# 한국스마트정보교육원
# 51기 3팀 프로젝트 완료 보고서

| 프로젝트명 |
| --- |
| 반려동물건강관련플랫폼 : PAL(Pet All Life) |

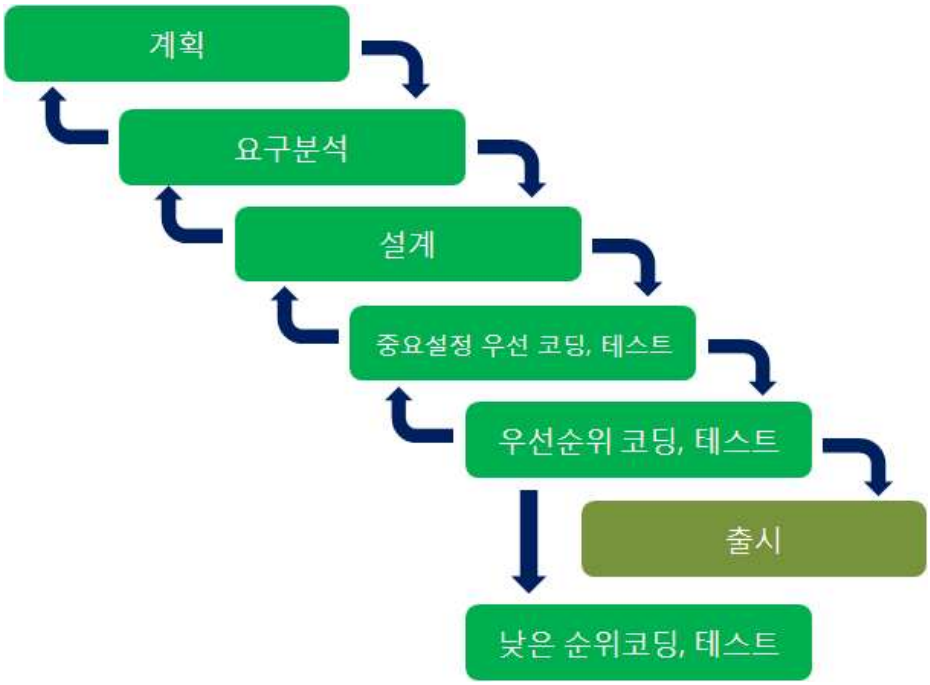
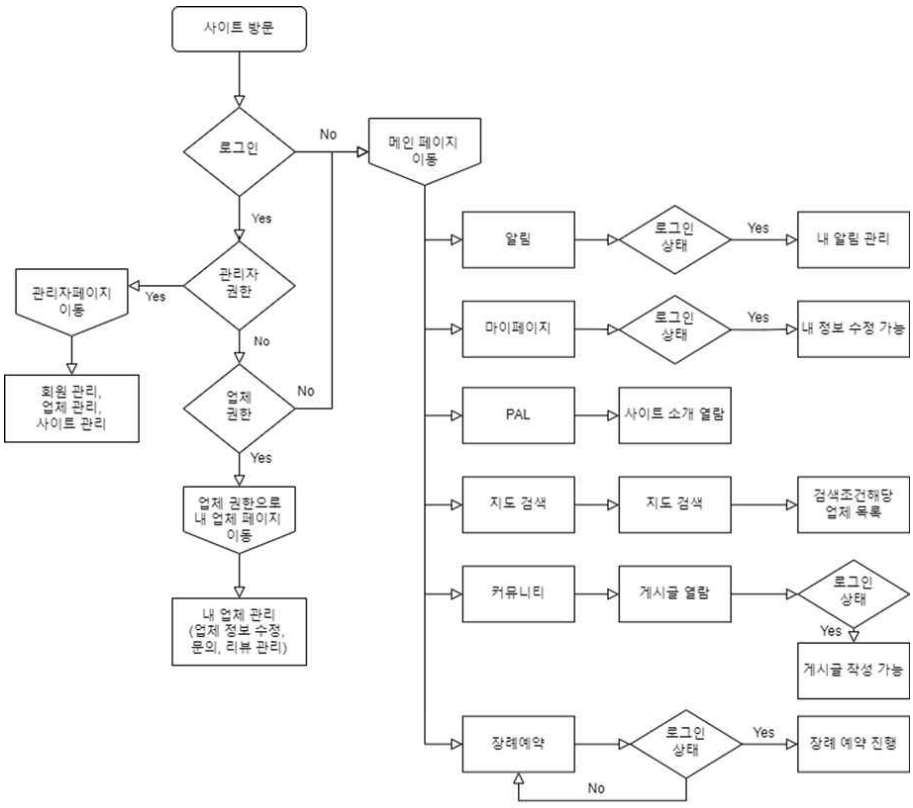
| | |
| --- | --- |
| 팀장 | 송재익 |
| 팀 원 | 박은수 |
| 팀 원 | 한송이 |
| 작성일 | 2024.07.29 |

| | |
| --- | --- |
| 호스팅 | https://petalllife.duckdns.org/ |
| GitHub | https://github.com/EunsooPakr/ks51team03 |
| Notion | https://assorted-hourglass-bdnotion.site/Pet-All-Life-PAL-f4855b1a5a74ba28ab56be16557b60 |

# INDEX

## 【 1. 프로젝트 기본 정보 】

<table>
<tr><td>프로젝트명</td><td colspan="2">반려동물건강관련플랫폼 : PAL(Pet All Life)</td></tr>
<tr><td rowspan="7">개발기간</td><td colspan="2">■ 2024.04.16 ~2024.07.29 (104일)</td></tr>
<tr><td>2024.04.16 ~ 2024.05.08 (22일)</td><td>- 여러 시각에서 프로젝트 주제 선정<br>- 컨퍼런스, 기능 구현 등 자료 조사<br>- 개인 기능 역량 파악</td></tr>
<tr><td>2024.05.08 ~ 2024.06.14 (38일)</td><td>- 시스템 구조도, 초안 작성<br>- 설계 자료 취합, 수정 및 보완<br>- DB 및 ERD 구조 수정</td></tr>
<tr><td>2024.06.14 ~ 2024.06.26 (12일)</td><td>- 시스템 서버 설정<br>- 개발 환경 설정<br>- 부트스트랩을 이용한 화면 구현</td></tr>
<tr><td>2024.06.26 ~2024.07.24 (34일)</td><td>- 기능 구현 (개인 분담)<br>- nginx, 파일지라, 도메인 설정</td></tr>
<tr><td>2024.07.24 ~ 2024.07.26(3일)</td><td>- 테스트, 오류 수정 및 보완<br>- 미완성, 기능 구현</td></tr>
<tr><td>2024.07.26 ~ 2024.07.29(4일)</td><td>- 완료보고서 작성</td></tr>
<tr><td>개발목적</td><td colspan="2">* 플랫폼 정보 부족 및 사랑과 정성으로 키운 반려동물에 대한 서비스 집약체<br><br>21세기, 출산율 0인 현재를 살아가는 모든 반려인 및 반려예정인을 위한, 가족처럼 사랑과 더불어 살아가는 데 필요한 [ 반려동물 보호 및 건강 지킴이 ]<br><br>태어나줘서 고맙고, 떠나는 그 순간까지 모든 것을 함께 나누며, 아낌없이 해주고픈 [ 반려동물 케어 서비스, 반려인 기대수명 및 행복수치 상승 효과 ]<br><br>반려 동물을 키우는 1인 가구 및 반려동물의 도움이 필요한 계층 등 '플랫폼 서비스'를 통해 삶의 질을 향상 시키는 [ 사회 전반적인 발전 및 공정성 증진 ]</td></tr>
</table>

| | |
|---|---|
| | |
| 개발방식 |  |
| 프로젝트 흐름도 |  |

## 【 2. 요구사항 정의서 】

*기능적 요구사항*

| 관리자 | 의료 | | 장례 | | 일반 회원 |
|---|---|---|---|---|---|
| | 업체 | 직원 | 업체 | 직원 | |
| 1. 회원 기능 | | | | | |
| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
| 회원관리 | 회원관리 | 회원관리 | 회원관리 | 회원관리 | 회원관리 |
| 회원등록 | 회원등록 | 회원등록 | 회원등록 | 회원등록 | 회원등록 |
| 회원수정 | 회원수정 | 회원수정 | 회원수정 | 회원수정 | 회원수정 |
| 회원탈퇴 | 회원탈퇴 | 회원탈퇴 | 회원탈퇴 | 회원탈퇴 | 회원탈퇴 |
| 개인회원조회 | 개인회원조회 | 개인회원조회 | 개인회원조회 | 개인회원조회 | 개인회원조회 |
| 업체회원조회(의업/장업/판업/관) | 업체회원조회(의업/장업/관) | | 업체회원조회(의업/장업/관) | | |
| 전체회원검색(관) | | | | | |
| 업체직원등록(의업/장업/판업/관) | 업체직원등록(의업/장업/판업/관) | | 업체직원등록(의업/장업/판업/관) | | |
| 2. 마이페이지 관리 | | | | | |
| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
| 사용자프로필설정(사용자,반려동물) | 사용자프로필설정(사용자,반려동물) | 사용자프로필설정(사용자,반려동물) | 사용자프로필설정(사용자,반려동물) | 사용자프로필설정(사용자,반려동물) | 사용자프로필설정(사용자,반려동물) |
| | | | | | |
| | | 결제 내역 확인 | | 결제 내역 확인 | 결제 내역 확인 |

| 3. 의료업체 정보 페이지 관리 | | | | | |
|---|---|---|---|---|---|
| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
| 정보 페이지 등록(의업) | 정보 페이지 등록(의업) | | | | |
| | 정보 페이지 수정(의) | 정보 페이지 수정(의) | | | |
| 정보 페이지 삭제(의업/관) | 정보 페이지 삭제(의업/관) | | | | |
| 정보 페이지 검색(일,판,의,장,관) | 정보 페이지 검색(일,판,의,장,관) | 정보 페이지 검색(일,판,의,장,관) | 정보 페이지 검색(일,판,의,장,관) | 정보 페이지 검색(일,판,의,장,관) | 정보 페이지 검색(일,판,의,장,관) |

| 4. 의료업체평 관리 | | | | | |
|---|---|---|---|---|---|
| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
| | | 업체평등록(일) | | 업체평등록(일) | 업체평등록(일) |
| | | 업체평수정(일) | | 업체평수정(일) | 업체평수정(일) |
| 업체평삭제(일) | | 업체평삭제(일) | | 업체평삭제(일) | 업체평삭제(일) |
| 업체평검색(일,판,의,장,관) | 업체평검색(일,판,의,장,관) | 업체평검색(일,판,의,장,관) | 업체평검색(일,판,의,장,관) | 업체평검색(일,판,의,장,관) | 업체평검색(일,판,의,장,관) |

| 의료업체 문의 | | | | | |
|---|---|---|---|---|---|
| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
| | | 문의 발신 | | 문의 발신 | 문의 발신 |
| | 문의 수신 | 문의 수신 | | | |
| | | | | | |

| 장례업체 정보 페이지 관리 | | | | | |
|---|---|---|---|---|---|
| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
| | | | 정보 페이지 등록(장업) | | |

| | | | | | |
|---|---|---|---|---|---|
| | | | 정보 페이지<br>수정(장) | 정보 페이지<br>수정(장) | |
| 정보 페이지<br>삭제(장업/관) | | | 정보 페이지<br>삭제(장업/관) | | |
| 정보 페이지<br>검색(일,판,의,장,관) | 정보 페이지<br>검색(일,판,의,장,관) | 정보 페이지<br>검색(일,판,의,장,관) | 정보 페이지<br>검색(일,판,의,장,관) | 정보 페이지<br>검색(일,판,의,장,관) | 정보 페이지<br>검색(일,판,의,장,관) |
| | | | | | |

**7. 장례 서비스 예약**

| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
|---|---|---|---|---|---|
| | | 서비스 예약 | | 서비스 예약 | 서비스 예약 |
| | | | 서비스 예약 승인 | 서비스 예약 승인 | |

**8. 장례 서비스 결제관리**

| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
|---|---|---|---|---|---|
| | 장례 서비스<br>결제등록 | 장례 서비스<br>결제등록 | | | 장례 서비스<br>결제등록 |
| | 장례 서비스<br>결제취소 | 장례 서비스<br>결제취소 | 장례 서비스<br>결제취소 | 장례 서비스<br>결제취소 | 장례 서비스<br>결제취소 |
| 장례 서비스<br>결제확인 | 장례 서비스<br>결제확인 | 장례 서비스<br>결제확인 | 장례 서비스<br>결제확인 | 장례 서비스<br>결제확인 | 장례 서비스<br>결제확인 |

**9. 장례업체평 관리**

| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
|---|---|---|---|---|---|
| | | . 업체평등록(일) | | . 업체평등록(일) | . 업체평등록(일) |
| | | 업체평수정(일) | | 업체평수정(일) | 업체평수정(일) |
| 업체평삭제(일) | | 업체평삭제(일) | | 업체평삭제(일) | 업체평삭제(일) |
| 업체평검색(일,판,의,장,관) | | 업체평검색(일,판,의,장,관) | 업체평검색(일,판,의,장,관) | 업체평검색(일,판,의,장,관) | 업체평검색(일,판,의,장,관) |

| 10. 장례업체 문의 | | | | | |
|---|---|---|---|---|---|
| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
| 문의 발신 | | 문의 발신 | | 문의 발신 | 문의 발신 |
| | | | 문의 수신(장) | 문의 수신(장) | |
| | | | | | |

| 11. 게시판 (생/수/삭) | | | | | |
|---|---|---|---|---|---|
| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
| 게시판 생성 | | | | | |
| 게시판 수정 | | | | | |
| 게시판 삭제 | | | | | |

| 12. 게시판 카테고리 | | | | | |
|---|---|---|---|---|---|
| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
| 게시판 카테고리 생성 | | | | | |
| 게시판 카테고리 수정 | | | | | |
| 게시판 카테고리 삭제 | | | | | |
| 게시판 카테고리 검색 | | | | | |

| 13. 게시글 관리 | | | | | |
|---|---|---|---|---|---|
| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
| | 게시글 생성 | 게시글 생성 | 게시글 생성 | 게시글 생성 | 게시글 생성 |
| | 게시글 수정 | 게시글 수정 | 게시글 수정 | 게시글 수정 | 게시글 수정 |
| 게시글 삭제 | 게시글 삭제 | 게시글 삭제 | 게시글 삭제 | 게시글 삭제 | 게시글 삭제 |

| | | | | | |
|---|---|---|---|---|---|
| 게시글 검색 | 게시글 검색 | 게시글 검색 | 게시글 검색 | 게시글 검색 | 게시글 검색 |
| | 답변 생성 | 답변 생성 | 답변 생성 | 답변 생성 | 답변 생성 |
| | 답변 수정 | 답변 수정 | 답변 수정 | 답변 수정 | 답변 수정 |
| 답변 삭제 | 답변 삭제 | 답변 삭제 | 답변 삭제 | 답변 삭제 | 답변 삭제 |

**14. 지도관리**

| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
|---|---|---|---|---|---|
| 지도 마커 생성 | | | | | |
| 지도 마커 수정 | | | | | |
| 지도 마커 삭제 | | | | | |
| 지도 마커 검색 | 지도 마커 검색 | 지도 마커 검색 | 지도 마커 검색 | 지도 마커 검색 | 지도 마커 검색 |

**15. 알림항목관리 (업체별/병원별/게시판별)**

| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
|---|---|---|---|---|---|
| 알림 항목 생성 | | | | | |
| 알림 항목 수정 | | | | | |
| 알림 항목 삭제 | | | | | |

**16. 알림 관리**

| 관리자 | 업체(의료) | 직원(의료) | 장례(장례) | 직원(장례) | 일반 회원 |
|---|---|---|---|---|---|
| 알림 허용 | 알림 허용 | 알림 허용 | 알림 허용 | 알림 허용 | 알림 허용 |
| 알림 불허용 | 알림 불허용 | 알림 불허용 | 알림 불허용 | 알림 불허용 | 알림 불허용 |
| 알림 확인 | 알림 확인 | 알림 확인 | 알림 확인 | 알림 확인 | 알림 확인 |

## 【 2. 요구사항 정의서 】

*상세기능 정의*

### 1) 회원가입, 로그인

| NO | 요구 기능 사항 | 요구사항 상세 (처리 process) |
|---|---|---|
| 1 | 로그인 | 사용자가 입력한 정보와 DB에 있는 회원의 정보가 일치하면 로그인 |
| 2 | 로그아웃 | 세션에서 로그인한 사용자의 정보 제거 |
| 3 | 아이디 찾기 | 사용자 정보가 DB회원 정보와 일치 시 이메일 인증 후 아이디 조회 |
| 4 | 비밀번호 찾기 | 사용자 정보와 DB회원 정보 일치 시 이메일 인증 후 비밀번호 변경 |
| 5 | 개인정보수집동의(회원) | 일반회원의 개인정보 수집 동의 |
| 6 | 개인정보수집동의(업체) | 업체대표의 개인정보 수집 동의 |
| 7 | 일반회원 회원가입 | 새로운 일반회원 정보 등록 |
| 8 | 업체 회원가입 | 새로운 업체 정보 등록 |

### 2) 지도 검색

| NO | 요구 기능 사항 | 요구사항 상세 (처리 process) |
|---|---|---|
| 1 | 업체 검색 | 키워드를 통해 DB의 정보 나열 |
| 2 | 리뷰 작성 | 해당 업체에 리뷰 작성 기능 |
| 3 | 문의 작성 | 해당 업체에 문의 작성 기능 |
| 4 | 직원 신청 | 해당 업체에 직원 신청 기능 |

### 3) 업체 정보

| NO | 요구 기능 사항 | 요구사항 상세 (처리 process) |
|---|---|---|
| 1 | 업체 정보 | 업체의 정보를 볼 수 있다. |
| 2 | 리뷰 | 업체의 리뷰 목록을 볼 수 있다. |
| 3 | 구독 | 해당 업체 구독 기능 |

### 4) 업체 관리

| NO | 요구 기능 사항 | 요구사항 상세 (처리 process) |
|---|---|---|
| 1 | 업체 수정 | 업체 정보를 수정 할 수 있는 기능 |
| 2 | 업체 삭제 | 대표 회원) 업체 정보를 삭제 할 수 있는 기능 |
| 3 | 업체 대표 이미지 | 업체 대표 이미지를 등록할 수 있는 기능 |
| 4 | 문의 관리 | 업체의 문의 리스트 확인 및 답변 삭제 기능 |
| 5 | 리뷰 관리 | 업체의 리뷰 리스트 확인 및 삭제 기능 |
| 6 | 알림 관리 | 구독자에게 알림 발송 기능 |
| 7 | 직원 관리 | 직원 신청 목록 조회 및 승낙 기능 |

### 5) 마이페이지

| NO | 요구 기능 사항 | 요구사항 상세 (처리 process) |
|---|---|---|
| 1 | 문의 리뷰 확인 | 사용자가 작성한 리뷰와 문의 확인 기능 |
| 2 | 문의 리뷰 수정 | 사용자가 작성한 리뷰와 문의 수정 기능 |
| 3 | 문의 리뷰 삭제 | 사용자가 작성한 리뷰와 문의 삭제 기능 |
| 4 | 회원 정보 조회 | 사용자의 회원 정보 조회 |
| 5 | 회원 정보 수정 | 사용자의 회원 정보 수정 |
| 6 | 회원 탈퇴 | 사용자를 일반회원에서 휴면회원으로 전환(로그인 불가) |
| 7 | 반려동물 목록 조회 | 사용자가 등록한 반려동물 목록 조회 |
| 8 | 반려동물 정보 조회 | 반려동물의 정보 조회 |
| 9 | 반려동물 정보 수정 | 반려동물의 정보 수정 |
| 10 | 반려동물 정보 삭제 | 반려동물의 정보 삭제 |

### 6) 장례 서비스

| NO | 요구 기능 사항 | 요구사항 상세 (처리 process) |
|---|---|---|
| 1 | 장례 서비스 등록 | 장례 업체별 타이틀명, 코멘트, 가격, 이미지 정보 서비스 등록 |
| 2 | 장례 서비스 수정 | 타이틀명 제외 코멘트, 가격 정보 서비스 수정 |
| 3 | 장례 서비스 노출 | 등록시 장례 서비스 노출 기능 |

| | | |
|---|---|---|
| 4 | 장례 서비스 활성화 | 장례 서비스 수정 시 활성화/비활성화를 통해 노출 기능 옵션화 |

### 7) 장례 서비스 예약

| NO | 요구 기능 사항 | 요구사항 상세 (처리 process) |
|---|---|---|
| 1 | 장례 서비스 선택 | 장례 업체 클릭 시 선택하려는 장례 서비스 선택 기능 |
| 2 | 장례 서비스 옵션 | 예약일, 반려동물 등 장례 서비스에 필요한 정보 선택 기능 |
| 3 | 장례 서비스 예약 | 모든 선택사항 선택 시, 장례 예약 테이블에 DB 저장 |
| 4 | 예약 내역 | 예약 시, 예약 정보 노출 기능 |

### 8) 장례 서비스 결제

| NO | 요구 기능 사항 | 요구사항 상세 (처리 process) |
|---|---|---|
| 1 | 장례 예약 정보 | 결제 전, 예약한 장례 예약 정보 및 개인 정보 노출 기능 |
| 2 | 장례 결제 수단 | PG사 카카오페이 결제 방식 선택 |
| 3 | 장례 결제 확인 | 결제 버튼 클릭 시, 결제 테이블에 DB 저장 |
| 4 | 장례 결제 내역 | 결제 내역 버튼 클릭 시, 결제 정보 노출 기능 |

### 9) 커뮤니티

| NO | 요구 기능 사항 | 요구사항 상세 (처리 process) |
|---|---|---|
| 1 | 게시판 글 조회 | 게시판 종류에 따른 게시글 목록 조회 |
| 2 | 일반 게시글 작성 | 일반 게시글 등록 |
| 3 | 일반 게시글 조회 | 일반 게시글 조회 |
| 4 | 일반 게시글 수정 | 일반 게시글 수정 |
| 5 | 일반 게시글 삭제 | 일반 게시글 삭제 |
| 6 | 갤러리 게시글 작성 | 갤러리 게시글 등록 |
| 7 | 갤러리 게시글 조회 | 갤러리 게시글 조회 |
| 8 | 갤러리 게시글 수정 | 갤러리 게시글 수정 |
| 9 | 갤러리 게시글 삭제 | 갤러리 게시글 삭제 |

## 【 3. 업무 분담 】

*개인 담당 업무*

| 이 름 | 담당 업무 | 업무 내용 |
|---|---|---|
| 송재익 | 팀장 | 주기적인 회의 진행, 프로젝트 일정관리, 업무 조율 |
| | URI 매핑 작성 | 반려동물 장례와 관련된 연결 클래스와 전송방식 작성 |
| | DB 목록 작성 | 개인 담당 기능으로 사용할 논리적, 물리적 DB 구현 |
| | 테이블 정의서 작성 | 각 테이블의 컬럼 정보 정리 및 작성 |
| | 프로세스 흐름도 작성 | 프로세스 흐름도 작성 |
| | ERD 작성 | Dbeaver를 사용하여 프로젝트의 전체적인 ERD 작성 |
| | DB data 작성 | 개인 기능으로 사용할 Table 상세 데이터 입력 |
| | 네이밍 규칙 | 공통된 네이밍 규칙 정리 |
| | DB 상세 자료 작성 | 기본키, 외래키 구분 및 상세 자료 작성 |
| | 화면 설계서 작성 | 화면 설계서 작성 |
| | 화면 경로 작성 | 서비스 노출 - 서비스 선택 - 예약 - 결제 - 내역 화면 경로 작성 |
| | 클래스 경로 작성 | Package 및 Class Path 작성 |
| | 프로젝트 상세 일정 작성 | 주 단위 프로젝트 상세 일정 작성 |
| | 결제 기능 | 포트원 API를 활용한 KAKAOPAY PG사를 통한 결제 방식 기능 |
| | 결제 내역 관리 | 업체별 및 사용자별 결제 내역 기능 구현 |
| | 장례 서비스 관리 | 장례 서비스 등록 및 수정 기능 구현 |
| 박은수 | 팀원 | 회의 진행시 회의록 작성 및 정리 |
| | URI 매핑 작성 | 연결 클래스와 전송방식 작성 |
| | DB 목록 작성 | 사용할 논리적, 물리적 DB 구현 |
| | 테이블 정의서 작성 | 각 테이블의 컬럼 정보 정리 및 작성 |

| | | |
|---|---|---|
| | | |
| | 프로세스 흐름도 작성 | 프로세스 흐름도를 도표로 표현 |
| | ERD 작성 | dbeaver프로젝트의 전체적인 ERD 작성 |
| | DB data 작성 | 사용할 Table 상세 데이터 입력 |
| | 네이밍 규칙 | 공통된 네이밍 규칙 정리 |
| | DB 상세 자료 작성 | 상세 자료 작성 |
| | 화면 설계서 작성 | 화면 설계서 작성 |
| | 화면 경로 작성 | 화면 경로 작성 |
| | 클래스 경로 작성 | 클래스 경로 작성 |
| | 프로젝트 상세 일정 작성 | 프로젝트 상세 일정 작성 |
| | 지도 관리 | 지도 표시 및 업체 검색 |
| | 업체 관리 | 업체 관리 화면 구현 |
| | 문의 & 리뷰 관리 | 문의 & 리뷰 관리 화면 구현 |
| **한송이** | 팀원 | 회의 진행시 회의록 작성 및 정리 |
| | URI 매핑 작성 | 연결 클래스와 전송방식 작성 |
| | DB 목록 작성 | 사용할 논리적, 물리적 DB 구현 |
| | 테이블 정의서 작성 | 각 테이블의 컬럼 정보 정리 및 작성 |
| | 프로세스 흐름도 작성 | 프로세스 흐름도를 도표로 표현 |
| | ERD 작성 | dbeaver프로젝트의 전체적인 ERD 작성 |
| | DB data 작성 | 사용할 Table 상세 데이터 입력 |
| | 네이밍 규칙 | 공통된 네이밍 규칙 정리 |
| | DB 상세 자료 작성 | 상세 자료 작성 |
| | 화면 설계서 작성 | 화면 설계서 작성 |

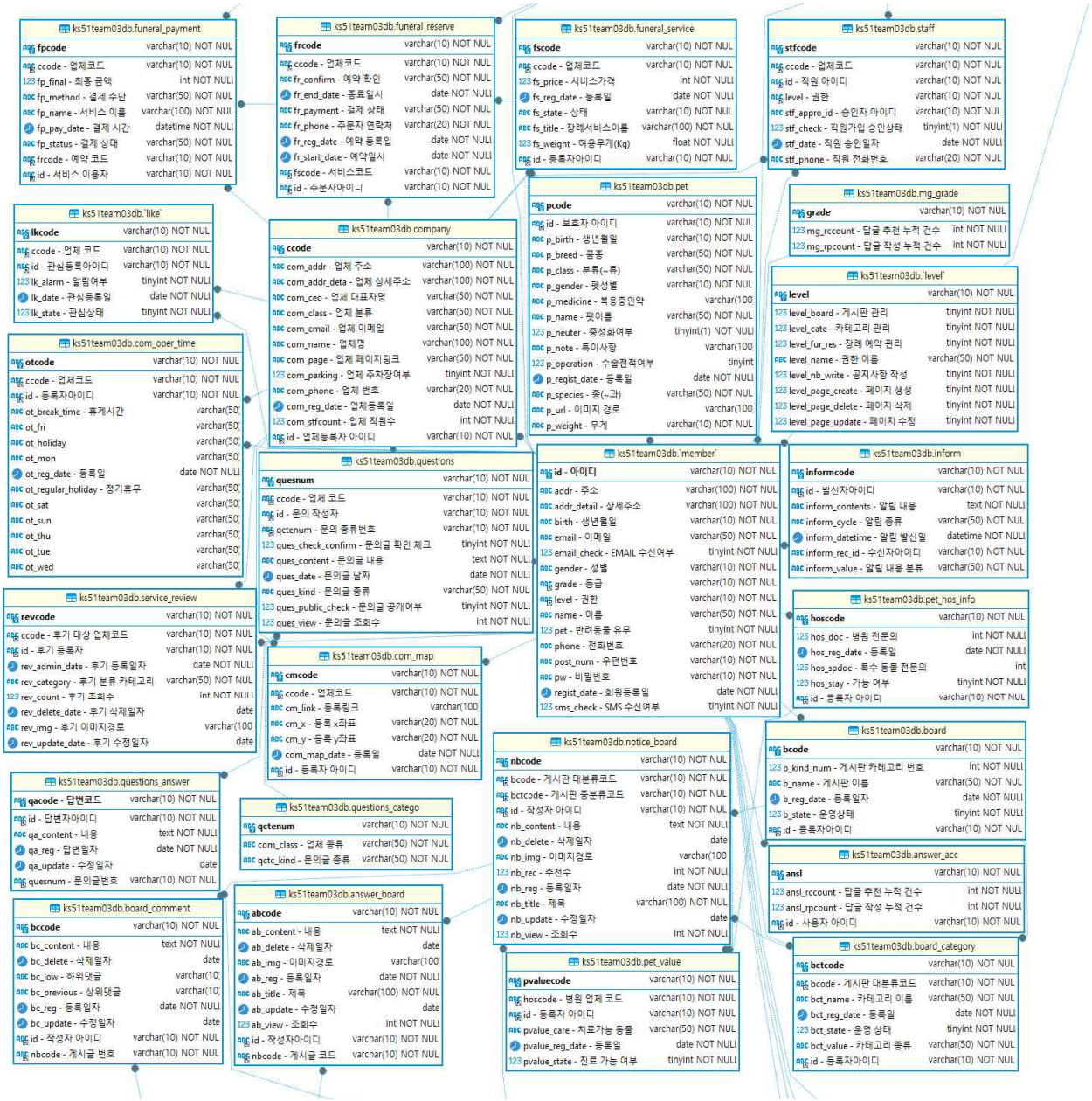
| | | |
|---|---|---|
| | 화면 경로 작성 | 화면 경로 작성 |
| | 클래스 경로 작성 | 클래스 경로 작성 |
| | 프로젝트 상세 일정 작성 | 프로젝트 상세 일정 작성 |
| | 로그인 | 사용자 로그인/아이디 찾기/비밀번호 찾기 구현 |
| | 회원가입 | 일반회원/업체회원 회원가입 구현 |
| | 마이페이지 | 회원의 정보와 활동 조회 및 수정 구현 |
| | 커뮤니티 | 공지/일반/질문/갤러리 게시판 조회, 등록, 수정, 삭제 구현 |

# 【 4. 설계 사양서 】

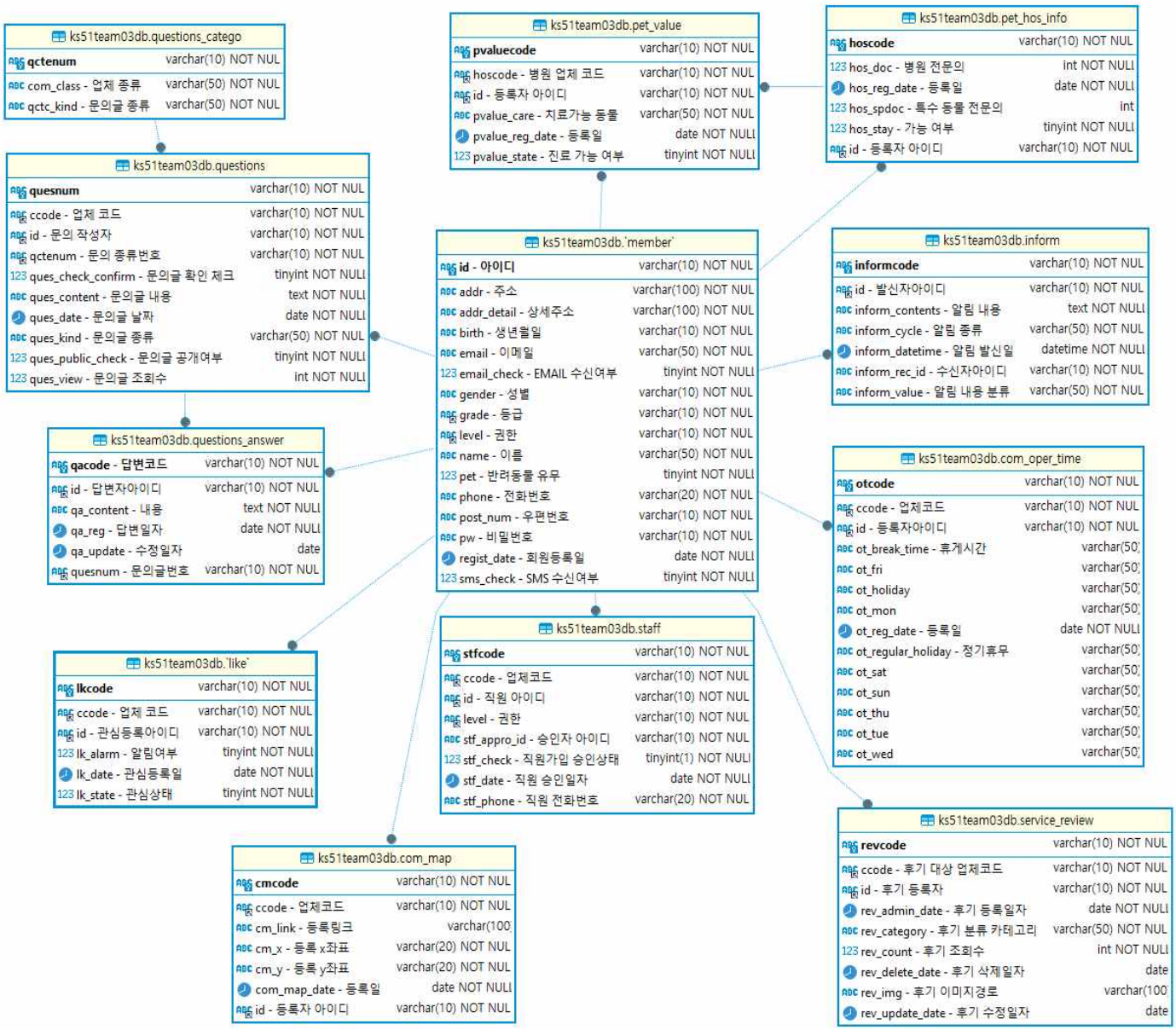
*ERD 물리 구조(테이블)*

ERD - 물리 모드

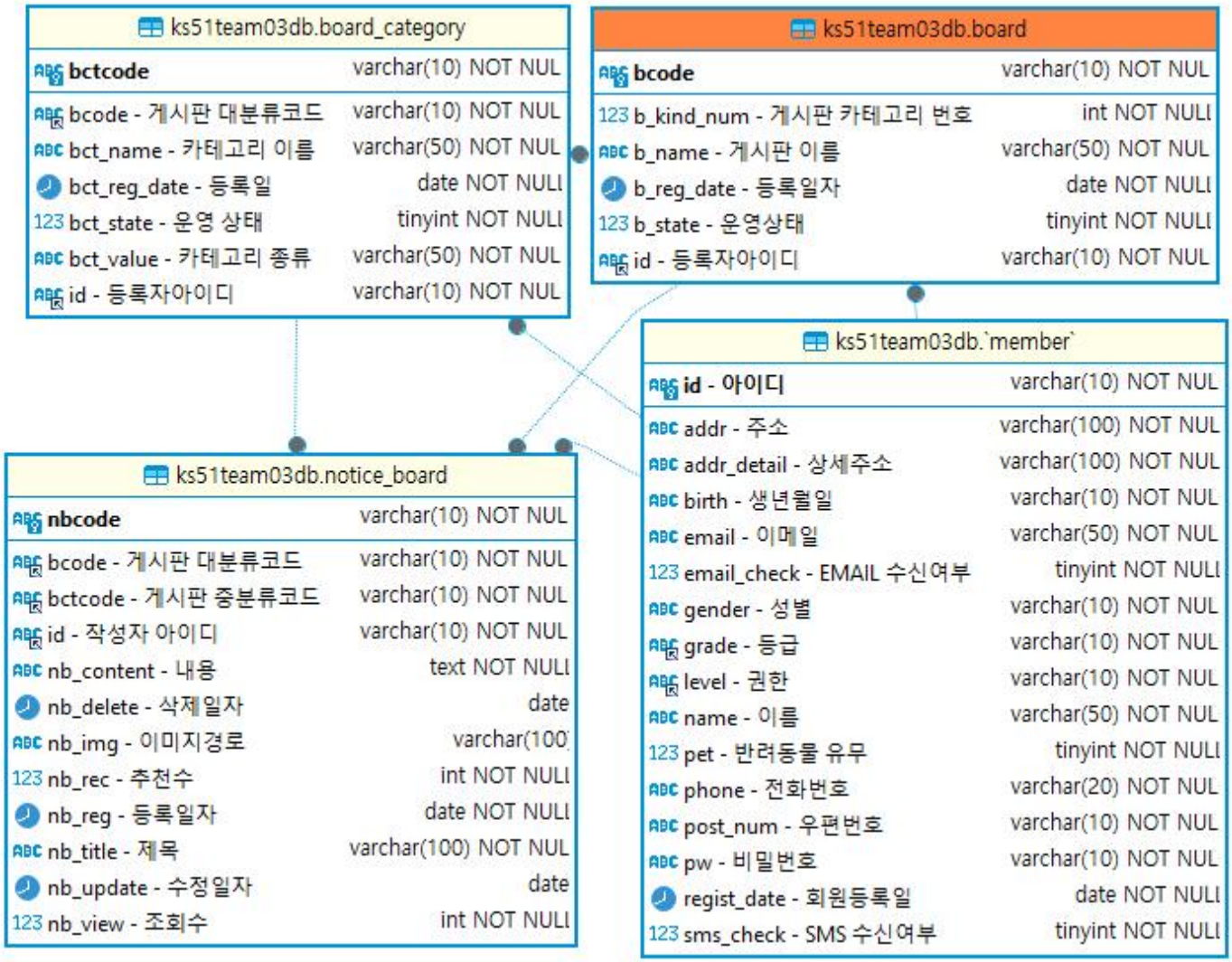
| 1. 전체 ERD | |
|---|---|



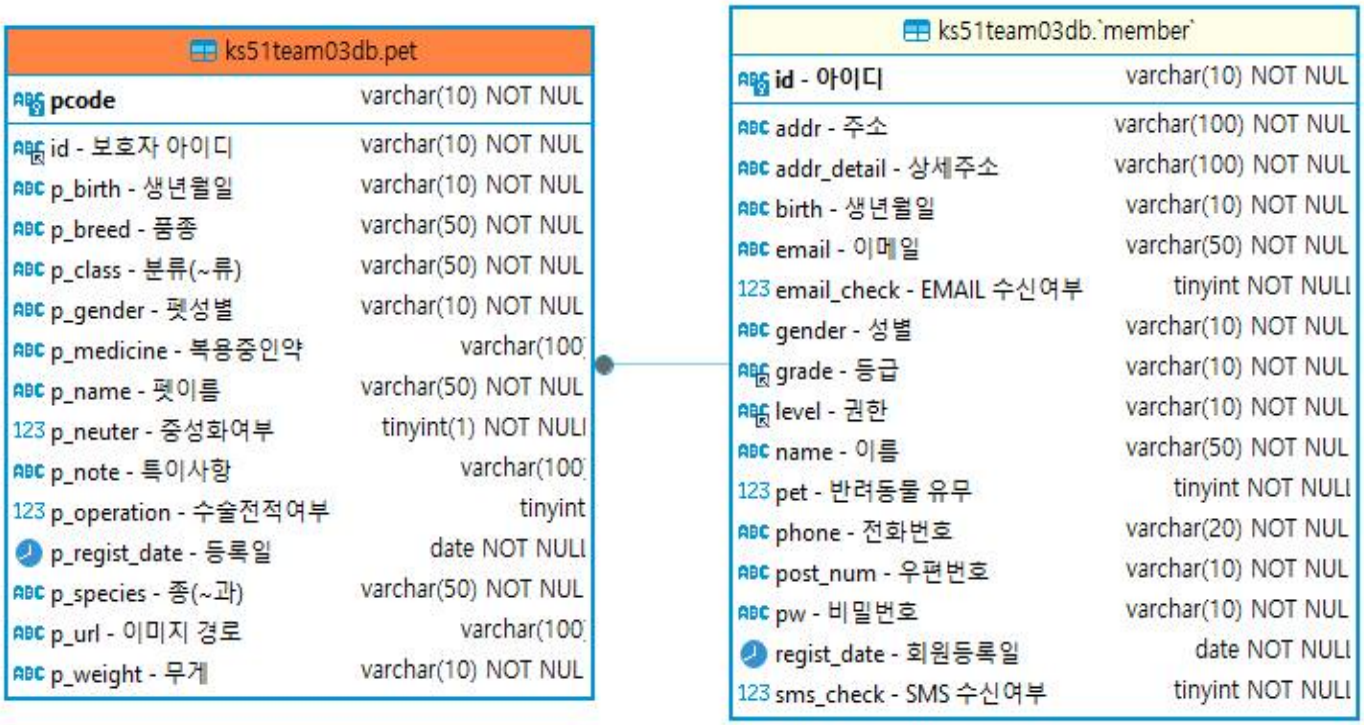
| Company & Mypage | | 담당 : 박은수 |
|---|---|---|



