





INDEX

- 프로젝트 개요

- 팀 구성 및 개발 일정

- 시스템 구조

- 분석 및 설계

- 주요 기능과 코드

- 프로젝트 시연

- 프로젝트 마이킹 리뷰

- Q&A

1. 프로젝트

개요



축구 , 풋살 좋아하세요?

‘날이 갈수록 증가 하는 풋살에 대한 관심’

생활체육 동호회 가입 1위는 축구·풋살...골프는 2위

연합뉴스 | 승인 2023.01.12 16:13 | 13면 | 댓글 0

| 2022년 우리 국민 생활체육 참여율 61.2%로 회복세

지역	20년	전년대비	21년	전년대비	22년	전년대비	23년	전년대비
서울북동	2029	-	2261	11%p	2935	30%p	2816	-4%p
서울북서	2031	-	1634	-20%p	2299	41%p	2422	5%p
서울남동	997	-	1499	50%p	2508	67%p	2353	-6%p
서울남서	678	-	1168	72%p	1582	35%p	1803	14%p
인천	1171	-	2445	109%p	3781	55%p	3918	4%p
총	6906	-	9007	30%p	13105	45%p	13312	2%p

새롭게 유입되는 많은 사람들

점점 많아지는 구장
이용률

생활체육 동호회 가입 1위는 축구·풋살...골프는 2위

연합뉴스 | 승인 2023.01.12 16:13 | 13면 | 댓글 0

| 2022년 우리 국민 생활체육 참여율 61.2%로 회복세



풋살은 팀 스포츠 다

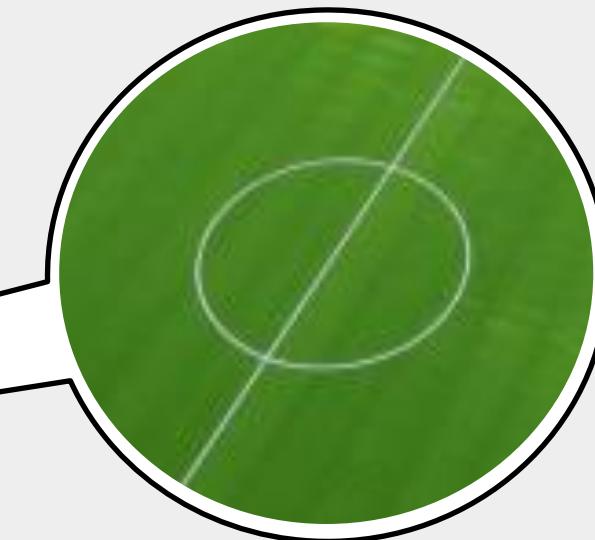
혼자서 할 수 없는 풋살을 하기위해

막상 어디서 가입 해야 할지, 구장을 어떻게 예약해야 할지

막막해 하는 사람들과 구단들이 많습니다.



만들자 CHAL-LAE ?!



CHAL-LAE?

팀원을 모집하는
구단

CHAL-LAE!

구단에 가입하는
팀원

CHAL – LAE is for FUTSAL

좋은 구장들은 알리고

구단 관리는 더 쉽게



CHAL – LAE is for FUTSAL



‘ 좋은 구장은 알리고,
구단 관리는 더
쉽게 ’



시설을 좋은데 홍보가 안된 구장



지인으로만 팀을 꾸렸던 구단



새로운 구장과 팀원이 필요한 구단

비즈니스 모델

핵심파트너	핵심활동	제공가치	고객관계	목표고객
<div>광고주</div> <div>경기장 관리인</div> <div>구장관계자</div>	<div>구장 예약 서비스</div> <div>팀 전략판 생성</div> <div>핵심자원</div> <div>OPEN API(카카오맵, 날씨)</div> <div>풋살장 정보</div>	<div>풋살 동호회 가입</div> <div>풋살 동호회 팀 생성</div> <div>팀 경기 일정 확인</div> <div>경기 정보 확인</div> <div>팀원 소통공간 마련</div>	<div>유저 → 커뮤니티</div> <div>구장 → 중개 플랫폼</div> <div>채널(경로)</div> <div>웹(PC)</div>	<div>풋살 동호회 감독</div> <div>풋살 동호회 선수</div>
비용구조		수익구조(수익원)		
<div>사이트 유지 관리비</div> <div>인건비</div>		<div>기업광고</div> <div>구단 구독료</div> <div>구장 인센티브</div>		

2. 팀 구성과

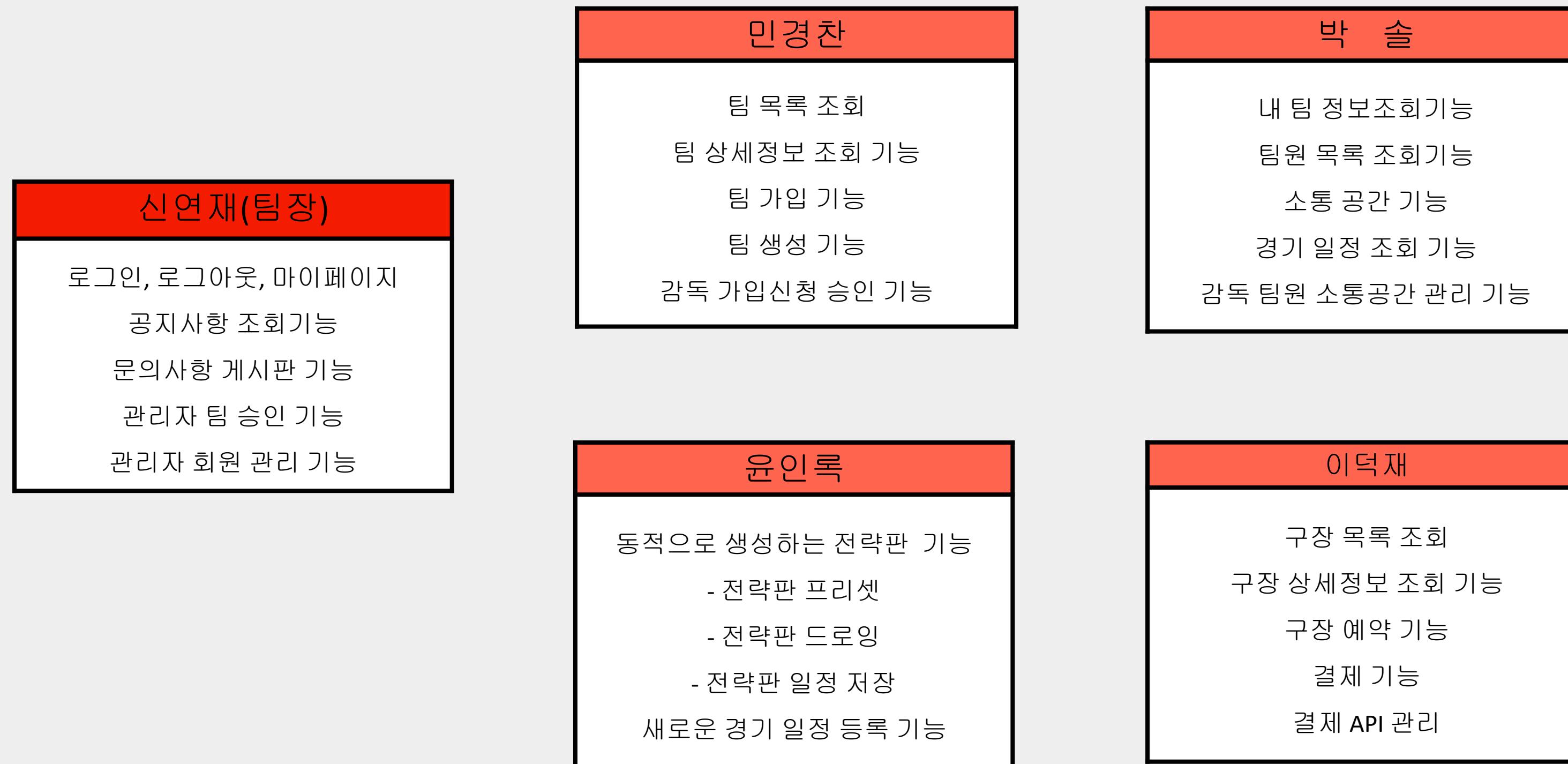
일정



팀원 소개 및 역할 [디자인]



팀원 소개 및 역할 [기능]



팀원 소개 및 역할 [기타]

신연재(팀장)

팀 총괄 관리

SQL문 제작 , DB 관리, erd 제작
프로젝트 명세
- 간트차트, 기능 코드, 비즈니스 모델

민경찬

SQL문 제작 , DB 관리, erd 제작
프로젝트 명세
- 도메인 모델

박 솔

SQL문 제작 , DB 관리, erd 제작
PPT 제작 및 프로젝트 명세
- 업무 분장, 유스케이스

윤인록

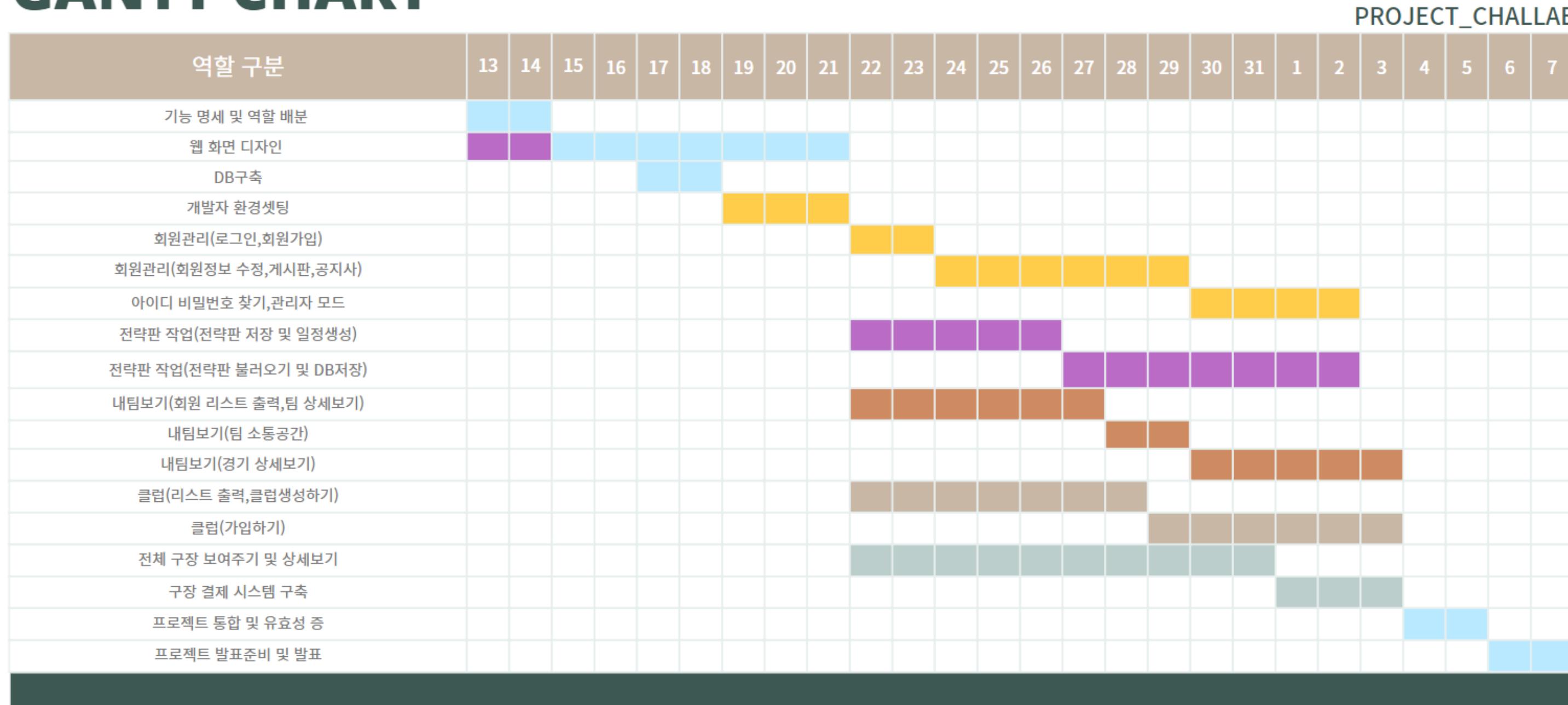
SQL문 제작 , DB 관리, erd 제작
프로젝트 명세
- 마인드 맵, 메뉴 구조도

이덕재

SQL문 제작 , DB 관리, erd 제작
프로젝트 명세
- 스토리 보드

개발 일정 (간트 차트)

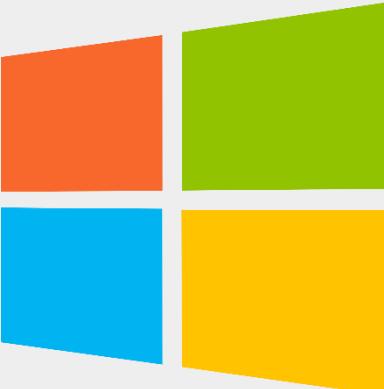
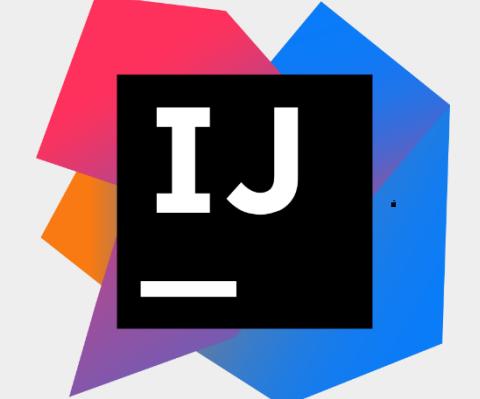
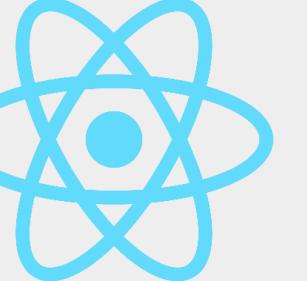
GANTT CHART



