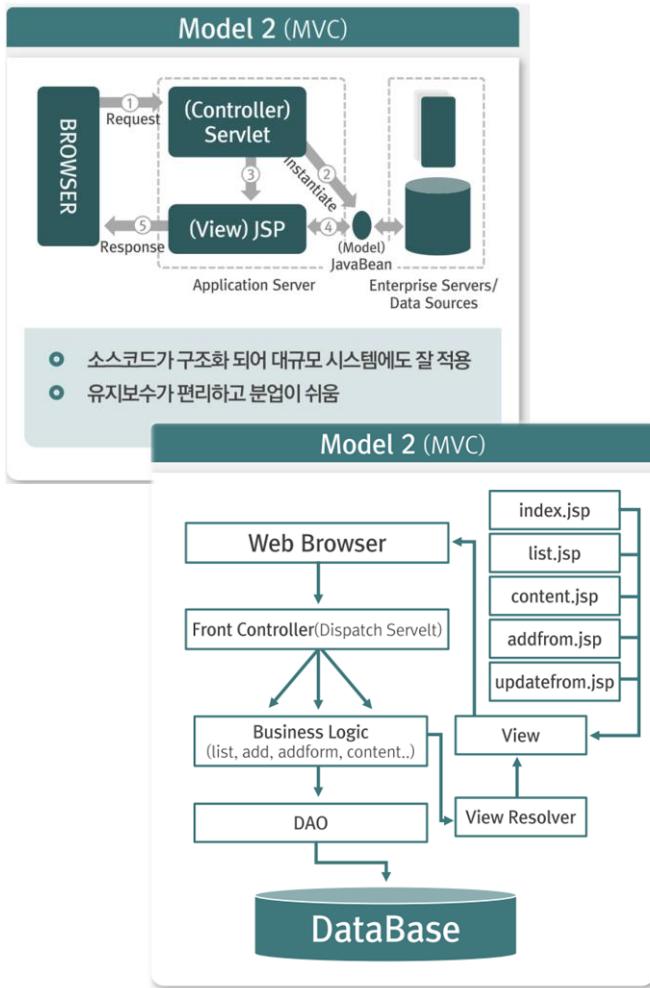
홈 알림 # 마이태그 내정보

🔧 프로젝트 개발활용 도구



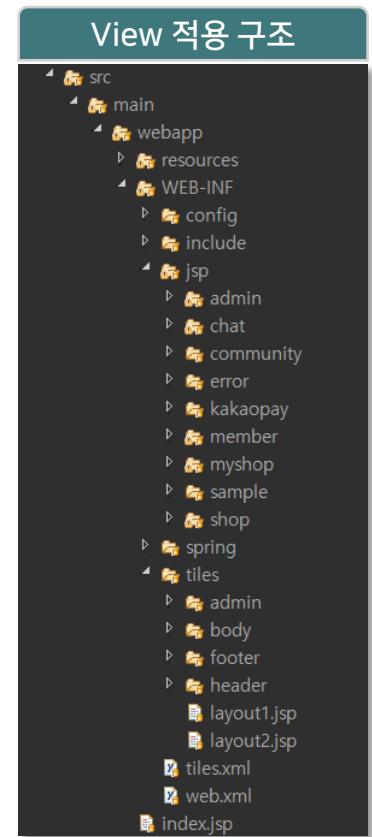
- OS / BROWSER – Windows server 2016 / Google chrome
- Language – Java, JSP, Javascript, Jquery, Oracle SQL, HTML, CSS
- Web Application Server – Tomcat 8.5
- Tool – Spring Tool Suite, VS_code, GitHub, SqlDeveloper
- DB – Oracle 11g
- JDK – 1.7

프레임워크 적용



적용 방식

데이터전달은 Map객체 활용.
컨트롤-서비스-DAO 별 패키지 구성.



View 구조

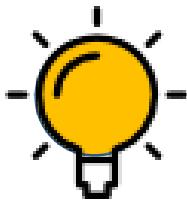
목적별 View폴더를 구분 구현.
타일즈 레이아웃을 활용.

프로젝트 수행과정 요약



기획 및 설계

주제 및 벤치마킹 대상 회의, 스토리보드 작성 및 기획내용 문서화, 개발환경 정의, DB스키마 설계



구현 및 버전관리

애자일 스크럼 방식으로 각각 기능구현,
깃허브를 활용한 소스통합 및 버전관리



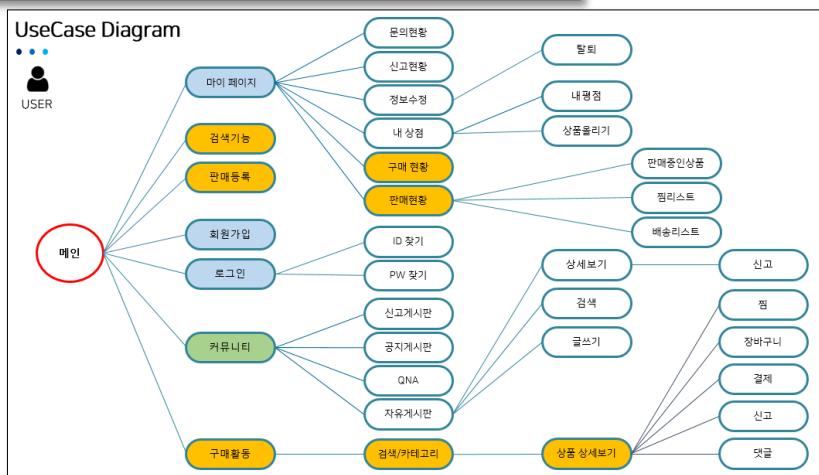
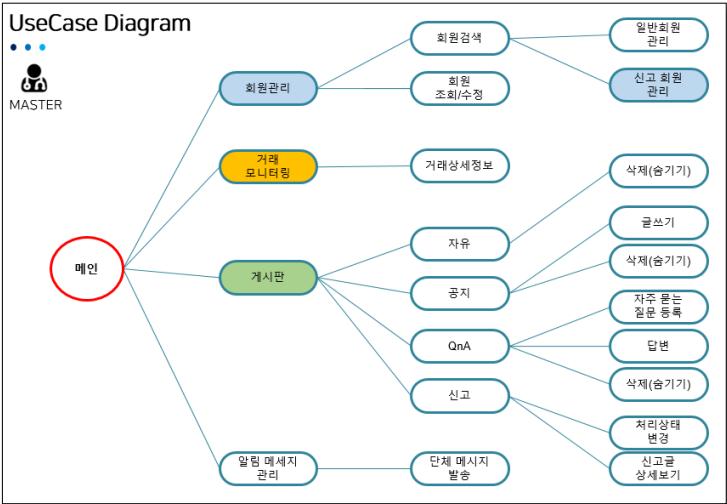
테스트 및 배포

