

캡스톤디자인 졸업작품 프로젝트 보고서

<< 프로젝트명 : 컨비홈(ConviHome) >>

IT콘텐츠학부 소프트웨어융합과

지도교수 : 장은겸 교수님

팀명 : Y2K

제출자 : 20221043 김영민

: 20211304 양하늘

: 20221024 김지오

제출일 : 2024년 12월 13일

프로젝트 보고서 제출문

프로젝트 명	컨비홈(ConvHome)		
팀원	김영민, 양하늘, 김지오	팀명	Y2K
분야	Spring기반 Web Application		
소속	장안대학교 소프트웨어융합과	지도교수	장은겸
주요산출물	<div>1. 프로젝트 계획</div> <div>1.1 개요</div> <div>1.2 목표</div> <div>1.3 도메인분석</div> <div>1.4 일정계획</div> <div>1.5 예산계획</div> <div>1.6 개발환경</div> <div>2. 요구사항 조사</div> <div>2.1 Usecase Diagram</div> <div>2.2 Uscase 명세서</div> <div>3. 프로젝트 분석</div> <div>3.1 Class Diagram</div> <div>3.2 Sequence Diagram</div> <div>4. 프로젝트 설계</div> <div>4.1 메뉴 구조도</div> <div>4.2 화면설계도</div> <div>4.3 ER Diagram</div> <div>5. 중심 클래스 구조 및 소스 분석</div> <div>6. 사용자 메뉴얼</div>		
<div>이 보고서를 2024학년도 IT콘텐츠학부 소프트웨어융합과 캡스톤디자인 졸업작품 프로젝트 완료 보고서로 제출합니다.</div> <div>2024년 12월 13일</div> <div>프로젝트 담당자 : 김영민 (인) 양하늘 (인) 김지오 (인)</div> <div>소프트웨어융합과 학과장 귀하</div>			

목차

1. 프로젝트 계획	
1.1 프로젝트 개요	2
1.2 프로젝트 목표	3
1.3 도메인분석	4
1.4 프로젝트 일정계획	5
1.5 프로젝트 예산계획	6
1.6 개발환경	6
2. 요구사항 조사	
2.1 산업체 요구사항	7
2.2 Usecase Diagram	9
2.3 Usecase 명세서	10
3. 시스템 분석	
3.1 Class Diagram	20
3.2 Sequence Diagram	25
4. 시스템 설계	
4.1 메뉴 구조도	29
4.2 데이터베이스 설계	30
5. 중심 클래스 구조 및 소스 분석	32
6. 구현 화면	
6.1 Web	42
7. 프로젝트 후기	51
8. 참고 자료 및 소스 CD	52

1. 프로젝트 계획

1.1 프로젝트 개요

이 작품은 공동주택에 거주하는 사람들의 불편함과 고민을 덜어주고, 삶의 질을 향상시키기 위한 공동주택 편의 서비스다. 공동주택에서 자주 발생하는 층간소음 문제, 이웃 간의 소통 부족, 관리비 사용처에 대한 불신등의 문제를 해결하기 위한 프로젝트이다. 이 서비스는 게시판, 관리비 조회, 편의시설 예약, 공지사항 캘린더 등의 기능을 통해 입주민 간의 신뢰를 형성하고, 효율적인 주거 관리 기능을 제공한다.

먼저, 게시판 기능은 입주민들이 자유롭게 의견을 교환하고, 이웃 간의 이해관계를 형성할 수 있는 소통의 공간을 제공한다. 게시판은 자유게시판, 공지 사항, 입주민 간 정보공유, 물건 나눔 등 다양한 카테고리로 나뉘어 있어 목적에 맞는 정보를 쉽게 확인하고 공유하는 기능을 제공한다.

공지사항과 일정을 한눈에 볼 수 있는 캘린더 기능도 포함되어 있어, 공동주택에서 진행되는 행사나 공지를 쉽게 확인할 수 있다. 사용자는 자신의 일정과 커뮤니티 일정을 동시에 관리할 수 있으며, 관리자는 아파트 전체 공지사항을 등록하고 관리할 수 있는 기능을 추가로 제공한다.

이 서비스는 사용자 친화적인 UI/UX를 갖추고 있어 웹에서 쉽게 이용할 수 있으며, 데이터 기반의 투명성을 통해 입주민과 관리자 간의 신뢰를 증진한다. 나아가, 입주민 간의 소통 활성화와 공동체 의식 형성에 기여하며, 관리 투명성을 확보하고 생활의 효율성을 높인다. 더불어, 다양한 편의 기능을 통해 입주민들이 일상에서 겪는 불편함을 줄이고, 안전하고 편리한 생활 환경을 제공한다.

관리비 조회 기능은 공동주택 입주민들에게 관리비 내역을 투명하게 제공함으로써 관리비와 관련된 불신과 갈등을 해소하는 데 중점을 둔다. 입주민들은 이 기능을 통해 자신이 부담한 관리비가 어떤 항목에 사용되었는지 상세하게 확인할 수 있다. 예를 들어, 공용 전기료, 청소비, 시설 유지보수비 등 세부 내역이 명확히 공개되며, 이를 통해 관리비 사용 내역에 대한 입주민들의 신뢰를 구축할 수 있다. 관리비 조회 기능은 공동주택 거주자들의 만족도를 높이고, 더 나은 주거 환경을 조성하는 데 핵심적인 역할 기능을 제공한다.

이처럼 본 작품은 공동주택 거주자들의 다양한 니즈를 충족시키며, 거주 환경을 개선하고 더 나은 공동체 문화를 조성하는데 기여할 수 있는 종합적인 서비스다.

1.2 프로젝트 목표

□ 사용자 요구사항 정의서 (Web 페이지)

구분	업무(예시)	주요 기능
기능적 요구사항	예약	<ul style="list-style-type: none"> 주요기능 헬스장, 독서실 등 공동주택 내 시설 예약을 가능하게 한다. 예약을 완료한 후에 개인 캘린더에 예약일과 시간이 표시된다.
	게시판	<ul style="list-style-type: none"> 공동주택 입주민들이 관리소 및 다른 입주민들과 소통하고, 공지사항, 자유게시판, 건의사항 등을 통해 정보를 공유하며 커뮤니티를 활성화할 수 있도록 지원하는 게시판 기능을 제공한다.
	관리비	<ul style="list-style-type: none"> 입주민들이 개인별 관리비 내역과 납부 현황을 쉽게 확인할 수 있는 시스템을 제공함으로써 투명하고 편리한 관리비 관리를 지원한다. 이를 통해 납부 내역 확인, 청구된 금액의 상세 내역 파악, 미납 내역 관리 및 납부 계획 수립을 가능하게 한다.
	캘린더	<ul style="list-style-type: none"> 입주민들이 개인 일정을 관리하고, 시설 예약, 공용 일정 확인 및 참여, 개인 일정을 기록할 수 있도록 지원하는 직관적이고 사용하기 쉬운 캘린더 기능을 제공한다.
비기능적 요구사항	시스템 성능	<ul style="list-style-type: none"> 최소 사양 - 팬티엄 4 CPU :3.60GHz - RAM :8GB - HDD :341GB
	유지보수	<ul style="list-style-type: none"> 프로그램에 오류가 생길 경우 즉시 관리자에게 연락한다.
	인터페이스	<ul style="list-style-type: none"> 운영체제는 window10을 권장하며 DB는 사용 가능한 SQL 10g을 권장한다.
	신뢰성	<ul style="list-style-type: none"> 시스템이 멈추거나 다운 되어서는 안된다. 시스템은 사용자의 요청에 정확하게 응답할 수 있어야 한다
	보안성	<ul style="list-style-type: none"> 미리 등록된 사용자만이 시스템에 접근할 수 있어야 한다.
	가용성	<ul style="list-style-type: none"> 시스템은 점검시간을 제외한 24시간 서비스를 제공해야 한다.

1.3 도메인 분석

아파트리

장점

편리한 열람 기능

관리비 조회, 아파트 소식·민원 등을 편리하게 열람 가능

관리비 관련 편의

월 별 관리비 조회 및 납부 가능
관리비를 차감해서 낼 수 있는
포인트 적립 (A-Point)

아파트아이

단점

익명 보장 X

익명 제도가 없어 입주민들이
표현의 자유를 억압 받을 수 있음
의견 공유가 제대로 이루어 지지
않을 가능성이 높음

소통 공간 알림 X

소통 공간의 알림이 활성화되지 않아
존재 의미가 흐릿해짐

보완

익명 보장

서비스 이용 시 거주 중인
공동주택의 '동'만 표기
옆에 닉네임 기재로 익명성 확보

알림 기능 설정 활성화

게시판 이용을 비롯한
프로젝트 내의 모든 활동에
알림 ON/OFF 설정



	아파트리	아파트아이	컨비홈
관리비 조회 및 납부	O	O	O
단지 내 편의시설 예약	O	O	O
게시판 이용	O	O	O
통합 캘린더	X	X	O

1.4 프로젝트 일정 계획

단계	주차 테스크	5월 4주	6월 1주	6월 2주	6월 3주	6월 4주	7월	8월	9월	10 월	11 월 1주	11 월 2주	11 월 3주	11 월 4주	12 월 1주	12 월 2주
계획	요구분석															
	업무정의서															
	디자인 시안															
요구사항	유스케이스 다이어그램															
	유스케이스 명세서															
분석	클래스다이어 그램															
	시퀀스 다이어그램															
설계	인터페이스 설계															
	데이터베이스 설계															
구현/테스 트	프로그래밍/ 디자인															
	테스트 계획 및 결과															
문서작성	사용자매뉴얼															
	보고서															

1.5 프로젝트 예산 계획

구분	소프트웨어 노임단가	투입인원	한달일수	투입기간	금 액(원)
초급 기술자	197,212 원	3 명	22 일	8 개월	104,127,936원
직접인건비 합계					104,127,936원
재경비(직접인건비의 110% ~ 120%)					119,747,126원
기술료([직접인건비 + 재경비]의 20 ~ 40%)					67,162,519원
합 계 (부가세 별도)					291,037,581원

Y2K 가상 예산계획안

(소프트웨어사업대가산정가이드, 2023 KOSA)

1.6 개발 환경

1) 하드웨어

구분	하드웨어	비고
CPU	Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz 2.90 GHz	
RAM	32.0GB	
HDD	1.8TB	
Memory	32GB	
Storage	SSD 500GB + HDD 1.8TB	

2) 소프트웨어

구분	소프트웨어	비고
프로그램 구현	sts4	
문서작성	한글	
운영체제	window10	

2. 요구사항 조사

2.1 산업체 요구사항

최근 공동주택에서 제공되는 다양한 디지털 서비스는 거주자들의 생활 편의를 크게 향상시키고 있다. 관리비 납부, 편의시설 예약, 일정 관리, 주민 간 소통 등이 대표적인 서비스로, 이러한 기능들이 원활하게 운영되기 위해서는 다양한 기술적 요구사항과 함께 사용자 편의성 및 보안성을 확보하는 것이 필수적이다. 그러나 기존의 공동주택 서비스에서는 여러 가지 문제점들이 발생하고 있어, 이러한 문제들을 해결할 수 있는 기능적 개선이 요구된다.