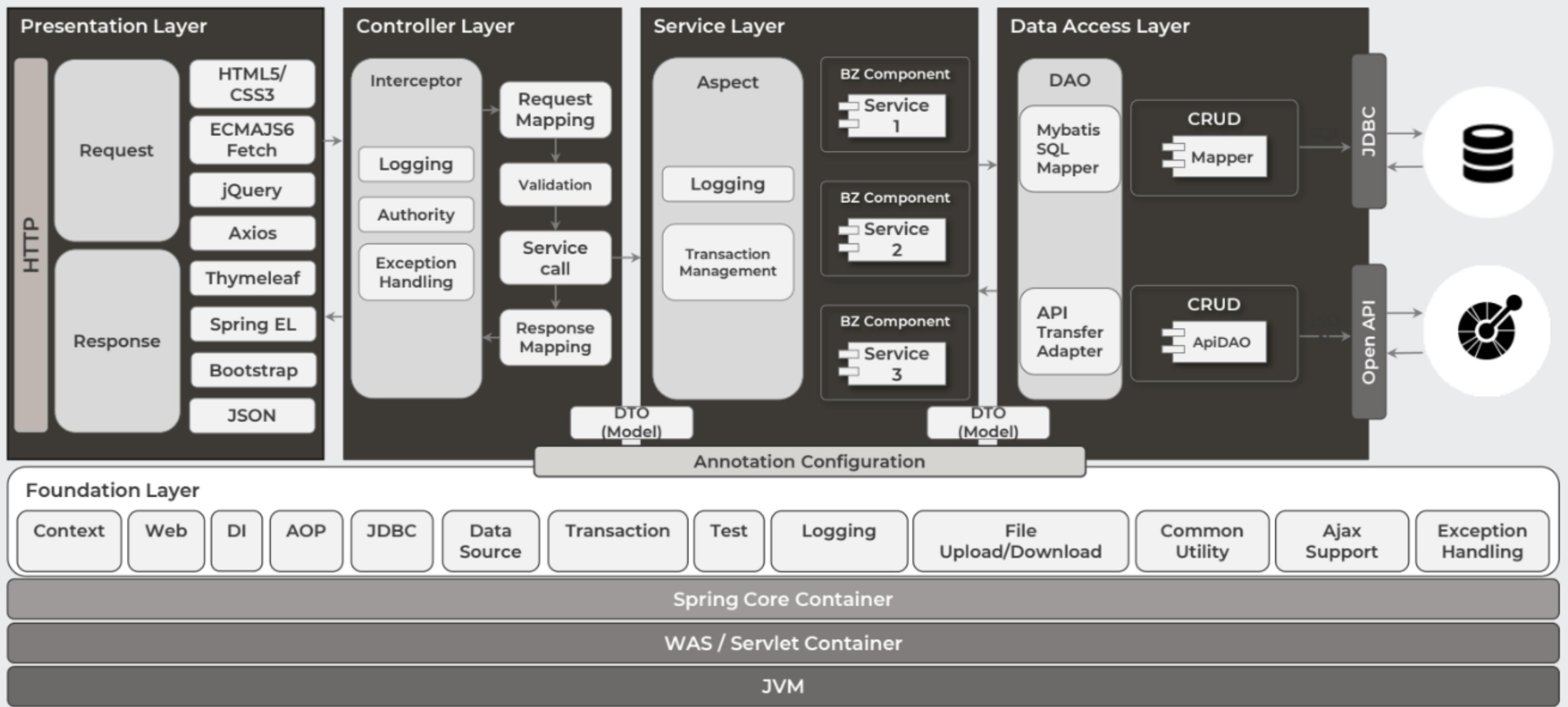
3. 시스템 구조



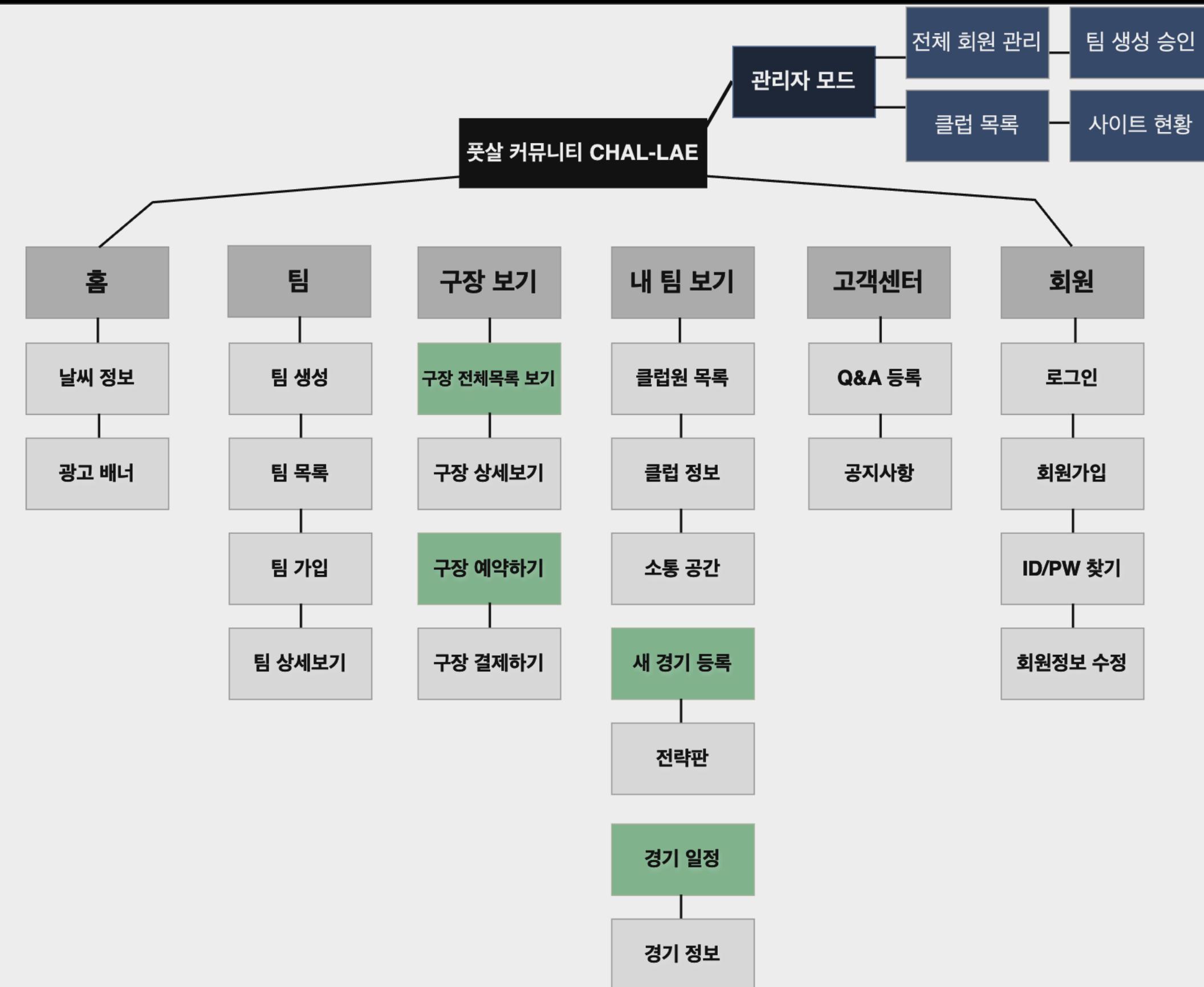
개발환경 및 적용기술

운영체제	프로그램	프론트엔드	백엔드
 	 	    	    

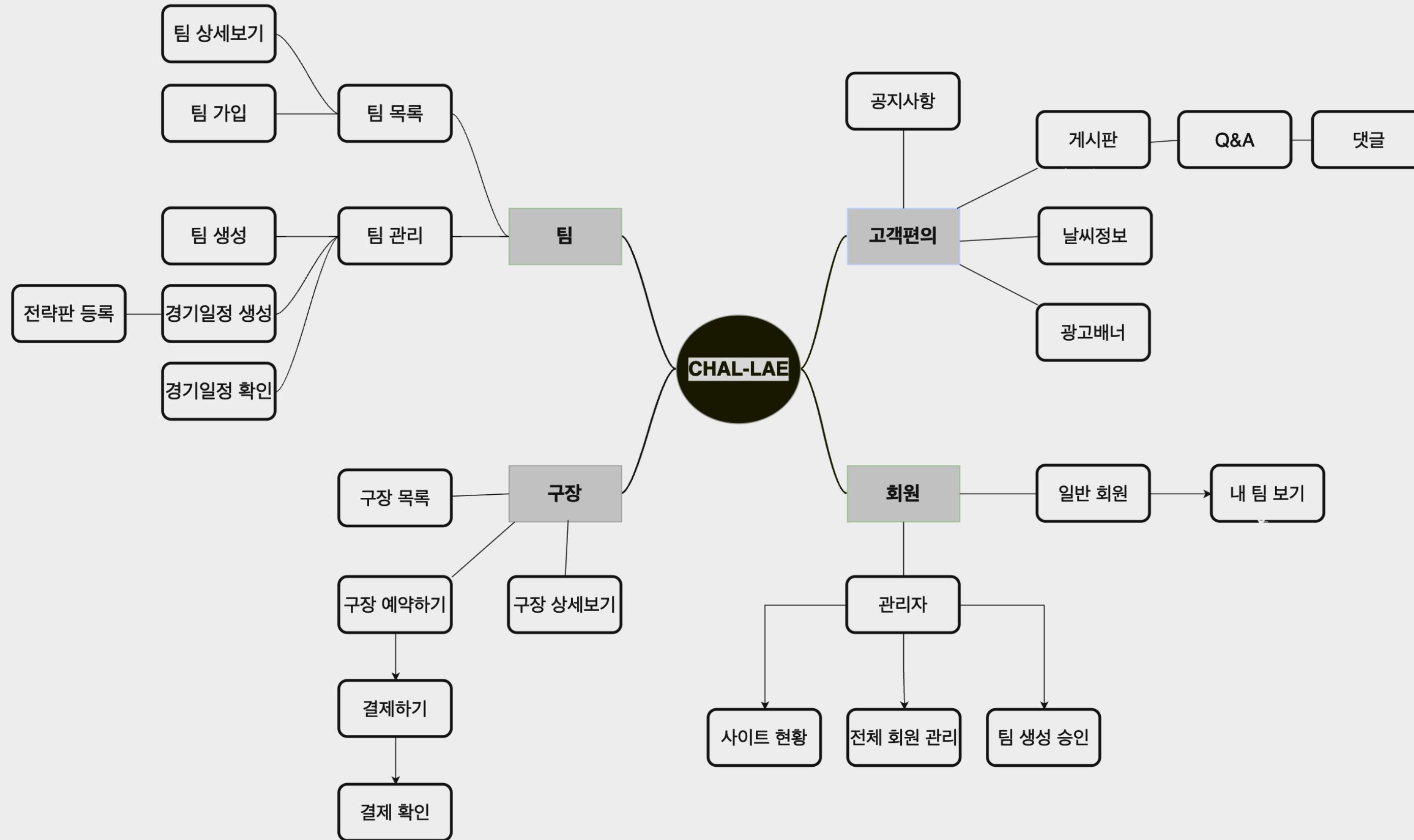
시스템 구성도



메뉴 구조도



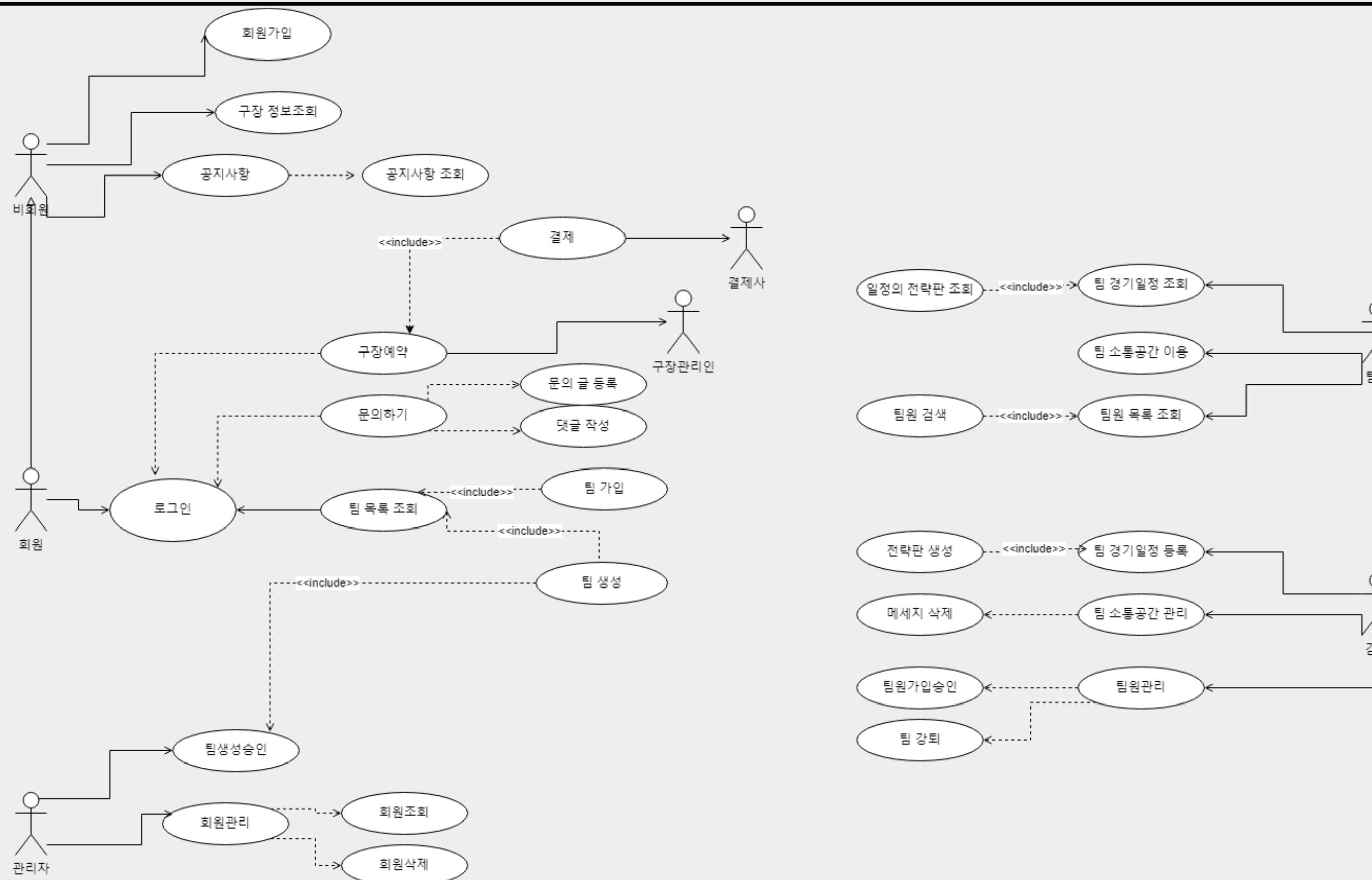
마인드 맵



4. 분석 및 설계

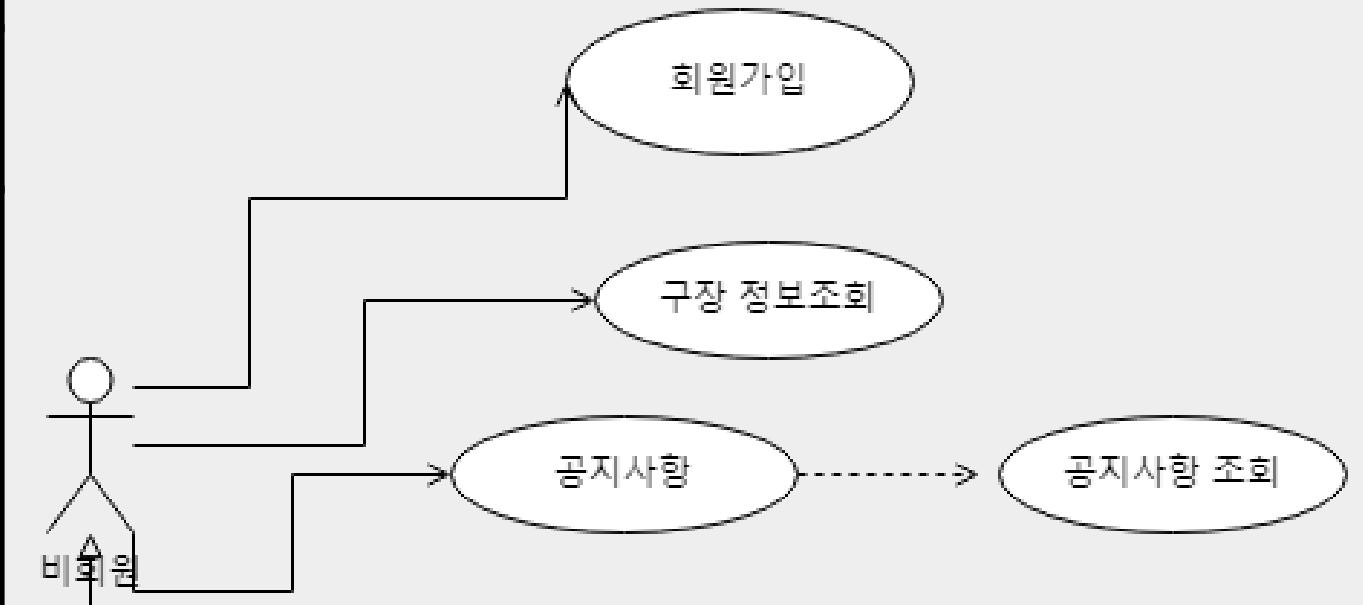


USECASE 모델



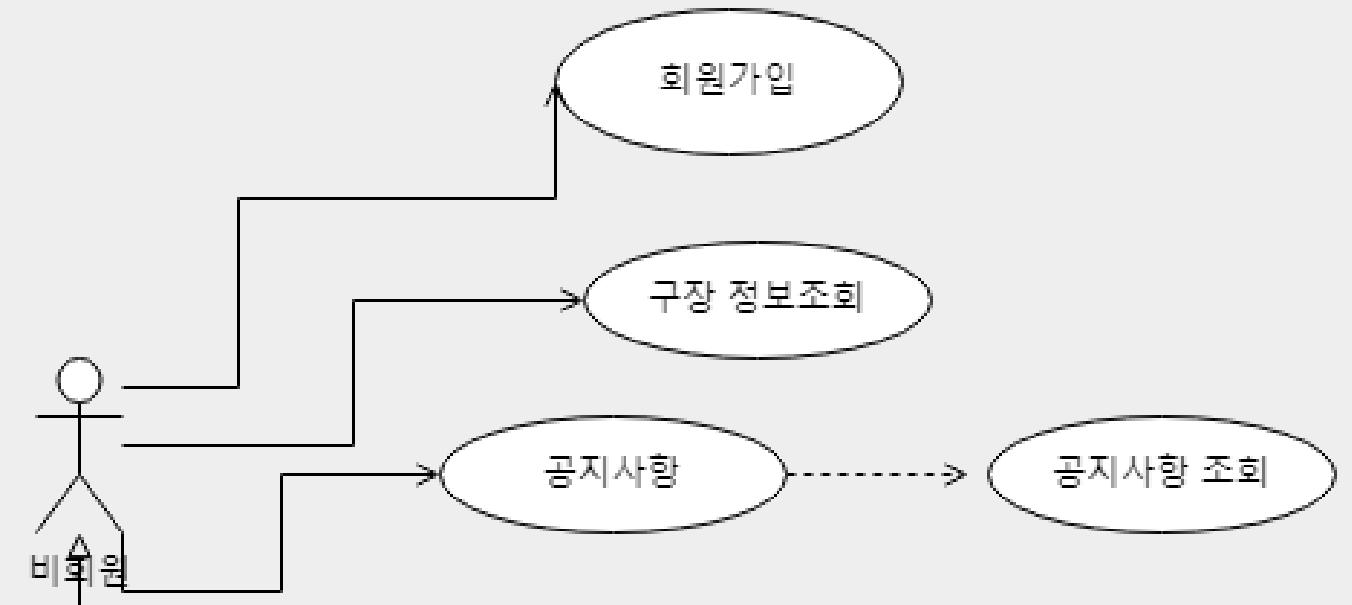
USECASE 명세서

유스케이스명	공지사항
액터명	비회원
개요	비회원이 선택한 공지사항 조회
사전조건	비회원이 사이트를 접속 해야한다
사후조건	단순 조회로 사후 조건 없음
기본 흐름	<p>1. 비회원은 메인에서 '고객센터' 선택</p> <p>2. 시스템은 '고객센터' 페이지를 비회원에게 제공</p> <p>3. 비회원은 공지사항 중 원하는 게시글을 선택</p> <p>4. 시스템은 비회원이 선택한 게시글에 대한 데이터를 서버에서 받아 출력 및 제공</p>
대체 흐름	<p><u>3-1 선택한 게시글이 없을 경우</u></p> <p>1. 페이지가 다시 로드되며, 없는 공지사항의 경우 목록에서 사라진 채로 비회원에게 제공</p> <p>[흐름 종료]</p>



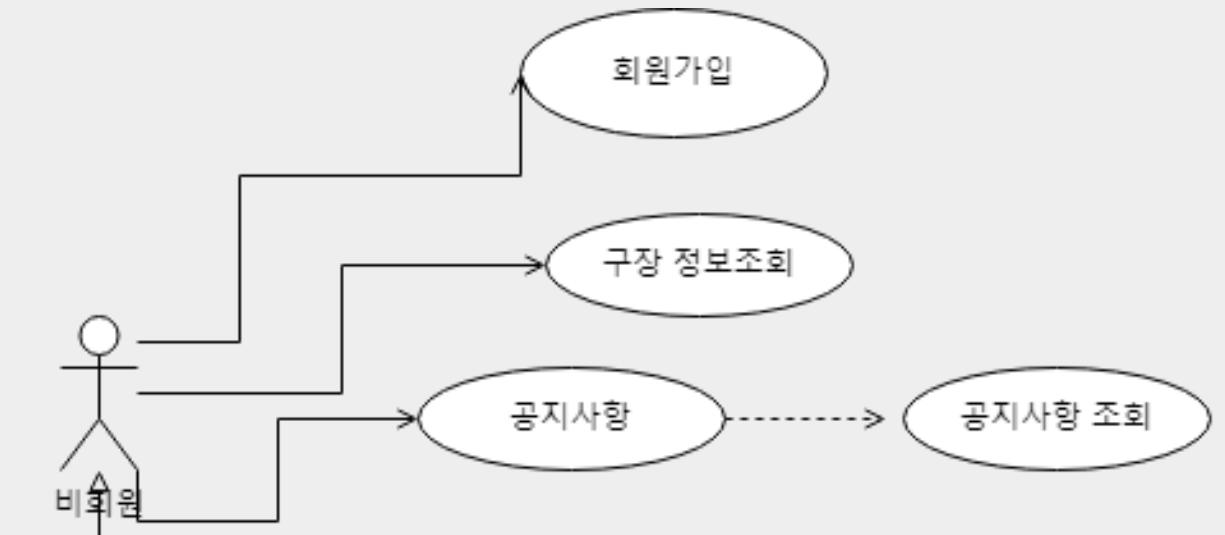
USECASE 명세서

유스케이스명	구장정보조회
액터명	비회원
개요	비회원이 선택한 구장 정보 조회
사전조건	비회원이 사이트를 접속 해야한다
사후조건	단순 조회로 사후 조건 없음
기본 흐름	<ol style="list-style-type: none"> 비회원은 메인에서 '구장정보' 선택 시스템은 서버가 가지고 있는 구장목록 제공 비회원은 구장목록에서 원하는 구장을 선택 시스템은 비회원이 선택한 구장의 정보를 제공
대체 흐름	<p>2-1 이미 삭제된 구장 정보일 경우</p> <ol style="list-style-type: none"> 시스템은 페이지를 새로고침하여 서버에서 가져온 데이터로 목록을 제공 [기본흐름 3부터 시작]



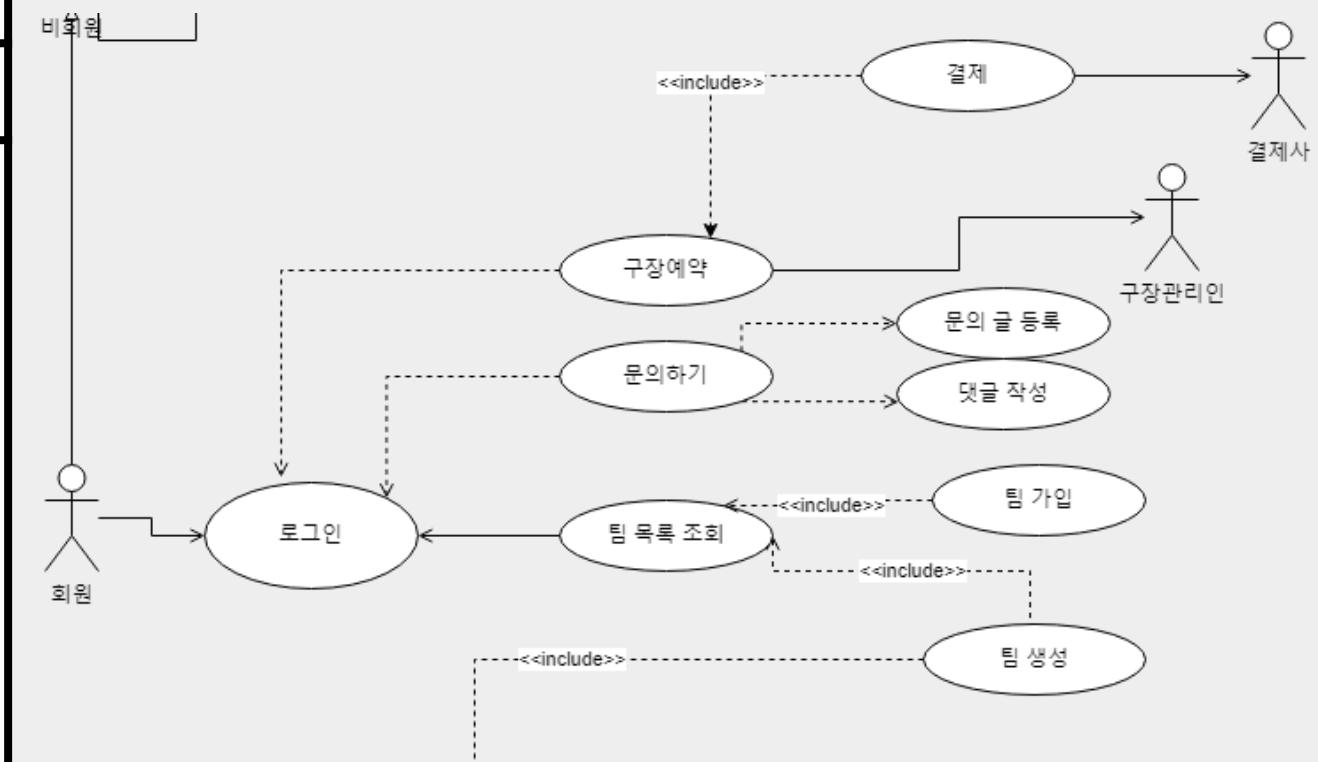
USECASE 명세서

유스케이스명	회원가입
액터명	비회원
개요	비회원은 사이트에 회원가입 가능
사전조건	사용하려는 회원의 정보가 사이트 회원의 정보 중 중복이 없어야 한다.
사후조건	서버 데이터에 저장되어 있어야 한다.
기본 흐름	<ol style="list-style-type: none"> 1. 비회원은 메인에서 '회원가입'을 선택 2. 시스템은 회원가입 정보입력을 위한 양식을 제공 3. 비회원은 제공된 양식에 맞는 정보를 입력 4. 시스템은 입력된 정보를 서버데이터에 저장 및 비회원에게 회원가입완료 확인창을 제공
대체 흐름	<p><u>3-1 비회원이 양식에 맞지 않은 정보를 입력할 경우</u></p> <ol style="list-style-type: none"> 1. 시스템은 입력한 정보 중 양식이 맞지 않는 정보에 대한 사항에 대한 경고창을 비회원에게 제공 2. 비회원은 안내된 창에 대한 내용을 확인하고 올바른 정보로 입력 (이하 기본흐름 4부터 시작)



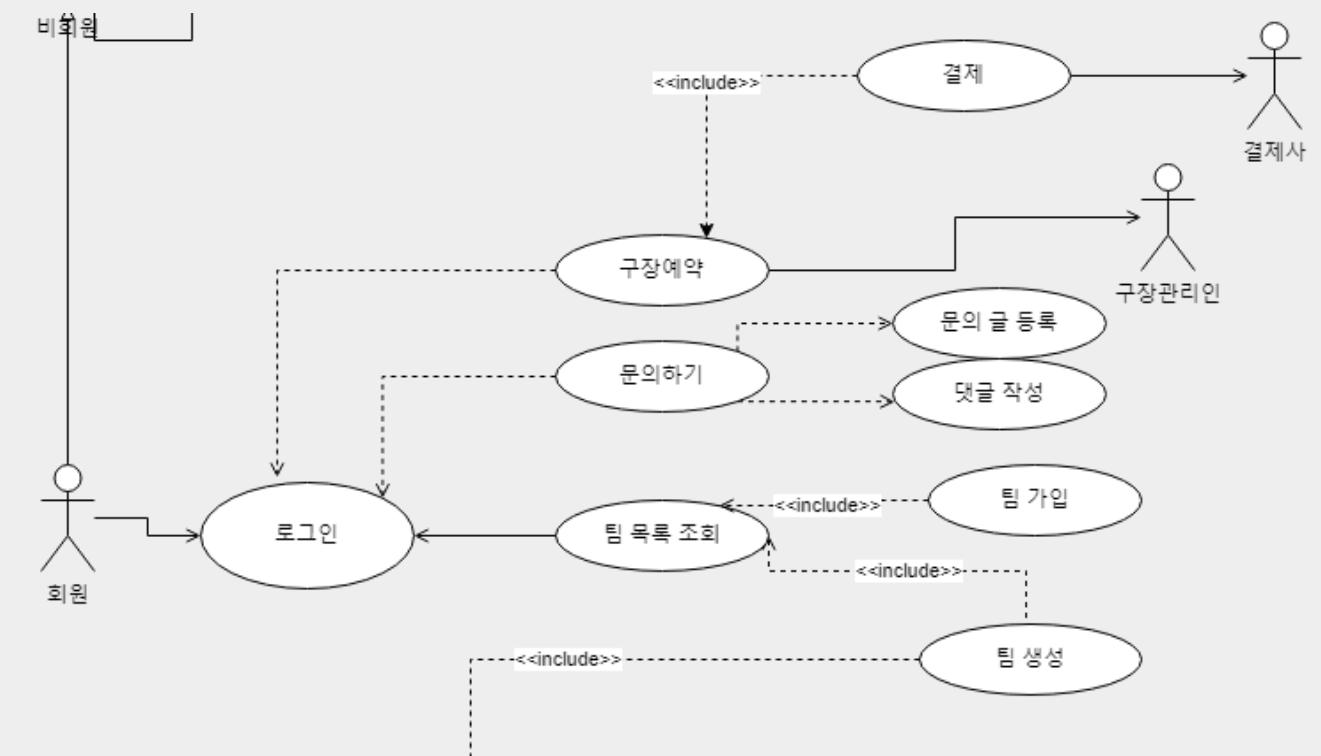
USECASE 명세서

유스케이스명	문의하기
액터명	회원
개요	회원은 문의하고 싶은 사항에 대해 게시글이나 댓글로 입력 가능
사전조건	로그인 되어있는 회원이어야 한다.
사후조건	게시글 댓글 수가 게시글 타이틀에 제공되어야 한다.
기본 흐름	<ol style="list-style-type: none"> 1.로그인한 회원이 메인화면의 '고객센터'를 선택 2.시스템은 '고객센터' 페이지를 회원에게 제공 3.회원은 문의사항부분의 '작성'을 선택 4.시스템은 문의사항게시글 작성 양식을 제공 5.회원은 양식에 맞게 문의사항 게시글을 작성 및 '등록' 선택 6.시스템은 회원이 작성한 게시글을 서버에 저장 및 회원에게 게시글 목록 제공 7.회원은 원하는 게시글 선택 8.시스템은 회원이 선택한 게시글의 상세정보를 회원에게 제공 9.회원은 게시글 상세정보 페이지의 입력칸에 내용을 작성 후 '등록' 선택 10.시스템은 작성한 회원과 입력한 내용, 댓글작성시간, 댓글작성한 게시글에 대한 데이터를 서버에 저장
대체 흐름	<p><u>5-1 회원이 양식에 맞지 않게 작성 했을 경우</u></p> <ol style="list-style-type: none"> 1. 시스템은 회원에게 양식에 맞지 않은 부분을 안내 2.회원은 양식의 맞지 않는 부분을 수정 및 '등록' 선택 <p>[이하 기본흐름 6부터 시작]</p>



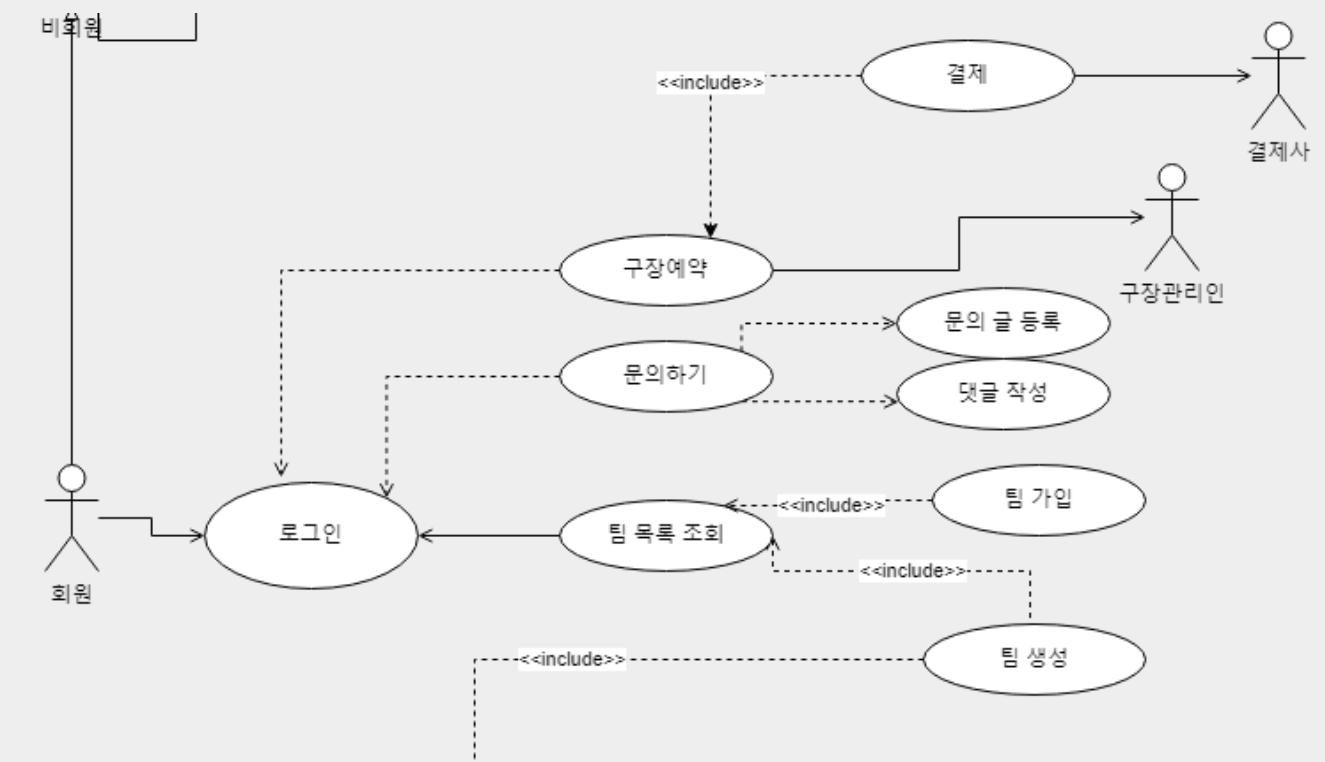
USECASE 명세서

유스케이스명	구장예약
액터명	회원, 구장관리인, 결제사
개요	회원은 선택한 구장의 정보로 원하는 시간대와 날짜에 구장예약 및 금액 결제
사전조건	회원이 로그인 상태여야 한다.
사후조건	구장예약 및 결제완료
기본 흐름	<p>1.로그인 한 회원이 메인에서 '구장보기'를 선택</p> <p>2.시스템은 서버에 저장되어있는 구장의 목록 화면을 제공</p> <p>3.회원은 원하는 구장의 "구장예약"을 선택</p> <p>4.시스템은 선택한 구장의 데이터, 로그인한 회원 데이터와 함께 회원이 선택할 캘린더와 시간선택 창을 제공</p> <p>5.회원은 예약을 희망하는 날짜와 시간을 선택 후 '결제'버튼을 선택</p> <p>6.시스템은 선택했던 구장의 가격을 결제 api로 안내, 결제기능을 제공</p> <p>7.회원은 결제, 결제 사에서 결제 승인</p> <p>8.시스템은 결제완료를 확인하고 예약한 내역을 저장 및 구장관리인에게 전송, 화면으로 회원에게 제공</p>
대체 흐름	<p><u>5-1 회원이 날짜와 시간중 하나라도 선택하지 않을 경우</u></p> <p>1. 시스템이 어쩌구 저쩌구 메세지를 출력한다.</p> <p>2. 회원은 선택하지 않았던 부분을 선택후 '결제'를 선택한다.</p> <p>[이하 기본 흐름 6부터 시작]</p>



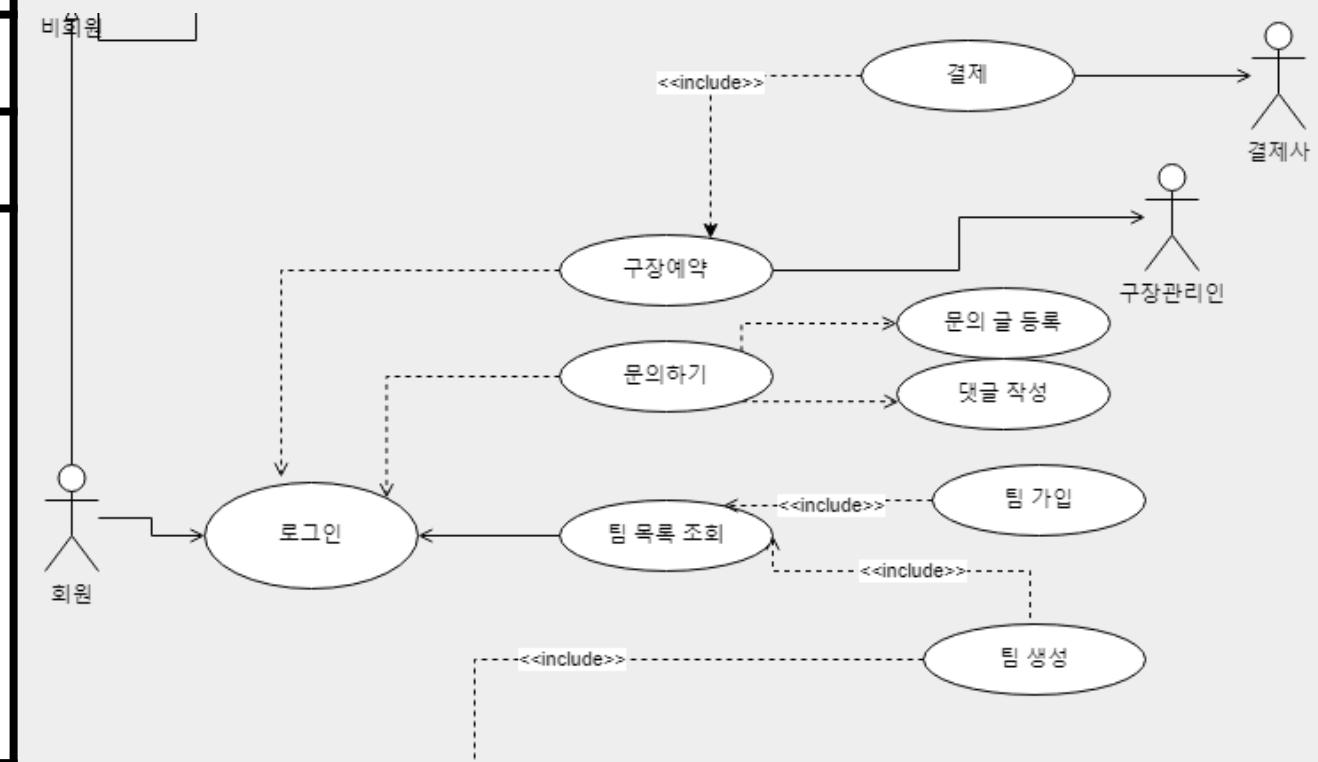
USECASE 명세서

유스케이스명	팀 가입
액터명	회원, 감독
개요	회원은 선택한 팀의 정보로 원하는 팀 가입신청 후 승인절차를 통해 팀 가입
사전조건	회원이 로그인 상태여야 한다.
사후조건	감독의 가입승인을 받아 팀에 가입이 되고 회원데이터에 클럽데이터 연결
기본 흐름	<p>1.로그인 한 회원은 메인화면에서 '팀 목록'을 선택</p> <p>2.시스템은 서버에 저장되어있는 팀 목록 화면을 제공</p> <p>3.회원은 원하는 팀의 '가입신청' 버튼 선택</p> <p>4.시스템은 현재 로그인되어있는 회원의 정보를 가입신청 데이터로 저장 클럽의 감독에게 제공</p> <p>5.감독은 제공받은 데이터 확인 후 승인</p> <p>6.시스템은 승인받은 회원의 데이터에 신청한 클럽데이터 연결</p>
대체 흐름	<p><u>5-1 감독이 승인거절 했을 경우</u></p> <p>1.시스템은 신청한 회원의 정보를 가입신청 데이터에서 삭제</p> <p>2.회원은 로그인 후 '내 팀보기'를 선택</p> <p>3.시스템은 연결될 팀의 정보가 없음을 경고</p> <p>[흐름 종료]</p>