시나리오 테스트, 구조기반 테스트 서비스 최종점검 후 AWS 활용 배포

프로젝트 일정표

유스케이스 다이어그램 & 요구사항 명세서

유스케이스 다이어그램



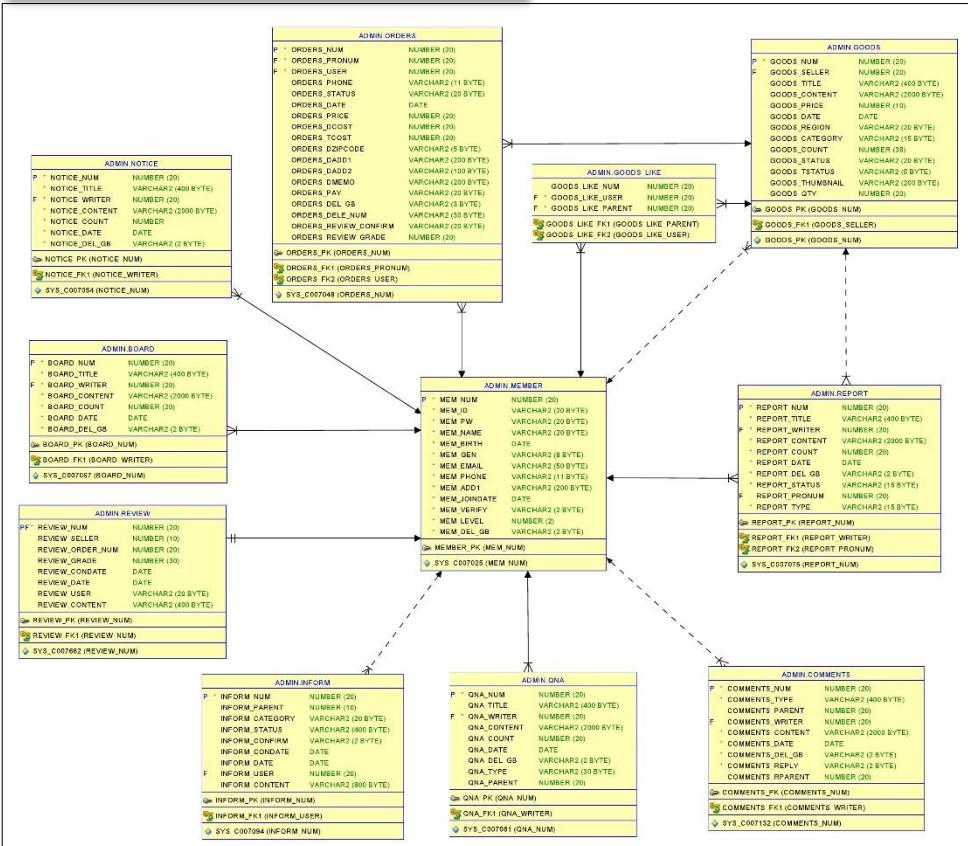
요구사항 명세서

01. SECOND마켓 요구사항명세서					
중분류	상세구분	기능	세부기능	설명	비고
커뮤니티	공지사항	공지사항 상세보기	-	사용자는 공지사항 상세보기를 할 수 있다	-
		(관) 공지사항 작성	-	관리자는 공지사항을 작성할 수 있다	관리자기능
		(관) 공지사항 수정	-	관리자는 공지사항을 수정할 수 있다	관리자기능
		(관) 공지사항 삭제	-	관리자는 공지사항을 삭제할 수 있다	관리자기능
	자유게시판	자유게시판 조회	-	사용자는 자유게시판에 작성된 게시글을 조회할 수 있다	-
		자유게시판 상세보기	-	사용자는 자유게시판에 작성된 게시글을 상세보기 할 수 있다	-
		자유게시판 작성	-	사용자는 자신이 자유게시판에 게시글을 작성할 수 있다.	-
		자유게시판 수정	-	사용자는 자신이 자유게시판에 작성한 게시글을 수정할 수 있다	-
		자유게시판 삭제	-	사용자는 자신이 자유게시판에 작성한 게시글을 삭제할 수 있다	-
	댓글	댓글 쓰기	-	사용자는 상품 상세보기 페이지에서 댓글을 작성할 수 있다	-
		댓글 삭제	-	사용자는 자신이 작성한 댓글을 삭제할 수 있다	-
	QNA	QNA 조회	-	사용자는 Q&A를 조회할 수 있다	-
		내 질문 내역 조회	-	사용자는 자신이 작성한 Q&A를 모아볼 수 있다	-
		QNA 상세보기	-	사용자는 Q&A를 상세보기 할 수 있다	-
		QNA 작성	-	사용자는 Q&A를 작성할 수 있다	-
		QNA 수정	-	사용자는 자신이 작성한 Q&A를 수정할 수 있다	-

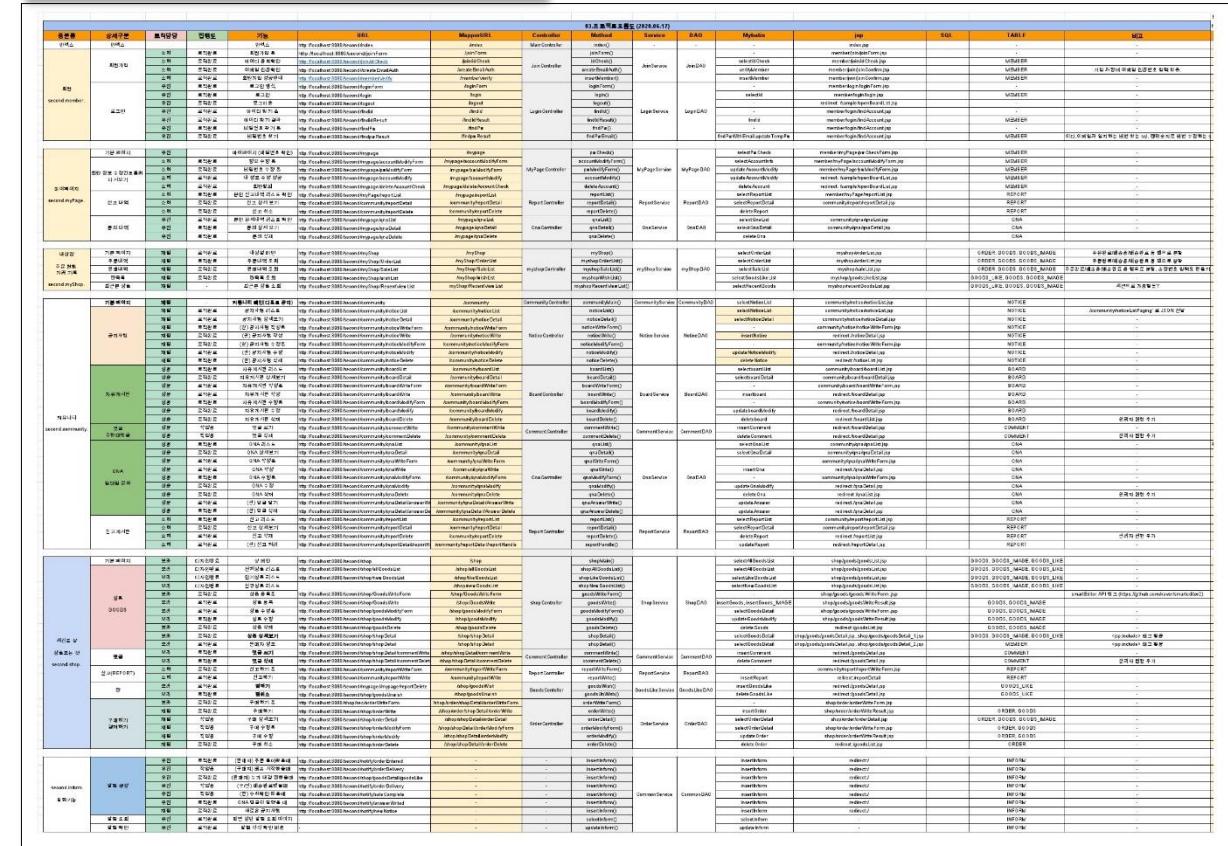
SECOND SHOP		상품 등록	-	사용자는 자신이 판매할 상품을 등록할 수 있다	-
		상품 수정	-	사용자는 자신이 등록한 상품을 수정할 수 있다	-
		상품 삭제	-	사용자는 자신이 등록한 상품을 삭제할 수 있다	-
		상품 상세보기	-	사용자는 등록된 상품을 상세보기할 수 있다	-
		판매자 정보	-	사용자는 판매자의 정보를 볼 수 있다	-
		상품 문의	-	사용자는 등록된 상품에 대해 판매자에게 문의할 수 있다	-
		댓글	-	사용자는 상품 상세보기 페이지에서 댓글을 작성할 수 있다	-
신고		댓글 삭제	-	사용자는 자신이 작성한 댓글을 삭제할 수 있다	-
		신고하기	-	사용자는 판매 등록된 상품에 대하여 신고할 수 있다	-
		찜	-	사용자는 관심 상품에 대하여 찜하기를 할 수 있다	-
		찜하기	-	사용자는 자신이 찜한 상품에 대하여 찜 취소를 할 수 있다	-
		찜취소	-	사용자는 자신이 찜한 상품에 대하여 찜 취소를 할 수 있다	-
		구매정보	-	사용자는 등록된 판매상품에 대하여 구매 신청을 할 수 있다	-
		구매 상세보기	-	사용자는 자신이 구매한 내역에 대하여 상세보기를 할 수 있다	-
내상점		구매 수정	-	사용자는 자신이 구매한 내역에서 배송지 등을 수정할 수 있다	-
		구매 취소	-	사용자는 배송 전인 상품에 대하여 구매 취소를 할 수 있다	-
		기본 페이지	-	사용자는 판매내역, 주문내역 등을 볼 수 있는 내상점에 들어갈 수 있다	-
		주문내역	-	사용자가 주문한 내역을 조회할 수 있다	-
		판매내역	-	사용자가 판매한 내역을 조회할 수 있다	-
		찜목록	-	사용자가 찜한 상품 리스트를 조회할 수 있다	-
		찜목록 조회	-	사용자가 찜한 상품 리스트를 조회할 수 있다	-
알림	알림 주가	(판매자) 주문 들어왔을때	-	판매자 관점에서 자신의 판매내역에 대하여 주문이 들어온 경우 알림을 추가한다	-
		(판매자) 누가 내 걸 품했을때	-	판매자 관점에서 자신의 판매내역에 대하여 다른 사용자가 품했는 경우 알림을 추가한다	-
		QNA 담글	-	자신이 작성한 QNA 게시판에 대한 담글이 작성되는 경우 알림을 추가한다	-
		새로운 공지사항	-	관리자에 의해 새로운 공지사항이 작성되는 경우 알림을 추가한다	-

기획 및 설계 (ERD / 프로젝트 구현 흐름도)

ERD



프로젝트 구현 흐름도





와이어프레임 및 웹 구현화면 (1 / 3)

■ 회원가입 폼 이메일 인증방식 및 AJAX 입력사항 중복, 일치여부 체크

회원가입

아이디: ID중복확인

비밀번호:

비밀번호 확인:

이름:

후다음 변경: [인증번호 전송](#)

인증번호 입력:

[로그인](#) [회원가입](#) [아이디 찾기](#)

SECOND MARKET

회원가입

Join to our community now!