회원이입 및 로그인 시스템은 공동주택 서비스에 접속하는 첫 번째 단계로, 거주자가 손쉽게 서비스를 이용할 수 있도록 해야 한다. 또한 세대원과 세대주 각각의 역할을 고려한 로그인 방식도 중요하다. 세대원은 세대주가 아닌 다른 가족 구성원으로서 간편하게 서비스를 이용할 수 있도록 해야 한다. 자동 로그인 기능을 제공하여 사용자가 매번 로그인하지 않고도 서비스를 편리하게 이용할 수 있게 해야 한다.

관리비 조회 및 납부 시스템은 공동주택 거주자들에게 실시간으로 관리비 내역을 확인하고, 이를 효율적으로 관리할 수 있는 기능을 제공한다. 거주자는 언제든지 관리비 내역을 확인하고, 불필요한 지출을 피할 수 있도록 해야 하며, 결제 방법 역시 카드 결제, 계좌 이체, 모바일 결제 등 다양한 수단을 제공하여 사용자 편의성을 높여야 한다. 그러나 기존 시스템에서는 결제 오류가 발생할 수 있으며, 결제 정보가 지연되거나 결제 처리에서 오류가 발생할 경우 사용자가 불편을 겪을 수 있다. 결제 수단이 제한적일 경우 일부 사용자에게는 불편을 초래할 수 있으며, 다양한 결제 방법을 제공해야 한다는 점도 중요한 문제다.

공동주택 내 편의시설(헬스장, 독서실 등)의 예약 시스템은 여러 사용자가 동시에 예약을 시도할 때 중복 예약을 방지하고, 실시간 예약 현황을 제공하여 사용자들이 효율적으로 시설을 이용할 수 있도록 해야 한다. 예약 시스템은 사용자가 원활하게 예약을 취소하거나 변경할 수 있는 기능도 제공해야 한다. 하지만 여러 사용자가 동시에 예약을 시도할 경우 중복 예약이 발생할 수 있으며, 예약 시스템의 오류로 인해 예약이 제대로 반영되지 않는 경우도 있을 수 있다. 또한 예약 취소나 변경 절차가 복잡하면 사용자 경험이 저하되므로, 예약 시스템을 간편하고 직관적으로 개선할 필요가 있다.

공동주택 내에서 중요한 일정(관리비 납부일, 회의 일정, 주민 행사 등)을 효율적으로 관리할 수 있는 캘린더 기능은 필수적입니다. 이를 통해 거주자는 중요한 일정을 한눈에 확인하고 계획을 세울 수 있다.

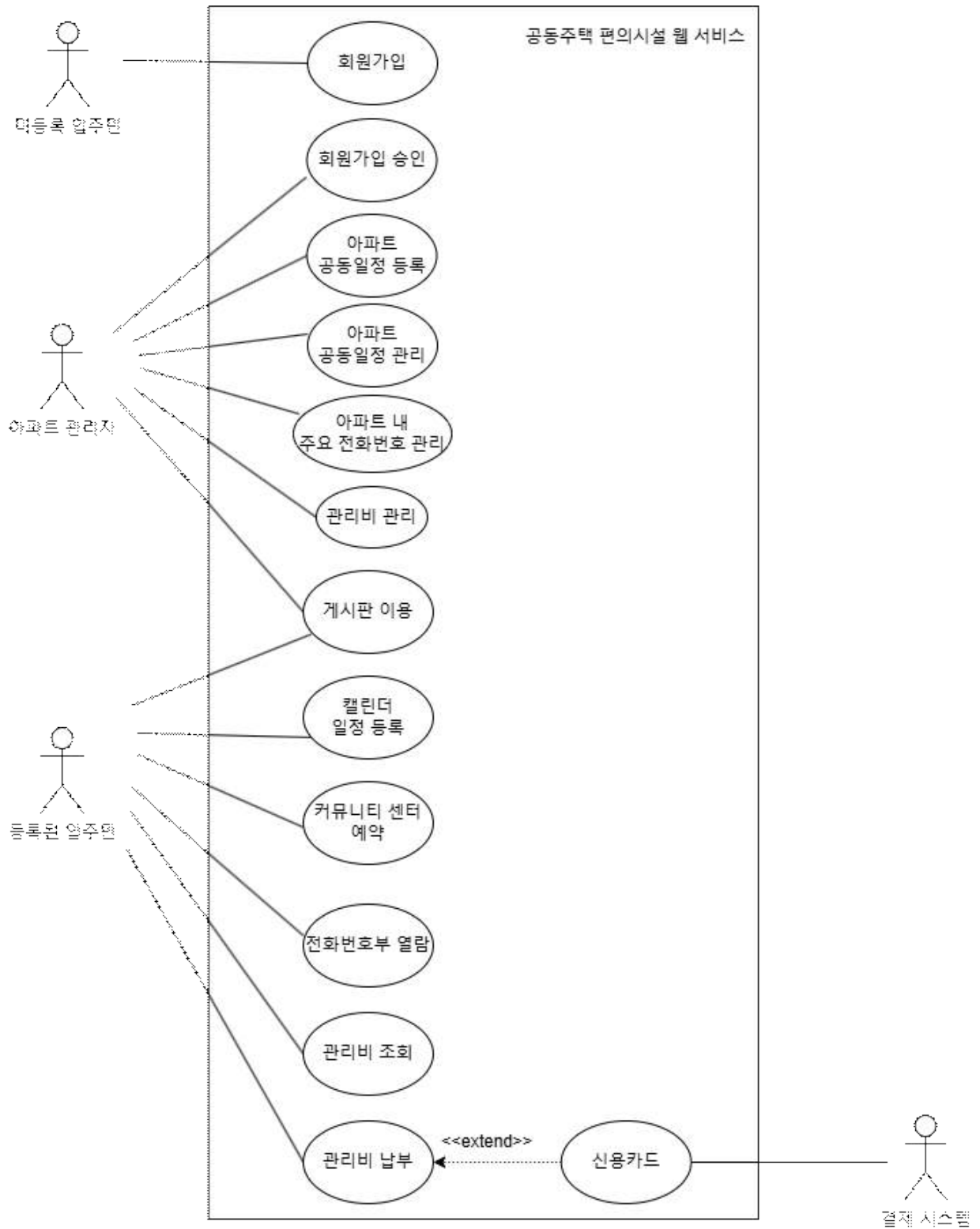
게시판은 공동주택 내에서 주민들이 중요한 정보를 공유하고, 공지사항을 확인하는 데 중요한 역할을 한다. 또한 주민들이 자유롭게 의견을 나누고 커뮤니티를 활성화하는 데 도움을 준다. 관리자는 게시판에 올라오는 게시물을 검토하고 불법적인 게시물이나 부적절한 내용이 게시되지 않도록 해야 한다. 따라서 관리자 제어 기능을 통해 게시물을 적절하게 관리

할 수 있는 시스템이 필요하다.

스팸 글이나 불법적인 게시물이 올라올 수 있기 때문에 이를 관리자가 적절하게 처리하지 못하면 커뮤니티의 신뢰도에 문제가 발생할 수 있다. 또한, 게시판 내에서 효율적인 검색 기능이 부족하면 사용자가 필요한 정보를 찾는 데 어려움을 겪을 수 있다. 이를 해결하기 위해서는 검색 기능과 필터링 시스템을 개선하고, 불법적인 게시물에 대한 빠른 처리가 가능하도록 해야 한다.

공동주택의 편의 서비스는 거주자들의 생활 편의를 위한 중요한 요소다. 그러나 현재 시스템에서는 여러 가지 기술적 문제와 사용자 경험에 대한 미비점이 존재한다. 각 서비스의 기능을 개선하고, 사용자 편의성 향상, 문제 해결을 위한 기술적 접근이 반드시 필요하다. 산업체는 이러한 요구사항을 반영하여, 효율적이고 안전한 시스템을 구축함으로써 공동주택 서비스의 품질을 높이고, 거주자들에게 보다 나은 서비스를 제공할 수 있을 것이다.

2.2 Usecase Diagram (Web 페이지)



2.3 유스케이스 명세서

□ '세대주 회원가입' 유스케이스 명세서

유스케이스명	회원가입
액터	세대주
개요	등록되지 않은 사용자가 서비스를 이용하기 전 회원 등록을 진행한다.
사전 조건	회원에 가입되어 있지 않은 상태여야 한다.
정상 흐름	<ol style="list-style-type: none"> 1. 시스템에서 회원정보 입력 항목을 보여준다. 2. 사용자는 회원정보(아파트명, 동, 호수, 이름, 핸드폰번호 등)을 입력하고 등록 버튼을 클릭한다. <ol style="list-style-type: none"> 2-1. 입주민 정보가 이미 존재할 경우 a로 돌아간다. 2-2. 입주민 정보가 실체와 일치하지 않을 경우 b로 돌아간다. 3. 시스템은 입력된 정보를 확인하고 데이터베이스에 회원 정보를 저장한다.
선택 흐름	<ol style="list-style-type: none"> a. 기존에 가입되어 있는 회원인 경우 "가입된 회원"이라는 메시지를 보여준다. b. 입주민 인증을 실패하거나 꼭 입력해야 하는 정보를 입력하지 않는다면 오류 메시지를 출력하고 1번으로 돌아간다.