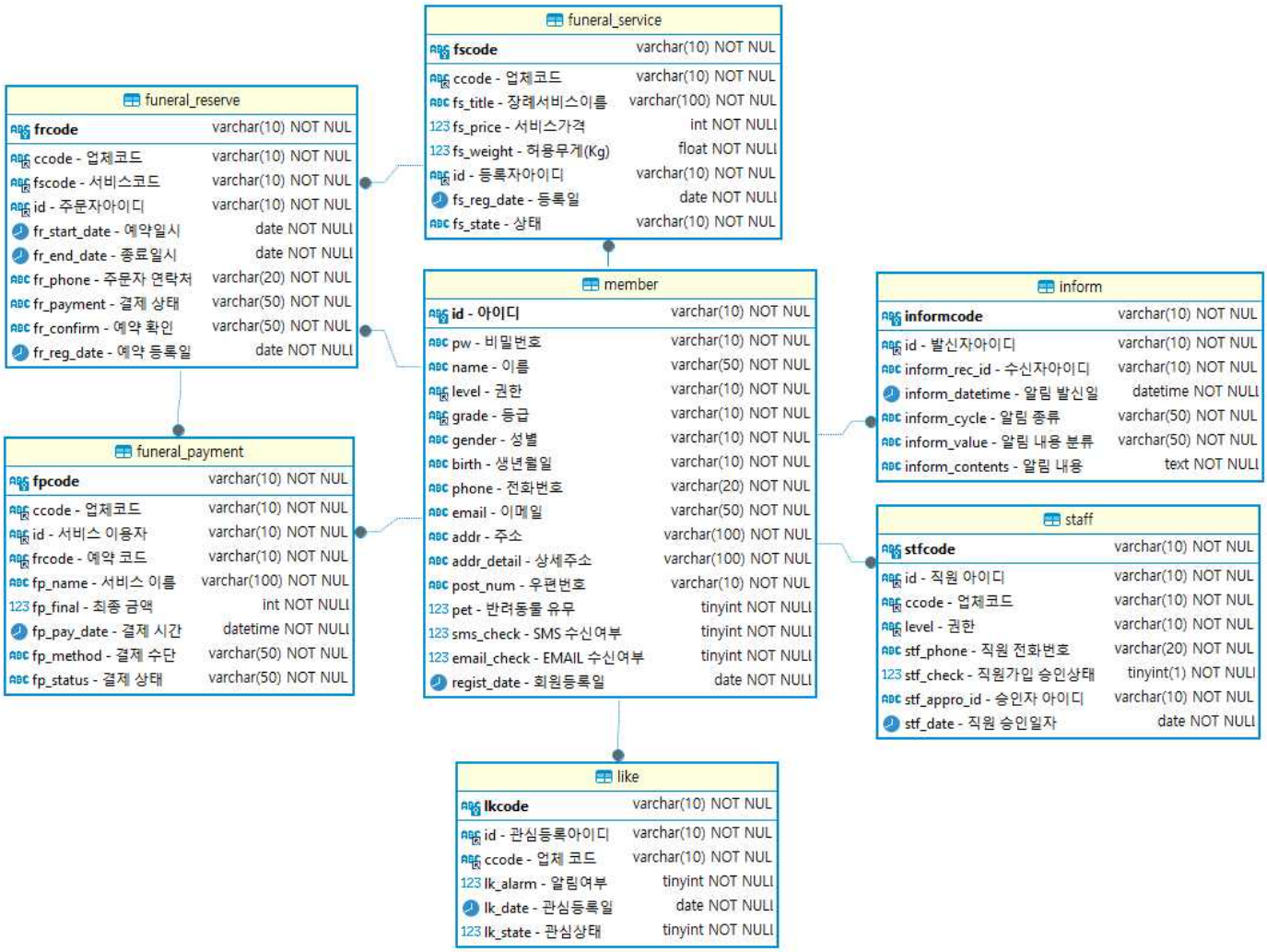
| 3. Member & Board | | 담당 : 한송이 |
|---|---|---|



| Member & Pet | | 담당 : 한송이 |
|---|---|---|



| Funeral | | 담당 : 송재익 |
|---|---|---|



## PAL 테이블 구조 (30개)

### 테이블 1 answer_acc

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 답글 내역 번호(pk) | ansl | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 아이디(f.k) | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 3 | 답글 작성 누적 건수 | ansl_rpcount | INT | 10 | N | N | NOT NULL | 0 | | | |
| 4 | 답글 추천 누적 건수 | ansl_rccount | INT | 10 | N | N | NOT NULL | 0 | | | |

### 테이블 2 company

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 업체 코드 | ccode(pk) | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 업체 분류 | com_class | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 3 | 업체 명 | com_name | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 4 | 업체 대표자명 | com_ceo | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 5 | 업체 주소 | com_addr | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 6 | 상세 주소 | com_addr_deta | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 7 | 업체 전화번호 | com_phone | VARCHAR | 20 | N | N | NOT NULL | | | | |
| 8 | 업체 직원수 | com_stfcount | int | 10 | N | N | NOT NULL | | | | |
| 9 | 업체 이메일 안내 | com_email | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 10 | 업체 주차장 유무 | com_parking | TINTYINT | 3 | N | N | NOT NULL | 0 | | | |
| 11 | 업체 페이지 링크 | com_page | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 12 | 업체 등록자 ID | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 13 | 업체 등록자 ID | com_post_num | VARCHAR | 10 | N | N | NOT NULL | | | | |
| 14 | 업체 등록자 ID | com_active | TINTYINT | | N | N | NOT NULL | | | | |
| 15 | 업체 등록일 | com_reg_date | DATE | | N | N | NOT NULL | | | | |

### 테이블 3 com_map

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 지도 코드(pk) | cmcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 업체 코드(fk) | ccode | VARCHAR | 10 | N | Y | NOT NULL | | company | ccode | |
| 3 | 등록 X 좌표 | cm_x | VARCHAR | 20 | N | N | NOT NULL | | | | |
| 4 | 등록 Y 좌표 | cm_y | VARCHAR | 20 | N | N | NOT NULL | | | | |
| 5 | 업체 지도 등록자 ID | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 6 | 업체 지도 등록일 | com_map_date | DATE | | N | N | NOT NULL | | | | |

테이블 4 com_oper_time

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 운영 시간 코드(pk) | otcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 업체 코드(fk) | ccode | VARCHAR | 10 | N | Y | NOT NULL | | company | ccode | |
| 3 | Mon | ot_mon | VARCHAR | 50 | N | N | NULL 허용 | | | | |
| 4 | Tue | ot_tue | VARCHAR | 50 | N | N | NULL 허용 | | | | |
| 5 | WEN | ot_wed | VARCHAR | 50 | N | N | NULL 허용 | | | | |
| 6 | Thu | ot_thu | VARCHAR | 50 | N | N | NULL 허용 | | | | |
| 7 | Fri | ot_fri | VARCHAR | 50 | N | N | NULL 허용 | | | | |
| 8 | Sat | ot_sat | VARCHAR | 50 | N | N | NULL 허용 | | | | |
| 9 | Sun | ot_sun | VARCHAR | 50 | N | N | NULL 허용 | | | | |
| 10 | 공휴일 | ot_hoilday | VARCHAR | 50 | N | N | NULL 허용 | | | | |
| 11 | 정기휴무 | ot_regular_holiday | VARCHAR | 50 | N | N | NULL 허용 | | | | |
| 12 | 휴게시간 | ot_break_time | VARCHAR | 50 | N | N | NULL 허용 | | | | |
| 13 | 등록자 ID | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 14 | 등록일 | ot_reg_date | DATE | | N | N | NOT NULL | | | | |

테이블 5 inform

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 알림 번호(pk) | informcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 발신자 ID(fk) | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 3 | 알림 발신일 | inform_datetime | DATETIME | | N | N | NOT NULL | | | | |
| 4 | 알림 종류 | inform_cycle | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 5 | 알림 내용 분류 | inform_value | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 6 | 알림 내용 | inform_contents | TEXT | | N | N | NOT NULL | | | | |

테이블 6 board

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 게시판 대분류 코드(pk) | bcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 게시판 이름 | b_name | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 3 | 게시판 카테고리 수 | b_kind_num | INT | 10 | N | N | NOT NULL | | | | |
| 4 | 등록자 ID | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 5 | 등록일 | b_reg_date | DATE | | N | N | NOT NULL | | | | |
| 6 | 운영 상태 | b_state | TINYINT | 3 | N | N | NOT NULL | 0 | | | |

테이블 7 board_category

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 게시판 중분류 코드(pk) | bctcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 게시판 대분류 코드(fk) | bcode | VARCHAR | 10 | N | Y | NOT NULL | | board | bcode | |
| 3 | 카테고리 이름 | bct_name | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 4 | 카테고리 종류 | bct_value | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 5 | 등록자 ID | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 6 | 등록일 | bct_reg_date | DATE | | N | N | NOT NULL | | | | |
| 7 | 운영 카테고리 안내 | bct_info | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 8 | 운영 상태 | bct_state | TINYINT | 3 | N | N | NOT NULL | | | | |

테이블 8 answer_board

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 답변 코드(pk) | abcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 작성자 id(fk) | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 3 | 게시글 코드(fk) | nbcode | VARCHAR | 10 | N | Y | NOT NULL | | board | nbcode | |
| 4 | 제목 | ab_title | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 5 | 내용 | ab_content | TEXT | | N | N | NOT NULL | | | | |
| 6 | 등록일자 | ab_reg | DATE | | N | N | NOT NULL | | | | |
| 7 | 수정일자 | ab_update | DATE | | N | N | NULL 허용 | | | | |
| 8 | 삭제일자 | ab_delete | DATE | | N | N | NULL 허용 | | | | |
| 9 | 이미지 | ab_img | VARCHAR | 100 | N | N | NULL 허용 | | | | |
| 10 | 조회수 | ab_view | INT | 10 | N | N | NOT NULL | | | | |

테이블 board_comment

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 댓글번호(pk) | bccode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 게시글 번호(fk) | nbcode | VARCHAR | 10 | N | Y | NOT NULL | | board | nbcode | |
| 3 | 작성자 id(fk) | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 4 | 내용 | bc_content | TEXT | | N | N | NOT NULL | | | | |
| 5 | 등록일자 | bc_reg | DATE | | N | N | NOT NULL | | | | |
| 6 | 수정일자 | bc_update | DATE | | N | N | NULL 허용 | | | | |
| 7 | 삭제일자 | bc_delete | DATE | | N | N | NULL 허용 | | | | |
| 8 | 상위댓글 | bc_previous | VARCHAR | 10 | N | N | NULL 허용 | | | | |
| 9 | 하위댓글 | bc_num | VARCHAR | 10 | N | N | NULL 허용 | | | | |

테이블 10 funeral_service

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 서비스코드(pk) | fscode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 업체코드(fk) | ccode | VARCHAR | 10 | N | Y | NOT NULL | | company | ccode | |
| 3 | 장례 서비스 | fs_title | VARCHAR | 100 | N | N | NOT NULL | | member | id | |
| 4 | 장례 서비스 가격 | fs_price | INT | 10 | N | N | NOT NULL | | | | |
| 5 | 허용 무게(kg) | fs_weight | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 6 | 등록자 ID | id | VARCHAR | 10 | N | Y | NOT NULL | | | | |
| 7 | 등록일 | fs_reg_date | DATE | | N | N | NOT NULL | | | | |
| 8 | 코멘트 | fs_comment | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 9 | 상태 | fs_state | VARCHAR | 10 | N | N | NOT NULL | | | | |

테이블 11 funeral_reserve

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 예약 코드(pk) | frcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 업체코드(fk) | ccode | VARCHAR | 10 | N | Y | NOT NULL | | company | ccode | |
| 3 | 서비스 코드(fk) | fscode | VARCHAR | 10 | N | Y | NOT NULL | | funeral_service | fscode | |
| 4 | 주문자 아이디(fk) | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 5 | 예약일시 | fr_start_date | DATE | | N | N | NOT NULL | | | | |
| 6 | 종료일시 | fr_end_date | DATE | | N | N | NOT NULL | | | | |
| 7 | 주문자 연락처 | fr_phone | VARCHAR | 20 | N | N | NOT NULL | | | | |
| 8 | 결제 상태 | fr_payment | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 9 | 예약 확인 | fr_confirm | VARCHAR | 50 | N | N | NOT NULL | 예약 대기 | | | |
| 10 | 예약 시간 | fr_reserve_time | TIME | | N | N | NOT NULL | | | | |
| 11 | 반려동물 명 | fr_pet_name | VARCHAR | 50 | N | N | NOT NULL | 예약 대기 | | | |
| 12 | 예약 등록일 | fr_reg_date | DATE | | N | N | NOT NULL | | | | |

테이블 12 funeral_payment

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 결제번호(PK) | fpcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 업체코드(fk) | ccode | VARCHAR | 10 | N | Y | NOT NULL | | company | ccode | |
| 3 | 서비스 이용자 ID(FK) | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 4 | 예약 코드(FK) | frcode | VARCHAR | 10 | N | Y | NOT NULL | | funeral_reserve | frcode | |
| 5 | 서비스 이름 | fp_name | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 6 | 최종금액 | fp_final | INT | 10 | N | N | NOT NULL | | | | |
| 7 | 결제시간 | fp_pay_date | DATETIME | | N | N | NOT NULL | | | | |
| 8 | 결제수단 | fp_method | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 9 | 트랜잭션아이디 | txid | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 10 | 오류코드 | code | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 11 | 오류문구 | message | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 12 | 결제상태 | fp_status | VARCHAR | 50 | N | N | NOT NULL | | | | |

테이블 13 level

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 권한 코드 | level | VARCHAR | 10 | Y | N | NOT NULL | 기본값 없음 | | | |
| 2 | 권한 이름 | level_name | VARCHAR | 50 | N | N | NULL 허용 | NULL | | | |
| 3 | 카테고리 관리 | level_cate | TINYINT | | N | N | NULL 허용 | DEFAULT '0' | | | |
| 4 | 게시판 관리 | level_board | TINYINT | | N | N | NULL 허용 | DEFAULT '0' | | | |
| 5 | 페이지 생성 | level_page_create | TINYINT | | N | N | NULL 허용 | DEFAULT '0' | | | |
| 6 | 페이지 삭제 | level_page_delete | TINYINT | | N | N | NULL 허용 | DEFAULT '0' | | | |
| 7 | 페이지 수정 | level_page_update | TINYINT | | N | N | NULL 허용 | DEFAULT '0' | | | |
| 8 | 장례 예약 관리 | level_fur_res | TINYINT | | N | N | NULL 허용 | DEFAULT '0' | | | |
| 9 | 공지사항 작성 | level_nb_write | TINYINT | | N | N | NULL 허용 | DEFAULT '0' | | | |

테이블 14 like

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 관심코드 | lkcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 아이디 | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 3 | 업체코드 | ccode | VARCHAR | 10 | N | Y | NOT NULL | | company | ccode | |
| 4 | 알림여부 | lk_alarm | TINYINT | | N | N | NOT NULL | 0 | | | |
| 5 | 관심등록일 | lk_date | DATE | | N | N | NOT NULL | | | | |
| 6 | 관심상태 | lk_state | TINYINT | 3 | N | N | NOT NULL | 0 | | | |
| | | | | | | | | | | | |

테이블 15 member

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 아이디 | id | VARCHAR | 16 | Y | N | NOT NULL | | | | |
| 2 | 비밀번호 | pw | VARCHAR | 16 | N | N | NOT NULL | | | | |
| 3 | 이름 | name | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 4 | 권한 | level | VARCHAR | 10 | N | N | NOT NULL | | level | level | |
| 5 | 등급 | grade | VARCHAR | 10 | N | N | NOT NULL | | mg_grade | grade | |
| 6 | 성별 | gender | VARCHAR | 10 | N | N | NOT NULL | | | | |
| 7 | 생년월일 | birth | VARCHAR | 10 | N | N | NOT NULL | | | | |
| 8 | 전화번호 | phone | VARCHAR | 20 | N | N | NOT NULL | | | | |
| 9 | 이메일 | email | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 10 | 주소 | addr | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 11 | 상세주소 | addr_detail | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 12 | 우편번호 | post_num | VARCHAR | 10 | N | N | NOT NULL | | | | |
| 13 | 반려동물 유무 | pet | TINYINT | 3 | N | N | NOT NULL | 0 | | | |
| 14 | SMS 수신여부 | sms_check | TINYINT | 3 | N | N | NOT NULL | 0 | | | |
| 15 | Email 수신여부 | email_check | TINYINT | 3 | N | N | NOT NULL | 0 | | | |
| 16 | 회원 등록일 | regist_date | DATE | | N | N | NOT NULL | | | | |

테이블 16 mg_grade

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 회원등급 | grade | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 답글 작성 누적 건수 | mg_rpcount | INT | | N | N | NOT NULL | 0 | | | |
| 3 | 답글 추천 누적 건수 | mg_rccount | INT | | N | N | NOT NULL | 0 | | | |

테이블 17 notice_board

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 게시글 코드 | nbcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | id | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 3 | 게시판 대분류 코드 | bcode | VARCHAR | 10 | N | Y | NOT NULL | | board | bcode | |
| 4 | 게시판 중분류 코드 | bctode | VARCHAR | 10 | N | Y | NOT NULL | | board_category | bctcode | |
| 5 | 제목 | nb_title | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 6 | 내용 | nb_content | TEXT | | N | N | NOT NULL | | | | |
| 7 | 등록일자 | nb_reg | DATE | | N | N | NOT NULL | | | | |
| 8 | 수정일자 | nb_update | DATE | | N | N | NOT NULL | | | | |
| 9 | 삭제일자 | nb_delete | DATE | | N | N | NOT NULL | | | | |
| 10 | 이미지(파일)경로 | nb_img | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 11 | 조회수 | nb_view | INT | | N | N | NOT NULL | | | | |
| 12 | 추천수 | nb_rec | INT | | N | N | NOT NULL | | | | |

테이블 18 pet

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 반려동물코드 | pcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 보호자 | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 3 | 팻이름 | p_name | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 4 | 성별 | p_gender | VARCHAR | 10 | N | N | NOT NULL | | | | |
| 5 | 생년월일 | p_birth | VARCHAR | 10 | N | N | NOT NULL | | | | |
| 6 | 분류 | p_class | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 7 | 종 | p_species | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 8 | 품종 | p_breed | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 9 | 무게 | p_weight | VARCHAR | 10 | N | N | NOT NULL | | | | |
| 10 | 중성화여부 | p_neuter | TINYINT | | N | N | NOT NULL | 0 | | | |
| 11 | 수술전적 | member_birth | TINYINT | | N | N | NOT NULL | 0 | | | |
| 12 | 복용중인약 | p_medicine | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 13 | 특이사항 | p_note | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 14 | 이미지경로 | p_url | VARCHAR | 100 | N | N | NOT NULL | | | | |
| 15 | 반려동물 등록일 | p_regist_date | DATE | | N | N | NOT NULL | | | | |

테이블 19 pet_hos_info

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 병원 정보 코드 | hoscode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 병원 전문의 | hos_doc | INT | | N | N | NOT NULL | | | | |
| 3 | 특수 동물 전문의 | hos_spdoc | INT | | N | N | NOT NULL | | | | |
| 4 | 입원 가능 유무 | hos_stay | TINYINT | | N | N | NOT NULL | | | | |
| 5 | 등록자 ID | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 6 | 등록일 | hos_reg_date | DATE | | N | N | NOT NULL | | | | |
| | | | | | | | | | | | |

테이블 20 pet_value

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 치료 가능 동물 코드 | pvaluecode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 병원 업체 코드(fk) | hoscode | VARCHAR | 10 | N | Y | NOT NULL | | pet_hos_info | hoscode | |
| 3 | 치료가능 동물 | pvalue_care | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 4 | 등록자 ID | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 5 | 등록일 | pvalue_reg_date | DATE | | N | N | NOT NULL | | | | |
| 6 | 상태 | pvalue_state | TINYINT | | N | N | NOT NULL | | | | |

테이블 21 questions

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 문의글 번호 | quesnum | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 업체 코드 | ccode | VARCHAR | 10 | N | Y | NOT NULL | company | ccode | | |
| 3 | 문의 종류 번호 | qctenum | VARCHAR | 10 | N | Y | NOT NULL | questions_catego | qctenum | | |
| 4 | 문의 작성자 ID | id | VARCHAR | 10 | N | Y | NOT NULL | member | id | | |
| 5 | 문의글 내용 | ques_content | TEXT | | N | N | NOT NULL | | | | |
| 6 | 문의글 날짜 | ques_date | DATE | | N | N | NOT NULL | | | | |

테이블 22 Questions_catego

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 문의 번호(pk) | qctenum | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 문의글 종류 | qcte_kind | VARCHAR | 50 | N | N | NOT NULL | | | | |
| 3 | 업체 종류 | com_class | VARCHAR | 50 | N | N | NOT NULL | | | | |

## 테이블 23 service_review

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 후기 등록 코드 | revcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 회원비밀번호 | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 3 | 후기 대상 업체 코드 | ccode | VARCHAR | 10 | N | Y | NOT NULL | | company | ccode | |
| 4 | 후기 조회수 | rev_count | INT | | N | N | NOT NULL | | | | |
| 5 | 후기 등록일자 | rev_admin_date | DATE | | N | N | NOT NULL | | | | |
| 6 | 후기 수정일자 | rev_update_date | DATE | | N | N | NULL 허용 | | | | |
| 7 | 후기 삭제 일자 | rev_delete_date | DATE | | N | N | NULL 허용 | | | | |
| 8 | 후기 이미지 | rev_img | VARCHAR | 100 | N | N | NULL 허용 | | | | |

## 테이블 24 staff

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 직원코드(PK) | stfcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 직원 아이디(fk) | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 3 | 업체 코드(fk) | ccode | VARCHAR | 10 | N | Y | NOT NULL | | company | ccode | |
| 4 | 업체 권한(fk) | level | VARCHAR | 10 | N | Y | NOT NULL | | level | level | |
| 5 | 사무용 전화번호 | stf_phone | VARCHAR | 20 | N | N | NOT NULL | | | | |
| 6 | 직원가입 승인 처리상태 | stf_check | TINYINT | | N | N | NOT NULL | | | | |
| 7 | 승인 ID | stf_appro_id | VARCHAR | 10 | N | N | NOT NULL | | | | |
| 8 | 직원 승인일 | stf_date | DATE | | N | N | NOT NULL | | | | |

## 테이블 25 question_answer

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 답변 코드(PK) | stfcode | VARCHAR | 10 | Y | N | NOT NULL | | | | |
| 2 | 작성자 id(fk) | id | VARCHAR | 10 | N | Y | NOT NULL | | member | id | |
| 3 | 문의글 번호(fk) | quesnum | VARCHAR | 10 | N | Y | NOT NULL | | questions | quesnum | |
| 4 | 내용 | qa_content | TEXT | | N | N | NOT NULL | | | | |
| 5 | 답변일자 | qa_reg | DATE | | N | N | NOT NULL | | | | |
| 6 | 수정일자 | qa_update | DATE | | N | N | NULL 허용 | | | | |

## 테이블 26 files

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 파일 코드 | file_idx | VARCHAR | 255 | Y | N | NOT NULL | | | | |
| 2 | 파일 이름 원본 | file_origin_name | VARCHAR | 255 | N | N | NULL 허용 | | | | |
| 3 | 파일 새로운 이름 | file_new_name | VARCHAR | 255 | N | N | NULL 허용 | | | | |
| 4 | 파일 경로 | file_path | VARCHAR | 255 | N | N | NULL 허용 | | | | |
| 5 | 파일 사이즈 | file_size | BIGINIT | 19 | N | N | NULL 허용 | | | | |
| 6 | 파일 등록일 | file_reg_date | DATE | | N | N | NULL 허용 | | | | |
| 7 | 파일 카테고리 | file_cate | VARCHAR | 50 | N | N | NULL 허용 | | | | |

## 테이블 27 company_img

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 업체 이미지 코드 | cicode | VARCHAR | 50 | Y | N | NOT NULL | | | | |
| 2 | 업체 코드 | ccode | VARCHAR | 10 | N | Y | NULL 허용 | | company | ccode | |
| 3 | 업체 이미지 주소 | file_idx | VARCHAR | 255 | N | Y | NULL 허용 | | files | file_idx | |
| 4 | 파일 카테고리 | file_cate | VARCHAR | 50 | N | N | NULL 허용 | | | | |

## 테이블 28 funeral_service_img

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 장례 서비스 이미지 코드 | fsicode | VARCHAR | 50 | Y | N | NOT NULL | | | | |
| 2 | 업체 코드 | ccode | VARCHAR | 10 | N | Y | NULL 허용 | | company | ccode | |
| 3 | 장례 서비스 코드 | fscode | VARCHAR | 255 | N | Y | NULL 허용 | | funeral_service | fscode | |
| 4 | 업체 이미지 주소 | file_idx | VARCHAR | 255 | N | Y | NULL 허용 | | files | file_idx | |
| 5 | 파일 카테고리 | file_cate | VARCHAR | 50 | N | N | NULL 허용 | | | | |

## 테이블 29 nboard_img

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 게시글 이미지 코드 | nbicode | VARCHAR | 50 | Y | N | NOT NULL | | | | |
| 2 | 게시글 코드 | nbcode | VARCHAR | 10 | N | Y | NULL 허용 | | notice_board | nbcode | |
| 3 | 업체 이미지 주소 | file_idx | VARCHAR | 255 | N | Y | NULL 허용 | | files | file_idx | |
| 4 | 파일 카테고리 | file_cate | VARCHAR | 50 | N | N | NULL 허용 | | | | |

## 테이블 30 inform_recipient

| 순번 | 컬럼명 | 컬럼ID | 타입 | 길이 | PK여부 | FK여부 | NULL여부 | 기본값 | 참조테이블 | 참조 컬럼 ID | 비고 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 알림 확인 코드 | ircode | VARCHAR | 50 | Y | N | NOT NULL | | | | |
| 2 | 회원 아이디 | id | VARCHAR | 16 | N | Y | NOT NULL | | member | id | |
| 3 | 알림 코드 | informcode | VARCHAR | 10 | N | Y | NOT NULL | | inform | informcode | |
| 4 | 알림 확인체크 | ircheck | TINYINT | 3 | N | N | NOT NULL | | | | |

# 【 4. 설계 사양서 】

*네이밍 규칙*

<table>
<tr><td>패키지 구조</td><td colspan="3">▶ Level 0 : 배포할 도메인 명을 뒤에서 부터 명칭 ( ex: 도메인명 rentravel.com -> com.rentrevel)<br><br>▶ Level : 업무 시스템 구분 ex:) 학사관리시스템(admin), 학생시스템(user)<br><br>▶ Level 2 : Web Application 구성 요소로 구분 (ex: controller, service, mapper, domain, config, interceptor)<br><br>▶ Level 3 : 실구현체 이름 + 접미사: 상위 패키지 네이밍 (파스칼 표기법) (ex: UserController, UserService, UserMapper )</td></tr>
<tr><td></td><td>Level 0</td><td>Level</td><td>Level 2</td><td>Level 3</td></tr>
<tr><td></td><td rowspan="6">ks51team03</td><td>admin</td><td>controller</td><td>RewardController.java</td></tr>
<tr><td></td><td>seller</td><td>service</td><td>DonationController.java</td></tr>
<tr><td></td><td>user</td><td>dto</td><td>InvestmentController.java</td></tr>
<tr><td></td><td>common</td><td>mapper</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>명명규칙</td><td colspan="3">▶ 공통</td></tr>
<tr><td></td><td colspan="3">모든 이름은 a-z, A-Z, 0-의 글자로 조합한다.</td></tr>
<tr><td></td><td colspan="3">프로그램 내에서 사용하는 모든 이름은 64자를 넘지 않는 것으로 한다.</td></tr>
<tr><td></td><td colspan="3">정확한 의미 전달을 위해 약어의 사용은 피한다.</td></tr>
<tr><td></td><td colspan="3">특수기호는 사용하지 않는 것을 원칙으로 한다.</td></tr>
<tr><td></td><td colspan="3">약어 사용이 의미 전달에 도움이 되는 경우에는 표준용어사전에 포함되어있는 약속된 약어를 사용할 수 있으나, 그렇지 않은 경우에는 단어의 spelling을 모두 적는 것으로 한다</td></tr>
<tr><td></td><td colspan="3">▶ 파일명</td></tr>
<tr><td></td><td rowspan="5">Java</td><td colspan="2">대소문자를 사용하며 단어의 시작부분은 대문자로 표기한다. (파스칼 표기법)</td></tr>
<tr><td></td><td colspan="2">단어는 표준용어사전을 따른다.</td></tr>
<tr><td></td><td colspan="2">명사 또는 동사 + 명사의 혼합된 형태로 이름을 구성한다.</td></tr>
<tr><td></td><td colspan="2">첫 번째 문자는 항상 대문자를 사용한다.</td></tr>
<tr><td></td><td colspan="2">클래스 유형에 따라 다음 표를 참고하여 구성한다.</td></tr>
<tr><td></td><td>mapper</td><td colspan="2">Dao (java파일)의 명명 규칙을 따른다.<br>(ex: Dao MemberMapper -> mapper MemberMapper)</td></tr>
</table>

| | | |
|---|---|---|
| | | 파일의 확장자는 .xml로 한다. |
| | html | 대소문자를 사용하여 변수명의 시작부분을 소문자로 표기한다. |
| | | 파일의 확장자는 .html로 한다. |
| | css | 대소문자를 사용하여 변수명의 시작부분을 소문자로 표기한다. |
| | | 파일의 확장자는 .css로 한다. |
| | JavaScript | 대소문자를 사용하여 변수명의 시작부분을 소문자로 표기한다. |
| | | 파일의 확장자는 .js로 한다. |
| | | 오픈소스 자바스크립트의 경우, 최초 배포된 이름 그대로 유지한다. |

▶ method

동사 혹은 동사 + 명사로 작성하며, 혼합된 형태로 이름을 구성할 때에는 각 단어의 첫 글자는 대문자로 작성한다. (카멜표기법)

상황에 따라 다음 표를 참고하여 일반적인 Prefix를 적용하며, 나머지는 필요에 따른 적절한 이름을 사용한다.

| 기능유형 | 대표용어 |
|---|---|
| get | 조회 |
| add | 등록 |
| modify | 수정 |
| remove | 삭제 |
| confirm | 확인 |
| cancel | 취소 |
| calc | 계산 |
| save | 저장 |
| call | 호출 |
| send | 송신 |
| receive | 수신 |
| transfer | 송수신 |
| check | 검토 |
| is | 유무(여부)확인 |

▶ 변수명

대소문자를 사용하여 변수명의 시작부분을 소문자로 표기한다.

명사로 작성하며 각 단어의 첫 글자에는 대문자로 표기한다.

변수명에는 기호나 특수문자를 사용하지 않는다.

변수명은 의미 있는 단어를 사용하도록 한다.(표준 용어사전 참고)

<table>
<tr><td rowspan="8"></td><td colspan="3">변수에 해당하는 컬럼이 테이블에 존재하는 경우에는 컬럼명을 기준으로 camel 표기법을 준수하여 사용한다</td></tr>
<tr><td colspan="3">변수명은 중복되는 이름이 나오지 않도록 하여 일련번호를 붙여 사용하는 경우가 없도록 한다.</td></tr>
<tr><td colspan="3">상수명은 공통코드를 제외한 변수명 전체를 대문자로 하고, 단어 연결에는 '_'를 사용한다</td></tr>
<tr><td colspan="3">예시</td></tr>
<tr><td rowspan="3">java</td><td>변수</td><td>int orderCnt; / double deliveryAmt; / String userName; / StringBuffer sqlStr;</td></tr>
<tr><td>상수</td><td>static final int WIDTH = 4; / static final long DEFAULT_EXPIRES_TIME = TimeUnit.MINUTES.toSeconds(30);</td></tr>
<tr><td>사용자 정의 타입</td><td>private MemberService memberService;</td></tr>
<tr><td rowspan="2">javaScript</td><td>변수</td><td>var orderCnt; / var deliveryAmt; / var userName;</td></tr>
<tr><td>상수</td><td>const WIDTH = 4; / const DEFAULT_PER_PAGE = 0;</td></tr>
</table>

## PAL 패키지 구조

### 1. Funeral 담당 : 송재익

| Funeral | | | | |
|---|---|---|---|---|
| **Funeral.ServiceList** | | | | |
| 1 | ks51team03.funeral.serviceList.controlle | ServiceListController | funeralServiceList | 장례 업체 조회 controller |
| 2 | ks51team03.funeral.serviceList.controlle | ServiceListController | funeralReserve | 장례 서비스 조회 및 필수 정보 선택 후 예약 기능 |
| 3 | ks51team03.funeral.serviceList.controlle | ServiceListController | funeralInsertService | 장례 서비스 등록 |
| 4 | ks51team03.funeral.serviceList.controlle | ServiceListController | serviceConfirm | 장례 서비스 등록 확인 페이지 |
| 5 | ks51team03.funeral.serviceList.controlle | ServiceListController | getRegServiceList | 등록된 장례 서비스 리스트 조회 |
| 6 | ks51team03.funeral.serviceList.service | ServiceListController | modifyFuneralService | 등록된 장례 서비스 수정 |
| 7 | ks51team03.funeral.serviceList.service | ServiceListService | getCompanyInfo | 장례 업체 조회 Service |
| 8 | ks51team03.funeral.serviceList.service | ServiceListService | getServiceList | 장례 서비스 조회 Service |
| 9 | ks51team03.funeral.serviceList.service | ServiceListService | insertFuneralService | 장례 서비스 등록 Service |
| 10 | ks51team03.funeral.serviceList.service | ServiceListService | addFuneralServiceImg | 장례 서비스 이미지 등록 Serivce |
| 11 | ks51team03.funeral.serviceList.mapper | ServiceListService | modifyServiceInfoByCode | 장례 서비스 수정 Service |
| 12 | ks51team03.funeral.serviceList.mapper | ServiceListService | getServiceListDto | 장례 서비스 조회 Mapper |
| 13 | ks51team03.funeral.serviceList.mapper | ServiceListMapper | getCompanyInfo | 장례 업체 조회 Mapper |
| 14 | ks51team03.funeral.serviceList.mapper | ServiceListMapper | getMemberPet | 반려동물 정보 가져오기 Mapper |
| 15 | ks51team03.funeral.serviceList.mapper | ServiceListMapper | updateServiceInfo | 등록된 장례 서비스 수정 Mapper |
| 16 | ks51team03.funeral.serviceList.mapper | ServiceListMapper | getServiceImgs | 장례 서비스 이미지 가져오기 Mapper |
| 17 | ks51team03.funeral.serviceList.dto | ServiceListDto | | 장례 서비스 DTO |
| 18 | ks51team03.funeral.serviceList.dto | ServiceImgDto | | 장례 서비스 이미지 DTO |
| 19 | ks51team03.funeral.serviceList.dto | FuneralCompanyImgDto | | 장례 업체 이미지 DTO |
| **Funeral.Reserve** | | | | |
| 20 | ks51team03.funeral.reserve.controller | ReserveController | callbackReceive | 결제 콜백 처리 |
| 21 | ks51team04.funeral.reserve.controller | ReserveController | reserveInfo | 결제 전 개인 정보 확인 |
| 22 | ks51team05.funeral.reserve.controller | ReserveController | reservePayment | 결제 전 예약 정보 확인 |
| 23 | ks51team06.funeral.reserve.controller | ReserveController | reserveConfirm | 결제 확인 |
| 24 | ks51team07.funeral.reserve.controller | ReserveService | funeralReserve | 장례 서비스 예약 내역 Service |
| 25 | ks51team03.funeral.reserve.service | ReserveService | funeralReserveInfo | 장례 서비스 예약자 인적사항 Serivce |
| 26 | ks51team04.funeral.reserve.service | ReserveService | handlePaymentCallback | 결제 콜백 처리 Serivce |
| 27 | ks51team05.funeral.reserve.service | ReserveService | generateFpcode | 결제 코드 처리 Service |
| 28 | ks51team03.funeral.reserve.entity | Payment | | 결제 처리 PG사 Entity |
| 29 | ks51team03.funeral.reserve.mapper | ReserveMapper | funeralReserve | 장례 서비스 예약 내역 Mapper |
| 30 | ks51team04.funeral.reserve.mapper | ReserveMapper | getLastFuneralServiceCode | 장례 예약 코드 Mapper |
| 31 | ks51team05.funeral.reserve.mapper | ReserveMapper | funeralReserveInfo | 장례 예약자 인적사항 Mapper |
| 32 | ks51team06.funeral.reserve.mapper | ReserveMapper | funeralReserveServiceInfo | 장례 예약 정보 확인 Mapper |
| 33 | ks51team07.funeral.reserve.mapper | ReserveMapper | insertPayment | 장례 결제 Mapper |
| 34 | ks51team03.funeral.reserve.dto | ReserveDto | | 장례 예약 DTO |
| 35 | ks51team04.funeral.reserve.dto | ReserveInfoDto | | 장례 예약자 인적사항 DTO |
| 36 | ks51team05.funeral.reserve.dto | ReserveMemberPet | | 장례 예약 반려동물 DTO |
| 37 | ks51team06.funeral.reserve.dto | ReservePaymentDto | | 장례 결제 DTO |
| **Funeral.Payment** | | | | |
| 38 | ks51team03.funeral.payment.controller | PaymentController | confirmPayment | 결제 리스트 Controller |
| 39 | ks51team04.funeral.payment.controller | PaymentController | confirmPaymentDetail | 결제 상세 내역 리스트 |
| 40 | ks51team05.funeral.payment.controller | PaymentController | paymentServiceListCompany | 업체별 장례 서비스 결제 리스트 |
| 41 | ks51team06.funeral.payment.controller | PaymentService | confirmPayment | 결제 리스트 Serivce |
| 42 | ks51team03.funeral.payment.service | PaymentService | getPaymentDetail | 결제 상세 내역 리스트 Service |
| 43 | ks51team03.funeral.payment.service | PaymentService | getPaymentServiceCompnay | 업체별 장례 서비스 결제 리스트 Service |
| 44 | ks51team03.funeral.payment.mapper | PaymentMapper | confirmPayment | 결제 리스트 Mapper |
| 45 | ks51team03.funeral.payment.mapper | PaymentMapper | getPaymentDetail | 결제 상세 내역 리스트 Mapper |
| 46 | ks51team03.funeral.payment.mapper | PaymentMapper | getPaymentServiceCompnay | 업체별 장례 서비스 결제 리스트 Mapper |
| 47 | ks51team03.funeral.payment.dto | PaymentDto | | 결제 정보 DTO |