아이디: admin [아이디 중복 확인](#)

사용불가능한 아이디입니다.

비밀번호:

비밀번호 확인:

암호가 일치합니다.

이름: 흥길동

■ 게시판 게시글 작성 시 이미지 첨부가능. 댓글 달기 가능

게시판

로그인 / 로그아웃

판매등록

마이페이지

공지게시판

자유게시판

Q&A

나의 내역

검색

최신글 | 인기글 | 저작글 | 고가글

번호	제목	작성일	작성자
120111	1도 걸어 예전의 느낌	2020-05-19 16:48	beni
120009	이게 뭔가요?	2020-05-19 16:40	purple
120001	딸기라떼 그린데 사이즈 테이크아웃이요	2020-05-19 16:40	purple
120000	고디바 초콜릿 먹고 싶다	2020-05-19 16:40	purple
110094	치킨 vs 피자	2020-05-19 16:40	purple

자유게시판

공지사항 [자유게시판](#) 신고게시판 Q&A게시판

글번호	제목	작성자	작성일	조회수
121	[물건찾아요] 힘癃 헬스기구 파일 분 계신가요?	test8	2020. 7. 10. 오전 2:41:47	1
101	[유머] 관광객 따라하기	test1	2020. 7. 3. 오전 5:50:24	2
81	[물건찾아요] 맥북 프로 2019년도 이후 모델 찾아요~	test1	2020. 7. 3. 오전 2:12:29	5

1

글쓰기



와이어프레임 및 웹 구현화면 (2 / 3)

■ 상품 게시물 리스트

등록순 /가격 / 인기 정렬 및 검색 기능

The screenshot shows a wireframe on the left and its corresponding implementation on the right. The interface includes a header with a search bar (circled in red), login/logout buttons, and navigation links. Below the header is a banner. The main area displays a search result for '검색어' (Search term) with sorting options: 등록순 (Registered first), 가격 (Price), and 인기 순 (Popular first). It also features a '최신순' (Newest first) button. The implementation shows placeholder images for products and their details: 대왕에어팟스피커 팝니다. 음질 좋아요 (30,000원), 1인용 게이밍 책상 팝니다. (80,000원), 몇번 적용 안한 시계 팝니다. (190,000원), and 사오미 전기자전거 팝니다. (399,000원). Each item includes a green '거래 가능' (Tradeable) badge.

■ 상품상세화면

구매하기, 신고하기 및 짐하기 기능

The screenshot shows a wireframe on the left and its corresponding implementation on the right. The implementation features a large cartoon character at the top. The product details include: 제작자 (Creator): 카카오 모델명 (Model name): 키보드 라이언, 출시일 (Release date): 25, 종고상태 (Status): A+ (green heart). The price information is as follows: 판매가 (Retail price): 65,000원, 할인가 (Discounted price): 45,000원, 배송비 (Shipping fee): 2,500원, 합계 (Total): 64,500원. Buttons for 비로구매 (Buy from Biro-gu) and 좋아요 (Like) are present. To the right, there are three images of the keyboard: a front view, a side view, and a top-down view showing its internal structure. Text on the right side includes: 1인용 게이밍 책상 팝니다. (Price: 80,000원, Location: 부산시), #80000 (Price: 80,000원, Location: 부산시), and 버로구매 (Buy from Biro-gu) with a red heart icon.



와이어프레임 및 웹 구현화면 (3 / 3)

■ 결제하기 IAMPORT - API 활용 카카오페이지 결제로직 구현

A composite screenshot of a Korean e-commerce website for outdoor gear. The main page shows a large image of a tent and a search bar for "bestcamp auto6 prime touch". Below it is a payment modal for pay. To the right is a detailed product page for a "BestCamp Auto6 Prime Touch" tent, showing its price (70,000 won), shipping information (Busan Haeundae-gu APEC 30, 210-1001), and a delivery note. At the bottom right is a pay button.

■ 나의 주문내역

주문 및 판매내역 확인 및 수취확인 기능

A Web Page
https://

로그인 / 로그아웃

로그인 / 로그아웃 ★ 즐겨찾기

상점명, 지역명, 배송장번호 등
검색

판매등록 마이페이지

☰ 반개장터 화트너센터 >

배송조회

구매완료 >>> 배송준비중 >>> 배송중 >>> 배송완료

경동택배
문송장번호 12346879846

받는 사람
주소 4조이자만 3조입니다
서울시 종로구 이현동
127-547
배송요청사항
경비실에 맡겨 주세요.

나의 주문내역 나의 판매내역 품 상품목록

주문/결제 배송중 배송완료

주문번호	주문일자	송장번호	주문금액	주문상태	주문상태 변경
1000037920	2020. 7. 8. 오전 6:50:04	[주문완료] 배송 준비 중입니다.	80000	주문/결제	주문취소

프로젝트 주요작업 요약

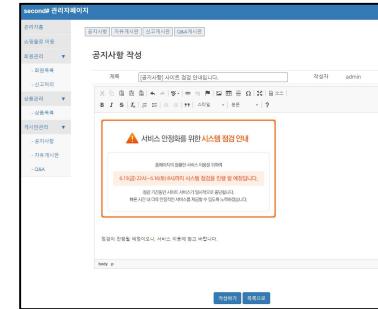
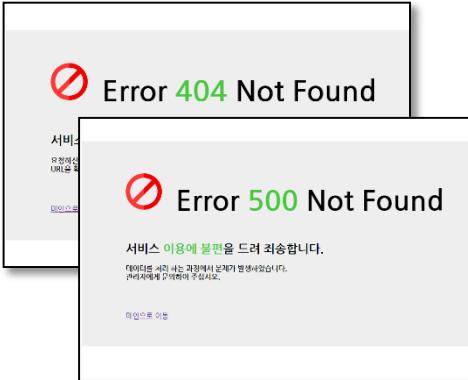
1. SPRING 메이븐 공통 설정(pom.xml / web.xml / log4j)

```

2020-07-10 11:48:48,634 DEBUG [second.common.logger.LoggerInterceptor] ----- START -----
2020-07-10 11:48:48,634 DEBUG [second.common.logger.LoggerInterceptor] ----- second/inform
2020-07-10 11:48:48,634 DEBUG [second.common.logger.LoggerAspect] --- Controller : second.main.controller.MainController.inform()
2020-07-10 11:48:48,635 DEBUG [second.common.logger.LoggerAspect] --- ServiceImpl : second.common.service.InformService.informList()
2020-07-10 11:48:48,635 DEBUG [second.common.logger.LoggerAspect] --- DAO : second.common.dao.InformDAO.informList()
2020-07-10 11:48:48,635 INFO SQL : SELECT
    INFORM_ID,
    INFORM_CONTENT,
    INFORM_DATE,
    INFORM_NUM,
    INFORM_USER
  FROM INFORM
  WHERE INFORM_NUM = (SELECT MEM_NUM FROM MEMBER WHERE MEM_ID = 'test0')
  AND INFORM_CONFIRM = 'N'
  ORDER BY INFORM_DATE DESC
2020-07-10 11:48:48,634 INFO [jdbc.resultsettable] [INFORM_NUM][INFORM_CONTENT][INFORM_CONFIRM][INFORM_DATE][INFORM_USER]
2020-07-10 11:48:48,634 INFO [jdbc.resultsettable] [347] [2020-07-05 10:35:01][57] [2020-07-05 09:10:27:16.0]
2020-07-10 11:48:48,634 INFO [jdbc.resultsettable] [403] [2020-07-10 02:27:16.0][57] [2020-07-10 02:27:16.0]
2020-07-10 11:48:48,634 DEBUG [second.common.logger.LoggerInterceptor] ----- END -----

```

2. 에러(error) 페이지 구현



3. 공지사항 게시판 Logic 및 View 구현

- 3-1 글쓰기|VIEW화면구현
- 3-2 글쓰기|Business Logic
- 3-3 공지사항게재시 알림발송

4. 내상점의 3가지 탭 컨텐츠와 로직구현

- 4-1 내상점 > 나의주문내역
- 4-2 내상점 > 나의판매내역
- 4-3 내상점 > 짐한상품 List Logic 및 View 구현

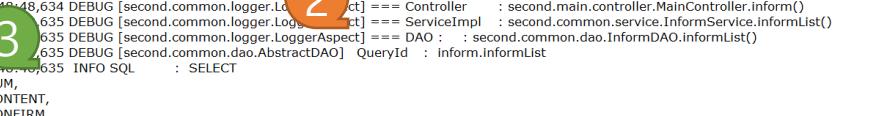
5. 상품주문 시 주문서 입력폼과 Logic 구현

- 5-1 Daum에서 제공하는 우편번호API 적용
- 5-2 IMPORT 결제 API 카카오페이 활용

1. 메이븐 공통 설정 (Interceptor / Aspect / log4j 적용)

- 프로세스 흐름파악 및 원활한 디버깅을 위해 console창 로그 출력설정 적용.