□ '세대원 회원가입' 유스케이스 명세서

유스케이스명	회원가입
액터	세대원
개요	등록되지 않은 사용자가 서비스를 이용하기 전 회원 등록을 진행한다.
사전 조건	회원에 가입되어 있지 않은 상태여야 한다.
정상 흐름	<ol style="list-style-type: none"> 1. 시스템에서 회원 정보 입력 화면을 보여준다. 2. 사용자는 회원 정보(입주민 인증, 이름, 주민등록번호, 핸드폰 번호, 이메일, 세대주, 승인 코드 등)을 입력하고 등록 버튼을 클릭한다. <ol style="list-style-type: none"> 2-1. 이미 가입된 경우 a로 돌아간다. 2-2. 세대주명과 승인코드가 일치하지 않을 경우 b로 돌아간다. 3. 시스템은 입력된 정보를 확인하고 세대주 명과 승인 코드가 일치하면 데이터베이스에 회원 정보를 저장한다.
선택 흐름	<ol style="list-style-type: none"> a. 기존에 가입되어 있는 회원인 경우 "가입된 회원"이라는 메시지를 보여준다. b. 세대주명과 승인코드가 일치하지 않거나 필수 입력 정보를 입력하지 않으면 오류 메시지를 출력하고 2번으로 돌아간다.

□ '로그인' 유스케이스 명세서

유스케이스명	로그인
액터	입주민 및 관리자
개요	등록된 사용자가서비스를 이용하기 전 로그을 진행한다.
사전 조건	회원에 가입되어 있는 상태여야 한다.
정상 흐름	<ol style="list-style-type: none"> 1. 사용자가 아이디와 비밀번호를 입력한다. 2. 시스템은 데이터베이스에 저장되어 있는 아이디와 비밀번호를 대조한다. 2-1. 일치하지 않을 경우 a로 돌아간다. 3. 아이디와 비밀번호가 일치하면 자동으로 메인 페이지로 이동한다.
선택 흐름	<ol style="list-style-type: none"> a. 아이디나 비밀번호가 일치하지 않으면 오류 메시지를 출력하고 1번으로 돌아간다.

□ '캘린더' 유스케이스 명세서

유스케이스명	캘린더
엑터	입주민 및 관리자
개요	입주민 개인 캘린더에서 커뮤니티 센터 예약 정보와 단지 일정을 함께 확인한다.
사전 조건	회원에 등록되어있으며 로그인 된 상태여야 한다.
정상 흐름	<ol style="list-style-type: none"> 1. 아파트 내 일정이 발생한다. 2. 관리자는 캘린더에 들어가 일정의 해당 날짜를 선택한다. 3. 관리자는 일정의 제목, 세부 내용을 입력한다. 3-1. 일정의 제목, 내용 모두를 입력하지 않으면 a로 돌아간다. 4. 입주민은 캘린더에 들어가 일정을 확인한다.
선택 흐름	<ol style="list-style-type: none"> a. 관리자가 제목을 입력하지 않았을 때 오류 메시지와 함께 3번으로 돌아간다.

□ '관리비 등록 및 조회' 유스케이스 명세서

유스케이스명	관리비 등록 및 조회
엑터	관리자
개요	항목 별, 월 별 관리비를 등록한다.
사전 조건	아이디와 비밀번호를 부여 받은 관리자여야 한다.
정상 흐름	<ol style="list-style-type: none"> 1. 엑셀에 관리비를 부여할 동, 호수, 관리비 유형,금액을 입력한다. 1-1. 양식에 벗어난 값을 입력할 경우 a로 돌아간다. 2. 관리비 탭에 등록 버튼을 클릭해 파일을 등록한다. 3. 수정 사항이 발생할 경우 수정을 눌러 진행한다. 4. 추가 사항이 발생할 경우목록에서 등록을 눌러 개별 등록을 진행한다. 4-1.양식에 벗어난 값을 입력할 경우 a로 돌아간다.
선택 흐름	<ol style="list-style-type: none"> a. 입력 값이 양식을 벗어날 경우 관리비를 재입력한다.

□ '관리비 조회 및 납부' 유스케이스 명세서

유스케이스명	관리비 조회 및 납부
액터	입주민
개요	관리비를 조회하고 납부 방식을 선택해 진행한다.
사전 조건	등록된 입주민이어야 하며 로그인 되어야 한다.
정상 흐름	<ol style="list-style-type: none"> 1. 입주민이 '관리비 조회 및 납부'를 선택한다. 2. 입주민에 대한 월별 관리비 리스트를 제공한다. 3. 당월 관리비를 납부하고자 하면 납부 버튼 클릭한다. 4. 결제 창이 띄워지며 결제를 진행한다. 4-1. 결제에 실패 시 a로 돌아간다. 5. 결제가 완료되면 납부는 완납으로 바뀌어 영수증을 다운 받는다.
선택 흐름	<ol style="list-style-type: none"> a. 결제에 실패할 경우 오류 메시지와 함께 4번으로 돌아간다.

□ '편의시설 예약' - '헬스장' 유스케이스 명세서

유스케이스명	편의시설 (헬스장) 예약
엑터	입주민
개요	아파트 내 헬스장 이용 전 예약을 진행한다.
사전 조건	회원에 등록되어있으며 로그인 된 상태여야 한다.
정상 흐름	<ol style="list-style-type: none"> 1. 사용자는 커뮤니티 예약을 클릭해 예약할 커뮤니티 센터를 선택한다. 2. 사용자는 이용할 날짜 및 이용 개월 수를 선택한다. 3. 예약이 완료되면 관리비에 해당 이용료가 추가된다.
선택 흐름	없음

□ '편의시설 예약' - '독서실' 유스케이스 명세서

유스케이스명	편의시설 (독서실) 예약
액터	입주민
개요	아파트 내 독서실 이용 전 예약을 진행한다.
사전 조건	회원에 등록되어있으며 로그인 된 상태여야 한다.
정상 흐름	<ol style="list-style-type: none"> 1. 사용자는 편의시설 예약을 클릭해 독서실 예약을 클릭한다. 2. 사용자는 좌석 및 입/퇴실 시간을 선택한다. 2-1. 이미 예약된 좌석이나 시간을 선택할 경우 a로 돌아간다. 3. 예약 완료 후 접수된 예약 정보를 확인한다.
선택 흐름	<ol style="list-style-type: none"> a. 이미 예약된 좌석이나 이용시간을선택할 경우 오류 메시지와 함께 2로 돌아간다.

□ '게시판' 유스케이스 명세서

유스케이스명	게시판
액터	입주민 및 관리자
개요	입주민끼리 카테고리 별 게시판 글, 댓글을 등록하고 열람한다.
사전 조건	회원에 등록되어 있는 상태여야 한다.
정상 흐름	<ol style="list-style-type: none"> 1. 게시판을 클릭해 카테고리별로 분류하거나 좋아요, 댓글을 남기는 등 다른 사용자들이 올린 게시판을 확인한다. 2. 사용자는 게시글을 등록할 때 카테고리, 제목, 글, 파일, 그림 등을 작성해 등록한다. <ol style="list-style-type: none"> 2-1. 카테고리, 제목, 내용을 모두 입력하지 않을 경우 a로 돌아간다. 3. 사용자가 작성한 게시글을 다른 사용자가 열람한다.
선택 흐름	<ol style="list-style-type: none"> a. 게시글을 등록할 때 카테고리, 제목, 글을 입력하지 않으면 입력해 달라는 메시지를 출력한다.

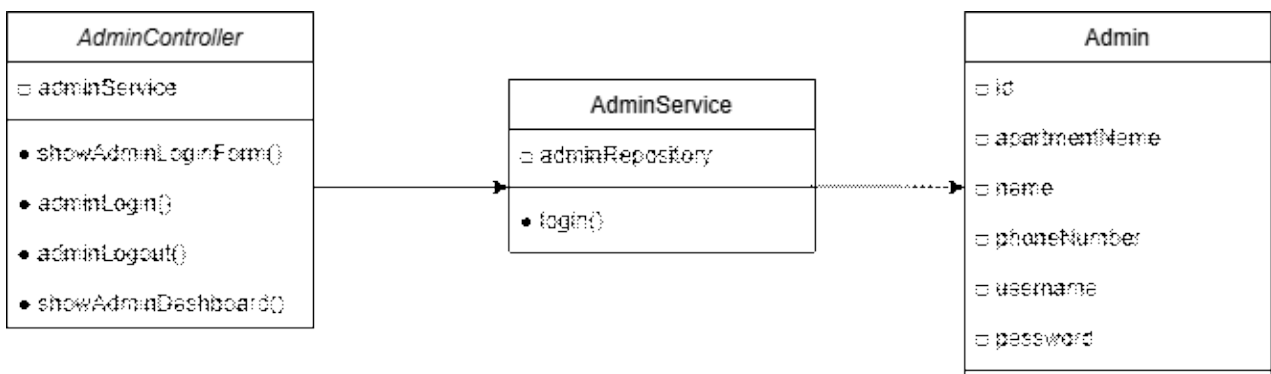
□ '아파트 내 전화번호 목록' 유스케이스 명세서

유스케이스명	아파트 내 전화번호 목록
엑터	입주민
개요	아파트 내 시설 관련 전화번호와 관리사무소 번호 목록을 나타낸다.
사전 조건	등록된 입주민이어야 하며 로그인 되어야 한다.
정상 흐름	<ol style="list-style-type: none"> 1. 입주민이 '아파트 내 전화번호 목록'을 선택한다. 2. 등록된 아파트 내 전화번호 리스트 보여준다. 3. 입주민과 시설 담당자와 통화 진행한다.
선택 흐름	없음