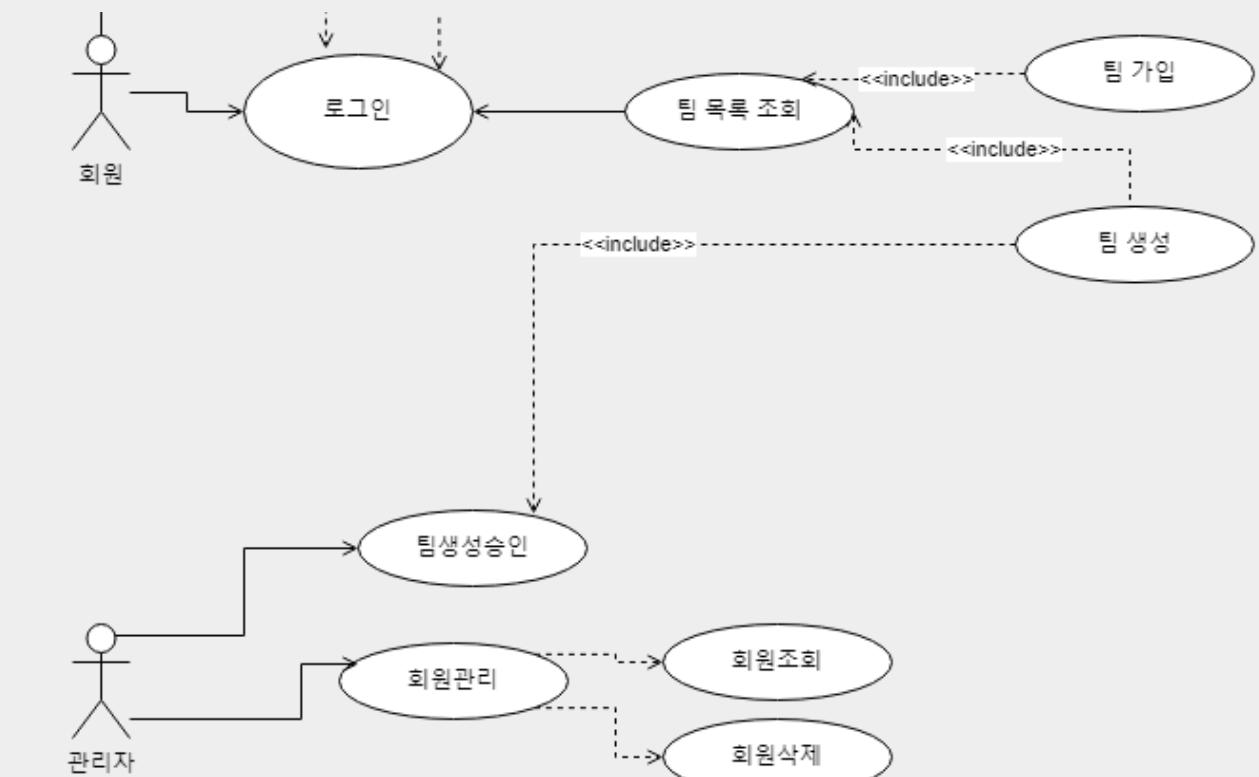
USECASE 명세서

유스케이스명	팀 생성
액터명	회원, 관리자
개요	회원은 생성하려는 팀의 정보를 입력해 팀 생성 신청 후 승인 절차를 거친 후 감독역할을 부여받고 팀 생성
사전조건	회원이 로그인 상태여야 한다.
사후조건	관리자의 팀 생성승인을 받아 팀이 생성되고, 생성 신청한 사람은 감독역할 부여
기본 흐름	<p>1.로그인 한 회원은 메인화면에서 '팀목록'을 선택 2.시스템은 서버에 저장되었는 팀 목록 화면을 제공 3.회원은 화면에서 '팀 생성하기' 선택 4.시스템은 팀 생성 시 필요한 정보를 입력하는 양식을 제공 5.회원은 자신이 생성할 팀에 대한 정보 입력 6.시스템은 입력된 정보 확인 후 대기상태로 작성한 회원의 데이터와 함께 저장, 제공 7.관리자는 관리자페이지에 대기상태인 팀리스트 확인 및 승인 8.시스템은 팀을 승인상태로 변경 및 작성한 회원의 이메일정보로 생성완료알림, 감독역할 부여</p>
대체 흐름	<p>5-1 회원이 필수 입력 조건을 다 입력하지 않을 경우</p> <ol style="list-style-type: none"> 1.시스템이 입력하지 않은 부분을 안내한다. 2. 사용자는 입력하지 않았던 부분의 내용을 작성한다. <p>[이하 기본 흐름 6부터 시작]</p> <p>7-1 관리자가 승인하지 않을 경우</p> <ol style="list-style-type: none"> 1. 시스템은 대기상태의 팀 데이터를 삭제 2. 신청한 회원이 메인의 '내팀보기'를 선택 3. 시스템은 회원의 팀이 존재하지 않는다는 안내창 제공 4. 회원은 안내창 확인 <p>[흐름종료]</p>



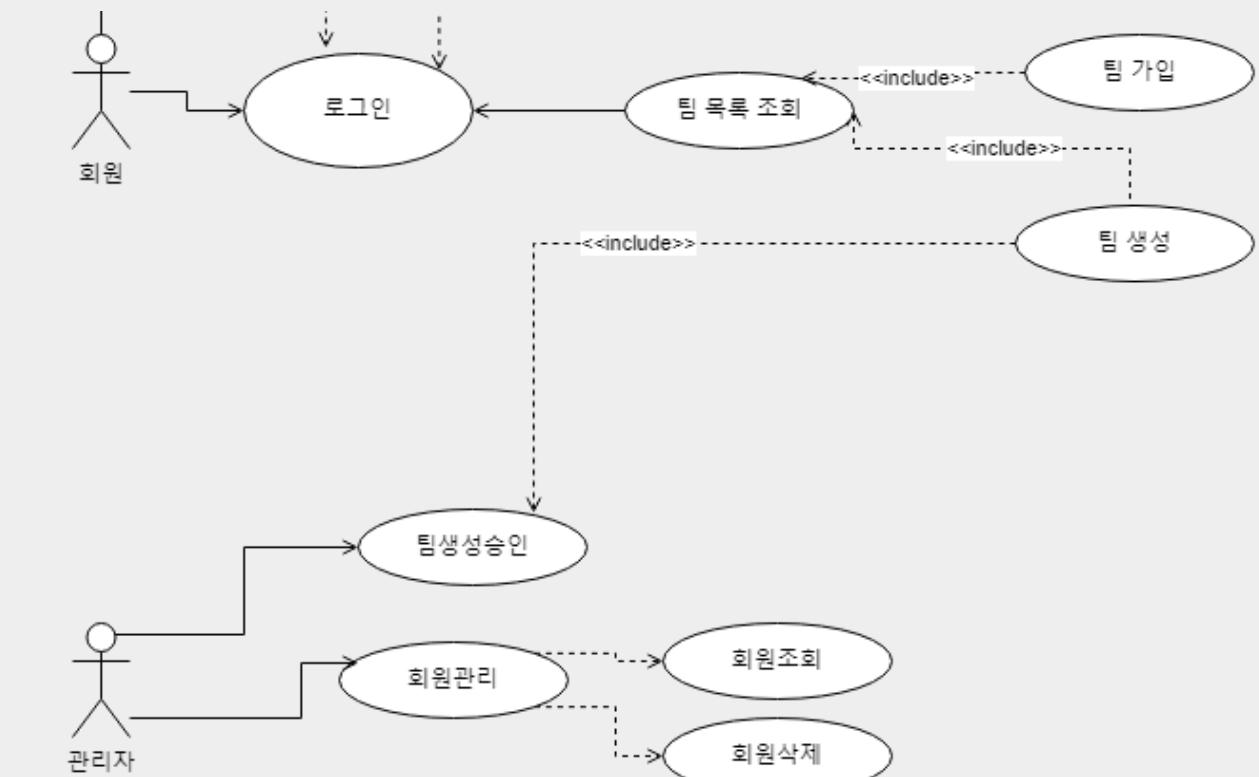
USECASE 명세서

유스케이스명	팀 생성 승인
액터명	관리자
개요	관리자는 회원이 만든 승인대기상태의 클럽을 승인해 승인상태로 변경
사전조건	관리자 권한으로 로그인 되어 있어야 한다.
사후조건	회원이 메인에서 '내 팀보기' 선택 시 승인대기상태 안내창이 뜨지 않아야 함
기본 흐름	<ol style="list-style-type: none"> 1. 관리자는 메인의 '관리자'를 선택 2. 시스템은 '관리자' 페이지 제공 3. 관리자는 페이지에서 '팀승인' 선택 4. 시스템은 승인대기상태의 팀목록 제공 5. 관리자는 팀의 '승인'을 선택 6. 시스템은 관리자가 '승인' 한 팀의 승인대기상태 데이터를 승인상태로 변경
대체 흐름	<p>5-1 관리자가 팀의 '거절'을 선택</p> <ol style="list-style-type: none"> 1. 시스템은 관리자가 '거절' 한 팀의 데이터를 서버에서 삭제 2. 팀 승인신청한 회원은 '내 팀보기' 선택 3. 시스템은 회원의 클럽정보가 없으므로 클럽정보가 없다는 것을 안내 <p>[흐름 종료]</p>



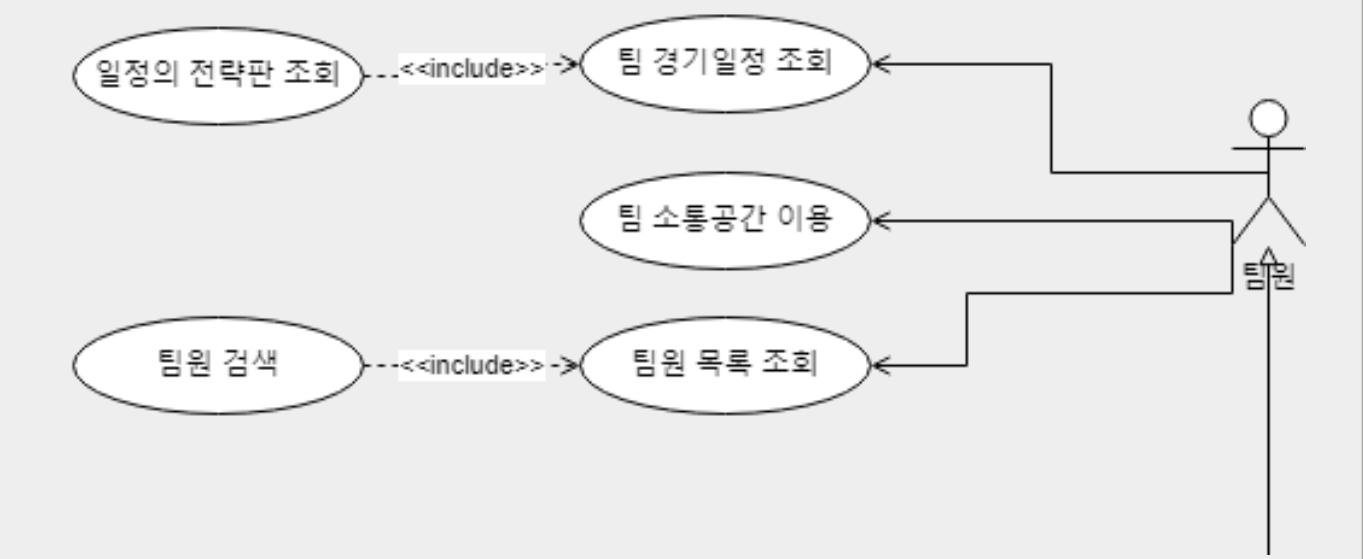
USECASE 명세서

유스케이스명	회원관리
액터명	관리자
개요	관리자는 사이트에 가입된 회원의 정보 조회, 삭제가 가능
사전조건	관리자 권한으로 로그인 되어 있어야 한다.
사후조건	회원 삭제 시 가입되어있는 팀의 팀원목록에서도 삭제 되어야한다.
기본 흐름	<ol style="list-style-type: none"> 1. 관리자는 메인의 '관리자'를 선택 2. 시스템은 '관리자' 페이지 제공 3. 관리자는 페이지에서 '회원목록' 선택 4. 시스템은 사이트에 가입되어있는 회원의 정보를 목록으로 제공 5. 관리자는 가입되어있는 회원의 '탈퇴' 선택 6. 시스템은 관리자가 '탈퇴' 선택한 회원 데이터를 서버에서 삭제
대체 흐름	<p>5-1 이미 탈퇴한 회원일 경우</p> <ol style="list-style-type: none"> 1. 시스템은 창을 새로고침 후 없어진 회원이 반영된 목록을 제공한다. <p>[흐름 종료]</p>



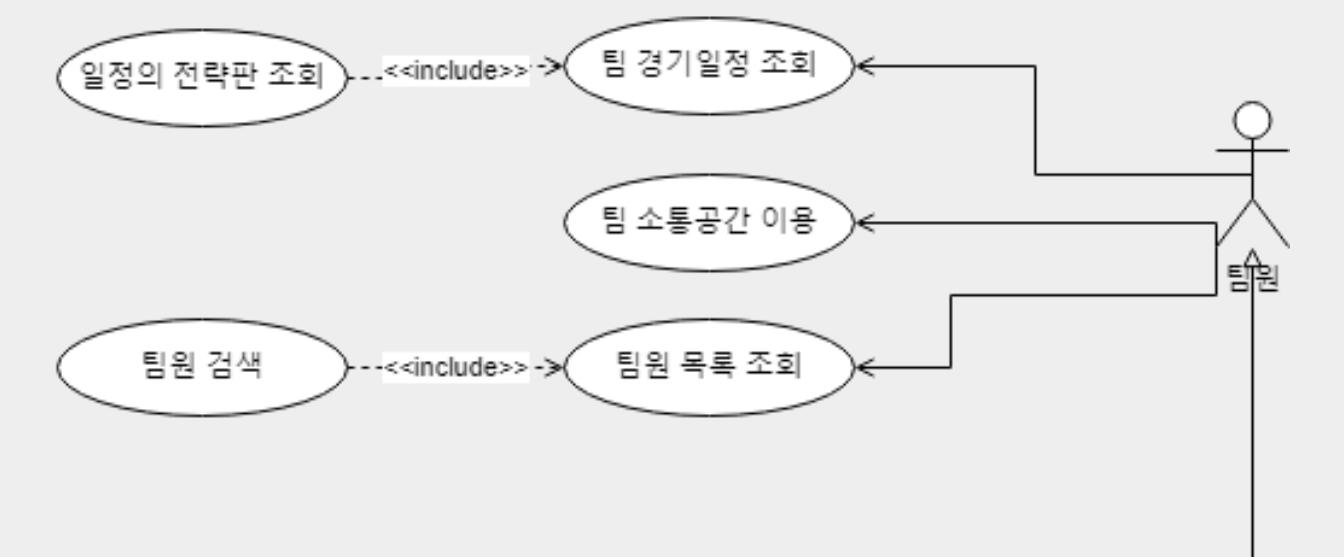
USECASE 명세서

유스케이스명	팀 경기 일정 조회
액터명	팀원
개요	팀원 데이터에 연결되어있는 클럽데이터로 클럽의 경기일정 정보를 조회한다.
사전조건	로그인한 회원에게 팀의 데이터 연결되어 있어야 한다.
사후조건	단순 조회 기능이라 없음
기본 흐름	<p>1.로그인한 팀원은 메인화면에서 '내팀보기'를 선택한다</p> <p>2.시스템은 팀원의 '내팀보기' 페이지를 제공</p> <p>3.팀원은 페이지에 출력되어있는 간략한 경기일정들을 확인하고 원하는 일정의 '상세보기'를 선택한다</p> <p>4.시스템은 팀원이 선택한 경기일정에 대한 상세정보와 전략판이미지를 제공한다</p>
대체 흐름	<p><u>3-1 클럽 현재 달의 경기일정이 없을경우</u></p> <p>1. 시스템은 경기일정이 출력되어있지 않은 달 페이지렌더링을 제공한다.</p> <p>[흐름종료]</p>



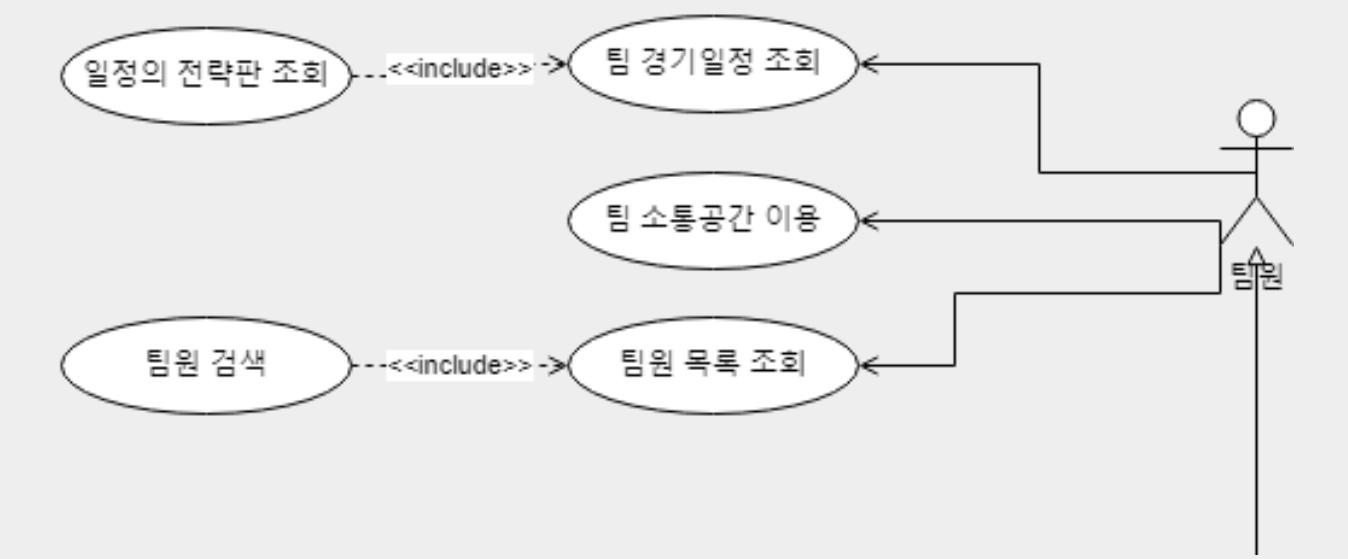
USECASE 명세서

유스케이스명	팀 소통공간 이용
액터명	팀원
개요	팀원은 팀원들이 작성한 소통공간의 내용, 작성자, 작성일자를 확인 및 작성
사전조건	로그인한 회원에게 팀의 데이터 연결되어 있어야 한다.
사후조건	팀원이 작성한 소통공간 데이터가 저장 및 출력되어 있어야 한다.
기본 흐름	<p>1. 로그인한 팀원은 메인화면에서 '내팀보기'를 선택</p> <p>2. 시스템은 팀원의 '내팀보기' 페이지를 제공</p> <p>3. 팀원은 페이지에 출력되어 있는 소통공간 내용, 작성자, 작성일자들을 확인하고 자신이 입력하고 싶은 내용을 입력 후 '등록'을 선택</p> <p>4. 시스템은 입력한 팀원의 이름, 내용, 날짜, 속한 클럽데이터를 서버에 저장 후 화면에 제공</p>
대체 흐름	<p>4-1 다른 팀원도 소통공간에 작성한 경우</p> <p>1. 시스템은 서버에 저장된 해당클럽의 소통공간 데이터를 시간순으로 정렬해 제공한다.</p> <p>[흐름종료]</p>



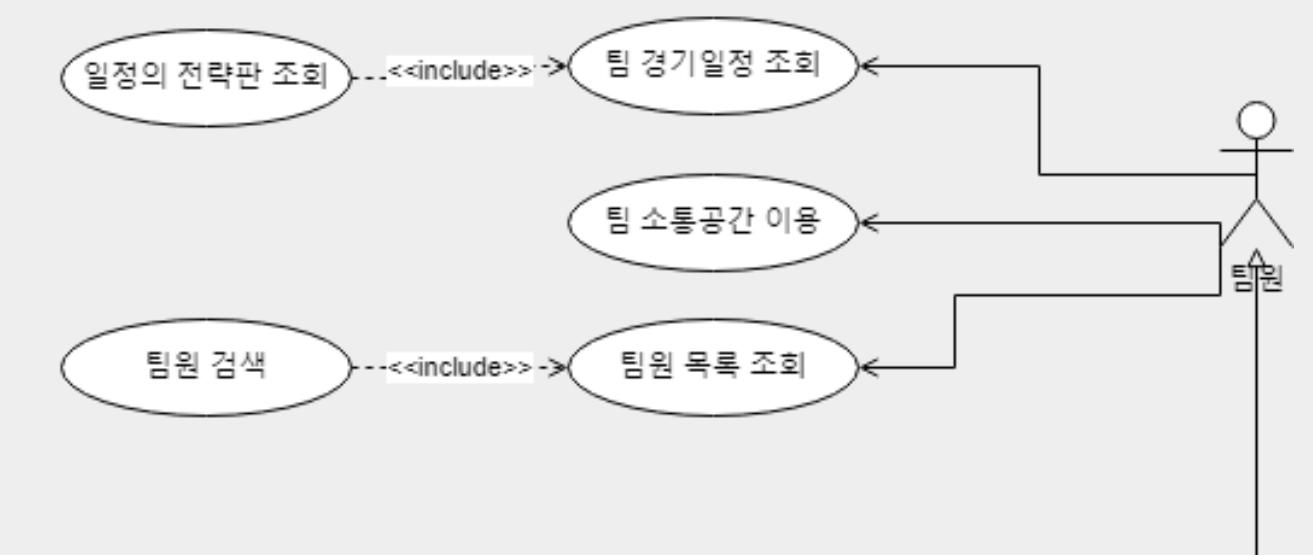
USECASE 명세서

유스케이스명	팀원 목록 조회
액터명	팀원
개요	팀원은 팀원들의 목록을 조회, 검색 가능
사전조건	로그인한 회원에게 팀의 데이터 연결되어 있어야 한다.
사후조건	페이지를 새로 로드하면 검색조건이 아닌 전체 팀원목록이어야 한다.
기본 흐름	<ol style="list-style-type: none"> 1.로그인한 팀원은 메인화면에서 '내팀보기'를 선택 2.시스템은 팀원의 '내팀보기' 페이지를 제공 3.팀원은 페이지에 출력된 팀원 목록을 조회 및 입력칸에 최소 이름의 한 자를 입력하고 '검색' 선택 4.시스템은 팀원이 입력한 내용을 포함하고 있는 이름의 팀원목록 제공
대체 흐름	<p><u>4-1 팀에 없는 팀원의 이름을 검색할 경우</u></p> <ol style="list-style-type: none"> 1. 시스템은 빈 목록을 제공한다. [흐름 종료]



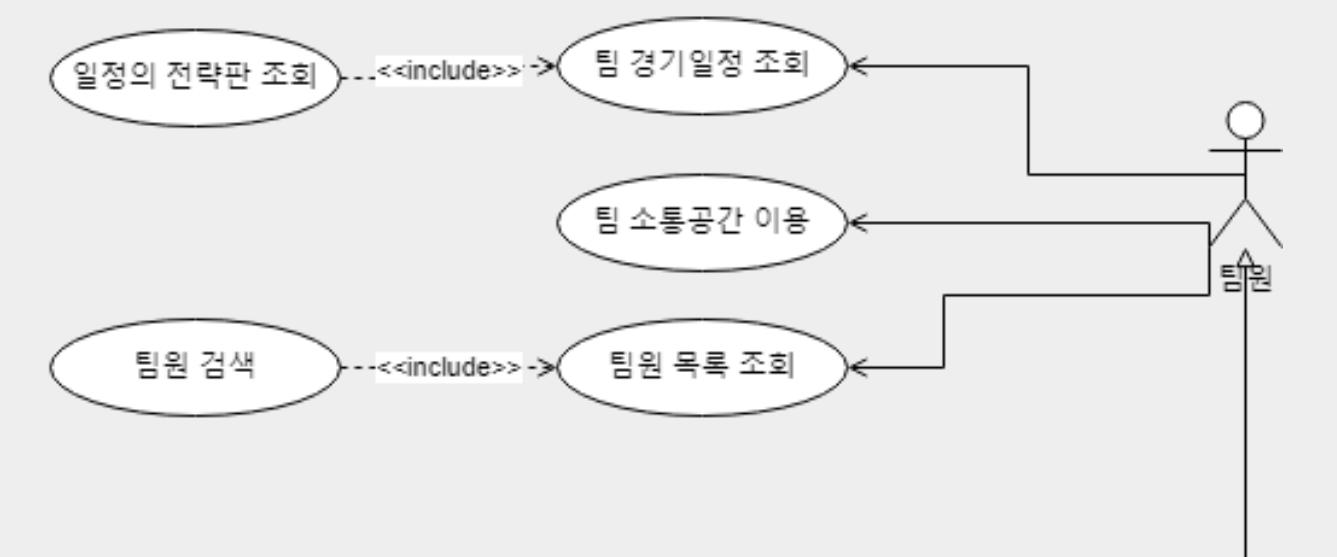
USECASE 명세서

유스케이스명	팀 경기일정 등록
액터명	감독
개요	감독은 날짜와 시간, 전달사항을 입력하고 전력판을 제작해 이루어진 경기일정 정보를 등록한다.
사전조건	회원의 역할이 감독이어야 한다.
사후조건	등록한 일정정보가 서버에 저장되어 '내 팀보기' 페이지에 출력
기본 흐름	<ol style="list-style-type: none"> 1.로그인한 감독은 메인화면에서 '내팀보기' 를 선택 2.시스템은 '내팀보기' 페이지를 제공 3.감독은 페이지의 일정 출력되는 부분 상단의 '+ADD' 를 선택 4.시스템은 일시와 전달사항을 입력하는 부분과 전력판 제작을 위한 제작판, 클럽원목록, 프리셋을 제공 5.감독은 일정에 대한 정보를 입력하고 전력판을 제작 후 '등록'을 누른다. 6.시스템은 감독이 입력한 정보와 전력판을 이미지로 서버에 일정데이터로 저장한다
대체 흐름	<p>5-1 프리셋 저장 후 등록할 경우</p> <ol style="list-style-type: none"> 1. 감독은 원하는 프리셋 선택 후 '저장' 선택 2. 시스템은 감독이 선택한 프리셋에 현재 상태의 전력판 저장 <p>[이하 기본흐름 5부터 시작]</p> <p>5-2 이전 프리셋으로 불러오기할 경우</p> <ol style="list-style-type: none"> 1. 감독은 저장된 프리셋을 불러오기 위한 '불러오기' 선택 2. 시스템은 저장되어있는 전력판데이터를 가져와 제공 <p>[이하 기본흐름 5부터 시작]</p>