■ AOP 활용 로그 출력



```
2020-07-10 11:48:48,634 DEBUG [second.common.logger.LoggerInterceptor] ===== START =====
2020-07-10 11:48:48,634 DEBUG [second.common.logger.LoggerInterceptor] Request URI : /second/inform
2020-07-10 11:48:48,634 DEBUG [second.common.logger.LoggerInterceptor] Controller : second.main.controller.MainController.inform()
2020-07-10 11:48:48,635 DEBUG [second.common.logger.LoggerInterceptor] ServiceImpl : second.common.service.InformService.informList()
2020-07-10 11:48:48,635 DEBUG [second.common.logger.LoggerAspect] === DAO : second.common.dao.InformDAO.informList()
2020-07-10 11:48:48,635 INFO SQL      : SELECT
INFORM_NUM,
INFORM_CONTENT,
INFORM_CONFIRM,
INFORM_DATE,
INFORM_USER
FROM INFORM
WHERE (INFORM_USER= (SELECT MEM_NUM FROM MEMBER WHERE MEM_ID ='test8')
OR INFORM_USER IS NULL)
AND INFORM_CONFIRM = 'N'
ORDER BY INFORM_NUM
2020-07-10 11:48:48,654 INFO [jdbc.resultsettable] |-----|-----|-----|-----|-----|-----|
2020-07-10 11:48:48,654 INFO [jdbc.resultsettable] |INFORM_NUM |INFORM_CONTENT |INFORM_CONFIRM |INFORM_DATE   |INFORM_USER |
2020-07-10 11:48:48,654 INFO [jdbc.resultsettable] |-----|-----|-----|-----|-----|
2020-07-10 11:48:48,654 INFO [jdbc.resultsettable] |347  |새로운 공지사항이 게시되었습니다. |N          |[2020-07-05 09:10:35.0 |57
2020-07-10 11:48:48,654 INFO [jdbc.resultsettable] |403  |상대방이 내 상품을 좋아합니다. |N          |[2020-07-10 02:27:16.0 |57
2020-07-10 11:48:48,654 INFO [jdbc.resultsettable] |-----|-----|-----|-----|-----|
2020-07-10 11:48:48,654 DEBUG [second.common.logger.LoggerInterceptor] ===== END =====
```

■ 인터셉터 활용 로그출력



```
LoggerInterceptor.java x LoggerAspectJava
1 package second.common.logger;
2
3 import javax.servlet.http.HttpServletRequest;
4
5 public class LoggerInterceptor extends HandlerInterceptorAdapter {
6     protected Log log = LogFactory.getLog(LoggerInterceptor.class);
7
8     @Override
9     public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) {
10         if (log.isDebugEnabled()) {
11             log.debug("==== START =====");
12             log.debug(" Request URI \t: " + request.getRequestURI());
13         }
14         return super.preHandle(request, response, handler);
15     }
16
17     @Override
18     public void postHandle(HttpServletRequest request, HttpServletResponse response, Object handler,
19                           ModelAndView modelAndView) {
20         if (log.isDebugEnabled()) {
21             log.debug("==== END =====");
22         }
23     }
24
25     @Override
26     public void afterCompletion(HttpServletRequest request, HttpServletResponse response, Object handler,
27                                Exception ex) {
28         if (log.isDebugEnabled()) {
29             log.debug("==== =====");
30         }
31     }
32 }
```

```
1 package second.common.logger;
2
3 import org.apache.commons.logging.Log;
4
5 @Aspect
6 public class LoggerAspect {
7     protected Log log = LogFactory.getLog(LoggerAspect.class);
8     static String name="";
9     static String type="";
10
11     @Around("execution(* second..controller.*Controller(..)) or "
12             + "execution(* second..service.*Impl(..)) or execution(* second..dao.*(..))")
13     public Object logPrint(ProceedingJoinPoint joinPoint) throws Throwable{
14         type = joinPoint.getSignature().getDeclaringTypeName();
15
16         if(type.indexOf("Controller")>-1) {
17             name="== Controller \t : ";
18         }
19         else if(type.indexOf("Service")>-1) {
20             name="== ServiceImpl \t : ";
21         }
22         else if(type.indexOf("DAO")>-1) {
23             name="== DAO : \t : ";
24         }
25         log.debug(name+type+" "+joinPoint.getSignature().getName()+"()");
26         return joinPoint.proceed();
27     }
28 }
29
30 }
```

The screenshot shows an IDE interface with three tabs open:

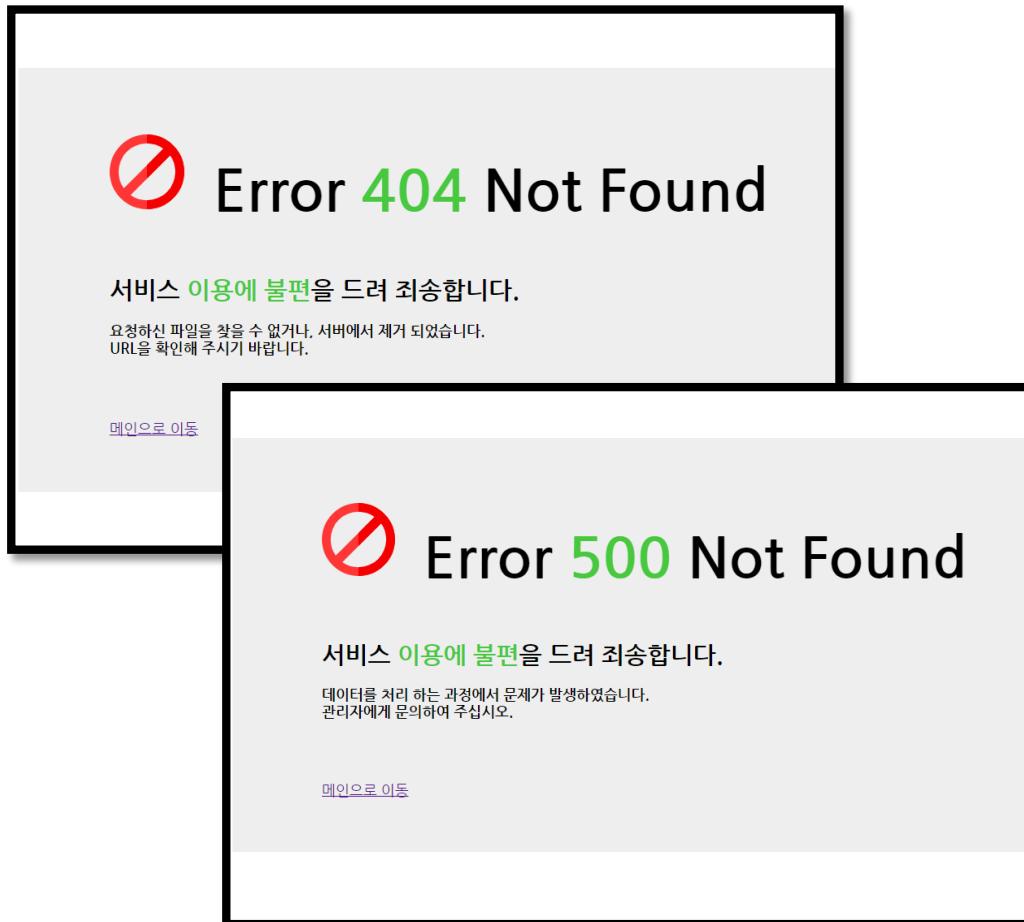
- AbstractDAO.java**: Contains code for printing query IDs.
- CommonDAO.java**: Contains code for printing query IDs.
- LoggerInterceptor.java**: Contains a log4j.xml configuration block.

The **LoggerInterceptor.java** tab is active, displaying the following log4j configuration:

```
<!-- Query Loggers -->
<logger name="jdbc.sqlonly" additivity="false">
    <level value="INFO"/>
    <appender-ref ref="console-infolog"/>
</logger>
<br/>
<logger name="jdbc.resultsettable" additivity="false">
    <level value="INFO"/>
    <appender-ref ref="console"/>
</logger>
```

2. 404 / 500 에러 발생 시 페이지 전달(web.xml 및 Controller 별 적용)

- 404에러, 500에러가 발생했을 시 유저에게 안내메시지가 담긴 View 페이지 전환 로직 구현.