## 2. Member 담당 = 한송이

| Member | | | | |
|---|---|---|---|---|
| 48 | ks51team03.member.controller | MemberController | login | 회원 로그인 기능 Controller |
| 49 | ks51team04.member.controller | MemberController | loginGara | 버튼 클릭 시 권한별 로그인 기능 |
| 50 | ks51team05.member.controller | MemberController | logout | 회원 로그아웃 |
| 51 | ks51team06.member.controller | MemberController | idCheck | 아이디 중복 체크 |
| 52 | ks51team07.member.controller | MemberController | insertMember | 회원 가입 |
| 53 | ks51team08.member.controller | MemberController | ceoIdCheck | 업체 대표 아이디 중복 체크 |
| 54 | ks51team09.member.controller | MemberController | userMainPage | 사용자 메인 페이지 기능 |
| 55 | ks51team10.member.controller | MemberController | userMyPageMemberInfo | 메인 페이지 회원 정보 조회 |
| 56 | ks51team11.member.controller | MemberController | modifyMember | 회원 정보 수정 |
| 57 | ks51team12.member.controller | MemberController | userMyPageMyQandR | 사용자 페이지 문의 목록 조회 |
| 58 | ks51team13.member.controller | MemberController | userQuestionModify | 사용자 페이지 문의 수정 |
| 59 | ks51team14.member.controller | MemberController | userQuestionModifyAction | 사용자 페이지 문의 |
| 60 | ks51team15.member.controller | MemberController | userQuestionDelete | 사용자 페이지 문의 수정 |
| 61 | ks51team16.member.controller | MemberController | userReviewModify | 사용자 페이지 리뷰 수정 |
| 62 | ks51team17.member.controller | MemberController | userMyPageListPet | 사용자 페이지 반려동물 목록 조회 |
| 63 | ks51team18.member.controller | MemberController | disableNotification | |
| 64 | ks51team03.member.service | MemberServiceImpl | getInform | 회원 알림 조회 ServiceImpl |
| 65 | ks51team04.member.service | MemberServiceImpl | getInformCount | 회원의 알림 수 조회 ServiceImpl |
| 66 | ks51team05.member.service | MemberServiceImpl | memberReviewDelete | 회원의 리뷰 삭제 ServiceImpl |
| 67 | ks51team06.member.service | MemberServiceImpl | memberReviewModify | 회원의 특정 리뷰 수정 ServiceImpl |
| 68 | ks51team07.member.service | MemberServiceImpl | getCompanyReviewByRevCode | 회원의 특정 리뷰 검색 ServiceImpl |
| 69 | ks51team08.member.service | MemberServiceImpl | getCompanyReview | 회원 리뷰 검색 ServiceImpl |
| 70 | ks51team09.member.service | MemberServiceImpl | memberQuestionDelete | 회원 문의 삭제 ServiceImpl |
| 71 | ks51team10.member.service | MemberServiceImpl | memberQuestionModify | 회원 문의 수정 ServiceImpl |
| 72 | ks51team11.member.service | MemberServiceImpl | getQuestionById | 회원 문의 조회 ServiceImpl |
| 73 | ks51team12.member.service | MemberServiceImpl | getLoginHistory | 로그인 이력 조회 ServiceImpl |
| 74 | ks51team13.member.service | MemberServiceImpl | getSearchList | 회원 검색 조회 ServiceImpl |
| 75 | ks51team14.member.service | MemberServiceImpl | removeMember | 회원 탈퇴 ServiceImpl |
| 76 | ks51team15.member.service | MemberServiceImpl | checkMemberInfo | 특정 회원 확인 ServiceImpl |
| 77 | ks51team16.member.service | MemberServiceImpl | updateMember | 특정 회원 수정 ServiceImpl |
| 78 | ks51team17.member.service | MemberServiceImpl | getMemberInfoById | 특정 회원 조회 ServiceImpl |
| 79 | ks51team18.member.service | MemberServiceImpl | insertMember | 회원가입 프로세스 ServiceImpl |
| 80 | ks51team19.member.service | MemberServiceImpl | getMemberLevelList | 회원 등급 조회 ServiceImpl |
| 81 | ks51team20.member.service | MemberServiceImpl | getMemberList | 회원 조회 ServiceImpl |
| 82 | ks51team21.member.service | MemberServiceImpl | getCompanyCodeByMemberId | 업체 회원 조회 ServiceImpl |
| 83 | ks51team03.member.mapper | MemberMapper | getInform | 회원 알림 내용 조회 Mapper |
| 84 | ks51team04.member.mapper | MemberMapper | getInformCount | 회원 알림 수 조회 Mapper |
| 85 | ks51team05.member.mapper | MemberMapper | memberReviewDelete | 회원 리뷰 삭제 Mapper |
| 86 | ks51team06.member.mapper | MemberMapper | memberReviewModify | 회원 리뷰 수정 Mapper |
| 87 | ks51team07.member.mapper | MemberMapper | getCompanyReviewByRevCode | 회원 리뷰 수정을 위한 특정 리뷰 검색 Mapper |
| 88 | ks51team08.member.mapper | MemberMapper | getCompanyReview | 회원 리뷰 검색 Mapper |
| 89 | ks51team09.member.mapper | MemberMapper | deleteAnswersByQuesNum | 회원 문의 답변 삭제 Mapper |
| 90 | ks51team10.member.mapper | MemberMapper | memberQuestionDelete | 회원 문의 삭제 Mapper |
| 91 | ks51team11.member.mapper | MemberMapper | memberQuestionModify | 회원 문의 수정 Mapper |
| 92 | ks51team12.member.mapper | MemberMapper | getQuestionById | 회원 문의 검색 Mapper |
| 93 | ks51team13.member.mapper | MemberMapper | getLoginHistoryCnt | 로그인 테이블 행의 개수 조회 Mapper |
| 94 | ks51team14.member.mapper | MemberMapper | getLoginHistory | 로그인 이력 조회 Mapper |
| 95 | ks51team15.member.mapper | MemberMapper | getSearchList | 회원 검색 조회 Mapper |
| 96 | ks51team16.member.mapper | MemberMapper | removeMemberById | 회원 탈퇴 Mapper |
| 97 | ks51team17.member.mapper | MemberMapper | removeLoginHistoryById | 회원 로그인 이력 삭제 Mapper |
| 98 | ks51team18.member.mapper | MemberMapper | updateMember | 회원 수정 Mapper |
| 99 | ks51team19.member.mapper | MemberMapper | getMemberInfoById | 특정 회원 정보 조회 Mapper |
| 100 | ks51team20.member.mapper | MemberMapper | insertMember | 회원 가입 Mapper |
| 101 | ks51team21.member.mapper | MemberMapper | idCheck | 아이디 중복 체크 Mapper |
| 102 | ks51team22.member.mapper | MemberMapper | getMemberLevelList | 회원등급 조회 Mapper |
| 103 | ks51team23.member.mapper | MemberMapper | getMemberList | 회원목록 조회 Mapper |
| 104 | ks51team24.member.mapper | MemberMapper | IncreasePetByMemberId | 회원의 반려동물 수 추가 Mapper |
| 105 | ks51team25.member.mapper | MemberMapper | DeclinePetByMemberId | 회원의 반려동물 수 감소 Mapper |
| 106 | ks51team26.member.mapper | MemberMapper | updateInformStatus | 회원의 알림 확인 Mapper |
| 107 | ks51team03.member.dto | Member | | 회원 정보 DTO |
| 108 | ks51team04.member.dto | MemberLevel | | 회원 등급 DTO |
| 109 | ks51team05.member.dto | Search | | 회원 정보 검색 DTO |

## 3. Board 담당 : 한송이

| Board | | | | |
|---|---|---|---|---|
| 213 | ks51team13.board.controller | BoardController | getNoticeBoardList | 일반 게시글 목록 조회 |
| 214 | ks51team13.board.controller | BoardController | searchNoticeBoardList | 일반 게시글 목록 카테고리별로 검색 |
| 215 | ks51team13.board.controller | BoardController | boardWriteNormalPage | 일반 게시글 등록 |
| 216 | ks51team13.board.controller | BoardController | boardViewNormalPage | 일반 게시글 열람 |
| 217 | ks51team13.board.controller | BoardController | boardEditNormalPage | 일반 게시글 수정 |
| 218 | ks51team13.board.controller | BoardController | recommendBoard | 게시글 추천 |
| 219 | ks51team13.board.controller | BoardController | getGalleryBoardList | 갤러리 게시글 목록 조회 |
| 220 | ks51team13.board.controller | BoardController | searchGalleryBoardList | 갤러리 게시글 목록 카테고리별로 검색 |
| 221 | ks51team13.board.controller | BoardController | boardWriteGalleryPage | 갤러리 게시글 등록 |
| 222 | ks51team13.board.controller | BoardController | boardViewGalleryPage | 갤러리 게시글 열람 |
| 223 | ks51team13.board.controller | BoardController | boardEditGalleryPage | 갤러리 게시글 수정 |
| 224 | ks51team13.board.controller | BoardImgController | imageUpload | ckEditor 이미지 업로드 |
| 225 | ks51team13.board.controller | BoardImgController | getImage | ckEditor 이미지 불러오기 |
| 226 | ks51team13.board.dto | Board | Board | 게시판 DTO |
| 227 | ks51team13.board.dto | BoardCategory | BoardCategory | 게시판 카테고리 DTO |
| 228 | ks51team13.board.dto | Criteria | Criteria | 게시판 페이징 DTO |
| 229 | ks51team13.board.dto | NoticeBoard | NoticeBoard | 게시글 DTO |
| 230 | ks51team13.board.dto | NBoardSearch | NBoardSearch | 게시글 검색 카테고리 DTO |
| 231 | ks51team13.board.dto | NBoardImg | NBoardImg | 게시글 이미지 DTO |
| 232 | ks51team13.board.mapper | BoardMapper | insertNBoard | 게시글 등록 |
| 233 | ks51team14.board.mapper | BoardMapper | getNBoardCode | 게시글 코드 도출 |
| 234 | ks51team15.board.mapper | BoardMapper | getBoardSearchList | 게시글 검색 |
| 235 | ks51team16.board.mapper | BoardMapper | getNBoardByNBCode | 게시글 코드로 게시글 조회 |
| 236 | ks51team17.board.mapper | BoardMapper | increaseViewByNBCode | 게시글 코드에 해당하는 게시글 조회수 증가 |
| 237 | ks51team18.board.mapper | BoardMapper | increaseRecByNBCode | 게시글 코드에 해당하는 게시글 추천수 증가 |
| 238 | ks51team19.board.mapper | BoardMapper | updateNBaord | 게시글 수정 |
| 239 | ks51team20.board.mapper | BoardMapper | getBCTValueNameByBCTCode | 게시글로 게시글 카테고리 이름 조회 |
| 240 | ks51team21.board.mapper | BoardMapper | getBoardCateCodeByBCTValue | 게시글로 게시글 카테고리 코드 조회 |
| 241 | ks51team22.board.mapper | BoardMapper | getBoardCodeByNBoard | 게시글로 게시글 보드 코드 도출 |
| 242 | ks51team23.board.mapper | BoardMapper | getBCTValueByNBCode | 게시글 코드로 게시판 이름 조회 |
| 243 | ks51team24.board.mapper | BoardMapper | getBoardCodeByBCTValue | 게시글 카테고리 이름으로 게시글 보드 코드 조회 |
| 244 | ks51team25.board.mapper | BoardMapper | getBoardInfoByBCTValue | 게시글 카테고리 이름으로 게시글 설명 조회 |
| 245 | ks51team26.board.mapper | BoardMapper | getMainNoticeBoard | 해당 게시판 게시글 최신순으로 조회 |
| 246 | ks51team27.board.mapper | BoardMapper | getMainViewBoard | 게시글 조회수 순으로 조회 |
| 247 | ks51team28.board.mapper | BoardMapper | getMainRecBoard | 게시글 추천수 순으로 조회 |
| 248 | ks51team29.board.mapper | BoardMapper | getMainLatestBoard | 게시글 최신순으로 조회 |
| 249 | ks51team30.board.mapper | BoardMapper | insertnBoardImg | 게시글 이미지 등록 |
| 250 | ks51team31.board.mapper | BoardMapper | getNBoardImgByNBCode | 게시글 코드로 게시글 이미지 경로 조회 |
| 251 | ks51team32.board.mapper | BoardMapper | deleteFromNBoardImg | 이미지 경로로 보드 이미지 테이블에 이미지 삭제 |
| 252 | ks51team33.board.mapper | BoardMapper | deleteFromFiles | 이미지 경로로 보드 파일 테이블에 이미지 삭제 |
| 253 | ks51team13.board.service | BoardService | insertNBoard | 게시글 등록 |
| 254 | ks51team14.board.service | BoardService | getNBoardCode | 게시글 코드 도출 |
| 255 | ks51team15.board.service | BoardService | getBoardSearchList | 게시글 검색 |
| 256 | ks51team16.board.service | BoardService | getNBoardByNBCode | 게시글 코드로 게시글 조회 |
| 257 | ks51team17.board.service | BoardService | increaseViewByNBCode | 게시글 코드에 해당하는 게시글 조회수 증가 |
| 258 | ks51team18.board.service | BoardService | increaseRecByNBCode | 게시글 코드에 해당하는 게시글 추천수 증가 |
| 259 | ks51team19.board.service | BoardService | updateNBaord | 게시글 수정 |
| 260 | ks51team20.board.service | BoardService | getBCTValueNameByBCTCode | 게시글로 게시글 카테고리 이름 조회 |
| 261 | ks51team21.board.service | BoardService | getBoardCateCodeByBCTValue | 게시글로 게시글 카테고리 코드 조회 |
| 262 | ks51team22.board.service | BoardService | getBoardCodeByNBoard | 게시글로 게시글 보드 코드 도출 |
| 263 | ks51team23.board.service | BoardService | getBCTValueByNBCode | 게시글 코드로 게시판 이름 조회 |
| 264 | ks51team24.board.service | BoardService | getBoardCodeByBCTValue | 게시글 카테고리 이름으로 게시글 보드 코드 조회 |
| 265 | ks51team25.board.service | BoardService | getBoardInfoByBCTValue | 게시글 카테고리 이름으로 게시글 설명 조회 |
| 266 | ks51team26.board.service | BoardService | getMainNoticeBoard | 해당 게시판 게시글 최신순으로 조회 |
| 267 | ks51team27.board.service | BoardService | getMainViewBoard | 게시글 조회수 순으로 조회 |
| 268 | ks51team28.board.service | BoardService | getMainRecBoard | 게시글 추천수 순으로 조회 |
| 269 | ks51team29.board.service | BoardService | getMainLatestBoard | 게시글 최신순으로 조회 |
| 270 | ks51team30.board.service | BoardService | insertnBoardImg | 게시글 이미지 등록 |
| 271 | ks51team31.board.service | BoardService | getNBoardImgByNBCode | 게시글 코드로 게시글 이미지 경로 조회 |
| 272 | ks51team32.board.service | BoardService | deleteFromNBoardImg | 이미지 경로로 보드 이미지 테이블에 이미지 삭제 |
| 273 | ks51team33.board.service | BoardService | deleteFromFiles | 이미지 경로로 보드 파일 테이블에 이미지 삭제 |

Pet & Mail 담당 : 한송이

| Pet | | | | |
|---|---|---|---|---|
| 274 | ks51team13.pet.controller | PetController | insertPet | 반려동물 등록 |
| 275 | ks51team13.pet.controller | BoardController | updatePet | 반려동물 정보 수정 |
| 276 | ks51team13.pet.controller | BoardController | removePet | 반려동물 삭제 |
| 277 | ks51team13.pet.controller | BoardController | userMyPagePetInfo | 마이페이지 회원 등록 반려동물 조회 |
| 278 | ks51team13.pet.dto | Pet | Pet | 반려동물 DTO |
| 279 | ks51team13.pet.mapper | PetMapper | insertPet | 반려동물 등록 |
| 280 | ks51team13.pet.mapper | PetMapper | getPetCode | 반려동물 코드 조회 |
| 281 | ks51team13.pet.mapper | PetMapper | getPetInfoByMemberId | 회원 아이디로 반려동물 검색 |
| 282 | ks51team13.pet.mapper | PetMapper | getPetInfoByPetCode | 반려동물 코드로 반려동물 정보 조회 |
| 283 | ks51team13.pet.mapper | PetMapper | updatePet | 반려동물 정보 수정 |
| 284 | ks51team13.pet.mapper | PetMapper | removePet | 반려동물 삭제 |
| 285 | ks51team13.pet.service | PetService | insertPet | 반려동물 등록 |
| 286 | ks51team13.pet.service | PetService | getPetCode | 반려동물 코드 조회 |
| 287 | ks51team13.pet.service | PetService | getPetInfoByMemberId | 회원 아이디로 반려동물 검색 |
| 288 | ks51team13.pet.service | PetService | getPetInfoByPetCode | 반려동물 코드로 반려동물 정보 조회 |
| 289 | ks51team13.pet.service | PetService | updatePet | 반려동물 정보 수정 |
| 290 | ks51team13.pet.service | PetService | removePet | 반려동물 삭제 |
| Mail | | | | |
| 291 | ks51team13.mail | MailController | MailSend | 이메일 전송 |
| 292 | ks51team13.mail | MailController | FindIdMailSend | 아이디 찾기 이메일 전송 |
| 293 | ks51team13.pet.controller | MailController | FildPwMailSend | 비밀번호 찾기 이메일 전송 |
| 294 | ks51team13.pet.controller | MailResult | MailResult | 메일 전송 결과DTO |
| 295 | ks51team13.pet.dto | MailService | createNumber | 메일로 전송할 랜덤한 번호 생성 |
| 296 | ks51team13.pet.mapper | PetMapper | CreateMail | 전송할 메일 생성 |
| 297 | ks51team13.pet.mapper | PetMapper | sendMail | 메일 전송 |

| Company (part 1)<br>Controller, Service | 담당 : 박은수 |
|---|---|

| Company | | | | |
|---|---|---|---|---|
| 110 | ks51team03.company.controller | CompanyController | companyInfo | 업체 정보 조회 Controller |
| 111 | ks51team04.company.controller | CompanyController | getOpeningHoursForDay | 업체 운영 시간 등록 |
| 112 | ks51team05.company.controller | CompanyController | companyModify | 업체 정보 수정 |
| 113 | ks51team06.company.controller | CompanyController | companyStaffSetting | 업체 직원 등록 |
| 114 | ks51team07.company.controller | CompanyController | deleteStaff | 업체 직원 삭제 |
| 115 | ks51team08.company.controller | CompanyController | acceptStaff | 업체 직원 신청 |
| 116 | ks51team09.company.controller | CompanyController | rejectStaff | 업체 직원 승인 거절 |
| 117 | ks51team10.company.controller | CompanyController | companySignUp | 업체 직원 신청 |
| 118 | ks51team11.company.controller | CompanyController | companyQuestion | 업체 문의 조회 |
| 119 | ks51team12.company.controller | CompanyController | submitQuestion | 업체 문의 등록 |
| 120 | ks51team13.company.controller | CompanyController | getQctenum | 업체 문의 분류 |
| 121 | ks51team14.company.controller | CompanyController | submitReview | 업체 리뷰 등록 |
| 122 | ks51team15.company.controller | CompanyController | deleteQuestion | 업체 문의 삭제 |
| 123 | ks51team16.company.controller | CompanyController | companyQuestionAnswer | 업체 문의 답변 |
| 124 | ks51team17.company.controller | CompanyController | updateAnswer | 업체 문의 수정 |
| 125 | ks51team18.company.controller | CompanyController | deleteReview | 업체 문의 삭제 |
| 126 | ks51team19.company.controller | CompanyController | companySendAlarm | 업체 알림 |
| 127 | ks51team20.company.controller | CompanyController | sendAlarm | 업체 알림 등록 |
| 128 | ks51team21.company.controller | CompanyController | insertCompany | 업체 등록 |
| 129 | ks51team03.company.service | CompanyService | getCompanyImgByCcode | 업체 대표 이미지 조회 Service |
| 130 | ks51team04.company.service | CompanyService | insertComInformReciPient | 업체 알림 수신자 등록 |
| 131 | ks51team05.company.service | CompanyService | insertComInform | 업체 알림 내용 저장 |
| 132 | ks51team06.company.service | CompanyService | getCompanyLikeMemberByCcode | 업체 구독자 리스트 |
| 133 | ks51team07.company.service | CompanyService | deleteReview | 업체 리뷰 삭제 |
| 134 | ks51team08.company.service | CompanyService | addReviewWithFile | 리뷰 파일 등록 |
| 135 | ks51team09.company.service | CompanyService | addCompanyWithFile | 업체 파일 등록 |
| 136 | ks51team10.company.service | CompanyService | addCompanyImg | 업체 대표 이미지 등록 |
| 137 | ks51team11.company.service | CompanyService | addReview | 리뷰 등록하기 |
| 138 | ks51team12.company.service | CompanyService | addQuestion | 문의 등록하기 |
| 139 | ks51team13.company.service | CompanyService | insertStaff | 직원 신청 등록 |
| 140 | ks51team14.company.service | CompanyService | getCompanyQuestionById | 특정 문의 조회 |
| 141 | ks51team15.company.service | CompanyService | getAnswerByQuesNum | 특정 문의 답변 조회 |
| 142 | ks51team16.company.service | CompanyService | updateAnswer | 문의 답변 수정 로직 |
| 143 | ks51team17.company.service | CompanyService | modifyCompany | 업체 정보 수정 |
| 144 | ks51team18.company.service | CompanyService | addAnswer | 문의 답변 등록 |
| 145 | ks51team19.company.service | CompanyService | getCompanyQuestionAnswer | 업체 문의 답변 리스트 반환 |
| 146 | ks51team20.company.service | CompanyService | getCompanyQuestion | 업체 문의 리스트 반환 |
| 147 | ks51team21.company.service | CompanyService | deleteQuestion | 업체 문의 삭제 |
| 148 | ks51team22.company.service | CompanyService | deleteAnswersByQuesNum | 문의 답변 먼저 삭제 |
| 149 | ks51team23.company.service | CompanyService | getCompanyList | 전체 업체 리스트 반환 |
| 150 | ks51team24.company.service | CompanyService | getCompanyOperTime | 업체 운영시간 리스트 반환 |
| 151 | ks51team25.company.service | CompanyService | getCompanyInfoById | 세션 아이디를 통해 업체 리스트 반환 |
| 152 | ks51team26.company.service | CompanyService | getCompanyCodeByMemberId | 회원 아이디로 업체 코드 검색 |
| 153 | ks51team27.company.service | CompanyService | getCompanyInfoByCcode | 업체 코드로 업체 정보 조회 |
| 154 | ks51team28.company.service | CompanyService | getCompanyListByKeyWord | 업체 리스트 키워드로 반환 |
| 155 | ks51team29.company.service | CompanyService | getComMapByCCode | 좌표 반환 |
| 156 | ks51team30.company.service | CompanyService | getAvgReviewScore | 별점 평균 반환 |
| 157 | ks51team31.company.service | CompanyService | getCompanyReview | 업체 리뷰 반환 |
| 158 | ks51team32.company.service | CompanyService | getCompanyReviewCount | 업체 리뷰수 반환 |
| 159 | ks51team33.company.service | CompanyService | getStaffSingList | 직원 신청 회원 조회 |
| 160 | ks51team34.company.service | CompanyService | getStaffList | 직원 리스트 반환 |
| 161 | ks51team35.company.service | CompanyService | updateLevel | 직원 승인 전 회원 레벨 변경 |
| 162 | ks51team36.company.service | CompanyService | acceptStaff | 직원 승인 전 회원 레벨 변경 |
| 163 | ks51team37.company.service | CompanyService | deleteStaff | 직원 해고 |
| 164 | ks51team38.company.service | CompanyService | insertCompany | 업체 등록 |

| Company (part 2)<br>Mapper, Dto | 담당 : 박은수 |
|---|---|

| Company | | | | |
|---|---|---|---|---|
| 165 | ks51team03.company.mapper | CompanyMapper | getCompanyImgByCcode | 업체 대표 이미지 리스트 조회 Mapper |
| 166 | ks51team04.company.mapper | CompanyMapper | insertComInformReciPient | 업체 알림 수신자 등록 |
| 167 | ks51team05.company.mapper | CompanyMapper | insertComInform | 업체 알림 내용 저장 |
| 168 | ks51team06.company.mapper | CompanyMapper | getCompanyLikeMemberByCcode | 업체 구독자 리스트 |
| 169 | ks51team07.company.mapper | CompanyMapper | avgReviewScore | 업체 리뷰 별점 평균 반환 |
| 170 | ks51team08.company.mapper | CompanyMapper | deleteReview | 업체의 리뷰 삭제 |
| 171 | ks51team09.company.mapper | CompanyMapper | insertCompanyImg | 업체 대표 이미지 등록 |
| 172 | ks51team10.company.mapper | CompanyMapper | insertReview | 리뷰 등록하기 |
| 173 | ks51team11.company.mapper | CompanyMapper | insertQuestion | 문의 등록하기 |
| 174 | ks51team12.company.mapper | CompanyMapper | deleteAnswersByQuesNum | 업체의 문의 삭제 전 답변부터 삭제 |
| 175 | ks51team13.company.mapper | CompanyMapper | deleteQuestion | 업체 문의 삭제 |
| 176 | ks51team14.company.mapper | CompanyMapper | getAnswerByQuesNum | 문의 답변 검색 |
| 177 | ks51team15.company.mapper | CompanyMapper | updateAnswer | 문의 답변 수정 로직 |
| 178 | ks51team16.company.mapper | CompanyMapper | insertAnswer | 문의 답변 등록 |
| 179 | ks51team17.company.mapper | CompanyMapper | insertStaff | 직원 신청 로직 |
| 180 | ks51team18.company.mapper | CompanyMapper | modifyCompany | 업체 수정 |
| 181 | ks51team19.company.mapper | CompanyMapper | getCompanyQuestionAnswer | 업체 문의 답변 리스트 가져오기 |
| 182 | ks51team20.company.mapper | CompanyMapper | getCompanyQuestion | 업체 문의 리스트 가져오기 |
| 183 | ks51team21.company.mapper | CompanyMapper | getCompanyList | 전체 업체 리스트 가져오기 |
| 184 | ks51team22.company.mapper | CompanyMapper | getCompanyQuestionById | 특정 문의 조회 |
| 185 | ks51team23.company.mapper | CompanyMapper | getCompanyOperTime | 업체 운영 시간리스트 가져오기 |
| 186 | ks51team24.company.mapper | CompanyMapper | getCompanyInfoById | 업체 정보 아이디로 가져오기 |
| 187 | ks51team25.company.mapper | CompanyMapper | getCompanyCodeByMemberId | 회원아이디로 가져오기 |
| 188 | ks51team26.company.mapper | CompanyMapper | getCompanyInfoByCcode | 업체 코드로 업체 정보 가져오기 |
| 189 | ks51team27.company.mapper | CompanyMapper | getCompanyListByKeyWord | 업체 리스트 키워드로 가져오기 |
| 190 | ks51team28.company.mapper | CompanyMapper | getComMapByCCode | 업체 좌표 가져오기 |
| 191 | ks51team29.company.mapper | CompanyMapper | getStaffSignList | 신청 직원 가져오기 |
| 192 | ks51team30.company.mapper | CompanyMapper | getStaffList | 직원 리스트 가져오기 |
| 193 | ks51team31.company.mapper | CompanyMapper | updateLevel | 직원 승인 전 회원 레벨 변경 |
| 194 | ks51team32.company.mapper | CompanyMapper | acceptStaff | 직원 승인 |
| 195 | ks51team33.company.mapper | CompanyMapper | deleteStaff | 직원 해고 |
| 196 | ks51team34.company.mapper | CompanyMapper | getCompanyByMemberId | 아이디로 업체 정보 가져오기 |
| 197 | ks51team35.company.mapper | CompanyMapper | getCompanyReview | 업체 코드로 해당 업체 리뷰 불러오기 |
| 198 | ks51team36.company.mapper | CompanyMapper | getCompanyReviewCount | 업체 코드로 해당 업체 리뷰수 불러오기 |
| 199 | ks51team37.company.mapper | CompanyMapper | insertCompany | 업체 등록 |
| 200 | ks51team38.company.mapper | CompanyMapper | getCompanyCode | 업체 코드 도출 |
| 201 | ks51team39.company.mapper | CompanyMapper | updateCeo | 업체 대표 권한 변경 |
| 202 | ks51team03.company.dto | ComInform | | 업체 알림 |
| 203 | ks51team04.company.dto | ComInformReciPient | | 업체 알림 확인 |
| 204 | ks51team05.company.dto | ComLike | | 업체 구독자 |
| 205 | ks51team06.company.dto | ComMap | | 업체 좌표 |
| 206 | ks51team07.company.dto | ComOperTime | | 운영 시간 |
| 207 | ks51team08.company.dto | Company | | 업체 정보 |
| 208 | ks51team09.company.dto | CompanyImg | | 업체 이미지 |
| 209 | ks51team10.company.dto | ComQuestion | | 업체 문의 |
| 210 | ks51team11.company.dto | ComQuestionAnswer | | 업체 문의 답변 |
| 211 | ks51team12.company.dto | ComReview | | 업체 리뷰 |
| 212 | ks51team13.company.dto | ComStaff | | 업체 직원 |

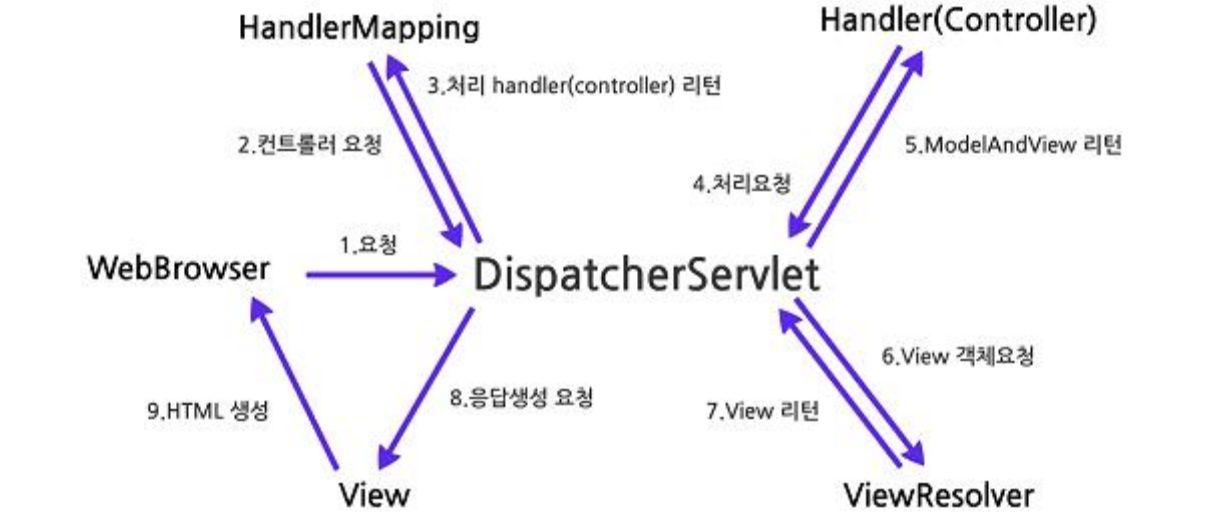
## 【 5. 개발 환경 】

*개발 / 서비스 환경*

| * | 구 분 | 개발 환경 | | 서비스 환경 |
|---|---|---|---|---|
| S / W | OS | Windows 10 | | Linux - CentOS Stream release 8 |
| | 웹 브 라 우 저 | Chrome, Safari | | Chrome, Safari, Microsoft Edge, Firefox |
| | WAS | apache-tomcat-10.x | | apache-tomcat-10.x |
| | 개발언어 | 서버 | JDK17 / Servlet 6.x / UTF-8 | - |
| | | 클라이언트 | JavaScript / HTML5 / CSS / thymeleaf / jQuery | |
| | Framework | bootstrap 5 / Spring Framework 6.x / Mybatis 3.x | | |
| | DBMS | MySQL 8.0.26 | | MySQL 8.0.26 |
| | Tools | 소프트웨어 개발도구 | JDK 17 | OpenJDK 17.x |
| | | 통합 개발도구 | Eclipse, Spring Tool Suite, Intellij IDEA, Visual Studio Code | |

## 【 6. 어플리케이션 】

*어플리케이션 구성*



```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
	<modelVersion>4.0.0</modelVersion>
	<parent>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-parent</artifactId>
		<version>3.3.1</version>
		<relativePath/> <!-- lookup parent from repository -->
	</parent>
	<groupId>ksmart</groupId>
	<artifactId>ks51team03</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>ks51team03</name>
	<description>teamproject for Spring Boot</description>
	<url/>
	<licenses>
		<license/>
	</licenses>
	<developers>
		<developer/>
	</developers>
	<scm>
		<connection/>
		<developerConnection/>
		<tag/>
		<url/>
	</scm>
	<properties>
		<java.version>17</java.version>
	</properties>
	<dependencies>
		<dependency>
			<groupId>org.springframework.boot</groupId>
```

```
        <artifactId>spring-boot-starter-data-jdbc</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-thymeleaf</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
        <groupId>org.mybatis.spring.boot</groupId>
        <artifactId>mybatis-spring-boot-starter</artifactId>
        <version>3.0.3</version>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-devtools</artifactId>
        <scope>runtime</scope>
        <optional>true</optional>
    </dependency>
    <dependency>
        <groupId>com.mysql</groupId>
        <artifactId>mysql-connector-j</artifactId>
        <scope>runtime</scope>
    </dependency>
    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <optional>true</optional>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.mybatis.spring.boot</groupId>
        <artifactId>mybatis-spring-boot-starter-test</artifactId>
        <version>3.0.3</version>
        <scope>test</scope>
```

```
		</dependency>
		<!--	https://mvnrepository.com/artifact/nz.net.ultraq.thymeleaf/thymeleaf-layout-dialect -->
		<dependency>
			<groupId>nz.net.ultraq.thymeleaf</groupId>
			<artifactId>thymeleaf-layout-dialect</artifactId>
		</dependency>
		<!-- https://mvnrepository.com/artifact/org.bgee.log4jdbc-log4j2/log4jdbc-log4j2-jdbc4.1 -->
		<dependency>
			<groupId>org.bgee.log4jdbc-log4j2</groupId>
			<artifactId>log4jdbc-log4j2-jdbc4.1</artifactId>
			<version>1.16</version>
		</dependency>
		<dependency>
			<groupId>com.h2database</groupId>
			<artifactId>h2</artifactId>
			<scope>runtime</scope>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-mail</artifactId>
		</dependency>
		<dependency>
			<groupId>javax.servlet</groupId>
			<artifactId>jstl</artifactId>
			<version>1.2</version>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-data-jpa</artifactId>
		</dependency>
			<!-- fileupload(사진 업로드) -->
		<dependency>
			<groupId>commons-fileupload</groupId>
			<artifactId>commons-fileupload</artifactId>
			<version>1.3.1</version>
		</dependency>
		<dependency>
			<groupId>commons-io</groupId>
			<artifactId>commons-io</artifactId>
```

```
                        <version>2.4</version>
                </dependency>
        </dependencies>
        <build>
                <plugins>
                        <plugin>
                                <groupId>org.springframework.boot</groupId>
                                <artifactId>spring-boot-maven-plugin</artifactId>
                                <configuration>
                                        <excludes>
                                                <exclude>
                                                        <groupId>org.projectlombok</groupId>
                                                        <artifactId>lombok</artifactId>
                                                </exclude>
                                        </excludes>
                                </configuration>
                        </plugin>
                </plugins>
        </build>
</project>
```

application.properties

```
server.port=8088
# thymeleaf cache
spring.thymeleaf.cache=false

# Database
spring.datasource.driver-class-name=net.sf.log4jdbc.sql.jdbcapi.DriverSpy


# commit
spring.datasource.url=jdbc:log4jdbc:mysql://4:3306/ks48team02db?serverTimezone=UTC&characterEncoding=UTF8
spring.datasource.username=ks48team02id
spring.datasource.password=ks48team02pw

# mybatis
# clsspath: -> src/main/resources/
mybatis.mapper-locations=classpath:mapper/**/*.xml

# mybatis DTO
mybatis.type-aliases-package=ksmart.ks48team0admin.dto,ksmart.ks48team0seller.dto,ksmart.ks48team0user.dto,ksmart.ks48team0common.dto
```

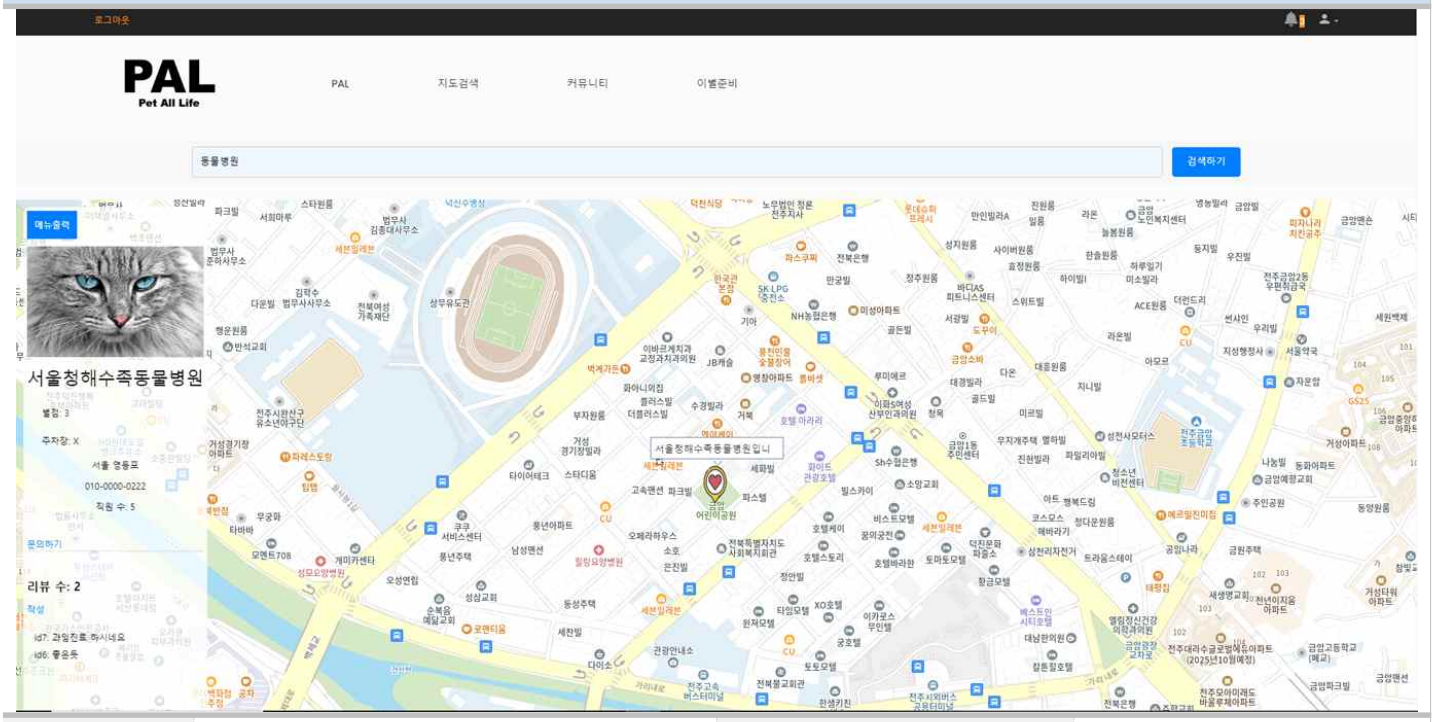
```
# logback
#logging.config=classpath:logback-spring.xml

# mac
 logging.config=classpath:logback-spring-mac.xml

#JSSESIONID
server.servlet.session.cookie.http-only=true
server.servlet.session.tracking-modes=cookie


# Enable multipart uploads
spring.servlet.multipart.enabled=true
# Max file size
spring.servlet.multipart.max-file-size=200MB
# Max Request Size
spring.servlet.multipart.max-request-size=25MB
#spring json
spring.mvc.converters.preferred-json-mapper=gson

#api key
kakao.pay.admin-key=c35d85cdeaa02e836d27e3ac705c60
```