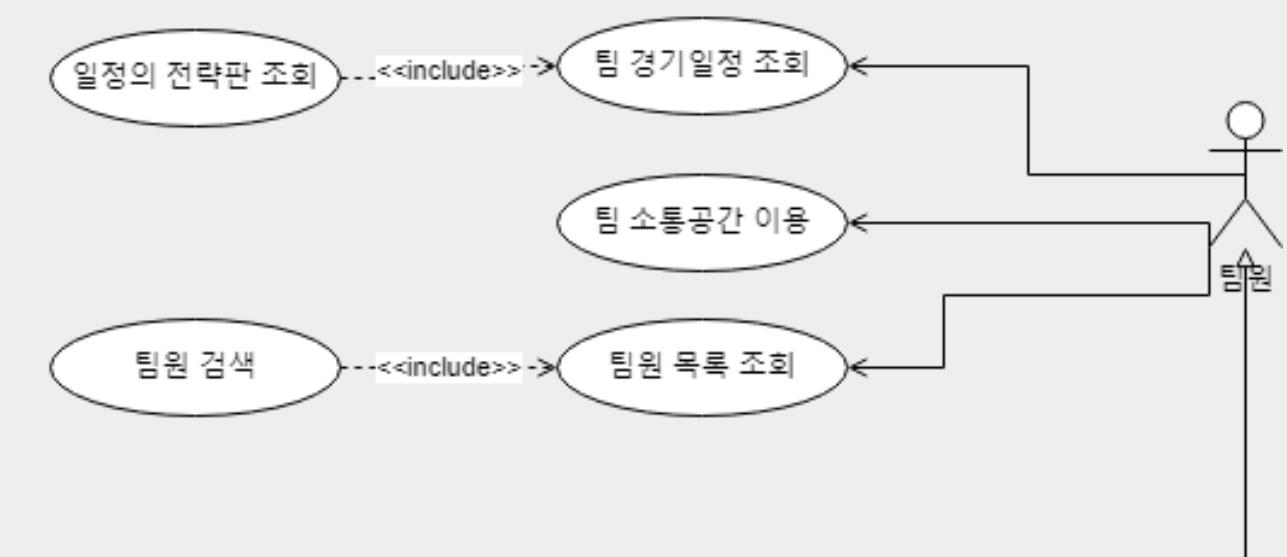
USECASE 명세서

유스케이스명	팀 소통공간 관리
액터명	감독
개요	팀원이 작성한 팀 소통공간의 글을 삭제가능
사전조건	로그인한 회원의 역할이 감독이어야한다.
사후조건	서버 데이터 상에서도 소통공간 글이 삭제 되어야한다.
기본 흐름	<ol style="list-style-type: none"> 1.로그인한 감독은 메인화면에서 '내팀보기'를 선택 2.시스템은 '내팀보기' 페이지를 제공 3.감독은 소통공간의 원하는 내용의 글의 'x' 선택 4.시스템은 이 글을 정말 삭제할건지 한번 더 질문하는 경고창 제공 5.감독은 경고창에서 '확인'을 선택 6.시스템은 감독이 선택한 소통공간의 글을 서버데이터에서 삭제
대체 흐름	<p>5-1 감독이 확인이 아닌 취소를 선택한 경우</p> <ol style="list-style-type: none"> 1. 시스템의 삭제 시스템은 실행 취소 <p>[흐름 종료]</p>



USECASE 명세서

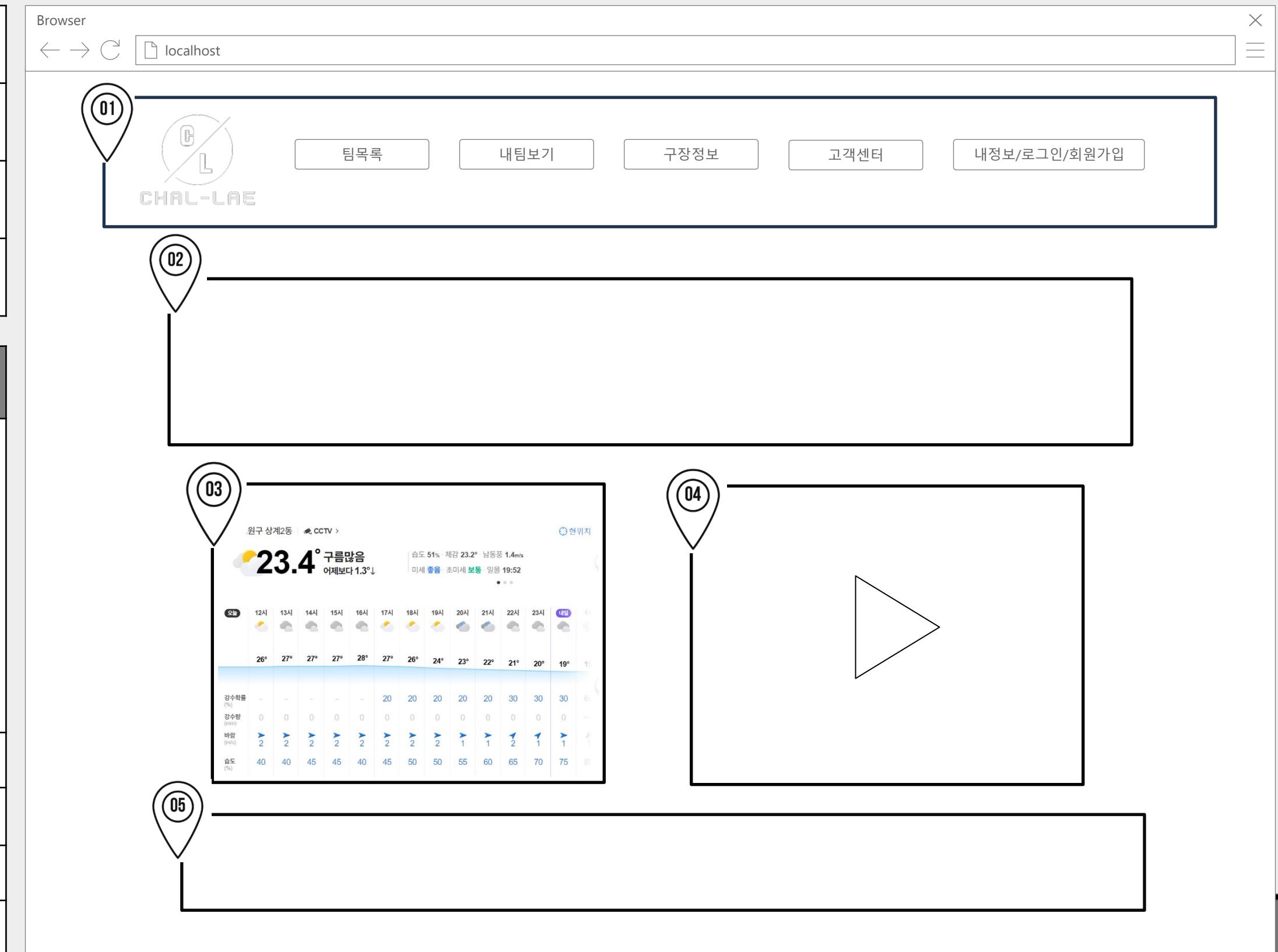
유스케이스명	팀원관리
액터명	감독
개요	다른 회원이 한 가입신청 승인 혹은 거절 가능, 팀원을 팀에서 방출 가능
사전조건	로그인한 회원의 역할이 감독이어야 한다.
사후조건	승인한 팀원의 정보가 등록, 방출한 팀원의 정보가 삭제되어 있어야 한다.
기본 흐름	<ol style="list-style-type: none"> 1. 로그인한 감독은 메인화면에서 '내팀보기'를 선택 2. 시스템은 감독의 '내팀보기' 페이지를 제공 3. 감독은 팀원을 승인, 방출
	<ol style="list-style-type: none"> 1. 감독은 '내팀보기' 페이지에서 '가입신청'을 선택 2. 시스템은 팀에 해당하는 가입신청리스트를 제공 3. 감독은 가입신청리스트를 확인 및 승인 4. 시스템은 가입신청한 회원의 데이터에 클럽데이터 연결
	<ol style="list-style-type: none"> 1. 감독은 '내팀보기' 페이지의 팀원목록에서 삭제할 회원의 '강퇴'를 선택 2. 시스템은 삭제를 진행 할 것인지 알림창을 다시한번 경고 3. 감독은 알림창 확인 및, '확인' 선택 4. 시스템은 선택된 회원의 데이터에서 해당 클럽의 데이터를 연결해제
대체 흐름	<p><u>방출 3-1</u> 감독이 알림창의 취소를 누른 경우</p> <ol style="list-style-type: none"> 1. 시스템은 삭제기능을 중단한다 <p>[흐름 종료]</p>



스토리보드

Project	CHAL-LAE
Page Title	메인 홈 페이지
File Path	/
File Name	Index.html

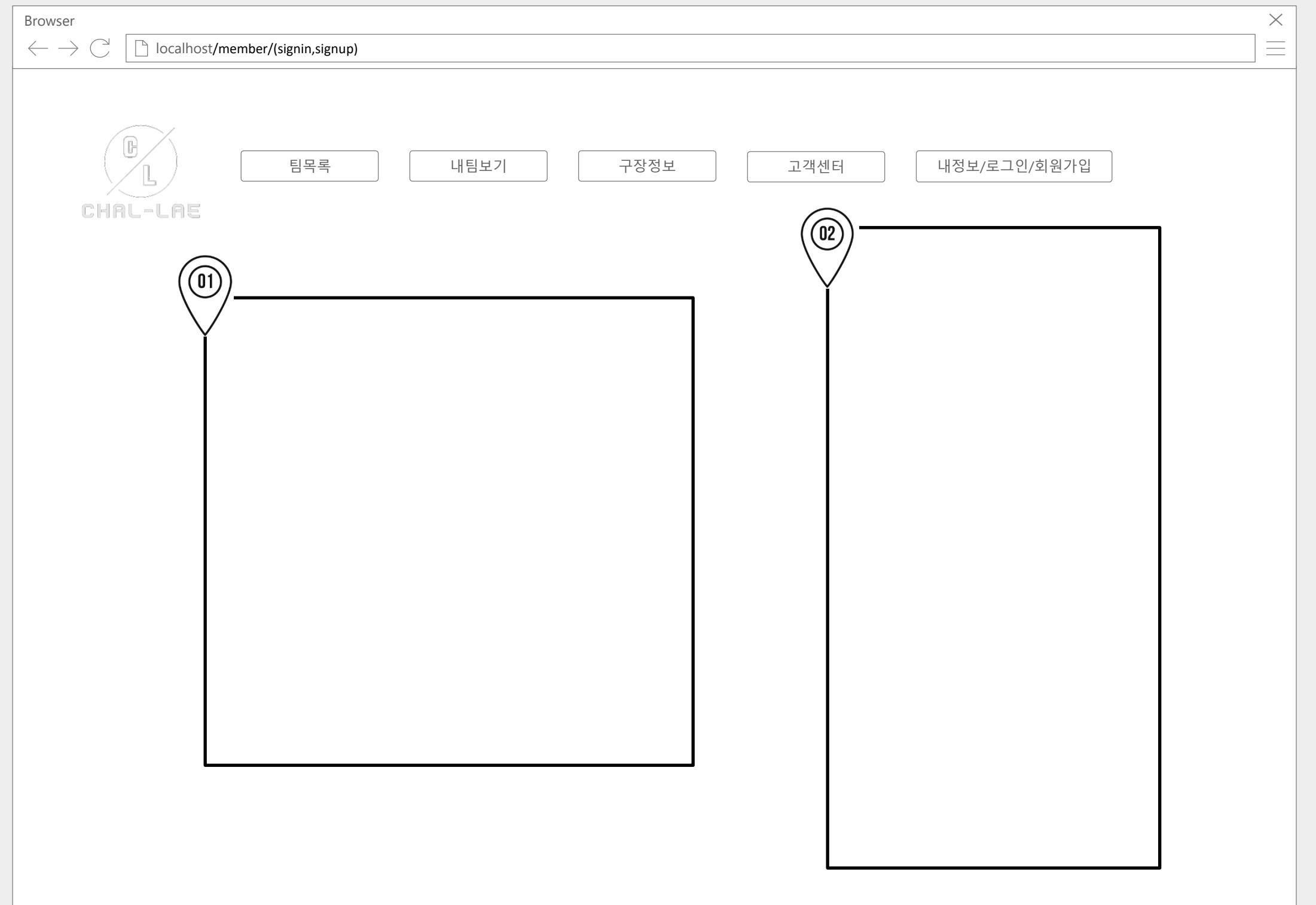
화면 설명	
1	홈 화면 이동, 팀 목록, 내 팀보기, 구장정보, 고객센터 버튼이 있는 헤더 로그인과 회원가입 기능, 모달로 나오는 창을 이용해 로그인, 회원가입을 할 수 있다.
2	메인 롤링 형식의 광고배너
3	날씨 API 이용해 출력되는 오늘의 날씨
4	사이트 메인 동영상 출력되는 부분
5	FOOTER 내용



스토리보드

Project	CHAL-LAE
Page Title	로그인, 회원가입, 내 정보
File Path	/member/(sign in, sign up)
File Name	(signin,signupForm,signupResult).html

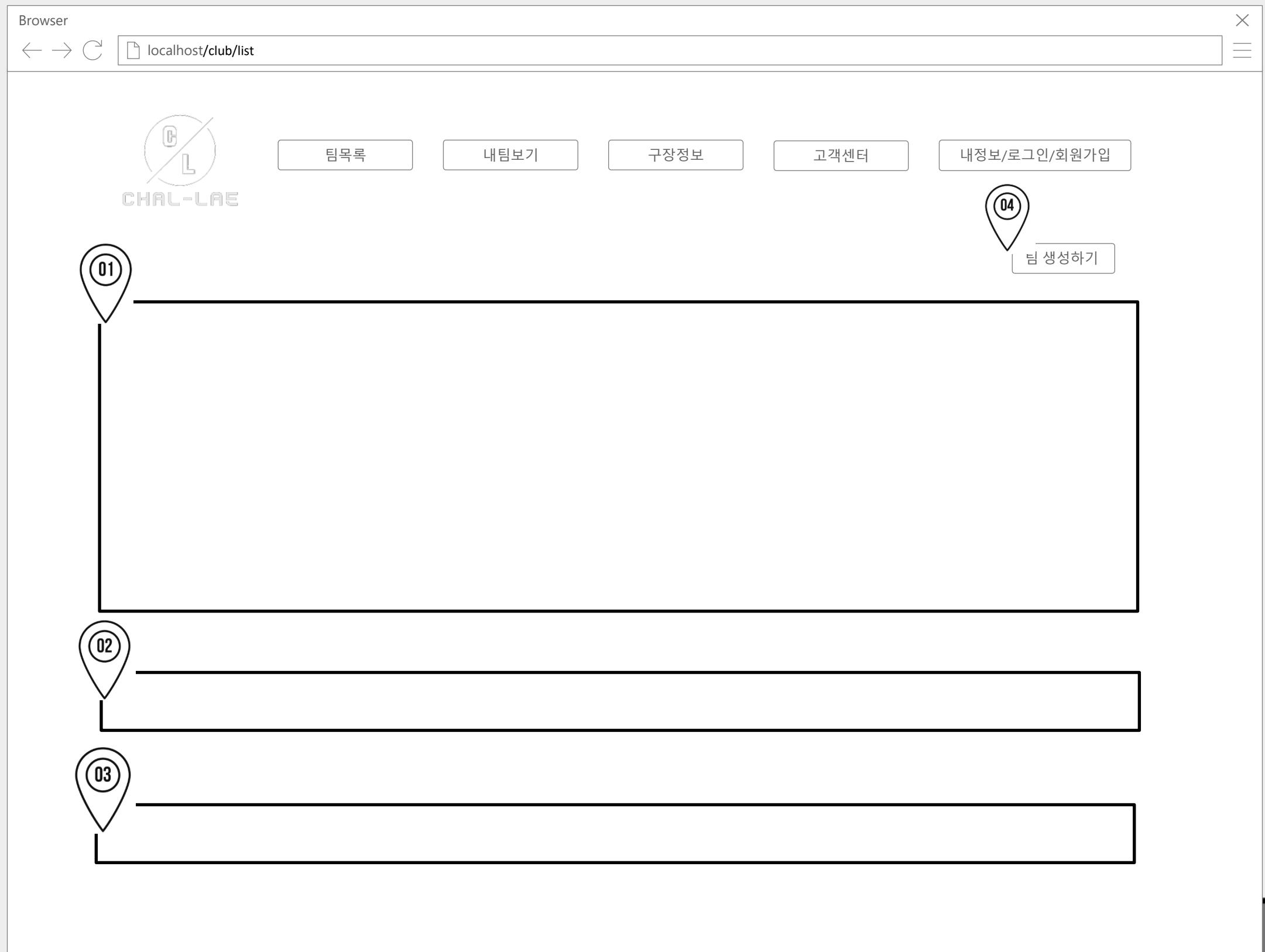
화면 설명	
1	로그인과 회원가입 기능, 모달로 나오는 창을 이용해 로그인, 회원가입을 할 수 있다.
2	로그인과 같은 모달 창으로 내 정보와 감독회원은 가입승인을 받을 수 있음



스토리보드

Project	CHAL-LAE
Page Title	팀 목록
File Path	/club/list
File Name	clublist.html

화면 설명	
1	팀의 간략한 소개가 리스트 형식으로 있으며, 클릭 시 팀의 상세정보가 모달 창으로 출력
2	특정 키워드로 선택해 검색할 수 있는 부분
3	팀 리스트 정보를 페이징 처리
4	팀생성하기 버튼을 통해 팀을 생성 할 수 있는 기능

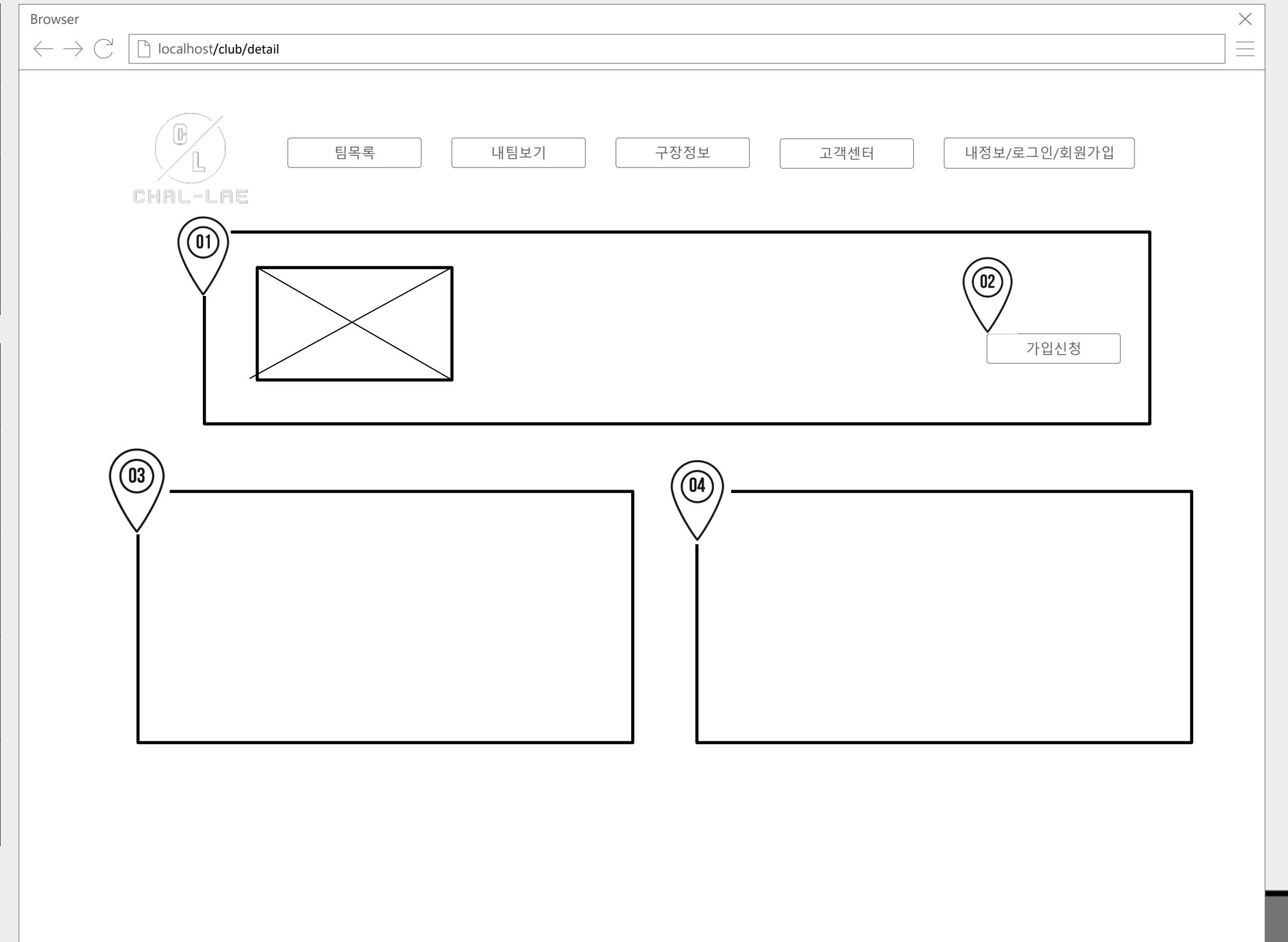


스토리보드

Project	CHAL-LAE
Page Title	팀 상세보기
File Path	/club/detail
File Name	clubdetail.html

화면 설명

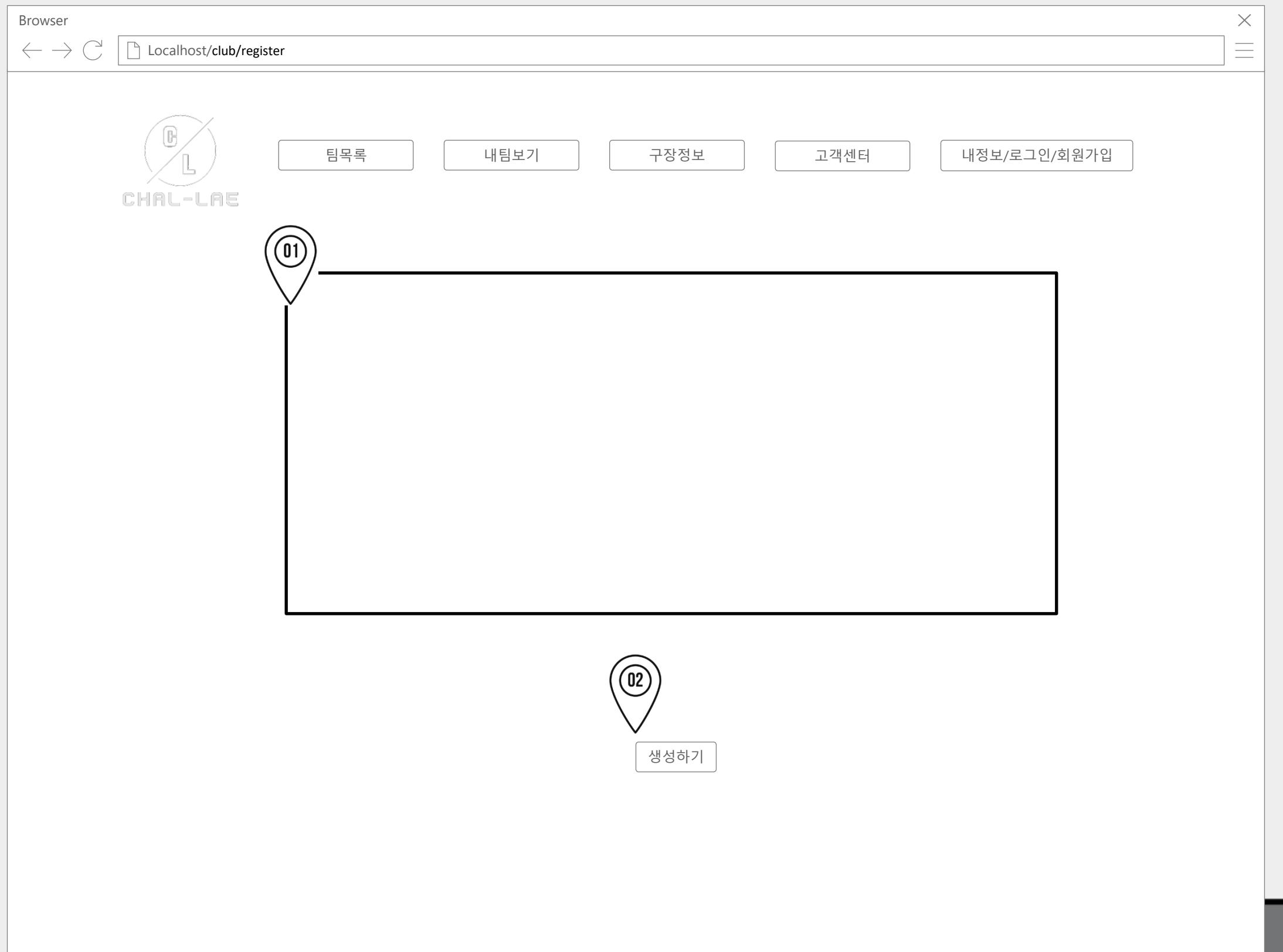
- 1 팀의 간단한 설명과 로고가 출력
- 2 클릭 시 가입신청을 위한 여러가지 정보를 입력하는 모달 창 출력
- 3 멤버 리스트를 출력
- 4 최근활동에 대한 기록이 리스트로 출력