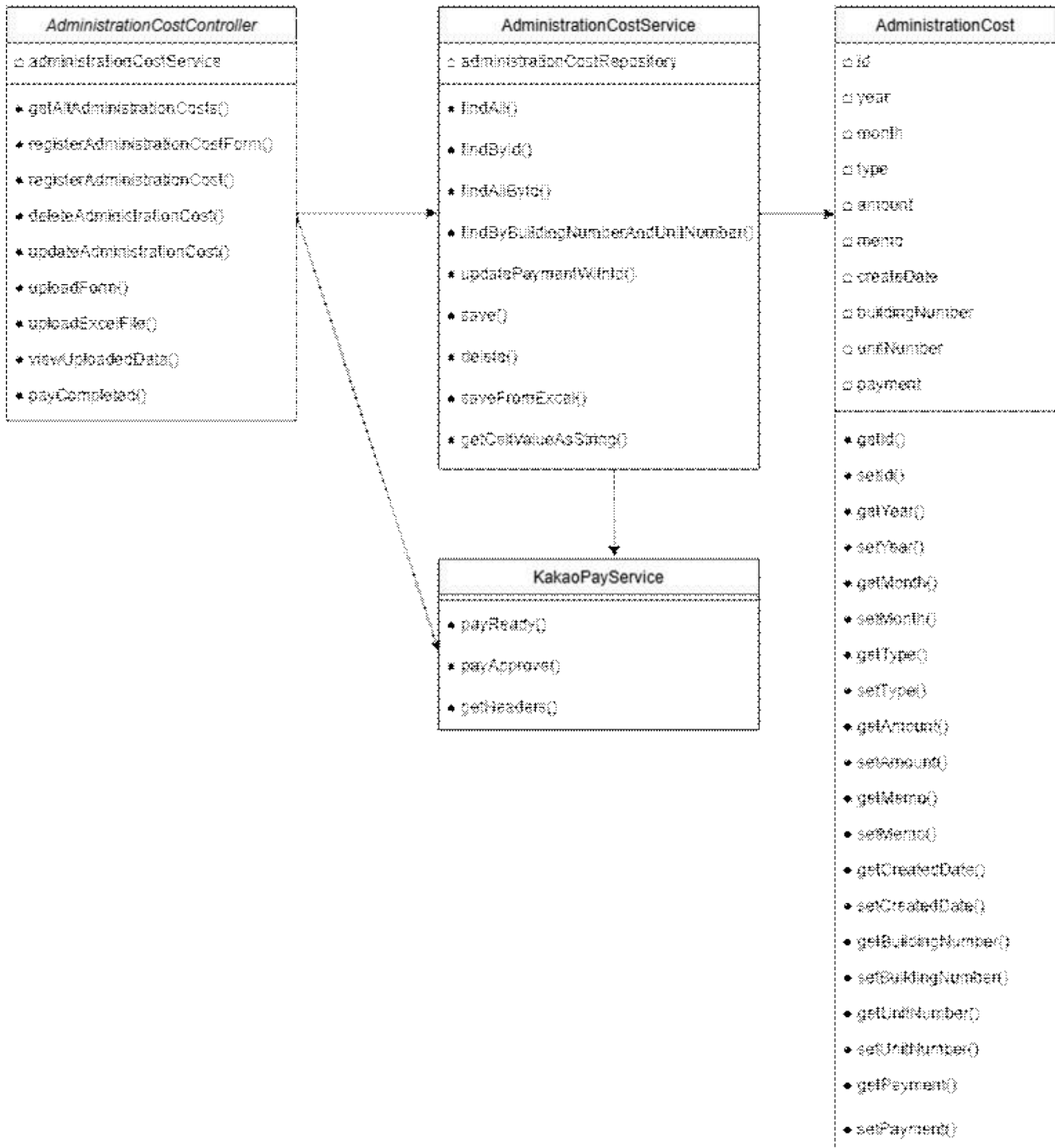
3. 분석

3.1 Class Diagram - Web 페이지

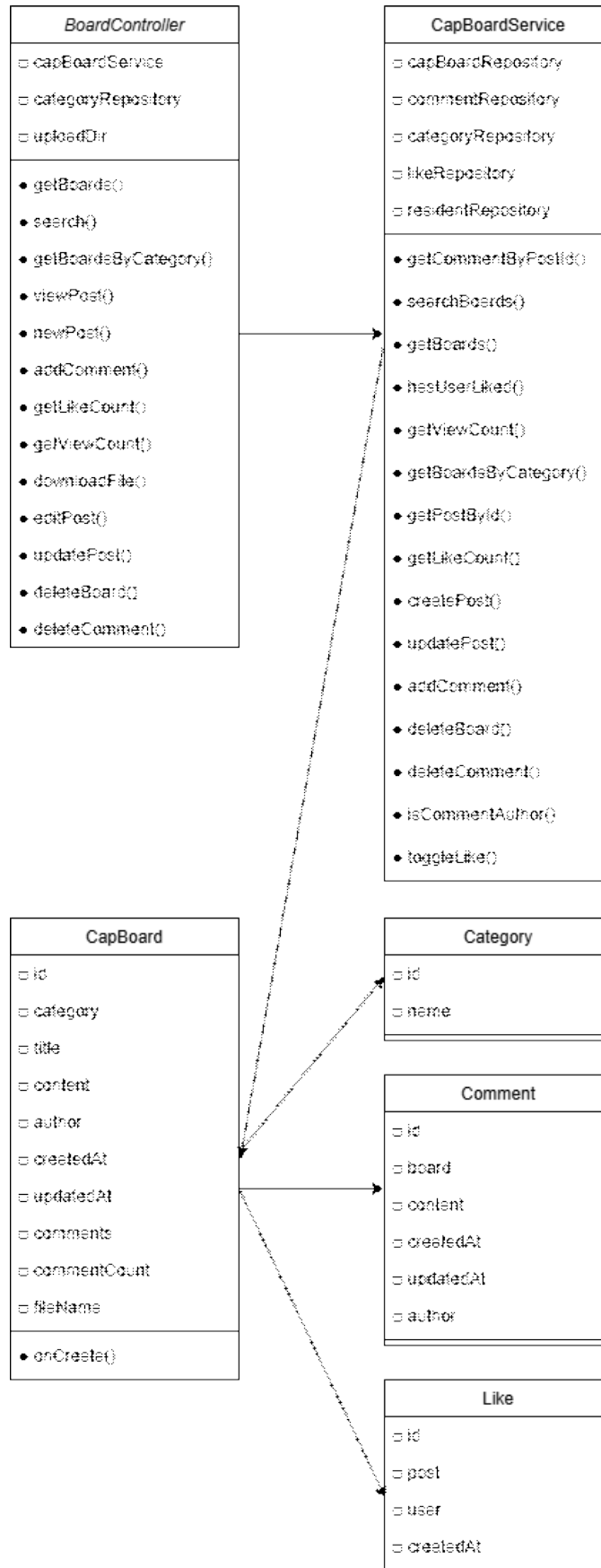
□ '사용자 로그인' 관련 클래스



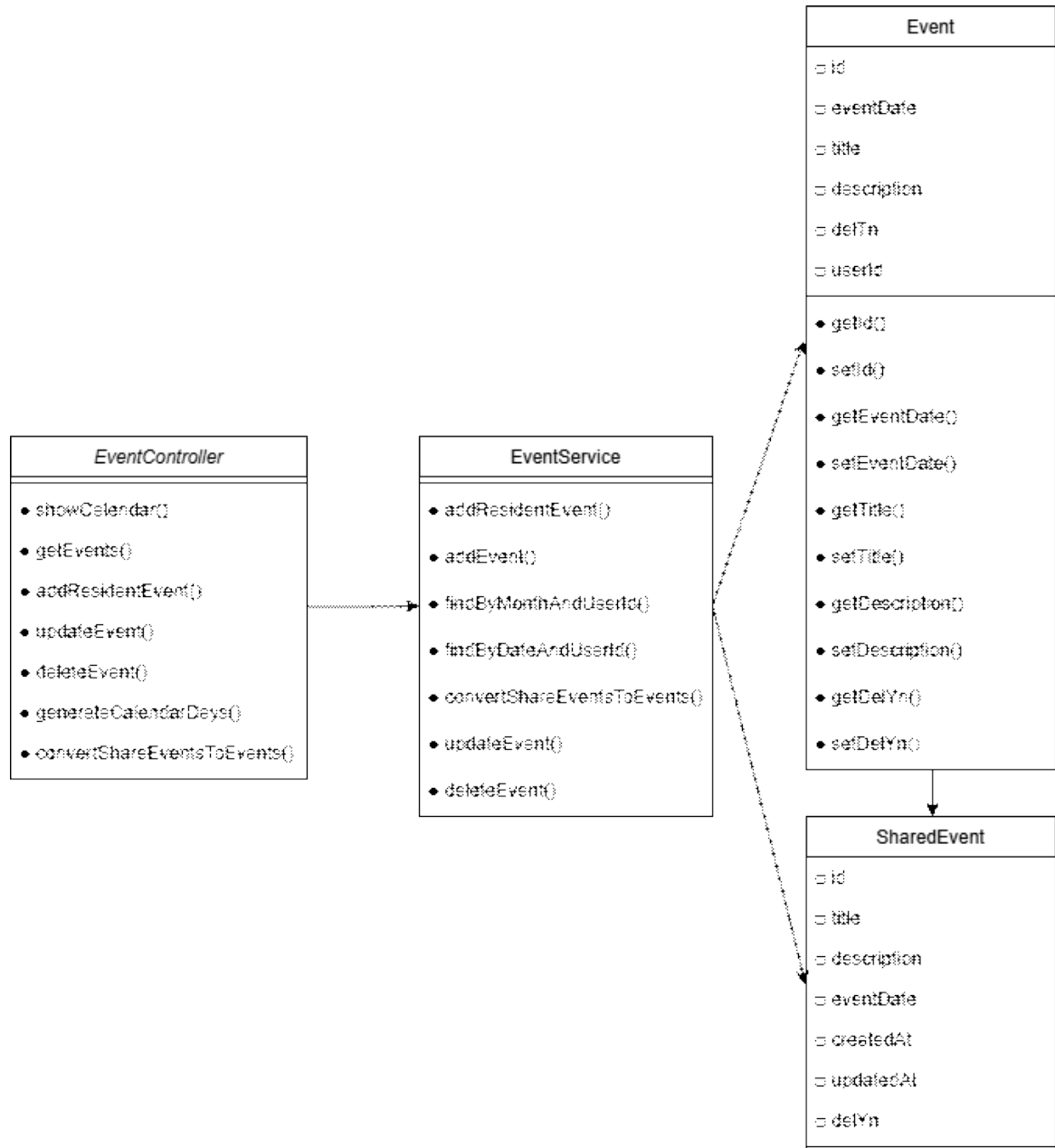
□ '관리비 조회 및 납부' 관련 클래스