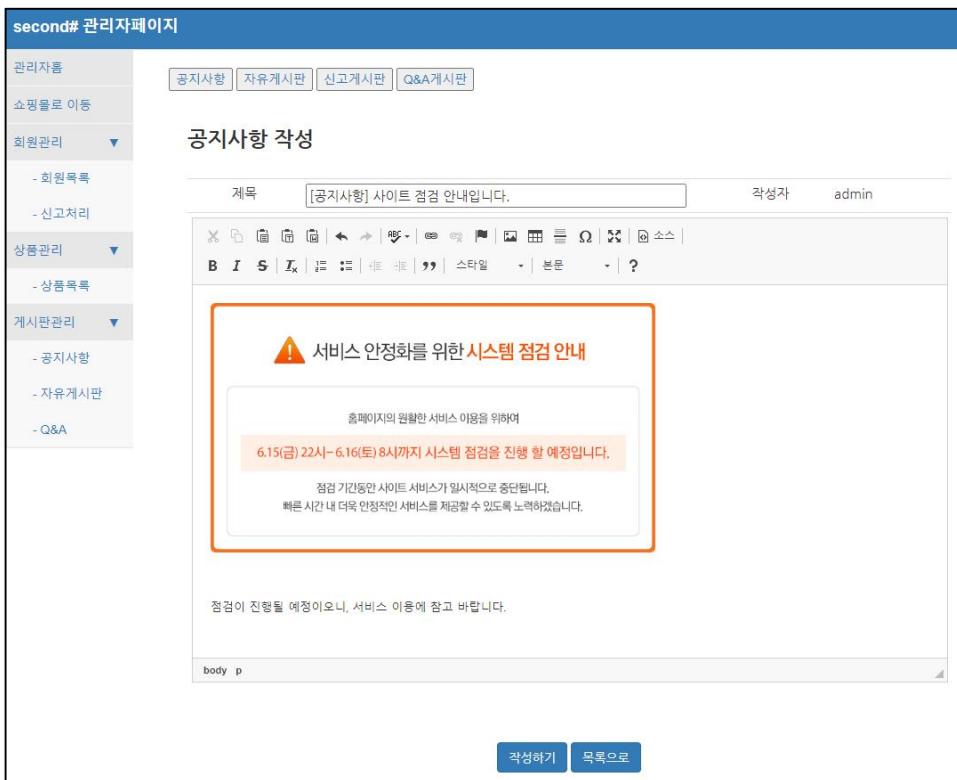


```
web.xml
56
57      <!-- Error Page -->
58      <error-page>
59          <error-code>404</error-code>
60          <location>/WEB-INF/jsp/error/404error.jsp</location>
61      </error-page>
62
63
64
65
```

```
NoticeController.java
1 package second.notice.controller;
2
3 import java.util.List;
4
5 @Controller
6 public class NoticeController {
7     Logger log = Logger.getLogger(this.getClass());
8
9     @ExceptionHandler(RuntimeException.class)
10    @ResponseStatus(value = HttpStatus.UNAUTHORIZED)
11    public String exceptionHandler() {
12        log.debug("NoticeController_예외사항_발생!");
13        return "/error/exception";
14    }
15}
```

3-1. 공지사항 게시판 글쓰기(VIEW 화면 구현)

- CKeditor 를 활용하여 이미지 첨부 가능한 공지사항 게시물 작성.



noticeWriteForm.jsp

```
<script type="text/javascript" src="${pageContext.request.contextPath }/ckeditor/ckeditor.js"></script>
```

```
$function(){
    CKEDITOR.replace('NOTICE_CONTENT',{
        width:'100%',
        height:'600px',
        filebrowserUploadUrl: '${pageContext.request.contextPath }/ckeditor/fileupload'
    });
};
```

```
function fn_insertNotice(){ // 작성하기 누르면 #write 타고와서 등작
    var comSubmit = new ComSubmit("frm");
    comSubmit.setUrl("<c:url value='/community/noticeWrite' />");

    // 게시글 제목 필요
    if($("#NOTICE_TITLE").val() == ''){
        alert("제목을 입력해주세요.");
        $("#NOTICE_TITLE").focus();
        return false;
    }

    // 게시글 내용 필요
    if(CKEDITOR.instances.NOTICE_CONTENT.getData() == '' ||
       CKEDITOR.instances.NOTICE_CONTENT.getData().length == 0){
        alert("내용을 입력해주세요.");
        $("#NOTICE_CONTENT").focus();
        return false;
    }
    // 커스텀 구성 Form 전달 메서드
    comSubmit.submit();
}
```

3-2. 공지사항 게시판 로직 및 게시물 등록 알림 구현 (Controller–Service–DAO–DB로직)

- 공지사항 [작성하기] 버튼 클릭 후, 리퀘스트매핑에 따라 DB까지 Insert 전달 흐름.

NoticeController.java

```
1 // 공지사항 작성
@RequestMapping(value = "/community/noticeWrite")
public ModelAndView noticeWrite(CommandMap commandMap, HttpServletRequest request) throws Exception {
    ModelAndView mv = new ModelAndView("redirect:/community/noticeList");
    noticeService.insertNoticeWrite(commandMap.getMap(), request);
    return mv;
}
```

NoticeServiceImpl.java

```
2 @Override
public void insertNoticeWrite(Map<String, Object> map, HttpServletRequest request)
    throws Exception {
    noticeDAO.insertNotice(map);
    map.put("IDX", map.get("NOTICE_NUM"));

    /*전체유저 뽑아내고 회원수만큼 공지사항 알림 전달하기*/
    List<Map<String, Object>> list = informDAO.selectAllMember(map);

    for(int i=0, size=list.size(); i<size; i++) {
        map.put("IDX", list.get(i).get("MEM_NUM"));
        informDAO.informInsertNotice(map, "새로운 공지사항이 게시되었습니다."); // *회원에게 게재알림 전달
    }
}
```

NoticeDAO.java

```
3 // 공지사항 작성
public void insertNotice(Map<String, Object> map) throws Exception{
    insert("notice.insertNotice", map);
}
```

Notice_SQL.xml

```
4 <!-- 공지사항 작성 -->
<mapper namespace="notice">

<insert id="insertNotice" parameterType="hashmap" useGeneratedKeys="true" keyProperty="NOTICE_NUM">
    <selectKey keyProperty="NOTICE_NUM" resultType="string" order="BEFORE">
        SELECT NOTICE_SEQ.NEXTVAL FROM DUAL
    </selectKey>
    <![CDATA[

        INSERT
        INTO    NOTICE  (
                        NOTICE_NUM,
                        NOTICE_TITLE,
                        NOTICE_WRITER,
                        NOTICE_CONTENT
                    )
        VALUES
        (
            #{NOTICE_NUM},
            #{NOTICE_TITLE},
            #{MEM_NUM},
            #{NOTICE_CONTENT}
        )

    ]]>
</insert>
```

3-3. 공지사항 게시판 로직 및 게시물 등록 알림 구현

- 공지사항 게재 시 전체유저에게 알림 전달 흐름.



NoticeServiceImpl.java

```

@Override
public void insertNoticeWrite(Map<String, Object> map, HttpServletRequest request)
throws Exception {

    noticeDAO.insertNotice(map);
    map.put("IDX", map.get("NOTICE_NUM"));

    /*전체유저 뽑아내고 회원수만큼 공지사항 알림 전달하기*/
    List<Map<String, Object>> list = informDAO.selectAllMember(map);

    for(int i=0, size=list.size(); i<size; i++) {
        map.put("IDX", list.get(i).get("MEM_NUM"));
        informDAO.informInsertNotice(map, "새로운 공지사항이 게시되었습니다."); // *회원에게 게재알림 전달
    }
}

```

1

InformDAO.java

```

2 //공지사항 작성 시 알림 DB 입력
public void informInsertNotice(Map<String, Object> map, String str) throws Exception{
    map.put("INFORM_CONTENT", str);
    insert("inform.informinsertNotice",map);
}

```

2

Inform_SQL.xml

```

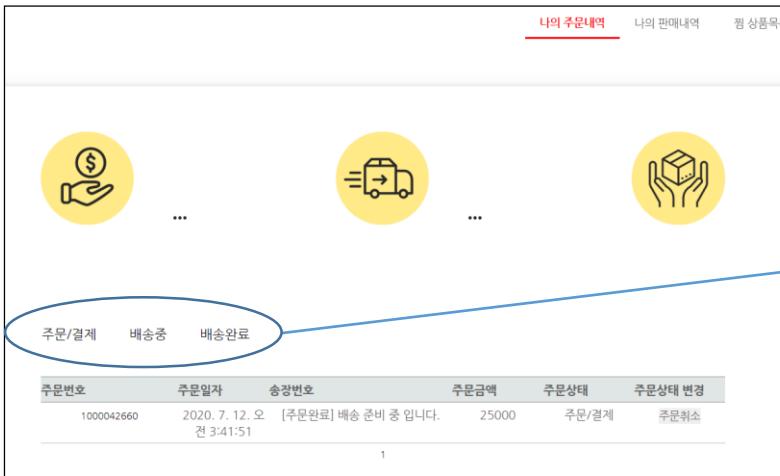
<!-- 공지사항 알림 -->
<mapper namespace="inform">
<insert id="informinsertNotice" parameterType="hashmap">
    INSERT INTO INFORM
        (INFORM_NUM,
        INFORM_PARENT,
        INFORM_CONTENT,
        INFORM_CONFIRM,
        INFORM_DATE,
        INFORM_USER
    )
    VALUES(
        INFORM_SEQ.NEXTVAL,
        #{NOTICE_NUM},
        #{INFORM_CONTENT},
        'N',
        SYSDATE,
        #{IDX}
    )
</insert>

```

3

4-1. 내상점 > 나의 주문(Order)내역

- 구매(결제) 후, 탭 클릭 시 거래상태에 맞는 '배송준비중 · 배송중 · 배송완료' 리스트가 생성 됨.