# 【 6. 주요 기능별 정리 】

*개인 기능*

| Fuction 1 | 지도 |
|---|---|

Image




| Package | Ks51team03/map | Structure | Controller |
|---|---|---|---|

```
package ks51team03.map.controller;
import com.fasterxml.jackson.core.JsonProcessingException;
import com.fasterxml.jackson.databind.ObjectMapper;
import jakarta.servlet.http.HttpSession;
import ks51team03.company.dto.*;
import ks51team03.company.service.CompanyService;
import ks51team03.member.dto.Member;
import ks51team03.member.dto.MemberLike;
import ks51team03.member.service.MemberServiceImpl;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
```

```
import org.springframework.web.servlet.mvc.support.RedirectAttributes;
import java.time.DayOfWeek;
import java.time.LocalDate;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
@Controller
@RequiredArgsConstructor
@Slf4j
public class MapController {
	private final CompanyService companyService;
	private final MemberServiceImpl memberService;
	@GetMapping("/map/map_main")
	public String mapMainPage() {
		return "map/map_main";
	}
	@ResponseBody
	@GetMapping("/map/get_company_list")
	public List<Company> getCompanyList(@RequestParam(value = "keyword", required = false) String
keyword) {
		List<Company> companyList = companyService.getCompanyListByKeyWord(keyword);
		return companyList;
	}
	@PostMapping("/map/get_com_map_list")
	@ResponseBody
	public List<ComMap> getComMapList(@RequestBody List<String> cCodes) {
		List<ComMap> comMapList = new ArrayList<>();
		for (String companyCode : cCodes) {
			ComMap comMap = companyService.getComMapByCCode(companyCode);
			if (comMap != null) {
				comMapList.add(comMap);
			}
		}
		return comMapList;
	}
	@ResponseBody
	@GetMapping("/map/getCompanyInfo")
	public Map<String, Object> getCompanyInfo(@RequestParam("cCode") String cCode) {
		Map<String, Object> response = new HashMap<>();
		int reviewCount = companyService.getCompanyReviewCount(cCode);
```

```
                double avgReviewScore = companyService.getAvgReviewScore(cCode);
                List<String> comImg = companyService.getCompanyImgByCcodeForMap(cCode);
                List<ComReview> reviews = companyService.getCompanyReview(cCode);
                if(reviews != null) {
                        Member revMember =
memberService.getMemberInfoById(reviews.get(0).getMemberId());
                        response.put("revMember", revMember);
                        response.put("reviews", reviews);
                }
                response.put("reviewCount", reviewCount);
                response.put("avgReviewScore", avgReviewScore);
                response.put("comImg", comImg);
                return response;
        }
}
```

| Package | Ks51team03/company | Service | CompanyService |
|---|---|---|---|

```
package ks51team03.company.service;
import ks51team03.company.dto.*;
import ks51team03.company.mapper.CompanyMapper;
import ks51team03.files.dto.FileRequest;
import ks51team03.files.mapper.FileMapper;
import ks51team03.files.util.FileUtils;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import java.util.List;
@Service("CompanyService")
@Transactional
@RequiredArgsConstructor
@Slf4j
public class CompanyService {
    private final CompanyMapper companyMapper;
    private final FileUtils fileUtils;
    private final FileMapper fileMapper;
    // 업체 이미지 조회 지도용
    public List<String> getCompanyImgByCcodeForMap(String cCode){
        return companyMapper.getCompanyImgByCcodeForMap(cCode);
    }
    // 모든 리뷰 조회
```

```
    public List<ComReview> getAllReview(){
        return companyMapper.getAllReview();
    }
    // 전체 업체 리스트 반환
    public List<Company> getCompanyList(){
        return companyMapper.getCompanyList();
    }
    // 업체 코드로 업체 운영시간 리스트 반환
    public List<ComOperTime> getCompanyOperTime(String cCode){
        return companyMapper.getCompanyOperTime(cCode);
    }
    // 좌표 반환
    public ComMap getComMapByCCode(String companyCode) {
        return companyMapper.getComMapByCCode(companyCode);
    }
    // 별점 평균 반환
    public Double getAvgReviewScore(String companyCode) {
        Double avgScore = companyMapper.avgReviewScore(companyCode);
        return (avgScore != null) ? avgScore : 0.0;
    }
    // 업체 리뷰 반환
    public List<ComReview> getCompanyReview(String companyCode) {
        return companyMapper.getCompanyReview(companyCode);
    }
    // 업체 리뷰수 반환
    public int getCompanyReviewCount(String companyCode) {
        int reviewCount = companyMapper.getCompanyReviewCount(companyCode);
        return reviewCount;
    }
}
```

| Package | Ks51team03/company | Xml | CompanyMapper.xml |
|---|---|---|---|

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
                PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
                "https://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="ks51team03.company.mapper.CompanyMapper">
        <resultMap type="ks51team03.company.dto.Company" id="CompanyResultMap">
                <id       column="ccode" property="companyCode" />
                <result column="com_class" property="companyClass" />
                <result column="com_name" property="companyName" />
```

```
        <result column="com_ceo" property="companyCeo" />
        <result column="com_addr" property="companyAddr" />
        <result column="com_addr_detail" property="companyAddrDetail" />
        <result column="com_phone" property="companyPhone" />
        <result column="com_stfcount" property="companyStfCount" />
        <result column="com_email" property="companyEmail" />
        <result column="com_parking" property="companyParking" />
        <result column="com_page" property="companyPage" />
        <result column="com_reg_date" property="companyRegDate" />
        <result column="com_post_num" property="comPostNum" />
        <result column="id" property="memberId" />
        <result column="com_post_num" property="comPostNum" />
</resultMap>
<resultMap type="ks51team03.company.dto.ComMap" id="ComMapResultMap">
        <id column="ccode" property="cCode"/>
        <result column="cm_link" property="cmLink"/>
        <result column="cm_x" property="cmX"/>
        <result column="cm_y" property="cmY"/>
        <result column="id" property="memberId"/>
</resultMap>

<resultMap id="ComOperTimeResultMap" type="ComOperTime">
        <id column="otcode" property="otCode" />
        <result column="ccode" property="cCode" />
        <result column="ot_mon" property="otMon" />
        <result column="ot_tue" property="otTue" />
        <result column="ot_wed" property="otWed" />
        <result column="ot_thu" property="otThu" />
        <result column="ot_fri" property="otFri" />
        <result column="ot_sat" property="otSat" />
        <result column="ot_sun" property="otSun" />
        <result column="ot_holiday" property="otHoliday" />
        <result column="ot_regular_holiday" property="otRegularHoliday" />
        <result column="ot_break_time" property="otBreakTime" />
        <result column="id" property="memberId" />
        <result column="ot_reg_date" property="otRegDate" />
</resultMap>
<resultMap id="ComReviewResultMap" type="ComReview">
        <id column="revcode" property="revCode" />
        <result column="id" property="memberId" />
        <result column="rev_category" property="revCategory" />
```

```
        <result column="ccode" property="cCode" />
        <result column="com_name" property="companyName" />
        <result column="rev_admin_date" property="revAdminDate" />
        <result column="rev_update_date" property="revUpdateDate" />
        <result column="fileidx" property="fileIdx" />
        <result column="rev_scope" property="revScope" />
        <result column="rev_content" property="revContent" />
        <result column="file_path" property="filePath" />
</resultMap>
<resultMap id="ComLikeResultMap" type="ComLike">
        <id column="lkcode" property="lkCode" />
        <result column="id" property="memberId" />
        <result column="ccode" property="cCode" />
        <result column="lk_alarm" property="lkAlarm" />
        <result column="lk_date" property="lkDate" />
        <result column="lk_state" property="lkState" />
        <result column="name" property="memberName" />
</resultMap>
<resultMap id="CompanyImgResultMap" type="CompanyImg">
        <id column="cicode" property="ciCode" />
        <result column="ccode" property="cCode" />
        <result column="file_idx" property="fileIdx" />
        <result column="file_path" property="filePath" />
</resultMap>
<select id="getAllReview" resultMap="ComReviewResultMap">
        -- 모든 리뷰 최신순 조회
        SELECT
                r.revcode, r.id, r.ccode, r.rev_admin_date, r.rev_update_date
                , r.file_idx, r.rev_scope, r.rev_content, c.com_name
        FROM
                service_reviews AS r
        INNER JOIN
                company AS c
        ON
                r.ccode = c.ccode
        ORDER BY
                r.rev_admin_date DESC
        LIMIT 6
</select>
<select id="getCompanyImgByCcode" parameterType="String" resultMap="CompanyImgResultMap">
        -- 업체 이미지 조회
```

```
        SELECT
        ci.cicode, ci.ccode, ci.file_idx, f.file_path
        FROM
        company_img AS ci
        LEFT JOIN
        files AS f
        ON
        ci.file_idx = f.file_idx
        WHERE
        ci.ccode = #{cCode}
</select>
<select id="getCompanyImgByCcodeForMap" parameterType="List">
        -- 업체 이미지 조회 지도용
        SELECT
        f.file_path
        FROM
        company_img AS ci
        LEFT JOIN
        files AS f
        ON
        ci.file_idx = f.file_idx
        WHERE
        ci.ccode = #{cCode}
</select>
<select id="avgReviewScore" resultType="Double" parameterType="String">
        -- 리뷰 별점 평균
        SELECT ROUND(AVG(rev_scope), 1)
        FROM service_reviews
        WHERE ccode = #{cCode}
        GROUP BY ccode
</select>
<select id="getCompanyReviewCount" resultType="Integer" parameterType="String" >
        -- 업체 리뷰수 검색
        SELECT
                count(ccode)
        FROM
                service_reviews
        WHERE
                ccode = #{cCode}
</select>
```

```
	<select id="getCompanyOperTime" resultMap="ComOperTimeResultMap">
		-- 업체 운영시간 검색
		SELECT
			otcode, ccode, ot_mon, ot_tue, ot_wed, ot_thu, ot_fri, ot_sat, ot_sun, ot_holiday,
ot_regular_holiday, ot_break_time, id, ot_reg_date
		FROM
			com_oper_time
		WHERE
			ccode = #{cCode}
	</select>
	<select id="getCompanyList" resultMap="CompanyResultMap">
		-- 전체 업체 검색
		SELECT
			ccode,
			com_class,
			com_name,
			com_ceo,
			com_addr,
			com_addr_detail,
			com_phone,
			com_stfcount,
			com_email,
			com_parking,
			com_page,
			com_reg_date,
			id,
			com_post_num
		FROM
			company
	</select>
	<select id="getCompanyListByKeyWord" resultMap="CompanyResultMap">
		-- 키워드로 업체 리스트 검색
		SELECT
			ccode,
			com_class,
			com_name,
			com_ceo,
			com_addr,
			com_addr_detail,
			com_post_num,
			com_phone,
```

```
                com_stfcount,
                com_email,
                com_parking,
                com_page,
                com_reg_date,
                id
        FROM
                company
        WHERE
                com_class = #{keyword}
    </select>
    <select id="getComMapByCCode" resultMap="ComMapResultMap">
        -- 업체코드로 업체 좌표및 링크 검색
        SELECT
                c.ccode, c.cm_x, c.cm_y, c.id
        FROM
                com_map AS c
        WHERE
                c.ccode = #{cCode}
    </select>
    <select id="getCompanyCode" resultType="int">
        /* 업체코드 도출*/
        SELECT
        MAX(CAST(REGEXP_REPLACE(ccode, '[^0-9]', '') AS UNSIGNED)) AS max_number
        FROM
        company;
    </select>
    <insert id="insertComMap" parameterType="Company">
        <selectKey keyProperty="cmCode" resultType="String" order="BEFORE">
                SELECT
                CASE
                WHEN COUNT(*) = 0 THEN 'cm1'
                ELSE   CONCAT('cm',   (MAX(CAST(SUBSTRING_INDEX(cmcode,   'cm',   -1)   AS
UNSIGNED)) + 1))
                END
                FROM com_map
        </selectKey>
        -- 업체 위경도 등록
        INSERT INTO com_map
        (cmcode, ccode, cm_x, cm_y, id, com_map_date)
        VALUES   (#{cmCode},   #{companyCode},   #{comMapX},   #{comMapY},   #{memberId},
```

```
NOW())
        </insert>
</mapper>
```

| Package | Ks51team03/company | DTO | Company |
|---|---|---|---|

```
package ks51team03.company.dto;
import ks51team03.member.dto.Member;
import lombok.Data;
import org.springframework.web.multipart.MultipartFile;
@Data
public class Company {
    private String companyCode;
    private String companyClass;
    private String companyName;
    private String companyCeo;
    private String companyAddr;
    private String companyAddrDetail;
    private String companyPhone;
    private int companyStfCount;
    private String companyEmail;
    private boolean companyParking;
    private String companyPage;
    private String companyRegDate;
    private String memberId;
    private String comPostNum;
    private String comMapX;
    private String comMapY;
    private String cmCode;
}
```

| Package | Ks51team03/company | HTML | map_main.html |
|---|---|---|---|

```
<!DOCTYPE html>
<html layout:decorate="~{layout/map_default}"
xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
xmlns:th="http://www.thymeleaf.org">
<head>
    <link rel="stylesheet" href="/assets/css/map.css">
</head>
<th:block layout:fragment="customContent">
    <div class="main-wrapper home-city">
        <section class="bg-smoke py-12">
```

```
            <div th:if="${errorMessage != null}">
                <script>
                    alert("[[${errorMessage}]]");
                </script>
            </div>
            <div class="search_container">
                <div class="search-container">
                    <form id="searchForm" method="get" th:action="@{/map/map_main}">
                        <input id="keyword" name="keyword" placeholder="검색어를 입력하세요" th:value="${keyword}" type="text">
                        <button id="searchBtn" type="submit">검색하기</button>
                    </form>
                </div>
                <div class="map_wrap">
                    <div id="map"></div>
                    <div class="bg_white" id="menu_wrap">
                        <hr>
                        <ul id="placesList"></ul>
                        <div id="pagination"></div>
                    </div>
                    <div class="bg_white" id="info_wrap">
                        <div class="back_button">
                            <button id="returnMenu_wrap" type="button">메뉴출력</button>
                        </div>
                        <div>
                            <div class="company_image">
                                <img alt="PAL 업체 등록 이미지" id="placeIcon" src="" style="width: 240px; height: 150px;">
                            </div>
                            <div class="details" style="font-size: 16px;">
                                <div id="placeNameContainer">
                                    <a id="placeNameDb" style="display:none;" th:href="@{/map/map_company_info}"></a>
                                    <input id="placeNameKakao" class="detail-input" style="display:none;" type="text" readonly>
                                </div>
                                <input id="placeRating" class="detail-input" type="text" value="별점: 알수없음" style="text-align: justify;" readonly>
                                <input id="parking" class="detail-input" type="text" value="주차장: 알수없음" style="text-align: justify;" readonly>
                                <input id="placeAddress" class="detail-input" type="text" readonly>
                                <input id="placePhone" class="detail-input" type="text" readonly>
```

```
                                <input id="employee_count" class="detail-input" type="text" value="직원 수: 알수없음" readonly>
                                <input id="placeHours" class="detail-input" type="text" readonly>
                            </div>
                            <a class="write-review-link" id="write-question-link">문의하기</a>
                            <hr>
                            <div class="reviews">
                                <div class="review-container">
                                    <input id="placeReviews" class="detail-input" type="text" value="리뷰: 알수없음" style="text-align: left" readonly>
                                    <a class="write-review-link" id="write-review-link">작성</a>
                                </div>
                                <div class="review_list">
                                    <!-- 리뷰가 여기 표시됩니다. -->
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </section>
    </div>
</th:block>
<!-- 사용자정의 script -->
<th:block layout:fragment="customJs">
    <script src="//dapi.kakao.com/v2/maps/sdk.js?appkey=187a124bab49811facead4006ff52672&libraries=services" type="text/javascript"></script>
    <script src="/assets/js/map.js" type="text/javascript"></script>
</th:block>
</html>
```

| Package | Ks51team03/company | Structure | JAVASCRIPT |
|---|---|---|---|

```
let currentPosition; // 현재 위치를 저장할 변수
let allResults = []; // 모든 검색 결과를 저장할 배열
const resultsPerPage = 15; // 페이지당 표시할 결과 수
let dbPlaces = []; // 데이터베이스 검색 결과를 저장할 배열
let comMapList = /*[[${comMapList}]]*/ [];
// 현재 위치에 마커 찍는 메서드
function onGeoOkay(position) {
    const lat = position.coords.latitude;
```

```
    const lon = position.coords.longitude;
    currentPosition = new kakao.maps.LatLng(lat, lon);
    const marker = new kakao.maps.Marker({
        position: currentPosition
    });
    marker.setMap(map);
    const infowindow = new kakao.maps.InfoWindow({
        content: '<div style="padding:5px;z-index:1;">현재위치</div>'
    });
    infowindow.open(map, marker);
    map.setCenter(currentPosition);
}
function onGeoError() {
    alert("I can't find you. No weather for you.");
}
navigator.geolocation.getCurrentPosition(onGeoOkay, onGeoError);
let markers = [];
const mapContainer = document.getElementById('map'), // 지도를 표시할 div
    mapOption = {
        center: new kakao.maps.LatLng(37.566826, 126.9786567), // 지도의 중심좌표
        level: 3 // 지도의 확대 레벨
    };
// 지도를 생성합니다
const map = new kakao.maps.Map(mapContainer, mapOption);
// 장소 검색 객체를 생성합니다
const ps = new kakao.maps.services.Places();
const infowindow = new kakao.maps.InfoWindow({zIndex: 1});
// 키워드 검색을 요청하는 함수입니다
document.getElementById('searchForm').addEventListener('submit', function (event) {
    event.preventDefault(); // 기본 폼 제출 동작을 막습니다
    const keyword = document.getElementById('keyword').value;
    if (!keyword.replace(/^₩s+|₩s+$/g, '')) {
        alert('키워드를 입력해주세요!');
        return false;
    }
    // 검색을 진행합니다
    searchPlaces(keyword);
});
function searchPlaces(keyword) {
    // 모든 결과를 초기화합니다
    allResults = [];
```

```
    dbPlaces = []; // 검색마다 dbPlaces 초기화
    // 데이터베이스 검색 요청
    $.ajax({
        url: "/map/get_company_list",
        type: "GET",
        data: { keyword: keyword },
        success: function(response) {
            dbPlaces = response || []; // 응답이 없을 경우 빈 배열 할당
            // 각 dbPlace에서 cCode를 추출하여 추가 요청을 보냅니다
            const cCodes = dbPlaces.map(place => place.companyCode);
            getComMapList(cCodes, keyword);
        },
        error: function(error) {
            console.error("Error fetching company list", error);
        }
    });
}
function getComMapList(cCodes, keyword) {
    $.ajax({
        url: "/map/get_com_map_list",
        type: "POST",
        contentType: "application/json",
        data: JSON.stringify(cCodes),
        success: function(response) {
            comMapList = response || [];
            console.log("comMapList: " + JSON.stringify(comMapList));
            // 카카오에서 장소 검색
            ps.keywordSearch(keyword, placesSearchCB, {page: 1});
        },
        error: function(error) {
            console.error("Error fetching comMapList", error);
        }
    });
}
document.getElementById('menu_wrap').style.display = 'none';
// 장소검색이 완료됐을 때 호출되는 콜백함수입니다
function placesSearchCB(data, status, pagination) {
    if (status === kakao.maps.services.Status.OK) {
        allResults = allResults.concat(data); // 카카오 검색 결과를 allResults에 추가
        if (pagination.current < pagination.last && allResults.length < 45) {
            pagination.gotoPage(pagination.current + 1);
```

```
        } else {
            combineResults(); // 모든 페이지의 데이터를 다 받았거나 45개를 초과하면 호출
        }
        document.getElementById('menu_wrap').style.display = 'block';
    } else if (status === kakao.maps.services.Status.ZERO_RESULT) {
        alert('검색 결과가 존재하지 않습니다.');
    } else if (status === kakao.maps.services.Status.ERROR) {
        alert('검색 결과 중 오류가 발생했습니다.');
    }
}
// 데이터베이스 결과와 카카오 API 결과를 결합하는 함수
function combineResults() {
    if (dbPlaces.length === 0) {
        processResults(); // 결합된 결과를 처리
    } else {
        allResults = allResults.concat(dbPlaces);
        processResults(); // 결합된 결과를 처리
    }
}
// 모든 검색 결과를 처리하는 함수입니다
function processResults() {
    // 거리 기준으로 정렬합니다
    if (currentPosition) {
        allResults.sort(function (a, b) {
            const aDist = getDistance(currentPosition, new kakao.maps.LatLng(a.y, a.x));
            const bDist = getDistance(currentPosition, new kakao.maps.LatLng(b.y, b.x));
            return aDist - bDist;
        });
    }
    // 첫 페이지를 표시합니다
    displayPage(1);
    displayPagination(Math.ceil(allResults.length / resultsPerPage));
}
// 두 지점 간의 거리를 계산하는 함수입니다
function getDistance(position1, position2) {
    const R = 6371; // 지구의 반지름 (단위: km)
    const dLat = (position2.getLat() - position1.getLat()) * Math.PI / 180;
    const dLon = (position2.getLng() - position1.getLng()) * Math.PI / 180;
    const lat1 = position1.getLat() * Math.PI / 180;
    const lat2 = position2.getLat() * Math.PI / 180;
    const a = Math.sin(dLat / 2) * Math.sin(dLat / 2) +
```

```
            Math.sin(dLon / 2) * Math.sin(dLon / 2) * Math.cos(lat1) * Math.cos(lat2);
        const c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
        const distance = R * c;
        return distance * 1000; // 결과를 미터로 반환
}
// 특정 페이지를 표시하는 함수입니다
function displayPage(page) {
        const start = (page - 1) * resultsPerPage;
        const end = start + resultsPerPage;
        displayPlaces(allResults.slice(start, end));
        // 페이지 번호 업데이트
        const paginationEl = document.getElementById('pagination');
        const children = paginationEl.children;
        for (let i = 0; i < children.length; i++) {
            if (i + 1 === page) {
                children[i].className = 'on';
            } else {
                children[i].className = '';
            }
        }
}
// 검색결과 목록 하단에 페이지번호를 표시는 함수입니다
function displayPagination(totalPages) {
        const paginationEl = document.getElementById('pagination'),
            fragment = document.createDocumentFragment();
        // 기존에 추가된 페이지번호를 삭제합니다
        while (paginationEl.hasChildNodes()) {
            paginationEl.removeChild(paginationEl.lastChild);
        }
        for (let i = 1; i <= totalPages; i++) {
            const el = document.createElement('a');
            el.href = "#";
            el.innerHTML = i;
            if (i === 1) {
                el.className = 'on';
            } else {
                el.onclick = (function (i) {
                    return function () {
                        displayPage(i);
                    }
                })(i);
```

```
            }
            fragment.appendChild(el);
        }
        paginationEl.appendChild(fragment);
    }
    // 검색 결과 목록과 마커를 표출하는 함수입니다
    function displayPlaces(places) {
        const listEl = document.getElementById('placesList'),
            menuEl = document.getElementById('menu_wrap'),
            infoCompany = document.getElementById('info_wrap'),
            fragment = document.createDocumentFragment(),
            bounds = new kakao.maps.LatLngBounds();
        // 검색 결과 목록에 추가된 항목들을 제거합니다
        removeAllChildNods(listEl);
        // 지도에 표시되고 있는 마커를 제거합니다
        removeMarker();
        // 첫 번째 마커의 위치를 저장할 변수
        let firstPlacePosition;
        let listIdx = 0;
        // ================== 데이터베이스 데이터 반복 ====================================
        for (let i = 0; i < dbPlaces.length; i++) {
            // 마커를 생성하고 지도에 표시합니다
            const placePosition = new kakao.maps.LatLng(comMapList[i].cmY, comMapList[i].cmX);
            console.log("dbPlaces: " + JSON.stringify(dbPlaces));
            const marker = addMarkerForDb(placePosition);
            const itemEl = getDbListItem(i, dbPlaces[i]); // 검색 결과 항목 Element를 생성합니다
            // 첫 번째 장소의 위치를 저장합니다
            if (i === 0) {
                firstPlacePosition = placePosition;
            }
            // 검색된 장소 위치를 기준으로 지도 범위를 재설정하기 위해 LatLngBounds 객체에 좌표를 추가합니
다
            bounds.extend(placePosition);
            itemEl.setAttribute('data-index', i);
            (function (marker, title, idx) {
                itemEl.onclick = function () {
                    menuEl.style.display = 'none';
                    infoCompany.style.display = 'block';
                    displayInfowindow(marker, dbPlaces[idx].companyName);
                    map.setCenter(marker.getPosition());
                };
```

```
        })(marker, dbPlaces[i], i);
        fragment.appendChild(itemEl);
    }
    //            ====================            데이터베이스            데이터            반복            end
=======================================
    // DB 항목을 Element로 반환하는 함수입니다
    function getDbListItem(index, dbPlace) {
        let el = document.createElement('li'),
            itemStr = '<span class="markerbg2"></span>' +
                '<div class="info" onclick="showDetails(' + listIdx + ', true)">' +
                '    <h5>' + dbPlace.companyName + '<h5>';
        if (dbPlace.road_address_name) {
            itemStr += '     <span>' + dbPlace.road_address_name + '</span>' +
                '    <span class="jibun gray">' + dbPlace.companyAddr + '</span>';
        } else {
            itemStr += '     <span>' + dbPlace.companyAddr + '</span>';
        }
        itemStr += '  <span class="tel">' + dbPlace.companyPhone + '</span>' +
            '</div>';
        el.innerHTML = itemStr;
        el.className = 'item';
        listIdx++;
        return el;
    }
    //            ====================            카카오            데이터            반복            start
=========================================
    for (let i = 0; i < places.length; i++) {
        // 마커를 생성하고 지도에 표시합니다
        const placePosition2 = new kakao.maps.LatLng(places[i].y, places[i].x),
            marker2 = addMarkerForKakao(placePosition2, i),
            itemEl = getListItem(i, places[i]); // 검색 결과 항목 Element를 생성합니다
        if (dbPlaces.length === 0) {
            // 첫 번째 장소의 위치를 저장합니다
            if (i === 0) {
                firstPlacePosition = placePosition2;
            }
            // 검색된 장소 위치를 기준으로 지도 범위를 재설정하기 위해 LatLngBounds 객체에 좌표를 추가
합니다
            bounds.extend(placePosition2);
        }
        itemEl.setAttribute('data-index', i);
```

```
        (function (marker2, title, idx) {
            itemEl.onclick = function () {
                menuEl.style.display = 'none';
                infoCompany.style.display = 'block';
                displayInfowindow(marker2, title);
                map.setCenter(marker2.getPosition());
            };
        })(marker2, places[i].place_name, i);
        fragment.appendChild(itemEl);
    }
    //            ====================            카카오            데이터            반복            end
==========================================
    // 검색결과 항목들을 검색결과 목록 Element에 추가합니다
    listEl.appendChild(fragment);
    menuEl.scrollTop = 0;
    // 검색된 장소 위치를 기준으로 지도 범위를 재설정합니다
    map.setBounds(bounds);
    // 최대 확대 레벨 제한
    const maxZoomLevel = 3; // 최대 확대 레벨 설정
    if (map.getLevel() > maxZoomLevel) {
        map.setLevel(maxZoomLevel);
    }
    // 첫 번째 마커의 위치를 지도의 중심으로 설정합니다
    if (firstPlacePosition) {
        map.setCenter(firstPlacePosition);
    }
    // 카카오 항목을 Element로 반환하는 함수입니다
    function getListItem(index, places) {
        let el = document.createElement('li'),
            itemStr = '<span class="markerbg marker_' + (index + 1) + '"></span>' +
                '<div class="info" onclick="showDetails(' + listIdx + ', false)">' +
                '    <h5>' + places.place_name + '</h5>';
        if (places.road_address_name) {
            itemStr += '    <span>' + places.road_address_name + '</span>' +
                '    <span class="jibun gray">' + places.address_name + '</span>';
        } else {
            itemStr += '    <span>' + places.address_name + '</span>';
        }
        itemStr += '  <span class="tel">' + places.phone + '</span>' +
            '</div>';
        el.innerHTML = itemStr;
```

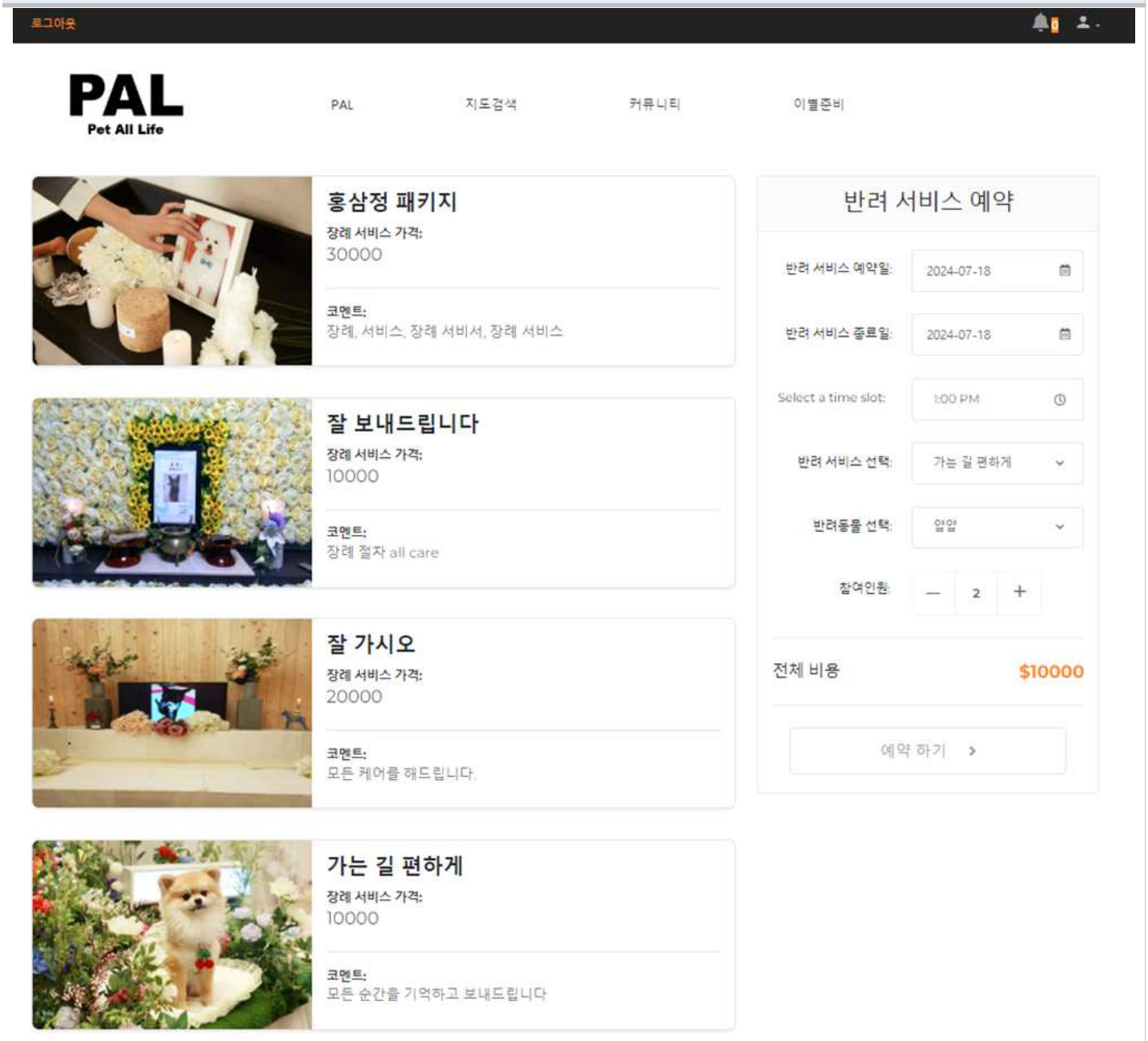
```
            el.className = 'item';
            listIdx++;
            return el;
        }
    }
    // 마커를 생성하고 지도 위에 마커를 표시하는 함수입니다
    function addMarkerForKakao(position, idx) {
        const imageSrc = 'https://t1.daumcdn.net/localimg/localimages/07/mapapidoc/marker_number_blue.png',
    // 마커 이미지 url, 스프라이트 이미지를 씁니다
            imageSize = new kakao.maps.Size(36, 37),  // 마커 이미지의 크기
            imgOptions = {
                spriteSize: new kakao.maps.Size(36, 691), // 스프라이트 이미지의 크기
                spriteOrigin: new kakao.maps.Point(0, (idx * 46) + 10), // 스프라이트 이미지 중 사용할 영역의
    좌상단 좌표
                offset: new kakao.maps.Point(13, 37) // 마커 좌표에 일치시킬 이미지 내에서의 좌표
            },
            markerImage = new kakao.maps.MarkerImage(imageSrc, imageSize, imgOptions),
            marker = new kakao.maps.Marker({
                position: position, // 마커의 위치
                image: markerImage
            });
        marker.setMap(map); // 지도 위에 마커를 표출합니다
        markers.push(marker);  // 배열에 생성된 마커를 추가합니다
        return marker;
    }
    // 마커를 생성하고 지도 위에 마커를 표시하는 함수입니다
    function addMarkerForDb(position) {
        const imageSrc = '/assets/img/map_marker.png', // 마커 이미지 url, 스프라이트 이미지를 씁니다
            imageSize = new kakao.maps.Size(64,69),  // 마커 이미지의 크기
            imgOptions = {
                spriteSize: new kakao.maps.Size(36, 50), // 스프라이트 이미지의 크기
                offset: new kakao.maps.Point(20, 40) // 마커 좌표에 일치시킬 이미지 내에서의 좌표
            },
            markerImage = new kakao.maps.MarkerImage(imageSrc, imageSize, imgOptions),
            marker = new kakao.maps.Marker({
                position: position, // 마커의 위치
                image: markerImage
            });
        marker.setMap(map); // 지도 위에 마커를 표출합니다
        markers.push(marker);  // 배열에 생성된 마커를 추가합니다
        return marker;
```

```
}
// 지도 위에 표시되고 있는 마커를 모두 제거합니다
function removeMarker() {
    for (let i = 0; i < markers.length; i++) {
        markers[i].setMap(null);
    }
    markers = [];
}
function displayInfowindow(marker, title) {
    const content = '<div style="padding:5px;z-index:1;">' + title + '</div>';
    infowindow.setContent(content);
    infowindow.open(map, marker);
}
// 검색결과 목록의 자식 Element를 제거하는 함수입니다
function removeAllChildNods(el) {
    while (el.hasChildNodes()) {
        el.removeChild(el.lastChild);
    }
}
function showDetails(index, isDB) {
    const infoWrap = document.getElementById('info_wrap');
    const menuWrap = document.getElementById('menu_wrap');
    let cCode;
    const placeNameDb = document.getElementById('placeNameDb');
    const placeNameKakao = document.getElementById('placeNameKakao');
    const placeIcon = document.getElementById('placeIcon');
    // 기존 정보를 초기화합니다.
    placeNameDb.value = '';
    placeNameDb.href = '';
    placeNameDb.style.display = 'none';
    placeNameKakao.value = '';
    placeNameKakao.style.display = 'none';
    placeIcon.src = ''; // 이미지 초기화
    document.getElementById('parking').value = '주차장: 알수없음';
    document.getElementById('placeAddress').value = '';
    document.getElementById('placePhone').value = '';
    document.getElementById('employee_count').value = '직원 수: 알수없음';
    document.getElementById('placeRating').value = '별점: 알수없음';
    document.getElementById('placeReviews').value = '리뷰: 알수없음';
    document.querySelector('.review_list').innerHTML = '';
    document.getElementById('write-question-link').style.display = 'none';
```

```
        document.getElementById('write-review-link').style.display = 'none';
        if (isDB) {
            const places = dbPlaces[index];
            cCode = places.companyCode; // cCode 설정
            placeNameDb.innerText  = places.companyName;
            placeNameDb.href = "/map/map_company_info?cCode=" + cCode;
            placeNameDb.style.display = 'block';
            document.getElementById('parking').value = '주차장: ' + (places.companyParking ? 'O' : 'X');
            document.getElementById('placeAddress').value = places.companyAddr;
            document.getElementById('placePhone').value = places.companyPhone;
            document.getElementById('employee_count').value = '직원 수: ' + places.companyStfCount;
            document.getElementById('write-question-link').setAttribute('href', '/map/map_write_question?cCode=' + cCode);
            document.getElementById('write-review-link').setAttribute('href',  '/map/map_write_review?cCode='  + cCode);
            document.getElementById('write-question-link').style.display = 'block';
            document.getElementById('write-review-link').style.display = 'block';
            // AJAX 요청으로 리뷰 데이터와 이미지 가져오기
            $.ajax({
                url: "/map/getCompanyInfo",
                type: "GET",
                data: { cCode: cCode },
                success: function(response) {
                    document.getElementById('placeRating').value = '별점: ' + (response.avgReviewScore !== null ? response.avgReviewScore : '없음');
                    document.getElementById('placeReviews').value = '리뷰 수: ' + (response.reviewCount !== null ? response.reviewCount : '없음');
                    if (response.comImg && response.comImg.length > 0) {
                        placeIcon.src = response.comImg[0]; // 이미지 설정
                    } else {
                        placeIcon.src = '/path/to/default/image.png'; // 기본 이미지 설정
                    }
                    const reviewList = document.querySelector('.review_list');
                    reviewList.innerHTML = ''; // 기존 리뷰 목록 초기화
                    if (response.reviews && response.reviews.length > 0) {
                        const reviewsToShow = response.reviews.slice(0, 3);
                        reviewsToShow.forEach(function(review) {
                            const reviewDiv = document.createElement('div');
                            reviewDiv.className = 'review';
                            // 리뷰 텍스트 추가
                            const reviewText = document.createElement('p');
                            reviewText.innerText = review.memberId + ': ' + review.revContent;
```

```
                        reviewDiv.appendChild(reviewText);
                        // 이미지 추가
                        if (review.filePath) {
                            const reviewImage = document.createElement('img');
                            reviewImage.src = review.filePath;
                            reviewImage.style.display = 'block'; // 이미지 블록 요소로 설정
                            reviewImage.style.width = '80px'; // 이미지 크기 조정
                            reviewImage.style.height = '80px'; // 이미지 비율 유지
                            reviewDiv.appendChild(reviewImage);
                        }
                        reviewList.appendChild(reviewDiv);
                    });
                } else {
                    reviewList.innerText = '리뷰 없음';
                }
            },
            error: function(error) {
                console.error("Error fetching company info", error);
            }
        });
    } else {
        const places = allResults[index - dbPlaces.length];
        cCode = places.id; // cCode 설정
        placeNameKakao.value = places.place_name;
        placeNameKakao.style.display = 'block';
        document.getElementById('placeAddress').value = places.address_name;
        document.getElementById('placePhone').value = places.phone;
        const reviewList = document.querySelector('.review_list');
        reviewList.innerHTML = '리뷰 없음'; // 기존 리뷰 목록 초기화
    }
    menuWrap.style.display = 'none';
    infoWrap.style.display = 'block';
}
document.getElementById('returnMenu_wrap').onclick = function () {
    document.getElementById('info_wrap').style.display = 'none';
    document.getElementById('menu_wrap').style.display = 'block';
};
```

| Fuction | funeral_service (장례 서비스) |
| --- | --- |
| Image | |



| Package | ks51team03/funeral/serviceList | Controller | ServiceListController |
|---|---|---|---|

```
package ks51team03.funeral.serviceList.controller;
import jakarta.servlet.http.HttpSession;
import ks51team03.company.dto.Company;
import ks51team03.funeral.reserve.dto.ReserveDto;
import ks51team03.funeral.reserve.dto.ReserveMemberPet;
import ks51team03.funeral.reserve.service.ReserveService;
import ks51team03.funeral.serviceList.dto.FuneralCompanyImgDto;
import ks51team03.funeral.serviceList.dto.ServiceImgDto;
import ks51team03.funeral.serviceList.dto.ServiceListDto;
import ks51team03.funeral.serviceList.mapper.ServiceListMapper;
import ks51team03.funeral.serviceList.service.ServiceListService;
import ks51team03.member.dto.Member;
import ks51team03.member.service.MemberServiceImpl;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.multipart.MultipartFile;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;
import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;
@Controller
@RequiredArgsConstructor
@Slf4j
public class ServiceListController {
```

```
    private final ServiceListService serviceListService;
    private final ReserveService reserveService;
    private final MemberServiceImpl memberService;
    @PostMapping("/funeral/funeral_service_detail")
    public String funeralReserve(HttpSession session, Model model, ReserveDto reserveDto) {
        String memberId = (String) session.getAttribute("SID");
        log.info("html에서 포스트맵핑되었습니다.");
        log.info("로그인한 회원 아이디 memberId={}", memberId);
        log.info("reserveDto:{}", reserveDto);
        // 회원 정보 조회
        Member member = memberService.getMemberInfoById(memberId);
        String reservePhone = member.getMemberPhone();
        log.info("선택한 반려동물 이름: {}", reserveDto.getReservePetName());
        // 세션에서 ccode 값 가져오기
        String ccode = (String) session.getAttribute("CCODE");
        log.info("세션에서 가져온 ccode: {}", ccode);
        if (ccode == null || ccode.isEmpty()) {
            log.error("ccode가 세션에 없습니다.");
            return "redirect:/errorPage"; // 에러 페이지로 리디렉션
        }
        reserveDto.setReserveId(memberId);
        reserveDto.setReservePhone(reservePhone);
        reserveDto.setReserveCompanyCode(ccode); // ccode 설정
        reserveService.funeralReserve(reserveDto);
        log.info("reserveDto={}", reserveDto);
        model.addAttribute("reserveDto", reserveDto);
        return "redirect:/funeral/funeral_reserve_info";
    }
    // 컨트롤러 부분 수정
    @GetMapping("funeral/funeral_service_detail")
    public String funeralReserve(
            @RequestParam(value="funeralserviceCcode") String funeralserviceCcode,
            Model model, HttpSession session) {
        log.info("funeralserviceCcode: " + funeralserviceCcode);
        String memberId = (String) session.getAttribute("SID");
        log.info("로그인한 회원 아이디 memberId={}", memberId);
        List<ServiceListDto> serviceListCcode =
serviceListService.getServiceInfoByCode(funeralserviceCcode);
        List<String> fscodeList = serviceListCcode.stream()
                .map(ServiceListDto::getFuneralserviceCode)
                .collect(Collectors.toList());
```