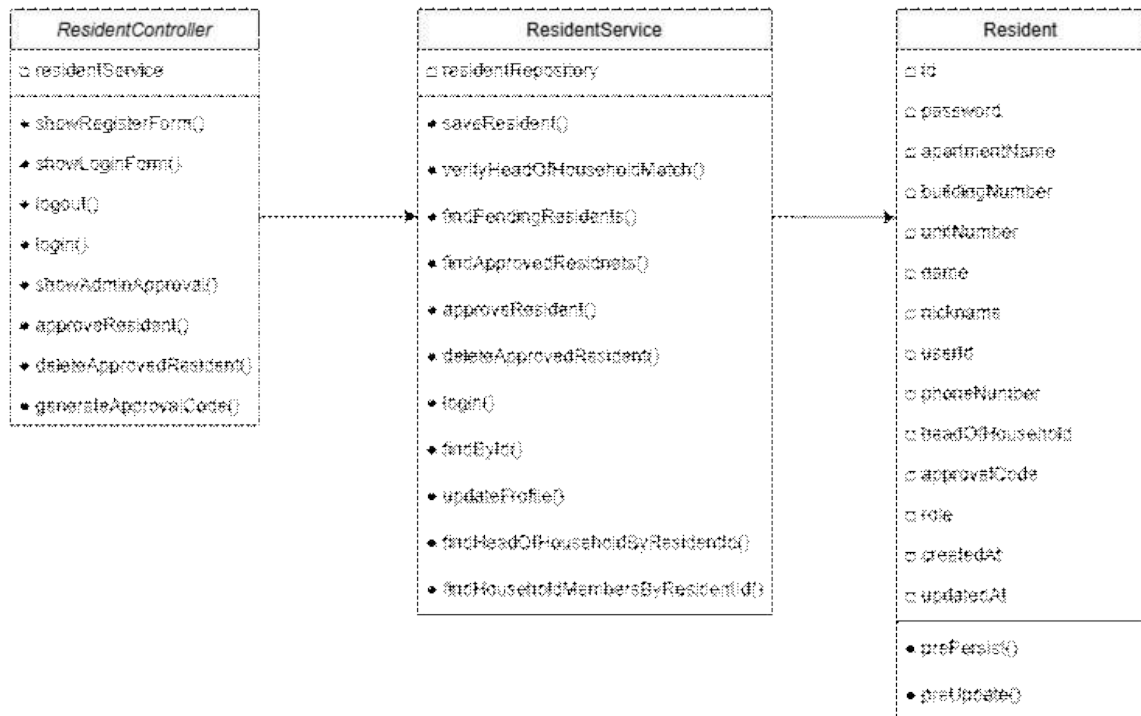
□ '게시판' 관련 클래스



□ '캘린더' 관련 클래스

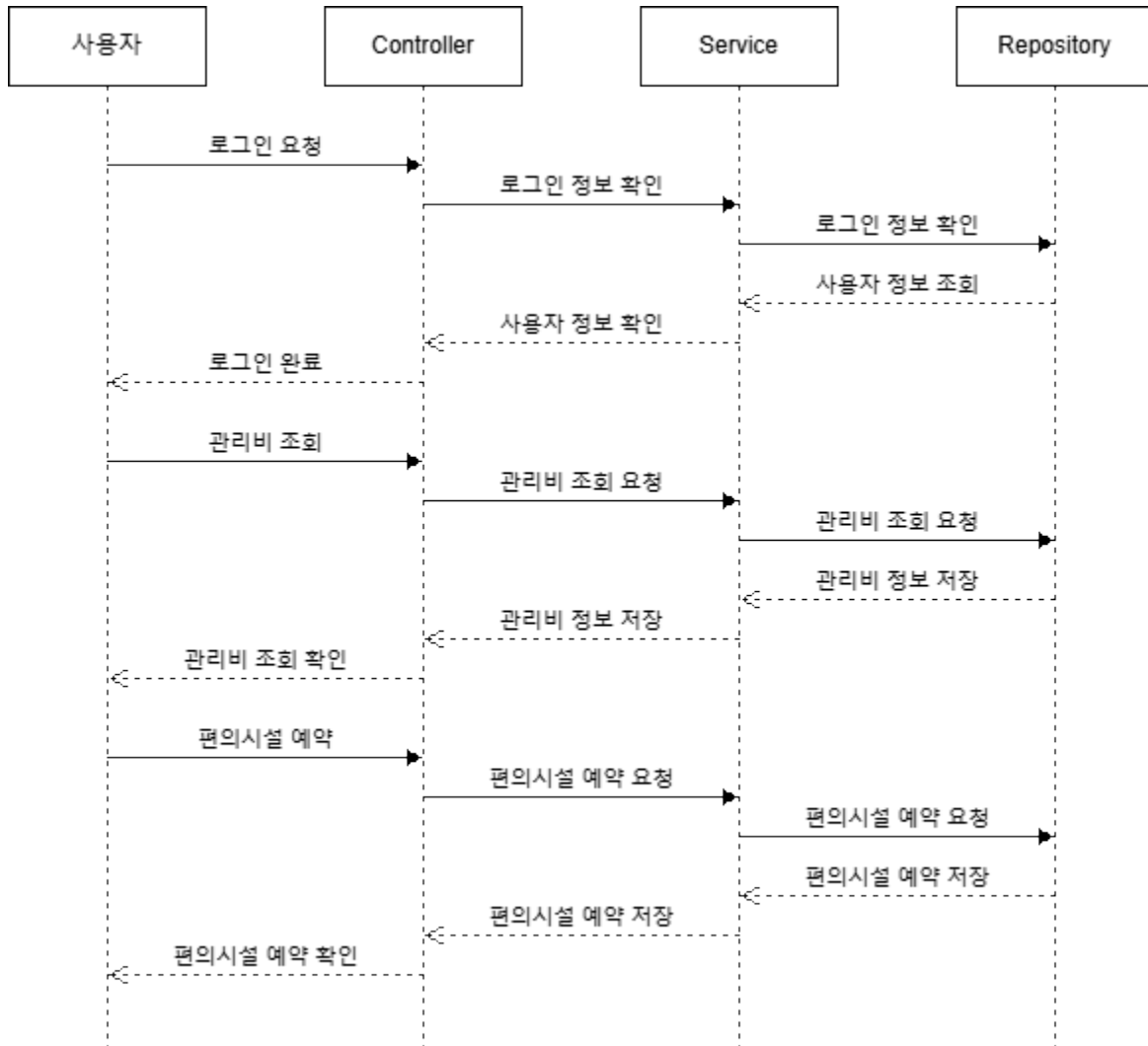


□ '입주민 회원가입' 관련 클래스

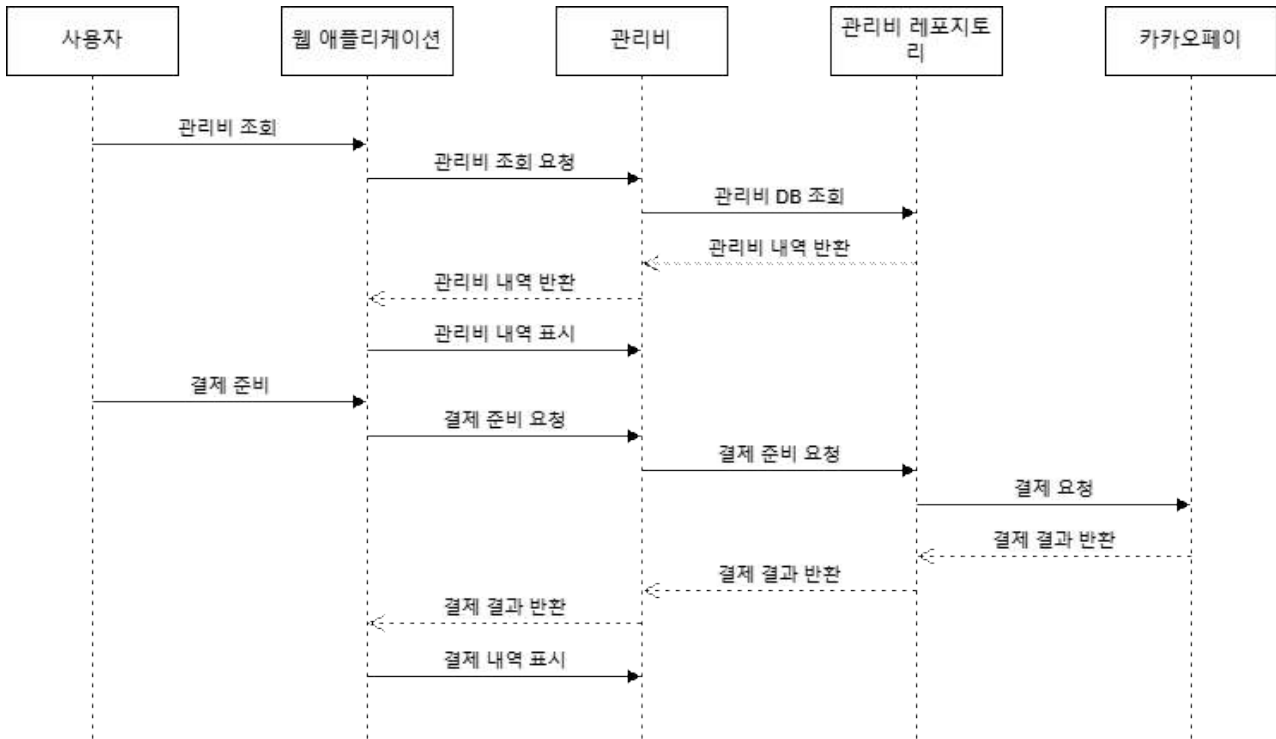