1

1 Json 을 활용해 비동기 탭 전환을 구현

2

2 조건에 해당하는 서비스 - DAO - SQL
로직을 처리하고 list 객체에 적용함.

MyshopController.java

```

47 }
48
49 @RequestMapping(value="/myshop/selectOrderList")
50 public ModelAndView selectOrderList(CommandMap commandMap, HttpServletRequest request,
51                                     @RequestParam(value = "tabNo", defaultValue="") String tabNo) throws Exception {
52
53     ModelAndView mv = new ModelAndView("jsonView");
54     HttpSession session = request.getSession();
55     commandMap.put("MEM_ID", session.getAttribute("session_MEM_ID"));
56     List<Map<String, Object>> list;
57
58     if(tabNo.equals("1")){
59         list = myshopService.selectMyOrderList1(commandMap.getMap());
60     }else if(tabNo.equals("2")){
61         list = myshopService.selectMyOrderList2(commandMap.getMap());
62     }else if(tabNo.equals("3")){
63         list = myshopService.selectMyOrderList3(commandMap.getMap());
64     }else{
65         list = myshopService.selectMyOrderList1(commandMap.getMap());
66
67     }
68     mv.addObject("list", list);
69     mv.addObject("tabNo", tabNo);
70     System.out.println(tabNo);
71
72     if(list.size() > 0){
73         mv.addObject("TOTAL", list.get(0).get("TOTAL_COUNT"));
74     } else{
75         mv.addObject("TOTAL", 0);
76     }
77
78     return mv;
79 }

```

MyshopService.java

```

public interface MyshopService {
    public List<Map<String, Object>> selectMyOrderList1(Map<String, Object> map) throws Exception;
    public List<Map<String, Object>> selectMyOrderList2(Map<String, Object> map) throws Exception;
    public List<Map<String, Object>> selectMyOrderList3(Map<String, Object> map) throws Exception;
}

```

4-1. 내상점 > 나의 주문(Order)내역

- Service - DAO - SQL 로 진행되는 상세 코드.

MyshopServiceImp.java

```
@Service("myshopService")
public class MyshopServiceImp implements MyshopService{

    @Resource(name="myshopDAO")
    private MyshopDAO myshopDAO;

    @Override
    public List<Map<String, Object>> selectMyOrderList1(Map<String, Object> map) throws Exception {
        return myshopDAO.selectMyOrderList1(map);
    }

    @Override
    public List<Map<String, Object>> selectMyOrderList2(Map<String, Object> map) throws Exception {
        return myshopDAO.selectMyOrderList2(map);
    }

    @Override
    public List<Map<String, Object>> selectMyOrderList3(Map<String, Object> map) throws Exception {
        return myshopDAO.selectMyOrderList3(map);
    }
}
```

MyshopDAO.java

```
@Repository("myshopDAO")
public class MyshopDAO extends AbstractDAO {

    @SuppressWarnings("unchecked")
    public List<Map<String, Object>> selectMyOrderList1(Map<String, Object> map) throws Exception {
        return (List<Map<String, Object>>) selectPagingList("myshop.selectMyOrderList1", map);
    }

    @SuppressWarnings("unchecked")
    public List<Map<String, Object>> selectMyOrderList2(Map<String, Object> map) throws Exception {
        return (List<Map<String, Object>>) selectPagingList("myshop.selectMyOrderList2", map);
    }

    @SuppressWarnings("unchecked")
    public List<Map<String, Object>> selectMyOrderList3(Map<String, Object> map) throws Exception {
        return (List<Map<String, Object>>) selectPagingList("myshop.selectMyOrderList3", map);
    }
}
```

MyShop_SQL.xml

```
<!-- 내 주문내역 조회 -->
<select id="selectMyOrderList1" parameterType="hashMap" resultType="hashmap">
<include refid="common.pagingPre"/>
<![CDATA[
    SELECT ROW_NUMBER() OVER (ORDER BY O.ORDERS_NUM DESC) RNUM,
    O.ORDERS_NUM,
    O.ORDERS_PRNUM,
    G.GOODS_TITLE,
    O.ORDERS_STATUS,
    O.ORDERS_PRICE,
    O.ORDERS_DCOST,
    O.ORDERS_TCOST,
    O.ORDERS_DATE,
    NVL(O.ORDERS_DELETE_NUM, '[주문완료] 배송 준비 중입니다.') AS ORDERS_DELETE_NUM,
    TO_CHAR(O.ORDERS_DATE, 'hh24:mi:ss') AS ORDERS_TIME
    FROM ORDERS O,
    GOODS G
    WHERE O.ORDERS_USER = (SELECT MEM_NUM FROM MEMBER WHERE MEM_ID = #{MEM_ID})
    AND O.ORDERS_PRNUM = G.GOODS_NUM
    AND O.ORDERS_DEL_GB = 'N'
    AND O.ORDERS_STATUS = '주문/결제'
    ORDER BY O.ORDERS_DATE DESC
]]>
<include refid="common.pagingPost"/>
</select>
```

SelectMyOrderList1

```
    AND O.ORDERS_DEL_GB = 'N'
    AND O.ORDERS_STATUS = '배송중'
    ORDER BY O.ORDERS_DATE DESC
]]>
<include refid="common.pagingPost"/>
</select>
```

SelectMyOrderList2

```
    AND O.ORDERS_DEL_GB = 'N'
    AND O.ORDERS_STATUS = '거래완료'
    ORDER BY O.ORDERS_DATE DESC
]]>
<include refid="common.pagingPost"/>
</select>
```

SelectMyOrderList3

4-2. 내상점 > 나의 판매(Sale)내역

- 판매 진행중인 물품과 배송중 · 판매완료 리스트가 생성 됨.



1

1 Json 을 활용해 비동기 탭 전환을 구현

2

2 조건에 해당하는 서비스 - DAO - SQL
로직을 처리하고 list 객체에 적용함.

MyshopController.java

```

@RequestMapping(value="/myshop/selectSaleList")
public ModelAndView myshopSaleList(CommandMap commandMap, HttpServletRequest request,
        @RequestParam(value = "tabNo", defaultValue "") String tabNo) throws Exception{
    ModelAndView mv = new ModelAndView("jsonView");
    HttpSession session = request.getSession();
    commandMap.put("MEM_ID", session.getAttribute("session_MEM_ID"));
    String filePath_temp = request.getSession().getServletContext().getRealPath("") + "/file/";
    mv.addObject("path", filePath_temp);
    request.setAttribute("path", filePath_temp);

    List<Map<String, Object>> list;

    if(tabNo.equals("1")){
        list = myshopService.selectMySaleList1(commandMap.getMap());
    }else if(tabNo.equals("2")){
        list = myshopService.selectMySaleList2(commandMap.getMap());
    }else if(tabNo.equals("3")){
        list = myshopService.selectMySaleList3(commandMap.getMap());
    }else{
        list = myshopService.selectMySaleList1(commandMap.getMap());
    }
    mv.addObject("list",list);
    mv.addObject("tabNo", tabNo);
    System.out.println(tabNo);

    if(list.size() > 0){
        mv.addObject("TOTAL", list.get(0).get("TOTAL_COUNT"));
    } else{
        mv.addObject("TOTAL", 0);
    }
    return mv;
}

```

MyshopService.java

```

public List<Map<String, Object>> selectMySaleList1(Map<String, Object> map) throws Exception;
public List<Map<String, Object>> selectMySaleList2(Map<String, Object> map) throws Exception;
public List<Map<String, Object>> selectMySaleList3(Map<String, Object> map) throws Exception;

```

4-2. 내상점 > 나의 판매(Sale)내역

- Service - DAO - SQL로 진행되는 상세 코드.