```
                // memberId를 전달
                List<ReserveMemberPet> getMemberPetList = serviceListService.getMemberPet(memberId);
                log.info("getMemberPetList:{}", getMemberPetList);
                // ccode 설정
                String ccode = funeralserviceCcode;
                log.info("ccode: {}", ccode);
                if (ccode == null || ccode.isEmpty()) {
                        log.error("ccode가 세션에 없습니다.");
                        return "redirect:/errorPage"; // 에러 페이지로 리디렉션
                }
                // 세션에 ccode 저장
                session.setAttribute("CCODE", ccode);
                // 장례 서비스 이미지 가져오기
                List<ServiceImgDto> serviceImgDtos = serviceListService.getServiceImgs(fscodeList, ccode);
                log.info("serviceImgDtos : {}", serviceImgDtos);
                Map<String, List<ServiceImgDto>> imagesByServiceCode = serviceImgDtos.stream()
                                .collect(Collectors.groupingBy(ServiceImgDto::getFscode));
                model.addAttribute("funeralserviceCcode", ccode);
                model.addAttribute("serviceListCcode", serviceListCcode);
                model.addAttribute("getMemberPetList", getMemberPetList);
                model.addAttribute("imagesByServiceCode", imagesByServiceCode);
                return "funeral/funeral_service_detail";
        }
}
```

| Package | ks51team03/funeral/serviceList | Service | ServiceListService |
|---|---|---|---|

```
package ks51team03.funeral.serviceList.service;
import ks51team03.company.dto.Company;
import ks51team03.files.dto.FileRequest;
import ks51team03.files.mapper.FileMapper;
import ks51team03.files.util.FileUtils;
import ks51team03.funeral.reserve.dto.ReserveDto;
import ks51team03.funeral.reserve.dto.ReserveMemberPet;
import ks51team03.funeral.serviceList.dto.FuneralCompanyImgDto;
import ks51team03.funeral.serviceList.dto.ServiceImgDto;
import ks51team03.funeral.serviceList.dto.ServiceListDto;
import ks51team03.funeral.serviceList.mapper.ServiceListMapper;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
```

```
import java.util.List;
@Service
@Slf4j
@Transactional
@RequiredArgsConstructor
public class ServiceListService {
    private final ServiceListMapper serviceListMapper;
    private final FileUtils fileUtils;
    private final FileMapper fileMapper;
    public List<ServiceListDto> getServiceList() {
        return serviceListMapper.getServiceListDto();
    }
    public List<ServiceListDto> getServiceInfoByCode(String funeralserviceCcode){
        log.info("getServiceInfoByCode: {}", serviceListMapper.getServiceInfoByCode(funeralserviceCcode));
        return serviceListMapper.getServiceInfoByCode(funeralserviceCcode);
    }
    /**
     *  장례 예약 프로세스
     */
    public void funeralReserve(ReserveDto reserveDto){
        int result = serviceListMapper.funeralReserve(reserveDto);
    }
    public List<Company> getCompanyInfo(Company company){
        return serviceListMapper.getCompanyInfo(company);
    }
    public String insertFuneralService(ServiceListDto serviceListDto){
        serviceListMapper.insertFuneralService(serviceListDto);
        return serviceListDto.getFuneralserviceCode(); // 생성된 fscode 반환
    }
    // 장례 업체에 등록된 장례 서비스를 가져오기 위한 company 코드
    public List<ServiceListDto> getServiceList(ServiceListDto serviceListDto){
        return serviceListMapper.getServiceList(serviceListDto);
    };
    // 장례 회사 정보를 가져오기 위한 조회
    public List<FuneralCompanyImgDto> getCompanyInfoList(){
        return serviceListMapper.getCompanyInfoList();
    }
    // 장례 예약 전 회원 반려동물 가져오기 위한 코드
    public List<ReserveMemberPet> getMemberPet(String memberId){
        return serviceListMapper.getMemberPet(memberId);
    }
```

```
    // 장례 예약 서비스 이미지 업로드
    public void addFuneralServiceImg(ServiceImgDto serviceImgDto, String companyCode){
        FileRequest fileRequest = fileUtils.uploadFile(serviceImgDto.getFurImgFile(), companyCode);
        if(fileRequest != null){
            fileRequest.setFileCate(companyCode);
            log.info("fileRequest: {}", fileRequest);
            fileMapper.addFile(fileRequest);
            serviceImgDto.setFileIdx(fileRequest.getFileIdx());
        }
        serviceListMapper.insertFuneralServiceImg(serviceImgDto);
    }

}
```

| Package | ks51team03/funeral/serviceList | Mapper/xml | FuneralMapper.xml |
|---|---|---|---|

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"https://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="ks51team03.funeral.serviceList.mapper.ServiceListMapper">
    <resultMap type="ks51team03.funeral.serviceList.dto.ServiceListDto" id="getServiceListMap">
        <id          column="fscode"          property="funeralserviceCode"/>
        <result      column="ccode"           property="funeralserviceCcode"/>
        <result      column="fs_title"        property="funeralserviceTitle"/>
        <result      column="fs_price"        property="funeralservicePrice"/>
        <result      column="fs_comment"      property="funeralserviceComment"/>
        <result      column="fs_weight"       property="funeralserviceWeight"/>
        <result      column="id"              property="funeralserviceId"/>
        <result      column="fs_reg_date"     property="funeralserviceRegDate"/>
        <result      column="fs_state"        property="funeralserviceState"/>
    </resultMap>
    <resultMap id="getFureralReserve" type="ks51team03.funeral.reserve.dto.ReserveDto">
        <id          column="frcode"              property="reserveCode"/>
        <result      column="ccode"               property="reserveCompanyCode"/>
        <result      column="fscode"              property="reserveServiceCode"/>
        <result      column="id"                  property="reserveId"/>
        <result      column="fr_start_date"       property="reserveStartDate"/>
        <result      column="fr_end_date"         property="reserveEndDate"/>
        <result      column="fr_phone"            property="reservePhone"/>
        <result      column="fr_payment"          property="reservePayment"/>
        <result      column="fr_confirm"          property="reserveConfirm"/>
```

```
        <result      column="fr_reg_date"          property="reserveRegDate"/>
        <result      column="fr_reserve_time"      property="reserveTime"/>
    </resultMap>
    <resultMap id="getCompanyInfo" type="ks51team03.funeral.serviceList.dto.FuneralCompanyImgDto">
        <id       column="ccode"                   property="companyCode" />
        <result column="com_class"                 property="companyClass" />
        <result column="com_name"                  property="companyName" />
        <result column="com_ceo"                   property="companyCeo" />
        <result column="com_addr"                  property="companyAddr" />
        <result column="com_phone"                 property="companyPhone" />
        <result column="com_stfcount"              property="companyStfCount" />
        <result column="com_email"                 property="companyEmail" />
        <result column="com_reg_date"              property="companyRegDate" />
        <result column="id"                        property="memberId" />
        <result column="file_idx"                  property="fileIdx" />
        <result column="file_path"                 property="filePath" />
    </resultMap>
    <resultMap id="getCompanyInfoModify" type="ks51team03.company.dto.Company">
        <id       column="ccode"                   property="companyCode" />
        <result column="com_class"                 property="companyClass" />
        <result column="com_name"                  property="companyName" />
        <result column="com_ceo"                   property="companyCeo" />
        <result column="com_addr"                  property="companyAddr" />
        <result column="com_phone"                 property="companyPhone" />
        <result column="com_stfcount"              property="companyStfCount" />
        <result column="com_email"                 property="companyEmail" />
        <result column="com_reg_date"              property="companyRegDate" />
        <result column="id"                        property="memberId" />
    </resultMap>
    <resultMap id="getMemberPet" type="ks51team03.funeral.reserve.dto.ReserveMemberPet">
        <id           column="id"                  property="memberId"/>
        <result      column="name"                 property="memberName"/>
        <result      column="level"                property="memberLevel"/>
        <result      column="level_name"           property="memberLevelName"/>
        <result      column="phone"                property="memberPhone"/>
        <result      column="email"                property="memberEmail"/>
        <result      column="pcode"                property="petCode"/>
        <result      column="p_name"               property="petName"/>
        <result      column="p_weight"             property="petWeight"/>
    </resultMap>
    <resultMap id="getServiceImgResultMap" type="ks51team03.funeral.serviceList.dto.ServiceImgDto">
```

```
        <id column="fsicode" property="fsicode"/>
        <result column="ccode" property="ccode"/>
        <result column="fscode" property="fscode"/>
        <result column="file_idx" property="fileIdx"/>
        <result column="file_cate" property="fileCate"/>
        <result column="file_origin_name" property="fileOriginName" />
        <result column="file_new_name" property="fileNewName" />
        <result column="file_path" property="filePath" />
        <result column="file_size" property="fileSize" />
        <result column="file_reg_date" property="fileRegDate" />
    </resultMap>
    <!-- 장례 서비스 전체 조회 -->
    <select id="getServiceListDto" resultMap="getServiceListMap">
        SELECT
        fs.fscode,
        fs.ccode,
        fs.fs_title,
        fs.fs_price,
        fs.fs_weight,
        fs.id,
        fs.fs_reg_date,
        fs.fs_state
        FROM
        funeral_service AS fs
    </select>
    <!-- 장례 서비스 전체 조회 -->
    <select id="getServiceInfoByCode" parameterType="String" resultMap="getServiceListMap">
        /* 장례 업체별 장례 서비스 정보*/
        SELECT
        fs.fscode,
        fs.ccode,
        fs.fs_title,
        fs.fs_price,
        fs_comment,
        fs.fs_weight,
        fs.id,
        fs.fs_reg_date,
        fs.fs_state
        FROM
        funeral_service AS fs
        where
```

```
        fs.ccode = #{funeralserviceCcode};
    </select>
    <insert id="funeralReserve" parameterType="Map" >
        /* 장례 예약 */
        INSERT INTO funeral_reserve
        (    frcode,
        ccode,
        fscode,
        id,
        fr_start_date,
        fr_end_date,
        fr_phone,
        fr_payment,
        fr_confirm,
        fr_reg_date,
        fr_reserve_time)
        VALUES (
        #{reserveCode},
        #{reserveCompanyCode},
        #{reserveServiceCode},
        #{reserveId},
        #{reserveStartDate},
        #{reserveEndDate},
        #{reservePhone},
        '결제 전',
        '예약 완료',
        NOW(),
        #{reserveTime}
        )
    </insert>
    <!-- 장례 서비스 등록시 회사 정보 확인 -->
    <select id="getCompanyInfo" resultMap="getCompanyInfoModify">
        /* 장례 업체 정보 */
        SELECT
        c.ccode,
        c.com_class,
        c.com_name,
        c.com_ceo,
        c.com_addr,
        c.com_phone,
        c.com_stfcount,
```

```
        c.id,
        c.com_parking,
        c.com_reg_date
        FROM
        company AS c
        WHERE
        c.id = #{memberId}
    </select>
    <insert id="insertFuneralService" parameterType="Map">
        <selectKey keyProperty="funeralserviceCode" resultType="String" order="BEFORE">
            /* 장례 서비스 등록을 위한 fscode 조회*/
            SELECT
            CASE
            WHEN COUNT(fscode) = 0 THEN 'fs1'
            ELSE
            CONCAT(
            'fs',
            LPAD(MAX(CAST(SUBSTRING(fscode, 3) AS UNSIGNED))+1, 3, '0')
            )
            END AS funeralserviceCode
            FROM
            funeral_service;
        </selectKey>
        /* 장례 서비스 등록 insert문*/
        INSERT INTO funeral_service (
        fscode,
        ccode,
        fs_title,
        fs_comment,
        fs_price,
        fs_weight,
        id,
        fs_reg_date,
        fs_state
        ) VALUES (
        #{funeralserviceCode},
        #{funeralserviceCcode},
        #{funeralserviceTitle},
        #{funeralserviceComment},
        #{funeralservicePrice},
        #{funeralserviceWeight},
```

```
        #{funeralserviceId},
        CURDATE(),
        '활성화'
        );
    </insert>
    <select id="getServiceList" resultMap="getServiceListMap">
        /* 업체별 등록된 장례 서비스 리스트*/
        SELECT
            fscode,
            ccode,
            fs_title,
            fs_price,
            fs_comment,
            fs_weight,
            id,
            fs_reg_date,
            fs_state
            FROM
            funeral_service
        WHERE
        id = #{funeralserviceId}
    </select>
    <!-- 장례 서비스 노출을 위한 회사 정보 가져오기 -->
    <select id="getCompanyInfoList" resultMap="getCompanyInfo">
        /* 장례 서비스 노출을 위한 회사 정보 select문*/
        SELECT
        c.ccode,
        c.com_class,
        c.com_name,
        c.com_ceo,
        c.com_addr,
        c.com_phone,
        c.com_stfcount,
        c.id,
        c.com_reg_date,
        f.file_path,
        f.file_idx
        FROM
        company AS c
        INNER join
        company_img AS ci
```

```
        ON
        c.ccode = ci.ccode
        INNER join
        files AS f
        on
        ci.file_idx = f.file_idx
        where
        c.com_class like '%장례%'
    </select>
    <!-- 장례 예약 시 반려동물 정보 가져오기 위한 코드 -->
    <select id="getMemberPet" resultMap="getMemberPet">
        /* 예약시 반려동물 가져오기 위한 select문*/
        SELECT
        p.p_name,
        m.id,
        m.name
        FROM
        member AS m
        INNER join
        pet AS p
        on
        m.id = p.id
        where
        m.id = #{memberId}
    </select>
    <insert id="insertFuneralServiceImg" parameterType="Map">
        <selectKey keyProperty="fsicode" resultType="String" order="BEFORE">
            SELECT
            CASE
            WHEN COUNT(*) = 0 THEN 'fsi1'
            ELSE CONCAT('fsi', MAX(CAST(SUBSTRING(fsicode, 4) AS UNSIGNED)) + 1)
            END
            FROM funeral_service_img
        </selectKey>
        -- 장례 서비스 이미지 등록
        INSERT INTO funeral_service_img
        (fsicode, ccode, fscode, file_idx, file_cate)
        VALUES (#{fsicode}, #{ccode}, #{fscode}, #{fileIdx}, '장례 예약')
    </insert>
    <!-- 장례 서비스 수정 -->
    <select id="modifyServiceInfoByCode" parameterType="String" resultMap="getServiceListMap">
```

```
        /* 등록된 장례 서비스 수정을 위한 select문 */
        SELECT
        fs.fscode,
        fs.ccode,
        fs.fs_title,
        fs.fs_price,
        fs_comment,
        fs.fs_weight,
        fs.id,
        fs.fs_reg_date,
        fs.fs_state
        FROM
        funeral_service AS fs
        where
        fs.fscode = #{funeralserviceCode};
    </select>
    <update id="updateServiceInfo" parameterType="ks51team03.funeral.serviceList.dto.ServiceListDto">
        /* 등록된 장례 서비스 수정 */
        UPDATE
            funeral_service
        SET
        fs_title = #{funeralserviceTitle},
        fs_comment = #{funeralserviceComment},
        fs_price = #{funeralservicePrice},
        fs_weight = #{funeralserviceWeight},
        fs_state = #{funeralserviceState}
        WHERE
        fscode = #{funeralserviceCode}
    </update>
    <!-- 기존 이미지 정보 가져오기 -->
    <select id="getServiceImg" resultMap="getServiceImgResultMap" parameterType="map">
        /* 기존 이미지 정보 가져오는 select문 */
        SELECT
        fsi.fsicode,
        fsi.ccode,
        fsi.fscode,
        fsi.file_idx,
        fsi.file_cate,
        f.file_origin_name,
        f.file_new_name,
        f.file_path,
```

```
        f.file_size,
        f.file_reg_date
        FROM
        funeral_service_img fsi
        INNER JOIN
        files f ON fsi.file_idx = f.file_idx
        WHERE
        fsi.fscode = #{fscode} AND fsi.ccode = #{ccode}
    </select>
    <!-- 기존 파일을 삭제하는 쿼리 -->
    <delete id="deleteServiceImg" parameterType="ks51team03.funeral.serviceList.dto.ServiceImgDto">
        /* 기존 이미지 파일 삭제하는 delete문 */
        DELETE FROM funeral_service_img
        WHERE fscode = #{fscode} AND ccode = #{ccode}
    </delete>
    <select id="getServiceImgs" resultMap="getServiceImgResultMap" parameterType="map">
        /* 장례 업체별 등록된 이미지를 가져오기 위한 select문 */
        SELECT
        fsi.fsicode,
        fsi.ccode,
        fsi.fscode,
        fsi.file_idx,
        fsi.file_cate,
        f.file_origin_name,
        f.file_new_name,
        f.file_path,
        f.file_size,
        f.file_reg_date
        FROM
        funeral_service_img fsi
        INNER JOIN files f ON fsi.file_idx = f.file_idx
        WHERE
        fsi.fscode IN
        <foreach item="fscode" collection="fscodeList" open="(" separator="," close=")">
            #{fscode}
        </foreach>
        AND fsi.ccode = #{ccode}
    </select>
</mapper>
```

| Package | Ks51team03/company | DTO | ServiceListDto |
|---|---|---|---|

```
package ks51team03.funeral.serviceList.dto;
import lombok.Data;
import org.springframework.web.multipart.MultipartFile;
@Data
public class ServiceListDto {
    private String funeralserviceCode;
    private String funeralserviceCcode; // 업체코드
    private String funeralserviceTitle; // 장례 서비스 이름
    private int funeralservicePrice; // 서비스가격
    private String funeralserviceComment; // 서비스 코멘트
    private float funeralserviceWeight; // 허용무게(kg)
    private String funeralserviceId; // 등록자 아이디
    private String funeralserviceRegDate; // 등록일
    private String funeralserviceState; //상태
    private String fileIdx;
    private MultipartFile revImgFile;
    private String filePath;
}
```

| Package | Ks51team03/funeral/serviceList | HTML | funeral_service_detail.html |
|---|---|---|---|

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org"
xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout" xmlns=""
layout:decorate="~{layout/default}">
<head>
    <style>
        .service-card {
            display: flex; /* Flexbox 사용 */
            border: 1px solid #ddd;
            border-radius: 8px;
            box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
            overflow: hidden;
            margin-bottom: 32px;
        }
        .service-card img {
            width: 300px; /* 이미지의 너비 설정 */
            height: 200px; /* 이미지의 높이 설정 */
            object-fit: cover; /* 이미지가 영역에 맞도록 조정 */
            image-rendering: optimizeQuality; /* 이미지 선명도 최적화 */
            -webkit-optimize-contrast: crisp-edges; /* Webkit 기반 브라우저 선명도 최적화 */
```

```
            image-rendering: -moz-crisp-edges; /* Firefox 선명도 최적화 */
            image-rendering: -o-crisp-edges; /* Opera 선명도 최적화 */
            image-rendering: crisp-edges; /* 기타 브라우저 선명도 최적화 */
        }
        .service-card-content {
            padding: 16px;
            width: calc(100% - 300px); /* 이미지와 콘텐츠의 너비 설정 */
            display: flex;
            flex-direction: column; /* 세로 정렬 */
            justify-content: center; /* 중앙 정렬 */
        }
        .service-card h2, .service-card h3, .service-card p {
            margin-bottom: 8px;
        }
        .service-card h2 {
            font-size: 1.5rem;
            font-weight: bold;
        }
        .service-card h3 {
            font-size: 1.25rem;
            color: #555;
        }
        .service-card p {
            font-size: 1rem;
            color: #777;
        }
        .service-card .label {
            font-weight: bold;
            color: #333;
        }
        .service-card-divider {
            border-top: 1px solid #ddd;
            margin: 16px 0;
        }
    </style>
</head>
<th:block layout:fragment="customContent">
    <div class="main-wrapper single-hotel-right-sidebar">
        <section class="py-12">
            <div class="container">
                <div class="row">
```

```
                <div class="col-md-5 col-lg-4 order-2" style=" position: sticky; top: 140px; height: fit-content;">
                    <form id="funeralReserveForm" th:action="@{/funeral/funeral_service_detail}" method="POST">
                        <div class="card border">
                            <h2 class="card-header text-uppercase text-center bg-smoke border-bottom">
                                반려 서비스 예약
                            </h2>
                            <div class="card-body px-3 py-4">
                                <div class="border-bottom mb-5">
                                    <div class="form-group mb-5">
                                        <div class="row">
                                            <label for="inputTime" class="col-xl-5 col-form-label text-center text-xl-end px-2">반려 서비스 예약일:</label>
                                            <div class="col-xl-7">
                                                <div class="form-group form-group-icon form-group-icon-category-2 mb-0">
                                                    <i class="far fa-calendar-alt" aria-hidden="true"></i>
                                                    <input type="text" id="reserveStartDate" class="form-control daterange-picker-category-2" autocomplete="off" name="reserveStartDate" value="" placeholder="MM/DD/YYYY">
                                                </div>
                                            </div>
                                        </div>
                                    </div>
                                </div>
                                <div class="form-group mb-5">
                                    <div class="row">
                                        <label for="inputTime" class="col-xl-5 col-form-label text-center text-xl-end px-2">반려 서비스 종료일:</label>
                                        <div class="col-xl-7">
                                            <div class="form-group form-group-icon form-group-icon-category-2 mb-0">
                                                <i class="far fa-calendar-alt" aria-hidden="true"></i>
                                                <input type="text" class="form-control daterange-picker-category-2" autocomplete="off" name="reserveEndDate" id="inputTime" value="" placeholder="MM/DD/YYYY">
                                            </div>
                                        </div>
                                    </div>
                                </div>
```

```
                                    <div class="form-group mb-5">
                                        <div class="row">
                                            <label for="inputTime" class="col-xl-5 col-form-label text-center text-xl-right px-2">Select a time slot:</label>
                                            <div class="col-xl-7">
                                                <div class="form-group mb-0">
                                                    <div class="select-default select-category-2 timer">
                                                        <select th:id="reserveTime" name="reserveTime" class="select-option">
                                                            <option value="09:00:00">9:00 AM</option>
                                                            <option value="10:00:00">10:00 AM</option>
                                                            <option value="11:00:00">11:00 AM</option>
                                                            <option value="12:00:00">12:00 PM</option>
                                                            <option value="13:00:00">1:00 PM</option>
                                                            <option value="14:00:00">2:00 PM</option>
                                                            <option value="15:00:00">3:00 PM</option>
                                                            <option value="16:00:00">4:00 PM</option>
                                                        </select>
                                                    </div>
                                                </div>
                                            </div>
                                        </div>
                                    </div>
                                    <div class="form-group mb-5">
                                        <div class="row">
                                            <label for="funeralserviceTitle" class="col-xl-5 col-form-label text-center text-xl-end px-2">반려 서비스 선택:</label>
                                            <div class="col-xl-7">
                                                <div class="form-group mb-0">
                                                    <div class="select-default select-category-2">
                                                        <select id="funeralserviceTitle" name="reserveServiceCode" class="select-option" onchange="updateTotalPrice()">
                                                            <option th:each="service : ${serviceListCcode}"
                                                                th:value="${service.getFuneralserviceCode()}"
```

```
th:text="${service.getFuneralserviceTitle()}"
th:data-price="${service.getFuneralservicePrice()}"
th:attr="data-ccode=${service.getFuneralserviceCcode()}"></option>
                                                    </select>
                                                </div>
                                            </div>
                                        </div>
                                    </div>
                                    <!-- 반려동물 선택 옵션 -->
                                    <div class="form-group mb-5">
                                        <div class="row">
                                            <label for="reservePetName" class="col-xl-5 col-form-label text-center text-xl-end px-2">반려동물 선택:</label>
                                            <div class="col-xl-7">
                                                <div class="form-group mb-0">
                                                    <div class="select-default select-category-2">
                                                        <select id="reservePetName" name="reservePetName" class="select-option">
                                                            <option value="" disabled selected>반려동물을 선택하세요</option>
                                                            <option th:each="pet : ${getMemberPetList}" th:value="${pet.petName}" th:text="${pet.petName}">Pet Name</option>
                                                        </select>
                                                    </div>
                                                </div>
                                            </div>
                                        </div>
                                    </div>
                                    <!-- 반려동물이 없을 때 메시지 표시 -->
                                    <div th:if="${#lists.isEmpty(getMemberPetList)}">
                                        <p>등록된 반려동물이 없습니다. 반려동물을 등록해주세요.</p>
                                    </div>
                                    <div class="form-group mb-5">
                                        <div class="row align-items-center">
                                            <label class="control-label col-xl-5 text-center text-xl-end">참여인원:</label>
                                            <div class="col-xl-5">
                                                <div class="count-input mx-auto">
                                                    <a class="incr-btn" data-action="decrease" href="javascript:void(0)">-</a>
                                                    <input class="quantity" type="number" value="1">
```

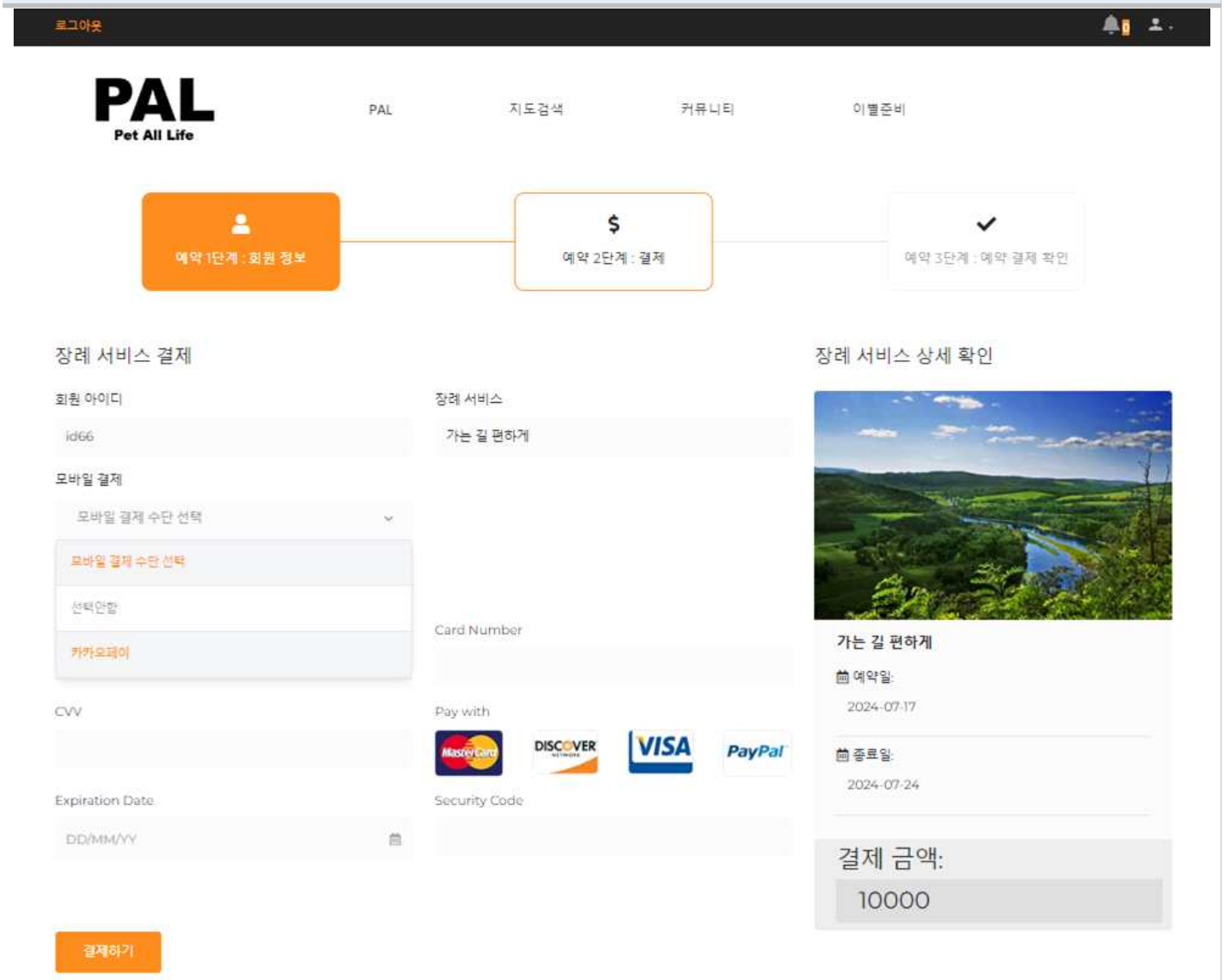
```
                                        <a class="incr-btn" data-action="increase" href="javascript:void(0)">+</a>
                                    </div>
                                </div>
                            </div>
                        </div>
                        <div class="d-flex justify-content-between border-bottom mb-5 pb-5">
                            <span class="text-uppercase h4 mb-0">전체 비용</span>
                            <span class="font-weight-bold text-primary h3 mb-0" id="totalPrice" th:text="0"></span>
                        </div>
                        <input type="hidden" name="reserveCompanyCode" th:value="${reserveCompanyCode}" />
                        <div class="d-grid text-center px-4">
                            <button type="submit" class="btn btn-hover btn-lg btn-outline-secondary text-uppercase">
                                예약 하기 <span class="ms-4"><i class="fa fa-angle-right" aria-hidden="true"></i></span>
                            </button>
                        </div>
                    </div>
                </div>
            </form>
        </div>
        <div class="col-md-7 col-lg-8">
            <div class="col-md-12 col-lg-12" th:if="${serviceListCcode != null}" th:each="ml : ${serviceListCcode}">
                <div class="service-card">
                    <div th:each="img : ${imagesByServiceCode[ml.funeralserviceCode]}">
                        <a href="javascript:void(0)" class="position-relative">
                            <img class="card-img-top" th:src="@{${img.filePath}}" alt="이미지">
                            <div class="card-img-overlay card-hover-overlay rounded-top d-flex flex-column">
                            </div>
                        </a>
                    </div>
                    <div class="service-card-content">
                        <div class="service-detail">
                            <h2 th:text="${ml.funeralserviceTitle}"></h2>
                        </div>
```

```
                                    <div class="service-detail">
                                        <span class="label">장례 서비스 가격:</span>
                                        <h3 th:text="${ml.funeralservicePrice}"></h3>
                                    </div>
                                    <div class="service-card-divider"></div>
                                    <div class="service-detail">
                                        <span class="label">코멘트:</span>
                                        <p th:text="${ml.getFuneralserviceComment()}"></p>
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </section>
    </div><!-- element wrapper ends -->
</th:block>
<!-- 사용자정의 javascript file -->
<th:block layout:fragment="customJsFile">
    <script th:src="@{/js/jquery-3.7.1.js}"></script>
</th:block>
<!-- 사용자정의 javascript -->
<th:block layout:fragment="customJs">
    <script th:inline="javascript">
        $(document).ready(function (){
            const reserveStartDate = $('input[name="reserveStartDate"]');
            const reserveEndDate = $('input[name="reserveEndDate"]');
            if (reserveStartDate.length !== 0) {
                reserveStartDate.daterangepicker({
                    autoUpdateInput: false,
                    singleDatePicker: true,
                    locale: {
                        cancelLabel: 'Clear'
                    }
                });
            }
            if (reserveEndDate.length !== 0) {
                reserveEndDate.daterangepicker({
                    autoUpdateInput: false,
                    singleDatePicker: true,
```

```
                    locale: {
                        cancelLabel: 'Clear'
                    }
                });
            }
            $('input[name="reserveStartDate"],input[name="reserveEndDate"]').on('apply.daterangepicker',
function (ev, picker) {
                $(this).val(picker.startDate.format('YYYY-MM-DD'));
            });
            $('input[name="reserveStartDate"],input[name="reserveEndDate"]').on('cancel.daterangepicker',
function (ev, picker) {
                $(this).val('');
            });
            $('#funeralserviceTitle').change(function (){
                const selectedOption = $(this).find('option:selected');
                const price = parseInt(selectedOption.data('price'));
                const ccode = selectedOption.data('ccode');
                $('#totalPrice').text(`$${price.toFixed(0)}`);
                $('#reserveCompanyCode').val(ccode);
            });
        });
        $('#reserveBtn').click(function (event){
            event.preventDefault(); // 폼의 기본 제출을 막음
            let isSubmit = true;
            $('#funeralReserveForm input, #funeralReserveForm select').each((idx, element) => {
                const value = element.value.trim();
                if(!value){
                    alert('필수 입력 항목입니다.');
                    $(element).focus();
                    isSubmit = false;
                    return false;
                }
            });
            if (isSubmit) {
                const selectedOption = $('#funeralserviceTitle').find('option:selected');
                const ccode = selectedOption.data('ccode');
                $('#reserveCompanyCode').val(ccode);
                // AJAX 요청 보내기
                $.ajax({
                    url: '/funeral/funeral_service_detail',
                    type: 'POST',
```

```
                    data: $('#funeralReserveForm').serialize(), // 폼 데이터를 직렬화하여 전송
                    success: function(response) {
                        // 성공 시 처리할 코드 작성
                        alert('예약이 성공적으로 완료되었습니다.');
                        window.location.href = '/funeral/funeral_reserve_info'; // 예약 정보 페이지로 리디렉
션
                    },
                    error: function(xhr, status, error) {
                        // 에러 시 처리할 코드 작성
                        alert('예약 중 오류가 발생하였습니다.');
                        console.error('Error: ' + error);
                    }
                });
            }
        });
    </script>
</th:block>
</html>
```

| Fuction | funeral_payment (장례 서비스 결제) |
| --- | --- |
| Image | |



| Package | ks51team03/funeral/ReserveController | Controller | ReserveController |
|---|---|---|---|

```
package ks51team03.funeral.reserve.controller;
import jakarta.servlet.http.HttpSession;
import ks51team03.funeral.payment.dto.PaymentDto;
import ks51team03.funeral.reserve.dto.ReserveDto;
import ks51team03.funeral.reserve.dto.ReserveInfoDto;
import ks51team03.funeral.reserve.dto.ReservePaymentDto;
import ks51team03.funeral.reserve.dto.ReserveServiceInfoDto;
import ks51team03.funeral.reserve.service.ReserveService;
import ks51team03.funeral.serviceList.service.ServiceListService;
import ks51team03.member.dto.Member;
import ks51team03.member.service.MemberServiceImpl;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
@Controller
@RequiredArgsConstructor
@Slf4j
public class ReserveController {
    private final ServiceListService serviceListService;
    private final ReserveService reserveService;
    private final MemberServiceImpl memberService;
    // 결제 콜백 처리
    @PostMapping("/reserve/callback")
    public ResponseEntity<?> callbackReceive(@RequestBody ReservePaymentDto reservePaymentDto) {
        log.info("Received payment callBack: {}", reservePaymentDto);
        try {
```

```
            String txId = reservePaymentDto.getTxId();
            String code = reservePaymentDto.getCode();
            String message = reservePaymentDto.getMessage();
            log.info("--- after payment receive ---");
            log.info("txId: {}", txId);
            log.info("error_code: {}", code);
            log.info("error_message: {}", message);
            // 결제 조회 API를 통해 가맹점이 의도한 금액과 결과 금액이 같은지 검증단계
            reserveService.handlePaymentCallback(reservePaymentDto);
            log.info("여기는 handlePaymentCallback 받는 곳");

            log.info("reserveService.handlePaymentCallback(reservePaymentDto): {}", reservePaymentDto);
            // 정상 처리 응답
            Map<String, String> response = new HashMap<>();
            response.put("status", "성공");
            return ResponseEntity.ok(response);
        } catch (Exception e) {
            log.error("Exception during payment callback processing", e);
            // 예외 처리 응답
            Map<String, String> response = new HashMap<>();
            response.put("status", "실패");
            response.put("message", "처리 실패 : 송중기에게 문의해 주세요");
            return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).body(response);
        }
    }
    @GetMapping("funeral/funeral_reserve_info")
    public String reserveInfo(Model model, HttpSession session, ReserveInfoDto reserveInfoDto,
ReserveServiceInfoDto reserveServiceInfoDto) {
            String memberId = (String) session.getAttribute("SID");
        log.info("장례 예약한 회원 아이디 memberId = {}", memberId);
        log.info("여기는 멤버아이디 나오는 곳");
        //회원 정보 조회
        Member member = memberService.getMemberInfoById(memberId);
        String reservePhone = member.getMemberPhone();
        reserveInfoDto.setReserveId(memberId);
        reserveInfoDto.setReservePhone(reservePhone);
        reserveServiceInfoDto.setReserveId(memberId);
        List<ReserveInfoDto> reserveInfoList = reserveService.funeralReserveInfo(reserveInfoDto);
        List<ReserveServiceInfoDto>                    reserveServiceInfoDtoList                    =
reserveService.funeralReserveServiceInfo(reserveServiceInfoDto);
        log.info("reserveInfoList = {}", reserveInfoList);
```

```
        log.info("reserveServiceInfoDtoList = {}", reserveServiceInfoDtoList);
        model.addAttribute("reserveInfoList", reserveInfoList);
        model.addAttribute("reserveServiceInfoDtoList", reserveServiceInfoDtoList);
        return "funeral/funeral_reserve_info";
    }
    @GetMapping("funeral/funeral_reserve_payment")
    public String reservePayment(Model model, HttpSession session, ReserveInfoDto reserveInfoDto,
ReserveServiceInfoDto reserveServiceInfoDto) {
        String memberId = (String) session.getAttribute("SID");
        if(memberId == null) {
            return "redirect:/";
        }
        log.info("장례 예약한 회원 아이디 memberId = {}", memberId);
        log.info("여기는 멤버아이디 나오는 곳");
        //회원 정보 조회
        Member member = memberService.getMemberInfoById(memberId);
        String reservePhone = member.getMemberPhone();
        reserveInfoDto.setReserveId(memberId);
        reserveInfoDto.setReservePhone(reservePhone);
        reserveServiceInfoDto.setReserveId(memberId);
        List<ReserveInfoDto> reserveInfoList = reserveService.funeralReserveInfo(reserveInfoDto);
        List<ReserveServiceInfoDto> reserveServiceInfoDtoList =
reserveService.funeralReserveServiceInfo(reserveServiceInfoDto);
        log.info("reserveInfoList = {}", reserveInfoList);
        log.info("reserveServiceInfoDtoList = {}", reserveServiceInfoDtoList);
        model.addAttribute("reserveInfoList", reserveInfoList);
        model.addAttribute("reserveServiceInfoDtoList", reserveServiceInfoDtoList);
        // fpcode 생성
        String fpcode = reserveService.generateFpcode();
        model.addAttribute("fpcode", fpcode);
        log.info("fpcode = {}", fpcode);
        // 기타 필요한 데이터 설정
        model.addAttribute("memberId", memberId);
        model.addAttribute("reservePhone", reservePhone);
        return "funeral/funeral_reserve_payment";
    }
    @GetMapping("funeral/funeral_reserve_confirm")
    public String reserveConfirm() {
        return "funeral/funeral_reserve_confirm";
    }
}
```

| Package | ks51team03/funeral/reserve | Service | ReserveService |
|---|---|---|---|

```
package ks51team03.funeral.reserve.service;
import ks51team03.funeral.reserve.dto.ReserveDto;
import ks51team03.funeral.reserve.dto.ReserveInfoDto;
import ks51team03.funeral.reserve.dto.ReservePaymentDto;
import ks51team03.funeral.reserve.dto.ReserveServiceInfoDto;
import ks51team03.funeral.reserve.mapper.ReserveMapper;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import java.util.List;
@RequiredArgsConstructor
@Transactional
@Service
@Slf4j
public class ReserveService {
    private final ReserveMapper reserveMapper;
    public void funeralReserve(ReserveDto reserveDto) {
        reserveMapper.funeralReserve(reserveDto);
    }
    //장례 예약 서비스 코드 만들기위한 마지막 코드 찾기
    public String getLastFuneralServiceCode(){
        String getLastFuneralServiceCode = reserveMapper.getLastFuneralServiceCode();
        int newCodeNumber = Integer.parseInt(getLastFuneralServiceCode.substring(2)) + 1;
        return "fr" + newCodeNumber;
    }
    // 장례 예약자 인적사항 확인 맵핑
    public List<ReserveInfoDto> funeralReserveInfo(ReserveInfoDto reserveInfoDto) {
        reserveMapper.funeralReserveInfo(reserveInfoDto);
        return  reserveMapper.funeralReserveInfo(reserveInfoDto);
    }
    // 장례 예약 내역 확인
    public List<ReserveServiceInfoDto> funeralReserveServiceInfo(ReserveServiceInfoDto
reserveServiceInfoDto) {
        reserveMapper.funeralReserveServiceInfo(reserveServiceInfoDto);
        return  reserveMapper.funeralReserveServiceInfo(reserveServiceInfoDto);
    }
    // 결제 콜백 로직
    @Transactional
```