3.2 Sequence Diagram (Web 페이지)

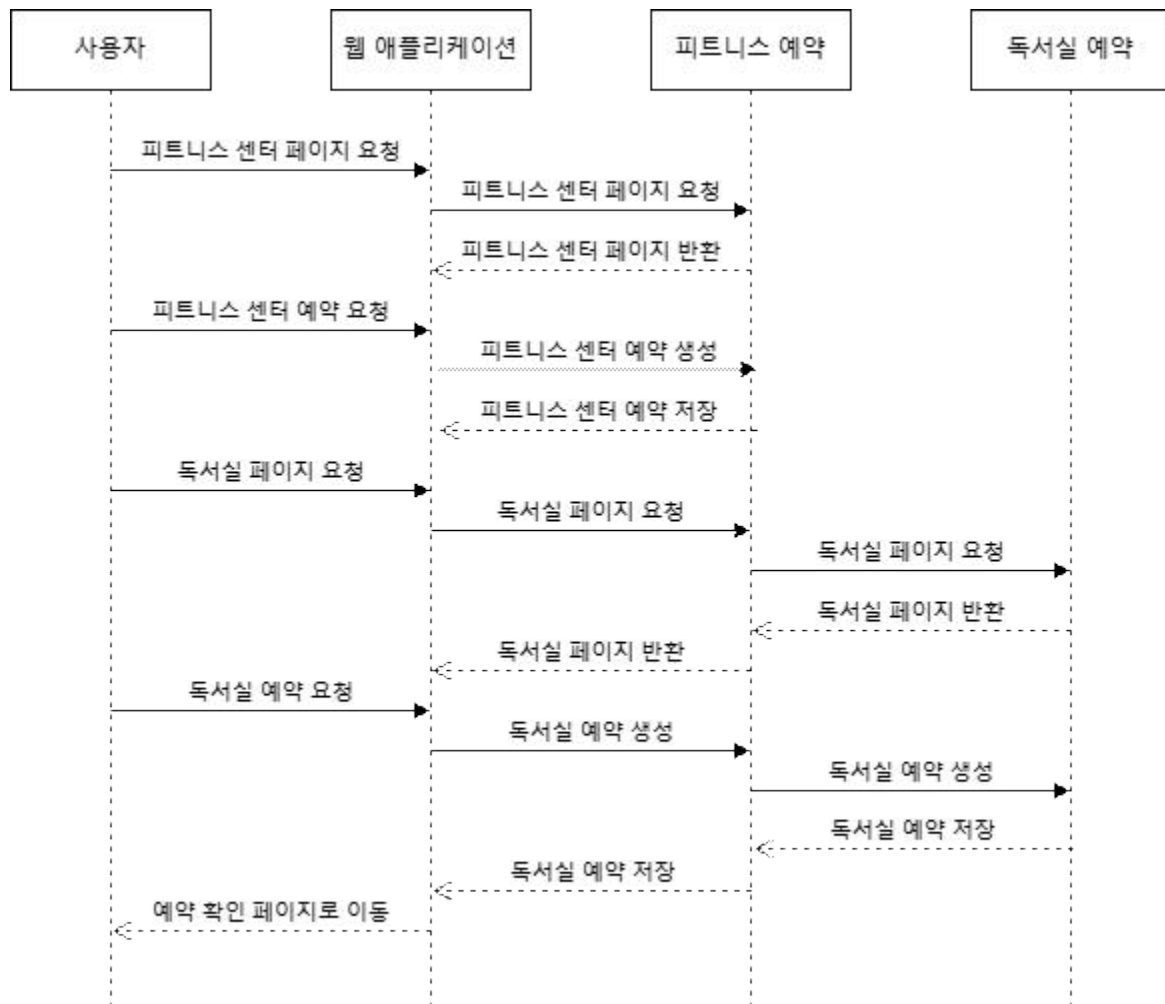
□ '전체적인 로직'



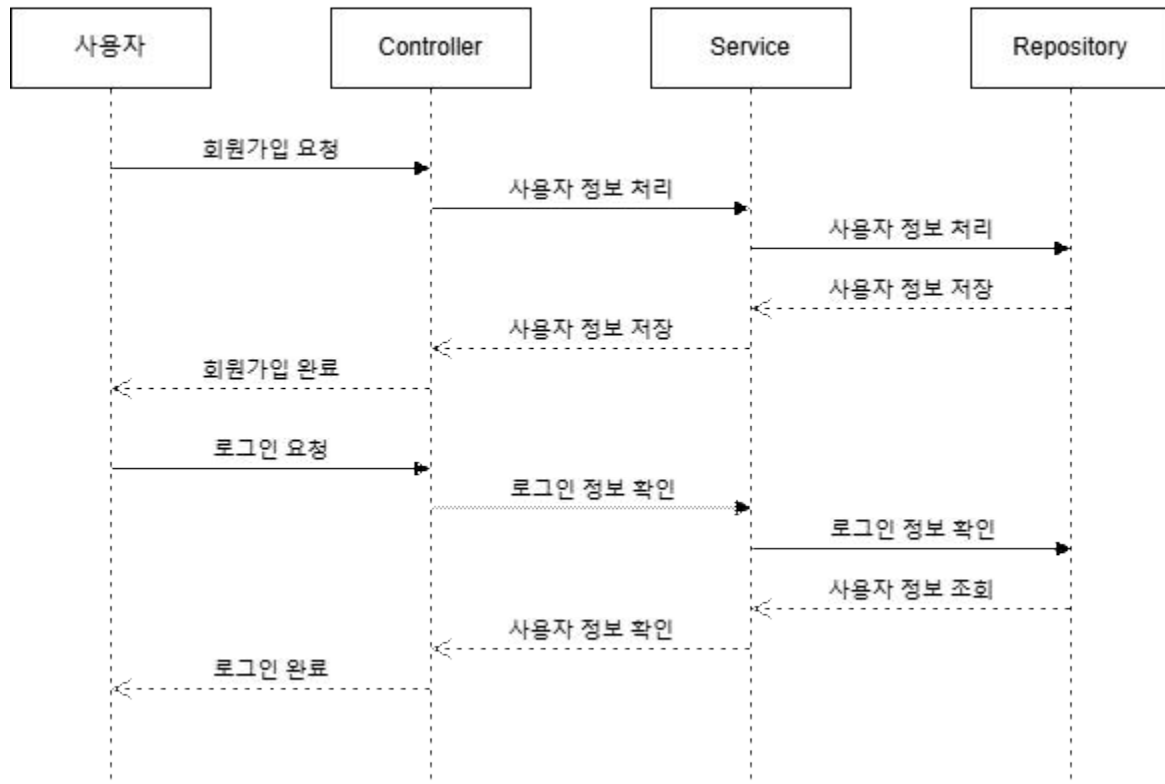
□ '관리비 조회 및 납부'



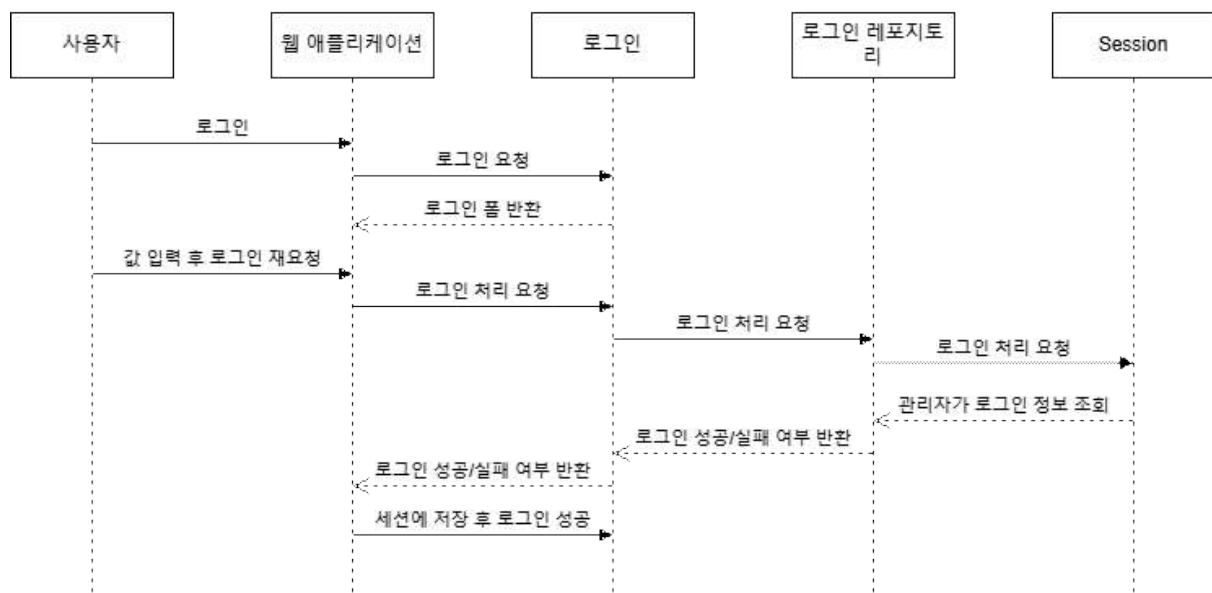
□ '편의시설 예약'