MyshopServiceImp.java

```
@Override
public List<Map<String, Object>> selectMySaleList1(Map<String, Object> map) throws Exception {
    return myshopDAO.selectMySaleList1(map);
}
@Override
public List<Map<String, Object>> selectMySaleList2(Map<String, Object> map) throws Exception {
    return myshopDAO.selectMySaleList2(map);
}
@Override
public List<Map<String, Object>> selectMySaleList3(Map<String, Object> map) throws Exception {
    return myshopDAO.selectMySaleList3(map);
}
```

MyshopDAO.java

```
@SuppressWarnings("unchecked")
public List<Map<String, Object>> selectMySaleList1(Map<String, Object> map) throws Exception {
    // TODO Auto-generated method stub
    return (List<Map<String, Object>>) selectPagingList("myshop.selectMySaleList1", map);
}

@SuppressWarnings("unchecked")
public List<Map<String, Object>> selectMySaleList2(Map<String, Object> map) throws Exception {
    // TODO Auto-generated method stub
    return (List<Map<String, Object>>) selectPagingList("myshop.selectMySaleList2", map);
}

@SuppressWarnings("unchecked")
public List<Map<String, Object>> selectMySaleList3(Map<String, Object> map) throws Exception {
    // TODO Auto-generated method stub
    return (List<Map<String, Object>>) selectPagingList("myshop.selectMySaleList3", map);
}
```

MyShop_SQL.xml

```
<!-- 내 판매 전체내역 조회 tab1 전체판매상품에 활용-->
<select id="selectMySaleList1" parameterType="hashMap" resultType="hashmap">
<include refid="common.pagingPre"/>
<![CDATA[
    SELECT  ROW_NUMBER() OVER (ORDER BY G.GOODS_NUM DESC) RNUM,
            G.GOODS_NUM,
            O.ORDERS_NUM,
            G.GOODS_TITLE,
            G.GOODS_PRICE,
            G.GOODS_DATE,
            TO_CHAR(G.GOODS_DATE,'hh24:mi:ss') AS GOODS_TIME,
            O.ORDERS_DATE,
            NVL(O.ORDERS_STATUS, '판매중(미결제)' ) AS ORDERS_STATUS,
            G.GOODS_THUMBNAIL,
            G.GOODS_STATUS,
            O.ORDERS_DELETE_NUM,
            G.GOODS_TSTATUS
        FROM    GOODS G,
        (
            SELECT  MEM_NUM, MEM_ID
            FROM    MEMBER
        ) M,
        ORDERS O
       WHERE   G.GOODS_SELLER = M.MEM_NUM
       AND     G.GOODS_STATUS = 'N'
       AND     G.GOODS_NUM = O.ORDERS_PRNUM(+)
       AND     M.MEM_ID = #{MEM_ID}
       AND     G.GOODS_TSTATUS IN ('N', 'ING', 'END')
       ORDER BY G.GOODS_DATE DESC
]]>
<include refid="common.pagingPost"/>
</select>
```

SelectMySaleList1

```
FROM    ORDERS O,
        GOODS G
       WHERE   G.GOODS_SELLER = (SELECT MEM_NUM FROM MEMBER WHERE MEM_ID = #{MEM_ID})
       AND     O.ORDERS_PRNUM = G.GOODS_NUM
       AND     O.ORDERS_DEL_GB = 'N'
       AND     O.ORDERS_STATUS = '배송중'
       ORDER BY O.ORDERS_DATE DESC
]]>
<include refid="common.pagingPost"/>
</select>
```

SelectMySaleList2

```
AND     O.ORDERS_DEL_GB = 'N'
AND     O.ORDERS_STATUS = '거래완료'
```

SelectMySaleList3

4-3. 내상점>찜상품목록 List (Logic 및 View 구현)

- 물품 상세페이지에서 찜하기(♥)를 활성화 한 물품들이 리스트화 되어 보여짐.



- 1 찜 상품목록 클릭 시, View로 연결하는 컨트롤러가 동작함.
- 2 View 내에 리스트를 불러오는 스크립트를 구성함
- 3 순차적으로 콜백이 진행 된 후 찜한 리스트가 출력됨

MyshopController.java

```

@RequestMapping(value="/myshop/goodsLikeList")
public ModelAndView myshopGoodsLikeList(CommandMap commandMap, HttpServletRequest request) throws Exception{
    ModelAndView mv = new ModelAndView("goodsLikeList");
    String filePath_temp = request.getContextPath() + "/file/";
    mv.addObject("path", filePath_temp);
    request.setAttribute("path", filePath_temp);

    return mv;
}

@RequestMapping(value="/myshop/selectGoodsList")
public ModelAndView selectGoodsList(CommandMap commandMap, HttpServletRequest request) throws Exception {
    ModelAndView mv = new ModelAndView("jsonView");
    HttpSession session = request.getSession();
    commandMap.put("MEM_ID", session.getAttribute("session_MEM_ID"));

    List<Map<String, Object>> list = myshopService.selectLikeList(commandMap.getMap());
    mv.addObject("list", list);
    if(list.size() > 0){
        mv.addObject("TOTAL", list.get(0).get("TOTAL_COUNT"));
    } else{
        mv.addObject("TOTAL", 0);
    }
    return mv;
}

```

다음 슬라이드에서
상세 기술

goodsLikeList.jsp (View)

```

<script type="text/javascript">
$(document).ready(function() {
    fn_selectGoodsList(1);
});
function fn_selectGoodsList(pageNo) { 상세코드 생략 }
function fn_selectGoodsListCallback(data) { 상세코드 생략 }

```

4-3. 내상점>찜(Like)상품 목록 List (Logic 및 View 구현)

- Service - DAO - SQL로 진행되는 상세 코드.

MyshopController.java

```
@RequestMapping(value="/myshop/selectGoodsList")
public ModelAndView selectGoodsList(CommandMap commandMap, HttpServletRequest request) throws Exception {
    ModelAndView mv = new ModelAndView("jsonView");
    HttpSession session = request.getSession();
    commandMap.put("MEM_ID", session.getAttribute("session_MEM_ID"));

    List<Map<String, Object>> list = myshopService.selectLikeList(commandMap.getMap());
    mv.addObject("list", list);
    if(list.size() > 0){
        mv.addObject("TOTAL", list.get(0).get("TOTAL_COUNT"));
    } else{
        mv.addObject("TOTAL", 0);
    }
    return mv;
}
```

MyshopServiceImp.java

```
@Override
public List<Map<String, Object>> selectLikeList(Map<String, Object> map) throws Exception {
    return myshopDAO.selectLikeList(map);
}
```

MyshopDAO.java

```
@SuppressWarnings("unchecked")
public List<Map<String, Object>> selectLikeList(Map<String, Object> map) throws Exception {
    return (List<Map<String, Object>>) selectList("myshop.selectLikeList", map);
}
```

MyShop_SQL.xml

```
<!-- 내 찜목록 조회 -->
<select id="selectLikeList" parameterType="hashMap" resultType="hashmap">
<![CDATA[
    SELECT
        ROW_NUMBER() OVER (ORDER BY G.GOODS_NUM DESC) RNUM,
        G.GOODS_NUM,
        G.GOODS_TITLE,
        G.GOODS_PRICE,
        (select MEM_ID FROM MEMBER WHERE MEM_NUM= G.GOODS_SELLER) SELLER_ID,
        G.GOODS_REGION,
        G.GOODS_CATEGORY,
        G.GOODS_COUNT,
        G.GOODS_DATE,
        TO_CHAR(G.GOODS_DATE,'hh24:mi:ss') AS GOODS_TIME,
        G.GOODS_STATUS,
        G.GOODS_THUMBNAIL,
        G.GOODS_TSTATUS,
        G.GOODS_CONTENT
    FROM
        GOODS G,
        (
            SELECT MEM_NUM, MEM_ID
            FROM MEMBER
        ) M,
        (
            SELECT distinct GOODS_LIKE_PARENT,
            GOODS_LIKE_USER
            FROM GOODS_LIKE
        ) GL
    WHERE
        GL.GOODS_LIKE_USER = M.MEM_NUM(+)
        AND G.GOODS_NUM = GL.GOODS_LIKE_PARENT(+)
        AND G.GOODS_STATUS = 'N'
        AND G.GOODS_TSTATUS IN ('N', 'ING', 'END')
        AND M.MEM_ID = #{MEM_ID}
]]>
</select>
```

5-1. 상품주문 입력 로직 구현 (Daum에서 제공하는 우편번호 API 적용)

- 배송지 검색 및 기입 로직이 수행되는 페이지

주문/결제

상품정보	고양이 담는 용기(장난감) 팁니다.
주문시간	05:32:57
주문번호	201
최종 결제금액	25000
이름	김철수
휴대전화	01033334444
배송주소	우편번호 주소(우편번호 찾기로 진행해주세요) 상세 주소입력