```
    public void handlePaymentCallback(ReservePaymentDto reservePaymentDto) {
        // 결제 정보를 저장하는 로직
        log.info("Insering payment: {}", reservePaymentDto);
        reserveMapper.insertPayment(reservePaymentDto);
        log.info("Paymenet inserted successfully");
        // 필요한 추가 로직 구현 (예: 결제 상태 업데이트, 알림 전송 등)
    }
    //fpcode 생성 메서드
    public String generateFpcode(){
        String lastFpcode = reserveMapper.getLastFpcode();
        int newFpcdoe = (lastFpcode == null) ? 1 : Integer.parseInt(lastFpcode.substring(2)) + 1;
        return "fp" + newFpcdoe;
    }
}
```

| Package | ks51team03/funeral/reserve | Mapper/xml | FuneralReserveMapper.xml |
|---|---|---|---|

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"https://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="ks51team03.funeral.reserve.mapper.ReserveMapper">
    <resultMap id="getFuneralReserve" type="ks51team03.funeral.reserve.dto.ReserveDto">
        <id          column="frcode"          property="reserveCode"/>
        <result      column="ccode"           property="reserveCompanyCode"/>
        <result      column="fscode"          property="reserveServiceCode"/>
        <result      column="id"              property="reserveId"/>
        <result      column="fr_start_date"   property="reserveStartDate"/>
        <result      column="fr_end_date"     property="reserveEndDate"/>
        <result      column="fr_phone"        property="reservePhone"/>
        <result      column="fr_payment"      property="reservePayment"/>
        <result      column="fr_confirm"      property="reserveConfirm"/>
        <result      column="fr_reg_date"     property="reserveRegDate"/>
        <result      column="fr_reseve_time"  property="reserveTime"/>
        <result      column="fr_pet_name"     property="reservePetName"/>
    </resultMap>
    <resultMap id="getFuneralReserveInfo" type="ks51team03.funeral.reserve.dto.ReserveInfoDto">
        <id          column="id"              property="reserveId"/>
        <result      column="name"            property="reserveName"/>
        <result      column="fr_pet_name"     property="reservePetName"/>
        <result      column="phone"           property="reservePhone"/>
        <result      column="email"           property="reserveEmail"/>
```

```
        <result      column="fr_reg_date"          property="reserveRegDate"/>
    </resultMap>
    <resultMap                                                          id="getFuneralReserveServiceInfo"
type="ks51team03.funeral.reserve.dto.ReserveServiceInfoDto">
        <id          column="id"                   property="reserveId"/>
        <result      column="frcode"               property="reserveCode"/>
        <result      column="ccode"                property="reserveCompanyCode"/>
        <result      column="fscode"               property="reserveServiceCode"/>
        <result      column="fr_start_date"        property="reserveStartDate"/>
        <result      column="fr_end_date"          property="reserveEndDate"/>
        <result      column="fs_title"             property="funeralserviceTitle"/>
        <result      column="fs_price"             property="funeralservicePrice"/>
        <result      column="com_name"             property="companyName"/>
    </resultMap>
    <resultMap  id="insertPayment"  type="ks51team03.funeral.reserve.dto.ReservePaymentDto">
        <id          column="fpcode"               property="fpcode"/>
        <result      column="txId"                 property="txId"/>
        <result      column="code"                 property="code"/>
        <result      column="message"              property="message"/>
        <result      column="ccode"                property="ccode"/>
        <result      column="id"                   property="reserveId"/>
        <result      column="frcode"               property="frcode"/>
        <result      column="fp_name"              property="fpName"/>
        <result      column="fp_final"             property="fpFinal"/>
        <result      column="fp_pay_date"          property="fpPayDate"/>
        <result      column="fp_method"            property="fpMethod"/>
        <result      column="fp_status"            property="fpStatus"/>
    </resultMap>
    <resultMap  id="getConfirmPaymentInfo"  type="ks51team03.funeral.payment.dto.PaymentDto">
        <id          column="fpcode"               property="fpcode"/>
        <result      column="ccode"                property="ccode"/>
        <result      column="fscode"               property="fscode"/>
        <result      column="frcode"               property="frcode"/>
        <result      column="id"                   property="reserveId"/>
        <result      column="fp_name"              property="fpName"/>
        <result      column="fp_final"             property="fpFinal"/>
        <result      column="fp_pay_date"          property="fpPayDate"/>
        <result      column="fp_method"            property="fpMethod"/>
        <result      column="fp_status"            property="fpStatus"/>
        <result      column="com_name"             property="comName"/>
        <result      column="fr_pet_name"          property="reservePetName"/>
```

```
</resultMap>
<select id="getLastFuneralServiceCode" parameterType="String" resultType="String">
    /* 장례 예약 기본키 코드 중 제일 큰 숫자 가져오기*/
    SELECT Max(frcode) FROM funeral_reserve
</select>
<insert id="funeralReserve" parameterType="Map">
    <selectKey keyProperty="reserveCode" resultType="String" order="BEFORE">
        /* 상품 자동 증가 코드 */
        SELECT
        CASE
        WHEN COUNT(frcode) = 0 THEN 'fr1'
        ELSE
        CONCAT(
        'fr',
        LPAD(MAX(CAST(SUBSTRING_INDEX(frcode, 'fr', -1) AS UNSIGNED))+1, 3, '0')
        )
        END AS reserveCode
        FROM
        funeral_reserve;
    </selectKey>
    /* 장례 예약 */
    INSERT INTO funeral_reserve
    (   frcode,
        ccode,
        fscode,
        id,
        fr_start_date,
        fr_end_date,
        fr_phone,
        fr_payment,
        fr_confirm,
        fr_reg_date,
        fr_reserve_time,
        fr_pet_name)
    VALUES (
        #{reserveCode},
        #{reserveCompanyCode},
        #{reserveServiceCode},
        #{reserveId},
        #{reserveStartDate},
        #{reserveEndDate},
```

```
            #{reservePhone},
            '결제 전',
            '예약 완료',
            NOW(),
            #{reserveTime},
            #{reservePetName}
        )
    </insert>
    <!-- 장례 예약버튼 누른 후 인적사항 확인 -->
    <select id="funeralReserveInfo" resultMap="getFuneralReserveInfo">
        /* 예약 후 인적사항 조회*/
        SELECT
        fr.id,
        fr.fr_pet_name,
        m.name,
        m.phone,
        m.email,
        fr.fr_reg_date
        FROM
        funeral_reserve AS fr
        INNER join
        pet AS p
        ON
        p.id = fr.id
        INNER join
        member AS m
        on
        m.id = fr.id
        WHERE
        fr.id = #{reserveId}
        ORDER BY
        CAST(SUBSTRING(fr.frcode, 3) AS UNSIGNED) DESC
        LIMIT 1
    </select>
    <!-- 장례 예약버튼 누른 후 인적사항 확인 -->
    <select id="funeralReserveServiceInfo" resultMap="getFuneralReserveServiceInfo">
        /* 예약 후 예약한 정보 조회*/
        SELECT
            fr.frcode,
            fr.fscode,
            fs.ccode,
```

```
            fs.fs_title,
            fs.fs_price,
            fr.fr_start_date,
            fr.fr_end_date,
            fr.id,
            c.com_name
        FROM
            funeral_reserve AS fr
            INNER join
            funeral_service AS fs
            ON
            fs.ccode = fr.ccode
            INNER join
            company AS c
            on
            fs.fscode = fr.fscode
        WHERE
         fr.id = #{reserveId}
        ORDER BY
        CAST(SUBSTRING(fr.frcode, 3) AS UNSIGNED) DESC
        LIMIT 1
    </select>
    <insert id="insertPayment" parameterType="ks51team03.funeral.reserve.dto.ReservePaymentDto">
        <selectKey keyProperty="fpcode" resultType="String" order="BEFORE">
            /* 결제 코드 자동 증가 코드 */
            SELECT
            CASE
            WHEN COUNT(fpcode) = 0 THEN 'fp001'
            ELSE
            CONCAT(
            'fp',
            LPAD(MAX(CAST(SUBSTRING(fpcode, 3) AS UNSIGNED)) + 1,
            LENGTH(MAX(CAST(SUBSTRING(fpcode, 3) AS UNSIGNED)) + 1), '0')
            )
            END AS fpcode
            FROM
            funeral_payment
            FOR UPDATE;
        </selectKey>
        /* 장례 결제 */
        INSERT INTO funeral_payment
```

```
        (    txId,
        code,
        message,
        fpcode,
        ccode,
        id,
        frcode,
        fp_name,
        fp_final,
        fp_pay_date,
        fp_method,
        fp_status)
        VALUES (
        #{txId},
        #{code},
        #{message},
        #{fpcode},
        #{ccode},
        #{reserveId},
        #{frcode},
        #{fpName},
        #{fpFinal},
        NOW(),
        #{fpMethod},
        #{fpStatus}
        )
    </insert>
    <select id="getLastFpcode" resultType="String">
        /*마지막 결제 코드 찾는 select문*/
        SELECT
            fpcode
        FROM
            funeral_payment
        ORDER BY CAST(SUBSTRING(fpcode, 3) AS UNSIGNED) DESC
        LIMIT 1;
    </select>
    <select id="confirmPayment" resultMap="getConfirmPaymentInfo">
        /* 장례 결제 내역 및 정보를 확인하기 위한 select문*/
        SELECT
            fp.fpcode,
            fp.ccode,
```

```
            fr.fscode,
            fp.frcode,
            fr.id,
            fp.fp_final,
            c.com_name,
            fr_pet_name,
            fp.fp_method,
            fp.fp_pay_date,
            fp.fp_status,
            fp.fp_name
        FROM
            funeral_reserve AS fr
            INNER join
            funeral_service AS fs
            ON
            fs.fscode = fr.fscode
            INNER join
            funeral_payment AS fp
            ON
            fp.frcode = fr.frcode
            INNER join
            pet AS p
            on
            fp.id = p.id
            INNER join
            company AS c
            ON
            c.ccode = fp.ccode
        WHERE
            fp.id = #{reserveId}
    </select>
</mapper>
```

| Package | Ks51team03/funeral/reserve | DTD | ReserveDto |
|---|---|---|---|

```
package ks51team03.funeral.reserve.dto;
import lombok.Data;
@Data
public class ReserveDto {
        private String reserveCode;
        private String reserveCompanyCode; // 업체코드
        private String reserveServiceCode; // 장례 서비스 이름
```

```
    private String reserveId; // 서비스가격
    private String reserveStartDate; // 예약일
    private String reserveEndDate; //예약 종료일
    private String reservePhone; // 주문자 연락처
    private String reservePayment; // 결제 상태
    private String reserveConfirm; // 예약 확인
    private String reserveRegDate;  //예약 등록일
    private String reserveTime; //예약 시간
    private String reservePetName; //예약된 반려동물 이름
}
```

| Package | Ks51team03/funeral/reserve | HTML | funeral_reserve_payment.html |
|---|---|---|---|

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org"
xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
layout:decorate="~{layout/default}">
<head>
</head>
<th:block layout:fragment="customContent">
    <div class="main-wrapper booking-step-2">
        <!-- ==================================
        ———     BOOKING SECTION
        ================================== -->
        <section class="py-8 py-md-10">
            <div class="container">
                <div class="mb-8 mt-10">
                    <div class="row progress-wizard">
                        <div class="col-4 progress-wizard-step complete">
                            <div class="progress">
                                <div class="progress-bar"></div>
                            </div>
                            <a th:href="@{#}" class="progress-wizard-dot">
                                <div class="progress-wizard-content">
                                    <i class="fa fa-user" aria-hidden="true"></i>
                                    <span class="d-block">예약 1단계 : 회원 정보</span>
                                </div>
                            </a>
                        </div>
                        <div class="col-4 progress-wizard-step active">
                            <div class="progress">
                                <div class="progress-bar"></div>
```

```
                            </div>
                            <a th:href="@{#}" class="progress-wizard-dot">
                                <div class="progress-wizard-content">
                                    <i class="fas fa-dollar-sign" aria-hidden="true"></i>
                                    <span class="d-block">예약 2단계 : 결제</span>
                                </div>
                            </a>
                        </div>
                        <div class="col-4 progress-wizard-step incomplete">
                            <div class="progress">
                                <div class="progress-bar"></div>
                            </div>
                            <a th:href="@{#}" class="progress-wizard-dot">
                                <div class="progress-wizard-content">
                                    <i class="fa fa-check" aria-hidden="true"></i>
                                    <span class="d-block">예약 3단계 : 예약 결제 확인</span>
                                </div>
                            </a>
                        </div>
                    </div>
                </div>
                <div class="row" th:if="${reserveServiceInfoDtoList != null}" th:each="l : ${reserveServiceInfoDtoList}">
                    <div class="col-md-7 col-lg-8 order-1 order-md-0">
                        <h3 class="text-capitalize mb-5">장례 서비스 결제</h3>
                        <form action="" method="post" target="_blank">
                            <div class="mb-5">
                                <div class="row">
                                    <div class="col-lg-6">
                                        <div class="form-group">
                                            <label for="reserveId">회원 아이디</label>
                                            <input type="text" class="form-control border-0 bg-smoke" id="reserveId" th:value="${l.getReserveId()}">
                                        </div>
                                    </div>
                                    <div class="col-lg-6">
                                        <div class="form-group">
                                            <label for="funeralserviceTitle">장례 서비스</label>
                                            <input type="text" class="form-control border-0 bg-smoke" id="funeralserviceTitle" th:value="${l.getFuneralserviceTitle()}">
                                        </div>
```

```
                                    </div>
                                    <div class="col-lg-6">
                                        <div class="form-group">
                                            <label for="fpMethod">모바일 결제</label>
                                            <div class="select-default select-confirm">
                                                <select class="select-option" id="fpMethod" name="fpMethod" onchange="togglePaymentOptions(this.value)">
                                                    <option value="EASY_PAY">모바일 결제 수단 선택</option>
                                                    <option value="">선택안함</option>
                                                    <option value="KAKAOPAY">카카오페이</option>
                                                </select>
                                            </div>
                                        </div>
                                    </div>
                                    <input type="hidden" id="ccode" th:value="${l.getReserveCompanyCode()}">
                                    <input type="hidden" id="frcode" th:value="${l.getReserveCode()}">
                                    <input type="hidden" id="fpName" th:value="${l.getFuneralserviceTitle()}">
                                    <input type="hidden" id="fpFinal" th:value="${l.getFuneralservicePrice()}">
                                    <input type="hidden" id="fpPayDate" th:value="${l.getReserveStartDate()}">
                                    <input type="hidden" id="fpMethod" th:value="CARD">
                                    <input type="hidden" id="fpStatus" th:value="'결제완료'">
                                </div>
                            </div>
                            <div class="mb-8" id="cardPaymentSection">
                                <h3 class="text-capitalize mb-5">카드 결제</h3>
                                <div class="row">
                                    <div class="col-lg-6">
                                        <div class="form-group">
                                            <label for="cardPayment">결제 카드 선택</label>
                                            <div class="select-default select-confirm">
                                                <select class="select-option" id="cardPayment">
                                                    <option>카드 결제</option>
                                                    <option>우리은행</option>
                                                    <option>신한은행</option>
                                                    <option>전북은행</option>
                                                    <option>농협</option>
                                                    <option>국민은행</option>
```

```
                                        <option>기업은행</option>
                                    </select>
                                </div>
                            </div>
                        </div>
                        <div class="col-lg-6">
                            <div class="form-group">
                                <label for="cardNumber">Card Number</label>
                                <input type="number" class="form-control border-0 bg-smoke" id="cardNumber">
                            </div>
                        </div>
                        <div class="col-lg-6">
                            <div class="form-group">
                                <label for="cvv">CVV</label>
                                <input type="text" class="form-control border-0 bg-smoke" id="cvv">
                            </div>
                        </div>
                        <div class="col-lg-6">
                            <div class="form-group">
                                <label for="payWith">Pay with</label>
                                <div class="row" id="payWith">
                                    <div class="col-3 pe-lg-1 pe-xl-3">
                                        <div class="img-overlay rounded">
                                            <img class="w-100 lazyestload" th:data-src="@{/assets/img/booking/master-card.jpg}" th:src="@{/assets/img/booking/master-card.jpg}" alt="Generic placeholder image">
                                            <a href="" class="hover-img-overlay-dark"></a>
                                        </div>
                                    </div>
                                    <div class="col-3 px-lg-2 px-xl-3">
                                        <div class="img-overlay rounded">
                                            <img class="w-100 lazyestload" th:data-src="@{/assets/img/booking/discover.jpg}" th:src="@{/assets/img/booking/discover.jpg}" alt="Generic placeholder image">
                                            <a href="" class="hover-img-overlay-dark"></a>
                                        </div>
                                    </div>
                                    <div class="col-3 px-lg-2 px-xl-3">
                                        <div class="img-overlay rounded">
                                            <img class="w-100 lazyestload"
```

```
th:data-src="@{/assets/img/booking/visa.jpg}" th:src="@{/assets/img/booking/visa.jpg}" alt="Generic placeholder image">
                                        <a href="" class="hover-img-overlay-dark"></a>
                                    </div>
                                </div>
                                <div class="col-3 ps-lg-1 ps-xl-3">
                                    <div class="img-overlay rounded">
                                        <img class="w-100 lazyestload" th:data-src="@{/assets/img/booking/paypal.jpg}" th:src="@{/assets/img/booking/paypal.jpg}" alt="Generic placeholder image">
                                        <a href="" class="hover-img-overlay-dark"></a>
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                    <div class="col-lg-6">
                        <label for="expirationDate">Expiration Date</label>
                        <div class="form-group form-group-icon form-group-icon-default" id="expirationDate">
                            <i class="far fa-calendar-alt" aria-hidden="true"></i>
                            <input type="text" class="form-control border-0 bg-smoke" name="dateRange" value="" placeholder="DD/MM/YY">
                        </div>
                    </div>
                    <div class="col-lg-6">
                        <div class="form-group">
                            <label for="securityCode">Security Code</label>
                            <input type="password" id="securityCode" class="form-control border-0 bg-smoke">
                        </div>
                    </div>
                </div>
            </div>
            <div class="d-flex justify-content-between">
                <button type="button" onclick="requestPayment()" class="btn btn-primary text-uppercase">
                    결제하기
                </button>
            </div>
        </form>
    </div>
```

```
                    <div class="col-md-5 col-lg-4" th:if="${reserveServiceInfoDtoList != null}" th:each="l : ${reserveServiceInfoDtoList}">
                        <h3 class="mb-5">장례 서비스 상세 확인</h3>
                        <div class="card bg-smoke mb-6 mb-md-0" >
                            <img class="card-img-top lazyestload" th:data-src="@{/assets/img/booking/booking.jpg}" th:src="@{/assets/img/booking/booking.jpg}" alt="Card image cap">
                            <div class="card-body">
                                <h6 class="card-title text-dark" th:text="${l.funeralserviceTitle}"></h6>
                                <ul class="list-group list-group-flush">
                                    <li class="list-group-item bg-transparent border-top-0 px-0 py-2" >
                                        <span><i class="far fa-calendar-alt me-1" aria-hidden="true"></i>예약일:</span>
                                        <input type="text" class="form-control border-0 bg-smoke" th:value="${l.reserveStartDate}">
                                    </li>
                                    <li class="list-group-item bg-transparent px-0 py-2 border-off-white">
                                        <span><i class="far fa-calendar-alt me-1" aria-hidden="true"></i>종료일:</span>
                                        <input type="text" class="form-control border-0 bg-smoke" th:value="${l.reserveEndDate}">
                                    </li>
                                    <li class="list-group-item bg-transparent px-0 py-4 d-none"></li>
                                </ul>
                            </div>
                            <div class="card-footer mt-2">
                                <h2 class="mb-0">
                                    <span>결제 금액:</span>
                                    <input type="text" class="card-footer mt-1" th:value="${l.funeralservicePrice}" readonly>
                                </h2>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </section>
    </div><!-- element wrapper ends -->
</th:block>
<!-- 사용자정의 javascript file -->
<th:block layout:fragment="customJsFile">
    <script type="text/javascript" th:src="@{https://cdn.portone.io/v2/browser-sdk.js}"></script>
```

```
    <script type="text/javascript" th:src="@{https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js}"></script>
</th:block>
<!-- 사용자정의 javascript -->
<th:block layout:fragment="customJs">
    <script th:inline="javascript">
        const fpcode =/*[[${fpcode}]]*/ '';
        console.log("fpcode: ", fpcode);
        const data = {
            storeId: 'store-867d1a56-0dcf-4f7c-adbf-6b76defce016',
            channelKey: 'channel-key-eba90cc8-136a-4232-80b4-fefa415d54d0',
            paymentId: fpcode,
            orderName: '송재익패키지',
            totalAmount: 1000,
            currency: 'CURRENCY_KRW',
            payMethod: "EASY_PAY",
            easyPay: {
                easyPayProvider: "KAKAOPAY",
            }
        };
        console.log("Payment data: ", data);
        async function requestPayment() {
            try {
                console.log("Requesting payment with data: ", data);
                const response = await PortOne.requestPayment(data);
                console.log("Payment response: ", response);
                if (response.code != null) {
                    return alert(response.message("사용자가 결제 프로세스를 중단하였습니다."));
                }
                const reserveIdElement = document.getElementById("reserveId");
                const fpNameElement = document.getElementById("fpName");
                const ccodeElement = document.getElementById("ccode");
                const fpFinalElement = document.getElementById("fpFinal");
                const fpPayDateElement = document.getElementById("fpPayDate");
                const frcodeElement = document.getElementById("frcode");
                const fpMethodElement = document.getElementById("fpMethod");
                const fpStatusElement = document.getElementById("fpStatus");
                if (!reserveIdElement || !fpNameElement || !ccodeElement || !fpFinalElement ||
!fpPayDateElement || !frcodeElement) {
                    throw new Error("Required input elements are missing");
                }
                const reserveId = reserveIdElement.value;
```

```
            const fpName = fpNameElement.value;
            const ccode = ccodeElement.value;
            const fpFinal = fpFinalElement.value;
            const fpPayDate = fpPayDateElement.value;
            const frcode = frcodeElement.value;
            const frMethod = fpMethodElement.value;
            const frStatus = fpStatusElement.value;
            const postData = {
                txId: response.txId,
                paymentId: fpcode,
                code: response.code,
                message: response.message,
                reserveId: reserveId,
                fpName: fpName,
                ccode: ccode,
                fpFinal: fpFinal,
                frcode: frcode,
                fpPayDate: fpPayDate,
                fpMethod: frMethod,
                fpStatus: frStatus
            };
            console.log("Post data: ", postData);
            const validation = await axios.post("/reserve/callback", postData);
            console.log("Validation response: ", validation);
            if (validation.data.status === "성공") {
                window.location.href = "/funeral/funeral_reserve_confirm";
            } else {
                alert(validation.data.message || "결제 처리 중 오류가 발생했습니다.");
            }
        } catch (error) {
            console.error("Error during payment process: ", error);
            alert("결제 처리 중 오류가 발생했습니다. 송송송에게 문의해 주세요.");
        }
    }
    function togglePaymentOptions(value) {
        const cardPaymentSection = document.getElementById('cardPaymentSection');
        if (value === "KAKAOPAY") {
            cardPaymentSection.style.display = 'none'; // Hide the card payment section
        } else {
            cardPaymentSection.style.display = 'block'; // Show the card payment section
        }
```

```
        }
    </script>
</th:block>
</html>
```

| Fuction | | 회원 관리 | |
|---|---|---|---|
| Package | Ks51team03/member | Mapper | MemberMapper |

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"https://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="ks51team03.member.mapper.MemberMapper">
        <resultMap type="Member" id="memberResultMap">
                <!-- pk, column -->
                <id      column="id"                    property="memberId"/>
                <result column="pw"                     property="memberPw"/>
                <result column="name"                   property="memberName"/>
                <result column="level"                  property="memberLevel"/>
                <result column="level_name"     property="memberLevelName"/>
                <result column="grade"                  property="memberGrade"/>
                <result column="gender"                 property="memberGender"/>
                <result column="birth"                  property="memberBirth"/>
                <result column="phone"                  property="memberPhone"/>
                <result column="email"                  property="memberEmail"/>
                <result column="addr"                   property="memberAddr"/>
                <result column="addr_detail"    property="memberAddrDetail"/>
                <result column="post_num"               property="memberPostNum"/>
                <result column="pet"                    property="memberPet"/>
                <result column="sms_check"              property="memberSmsCheck"/>
                <result column="email_check"    property="memberEmailCheck"/>
                <result column="regist_date"    property="memberRgstDate"/>

        </resultMap>

        <resultMap type="MemberLevel" id="memberLevelResultMap">
                <id      column="level"                 property="level" />
                <result column="level_name"     property="levelName" />
        </resultMap>
```

```
<resultMap id="QuestionResultMap" type="ComQuestion">
        <id column="quesnum" property="quesNum" />
        <result column="ccode" property="cCode" />
        <result column="qctenum" property="qcteNum" />
        <result column="id" property="memberId" />
        <result column="ques_content" property="quesContent" />
        <result column="ques_date" property="quesDate" />
        <result column="com_name" property="companyName" />
</resultMap>
<resultMap id="ComReviewResultMap" type="ComReview">
        <id column="revcode" property="revCode" />
        <result column="id" property="memberId" />
        <result column="rev_category" property="revCategory" />
        <result column="ccode" property="cCode" />
        <result column="rev_admin_date" property="revAdminDate" />
        <result column="rev_update_date" property="revUpdateDate" />
        <result column="file_idx" property="fileIdx" />
        <result column="rev_scope" property="revScope" />
        <result column="rev_content" property="revContent" />
        <result column="com_name" property="companyName" />
        <result column="file_path" property="filePath" />
</resultMap>
<resultMap id="ComInformReciPientResultMap" type="ComInformReciPient">
        <id column="ircode" property="irCode" />
        <result column="id" property="memberId" />
        <result column="informcode" property="informCode" />
        <result column="ircheck" property="irCheck" />
        <result column="inform_contents" property="informContent" />
        <result column="sendid" property="sendId" />
        <result column="inform_value" property="informValue" />
        <result column="inform_datetime" property="dateTime" />
</resultMap>
<resultMap id="MemberLikeResultMap" type="MemberLike">
        <id column="lkcode" property="lkCode" />
        <result column="id" property="memberId" />
        <result column="ccode" property="cCode" />
        <result column="lk_alarm" property="lkAlarm" />
        <result column="lk_date" property="lkDate" />
        <result column="lk_state" property="lkState" />
        <result column="com_name" property="companyName" />
</resultMap>
```

```
<select id="memberLikeCheckFirst" parameterType="String" resultMap="MemberLikeResultMap">
        -- 회원 구독 추가 전 이미 있는지 확인
        SELECT count(l.lk_state) AS 'lk_state', l.lkcode
        FROM `like`      AS l
        join
        member AS m
        on
        l.id = m.id
        WHERE ccode = #{cCode} AND l.id = #{memberId}
        GROUP BY l.lkcode;
</select>
<select id="memberLikeCheck" parameterType="String" resultMap="MemberLikeResultMap">
        -- 회원 구독한지 안 한지 작업중
        SELECT l.lk_state AS 'lk_state', l.lkcode
        FROM `like`      AS l
        join
        member AS m
        on
        l.id = m.id
        WHERE ccode = #{cCode} AND l.id = #{memberId}
        GROUP BY l.lkcode;
</select>
<update id="updateMemberLikeStateOn">
        -- 회원 구독 상태 0일때 1로 바꾸기
        UPDATE `like` SET lk_state=1 WHERE lkcode=#{lkCode}
</update>
<update id="updateMemberLikeState">
        -- 회원 구독 취소
        UPDATE `like` SET lk_state=0 WHERE lkcode=#{lkCode}
</update>
<update id="updateMemberLikeAlarm">
        -- 회원 알림 조작
        UPDATE `like` SET lk_alarm=#{lkAlarm} WHERE lkcode=#{lkCode}
</update>
<select id="memberGetLikeCompany" parameterType="String" resultMap="MemberLikeResultMap">
        -- 회원 구독 업체 리스트 조회
        SELECT l.lkcode, l.id, l.ccode, l.lk_alarm, l.lk_date, l.lk_state, c.com_name
        FROM `like` AS l
        JOIN company AS c
        ON l.ccode = c.ccode
        WHERE l.id=#{memberId} AND l.lk_state = 1
```

```
        </select>
        <insert id="memberAddLike" parameterType="String">
                <selectKey keyProperty="lkCode" resultType="String" order="BEFORE">
                        SELECT
                        CASE
                        WHEN COUNT(*) = 0 THEN 'like1'
                        ELSE CONCAT('like', (MAX(CAST(SUBSTRING_INDEX(lkcode, 'like', -1) AS
UNSIGNED)) + 1))
                        END
                        FROM `like`
                </selectKey>
                -- 회원 구독하기
                INSERT INTO `like`
                (lkcode, id, ccode, lk_alarm, lk_date, lk_state)
                VALUES (#{lkCode}, #{memberId}, #{cCode}, 1, NOW(), 1)
        </insert>
        <update id="updateInformStatus">
                -- 알림 확인 상태 수정
                UPDATE inform_recipient
                SET ircheck = 1
                WHERE ircode = #{informId}
        </update>
        <select id="getInform" parameterType="String" resultMap="ComInformReciPientResultMap">
                -- 회원 알림 내용 조회
                SELECT
                        ir.ircode, ir.id, i.inform_contents, ir.ircheck, i.id AS sendid, i.inform_value,
i.inform_datetime
                FROM
                        inform_recipient AS ir
                INNER JOIN
                        inform AS i
                ON
                        ir.informcode = i.informcode
                WHERE
                        ir.id = #{memberId} AND ir.ircheck = 0
        </select>
        <select id="getInformCount" parameterType="String" resultType="int">
                -- 회원 알림 수 조회
                SELECT
                        count(ircode)
                FROM
```

```
                inform_recipient
        WHERE
                id = #{memberId} AND ircheck = 0
    </select>
    <delete id="memberReviewDelete" parameterType="ComReview" >
        -- 회원 리뷰 삭제
        DELETE
        FROM
                service_reviews
        WHERE
                revcode=#{revCode}
    </delete>
    <update id="memberReviewModify" parameterType="ComReview" >
        -- 회원 리뷰 수정
        UPDATE
                service_reviews
        <set>
                rev_update_date = NOW(),
                file_idx = #{fileIdx},
                rev_scope = #{revScope},
                rev_content = #{revContent}
        </set>
        WHERE
                revcode = #{revCode}
    </update>
    <select id="getCompanyReviewByRevCode" parameterType="String"
resultMap="ComReviewResultMap">
        -- 수정용 특정 회원 리뷰 조회
        SELECT
                r.revcode, r.id, c.com_name, r.rev_admin_date, r.rev_update_date, r.file_idx,
r.rev_scope, r.rev_content, f.file_path
        FROM
                service_reviews AS r
        inner JOIN
                company AS c
        ON
                r.ccode = c.ccode
        LEFT JOIN
                files AS f
        ON
                r.file_idx = f.file_idx
```

```
                WHERE
                        r.revcode = #{revCode}
        </select>
        <select id="getCompanyReview" parameterType="String" resultMap="ComReviewResultMap">
                -- 회원 리뷰 조회
                SELECT
                        r.revcode, r.id, c.com_name, r.rev_admin_date, r.rev_update_date, r.file_idx,
r.rev_scope, r.rev_content, f.file_path
                FROM
                        service_reviews AS r
                inner JOIN
                        company AS c
                ON
                        r.ccode = c.ccode
                LEFT JOIN
                        files AS f
                ON
                        r.file_idx = f.file_idx
                WHERE
                        r.id = #{memberId}
                ORDER BY r.rev_admin_date DESC
        </select>
        <delete id="deleteAnswersByQuesNum" parameterType="String">
                -- 답변부터 삭제
                DELETE FROM questions_answer WHERE quesnum = #{quesNum}
        </delete>
        <delete id="memberQuestionDelete" parameterType="String" >
                -- 회원 문의 삭제
                DELETE
                FROM
                questions
                WHERE
                quesnum = #{quesNum}
        </delete>
        <update id="memberQuestionModify" parameterType="ComQuestion" >
                -- 회원 문의 수정
                UPDATE
                        questions
                <set>
                        qctenum = #{qcteNum},
                        ques_content = #{quesContent},
```

```
                ques_date= NOW()
        </set>
        WHERE
                quesnum = #{quesNum}
</update>
<select id="getQuestionById" parameterType="String" resultMap="QuestionResultMap">
        -- 특정 문의 회원 아이디로 검색
        SELECT
                q.quesnum, c.com_name, q.qctenum, q.id, q.ques_content, q.ques_date
        FROM
                questions AS q
        INNER JOIN
                company AS c
        ON
                q.ccode = c.ccode
        WHERE
                q.id= #{memberId}
</select>

<sql id="memberSearch">
        SELECT
                m.id,
                m.pw,
                m.name,
                m.level,
                l.level_name,
                m.email,
                m.addr,
                m.regist_date
        FROM
                member as m
                INNER JOIN
                level as l
                ON
                m.level = l.level
</sql>
<!-- 로그인 테이블 행의 갯수 조회 -->
<select id="getLoginHistoryCnt" resultType="int">
        /* 로그인 테이블 행의 갯수 조회 */
        SELECT
                COUNT(1)
```

```
        FROM
                tb_login;
</select>
<!-- 로그인 이력조회 -->
<select id="getLoginHistory" parameterType="int" resultType="map">
        /* 로그인 이력 조회 */
        SELECT
                l.login_id as loginId,
                m.m_name as memberName,
                m.m_email as memberEmail,
                ml.level_name as levelName,
                l.login_date as loginDate,
                l.logout_date as logoutDate
        FROM
                tb_login AS l
                INNER JOIN
                tb_member AS m
                ON
                l.login_id = m.m_id
                INNER join
                tb_member_level AS ml
                ON
                m.m_level = ml.level_num
        LIMIT #{startRow}, #{rowPerPage};
</select>

<!-- 회원 검색 조회 -->
<select id="getSearchList" parameterType="Search" resultMap="memberResultMap">
        /* 회원조회 */
        <include refid="memberSearch"></include>
        <where>
                <if test="searchKey != null and searchKey != ''">
                        ${searchKey} LIKE CONCAT('%', #{searchValue}, '%')
                </if>
        </where>
        ORDER BY m.id
</select>

<!-- 회원 탈퇴 -->
<delete id="removeMemberById" parameterType="String">
        /* 회원탈퇴 */
```

```
        DELETE
        FROM
                tb_member
        WHERE
                m_id = #{memberId};
</delete>

<!-- 회원 로그인 이력 삭제 -->
<delete id="removeLoginHistoryById" parameterType="String">
        /* 회원 로그인 이력 삭제 */
        DELETE
        FROM
                tb_login
        WHERE
                login_id = #{memberId};
</delete>

<update id="IncreasePetByMemberId" parameterType="String">
/*회원 반려동물 수 추가*/
        UPDATE
                member
        SET
                pet=pet+1
        WHERE
                id=#{memberId};

</update>

<update id="DeclinePetByMemberId" parameterType="String">
/*회원 반려동물 수 감소*/
        UPDATE
                member
        SET
                pet=pet-1
        WHERE
                id=#{memberId};

</update>

<update id="updateMember" parameterType="Member">
        /* 회원수정 */
```

```
UPDATE member
<set>
        <if test="memberPw != null and memberPw != ''">
                pw = #{memberPw},
        </if>
        <if test="memberName != null and memberName != ''">
                name = #{memberName},
        </if>
        <if test="memberLevel != null and memberLevel != ''">
                level = #{memberLevel},
        </if>
        <if test="memberGrade != null and memberGrade != ''">
                grade = #{memberGrade},
        </if>
        <if test="memberGender != null and memberGender != ''">
                gender = #{memberGender},
        </if>
        <if test="memberBirth != null and memberBirth != ''">
                birth = #{memberBirth},
        </if>
        <if test="memberPhone != null and memberPhone != ''">
                phone = #{memberPhone},
        </if>
        <if test="memberEmail != null and memberEmail != ''">
                email = #{memberEmail},
        </if>
        <if test="memberAddr != null and memberAddr != ''">
                addr = #{memberAddr},
        </if>
        <if test="memberAddrDetail != null and memberAddrDetail != ''">
                addr_detail = #{memberAddrDetail},
        </if>
        <if test="memberPostNum != null and memberPostNum != ''">
                post_num = #{memberPostNum},
        </if>
        <if test="memberPet != null and memberPet != ''">
                pet = #{memberPet},
        </if>
        <if test="memberSmsCheck != null and memberSmsCheck != ''">
                sms_check = #{memberSmsCheck},
        </if>
```

```
                <if test="memberEmailCheck != null and memberEmailCheck != ''">
                        email_check = #{memberEmailCheck},
                </if>
                <if test="memberRgstDate != null and memberRgstDate != ''">
                        regist_date = #{memberRgstDate}
                </if>
        </set>
        WHERE
                id = #{memberId}
</update>

<select id="getMemberInfoById" parameterType="String" resultMap="memberResultMap">
        /* 특정회원 정보조회 */
        SELECT
                id,
                pw,
                name,
                level,
                grade,
                gender,
                birth,
                phone,
                email,
                addr,
                addr_detail,
                post_num,
                pet,
                sms_check,
                email_check,
                regist_date
        FROM
                member
        WHERE
                id = #{memberId};
</select>

<insert id="insertMember" parameterType="Member">
        /* 회원가입 */
        INSERT INTO member
        (       id,
                pw,
```

```
                name,
                level,
                grade,
                gender,
                birth,
                phone,
                email,
                addr,
                addr_detail,
                post_num,
                pet,
                sms_check,
                email_check,
                regist_date
        )VALUES(
                #{memberId},
                #{memberPw},
                #{memberName},
                #{memberLevel},
                #{memberGrade},
                #{memberGender},
                #{memberBirth},
                #{memberPhone},
                #{memberEmail},
                #{memberAddr},
                #{memberAddrDetail},
                #{memberPostNum},
                #{memberPet},
                #{memberSmsCheck},
                #{memberEmailCheck},
                CURDATE()
        );
</insert>

<select id="idCheck" parameterType="string" resultType="boolean">
        /* 아이디 중복체크 */
        SELECT
                COUNT(id)
        FROM
                member
        WHERE
```

```
                id = #{memberId};
</select>

<select id="getMemberLevelList" resultMap="memberLevelResultMap">
        /* 회원 등급 조회 */
        SELECT
                level,
                level_name
        FROM
                level
</select>

<select id="getMemberList" resultMap="memberResultMap">
        /* 회원조회 */
        <include refid="memberSearch"></include>
        ORDER BY m.id;
</select>

<select id="getMemberIdByNameEmail" parameterType="Member" resultType="String">
        SELECT
            m.id
        FROM
            member AS m
        WHERE
            m.name = #{memberName} AND m.email = #{memberEmail};
</select>

<select id="getMemberPwByNameEmail" parameterType="Member" resultType="String">
        SELECT
            m.pw
        FROM
            member AS m
        WHERE
            m.id = #{memberId} AND m.email = #{memberEmail};
</select>

<update id="updateMemberPw" parameterType="Member">
        /* 회원비밀번호수정 */
        UPDATE member
        <set>
                <if test="memberPw != null and memberPw != ''">
```

```
                pw = #{memberPw}
            </if>
        </set>
        WHERE
            id = #{memberId}
    </update>

    <!-- application.properties에서 패키지 미리지정 -> mybatis.type-aliases-package=ksmart.mybatis.*.dto -->
    <!-- ksmart.mybatis.member.dto.Member => Member -->
    <!-- ksmart.mybatis.user.dto.User => User -->
    <!--
    <select id="getMemberList" resultType="Member">
        /* 회원조회 */
        SELECT
            m_id AS memberId,
            m_pw AS memberPw,
            m_name AS memberName,
            m_level AS memberLevel,
            m_email AS memberEmail,
            m_addr AS memberAddr,
            m_reg_date AS memberRegDate
        FROM
            tb_member
    </select>
     -->
</mapper>
```