스토리보드

Project	CHAL-LAE
Page Title	팀 생성하기
File Path	/club/register
File Name	clubregister.html

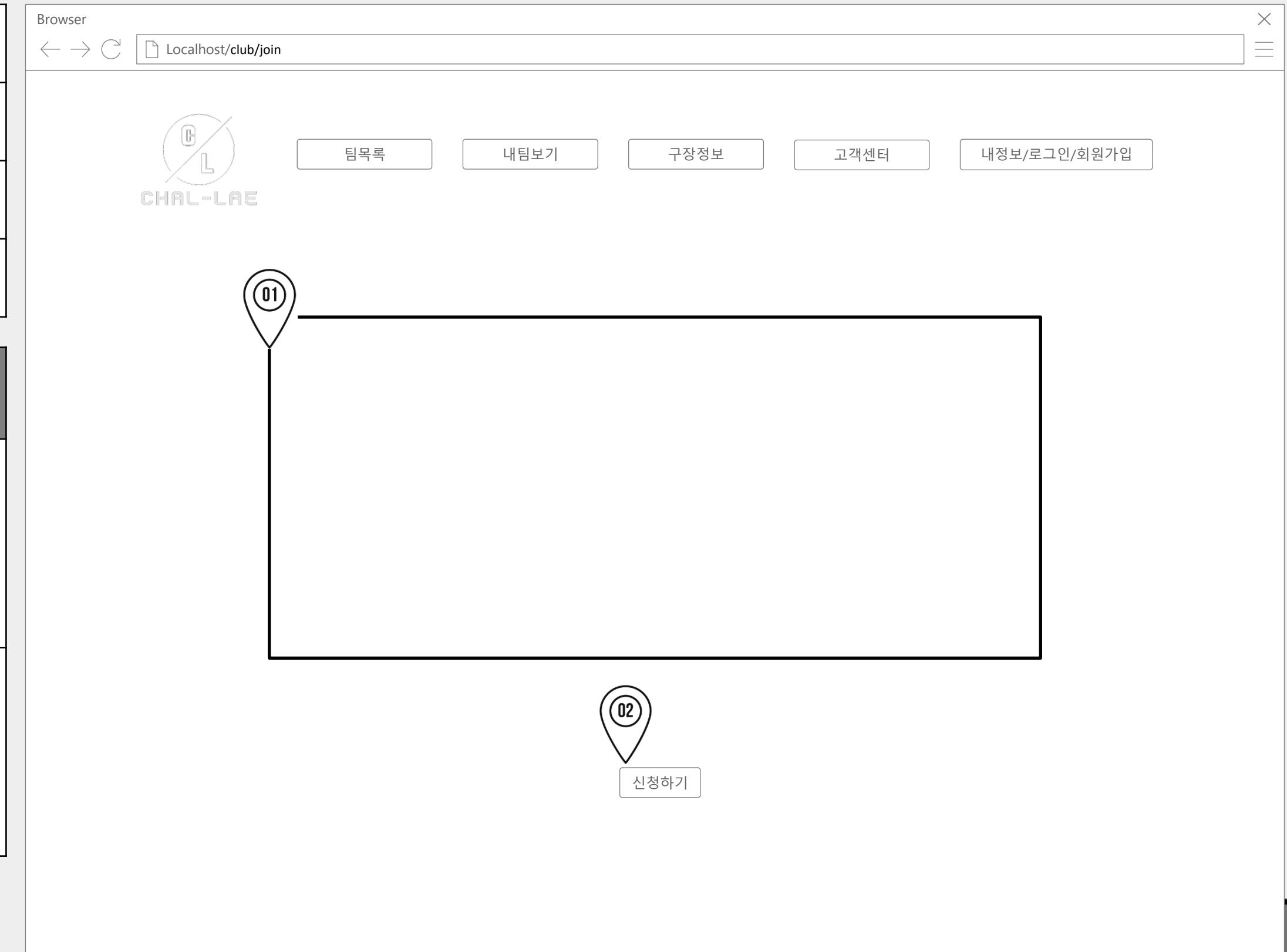
화면 설명	
1	팀의 상세정보(이름, 수준, 활동지역, 대표자 번호, 소개 글, 로그 이미지 등등) 기입
2	팀 신청하기 버튼과 팀 목록으로 되돌아갈 수 있는 버튼



스토리보드

Project	CHAL-LAE
Page Title	팀 가입하기
File Path	/club/join
File Name	clubjoin.html

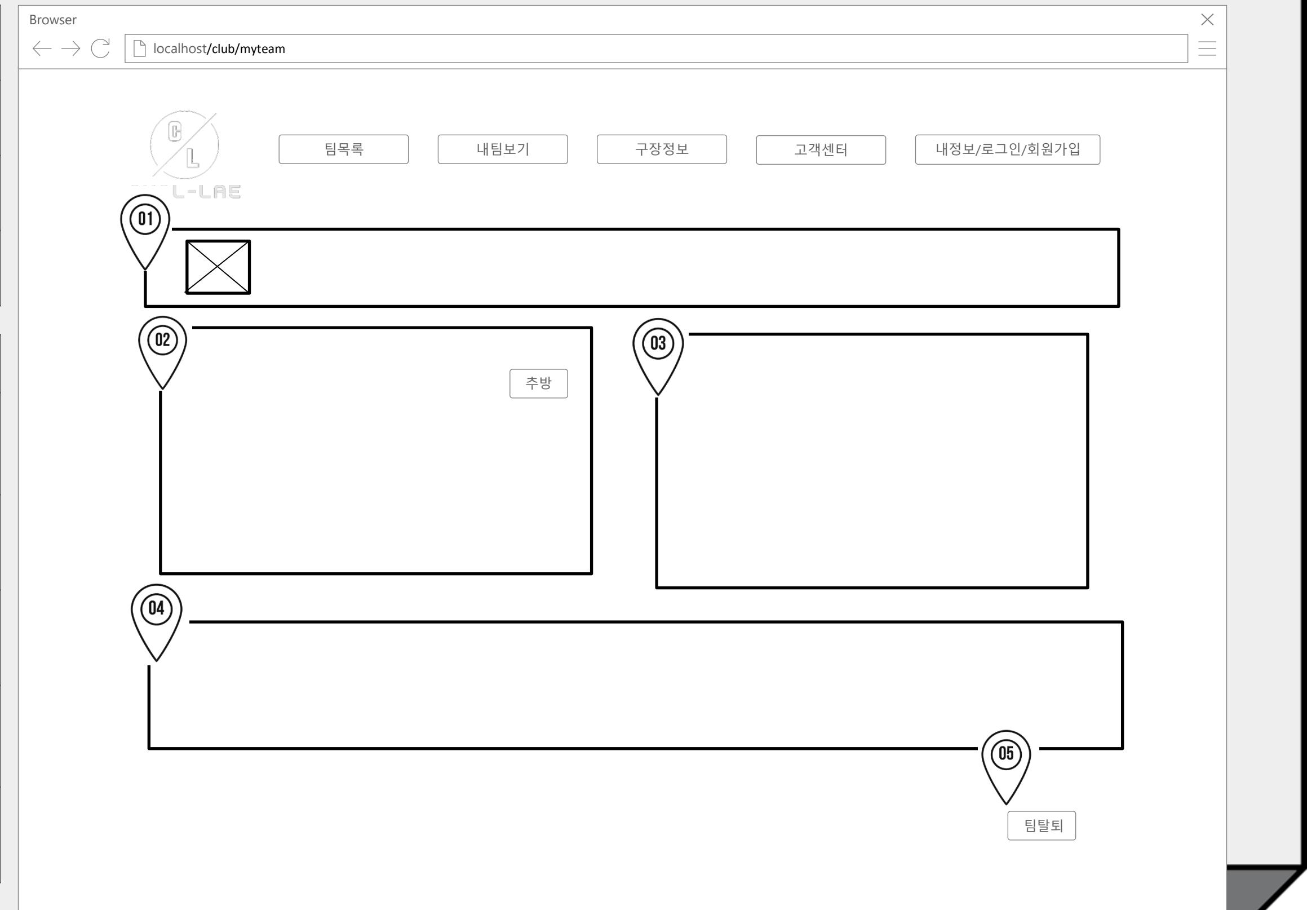
화면 설명	
1	팀 가입신청에 필요한 정보를 기입
2	팀에 가입을 신청할 수 있는 버튼



스토리보드

Project	CHAL-LAE
Page Title	내 팀보기
File Path	/club/myteam
File Name	myteam.html

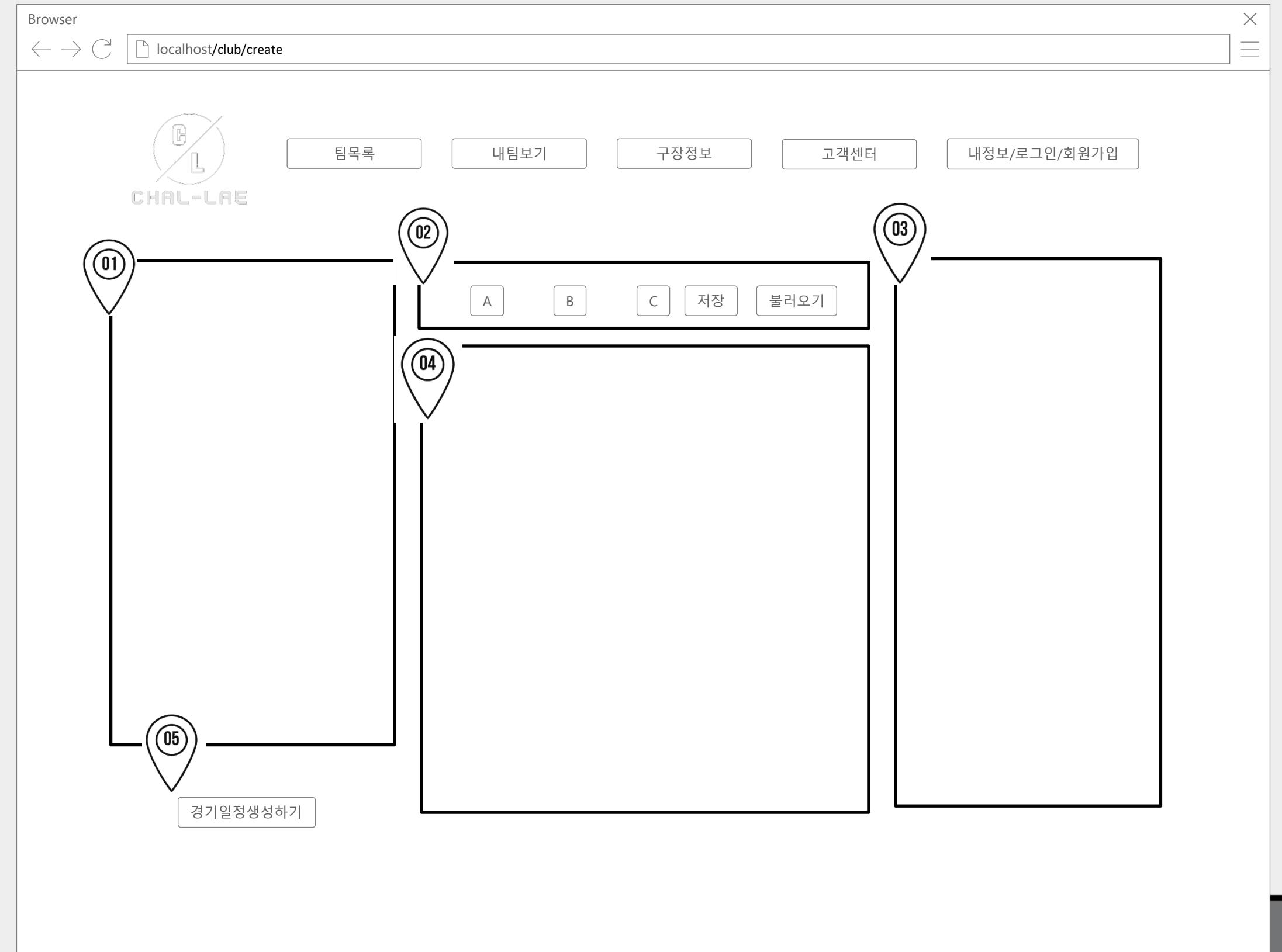
화면 설명	
1	DB에 등록된 팀의 이미지(로고), 이름과 정보, 팀의 인원 수가 출력됨
2	팀원 목록이 출력되는 부분, 회원 추방기능버튼
3	채팅을 이용한 소통공간
4	달마다 등록된 경기가 간단히 표시되는 부분 클릭 시 그 날의 경기정보로 이동함
5	팀을 탈퇴할 수 있는 탈퇴버튼



스토리보드

Project	CHAL-LAE
Page Title	경기등록
File Path	/club/create
File Name	createMatch.html

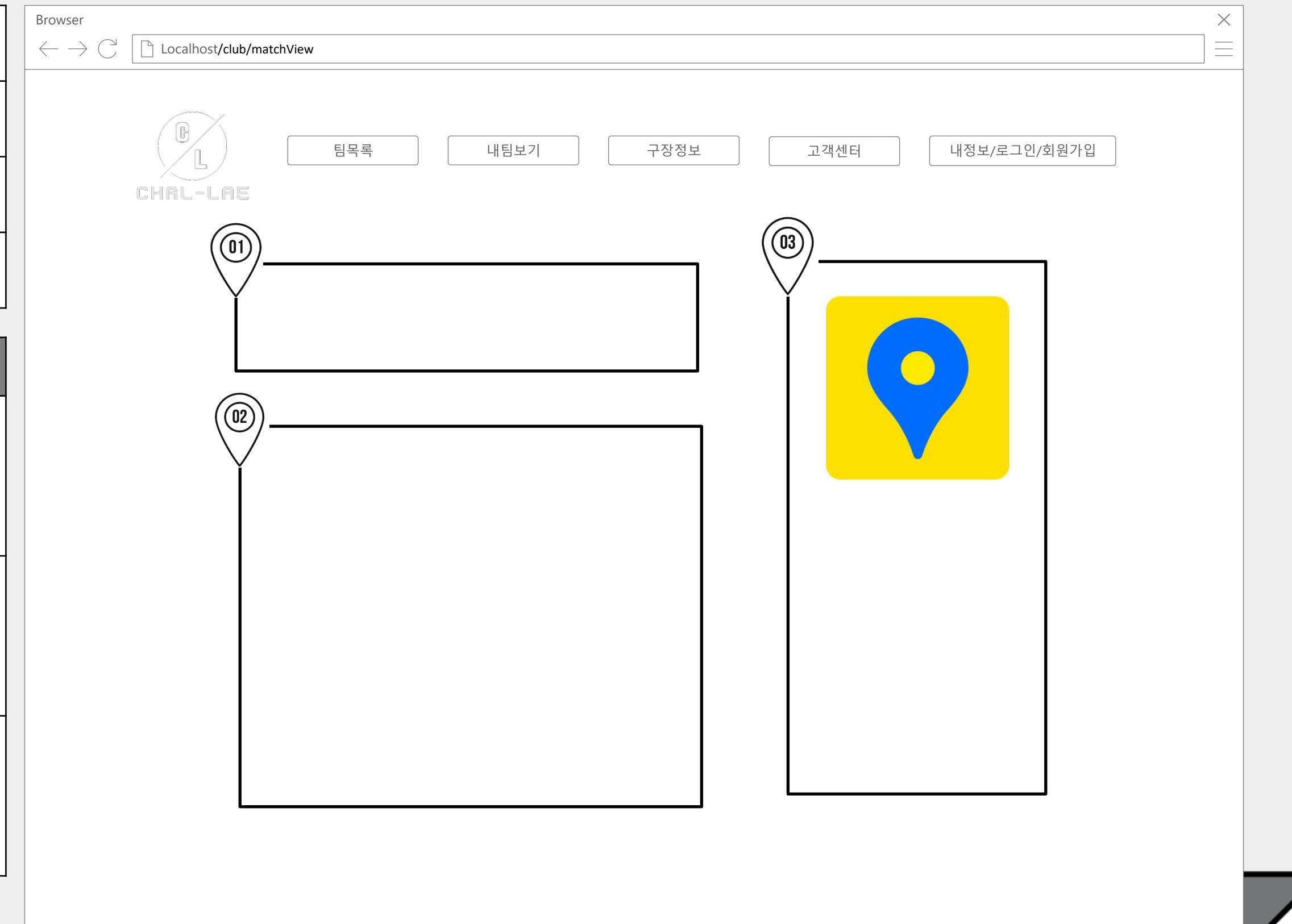
화면 설명	
1	등록 되어있는 구장을 선택, 경기 날짜와 시간을 기입할 수 있는 부분
2	a, b, c 프리셋 3개의 저장과 불러오기를 할 수 있는 버튼들 위치
3	팀원 목록이 출력되어 있으며 팀원을 드래그 해 위치 지정할 수 있는 부분
4	포메이션 설정이 가능한 전략판 출력
5	경기 데이터에 작성한 일자를 추가하는 경기일정 생성하기 버튼



스토리보드

Project	CHAL-LAE
Page Title	경기상세정보
File Path	/club/matchView
File Name	matchView.html

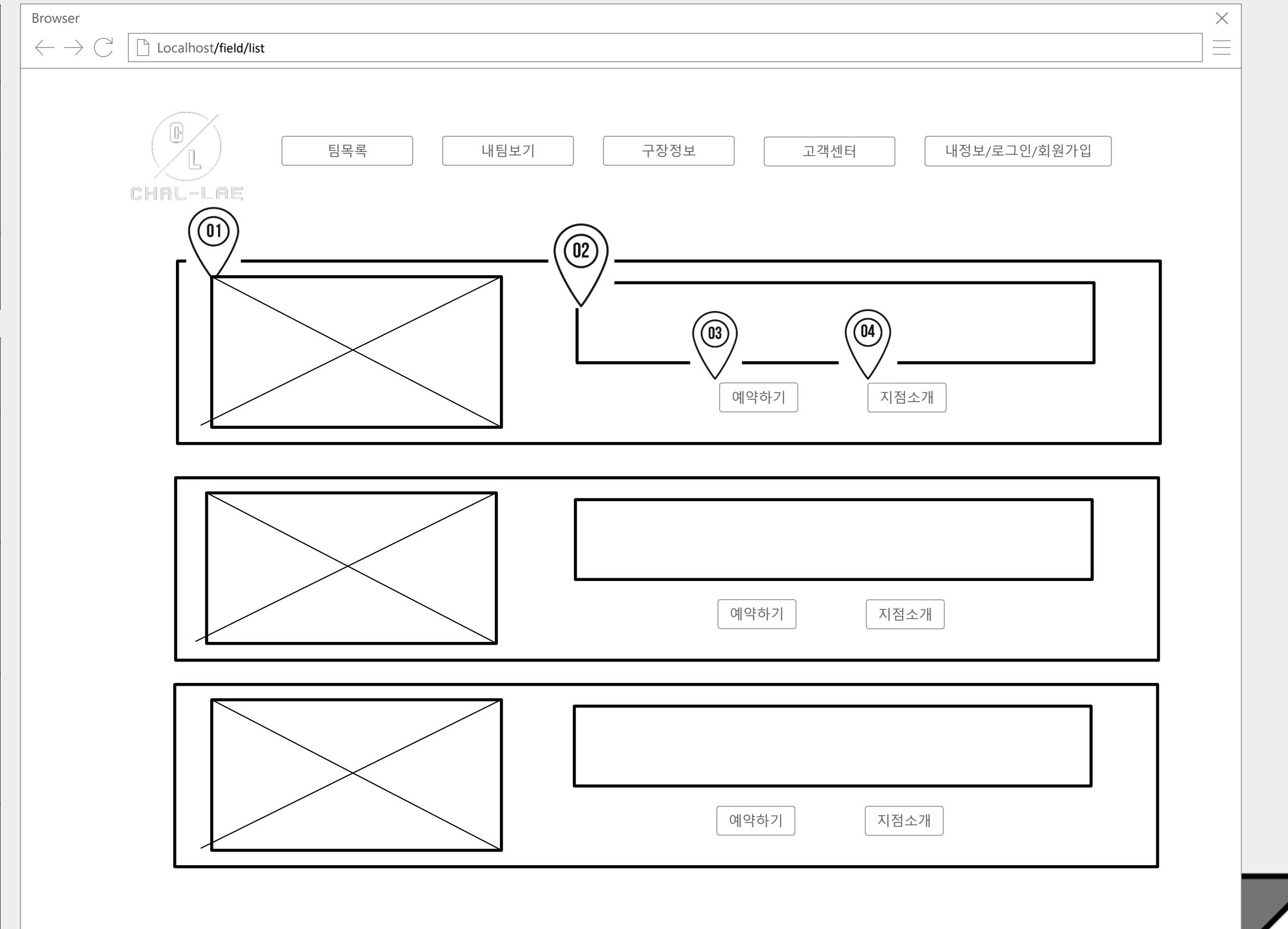
화면 설명	
1	감독이 등록한 일정의 정보가 출력되는 부분
2	감독이 등록한 전략판의 모습이 출력되는 부분
3	감독이 선택한 구장의 위치가 지도, 주소로 출력되는 부분



스토리보드

Project	CHAL-LAE
Page Title	구장리스트
File Path	/field/list
File Name	fieldlist.html

화면 설명	
1	데이터로 등록된 구장의 대표 이미지 출력부분
2	등록된 구장의 간단한정보들 출력부분
3	선택한 구장의 예약페이지로 이동함
4	클릭한 구장의 상세 정보로 이동함



스토리보드

Project	CHAL-LAE
Page Title	구장상세정보
File Path	/field/view
File Name	fieldview.html

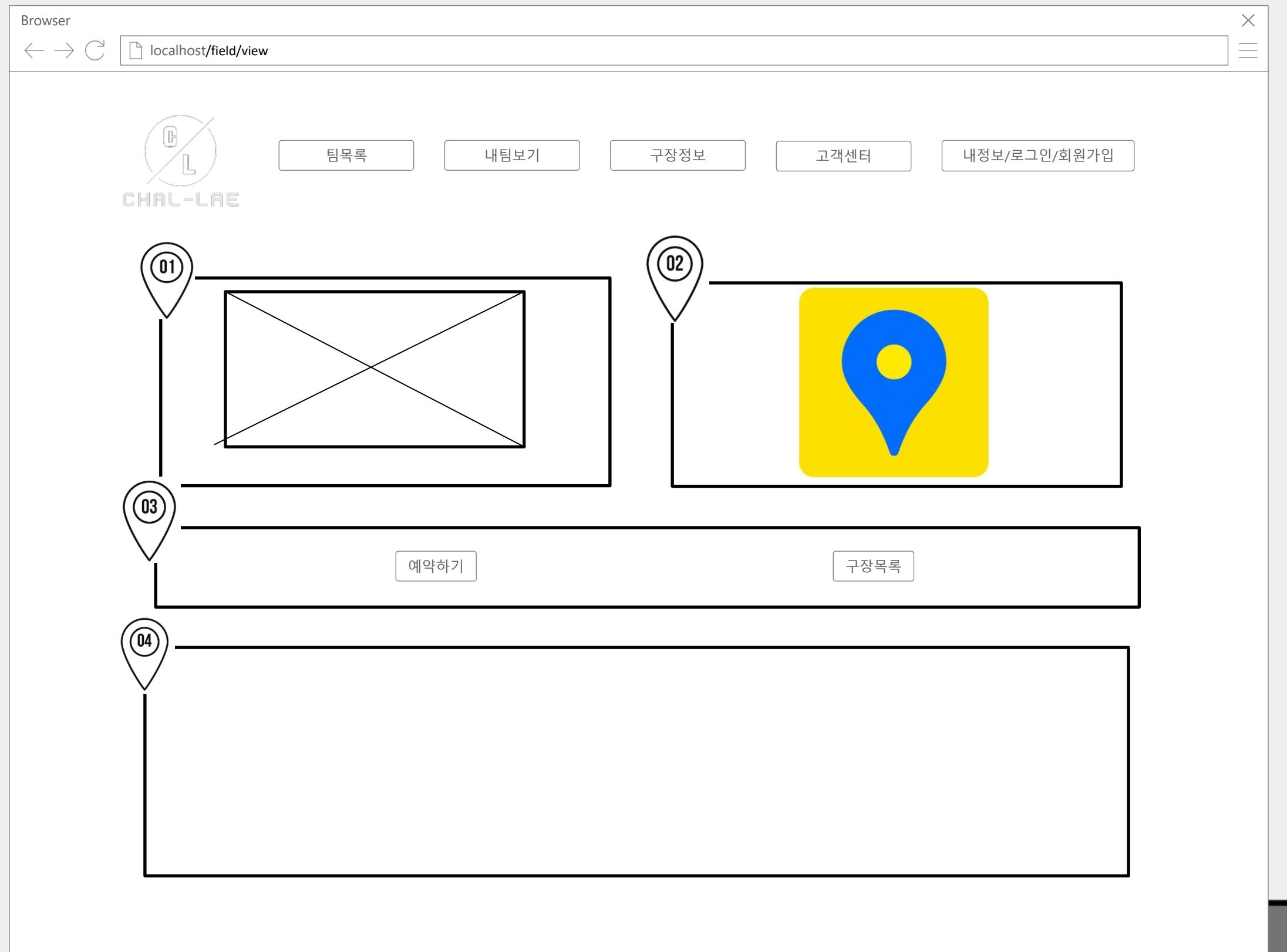
화면 설명

1 데이터에 저장된 구장의 대표 이미지와 구장전화번호를 함께 출력

2 구장의 위치로 지도 API를 이용, 구장 주소 출력 및 길 찾기와 큰 지도로 보기 버튼

3 예약하러 가기 버튼과 구장목록으로 갈 수 있는 버튼

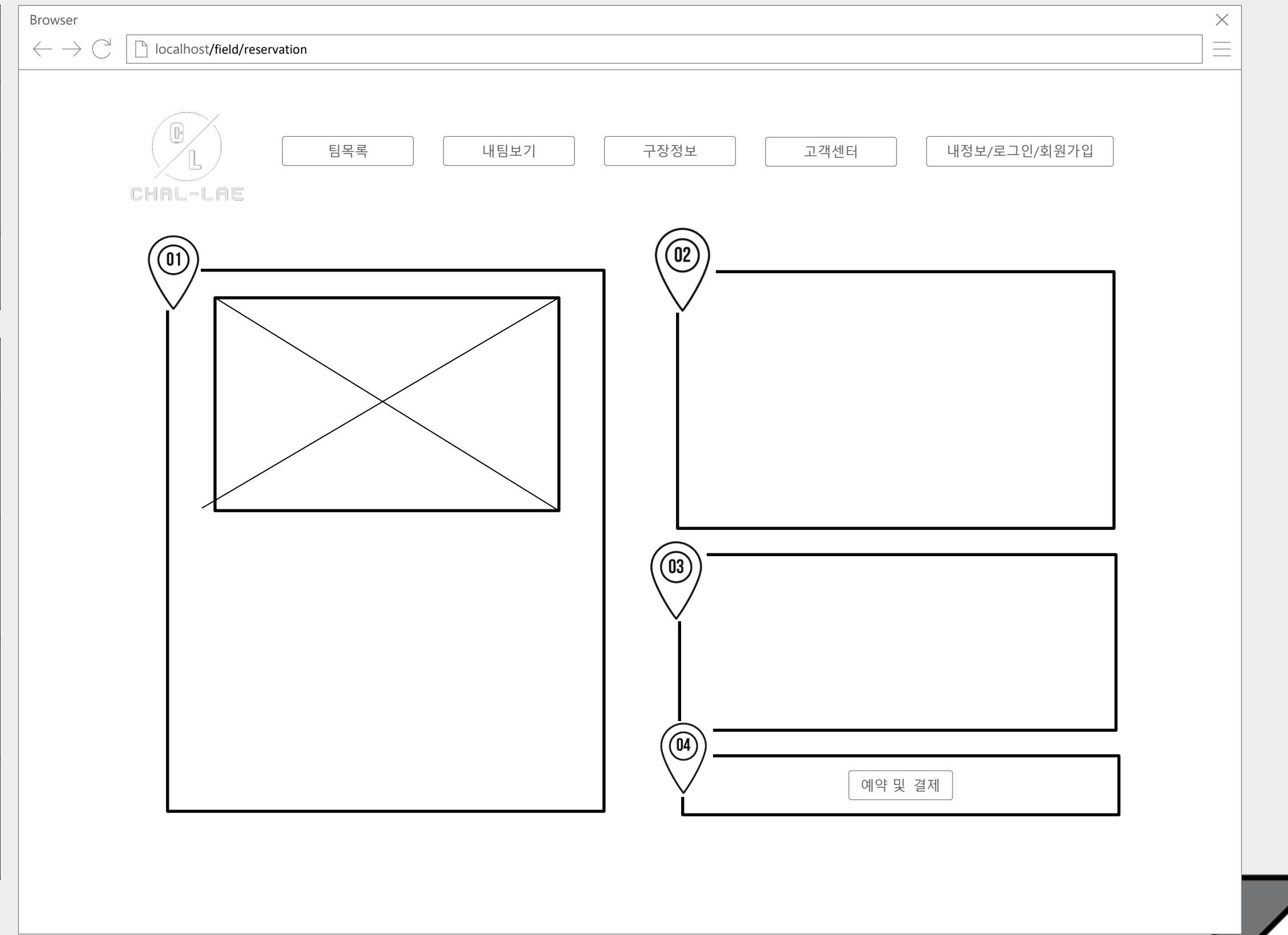
4 구장의 상세정보가 표시되는 부분
구장사이즈 및 평일과 주말의 가격이 동적변경