1

우편번호 찾기



2



orderWriteForm.jsp

```

12 </style>
13 <script src="https://t1.daumcdn.net/mapjsapi/bundle/postcode/prod/postcode.v2.js"></script>

79<tr style="border-bottom: 1px solid #dfdfdf;">
80  <th>배송주소</th>
81<td style="padding: 5px;padding-left: 20px;">
82    <input type="text" id="MEM_ZIP" name="MEM_ZIP" placeholder="우편번호" readonly>
83    <input type="button" id="searchAddr" name="searchAddr" onclick="zipcode()" value="우편번호 찾기"><br/>
84    <input type="text" id="ADD1" name="ADD1" placeholder="주소(우편번호 찾기로 진행해주세요)" readonly class="text_type"><br/>
85    <input type="text" id="ADD2" name="ADD2" placeholder="상세 주소입력" class="text_type">
86  </td>
87 </tr>

```

1

```

function zipcode() { //우편번호 검색창
  new daum.Postcode({
    oncomplete : function(data) {
      // 팝업에서 검색결과 항목을 클릭했을때 실행할 코드를 작성하는 부분.
      var addr = ''; // 주소 변수
      popupName: 'postcodePopup' // 팝업창 여려개 생성방지
      // 사용자가 선택한 주소 타입에 따라 해당 주소 값을 가져온다.
      if (data.userSelectedType === 'R') { // 사용자가 도로명 주소를 선택했을 경우
        addr = data.roadAddress;
        // 우편번호와 주소 정보를 해당 필드에 넣는다.
        document.getElementById('MEM_ZIP').value = data.zonecode;
        document.getElementById("ADD1").value = addr;
        // 커서를 상세주소 필드로 이동한다.
        document.getElementById("ADD2").focus();
        // 사용자가 지번 주소를 선택했을 경우(J)
      } else if (data.userSelectedType === 'J') {
        alert("도로명 주소를 입력해주세요.");
        return false;
      }
    }
  }).open();
}

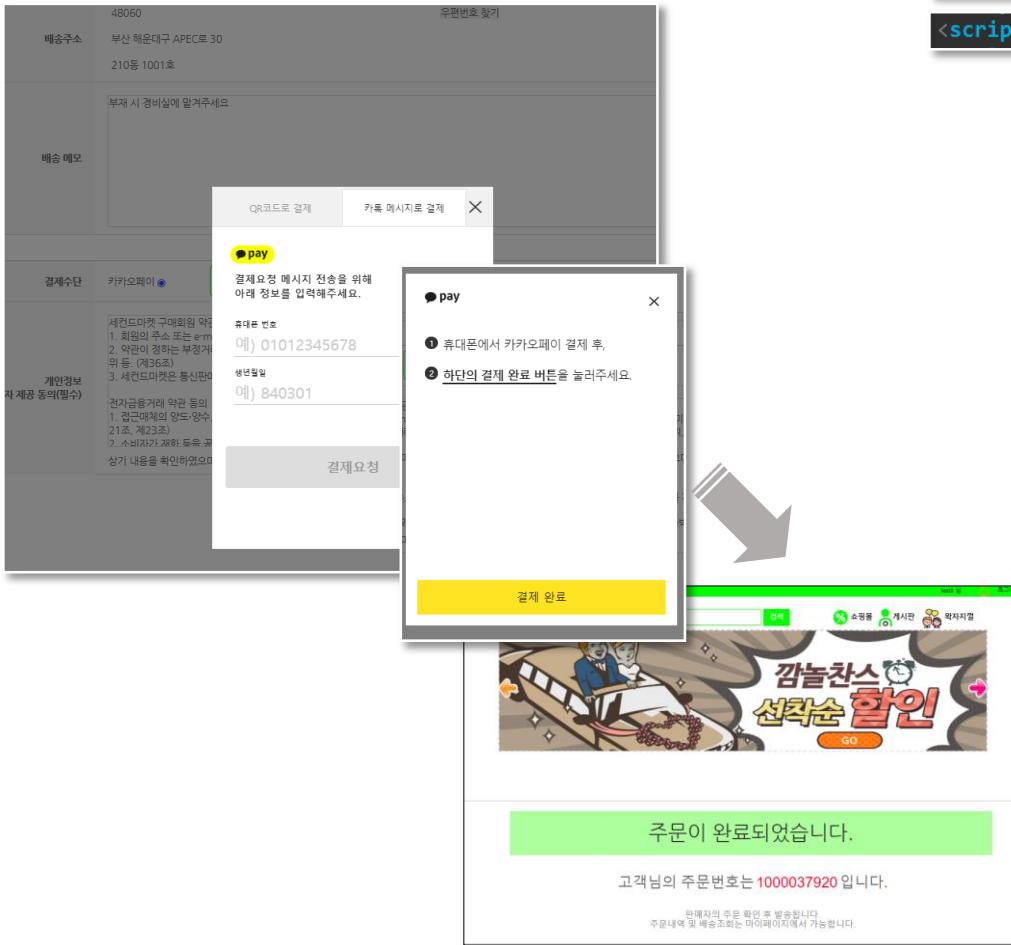
```

2

↳ 우편번호는 Daum에서 제공하는 API 활용

5-2. 상품주문 입력 로직 구현 (결제처리)

- 동의체크 후 주문진행 시 결제진행을 위한 팝업 등장



orderWriteForm.jsp

└ IAMPORT 결제API (카카오페이지) 활용

```
<script type="text/javascript" src="https://cdn.iamport.kr/js/iamport.payment-1.1.5.js"></script>
```

```
function popup(){ //카카오페이지 결제 동작
    var IMP = window.IMP; // 생략가능
    IMP.init('imp07872997'); // 'iamport' 대신 부여받은 "가맹점 식별코드"를 사용

    IMP.request_pay({
        pg : 'kakao',
        pay_method : 'card',
        merchant_uid : 'merchant_' + new Date().getTime(),
        name : '주문명:결제테스트',
        amount : Number(document.getElementById("GOODS_TCOST").value), //결제 총금액
        buyer_email : 'iamport@siot.do',
        buyer_name : $("#MEM_ID").val(),
        buyer_tel : $("#MEM_PHONE").val(),
        buyer_addr : $("#ADD1").val()+$("#ADD2").val(),
        buyer_postcode : '123-456',
        m_redirect_url : '/second/payEnd.action' // 결제 완료 후 보낼 컨트롤러의 메소드명
    }, function(rsp) {
        if ( rsp.success ) { // 성공시
            var msg = '결제가 완료되었습니다.';
            msg += '고유ID : ' + rsp.imp_uid;
            msg += '상점 거래ID : ' + rsp.merchant_uid;
            msg += '결제 금액 : ' + rsp.paid_amount;
            msg += '카드 승인번호 : ' + rsp.apply_num;
            document.getElementById("check").value = "true"
        } else { // 실패시
            var msg = '결제에 실패하였습니다.';
            msg += '에러내용 : ' + rsp.error_msg;
        }
        alert(msg);
    });
};
```

포트폴리오 소개

The screenshot shows a web-based marketplace interface. At the top, there's a navigation bar with a user icon, a search bar, and various menu items in Korean. A large, colorful promotional banner for "깜놀찬스 선착순 할인" (Surprise Giveaway First Come, First Served Discount) is displayed prominently. Below the banner, there are two rows of product cards. Each card includes a small image, a title, a price, and a seller name. The products listed include a cat in a box, a desk setup, a watch, a folding scooter, a laptop, a cat in a hat, and a dog wearing a jacket. At the bottom of the page, there's a footer section with links for "커뮤니티", "쇼핑몰정보", "고객센터", and a detailed disclaimer about the company's role.

SECOND MARKET

Search

쇼핑몰 계시판 혁자지별

등록순 가격높은순 인기순

고양이 담는 용기(장난
김) 팝니다.
25,000원 판매자: test7

대왕에어팟스피커 팝니다.
다. 음질 좋아요
30,000원 판매자: test8

1인용 게이밍 책상 팝니다.
80,000원 판매자: test7

몇번 착용 안한 시계 팝니다.
190,000원 판매자: test7

샤오미 전기자전거 팔아요
399,000원 판매자: test7

맥북 12인치 팝니다.
650,000원 판매자: test1

고양이 방석 팝니다.
30,000원 판매자: test1

개간지 자켓, 모자 급처
1,000,000원 판매자: test2

상품명 Search 검색

글쓰기

커뮤니티
공지사항
자유게시판
QnA게시판
친구게시판

쇼핑몰정보
운영정책
이용약관
개인정보취급방침

고객센터
010-5780-3412
평일 오전09:00 ~ 오후06:00
점장시간 12:30~13:30
로, 휴일 유무

SECOND MARKET
second샵은 통신판매중개자로서 중고거
래마켓 second샵의 거래 당사자가 아니
다.
임점판매가 등록한 상품정보 및 거래에
대해 책임을 지지 않습니다.

Thank You :)