| Package | Ks51team03/member | Service | MemberService |
|---|---|---|---|

```
package ks51team03.member.service;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import ks51team03.company.dto.*;
import ks51team03.company.mapper.CompanyMapper;
import ks51team03.files.dto.FileRequest;
import ks51team03.files.mapper.FileMapper;
import ks51team03.files.util.FileUtils;
import ks51team03.member.dto.MemberLike;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
```

```
import ks51team03.member.dto.Member;
import ks51team03.member.dto.MemberLevel;
import ks51team03.member.dto.Search;
import ks51team03.member.mapper.MemberMapper;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.web.multipart.MultipartFile;
/**
 * @Transactional : 트랜잭션 처리 (ACID)
 * 클래스위에 @Transactional 선언하게 되면 클래스 내부의 모든 메소드에 적용
 * 메소드위에 @Transactional 선언하게 되면 특정메소드에 적용
 */
@Service("memberService")
@Transactional
@RequiredArgsConstructor
@Slf4j
public class MemberServiceImpl implements MemberService{

	private final MemberMapper memberMapper;
	private final CompanyMapper companyMapper;
	private final FileMapper fileMapper;
	/**
	 * 회원 구독 상태 0일때 1로 바꾸기
	 */
	@Override
	public int updateMemberLikeStateOn(String lkCone) {
		return memberMapper.updateMemberLikeStateOn(lkCone);
	}
	/**
	 * 회원 구독 추가 전 존재 여부 확인
	 */
	@Override
	public MemberLike memberLikeCheckFirst(String cCode, String memberId) {
		return memberMapper.memberLikeCheckFirst(cCode,memberId);
	}
	/**
	 * 회원 구독 여부 확인
	 */
	@Override
	public MemberLike memberLikeCheck(String cCode, String memberId) {
		return memberMapper.memberLikeCheck(cCode, memberId);
```

```
}
/**
 * 회원 구독 취소 작업
 */
@Override
public int updateMemberLikeState(String lkCode) {
        return memberMapper.updateMemberLikeState(lkCode);
}
/**
 * 회원 알림 여부 설정
 */
@Override
public int updateMemberLikeAlarm(String lkCode, int lkAlarm) {
        return memberMapper.updateMemberLikeAlarm(lkCode,lkAlarm);
}
/**
 * 회원 구독 업체 리스트 조회
 */
@Override
public List<MemberLike> memberGetLikeCompany(String memberId) {
        return memberMapper.memberGetLikeCompany(memberId);
}
/**
 * 회원 구독 추가
 */
@Override
public int memberAddLike(String memberId, String cCode) {
        return memberMapper.memberAddLike(memberId, cCode);
}
/**
 * 회원의 알림 조회
 */
@Override
public List<ComInformReciPient> getInform(String memberId) {
        return memberMapper.getInform(memberId);
}
/**
 * 회원의 알림 수 조회
 */
@Override
public int getInformCount(String memberId) {
```

```
        Integer count = memberMapper.getInformCount(memberId);
        return count != null ? count : 0;
    }
    /**
     * 회원의 리뷰 삭제
     */
    @Override
    public int memberReviewDelete(ComReview review) {
        FileUtils fileUtils = new FileUtils();
        FileRequest fileRequest = fileMapper.getFileByRevCode(review.getRevCode());
        if (fileRequest != null) {
            fileUtils.deleteFile(fileRequest);
            review.setFileIdx(null);
            memberMapper.memberReviewDelete(review);
            fileMapper.deleteReview(fileRequest.getFileIdx());
        }
        else{
            memberMapper.memberReviewDelete(review);
        }
        return 0;
    }
    /**
     * 회원의 특정 리뷰 수정
     */
    @Override
    public int memberReviewModify(ComReview review, boolean deleteImage, boolean newImage,
MultipartFile revImgFile) {
        if (deleteImage) {
            handleImageDeletion(review);
        }
        if (newImage && revImgFile != null && !revImgFile.isEmpty()) {
            handleImageUpload(review, revImgFile);
        }
        return memberMapper.memberReviewModify(review);
    }
    private void handleImageDeletion(ComReview review) {
        FileRequest fileRequest = fileMapper.getFileByRevCode(review.getRevCode());
        if (fileRequest != null) {
            fileRequest.setFileCate("리뷰");
            FileUtils fileUtils = new FileUtils();
            fileUtils.deleteFile(fileRequest);
```

```
                fileMapper.updateReviewImg(fileRequest.getFileIdx());
                fileMapper.deleteReview(fileRequest.getFileIdx());
                review.setFileIdx(null);
        }
}
private void handleImageUpload(ComReview review, MultipartFile revImgFile) {
        FileUtils fileUtils = new FileUtils();
        FileRequest fileRequest = fileUtils.uploadFile(revImgFile, "리뷰");
        if (fileRequest != null) {
                fileMapper.addFile(fileRequest);
                review.setFileIdx(fileRequest.getFileIdx());
        }
}
/**
 * 회원의 특정 리뷰 검색
 */
@Override
public ComReview getCompanyReviewByRevCode(String revCode){
        return memberMapper.getCompanyReviewByRevCode(revCode);
}
/**
 * 회원 리뷰 검색
 */
@Override
public List<ComReview> getCompanyReview(String memberId) {
        return memberMapper.getCompanyReview(memberId);
}
/**
 * 회원 문의 삭제
 */
@Override
public int memberQuestionDelete(ComQuestion question){
        memberMapper.deleteAnswersByQuesNum(question.getQuesNum());
        return memberMapper.memberQuestionDelete(question);
}
/**
 * 회원 문의 수정
 */
@Override
public int memberQuestionModify(ComQuestion question){
        return memberMapper.memberQuestionModify(question);
```

```
	}
	/**
	 * 회원 문의 조회
	 */
	@Override
	public List<ComQuestion> getQuestionById(String memberId){
		return memberMapper.getQuestionById(memberId);
	}
	/**
	 * 로그인 이력 조회
	 */
	@Override
	public Map<String, Object> getLoginHistory(int currentPage) {
		// 보여줄 행의 갯수
		int rowPerPage = 10;
		// 첫번째 인수값
		int startRow = (currentPage - 1) * rowPerPage;
		// 시작페이지 설정 초기화
		int startPageNum = 1;
		// 마지막페이지 설정 초기화
		int endPageNum = 10;
		List<Map<String,Object>> loginHistory = memberMapper.getLoginHistory(startRow,
rowPerPage);
		// 전체 행의 갯수 조회
		double cnt = memberMapper.getLoginHistoryCnt();
		// 마지막 페이지
		int lastPage = (int)Math.ceil(cnt/rowPerPage);
		// 마지막페이지 보다 작을 경우 마지막페이지로 설정
		endPageNum = lastPage < 10 ? lastPage : endPageNum;
		// 동적 페이지설정
		if(currentPage > 6 && endPageNum > 9){
			startPageNum =  currentPage - 5;
			endPageNum = currentPage + 4;
			// 마지막페이지번호가 마지막페이지수보다 클 경우에 페이지번호를 고정
			if(endPageNum >= lastPage){
				startPageNum = lastPage - 9;
				endPageNum = lastPage;
			}
		}
		Map<String, Object> resultMap = new HashMap<String, Object>();
		resultMap.put("loginHistory", loginHistory);
```

```
            resultMap.put("lastPage", lastPage);
            resultMap.put("startPageNum", startPageNum);
            resultMap.put("endPageNum", endPageNum);
            return resultMap;
    }
    /**
     * 회원 검색 조회
     * @param search
     */
    @Override
    public List<Member> getSearchList(Search search) {
            String searchKey = search.getSearchKey();
            String columnName = "";
            switch (searchKey) {
                    case  "memberId" -> {
                            columnName = "m.m_id";
                    }
                    case  "memberName" -> {
                            columnName = "m.m_name";
                    }
                    case  "memberAddr" -> {
                            columnName = "m.m_addr";
                    }
            }
            search.setSearchKey(columnName);

            return memberMapper.getSearchList(search);
    }

    /**
     * 회원 탈퇴 프로세스
     * @param 회원등급, 회원아이디
     * @detail 회원등급 회원탈퇴
     */
    @Override
    public void removeMember(String memberLevel, String memberId) {
            // 회원등급
            switch (memberLevel) {
            /*
             * // 판매자 (상품테이블, 주문테이블) case 2 -> {
             *
```

```
             * } // 구매자 (주문테이블) case 3 -> {
             *
             * }
             */
            }
            // 공통 (로그인테이블, 회원테이블)
            memberMapper.removeLoginHistoryById(memberId);
            memberMapper.removeMemberById(memberId);
    }

    /**
     * 특정 회원 확인
     * @param memberId, memberPw
     * @return Map<String, Object> 결과 isCheck true: memberInfo , false
     */
    @Override
    public Map<String, Object> checkMemberInfo(String memberId, String memberPw) {
            Map<String, Object> resultMap = new HashMap<String, Object>();
            boolean isCheck = false;

            Member memberInfo = memberMapper.getMemberInfoById(memberId);

            if(memberInfo != null) {
                    String checkPw = memberInfo.getMemberPw();
                    if(checkPw != null && checkPw.equals(memberPw)) {
                            isCheck = true;
                            resultMap.put("memberInfo", memberInfo);
                    }
            }

            resultMap.put("isCheck", isCheck);

            return resultMap;
    }

    /**
     * 특정회원 수정
     */
    @Override
    public int updateMember(Member member) {
```

```
        return memberMapper.updateMember(member);
    }

    /**
     * 특정 회원 정보 조회
     */
    @Override
    public Member getMemberInfoById(String memberId) {

        return memberMapper.getMemberInfoById(memberId);
    }

    /**
     * 회원가입 프로세스
     */
    @Override
    public void insertMember(Member member) {
        int result = memberMapper.insertMember(member);

        //if(result > 0) // 다음 프로세스

    }

    /**
     * 회원등급 조회
     * @author : ksmart51 홍길동 (2024-06-17)
     * @detail : 특이사항에 대한 내용기술
     * @return List<MemberLevel>
     */
    @Override
    public List<MemberLevel> getMemberLevelList() {

        return memberMapper.getMemberLevelList();
    }

    @Override
    public List<Member> getMemberList() {

        List<Member> memberList = memberMapper.getMemberList();

        if(memberList != null) {
```

```
                memberList.forEach( member -> {
                        String memberLevel = member.getMemberLevel();
                        switch (memberLevel) {
                        case "level1":
                                member.setMemberLevelName("관리자");
                                break;
                        case "level2":
                                member.setMemberLevelName("업체 대표 회원");
                                break;
                        case "level3":
                                member.setMemberLevelName("업체 직원 회원");
                                break;
                        case "level4":
                                member.setMemberLevelName("일반 회원");
                                break;
                        case "level5":
                                member.setMemberLevelName("탈퇴 회원");
                                break;
                        default:
                                member.setMemberLevelName("비로그인");
                                break;
                        }
                });
        }

        return memberList;
}
@Override
public String getCompanyCodeByMemberId(String memberId) {
        Company company = companyMapper.getCompanyByMemberId(memberId);
        return company != null ? company.getCompanyCode() : null;
}

@Override
public void IncreasePetByMemberId(String memberId)
{
        memberMapper.IncreasePetByMemberId(memberId);
}

@Override
public void DeclinePetByMemberId(String memberId)
```

```
        {
                memberMapper.DeclinePetByMemberId(memberId);
        }
        // 알림 확인
        @Override
        public void disableNotification(String informId) {
                memberMapper.updateInformStatus(informId);
        }

        @Override
        public String getMemberIdByNameEmail(Member member)
        {
                return memberMapper.getMemberIdByNameEmail(member);
        }

        @Override
        public String getMemberPwByNameEmail(Member member)
        {
                return memberMapper.getMemberPwByNameEmail(member);
        }

        @Override
        public boolean updateMemberPw(Member member)
        {
                int result=memberMapper.updateMemberPw(member);

                boolean updateResult=(result>0)?true:false;
                return updateResult;
        }
}
```

| Package | Ks51team03/member | Controller | MemberController |
|---|---|---|---|

```
package ks51team03.member.controller;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import ks51team03.company.dto.*;
import ks51team03.member.dto.MemberLike;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Controller;
```

```
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
import jakarta.servlet.http.Cookie;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import jakarta.servlet.http.HttpSession;
import ks51team03.company.service.CompanyService;
import ks51team03.member.dto.Member;
import ks51team03.member.dto.MemberLevel;
import ks51team03.member.mapper.MemberMapper;
import ks51team03.member.service.MemberService;
import ks51team03.pet.dto.Pet;
import ks51team03.pet.mapper.PetMapper;
import ks51team03.pet.service.PetService;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.web.multipart.MultipartFile;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;
@Controller
@RequestMapping("/member")
@RequiredArgsConstructor
@Slf4j
public class MemberController {
        private final MemberService memberService;
        private final MemberMapper memberMapper;
        private final CompanyService companyService;
        private final PetService petService;
        private final PetMapper petMapper;

        @PostMapping("/login")
        @ResponseBody
        public boolean login(@RequestParam(value = "memberId") String memberId
                                        , @RequestParam(value = "memberPw") String memberPw
                                        , HttpServletRequest request
                                        , HttpServletResponse response
                                        , HttpSession session){


                //String viewName="/login";
                Map<String, Object> checkMap = memberService.checkMemberInfo(memberId, memberPw);
                boolean isCheck = (boolean) checkMap.get("isCheck");
```

```
                if(isCheck) {
                        Member memberInfo = (Member) checkMap.get("memberInfo");
                        int informCount = memberService.getInformCount(memberInfo.getMemberId());
                        List<ComInformReciPient> getInform =
memberService.getInform(memberInfo.getMemberId());
                        // 1. session
                        //HttpSession requestGetSession = request.getSession();  이렇게 가져올 수도 있다.
                        session.setAttribute("SID", memberInfo.getMemberId());
                        session.setAttribute("SNAME", memberInfo.getMemberName());
                        session.setAttribute("SLEVEL", memberInfo.getMemberLevel());
                        session.setAttribute("SINFOCOUNT", informCount);
                        session.setAttribute("SINFORM", getInform);
                        String ccode =
memberService.getCompanyCodeByMemberId(memberInfo.getMemberId());
                        if(ccode == null) {
                                ccode =
companyService.getCompanyCodeByMemberId(memberInfo.getMemberId());
                        }
                        session.setAttribute("CCODE", ccode);
                        // 2. cookies
                        Cookie cookie = new Cookie("loginId", memberInfo.getMemberId());
                        cookie.setPath("/");
                        cookie.setHttpOnly(true);
                        cookie.setMaxAge(60*60*1);  // 1시간 유지
                        // 생성된 쿠키를 응답객체에 담아 반환
                        response.addCookie(cookie);
                }

        return isCheck;
    }
        @PostMapping("/login/gara")
        public String loginGara(@RequestParam(value = "headerId") String memberId
                        , @RequestParam(value = "headerPw") String memberPw
                        , HttpServletRequest request
                        , HttpServletResponse response
                        , HttpSession session){
                //String viewName="/login";
                Map<String, Object> checkMap = memberService.checkMemberInfo(memberId, memberPw);
                boolean isCheck = (boolean) checkMap.get("isCheck");
                if(isCheck) {
                        Member memberInfo = (Member) checkMap.get("memberInfo");
```

```
                int informCount = memberService.getInformCount(memberInfo.getMemberId());
                List<ComInformReciPient> getInform = memberService.getInform(memberInfo.getMemberId());
                // 1. session
                //HttpSession requestGetSession = request.getSession();  이렇게 가져올 수도 있다.
                session.setAttribute("SID", memberInfo.getMemberId());
                session.setAttribute("SNAME", memberInfo.getMemberName());
                session.setAttribute("SLEVEL", memberInfo.getMemberLevel());
                session.setAttribute("SINFOCOUNT", informCount);
                session.setAttribute("SINFORM", getInform);
                String ccode = memberService.getCompanyCodeByMemberId(memberInfo.getMemberId());
                if(ccode == null) {
                        ccode = companyService.getCompanyCodeByMemberId(memberInfo.getMemberId());
                }
                session.setAttribute("CCODE", ccode);
                // 2. cookies
                Cookie cookie = new Cookie("loginId", memberInfo.getMemberId());
                cookie.setPath("/");
                cookie.setHttpOnly(true);
                cookie.setMaxAge(60*60*1);  // 1시간 유지
                // 생성된 쿠키를 응답객체에 담아 반환
                response.addCookie(cookie);
            }
            return "redirect:/";
        }
        @GetMapping("/logout")
        public String logout(HttpSession session, HttpServletResponse response){
            session.invalidate();
            // 2. cookie
            Cookie cookie = new Cookie("loginId", null);
            cookie.setPath("/");
            cookie.setHttpOnly(true);
            cookie.setMaxAge(0);
            response.addCookie(cookie);
            return "redirect:/";
        }

          @GetMapping("/login")
           public String login(Model model, @RequestParam(value="alertMsg", required = false) String alertMsg
```

```
                                                                    ,@RequestParam(value =
"currentPage", required = false) String currentPage )
        {
        if (alertMsg != null) { model.addAttribute("alertMsg", alertMsg); }

        System.out.println("로그인 버튼 누르기!!");

        return "redirect:"+currentPage;
        }


    @PostMapping("/idCheck")
    @ResponseBody
    public boolean idCheck(@RequestParam(value="memberId") String memberId) {
            boolean isMember = false;

            isMember = memberMapper.idCheck(memberId);

            return isMember;
    }

    @PostMapping("/insertMember")
    public String insertMember(Member member,HttpSession session) {

            String nextPage="redirect:/";
            //반려동물이 있는 경우
            if(member.getMemberPet()>0)
            {
                    //session.setAttribute("SID", member.getMemberId());
                    session.setAttribute("SMEM", member);
                    nextPage="redirect:/member/member_login_insert_pet";
            }
            else
            {
                    memberService.insertMember(member);
            }

            return nextPage;
    }

    @GetMapping("/insertMember")
```

```
	public String insertMember(Model model) {

		List<MemberLevel> memberLevelList = memberService.getMemberLevelList();

		model.addAttribute("title", "회원가입");
		model.addAttribute("levelList", memberLevelList);

		return "member/member_login_insert_mem";
	}

	@PostMapping("/ceoIdCheck")
	@ResponseBody
	public boolean ceoIdCheck(@RequestParam(value = "memberId") String memberId
		, @RequestParam(value = "memberPw") String memberPw
		, HttpServletRequest request
		, HttpServletResponse response){
		Map<String, Object> checkMap = memberService.checkMemberInfo(memberId, memberPw);
		boolean isCheck = (boolean) checkMap.get("isCheck");

	return isCheck;
    }


	@GetMapping("/index")
	public String userMainPage(Model model, @RequestParam(value="alertMsg", required = false) String
alertMsg)
	{
		model.addAttribute("title","PAL");
		if(alertMsg != null) model.addAttribute("alertMsg", alertMsg);

		return "index";
	}

	@GetMapping("/member_mypage_main")
	public String userMyPageMain(Model model)
	{
		model.addAttribute("title","member_mypage_main");

		return "member/member_mypage_main";
	}
```

```
	@GetMapping("/member_mypage_memberinfo")
	public String userMyPageMemberInfo(Model model,HttpSession session, RedirectAttributes redirectAttributes)
	{
		String memberId=(String) session.getAttribute("SID");
		if (memberId == null) {
			redirectAttributes.addFlashAttribute("errorMessage", "로그인을 하는게 좋을거같은데");
			return "redirect:/map/map_main"; // 로그인 페이지로 리다이렉트
		}
		Member memberInfo = memberService.getMemberInfoById(memberId);

		model.addAttribute("memberInfo",memberInfo);

		return "member/member_mypage_memberinfo";
	}
	@PostMapping("/updateMember")
	public String modifyMember(Member member) {
		memberService.updateMember(member);
		return "redirect:/member/member_mypage_memberinfo";
	}

	@GetMapping("/updateMember")
	public String modifyMember(@RequestParam(value="memberId") String memberId
								,Model model) {
		Member memberInfo = memberService.getMemberInfoById(memberId);
		List<MemberLevel> memberLevelList = memberService.getMemberLevelList();

		model.addAttribute("title", "회원수정");
		model.addAttribute("memberInfo", memberInfo);
		model.addAttribute("levelList", memberLevelList);

		return "member/member_login_update_mem";
	}

	@GetMapping("/member_mypage_myQandR")
	public String userMyPageMyQandR(Model model,HttpSession session, RedirectAttributes redirectAttributes)
	{
		String memberId=(String) session.getAttribute("SID");
		if (memberId == null) {
			redirectAttributes.addFlashAttribute("errorMessage", "로그인을 하는게 좋을거같은데");
```

```
                    return "redirect:/map/map_main"; // 로그인 페이지로 리다이렉트
                }
                List<ComQuestion> memberQuestion = memberService.getQuestionById(memberId);
                List<ComReview> comReviews = memberService.getCompanyReview(memberId);
                model.addAttribute("memberQuestion",memberQuestion);
                model.addAttribute("comReviews",comReviews);
                return "member/member_mypage_myQandR";
        }
        @GetMapping("/member_mypage_question_modify")
        public String userQuestionModify(@RequestParam("quesnum") String quesNum, Model model) {
                ComQuestion question = companyService.getCompanyQuestionById(quesNum);
                model.addAttribute("question", question);
                return "member/member_mypage_question_modify";
        }
        @PostMapping("/member_question_modify")
        public String userQuestionModifyAction(@RequestParam("quesNum") String quesNum,ComQuestion
question) {
                String qctenum = getQctenum(question.getQcteNum());
                question.setQuesNum(quesNum);
                question.setQcteNum(qctenum);
                memberService.memberQuestionModify(question);
                return "redirect:/member/member_mypage_myQandR";
        }
        private String getQctenum(String qcteNum) {
                Map<String, String> qctenumMap = new HashMap<>();
                qctenumMap.put("병원이용", "qctc1");
                qctenumMap.put("기타문의", "qctc11");
                qctenumMap.put("진료관련", "qctc2");
                qctenumMap.put("병원행정", "qctc3");
                qctenumMap.put("약국행정", "qctc4");
                qctenumMap.put("제품관련", "qctc5");
                qctenumMap.put("처방문의", "qctc9");
                qctenumMap.put("장례문의", "qctc7");
                qctenumMap.put("비용문의", "qctc8");
                qctenumMap.put("일정문의", "qctc9");
                return qctenumMap.getOrDefault(qcteNum, "기타문의");
        }
        @GetMapping("/member_mypage_question_delete")
        public String userQuestionDelete(@RequestParam("quesnum") String quesNum) {
                ComQuestion question = new ComQuestion();
                question.setQuesNum(quesNum);
```

```
                memberService.memberQuestionDelete(question);
                return "redirect:/member/member_mypage_myQandR";
        }
        @GetMapping("/member_mypage_review_modify")
        public String userReviewModify(@RequestParam("revcode") String revCode, Model model,
HttpSession session) {
                ComReview review = memberService.getCompanyReviewByRevCode(revCode);
                model.addAttribute("review", review);
                return "member/member_mypage_review_modify";
        }
        @PostMapping("/member_review_modify")
        public String userReviewModifyAction(@RequestParam("revCode") String revCode, ComReview
review,
                                                                        @RequestParam(value =
"deleteImage", defaultValue = "false") boolean deleteImage,
                                                                        @RequestParam(value =
"newImage", defaultValue = "false") boolean newImage,

@RequestParam("revImgFile") MultipartFile revImgFile) {
                review.setRevCode(revCode);
                memberService.memberReviewModify(review, deleteImage, newImage, revImgFile);
                return "redirect:/member/member_mypage_myQandR";
        }
        // 회원 리뷰 삭제
        @PostMapping("/member_mypage_review_delete")
        public String userReviewDeleteAction(@RequestParam("revCode") String revCode, ComReview
review) {
                review.setRevCode(revCode);
                memberService.memberReviewDelete(review);
                return "redirect:/member/member_mypage_myQandR";
        }
        @GetMapping("/member_mypage_list_pet")
        public String userMyPageListPet(Model model,HttpSession session)
        {
                String memberId=(String) session.getAttribute("SID");
                List<Pet> petInfoList = petService.getPetInfoByMemberId(memberId);

                model.addAttribute("petInfoList",petInfoList);

                return "member/member_mypage_list_pet";
        }
```

```
@GetMapping("/member_login_terms_mem")
public String userTermsPageMem(Model model)
{
	model.addAttribute("title","PAL");

	return "member/member_login_terms_mem";
}

@GetMapping("/member_login_terms_com")
public String userTermsPageCom(Model model)
{
	model.addAttribute("title","PAL");

	return "member/member_login_terms_com";
}

@GetMapping("/member_login_insert_mem")
public String userInsertPageMem(Model model)
{
	model.addAttribute("title","PAL");

	return "member/member_login_insert_mem";
}

@GetMapping("/member_login_insert_com")
public String userInsertPageCom(Model model)
{
	model.addAttribute("title","PAL");

	return "member/member_login_insert_com";
}

@GetMapping("/member_login_insert_pet")
public String userInsertPagePet(Model model)
{
	model.addAttribute("title","PAL");

	return "member/member_login_insert_pet";
}
@PutMapping("/disable/{informId}")
public ResponseEntity<Void> disableNotification(@PathVariable String informId, HttpSession
```

```
session) {
                try {
                        String memberId = (String) session.getAttribute("SID");
                        memberService.disableNotification(informId);
                        List<ComInformReciPient> getInform = memberService.getInform(memberId);
                        int informCount = memberService.getInformCount(memberId);
                        session.setAttribute("SINFORM", getInform);
                        session.setAttribute("SINFOCOUNT", informCount);
                        return ResponseEntity.ok().build();
                } catch (Exception e) {
                        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).build();
                }
        }
        @GetMapping("/member_main2")
        public String userTestPage(Model model)
        {
                model.addAttribute("title","PAL");

                return "member/member_main2";
        }

        @GetMapping("/member_login_find_id")
        public String userFindMemberIdPage(Model model)
        {
                return "member/member_login_find_id";
        }
        @GetMapping("/member_login_find_pw")
        public String userFindMemberPwPage(Model model)
        {
                return "member/member_login_find_pw";
        }
        @ResponseBody
        @PostMapping("/updatePw")
        public ResponseEntity<Boolean> updatePw(String memberId,String memberPw) {
                Member member=new Member();
                member.setMemberId(memberId);
                member.setMemberPw(memberPw);

                boolean isUpdated = memberService.updateMemberPw(member);

        if (isUpdated)
```

```
        {
            return ResponseEntity.ok(true);
        } else {
            return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).body(false);
        }
        }
        @PostMapping("/member_like")
        @ResponseBody
        public String userAddLike(@RequestBody Map<String, String> requestData, HttpSession session) {
                String memberId = (String) session.getAttribute("SID");
                String cCode = requestData.get("cCode");
                MemberLike memberLike = memberService.memberLikeCheckFirst(cCode, memberId);
                if(memberLike != null){
                        if(memberLike.getLkState().equals("1")){
                                memberService.updateMemberLikeStateOn(memberLike.getLkCode());
                        }
                }
                else{
                        memberService.memberAddLike(memberId, cCode);
                }
                return "redirect:/map/map_company_info";
        }
        @GetMapping("/member_mypage_like_setting")
        public String LikeList(Model model, HttpSession session, RedirectAttributes redirectAttributes){
                String memberId = (String) session.getAttribute("SID");
                // 세션에 아이디가 없으면 로그인 페이지로 리다이렉트
                if (memberId == null) {
                        redirectAttributes.addFlashAttribute("errorMessage", "로그인을 하는게 좋을거같은데");
                        return "redirect:/map/map_main"; // 로그인 페이지로 리다이렉트
                }
                List<MemberLike> comLikes = memberService.memberGetLikeCompany(memberId);
                model.addAttribute("comLikes",comLikes);
                return "member/member_mypage_like_setting";
        }
        @PostMapping("/updateAlarm")
        @ResponseBody
        public String updateMemberAlarm(@RequestParam("lkCode") String lkCode,

@RequestParam("currentAlarmState") int lkAlarm) {
                memberService.updateMemberLikeAlarm(lkCode,lkAlarm);
                return "redirect:/map/map_company_info";
```

```
        }
        // 마이 페이지에서 구독 취소
        @PostMapping("/updateLikeState")
        @ResponseBody
        public String updateMemberLikeState(@RequestParam("lkCode") String lkCode) {
                memberService.updateMemberLikeState(lkCode);
                return "redirect:/map/map_company_info";
        }
        // 업체 페이지에서 구독 취소
        @PostMapping("/updateLikeStateForComInfo")
        @ResponseBody
        public String updateLikeState(@RequestBody Map<String, String> request) {
                String lkCode = request.get("lkCode");
                System.out.println("lkCode: " + lkCode);
                memberService.updateMemberLikeState(lkCode);
                return "redirect:/map/map_company_info";
        }
}
```

| Package | Ks51team03/board | Board | BoardController |
|---|---|---|---|

```
package ks51team03.board.controller;
import java.nio.charset.StandardCharsets;
import java.util.ArrayList;
import java.util.List;
import java.util.Map;
import java.util.UUID;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.multipart.MultipartFile;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;
import org.springframework.web.util.UriUtils;
import ks51team03.board.dto.NBoardImg;
import ks51team03.board.dto.NBoardSearch;
import ks51team03.board.dto.NoticeBoard;
import ks51team03.board.mapper.BoardMapper;
```

```
import ks51team03.board.service.BoardService;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
@Controller
@RequestMapping("/board")
@RequiredArgsConstructor
@Slf4j
public class BoardController {
        private final BoardService boardService;
        private final BoardMapper boardmapper;

        @GetMapping("/board_list_normal")
        public String getNoticeBoardList(Model model,
                        @RequestParam(name = "currentPage", defaultValue = "1") int currentPage,
                        @RequestParam(name = "searchType", required = false, defaultValue = "nb_title")
String searchType,
                        @RequestParam(name = "searchKeyword", required = false, defaultValue = "")
String searchKeyword,
                        @RequestParam(name = "boardCateValue") String boardCateValue) {
                String boardTitle = boardCateValue; // 기본값 설정
                // 서비스 메서드 호출
                Map<String, Object> resultMap = boardService.getNoticeBoardList(boardCateValue,
currentPage, searchType,
                                searchKeyword);
                String boardCode = boardService.getBoardCodeByBCTValue(boardCateValue);
                String boardInfo = boardService.getBoardInfoByBCTValue(boardCateValue);
                // 결과를 모델에 담아서 뷰로 전달
                model.addAttribute("NoticeBoardList", resultMap.get("NoticeBoardList"));
                model.addAttribute("startPageNum", resultMap.get("startPageNum"));
                model.addAttribute("endPageNum", resultMap.get("endPageNum"));
                model.addAttribute("lastPage", resultMap.get("lastPage"));
                model.addAttribute("currentPage", currentPage);
                model.addAttribute("boardTitle", boardTitle); // boardTitle 변수를 모델에 추가
                model.addAttribute("boardCode", boardCode); // boardTitle 변수를 모델에 추가
                model.addAttribute("boardInfo", boardInfo); // boardTitle 변수를 모델에 추가
                // 검색 키워드 설정
                List<NBoardSearch> searchCate = new ArrayList<>();
                NBoardSearch search1 = new NBoardSearch();
                search1.setSearchKey("memberId");
                search1.setSearchText("회원아이디");
                NBoardSearch search2 = new NBoardSearch();
```

```
                search2.setSearchKey("nboardTitle");
                search2.setSearchText("제목");
                NBoardSearch search3 = new NBoardSearch();
                search3.setSearchKey("nboardContent");
                search3.setSearchText("내용");
                searchCate.add(search1);
                searchCate.add(search2);
                searchCate.add(search3);
                model.addAttribute("searchCate", searchCate);
                return "board/board_list_normal"; // 뷰 이름 리턴
        }
        @GetMapping("/board_list_normal_search")
        public String searchNoticeBoardList(Model model,
                        @RequestParam(name = "currentPage", required = false, defaultValue = "1") int
currentPage,
                        @RequestParam(name = "searchType", required = false) String searchType,
                        @RequestParam(name = "searchKeyword", required = false) String searchKeyword,
                        @RequestParam(name = "boardCateValue") String boardCateValue) {
                String boardTitle = boardCateValue; // 기본값 설정
                // 서비스 메서드 호출
                Map<String, Object> resultMap = boardService.getNoticeBoardList(boardCateValue,
currentPage, searchType,
                                searchKeyword);
                // 결과를 모델에 담아서 뷰로 전달
                model.addAttribute("NoticeBoardList", resultMap.get("NoticeBoardList"));
                model.addAttribute("startPageNum", resultMap.get("startPageNum"));
                model.addAttribute("endPageNum", resultMap.get("endPageNum"));
                model.addAttribute("lastPage", resultMap.get("lastPage"));
                model.addAttribute("currentPage", currentPage);
                model.addAttribute("boardTitle", boardTitle); // boardTitle 변수를 모델에 추가
                // 검색 키워드 설정
                List<NBoardSearch> searchCate = new ArrayList<>();
                NBoardSearch search1 = new NBoardSearch();
                search1.setSearchKey("memberId");
                search1.setSearchText("회원아이디");
                NBoardSearch search2 = new NBoardSearch();
                search2.setSearchKey("nboardTitle");
                search2.setSearchText("제목");
                NBoardSearch search3 = new NBoardSearch();
                search3.setSearchKey("nboardContent");
                search3.setSearchText("내용");
```

```
                searchCate.add(search1);
                searchCate.add(search2);
                searchCate.add(search3);
                model.addAttribute("searchCate", searchCate);
                return "board/board_list_normal"; // 뷰 이름 리턴
        }
        /* 자유게시글 작성 */
        @PostMapping("/board_write_normal")
        public String boardWriteNormalPage(NoticeBoard nboard, RedirectAttributes rttr) {
                log.info("board_write_normal");
                String boardCateCodeName = nboard.getBoardCateCode();
                String boardCateCode = boardService.getBoardCateCodeByBCTValue(nboard);
                String boardCode = boardService.getBoardCodeByNBoard(nboard);
                nboard.setBoardCateCode(boardCateCode);
                nboard.setBoardCode(boardCode);
                boardService.insertNBoard(nboard);
                String encodedBoardCateValue = UriUtils.encodeQueryParam(boardCateCodeName,
StandardCharsets.UTF_8);
                return "redirect:/board/board_list_normal?currentPage=1&boardCateValue=" +
encodedBoardCateValue;
        }
        @GetMapping("/board_write_normal")
        public String boardWriteNormalPage(Model model, @RequestParam(name = "boardCateValue")
String boardCateValue) {
                String boardTitle = boardCateValue; // 기본값 설정
                String boardInfo = boardService.getBoardInfoByBCTValue(boardCateValue);
                model.addAttribute("boardTitle", boardTitle); // boardTitle 변수를 모델에 추가
                model.addAttribute("boardInfo", boardInfo); // boardTitle 변수를 모델에 추가

                return "board/board_write_normal";
        }
        /* 자유 게시글 열람 */
        @GetMapping("/board_view_normal")
        public String boardViewNormalPage(@RequestParam(value = "nboardCode") String
nboardCode,Model model) {
                log.info("board_view_normal");
                // 게시글 코드를 숫자만 분리해서 넘기기
                // int nbcode=Integer.parseInt(nboardCode.replaceAll("[^₩₩d]", ""));
                String boardTitle = boardService.getBCTValueByNBCode(nboardCode); // 기본값 설정
                String boardInfo = boardService.getBoardInfoByBCTValue(boardTitle);
                model.addAttribute("boardTitle", boardTitle); // boardTitle 변수를 모델에 추가
                model.addAttribute("boardInfo", boardInfo); // boardTitle 변수를 모델에 추가
```

```
                model.addAttribute("noticeBoard", boardService.getNBoardByNBCode(nboardCode));
                // 조회수 증가
                boardService.increaseViewByNBCode(nboardCode);
                return "board/board_view_normal";
        }
        /* 자유 게시글 추천 */
        @PostMapping("/recommend")
        @ResponseBody
        public ResponseEntity<String> recommendBoard(@RequestParam(value = "nboardCode") String
nboardCode) {
                boardService.increaseRecByNBCode(nboardCode);
                // 클라이언트에게 성공 메시지 반환
                return ResponseEntity.ok("게시물을 추천하였습니다.");
        }
        /* 자유 게시글 수정 전 읽어오기 */
        @GetMapping("/board_edit_normal")
        public String boardEditNormalPage(@RequestParam(value = "nboardCode") String nboardCode,
                        @RequestParam(name = "boardCateValue") String boardCateValue, Model model) {
                log.info("board_edit_normal");
                String boardTitle = boardCateValue; // 기본값 설정
                String boardInfo = boardService.getBoardInfoByBCTValue(boardCateValue);
                model.addAttribute("boardTitle", boardTitle); // boardTitle 변수를 모델에 추가
                model.addAttribute("boardInfo", boardInfo); // boardTitle 변수를 모델에 추가
                model.addAttribute("noticeBoard", boardService.getNBoardByNBCode(nboardCode));
                return "board/board_edit_normal";
        }
        /* 자유 게시글 수정 */
        @PostMapping("/board_edit_normal")
        public String boardEditNormalPage(NoticeBoard nboard, RedirectAttributes rttr) {
                log.info("board_edit_normal");
                String BoardCateValueName = boardService.getBCTValueNameByBCTCode(nboard);
                boardService.updateNBoard(nboard);
                // 직접 URL 인코딩을 수행하여 파라미터를 추가
                /*
                 * String encodedBoardCateValue = UriUtils.encodeQueryParam("자유 게시글",
                 * StandardCharsets.UTF_8);
                 */
                String encodedBoardCateValue = UriUtils.encodeQueryParam(BoardCateValueName,
StandardCharsets.UTF_8);
                return "redirect:/board/board_list_normal?currentPage=1&boardCateValue=" +
encodedBoardCateValue;
```

```
    }


    @GetMapping("/board_list_gallery")
    public String getGalleryBoardList(Model model,
                                      @RequestParam(name = "currentPage", defaultValue = "1") int currentPage,
                                      @RequestParam(name = "searchType", required = false, defaultValue = "nb_title") String searchType,
                                      @RequestParam(name = "searchKeyword", required = false, defaultValue = "") String searchKeyword,
                                      @RequestParam(name = "boardCateValue") String boardCateValue) {
        String boardTitle = boardCateValue; // 기본값 설정
        // 서비스 메서드 호출
        Map<String, Object> resultMap = boardService.getNoticeBoardList(boardCateValue, currentPage, searchType, searchKeyword);
        String boardCode = boardService.getBoardCodeByBCTValue(boardCateValue);
        String boardInfo = boardService.getBoardInfoByBCTValue(boardCateValue);


        // 결과를 모델에 담아서 뷰로 전달
        model.addAttribute("NoticeBoardList", resultMap.get("NoticeBoardList"));
        model.addAttribute("startPageNum", resultMap.get("startPageNum"));
        model.addAttribute("endPageNum", resultMap.get("endPageNum"));
        model.addAttribute("lastPage", resultMap.get("lastPage"));
        model.addAttribute("currentPage", currentPage);
        model.addAttribute("boardTitle", boardTitle); // boardTitle 변수를 모델에 추가
        model.addAttribute("boardCode", boardCode); // boardCode 변수를 모델에 추가
        model.addAttribute("boardInfo", boardInfo); // boardInfo 변수를 모델에 추가
        // 검색 키워드 설정
        List<NBoardSearch> searchCate = new ArrayList<>();
        NBoardSearch search1 = new NBoardSearch();
        search1.setSearchKey("memberId");
        search1.setSearchText("회원아이디");
        NBoardSearch search2 = new NBoardSearch();
        search2.setSearchKey("nboardTitle");
        search2.setSearchText("제목");
        NBoardSearch search3 = new NBoardSearch();
        search3.setSearchKey("nboardContent");
        search3.setSearchText("내용");
```

```
        searchCate.add(search1);
        searchCate.add(search2);
        searchCate.add(search3);
        model.addAttribute("searchCate", searchCate);
        return "board/board_list_gallery"; // 뷰 이름 리턴
    }



    /*
    @GetMapping("/board_list_gallery")
    public String getGalleryBoardList(Model model,
                    @RequestParam(name = "currentPage", defaultValue = "1") int currentPage,
                    @RequestParam(name = "searchType", required = false, defaultValue = "nb_title")
String searchType,
                    @RequestParam(name = "searchKeyword", required = false, defaultValue = "")
String searchKeyword,
                    @RequestParam(name = "boardCateValue") String boardCateValue) {
            String boardTitle = boardCateValue; // 기본값 설정
            // 서비스 메서드 호출
            Map<String, Object> resultMap = boardService.getNoticeBoardList(boardCateValue,
currentPage, searchType,
                            searchKeyword);
            String boardCode = boardService.getBoardCodeByBCTValue(boardCateValue);
            String boardInfo = boardService.getBoardInfoByBCTValue(boardCateValue);
            // 결과를 모델에 담아서 뷰로 전달
            model.addAttribute("NoticeBoardList", resultMap.get("NoticeBoardList"));
            model.addAttribute("startPageNum", resultMap.get("startPageNum"));
            model.addAttribute("endPageNum", resultMap.get("endPageNum"));
            model.addAttribute("lastPage", resultMap.get("lastPage"));
            model.addAttribute("currentPage", currentPage);
            model.addAttribute("boardTitle", boardTitle); // boardTitle 변수를 모델에 추가
            model.addAttribute("boardCode", boardCode); // boardTitle 변수를 모델에 추가
            model.addAttribute("boardInfo", boardInfo); // boardTitle 변수를 모델에 추가
            // 검색 키워드 설정
            List<NBoardSearch> searchCate = new ArrayList<>();
            NBoardSearch search1 = new NBoardSearch();
            search1.setSearchKey("memberId");
            search1.setSearchText("회원아이디");
            NBoardSearch search2 = new NBoardSearch();
            search2.setSearchKey("nboardTitle");
```

```
                search2.setSearchText("제목");
                NBoardSearch search3 = new NBoardSearch();
                search3.setSearchKey("nboardContent");
                search3.setSearchText("내용");
                searchCate.add(search1);
                searchCate.add(search2);
                searchCate.add(search3);
                model.addAttribute("searchCate", searchCate);
                return "board/board_list_gallery"; // 뷰 이름 리턴
        }*/

        /* 갤러리게시글 작성 */
        @GetMapping("/board_write_gallery")
        public String boardWriteGalleryPage(Model model,@RequestParam(name = "boardCateValue") String
boardCateValue)
        {
                String boardTitle = boardCateValue; // 기본값 설정
                String boardInfo = boardService.getBoardInfoByBCTValue(boardCateValue);
                String nboardCode="nb" + String.valueOf(boardmapper.getNBoardCode()+1);

                model.addAttribute("boardTitle", boardTitle); // boardTitle 변수를 모델에 추가
                model.addAttribute("boardInfo", boardInfo); // boardTitle 변수를 모델에 추가
                model.addAttribute("nboardCode", nboardCode); // nboardCode 변수를 모델에 추가

                return "board/board_write_gallery";
        }
        @PostMapping("/board_write_gallery")
        public String boardWriteGalleryPage(NoticeBoard nboard, RedirectAttributes rttr) {
            log.info("board_write_gallery");
            String boardCateCodeName = nboard.getBoardCateCode();
            String boardCateCode = boardService.getBoardCateCodeByBCTValue(nboard);
            String boardCode = boardService.getBoardCodeByNBoard(nboard);

            nboard.setBoardCateCode(boardCateCode);
            nboard.setBoardCode(boardCode);
            String nbcode = "nb" + String.valueOf(boardmapper.getNBoardCode());

            nboard.setNboardCode(nbcode);

            String boardImgUrl=boardService.getNBoardImgByNBCode(nbcode);
            nboard.setNboardImg(boardImgUrl);
```

```
            System.out.println("boardImgUrl: " + boardImgUrl);
            System.out.println("contents:"+nboard.getNboardContent());
            // 대표 이미지가 내용에서 지워질 때까지 반복
            while (!nboard.getNboardContent().contains(boardImgUrl)) {
                boardService.deleteFilesAndNBoardImg(boardImgUrl);
                System.out.println("Deleted: " + boardImgUrl);
                // 이미지가 삭제된 후 업데이트된 내용을 다시 가져옵니다.
                boardImgUrl = boardService.getNBoardImgByNBCode(nbcode);
                nboard.setNboardImg(boardImgUrl);
                System.out.println("New boardImgUrl: " + boardImgUrl);
                // 만약 더 이상 삭제할 이미지가 없다면 루프를 종료합니다.
                if (boardImgUrl == null || boardImgUrl.isEmpty()) {
                        System.out.println("boardImgUrl.isEmpty()");
                    break;
                }
            }

            boardService.updateNBoard(nboard);

                    String encodedBoardCateValue = UriUtils.encodeQueryParam(boardCateCodeName,
StandardCharsets.UTF_8);

            // 이미지 업로드 로직은 필요하지 않음
            // boardService.upLoadImgByNBCode(nboardimg, nbcode);
                    return "redirect:/board/board_list_gallery?currentPage=1&boardCateValue=" +
encodedBoardCateValue;
        }
        /* 갤러리 게시글 열람 */
        @GetMapping("/board_view_gallery")
        public String boardGalleryNormalPage(@RequestParam(value = "nboardCode") String
nboardCode,Model model) {
            log.info("board_view_gallery");
            // 게시글 코드를 숫자만 분리해서 넘기기
            // int nbcode=Integer.parseInt(nboardCode.replaceAll("[^₩₩d]", ""));
            String boardTitle = boardService.getBCTValueByNBCode(nboardCode); // 기본값 설정
            String boardInfo = boardService.getBoardInfoByBCTValue(boardTitle);
            model.addAttribute("boardTitle", boardTitle); // boardTitle 변수를 모델에 추가
            model.addAttribute("boardInfo", boardInfo); // boardTitle 변수를 모델에 추가
            model.addAttribute("noticeBoard", boardService.getNBoardByNBCode(nboardCode));
            // 조회수 증가
            boardService.increaseViewByNBCode(nboardCode);
```

```
        return "board/board_view_gallery";
    }

    /* 자유 게시글 수정 전 읽어오기 */
    @GetMapping("/board_edit_gallery")
    public String boardEditGalleryPage(@RequestParam(value = "nboardCode") String nboardCode,
                    @RequestParam(name = "boardCateValue") String boardCateValue, Model model) {
        log.info("board_edit_normal");
        String boardTitle = boardCateValue; // 기본값 설정
        String boardInfo = boardService.getBoardInfoByBCTValue(boardCateValue);
        model.addAttribute("boardTitle", boardTitle); // boardTitle 변수를 모델에 추가
        model.addAttribute("boardInfo", boardInfo); // boardTitle 변수를 모델에 추가
        model.addAttribute("noticeBoard", boardService.getNBoardByNBCode(nboardCode));
        return "board/board_edit_gallery";
    }
    /* 자유 게시글 수정 */
    @PostMapping("/board_edit_gallery")
    public String boardEditGalleryPage(NoticeBoard nboard, RedirectAttributes rttr) {
        log.info("board_edit_normal");
        String BoardCateValueName = boardService.getBCTValueNameByBCTCode(nboard);
        String nbcode =nboard.getNboardCode();

        String boardImgUrl=boardService.getNBoardImgByNBCode(nbcode);
    nboard.setNboardImg(boardImgUrl);
    System.out.println("boardImgUrl: " + boardImgUrl);
    System.out.println("contents:"+nboard.getNboardContent());
    // 대표 이미지가 내용에서 지워질 때까지 반복
    if (boardImgUrl != null && nboard.getNboardContent() != null) {
            while (!nboard.getNboardContent().contains(boardImgUrl)) {
                boardService.deleteFilesAndNBoardImg(boardImgUrl);
                System.out.println("Deleted: " + boardImgUrl);

                // 이미지가 삭제된 후 업데이트된 내용을 다시 가져옵니다.
                boardImgUrl = boardService.getNBoardImgByNBCode(nbcode);
                nboard.setNboardImg(boardImgUrl);
                System.out.println("New boardImgUrl: " + boardImgUrl);

                // 만약 더 이상 삭제할 이미지가 없다면 루프를 종료합니다.
                if (boardImgUrl == null || boardImgUrl.isEmpty()) {
                        System.out.println("boardImgUrl.isEmpty()");
                    break;
```

```
                        }
                    }
                }
                boardService.updateNBoard(nboard);

                String encodedBoardCateValue = UriUtils.encodeQueryParam(BoardCateValueName,
StandardCharsets.UTF_8);
                return "redirect:/board/board_list_gallery?currentPage=1&boardCateValue=" +
encodedBoardCateValue;
            }

            /*
             * 자유 게시글 검색
             *
             * @PostMapping("/board_search_list")
             *
             * @ResponseBody public List<NoticeBoard> getSearchList(@RequestBody
             * NBoardSearch nbsearch){
             *
             * log.info("nbsearch: {}", nbsearch);
             *
             * return boardService.getBoardSearchList(nbsearch); }
             */
}
```