스토리보드

Project	CHAL-LAE
Page Title	구장예약
File Path	/field/reservation
File Name	reservation.html

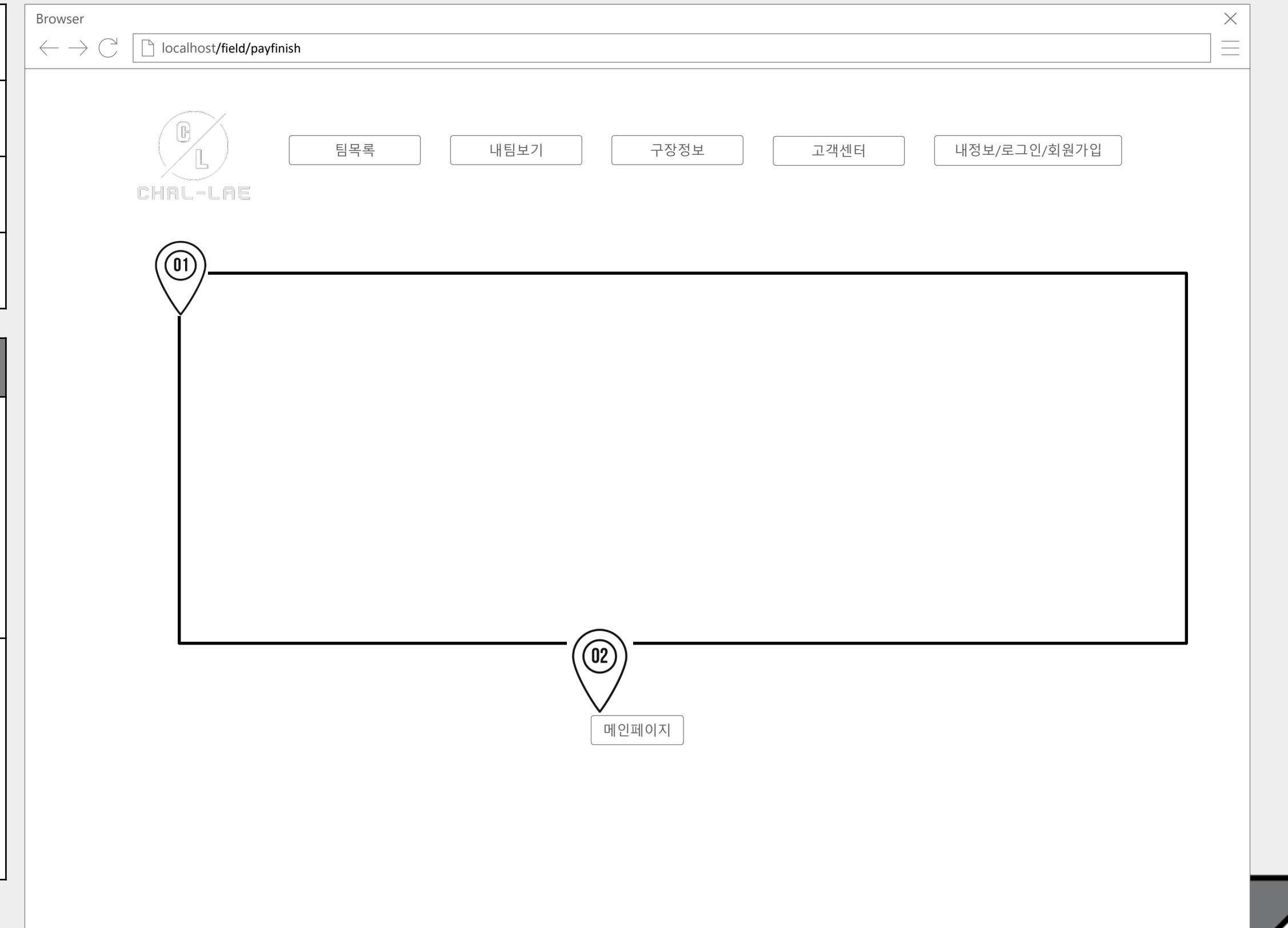
화면 설명	
1	데이터로 저장된 대표 이미지와 구장이름 출력
2	예약날짜와 시간을 선택할 수 있는 부분 연속된 시간만 선택가능
3	예약자정보와 선택한 예약정보를 볼 수 있는 부분
4	예약 완료된 데이터를 생성과 동시에 결제 API 호출



스토리보드

Project	CHAL-LAE
Page Title	결제완료
File Path	/field/payfinish
File Name	Payfinish.html

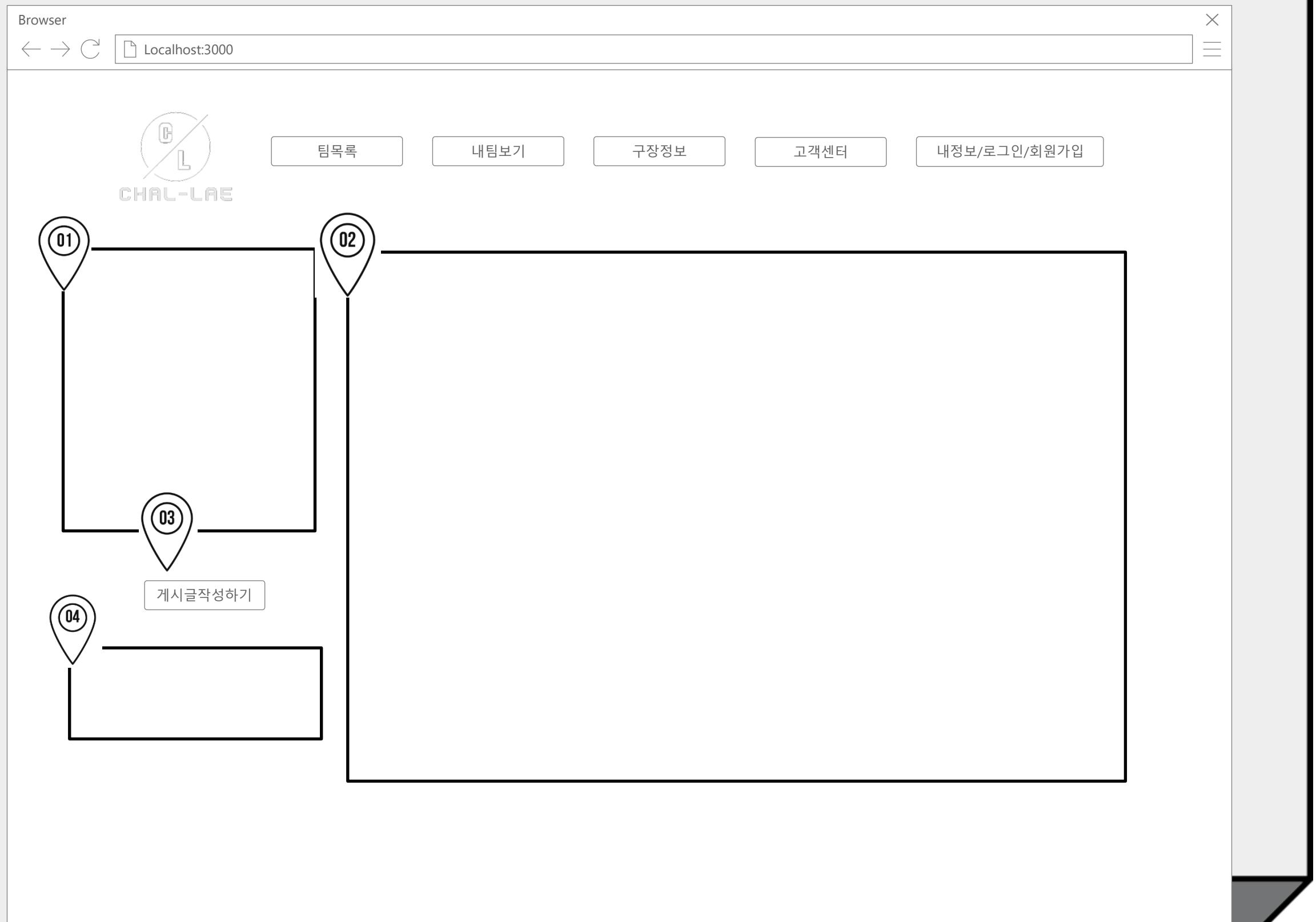
화면 설명	
1	데이터로 저장된 예약자 정보와 예약정보를 출력
2	예약확인이 완료되면 메인 페이지로 이동



스토리보드

Project	CHAL-LAE
Page Title	고객센터
File Path	/board/list
File Name	board.html

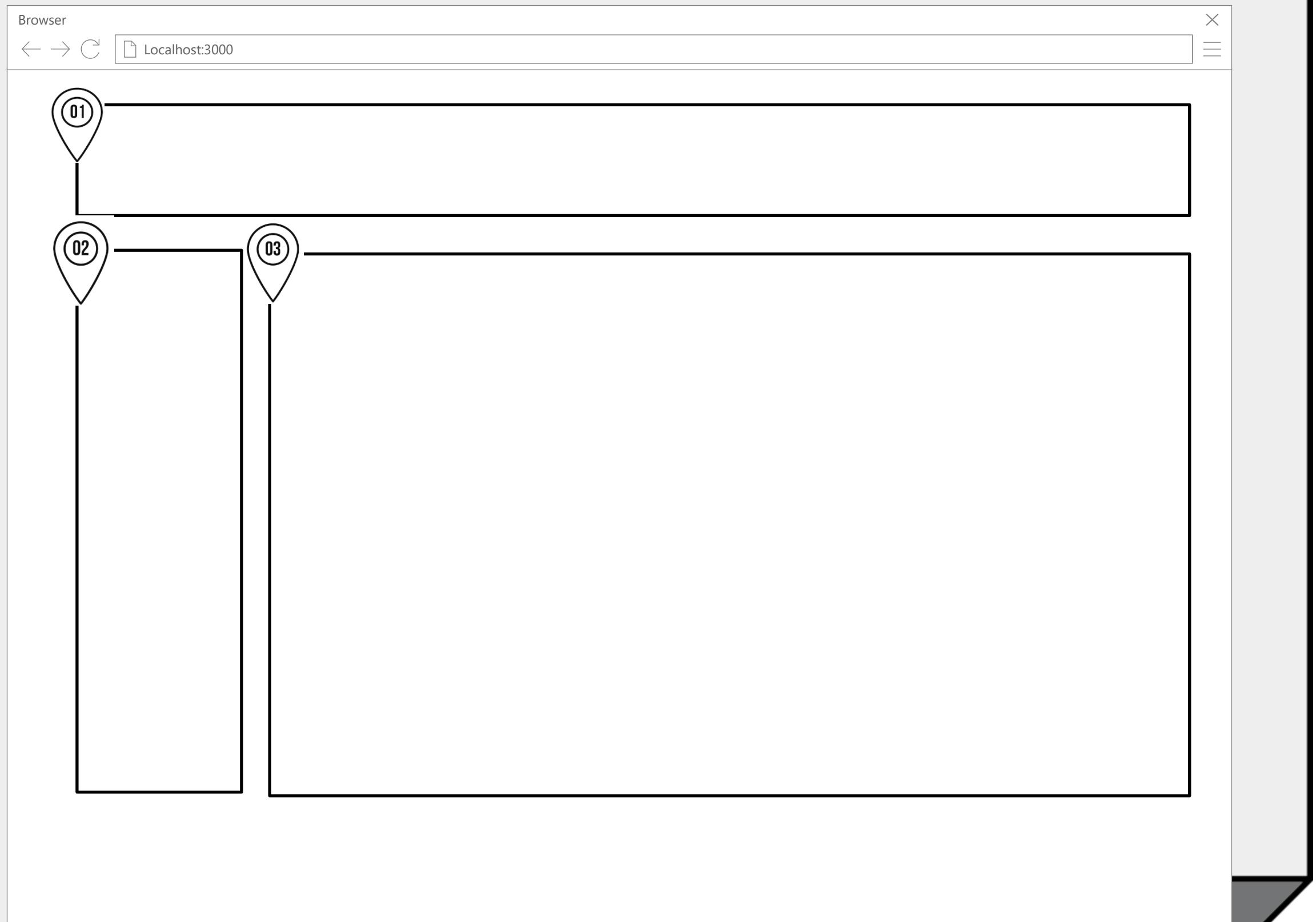
화면 설명	
1	게시글 리스트가 출력되는 부분
2	공지사항이 리스트가 출력되는 부분
3	게시글 등록 할 수 있는 등록버튼
4	게시글 페이지 처리



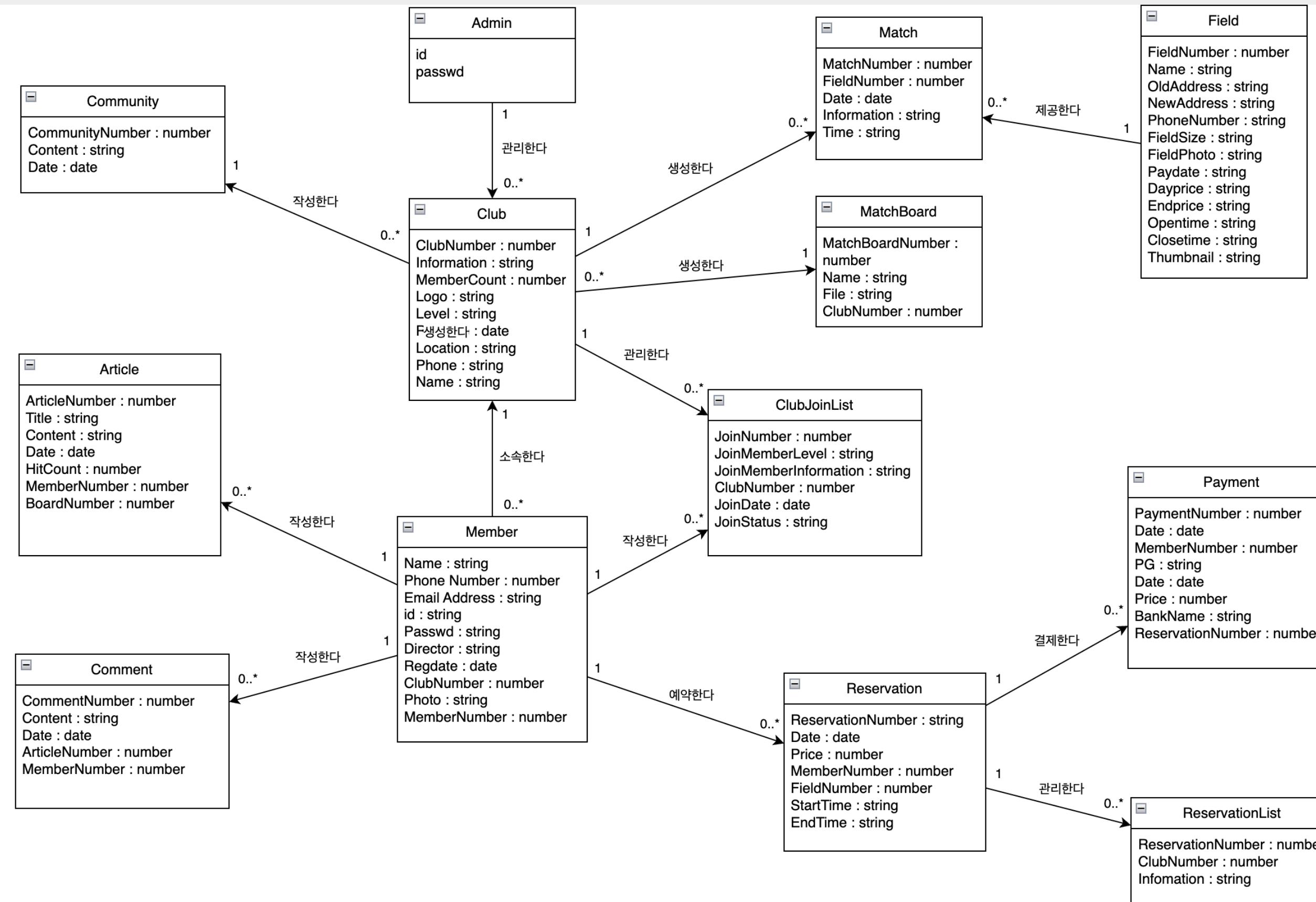
스토리보드

Project	CHAL-LAE
Page Title	관리자페이지
File Path	:3000
File Name	Payfinish.html

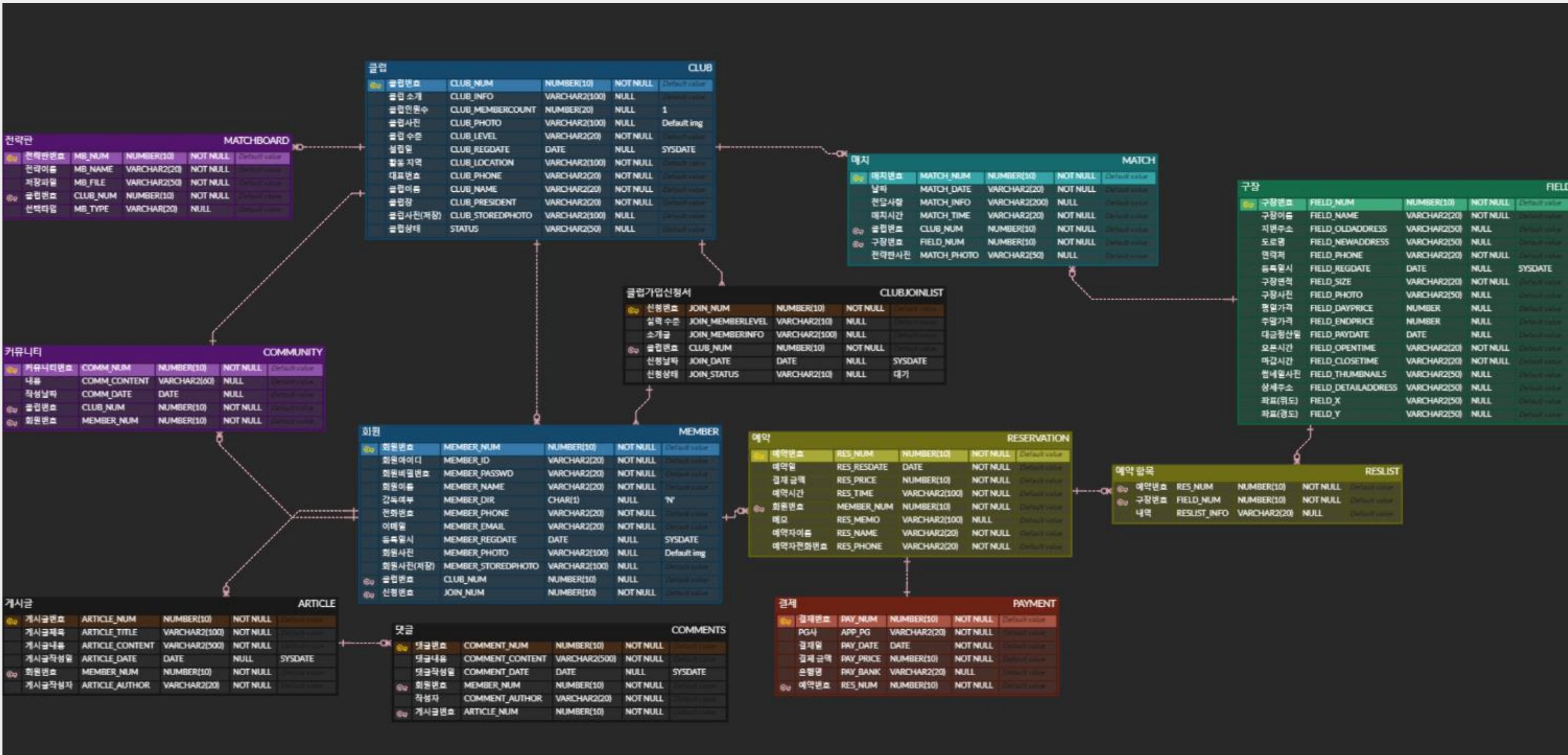
화면 설명	
1	어드민 페이지 로그인 확인 및 종료버튼
2	홈, 대시보드, 멤버관리, 클럽목록, 팀 승인 버튼으로 이동하여 해당 기능사용가능
3	해당 기능이 보여지는 구역



도메인 모델



데이터 모델 (ERD)



테이블 정의서

CLUB (클럽)

Column Name	Data Type	Constraint	Nulls	FK Table	FK Column	Default	Column Desc
CLUB_NUM	NUMBER(10)	PK	NOT NULL				클럽 번호
CLUB_INFO	VARCHAR2(100)		NONE				클럽 소개
CLUB_MEMBERCOUNT	NUMBER(20)		NONE			1	클럽 인원수
CLUB_PHOTO	VARCHAR2(100)		NONE			default.img	클럽 사진
CLUB_LEVEL	VARCHAR2(20)		NOT NULL				클럽 수준
CLUB_REGDATE	DATE		NONE			SYSDATE	설립일
CLUB_LOCATION	VARCHAR2(100)		NOT NULL				활동 지역
CLUB_PHONE	VARCHAR2(20)		NOT NULL				대표 번호
CLUB_NAME	VARCHAR2(20)		NOT NULL				클럽 이름
CLUB_PRESIDENT	VARCHAR2(20)		NOT NULL			SYSDATE	클럽 장
CLUB_STOREDPHOTO	VARCHAR2(100)		NONE				클럽사진(저장)
STATUS	VARCHAR2(50)		NONE				클럽 상태

MATCH (매치)

Column Name	Data Type	Constraint	Nulls	FK Table	FK Column	Default	Column Desc
MATCH_NUM	NUMBER(10)	PK	NOT NULL				매치 번호
MATCH_DATE	VARCHAR2(20)		NOT NULL				날짜
MATCH_INFO	VARCHAR2(200)		NONE				전달사항
MATCH_TIME	VARCHAR2(20)		NOT NULL				매치 시간
CLUB_NUM	NUMBER(10)	FK	NOT NULL	CLUB	CLUB_NUM		클럽 번호
FIELD_NUM	NUMBER(10)	FK	NOT NULL	FIELD	FIELD_NUM		구장 번호
MATCH_PHOTO	VARCHAR2(50)		NONE				전략판 사진

MATCHBOARD (전략판)

Column Name	Data Type	Constraint	Nulls	FK Table	FK Column	Default	Column Desc
MB_NUM	NUMBER(10)	PK	NOT NULL				전략판 번호
MB_NAME	VARCHAR2(20)		NOT NULL				전략 이름
MB_FILE	VARCHAR2(20)		NOT NULL				저장파일
CLUB_NUM	NUMBER(10)	FK	NOT NULL	CLUB	CLUB_NUM		클럽번호
MB_TYPE	VARCHAR2(20)		NONE				선택타입

MEMBER (회원)

Column Name	Data Type	Constraint	Nulls	FK Table	FK Column	Default	Column Desc
MEMBER_NUM	NUMBER(10)	PK	NOT NULL				회원 번호
MEMBER_ID	VARCHAR2(20)		NOT NULL				회원 아이디
MEMBER_PASSWD	VARCHAR2(20)		NOT NULL				회원 비밀번호
MEMBER_NAME	VARCHAR2(20)		NOT NULL				회원 이름
MEMBER_DIR	CHAR(1)	CK IN('Y, N')	NONE			N	감독 여부
MEMBER_PHONE	VARCHAR2(20)		NOT NULL				전화번호
MEMBER_EMAIL	VARCHAR2(20)		NOT NULL				이메일
MEMBER_REGDATE	DATE		NONE			SYSDATE	등록일시
MEMBER_PHOTO	VARCHAR2(100)		NONE			Default.img	회원 사진
MEMBER_STOREDPHOTO	VARCHAR2(100)		NONE				회원 사진(저장)
CLUB_NUM	NUMBER(10)	FK	NONE	CLUB	CLUB_NUM		클럽 번호
JOIN_NUM	NUMBER(10)	FK	NOT NULL	CLUBJOINLIST	JOIN_NUM		신청 번호

COMMUNITY (커뮤니티)

Column Name	Data Type	Constraint	Nulls	FK Table	FK Column	Default	Column Desc
COMM_NUM	NUMBER(10)	PK	NOT NULL				커뮤니티 번호
COMM_CONTENT	VARCHAR2(60)		NONE				내용
COMM_DATE	DATE		NONE				작성날짜
CLUB_NUM	NUMBER(10)	FK	NOT NULL	CLUB	CLUB_NUM		클럽번호
MEMBER_NUM	NUMBER(10)	FK	NOT NULL	MEMBER	MEMBER_NUM		회원번호

CLUBJOINLIST (클럽 가입 신청서)