□ '회원가입'

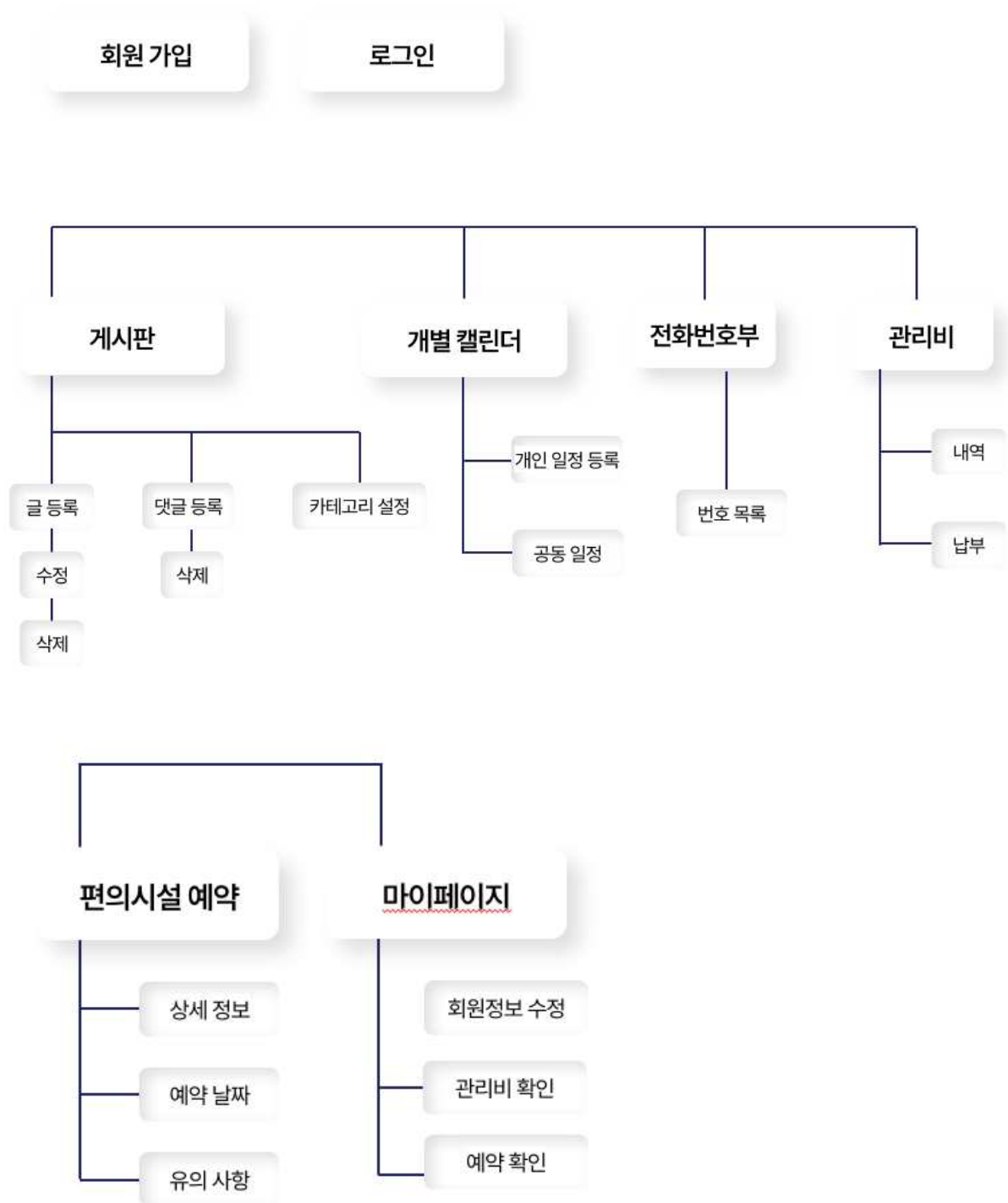


□ '로그인'

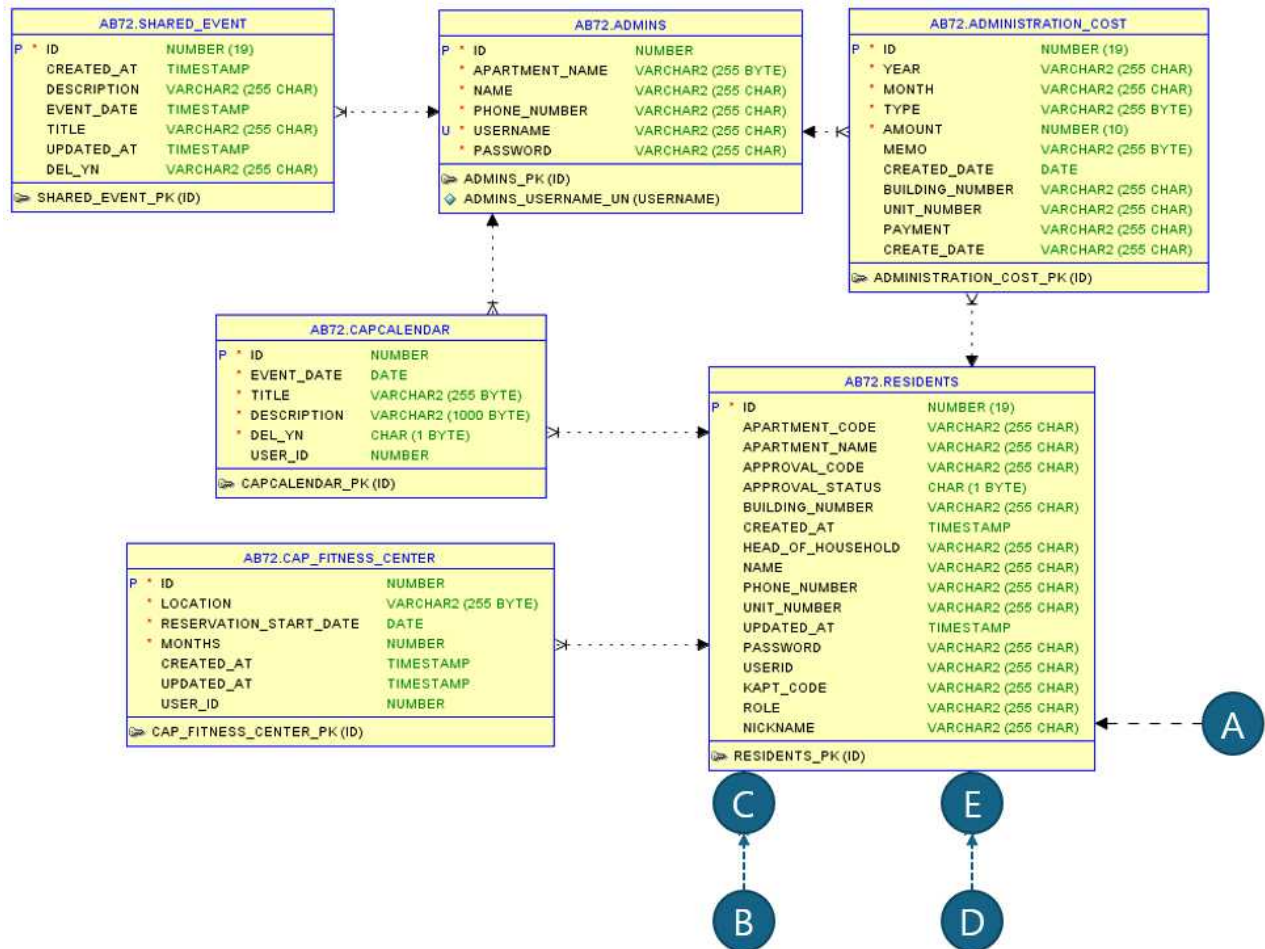


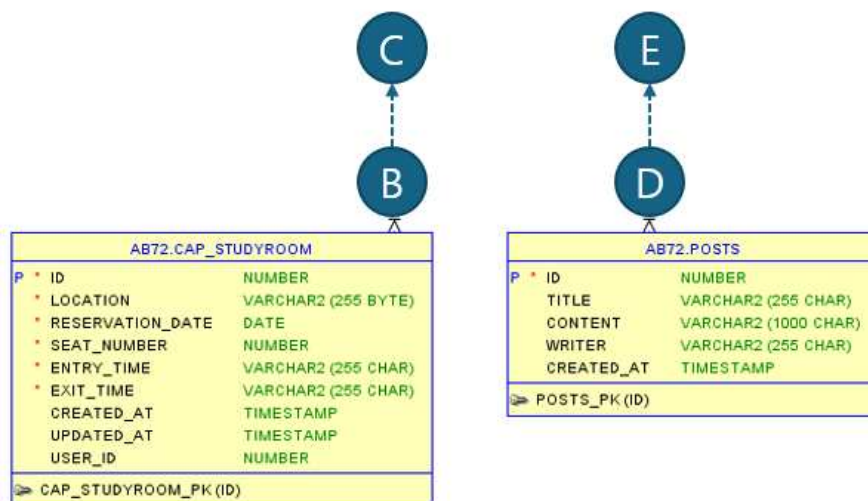
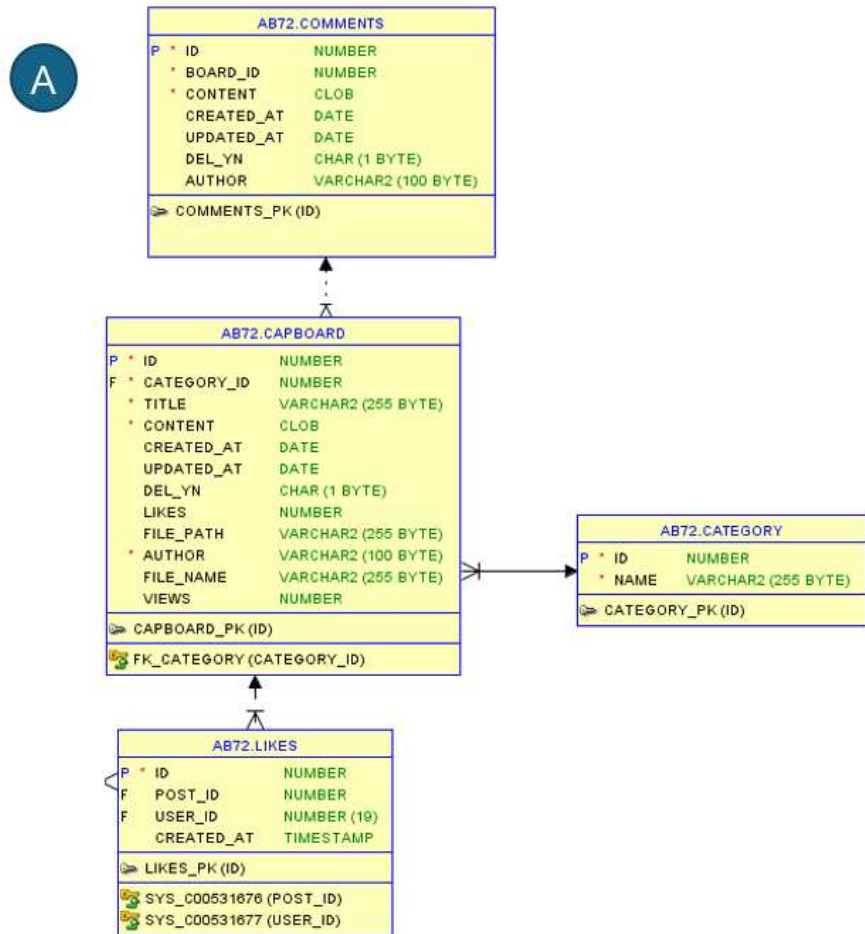
4. 설계