| Package | Ks51team03/board | Controller | BoardImgController |
|---|---|---|---|

```
package ks51team03.board.controller;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.io.PrintWriter;
import java.util.UUID;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.multipart.MultipartFile;
import jakarta.servlet.http.HttpServletRequest;
```

```
import jakarta.servlet.http.HttpServletResponse;
import ks51team03.board.dto.NBoardImg;
import ks51team03.board.dto.NoticeBoard;
import ks51team03.board.mapper.BoardMapper;
import ks51team03.board.service.BoardService;
import ks51team03.files.dto.FileRequest;
@RestController
public class BoardImgController {
	@Autowired
	private BoardMapper boardMapper;
	@Autowired
	private BoardService boardService;
	@PostMapping("/board/imageUpload")
    public void imageUpload(HttpServletRequest request,
                            HttpServletResponse response,
                            @RequestParam("upload") MultipartFile upload,
                            @RequestParam(value = "nboardCode", required = false, defaultValue = "")
String nboardCode,String mode) throws Exception {
                System.out.println("================Received nboardCode: " + nboardCode);

                UUID uid = UUID.randomUUID();
        OutputStream out = null;
        PrintWriter printWriter = null;
        response.setCharacterEncoding("utf-8");
        response.setContentType("application/json");
        try {
            String fileName = upload.getOriginalFilename();
            byte[] bytes = upload.getBytes();
            String nbcode = nboardCode;
            System.out.println("nboardCode:"+nbcode);
            if (nboardCode.equals(""))
            {
                nbcode="nb" + String.valueOf(boardMapper.getNBoardCode()+1);
                System.out.println("nboardCode is null:"+nbcode);
            }

            String path = "/home/ks51team03/attachment" + File.separator + nbcode;
            String ckUploadPath = path + File.separator + uid + "_" + fileName;
            File folder = new File(path);
            if (!folder.exists()) {
                folder.mkdirs();
```

```
            }
            out = new FileOutputStream(new File(ckUploadPath));
            out.write(bytes);
            out.flush();
            // 파일 URL을 상대 경로로 변경
            String fileUrl = "/attachments/" + nbcode + "/" + uid + "_" + fileName;
            System.out.println("File URL: " + fileUrl); // 업로드된 파일의 URL 출력
            String areadyImg=boardService.getNBoardImgByNBCode(nbcode);
            System.out.println("areadyImg:"+areadyImg);
            if ("".equals(nboardCode)&&areadyImg==null)
            {
                NoticeBoard noticeBoard = new NoticeBoard();
                noticeBoard.setNboardCode(nbcode);
                noticeBoard.setMemberId("id66");
                noticeBoard.setBoardCode("b3");
                noticeBoard.setBoardCateCode("bct3");
                noticeBoard.setNboardTitle("temp");
                noticeBoard.setNboardContent("temp");
                noticeBoard.setNboardRegistDate("temp");
                noticeBoard.setNboardImg(fileUrl);
                // NoticeBoard 객체를 데이터베이스에 삽입
                boardService.insertNBoard(noticeBoard);

                nbcode = "nb" + String.valueOf(boardMapper.getNBoardCode());
            }


            // nboard_img 테이블에 이미지 정보 삽입
            NBoardImg nBoardImg = new NBoardImg();

            nBoardImg.setNbCode(nbcode);
            nBoardImg.setFilePath(fileUrl);
            nBoardImg.setNBoardImgFile(upload);
            FileRequest fileRequest = boardService.upLoadImgByNBCode(nBoardImg, nbcode);
            // JSON 응답을 올바르게 생성
            String jsonResponse = String.format("{₩"uploaded₩": 1, ₩"fileName₩": ₩"%s₩", ₩"url₩":
₩"%s₩"}", fileName, fileRequest.getFilePath());
            printWriter = response.getWriter();
            printWriter.print(jsonResponse); // println 대신 print 사용
            printWriter.flush();
        } catch (IOException e) {
```

```
            e.printStackTrace();
            response.setStatus(HttpServletResponse.SC_INTERNAL_SERVER_ERROR);
            response.setContentType("application/json");
            String errorResponse = "{₩"uploaded₩": 0, ₩"error₩": {₩"message₩": ₩"" + e.getMessage() + "₩"}}";
            printWriter = response.getWriter();
            printWriter.print(errorResponse);
            printWriter.flush();
        } finally {
            if (out != null) out.close();
            if (printWriter != null) printWriter.close();
        }
    }

    /*
    @PostMapping("/board/imageUpload")
    public void imageUpload(HttpServletRequest request, HttpServletResponse response,
                            @RequestParam("upload") MultipartFile upload,
                            @RequestParam(value = "nboardCode", required = false, defaultValue = "") String nboardCode)
                            throws Exception {
                System.out.println("Received nboardCode: " + nboardCode); // 로그 출력
                UUID uid = UUID.randomUUID();
                OutputStream out = null;
                PrintWriter printWriter = null;
                response.setCharacterEncoding("utf-8");
                response.setContentType("application/json");
                try {
                        String fileName = upload.getOriginalFilename();
                        byte[] bytes = upload.getBytes();
                        if (nboardCode.isEmpty()) {
                                nboardCode = "nb" + String.valueOf(boardMapper.getNBoardCode() + 1);
                        }
                        String path = "/home/ks51team03/attachment" + File.separator + nboardCode;
                        String ckUploadPath = path + File.separator + uid + "_" + fileName;
                        File folder = new File(path);
                        if (!folder.exists()) {
                                folder.mkdirs();
                        }
                        out = new FileOutputStream(new File(ckUploadPath));
                        out.write(bytes);
```

```
                        out.flush();
// 파일 URL을 상대 경로로 변경
                        String fileUrl = "/attachments/" + nboardCode + "/" + uid + "_" + fileName;
                        String alreadyImg = boardService.getNBoardImgByNBCode(nboardCode);
                        if (alreadyImg == null) {
                                NoticeBoard noticeBoard = new NoticeBoard();
                                noticeBoard.setNboardCode(nboardCode);
                                noticeBoard.setMemberId("id66");
                                noticeBoard.setBoardCode("b3");
                                noticeBoard.setBoardCateCode("bct3");
                                noticeBoard.setNboardTitle("temp");
                                noticeBoard.setNboardContent("temp");
                                noticeBoard.setNboardRegistDate("temp");
                                noticeBoard.setNboardImg(fileUrl);
                                boardService.insertNBoard(noticeBoard);
                        }
// nboard_img 테이블에 이미지 정보 삽입
                        NBoardImg nBoardImg = new NBoardImg();
                        nBoardImg.setNbCode(nboardCode);
                        nBoardImg.setFilePath(fileUrl);
                        nBoardImg.setNBoardImgFile(upload);
                        boardService.upLoadImgByNBCode(nBoardImg, nboardCode);
// JSON 응답을 올바르게 생성
                        String jsonResponse = String.format("{₩"uploaded₩": 1, ₩"fileName₩": ₩"%s₩",
₩"url₩": ₩"%s₩"}", fileName,
                                        fileUrl);
                        printWriter = response.getWriter();
                        printWriter.print(jsonResponse);
                        printWriter.flush();
                } catch (IOException e) {
                        e.printStackTrace();
                        response.setStatus(HttpServletResponse.SC_INTERNAL_SERVER_ERROR);
                        response.setContentType("application/json");
                        String errorResponse = "{₩"uploaded₩": 0, ₩"error₩": {₩"message₩": ₩"" +
e.getMessage() + "₩"}}";
                        printWriter = response.getWriter();
                        printWriter.print(errorResponse);
                        printWriter.flush();
                } finally {
                        if (out != null)
                                out.close();
```

```
                        if (printWriter != null)
                                printWriter.close();
                }
        }
*/

        @GetMapping("/attachments/**")
        public void getImage(HttpServletRequest request, HttpServletResponse response) {
                System.out.println("test");
                String filePath = request.getRequestURI().replace("/attachments", "");
                File file = new File("/home/ks51team03/attachment" + filePath);
                if (file.exists()) {
                        response.setContentType("image/jpeg");
                        try (FileInputStream fis = new FileInputStream(file); OutputStream out =
response.getOutputStream()) {
                                byte[] buffer = new byte[1024];
                                int bytesRead;
                                while ((bytesRead = fis.read(buffer)) != -1) {
                                        out.write(buffer, 0, bytesRead);
                                }
                                out.flush();
                        } catch (IOException e) {
                                e.printStackTrace();
                        }
                } else {
                        response.setStatus(HttpServletResponse.SC_NOT_FOUND);
                }
        }
}
```

| Package | Ks51team03/board | Service | BoardService |
|---|---|---|---|

```
package ks51team03.board.service;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import ks51team03.board.dto.NBoardImg;
import ks51team03.board.dto.NBoardSearch;
import ks51team03.board.dto.NoticeBoard;
import ks51team03.board.mapper.BoardMapper;
```

```
import ks51team03.company.dto.CompanyImg;
import ks51team03.files.dto.FileRequest;
import ks51team03.files.mapper.FileMapper;
import ks51team03.files.util.FileUtils;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
@Service("BoardService")
@Transactional
@RequiredArgsConstructor
@Slf4j
public class BoardService {
        private final BoardMapper boardMapper;
        private final FileUtils fileUtils;
    private final FileMapper fileMapper;
        public Map<String, Object> getNoticeBoardList(String boardCateValue, int currentPage, String searchKey,
                        String searchValue) {
                // 보여줄 행의 갯수
                int rowPerPage = 8;
                // 첫번째 인수값
                int startRow = (currentPage - 1) * rowPerPage;
                // 시작페이지 설정 초기화
                int startPageNum = 1;
                // 마지막페이지 설정 초기화
                int endPageNum = 10;
                log.info("getNoticeBoardList: {}",
                                boardMapper.getNoticeBoardList(boardCateValue, startRow, rowPerPage, searchKey, searchValue));
                List<Map<String, Object>> NoticeBoardList = boardMapper.getNoticeBoardList(boardCateValue, startRow, rowPerPage,
                                searchKey, searchValue);
                // 전체 행의 갯수 조회
                double cnt = boardMapper.getNoticeBoardListCnt(boardCateValue, searchKey, searchValue);
                // 마지막 페이지
                int lastPage = (int) Math.ceil(cnt / rowPerPage);
                // 마지막페이지 보다 작을 경우 마지막페이지로 설정
                endPageNum = lastPage < 10 ? lastPage : endPageNum;
                // 동적 페이지설정
                if (currentPage > 6 && endPageNum > 9) {
                        startPageNum = currentPage - 5;
                        endPageNum = currentPage + 4;
```

```
                // 마지막페이지번호가 마지막페이지수보다 클 경우에 페이지번호를 고정
                if (endPageNum >= lastPage) {
                        startPageNum = lastPage - 9;
                        endPageNum = lastPage;
                }
        }
        Map<String, Object> resultMap = new HashMap<String, Object>();
        resultMap.put("NoticeBoardList", NoticeBoardList);
        resultMap.put("endPageNum", endPageNum);
        resultMap.put("lastPage", lastPage);
        resultMap.put("startPageNum", startPageNum);
        return resultMap;
}
public void insertNBoard(NoticeBoard nboard) {
        String nbcode = "nb" + String.valueOf(boardMapper.getNBoardCode() + 1);
        nboard.setNboardCode(nbcode);
        nboard.setNboardView(0);
        nboard.setNboardRec(0);
        int result = boardMapper.insertNBoard(nboard);
}
public void updateNBoard(NoticeBoard nboard) {
        int result = boardMapper.updateNBoard(nboard);
}
public NoticeBoard getNBoardByNBCode(String nboardCode) {
        return boardMapper.getNBoardByNBCode(nboardCode);
}
public void increaseViewByNBCode(String nboardCode) {
        boardMapper.increaseViewByNBCode(nboardCode);
}
public void increaseRecByNBCode(String nboardCode) {
        boardMapper.increaseRecByNBCode(nboardCode);
}
public List<NoticeBoard> getBoardSearchList(NBoardSearch nbsearch) {
        String searchKey = nbsearch.getSearchKey();
        String columnName = "";
        switch (searchKey) {
        case "memberId" -> {
                columnName = "id";
        }
        case "nboardTitle" -> {
                columnName = "nb_title";
```

```
            }
            case "nboardContent" -> {
                    columnName = "nb_content";
            }
            }
            nbsearch.setSearchKey(columnName);
            log.info("nbsearch: {}", nbsearch);
            return boardMapper.getBoardSearchList(nbsearch);
    }

    public String getBCTValueNameByBCTCode(NoticeBoard nboard)
    {
            return boardMapper.getBCTValueNameByBCTCode(nboard);
    }

    public String getBoardCateCodeByBCTValue(NoticeBoard nboard)
    {
            return boardMapper.getBoardCateCodeByBCTValue(nboard);
    }

    public String getBoardCodeByNBoard(NoticeBoard nboard)
    {
            return boardMapper.getBoardCodeByNBoard(nboard);
    }
    public String getBoardCodeByBCTValue(String bctCodeValue)
    {
            return boardMapper.getBoardCodeByBCTValue(bctCodeValue);
    }

    public String getBoardInfoByBCTValue(String bctCodeValue)
    {
            return boardMapper.getBoardInfoByBCTValue(bctCodeValue);
    }

    public String getBCTValueByNBCode(String nbCode)
    {
            return boardMapper.getBCTValueByNBCode(nbCode);
    }

    public List<NoticeBoard> getMainNoticeBoard(String bCode)
    {
```

```
            return boardMapper.getMainNoticeBoard(bCode);
    }

    public List<NoticeBoard> getMainViewBoard(String bCode)
    {
            return boardMapper.getMainViewBoard(bCode);
    }

    public List<NoticeBoard> getMainRecBoard(String bCode)
    {
            return boardMapper.getMainRecBoard(bCode);
    }

    public List<NoticeBoard> getMainLatestBoard(String bCode)
    {
            return boardMapper.getMainLatestBoard(bCode);
    }

    // 게시판 파일 업로드
    // @return fileRequest
public FileRequest upLoadImgByNBCode(NBoardImg nboardimg, String nBoardCode) {
    FileRequest fileRequest =  fileUtils.uploadFile(nboardimg.getNBoardImgFile(), nBoardCode);
    if(fileRequest != null){
        fileRequest.setFileCate(nBoardCode);
        log.info("fileRequest: {}", fileRequest);
        String filePath = fileRequest.getFilePath().replace("/attachment", "/attachments");
        fileRequest.setFilePath(filePath);
        fileMapper.addFile(fileRequest);
        nboardimg.setFileIdx(fileRequest.getFileIdx());
        nboardimg.setFilePath(filePath);
    }

    //이미지 등록
    boardMapper.insertnBoardImg(nboardimg);
    return fileRequest;
}


public String getNBoardImgByNBCode(String nbcode)
{
    return boardMapper.getNBoardImgByNBCode(nbcode);
```

```
    }

    @Transactional
    public void deleteFilesAndNBoardImg(String nboardImg) {
        // nboard_img 테이블에서 삭제
        boardMapper.deleteFromNBoardImg(nboardImg);
        // files 테이블에서 삭제
        boardMapper.deleteFromFiles(nboardImg);
    }

}
```

| Package | Ks51team03/board | Mapper | BoardMapper |
|---|---|---|---|

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
                PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
                "https://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="ks51team03.board.mapper.BoardMapper">

        <resultMap type="NoticeBoard" id="nBoardResultMap">
                <!-- pk, column -->
                <id column="nbcode" property="nboardCode"/>
                <result column="id" property="memberId"/>
                <result column="bcode" property="boardCode"/>
                <result column="bctcode" property="boardCateCode"/>
                <result column="nb_title" property="nboardTitle"/>
                <result column="nb_content" property="nboardContent"/>
                <result column="nb_reg" property="nboardRegistDate"/>
                <result column="nb_update" property="nboardUpdateDate"/>
                <result column="nb_delete" property="nboardDeleteDate"/>
                <result column="nb_img" property="nboardImg"/>
                <result column="nb_view" property="nboardView"/>
                <result column="nb_rec" property="nboardRec"/>
        </resultMap>
        <select id="getNoticeBoardList" resultMap="nBoardResultMap">
                /*게시글 목록 조회*/
                SELECT
                        *
                FROM
                        notice_board AS nb
                        INNER JOIN
```

```
                board_category AS bc
                ON
                nb.bctcode=bc.bctcode
        WHERE
        bc.bct_value = #{boardCateValue}
    <if test="searchKey != null and searchValue != null">
        AND nb.${searchKey} LIKE CONCAT('%', #{searchValue}, '%')
    </if>
        ORDER BY
                CAST(SUBSTRING_INDEX(nb.nbcode, 'nb', -1) AS SIGNED)
        LIMIT #{startRow}, #{rowPerPage};
</select>

<select id="getNoticeBoardListCnt" parameterType="String" resultType="int">
        SELECT
        COUNT(1)
        FROM
            notice_board AS nb
            INNER JOIN
            board_category AS bc
            ON nb.bctcode = bc.bctcode
        WHERE
        bc.bct_value = #{boardCateValue}
    <if test="searchKey != null and searchValue != null">
        AND nb.${searchKey} LIKE CONCAT('%', #{searchValue}, '%')
    </if>;
</select>

<insert id="insertNBoard" parameterType="NoticeBoard" >
/*게시글 작성*/
INSERT INTO notice_board
(       nbcode,
        id,
        bcode,
        bctcode,
        nb_title,
        nb_content,
        nb_reg,
        nb_update,
        nb_delete,
        nb_img,
```

```
            nb_view,
            nb_rec
    )VALUES(
            #{nboardCode},
            #{memberId},
            #{boardCode},
            #{boardCateCode},
            #{nboardTitle},
            #{nboardContent},
            CURDATE(),
            #{nboardUpdateDate},
            #{nboardDeleteDate},
            #{nboardImg},
            #{nboardView},
            #{nboardRec}
    );
    </insert>

    <select id="getNBoardCode" resultType="int">
            /* 게시글코드 도출*/
            SELECT
            MAX(CAST(REGEXP_REPLACE(nbcode, '[^0-9]', '') AS UNSIGNED)) AS max_number
            FROM
            notice_board;
    </select>

    <select id="getNBoardByNBCode" parameterType="String" resultMap="nBoardResultMap">
            /* 게시글 코드에 맞는 게시글 열람*/
            SELECT
                    *
            FROM
                    notice_board AS nb
            WHERE
                    nb.nbcode=#{nboardCode};
    </select>

    <update id="increaseViewByNBCode" parameterType="String">
            /* 게시글 코드에 맞는 조회수 증가*/
            UPDATE
                    notice_board
            SET
```

```
                nb_view=nb_view+1
        WHERE
                nbcode=#{nboardCode};
</update>

<update id="increaseRecByNBCode" parameterType="String">
        /* 게시글 코드에 맞는 추천수 증가*/
        UPDATE
                notice_board
        SET
                nb_rec=nb_rec+1
        WHERE
                nbcode=#{nboardCode};
</update>

<update id="updateNBoard" parameterType="NoticeBoard">
        /* 게시글 수정*/
        UPDATE
                notice_board
        <trim prefix="SET" suffixOverrides=",">
                <if test="memberId != null and memberId != ''">
                id=#{memberId},
                </if>
                <if test="boardCode != null and boardCode != ''">
                bcode=#{boardCode},
                </if>
                <if test="boardCateCode != null and boardCateCode != ''">
                bctcode=#{boardCateCode},
                </if>
                <if test="nboardTitle != null and nboardTitle != ''">
                nb_title=#{nboardTitle},
                </if>
                <if test="nboardContent != null and nboardContent != ''">
                nb_content=#{nboardContent},
                </if>
                <if test="nboardRegistDate != null and nboardRegistDate != ''">
                nb_reg=#{nboardRegistDate},
                </if>
                <if test="nboardImg != null and nboardImg != ''">
                nb_img=#{nboardImg}
                </if>
```

```
            </trim>
            WHERE
            nbcode = #{nboardCode};
    </update>

    <select id="getBCTValueNameByBCTCode" parameterType="NoticeBoard" resultType="String">
            SELECT
                    bct_value
            FROM
                    notice_board AS nb
                    INNER JOIN
                    board_category AS bc
                    ON
                    nb.bctcode=bc.bctcode
            WHERE
                    nb.nbcode=#{nboardCode};
    </select>

    <select id="getBoardCateCodeByBCTValue" parameterType="NoticeBoard" resultType="String">
            SELECT
                    bctcode
            FROM
                    board_category
            WHERE
                    bct_value=#{boardCateCode};
    </select>

    <select id="getBoardCodeByNBoard" parameterType="NoticeBoard" resultType="String">
            SELECT
                    bcode
            FROM
                    board_category
            WHERE
                    bct_value=#{boardCateCode};
    </select>

    <select id="getBoardCodeByBCTValue" parameterType="String" resultType="String">
            SELECT
                    bcode
            FROM
                    board_category
```

```
        WHERE
                bct_value=#{boardCateCode};
</select>


<select id="getBoardInfoByBCTValue" parameterType="String" resultType="String">
        /*게시글 게시글 카테고리 이름으로 게시글 설명 도출*/
        SELECT
                bct_info
        FROM
                board_category
        WHERE
                bct_value=#{boardCateCode};
</select>

<select id="getMainNoticeBoard" parameterType="String" resultMap="nBoardResultMap">
        /* 게시글 코드에 맞는 게시판 최신순*/
        SELECT
                *
        FROM
                notice_board
        WHERE
                bcode=#{boardCode}
        ORDER BY nb_reg DESC;
</select>

<select id="getMainViewBoard" parameterType="String" resultMap="nBoardResultMap">
        /* 게시판 조회순*/
        SELECT
                *
        FROM
                notice_board
        ORDER BY nb_view DESC
        LIMIT 10;
</select>

<select id="getMainRecBoard" parameterType="String" resultMap="nBoardResultMap">
        /* 게시판 추천순*/
        SELECT
                *
        FROM
```

```
                notice_board
        ORDER BY nb_rec DESC
        LIMIT 10;
</select>

<select id="getMainLatestBoard" parameterType="String" resultMap="nBoardResultMap">
        /* 게시판 최신순*/
        SELECT
                *
        FROM
                notice_board
        ORDER BY nb_reg DESC
        LIMIT 10;
</select>

<select id="getBCTValueByNBCode" parameterType="String" resultType="String">
        /* 게시글 코드로 게시판 이름 도출*/
        SELECT
                bc.bct_value
        FROM
                notice_board AS nb
                INNER JOIN
                board_category AS bc
                ON
                nb.bctcode=bc.bctcode
        WHERE
                nb.nbcode=#{nboardCode};
</select>

<insert id="insertnBoardImg" parameterType="NBoardImg">
         /* 게시글 이미지 추가 */
    <selectKey keyProperty="nbiCode" resultType="String" order="BEFORE">
        SELECT
            CONCAT('nbi', COALESCE(MAX(CAST(SUBSTRING(nbicode, 4) AS UNSIGNED)), 0) + 1)
        FROM nboard_img
    </selectKey>
        -- 게시글 대표 이미지 등록
        INSERT INTO nboard_img
        (       nbicode,
                nbcode,
                file_idx,
```

```
                file_cate
        )
        VALUES
        (       #{nbiCode},
                #{nbCode},
                #{fileIdx},
                '게시글대표이미지'
        )
</insert>

<select id="getNBoardImgByNBCode" parameterType="String" resultType="String">
        /*게시글 코드로 이미지 경로 가져오기*/
        SELECT
                f.file_path
        FROM
                nboard_img as ni
                LEFT JOIN
                files as f
                ON
                ni.file_idx=f.file_idx
        WHERE
        ni.nbcode = #{nbcode}
        LIMIT 1;
</select>

<delete id="deleteFromNBoardImg" parameterType="String">
        /*게시글이미지 삭제*/
    DELETE FROM nboard_img
    WHERE
        file_idx
    IN
    (
        SELECT file_idx
        FROM files
        WHERE file_path = #{nboardImg}
)
</delete>

<delete id="deleteFromFiles" parameterType="String">
        /*게시글 파일 삭제*/
DELETE FROM files
```

```
        WHERE
                file_path = #{nboardImg}
        </delete>
        <!--        <select        id="getBoardSearchList"        parameterType="NBoardSearch"
resultMap="nBoardResultMap">
                /*게시글 검색*/
                SELECT
                        *
                FROM
                        notice_board AS nb
                        INNER JOIN
                        board_category AS bc
                        ON
                        nb.bctcode=bc.bctcode
                WHERE
                        bc.bct_value=#{boardCateValue}
                        AND
                        nb.${searchKey} LIKE CONCAT('%', #{searchValue}, '%')
                ORDER BY
                        CAST(SUBSTRING_INDEX(nb.nbcode, 'nb', -1) AS SIGNED);
        </select> -->
</mapper>
```

| Package | Ks51team03/pet | Controller | PetController |
|---|---|---|---|

```
package ks51team03.pet.controller;
import java.util.List;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;
import jakarta.servlet.http.HttpSession;
import ks51team03.member.dto.Member;
import ks51team03.member.dto.MemberLevel;
import ks51team03.member.service.MemberService;
import ks51team03.pet.dto.Pet;
import ks51team03.pet.service.PetService;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
@Controller
@RequiredArgsConstructor
```

```
@Slf4j
public class PetController {
	private final PetService petService;
	private final MemberService memberService;

	/*업체 등록*/
	@PostMapping("/pet/insertPet")
	public String insertPet(Pet pet,HttpSession session) {
		String result="";

		if(session.getAttribute("SID")!=null)
		{
			String memberId=(String) session.getAttribute("SID");

			petService.insertPet(pet,memberId);
			memberService.IncreasePetByMemberId(memberId);

			result="/member/member_mypage_list_pet";
		}
		else
		{
			Member member=(Member)session.getAttribute("SMEM");
			log.info("반려동물등록 Company:{}", pet);
			log.info("회원가입 Member:{}", member);
			memberService.insertMember(member);
			petService.insertPet(pet,member.getMemberId());

			session.invalidate();

			result="/member/member_main";
		}


		return "redirect:"+result;
	}

	@GetMapping("/pet/insertPet")
	public String insertPet(Model model) {

		model.addAttribute("title", "업체등록");
```

```
        return "member/member_login_insert_pet";
    }

    @PostMapping("/pet/updatePet")
    public String updatePet(Pet pet) {
        log.info("반려동물수정 Pet:{}", pet);
        petService.updatePet(pet);
        return "redirect:/pet/member_mypage_petinfo?petCode=" + pet.getPetCode();
    }

    @GetMapping("/pet/updatePet")
    public String updatePet(@RequestParam(value="petCode") String petCode
                                                    ,Model model) {
        log.info("수정화면 petCode:{}", petCode);
        Pet petInfo = petService.getPetInfoByPetCode(petCode);

        model.addAttribute("title", "회원수정");
        model.addAttribute("petInfo", petInfo);

        return "member/member_login_update_pet";
    }

    @GetMapping("/pet/removePet")
    public String removePet(@RequestParam(value="petCode") String petCode
                                                    ,Model model,HttpSession session) {

        String memberId=(String) session.getAttribute("SID");

        petService.removePet(petCode);
        memberService.DeclinePetByMemberId(memberId);

        return "redirect:/member/member_mypage_list_pet";
    }


    @GetMapping("/pet/member_mypage_petinfo")
    public String userMyPagePetInfo(@RequestParam(value = "petCode") String petCode, Model model)
    {
        Pet petInfo = petService.getPetInfoByPetCode(petCode);
        model.addAttribute("petInfo", petInfo);
        return "member/member_mypage_petinfo";
```

```
        }
}
```

| Package | Ks51team03/pet | Service | PetService |
|---|---|---|---|

```
package ks51team03.pet.service;
import java.util.List;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import ks51team03.member.dto.Member;
import ks51team03.pet.dto.Pet;
import ks51team03.pet.mapper.PetMapper;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
@Service("PetService")
@Transactional
@RequiredArgsConstructor
@Slf4j
public class PetService {
        private final PetMapper petMapper;

        // 반려동물 등록
    public void insertPet(Pet pet,String memberId) {
        String pcode="pet"+String.valueOf(petMapper.getPetCode()+1);

        pet.setPetCode(pcode);
        pet.setMemberId(memberId);
        pet.setPetUrl("notFound");

                int result = petMapper.insertPet(pet);
        }

    // 회원 아이디로 반려동물 검색
    public List<Pet> getPetInfoByMemberId(String memberId) {

                return petMapper.getPetInfoByMemberId(memberId);
        }

    //펫코드로 반려동물 검색
    public Pet getPetInfoByPetCode(String petCode)
    {
        return petMapper.getPetInfoByPetCode(petCode);
```

```
    }

    //펫코드로 반려동물정보 수정
    public int updatePet(Pet pet) {

                return petMapper.updatePet(pet);
        }

    //펫코드로 반려동물 삭제
    public void removePet(String petCode)
    {
        petMapper.removePet(petCode);
    }

}
```

| Package | Ks51team03/pet | Mapper | PetMapper |
|---|---|---|---|

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
                PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
                "https://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="ks51team03.pet.mapper.PetMapper">
        <resultMap type="Pet" id="petResultMap">
                <!-- pk, column -->
                <id column="pcode" property="petCode"/>
                <result column="id" property="memberId"/>
                <result column="p_name" property="petName"/>
                <result column="p_gender" property="petGender"/>
                <result column="p_birth" property="petBirth"/>
                <result column="p_class" property="petClass"/>
                <result column="p_species" property="petSpecies"/>
                <result column="p_breed" property="petBreed"/>
                <result column="p_weight" property="petWeight"/>
                <result column="p_neuter" property="petNeuter"/>
                <result column="p_operation" property="petOperation"/>
                <result column="p_medicine" property="petMedicine"/>
                <result column="p_note" property="petNote"/>
                <result column="p_url" property="petUrl"/>
                <result column="p_regist_date" property="petRegistDate"/>
        </resultMap>
```

```
<insert id="insertPet" parameterType="Pet">
        /* 업체등록 */
        INSERT INTO pet
        (       pcode,
                id,
                p_name,
                p_gender,
                p_birth,
                p_class,
                p_species,
                p_breed,
                p_weight,
                p_neuter,
                p_operation,
                p_medicine,
                p_note,
                p_url,
                p_regist_date
        )VALUES(
                #{petCode},
                #{memberId},
                #{petName},
                #{petGender},
                #{petBirth},
                #{petClass},
                #{petSpecies},
                #{petBreed},
                #{petWeight},
                #{petNeuter},
                #{petOperation},
                #{petMedicine},
                #{petNote},
                #{petUrl},
                CURDATE()
        );
</insert>

<select id="getPetCode" resultType="int">
        /* 반려동물코드 도출*/
        SELECT
        MAX(CAST(REGEXP_REPLACE(pcode, '[^0-9]', '') AS UNSIGNED)) AS max_number
```

```
        FROM
        pet;
</select>

<select id="getPetInfoByMemberId" parameterType="String" resultMap="petResultMap">
        /* 특정회원 정보조회 */
        SELECT
                pcode,
                id,
                p_name,
                p_gender,
                p_birth,
                p_class,
                p_species,
                p_breed,
                p_weight,
                p_neuter,
                p_operation,
                p_medicine,
                p_note,
                p_url,
                p_regist_date
        FROM
                pet
        WHERE
                id = #{memberId};
</select>

<select id="getPetInfoByPetCode" parameterType="String" resultMap="petResultMap">
        /* 펫코드에 따라 특정 반려동물 정보조회 */
        SELECT
                pcode,
                id,
                p_name,
                p_gender,
                p_birth,
                p_class,
                p_species,
                p_breed,
                p_weight,
                p_neuter,
```

```
                p_operation,
                p_medicine,
                p_note,
                p_url,
                p_regist_date
        FROM
                pet
        WHERE
                pcode = #{petCode};
</select>

<update id="updatePet" parameterType="Pet">
        /* 반려동물수정 */
        UPDATE pet
        <set>
                <if test="petCode != null and petCode != ''">
                        pcode = #{petCode},
                </if>
                <if test="memberId != null and memberId != ''">
                        id = #{memberId}
                </if>
                <if test="petName != null and petName != ''">
                        p_name = #{petName},
                </if>
                <if test="petGender != null and petGender != ''">
                        p_gender = #{petGender},
                </if>
                <if test="petBirth != null and petBirth != ''">
                        p_birth = #{petBirth},
                </if>
                <if test="petClass != null and petClass != ''">
                        p_class = #{petClass},
                </if>
                <if test="petSpecies != null and petSpecies != ''">
                        p_species = #{petSpecies},
                </if>
                <if test="petBreed != null and petBreed != ''">
                        p_breed = #{petBreed},
                </if>
                <if test="petWeight != null and petWeight != ''">
                        p_weight = #{petWeight},
```

```
                        </if>
                        p_neuter = #{petNeuter},
                        p_operation = #{petOperation},
                        <if test="petMedicine != null">
                                p_medicine = #{petMedicine},
                        </if>
                        <if test="petNote != null and petNote != ''">
                                p_note = #{petNote},
                        </if>
                        <if test="petUrl != null and petUrl != ''">
                                p_url = #{petUrl},
                        </if>
                        <if test="petRegistDate != null and petRegistDate != ''">
                                p_regies_date = #{petRegistDate},
                        </if>
                </set>
                WHERE
                        pcode = #{petCode}
        </update>

        <delete id="removePet" parameterType="String">
                /*펫코드로 반려동물 삭제*/
                DELETE
                FROM
                        pet
                WHERE
                        pcode=#{petCode};
        </delete>

</mapper>
```