Column Name	Data Type	Constraint	Nulls	FK Table	FK Column	Default	Column Desc
JOIN_NUM	NUMBER(10)	PK	NOT NULL				신청번호
JOIN_MEMBERLEVEL	VARCHAR2(10)		NONE				실력수준
JOIN_MEMBERINFO	VARCHAR2(100)		NONE				소개글
CLUB_NUM	NUMBER(10)	FK	NOT NULL	CLUB	CLUB_NUM	SYSDATE	클럽번호
JOIN_DATE	DATE		NONE				신청날짜
JOIN_STATUS	VARCHAR2(10)		NONE			대기	신청상태

테이블 정의서

FIELD (구장)

Column Name	Data Type	Constraint	Nulls	FK Table	FK Column	Default	Column Desc
FIELD_NUM	NUMBER(10)	PK	NOT NULL				구장번호
FIELD_NAME	VARCHAR2(20)		NOT NULL				구장이름
FIELD_OLDADDRESS	VARCHAR2(50)		NONE				지번주소
FIELD_NEWWADDRESS	VARCHAR2(50)		NONE				도로명
FIELD_PHONE	VARCHAR2(20)		NOT NULL				연락처
FIELD_REGDATE	DATE		NONE			SYSDATE	등록일시
FIELD_SIZE	VARCHAR2(20)		NOT NULL				구장면적
FIELD_PHOTO	VARCHAR2(50)		NONE				구장사진
FIELD_DAYPRICE	NUMBER(10)		NONE				평일가격
FIELD_ENDPICE	NUMBER(10)		NONE				주말가격
FIELD_PAYDATE	DATE		NONE				대금정산일
FIELD_OPENTIME	VARCHAR2(20)		NOT NULL				오픈시간
FIELD_CLOSETIME	VARCHAR2(20)		NOT NULL				마감시간
FIELD_THUMBNAILS	VARCHAR2(50)		NONE				썸네일사진
FIELD_DETAILADDRESS	VARCHAR2(50)		NONE				상세주소
FIELD_X	VARCHAR2(50)		NONE				좌표(위도)
FIELD_Y	VARCHAR2(50)		NONE				좌표(경도)

COMMENT (댓글)

Column Name	Data Type	Constraint	Nulls	FK Table	FK Column	Default	Column Desc
COMMENT_NUM	NUMBER(10)	PK	NOT NULL				댓글 번호
COMMENT_CONTENT	VARCHAR2(500)		NOT NULL				댓글 내용
COMMENT_DATE	DATE		NONE			SYSDATE	댓글 작성일
MEMBER_NUM	NUMBER(10)	FK	NOT NULL	MEMBER	MEMBER_NUM		회원번호
COMMENT_AUTHOR	VARCHAR2(20)		NOT NULL				작성자
ARTICLE_NUM	NUMBER(10)	FK	NOT NULL	ARTICLE	ARTICLE_NUM		게시글 번호

RESERVATION (예약)

Column Name	Data Type	Constraint	Nulls	FK Table	FK Column	Default	Column Desc
RES_NUM	NUMBER(10)	PK	NOT NULL				예약번호
RES_RESDATE	DATE			NOT NULL			예약일
RES_PRICE	NUMBER(10)			NOT NULL			결제금액
RES_TIME	VARCHAR2(100)			NOT NULL			예약시간
MEMBER_NUM	NUMBER(10)	FK	NOT NULL	MEMBER	MEMBER_NUM		회원번호
RES_MEMO	VARCHAR2(100)			NULL			메모
RES_NAME	VARCHAR2(20)			NOT NULL			예약자이름
RES_PHONE	VARCHAR2(20)			NOT NULL			예약자 전화번호

RESLIST (예약항목)

Column Name	Data Type	Constraint	Nulls	FK Table	FK Column	Default	Column Desc
RES_NUM	NUMBER(10)	FK	NOT NULL	RESERVATION	RES_NUM		예약번호
FIELD_NUM	NUMBER(10)	FK	NOT NULL	FIELD	FIELD_NUM		구장번호
RESLIST_INFO	VARCHAR2(20)			NULL			내역

PAYMENT (결제)

Column Name	Data Type	Constraint	Nulls	FK Table	FK Column	Default	Column Desc
PAY_NUM	NUMBER(10)	PK	NOT NULL				결제번호
APP_PG	VARCHAR2(20)			NOT NULL			PG사
PAY_DATE	DATE			NOT NULL			결제일
PAY_PRICE	NUMBER(10)			NOT NULL			결제금액
PAY_BANK	VARCHAR2(20)			NULL			은행명
RES_NUM	NUMBER(10)	FK	NOT NULL	RESERVATION	RES_NUM		예약번호

ARTICLE (게시글)

Column Name	Data Type	Constraint	Nulls	FK Table	FK Column	Default	Column Desc
ARTICLE_NUM	NUMBER(10)	PK	NOT NULL				게시글 번호
ARTICLE_TITLE	VARCHAR2(100)			NOT NULL			게시글 제목
ARTICLE_CONTENT	VARCHAR2(500)			NOT NULL			게시글 내용
ARTICLE_DATE	DATE			NULL		SYSDATE	게시글 작성일
MEMBER_NUM	NUMBER(10)	FK	NOT NULL	MEMBER	MEMBER_NUM		회원번호
ARTICLE_AUTHOR	VARCHAR2(20)			NOT NULL			게시글 작성자

5. 주요기능과

코
고



주요 기능과 코드 (회원수정)



```
/** 회원 정보 수정 처리 */
@PostMapping(value = "/update") □ gerrard
public String updateMemberInfo(
    @RequestParam(value = "newEmail", required = false) String newEmail,
    @RequestParam(value = "password", required = false) String password,
    HttpSession session,
    RedirectAttributes redirectAttributes) {

    MemberDto loginMember = (MemberDto) session.getAttribute("loginMember");
    if (newEmail != null && !newEmail.isEmpty()) {
        loginMember.setEmail(newEmail);
    }

    if (password != null && !password.isEmpty()) {
        loginMember.setPasswd(password);
    }
    memberService.editMember(loginMember);

    // 수정이 완료된 후에는 인덱스 페이지로 리다이렉트
    return "redirect:/";
}
```

```
function updateMemberInfo() : void { Show usages □ gerrard
    const form : HTMLElement = document.getElementById('editForm');
    const newEmail = form.querySelector(selectors: '#newEmail').value;
    const password = form.querySelector(selectors: '#password').value;
    const repasswd = form.querySelector(selectors: '#repasswd').value;

    // 비밀번호와 비밀번호 재입력이 일치하는지 확인
    if (password !== repasswd) {
        document.getElementById(elementId: 'updateMessage').innerText = '비밀번호와 재입력이 일치하지 않습니다.';
        form.reset();
        return;
    }

    const formData : FormData = new FormData(form);

    fetch(input: '/member/update', init: {
        method: 'POST',
        body: formData
    })
    .then(response : Response => {
        if (response.ok) {
            document.getElementById(elementId: 'updateMessage').innerText = '회원 정보가 성공적으로 수정되었습니다.';
            form.reset();
        } else {
            document.getElementById(elementId: 'updateMessage').innerText = '회원 정보 수정에 실패했습니다.';
        }
        return response.text();
    })
}
```

1. 로그인한 회원의 정보를 수정된 정보로 DB를 업데이트

2. Fetch를 이용해서 비동기로 토글창에서 회원정보 수정작업

주요 기능과 코드 (관리자모드 - 화면구성)

CHAL-LAE ADMIN PAGE

관리자님 로그인중! 종료하기

홈
대시보드
멤버 관리
클럽 목록
팀 승인

CHAL-LAE 관리자님 환영합니다!

오늘의 우리 사이트 현황입니다.

총 회원 수
33명

총 클럽 수
32개

총 게시글 수
29개



CHAL-LAE ADMIN PAGE

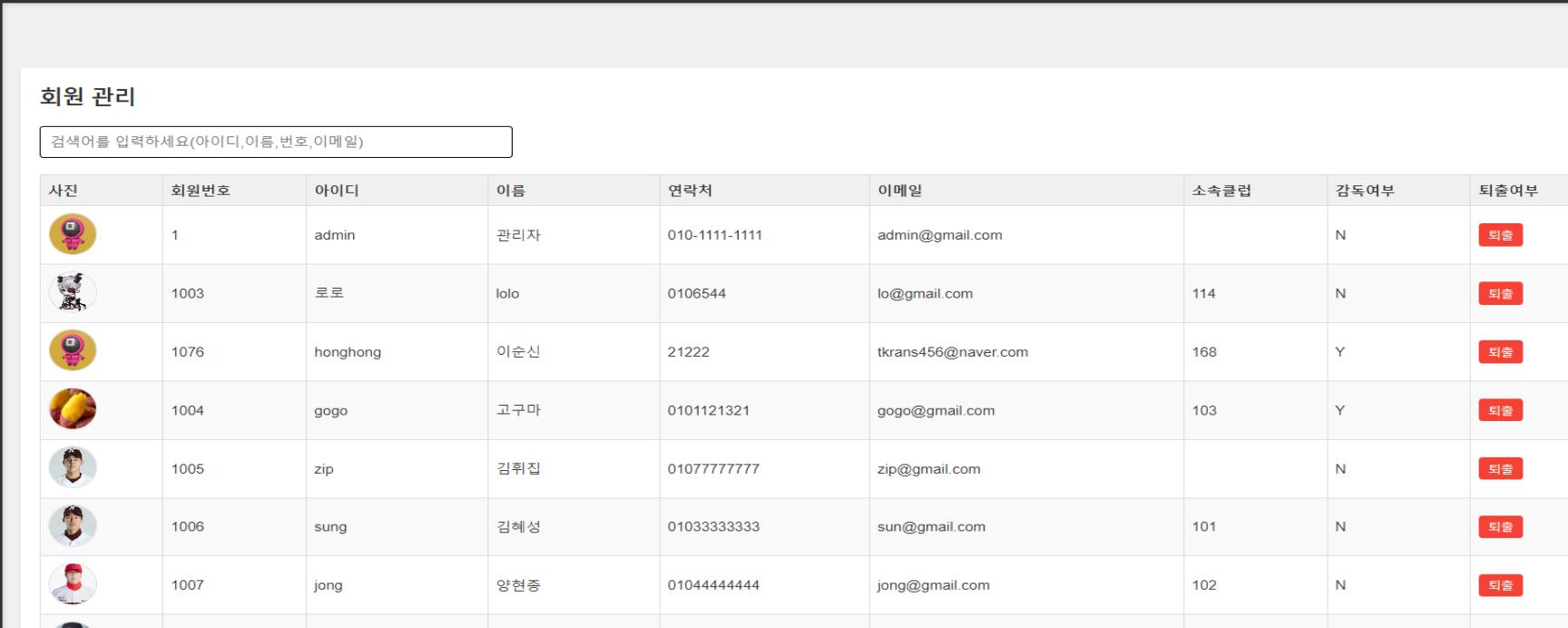
관리자님 로그인중! 종료하기

홈
대시보드
멤버 관리
클럽 목록
팀 승인

회원 관리

검색어를 입력하세요(아이디, 이름, 번호, 이메일)

사진	회원번호	아이디	이름	연락처	이메일	소속클럽	감독여부	퇴출여부
	1	admin	관리자	010-1111-1111	admin@gmail.com		N	<button>퇴출</button>
	1003	로로	lolo	0106544	lo@gmail.com	114	N	<button>퇴출</button>
	1076	honghong	이순신	21222	tkrans456@naver.com	168	Y	<button>퇴출</button>
	1004	gogo	고구마	0101121321	gogo@gmail.com	103	Y	<button>퇴출</button>
	1005	zip	김휘집	01077777777	zip@gmail.com		N	<button>퇴출</button>
	1006	sung	김혜성	01033333333	sun@gmail.com	101	N	<button>퇴출</button>
	1007	jong	양현종	01044444444	jong@gmail.com	102	N	<button>퇴출</button>
	1008	young	김도영	01055555555	young@gmail.com	102	N	<button>퇴출</button>



```
const MemberCount = () => {
  const [memberCount, setMemberCount] = useState(0);

  useEffect(() => {
    const memberCount = async () => {
      try {
        const response = await axios.get('/api/memberCount');
        let data;
        if (typeof response.data === 'string') {
          data = JSON.parse(response.data);
        } else {
          data = response.data;
        }
        setMemberCount(data);
      } catch (error) {
        console.error('회원 수 정보 받기 오류:', error);
      }
    };
    memberCount();
  }, []);
}
```

1. 서버에서 온 JSON데이터를 Parse해서 화면에 출력

2. 클럽수와 게시글수도 같은 방법으로 구현

주요 기능과 코드 (관리자모드 - 팀 승인)

```
/**  
 * 클럽 생성 승인 API  
 */  
  
@PostMapping("/approve")  
public void approveClub(@RequestParam int clubNum) {  
    clubService.updateStatus(clubNum, status: "승인");  
  
    String presidentEmail = clubService.findClubById(clubNum);  
  
    memberService.updateMemberWithClubInfo(clubNum);  
  
    memberService.updateClubNumByPresident(clubNum);  
  
    // 클럽 승인 이메일 발송  
    String subject = "[찰래 홈페이지] 클럽 승인 안내";  
    String body = "요청하신 클럽이 성공적으로 승인되었습니다.";  
  
    try {  
        sendEmail(presidentEmail, subject, body);  
    } catch (MessagingException e) {  
        e.printStackTrace();  
    }  
}
```

1. DB에서 식별자인 ClubNum을 이용하여 승인대기 상태의 클럽을 승인완료 또는 삭제로 업데이트.

```
/**  
 * 이메일 발송 메서드  
 */  
  
private void sendEmail(String to, String subject, String body) throws MessagingException {  
    MimeMessage message = javaMailSender.createMimeMessage();  
    MimeMessageHelper messageHelper = new MimeMessageHelper(message, multipart: true, encoding: "UTF-8");  
    messageHelper.setFrom(from);  
    messageHelper.setTo(to);  
    messageHelper.setSubject(subject);  
    messageHelper.setText(body);  
    javaMailSender.send(message);  
}
```

2. 이메일 발송 메서드를 통해서 승인 또는 거절을 메일 발송

```
const Teamapproval = () => {  
    const [clubs, setClubs] = useState([]);  
  
    useEffect(() => {  
        pendingClubs();  
    }, []);  
  
    const pendingClubs = async () => {  
        try {  
            const response = await axios.get('/api/pending');  
            setClubs(response.data);  
        } catch (error) {  
            console.error('승인대기 클럽 조회 실패', error);  
        }  
    };  
  
    const handleApprove = async (clubNum) => {  
        try {  
            await axios.post('/api/approve', null, { params: { clubNum } });  
            setClubs(prevClubs => prevClubs.filter(club => club.clubNum !== clubNum));  
        } catch (error) {  
            console.error('클럽승인 오류', error);  
        }  
    };  
  
    const handleReject = async (clubNum) => {  
        try {  
            await axios.post('/api/reject', null, { params: { clubNum } });  
            setClubs(prevClubs => prevClubs.filter(club => club.clubNum !== clubNum));  
        } catch (error) {  
            console.error('클럽 거절 오류', error);  
        }  
    };  
};
```

4. Axios를 통해 api호출 후 승인 또는 거절 처리

3. 가공된 데이터를 리액트 구현 폐이
지로 JSON데이터로 전송

주요 기능과 코드 (내 팀커뮤니티 - 감독)

감독만 볼 수 있는 추방 버튼으로
DB에서 팀원의 팀 정보를 null로 업데이트

```
<td th:text="${clubMember.dir} == 'N' ? '팀원' : '감독'"></td> ssol98, 2024-05-25 오전 12:1
<td th:text="${clubMember.phone}"></td>
<!-- 감독 제외 팀원에게만 뜨는 추방 --&gt;

&lt;td th:if="${clubMember.dir} == 'N' and ${loginMember.dir} == 'Y'"&gt;
    &lt;form th:action="@{/club/kick}" method="post"&gt;
        &lt;input type="hidden" name="memberNum" th:value="${clubMember.memberNum}" /&gt;
        &lt;button class="kickBtn" type="button" onclick="confirmKick(this.form)"&gt;추방&lt;/button&gt;
    &lt;/form&gt;
&lt;/td&gt;

&lt;update id="ClubMemberDelete" parameterType="MemberDto"&gt;
    UPDATE member
    SET club_num= null    ssol98, 2024-05-31 오후 5:50 .
    WHERE member_num = #{memberNum}
&lt;/update&gt;</pre>
```

```
@PostMapping("/kick")
public String deleteMember(@RequestParam("memberNum") String memberNum, RedirectAttributes redirectAttributes) {
    MemberDto memberDto = new MemberDto();
    memberDto.setMemberNum(memberNum);
    try {
        memberService.outClubMember(memberDto);
        redirectAttributes.addFlashAttribute("message", "회원이 성공적으로 추방되었습니다.");
    } catch (Exception e) {
        redirectAttributes.addFlashAttribute("message", "회원 추방에 실패했습니다.");
    }
    return "redirect:/club/myteam";
}
```



화면에 출력된 각 메시지의 'x' 표시로
감독만 팀원전체의 메시지를 삭제할 수
있다

```
<td class="comm-delete" th:if="${loginMember.memberNum} == ${community.memberNum} or ${loginMember.dir} == 'Y'">
    <form th:action="@{/club/delete}" method="post">
        <input type="hidden" name="commNum" th:value="${community.commNum}" />
        <button type="button" onclick="confirmDelete(this.form)">x</button>
    </form>
</td>

<delete id="ClubCommDelete" parameterType="CommunityDto">
    DELETE from community
    where comm_num = #{commNum}
</delete>
```

```
@PostMapping("/{}/delete")
public String deleteComm(@RequestParam("commNum") String commNum, RedirectAttributes redirectAttributes) {
    CommunityDto communityDto = new CommunityDto();
    communityDto.setCommNum(commNum);
    try {
        communityService.deleteCommContent(communityDto);
        redirectAttributes.addFlashAttribute("deleteMessage", "성공적으로 삭제되었습니다.");
    } catch (Exception e) {
        redirectAttributes.addFlashAttribute("deleteMessage", "삭제에 실패했습니다.");
    }
    return "redirect:/club/myteam";
}
```

ADD+ 버튼을 눌러 경기일정 등록할 수
있으며 품 인증
[설정]으로 이동하여 Windows를
등록 시 DB에 저장된다.
DB에 저장되어 있는 경기일정을
가져옴

주요 기능과 코드 (내 팀의 경기 일정)

1. Fetch요청을 이용해서 서버에서 경기 데이터 모두 가져 옴

```
/** 경기 일정 렌더링*/
let schedules : any[] = [];
try{
    /** fetch 요청을 사용하여 서버로부터 데이터 가져오기*/
    const response : Response = await fetch(`input: '/api/schedules/${clubNum}`);
    if(response.ok){
        schedules = await response.json(); /*서버로부터 받은 JSON 데이터를 파싱*/
    } else {
        console.error("서버로부터 데이터를 가져오는 데 실패했습니다.");
        console.error(`http 상태코드 : ${response.status}`);
    }
} catch (error) {
    /** 에러 발생 시 */
    console.error("데이터를 가져오는 과정에서 에러가 발생했습니다:", error);
}

/** 현재 선택한 달에 해당하는 경기 일정만 필터링*/
const scheduleKey : string = `${nowYear}-${String(nowMonth).padStart(2, '0')}`;
const monthSchedules : any[] = schedules.filter(schedule => {
    const scheduleDate : Date = new Date(schedule.matchDate);
    return scheduleDate.getFullYear() === nowYear && (scheduleDate.getMonth() + 1) === nowMonth;
});

/** 이전 달 */
const prevMonth = () : void => { Show usages ↗ gerrard +1
    date.setMonth(date.getMonth() - 1);
    renderMonth();
};

/** 다음 달 */
const nextMonth = () : void => { Show usages ↗ gerrard +1
    date.setMonth(date.getMonth() + 1);
    renderMonth();
};

/** 당일로 가는 기능 */
const goTodayMonth = () : void => { Show usages ↗ ssol98 +1
    date.setMonth(new Date().getMonth());
    renderMonth();
};

renderMonth();
```

2. 현재 설정된 달에 해당하는 경기 일정
데이터가 보이게끔 필터링 함
- 이전이나 다음으로 달 변경 가능

```
const schedulesContainer : Element = document.querySelector(selectors: ".team-match-schedule-detail");
schedulesContainer.innerHTML = ''; /* 이전 일정 초기화 */

monthSchedules.forEach(schedule => {
    const matchInfo : HTMLDivElement = document.createElement(tagName: "div");
    matchInfo.classList.add("match-Info");

    const ulInfo : HTMLULListElement = document.createElement(tagName: "ul");
    matchInfo.appendChild(ulInfo);

    /* 경기일자에서 일자부분만 추출 */
    const dateParts = schedule.matchDate.split('-');
    const dayOnly = dateParts[2];
    const dateLi : HTMLLIElement = document.createElement(tagName: "li");
    dateLi.textContent = dayOnly+" 일";
    dateLi.classList.add("date-li");
    ulInfo.appendChild(dateLi);

    // 필드 이름
    const fieldLi : HTMLLIElement = document.createElement(tagName: "li");
    const fieldName = schedule.fieldName;
    fieldLi.textContent = " "+fieldName+" ";
    fieldLi.classList.add("field-li");
    ulInfo.appendChild(fieldLi);

    // 경기시간
    const timeLi : HTMLLIElement = document.createElement(tagName: "li");
    timeLi.textContent = schedule.matchTime;
    timeLi.classList.add("time-li");
    ulInfo.appendChild(timeLi);
});
```

3. 각 달에 해당하는 일정을 html요소로 생성



주요 기능과 코드 (경기일정 등록)



1. DB에 저장되어있는 구장목록을 선택하고 시간과 전달사항을 입력
2. 프리셋을 불러오거나 새로 만들어서 저장하고 일정을 등록하면 DB에 저장

주요 기능과 코드 (전략판)

```
/* ----- */
/*
 * form 태그 안에 있는 등록 버튼을 누르면 캔버스 그림 상태를 서버에 전달해 저장
 *
 */
document.querySelector( selectors: '#createForm').addEventListener( type: 'submit', listener: function (event : Event) : void {
    event.preventDefault();
    saveCanvas();
});

// 클라이언트 사이드 렌더링
function saveCanvas() : void { Show usages ↗ inrok
    const selectBox : Element = document.querySelector( selectors: '#fieldSelect');
    const reservationDate : Element = document.querySelector( selectors: '#reservationDate');
    const reservationTime : Element = document.querySelector( selectors: '#reservationTime');

    // select 박스가 선택되지 않은 경우 경고 메시지 표시
    if (selectBox.selectedIndex === 0) {
        Swal.fire({
            icon: 'warning',
            title: '경고',
            text: '[필수] 구장을 선택해주세요.',
        });
        selectBox.focus();
        return;
    }
    // 등록 날짜가 없거나 공백이면 경고
    if (!reservationDate.value.trim()) {
        Swal.fire({
            icon: 'warning',
            title: '경고',
            text: '[필수] 날짜를 선택해주세요.',
        });
        reservationDate.focus();
        return;
    }
}
```

```
const canvas : Element = document.querySelector( selectors: '#soccer-board');
canvas.toBlob( callback: function (blob : Blob | null ) : void {
    const formData : FormData = new FormData();

    // 유니크한 파일 이름 생성
    const uniqueFileName : string = 'canvas_' + new Date().getTime() + '.png';
    formData.append( name: 'canvasImage', blob, uniqueFileName);

    fetch( input: '/club/uploadCanvas', init: {
        method: 'POST',
        body: formData
    }).then(response : Response => {
        if (response.ok) {
            response.json().then(data => {
                document.querySelector( selectors: '#canvasData').value = data.filePath;
                Swal.fire({
                    icon: 'success',
                    title: '등록 완료',
                    text: '일정이 등록되었습니다.',
                });
                // 폼을 자동 제출
                document.querySelector( selectors: '#createForm').submit();
            });
        } else {
            Swal.fire({
                icon: 'warning',
                title: '등록 실패', // Alert 제목
                text: '저장에 실패했습니다. 다시 시도해주세요.', // Alert 내용
            });
        }
    }).catch(error => {
        console.error('Error:', error);
    });
}, type: 'image/png');
```

1. 사용자로부터 입력받은 데이터와 현재 캔버스 상태를 fetch API를 사용해 POST요청을 보내고
서버에서 응답이 성공적인 경우 Form을 제출해 DB에 저장

주요 기능과 코드 (전략판 프리셋)

```
// SweetAlert 대화 상자를 통해 확인 버튼을 누르면 함수 실행
Swal.fire({
  title: '프리셋을 저장하시겠습니까?',
  text: "저장 시 기존의 프리셋은 사라집니다. 프리셋을 저장하시려면 확인을 누르세요.",
  icon: 'info',
  showCancelButton: true,
  confirmButtonColor: '#3085d6',
  cancelButtonColor: '#d33',
  confirmButtonText: '확인',
  cancelButtonText: '취소'
}).then((result) => {
  if (result.isConfirmed) {
    const canvas : Element = document.querySelector('#soccer-board');
    canvas.toBlob((blob : Blob | null) : void => {
      // 유니크한 파일 이름 생성 (예: "canvas_1653079812345.png")
      const uniqueFileName : string = `preset_${new Date().getTime()}.png`;
      formData.append('name', 'canvasImage', blob, uniqueFileName);

      fetch('/club/createMatchBoard', {method: 'POST', body: formData})
        .then(response => {
          if (response.ok) {
            Swal.fire(
              '프리셋 저장 완료',
              '저장완료',
              'success'
            );
          } else {
            Swal.fire(
              '프리셋 저장 실패',
              '저장실패',
              'error'
            );
          }
        })
        .catch(error => {
          console.error('Error:', error);
          Swal.fire(
            '오류 발생',
            '프리셋 저장 중 오류가 발생했습니다.',
            'error'
          );
        });
    });
  }
});
```

```
* 프리셋 저장하기
*/
function submitPresetForm() : void { Show usages ↗ inrok
  const form : Element = document.querySelector(selectors: '#presetForm');
  const formData : FormData = new FormData(form);
  const selectedOption : FormDataEntryValue = formData.get('option');
  const presetName : FormDataEntryValue = formData.get('presetName');

  // 라디오 버튼이 선택되지 않았거나 presetName 이 입력되지 않은 경우 경고 메시지 표시
  if (!selectedOption) {
    Swal.fire({
      icon: 'warning',
      title: '✓',
      text: '프리셋 조건을 선택해주세요',
    });
    return; // submitPresetForm 종료
  }

  if (!presetName) {
    Swal.fire({
      icon: 'warning',
      title: '✓',
      text: '프리셋 이름을 입력해주세요',
    });
    return;
  }

  // SweetAlert 대화 상자를 통해 확인 버튼을 누르면 함수 실행
  Swal.fire({
    title: '프리셋을 저장하시겠습니까?',
    text: "저장 시 기존의 프리셋은 사라집니다. 프리셋을 저장하시려면 확인을 누르세요.",
    icon: 'info',
    showCancelButton: true,
    confirmButtonColor: '#3085d6',
  });
}
```

1. 사용자가 선택한 옵션과 현재 캔버스에 그려진 프리셋 정보를 fetch API를 사용해 POST 요청을 보내 서버에 파일을 전송

주요 기능과 코드 (구장예약 및 결제)

```
<!--결제api 및 예약달력 스크립트-->
<script type="text/javascript">
var fields = '[${fields}]';
var today = new Date();
var currentMonth = today.getMonth(); //현재날짜 월
var currentYear = today.getFullYear(); //현재날짜 년도
var fieldOpentime = '[[${fieldDetail2.fieldOpentime}]]';
var fieldClosetime = '[[${fieldDetail2.fieldClosetime}]]';
var fieldDayprice = '[[${fieldDetail2.fieldDayprice}]]';
var fieldEndprice = '[[${fieldDetail2.fieldEndprice}]]';
var selectedTimes = []; //선택된시간을 배열에 저장
// 달력기본설정(토요일일요일 구분 및 일주일계산)
function buildCalendar(){ Show usages ↗ ledjk99 +1
    var row = null;
    var cnt = 0;
    var calendarTable = document.getElementById("calendar");
    var calendarTableTitle = document.getElementById("calendarTitle");
    calendarTableTitle.innerHTML = today.getFullYear()+"년"+(today.getMonth()+1)+"월";
    var firstDate = new Date(today.getFullYear(), today.getMonth(), 1);
    var lastDate = new Date(today.getFullYear(), today.getMonth()+1, 0);
    while(calendarTable.rows.length > 2){
        calendarTable.deleteRow(calendarTable.rows.length -1);
    }
}

```

1. JS를 사용하여 캘린더를 생성

The screenshot shows a booking application interface. On the left, there is a calendar for August 2024. The date 08/08 is highlighted in blue. Below the calendar, it says "선택된 날짜 : 2024-08-08". To the right of the calendar is a detailed booking form:

대관 예약자 정보	
신청자	치과
연락처	1231
예약일자	2024-08-08
예약시간	10:00-11:00, 11:00-12:00, 12:00-13:00
총 결제금액	180000원
비고	<input type="text"/>

At the bottom right of the form, there is a blue button labeled "예약 및 결제".

```
for (i = 1; i <= lastDate.getDate(); i++) {
    if (cnt % 7 === 0) {
        row = calendarTable.insertRow();
    }
    cell = row.insertCell();
    cnt += 1;
    cell.setAttribute('id', i);
    cell.innerHTML = i;
    cell.style.textAlign = "center";
    cell.onclick = function(){
        var clickedYear = today.getFullYear();
        var clickedMonth = (1 + today.getMonth());
        var clickedDate = this.getAttribute('id');
        clickedDate = clickedDate >= 10 ? clickedDate : '0' + clickedDate;
        clickedMonth = clickedMonth >= 10 ? clickedMonth : '0' + clickedMonth;
        var clickedYMD = clickedYear + "-" + clickedMonth + "-" + clickedDate;
        document.getElementById("selectedDate").innerText = clickedYMD;
        document.getElementById("selectedDate2").innerText = clickedYMD;
        var clickedDay = new Date(clickedYear, clickedMonth - 1, clickedDate).getDay();
        if (clickedDay == 0 || clickedDay == 6) {
            document.getElementById("priceBottom").innerText = fieldEndprice + "원"; // 주말 가격 표시
        } else {
            document.getElementById("priceBottom").innerText = fieldDayprice + "원"; // 주중 가격 표시
        }
    }
}
```

2. 구장의 오픈시간 마감시간을 불러와서

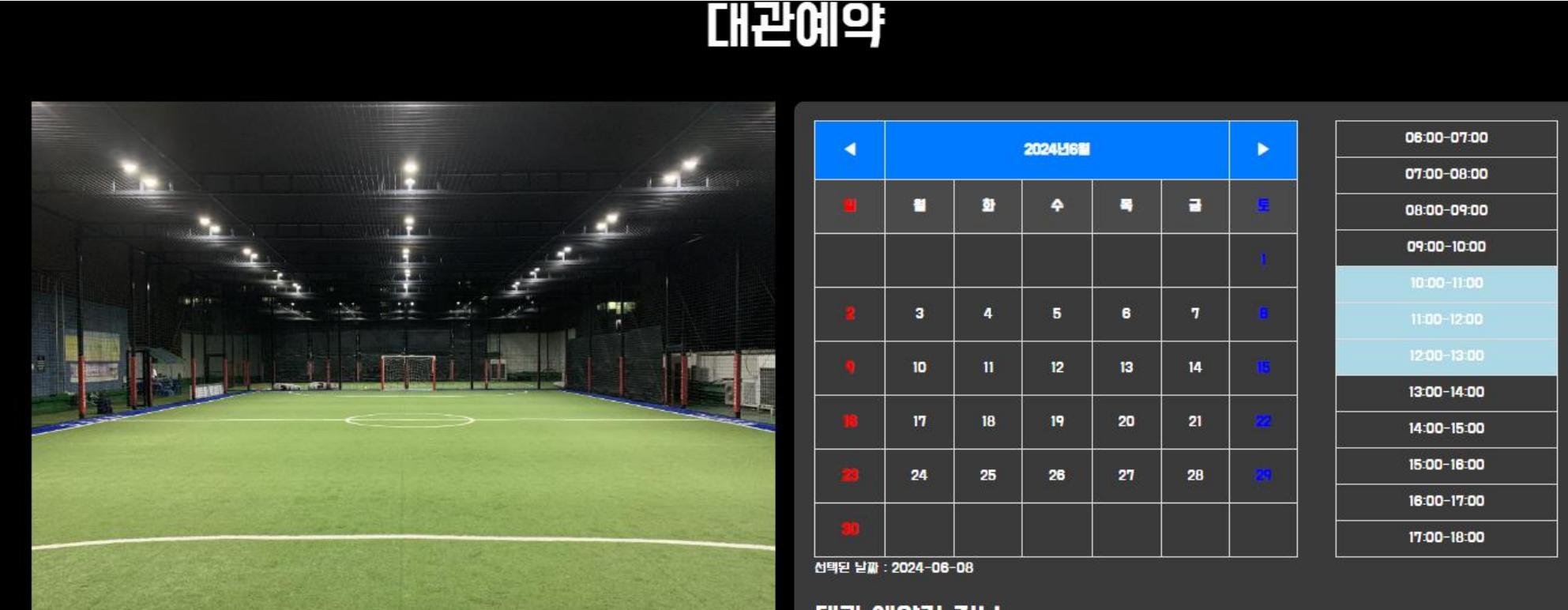
구장마다 동적으로 바뀌게 함

3. 구장의 금액도 주중요금과 주말요금을 다르게 나오게 구

현.

Windows 정품
[설정]으로 이동하

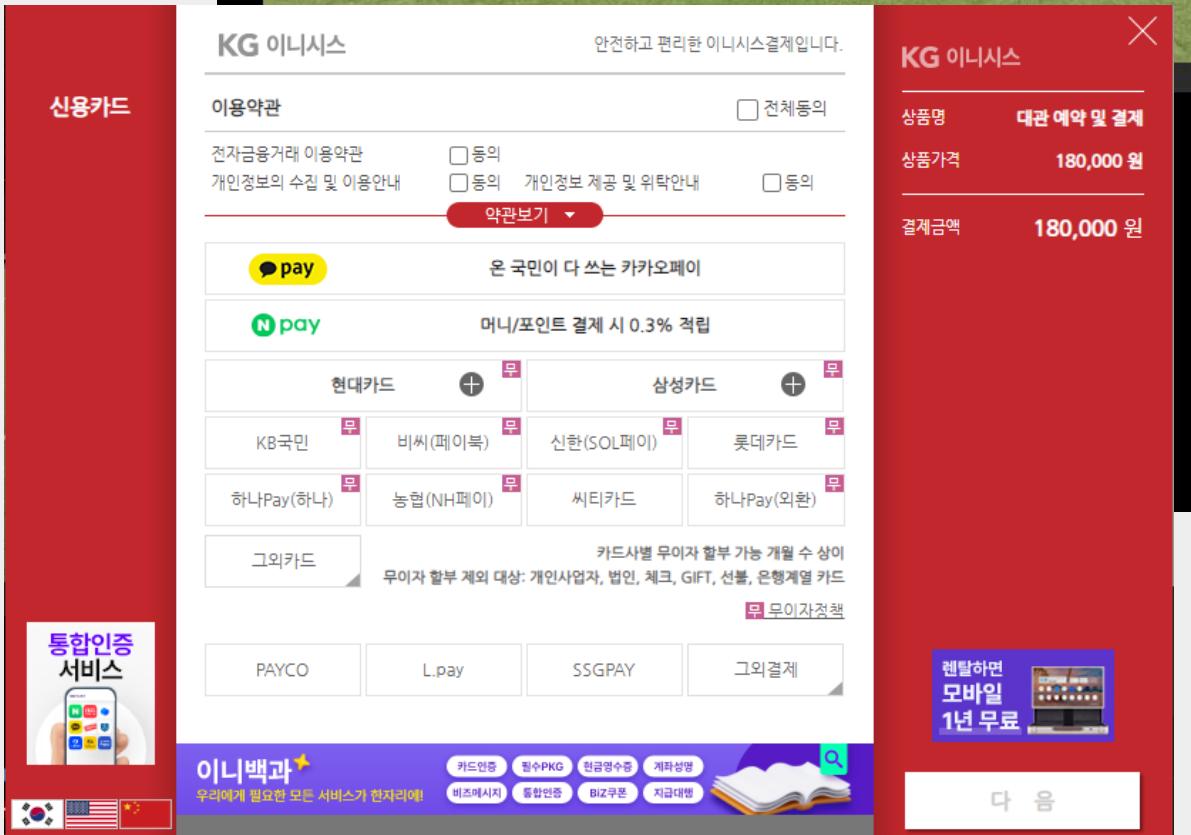
주요 기능과 코드 (구장예약 및 결제)



대관예약

선택된 날짜: 2024-06-08

2024년 6월						
일	월	화	수	목	금	토
1						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						



KG 이니시스

안전하고 편리한 이니시스결제입니다.

이용약관 전체동의
 전자금융거래 이용약관 동의
 개인정보의 수집 및 이용안내 개인정보 제공 및 위탁안내 동의
 약관보기

KG 이니시스

상품명 대관 예약 및 결제
 상품가격 180,000 원
 결제금액 180,000 원

대관 예약자 정보

신청자	지민
연락처	1231
예약일자	2024-06-08
예약시간	10:00-11:00, 11:00-12:00, 12:00-13:00
총 결제금액	180000원
별도	

Windows 정품
 [설정]으로 이동하라

예약 및 결제

1. 원하는 날짜를 선택하고 시간을 연속적으로 선

```
// 시간선택하면 선택확인 및 선택취소 관련 기능
function toggleTimeSelection(cell) { Show usages ⌘ ledjk99
  var timeText = cell.innerText;
  var index = selectedTimes.indexOf(timeText);
  if (index > -1) {
    // 이미 선택된 시간일 때 더블 클릭하면 선택 취소
    if (cell.classList.contains("selected")) {
      selectedTimes.splice(index, 1);
      cell.style.backgroundColor = ""; // 선택 해제 시 배경색 초기화
      cell.classList.remove("selected");
    } else {
      // 더블 클릭 시 선택 취소
      cancelTimeSelection(timeText);
    }
  } else {
    selectedTimes.push(timeText);
    selectedTimes.sort();
    if (selectedTimes.length > 1 && !isConsecutive(selectedTimes)) {
      Swal.fire({
        icon: 'warning',
        title: '연속된 시간을 선택해주세요.',
        text: '선택한 시간은 연속적이어야 합니다.',
        confirmButtonText: '확인'
      }).then(() => {
        selectedTimes.pop(); // 연속되지 않으면 마지막 선택을 취소
        updateSelectedTimes(); // 업데이트된 시간 다시 표시
      });
    } else {
      cell.style.backgroundColor = "lightblue"; // 선택 시 배경색 변경
      cell.classList.add("selected");
    }
  }
  updateSelectedTimes();
}
```

2. 결제 API를 이용하여 결제 진행

주요 기능과 코드 (팀가입신청)

가입 신청 Club Join

이름	민경찬
연락처	01012341234
이메일	rudcks@naver.com
수준	중상
소개	30대 직장인입니다. 학생선출입니다.

가입신청

1. 로그인 정보를 가져와 회원 정보에 맞게 신청

```
@PostMapping("/join")
public String clubJoinRequest(@RequestParam("clubNum") String clubNum,
                             @RequestParam String memberLevel,
                             @RequestParam String memberInfo,
                             HttpSession session,
                             Model model) {
    // 선택한 클럽번호 가져오기
    String getClubNum = (String) session.getAttribute("clubNum");

    // 로그인 된 사용자의 정보 가져오기
    MemberDto loginMember = (MemberDto) session.getAttribute("loginMember");

    // 클럽장에게 전달할 정보 생성
    ClubJoinListDto clubJoinRequest = new ClubJoinListDto();

    // 멤버넘버 식별
    clubJoinRequest.setMemberNum(Integer.parseInt(loginMember.getMemberNum()));

    // 전달할 정보(디비정보)
    clubJoinRequest.setClubNum(Integer.parseInt(getClubNum));
    clubJoinRequest.setJoinMemberPhoto(loginMember.getStoredProfile());
    clubJoinRequest.setJoinMemberName(loginMember.getName());
    clubJoinRequest.setJoinMemberPhone(loginMember.getPhone());
    clubJoinRequest.setJoinMemberEmail(loginMember.getEmail());

    // 전달할 정보(신청자 작성)
    clubJoinRequest.setJoinMemberLevel(memberLevel);
    clubJoinRequest.setJoinMemberInfo(memberInfo);

    // 클럽장에게 정보 전달
    clubService.clubJoinRequest(clubJoinRequest);

    return "redirect:/club/list";
}
```

가입 신청 관리

시퀀스	이름	연락처	이메일	수준	소개	상태	승인	거절
1	민경찬	01012341234	rudcks@naver.com	중상	30대 직장인입니다. 학생선출입니다.	대기	승인	거절

2024-06-04
치킨 | 치킨드시면서 할분

가입 신청 관리

2. 가입 신청 관리 버튼은 클럽장에게만 노출

```
function fetchJoinRequests() : void { 사용 위치 표시 신규*
const clubNum = document.querySelector(selectors: "#clubNum").innerText;
fetch( input: `/club/joinList?clubNum=${clubNum}` , init: {
    method: 'GET',
}) Promise<Response>
.then<Response>(response : Response => {
    if (!response.ok) {
        throw new Error('Network response was not ok');
    }
    return response.json(); // Parse JSON response
}) Promise<any>
.then(data => {
    updateModalContent(data);
    let table : HTMLElement = document.getElementById(elementId: "joinRequestsTable");

    let tr : string = ``;
    data.forEach(request => {
        if (request.joinStatus === '대기') {
            tr += `<tr>`;
            tr += `<td></td>`;
            tr += `<td>${request.joinMemberName}</td>`;
            tr += `<td>${request.joinMemberPhone}</td>`;
            tr += `<td>${request.joinMemberEmail}</td>`;
            tr += `<td>${request.joinMemberLevel}</td>`;
            tr += `<td>${request.joinMemberInfo}</td>`;
            tr += `<td>${request.joinStatus}</td>`;
            tr += `<td><button class="approve-btn" data-join-num="${request.joinNum}">승인</button></td>`;
            tr += `<td><button class="refuse-btn" data-join-num="${request.joinNum}">거절</button></td>`;
            tr += `</tr>`;
        }
    });
    table.innerHTML = tr;
}

// 승인 및 거절 버튼에 이벤트 리스너 추가
document.querySelectorAll(selectors: ".approve-btn").forEach(callbackfn: button : Element => {
    button.addEventListener(type: 'click', listener: () : void => handleJoinRequest(button.dataset.joinNum, action: 'approve'));
});

document.querySelectorAll(selectors: ".refuse-btn").forEach(callbackfn: button : Element => {
    button.addEventListener(type: 'click', listener: () : void => handleJoinRequest(button.dataset.joinNum, action: 'refuse'));
}) Promise<void>
.catch(error => {
    console.error('There was a problem with the fetch operation:', error);
});
```

3. Fetch를 이용 json형식으로
DB에서 저장된 데이터를
가져옴

```
@GetMapping("/joinList")
@ResponseBody
public List<ClubJoinListDto> clubJoinView(@RequestParam("clubNum") String clubNum) {
    return clubService.clubJoinView(clubNum);
}
```

```
@PostMapping("/joinApprove")
@ResponseBody
public ResponseEntity<Void> joinApprove(@RequestParam("joinNum") int joinNum) {
    // STATUS 업데이트
    clubService.clubJoinApprove(joinNum);
    // CLUB_NUM 업데이트
    memberService.clubJoinUpdateClubNum(joinNum);
    return ResponseEntity.ok().build();
}
```

```
@PostMapping("/joinRefuse")
@ResponseBody
public ResponseEntity<Void> joinRefuse(@RequestParam("joinNum") int joinNum) {
    clubService.clubJoinRefuse(joinNum);
    return ResponseEntity.ok().build();
}
```

단일 책임 원칙에 위배되지 않게
각 기능의 메소드를 나눔

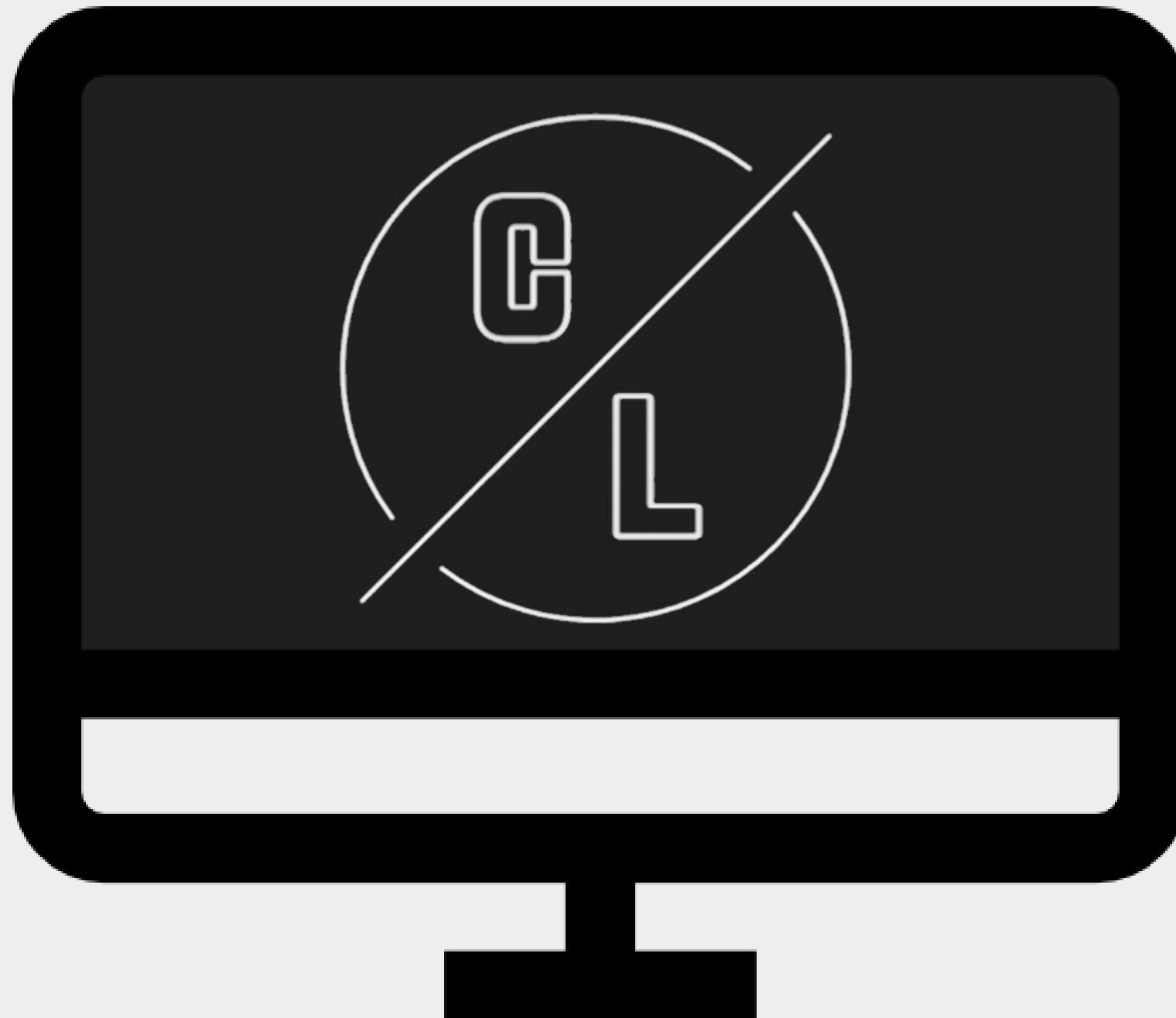
4. 모달창으로 목록을 보이고
데이터에 맞게 html 요소 추가

5. 비동기 통신을
통해
손쉬운 관리 가능

6. 프로젝트 시연



프로젝트 시연



7. 프로젝트 마이킹

리뷰



메이킹 리뷰

신연재(팀장)

수업시간에 배운 내용을 바탕으로
실습해보지 않은 기능들을 구현하려니
막막했는데
천천히 공부했던 내용을 복습하며 하니
공부에도 도움이 되고 구현도 모두 완료해서
뿌듯한 프로젝트였고, 같이 작업한 팀원들도
너무 잘해줘서 감사했습니다

민경찬

프로젝트를 진행하며 기능도 많았고 디자인도 직접
했기에 시간이 부족했지만 하나하나 차근차근
진행하며 팀 프로젝트라는 것에 대한 이해를 할 수
있었다.

우리 조의 아이디어는 평범하면서도 특별한
내용이었기에 프로젝트를 즐겁게 진행했고,
그 중 제가 맡은 업무는 수업시간에 배웠던 내용을
다양하게 활용할 수 있어서 기본기를 다지며
복습하기에 좋은 시간이었습니다.

박 솔

이번에 욕심이 나서 좀 더 다양한 기능이 많은
페이지를 맡아 걱정이 앞섰지만 하나하나
배우는 마음으로, 차근차근 기능 하나씩
만들어서 프로그래밍의 이해도가 높아졌다.
결과로 하나의 페이지를 구현 했다는 게
뿌듯하고 돌아보면 이번 프로젝트는 좋은
팀원들과 함께라 재밌었던 것 같다.

윤인록

이번 파이널 프로젝트를 하면서 아예 배우지
않았던 부분을 맡았었다 보니 우여곡절이
많았는데

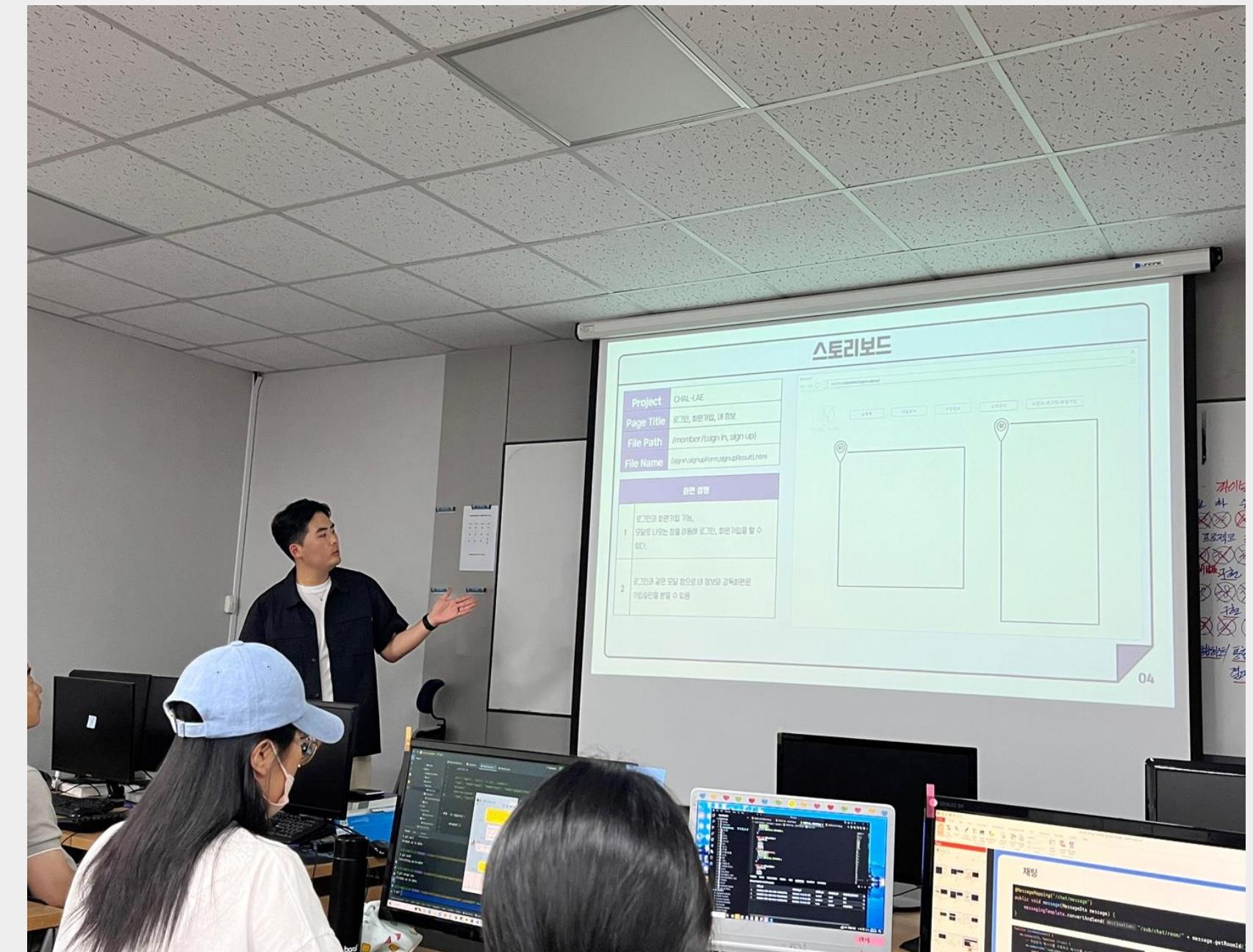
그래도 새로운 걸 찾아가며 코드를 만들다
보니 새롭게 알게 된 점이 많아서 좋았고
팀원들도 모두 열심히 프로젝트 참여해줘서
뜻 깊은 시간이었습니다.

이덕재

이번 파이널 프로젝트를 하면서
기본기를 다시 복습할 수 있는 시간이었고
추가적으로 잘 몰랐던 부분들도
공부하면서 알 수 있어서 뜻 깊었던
시간이었습니다.

제가 많이 부족했던 부분을 팀장님과
팀원분들의 도움으로 프로젝트를 무사히 마칠
수 있었던 것 같습니다.

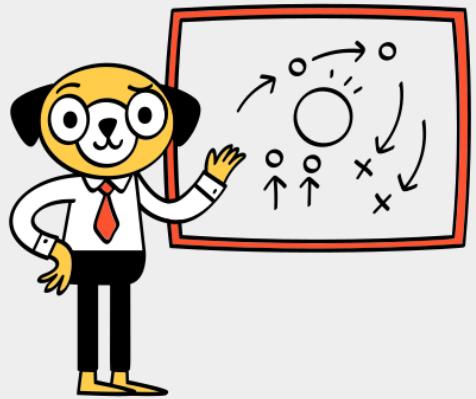
메이킹 리뷰



8. Q&A



질문주세요



요!