4.1 메뉴구조도



4.3 데이터베이스 설계





5. 중심클래스 구조 및 소스 분석

5.1 관리비 (Web)

□ 클래스 명 : AdministrationCostController

□ 클래스 개요 :

- 관리비 관리와 관련된 요청을 처리하는 컨트롤러 클래스
- 입주민은 본인의 관리비 내역을 조회하고, 납부 상태를 확인하거나 납부 진행
- 관리자는 관리비 데이터를 등록, 수정, 삭제하며, 필요시 영수증을 생성
- KakaoPay API와 연동하여 관리비 납부 기능을 제공하며, 관리비 관련 데이터를 Excel 파일로 업로드하여 관리

□ 클래스 구조도

```
AdministrationCostController
├─ @GetMapping
│   └─ 모든 관리비 내역 조회
├─ @GetMapping("/register")
│   └─ 관리비 등록 폼 표시
├─ @PostMapping("/register")
│   └─ 관리비 등록
├─ @GetMapping("/view/{buildingNumber}/{unitNumber}")
│   └─ 로그인한 사용자 동/호수 기준 관리비 조회
├─ @PostMapping("/delete/{id}")
│   └─ 관리비 데이터 삭제
├─ @PostMapping("/update/{id}")
│   └─ 관리비 데이터 수정
├─ @GetMapping("/upload")
│   └─ Excel 업로드 폼 표시
├─ @PostMapping("/upload")
│   └─ Excel 파일로 관리비 데이터 업로드
├─ @GetMapping("/receipt/{id}")
│   └─ 관리비 영수증 PDF 생성
├─ @PostMapping("/order/pay/ready")
│   └─ KakaoPay 결제 준비
├─ @GetMapping("/order/pay/completed")
│   └─ KakaoPay 결제 완료 처리
```

□ 소스 분석

-관리비 조회 및 등록

```

    @GetMapping("/view/{buildingNumber}/{unitNumber}")
    public String findByBuildingNumberAndUnitNumber(
        @PathVariable String buildingNumber,
        @PathVariable String unitNumber,
        HttpSession session,
        Model model) {
        String loggedBuildingNumber = (String) session.getAttribute("buildingNumber");
        String loggedUnitNumber = (String) session.getAttribute("unitNumber");

        if (loggedBuildingNumber == null || loggedUnitNumber == null) {
            return "redirect:/residents/login";
        }

        List<AdministrationCost> filteredCostList =
            administrationCostService.findByBuildingNumberAndUnitNumber(loggedBuildingNumber, loggedUnitNumber);

        if (filteredCostList.isEmpty()) {
            return "redirect:/administration-cost";
        }

        model.addAttribute("administrationCosts", filteredCostList);
        return "administrationcost/view";
    }

```

- kakaoPay 결제

```

    @PostMapping("/order/pay/ready")
    public @ResponseBody ReadyResponse payReady(@RequestBody OrderCreateForm orderCreateForm) {
        String name = orderCreateForm.getName();
        int totalPrice = orderCreateForm.getTotalPrice();
        int payId = orderCreateForm.getPayId();

        ReadyResponse readyResponse = kakaoPayService.payReady(name, totalPrice);
        SessionUtils.addAttribute("tid", readyResponse.getTid());
        SessionUtils.addAttribute("payId", payId);

        return readyResponse;
    }

    @GetMapping("/order/pay/completed")
    public String payCompleted(@RequestParam("pg_token") String pgToken) {
        String tid = SessionUtils.getStringAttributeValue("tid");
        int payId = SessionUtils.getIntAttributeValue("payId");

        ApproveResponse approveResponse = kakaoPayService.payApprove(tid, pgToken);
        administrationCostService.updatePaymentWithId(Long.valueOf(payId));

        return "redirect:/administration-cost/view/buildingNumber/unitNumber";
    }

```

5.2 캘린더 (Web)

□ 클래스 명 : EventController

□ 클래스 개요 :

- 캘린더 이벤트 관리와 관리와 관련된 요청을 처리하는 컨트롤러 클래스
- 사용자는 특정 날짜에 이벤트를 추가, 수정, 삭제할 수 있으며 월 단위로 일정을 확인 가능
- 이벤트 데이터를 사용자 별로 관리하며 관리자와 사용자의 권한에 따라 접근 가능

□ 클래스 구조도

```
CalendarController
├─ @GetMapping("/")
│   └─ 현재 월의 캘린더 표시
├─ @GetMapping("/month/{year}/{month}")
│   └─ 선택한 연월의 캘린더 표시
├─ @PostMapping("/addEvent")
│   └─ 새로운 일정 추가
├─ @PostMapping("/editEvent/{id}")
│   └─ 일정 수정
├─ @PostMapping("/deleteEvent/{id}")
│   └─ 일정 삭제
├─ @GetMapping("/download")
│   └─ 이벤트 데이터를 Excel 파일로 다운로드
├─ @PostMapping("/upload")
│   └─ Excel 파일을 업로드하여 일정 데이터 관리
```

□ 소스 분석

@Controller

```
public class EventController {
    private final EventService eventService;

    public EventController(EventService eventService) {
        this.eventService = eventService;
    }

    @GetMapping("/calendar")
    public String showCalendar(@RequestParam(value = "year", required = false) Integer year,
                               @RequestParam(value = "month", required = false) Integer month,
                               @RequestParam(value = "ajax", required = false) Boolean ajax,
                               Model model) {

        LocalDate today = LocalDate.now();
        if (year == null) year = today.getYear();
        if (month == null) month = today.getMonthValue();
        if (month < 1) {
            year -= 1;
            month = 12;
        } else if (month > 12) {
            year += 1;
            month = 1;
        }

        YearMonth yearMonth = YearMonth.of(year, month);
        LocalDate firstDayOfMonth = yearMonth.atDay(1);
        LocalDate lastDayOfMonth = yearMonth.atEndOfMonth();
        List<Event> events = eventService.findByMonth(year, month);

        Map<LocalDate, List<String>> eventsByDate = events.stream()
            .collect(Collectors.groupingBy(Event::getEventDate,
                Collectors.mapping(Event::getTitle, Collectors.toList())));

        List<List<LocalDate>> calendarDays = generateCalendarDays(firstDayOfMonth, lastDayOfMonth);
        model.addAttribute("calendarDays", calendarDays);
        model.addAttribute("eventsByDate", eventsByDate);
        model.addAttribute("year", year);
        model.addAttribute("month", month);
        model.addAttribute("years", IntStream.rangeClosed(today.getYear() - 5, today.getYear() +
            5).boxed().collect(Collectors.toList()));
        model.addAttribute("months", IntStream.rangeClosed(1, 12).boxed().collect(Collectors.toList()));
    }
}
```

```

// ajax 요청인 경우 calendarTable 프래그먼트만 반환

return "calendar/calendar";
}
// 이벤트 목록을 카드 오버레이로 가져옴
@GetMapping("/event/getEvents")
public String getEvents(@RequestParam("date") String date, Model model) {
    LocalDate selectedDate = LocalDate.parse(date);
    List<Event> events = eventService.findByDate(selectedDate);
    model.addAttribute("events", events);
    return "event/eventList"; // eventList.html에서 카드 오버레이에 표시
}
// 이벤트 추가
@PostMapping("/event/addEvent")
@ResponseBody
public String addEvent(@RequestParam("eventDate") LocalDate eventDate,
                       @RequestParam("title") String title,
                       @RequestParam("description") String description) {
    Event event = eventService.addEvent(eventDate, title, description);
    return String.valueOf(event.getId()); // Long 값을 String으로 변환하여 반환
}
// 이벤트 수정
@PostMapping("/event/updateEvent")
@ResponseBody
public String updateEvent(@RequestParam("id") Long id,
                          @RequestParam("title") String title,
                          @RequestParam("description") String description) {
    eventService.updateEvent(id, title, description);
    return "수정 완료";
}
// 이벤트 삭제
@PostMapping("/event/deleteEvent")
@ResponseBody
public String deleteEvent(@RequestParam("id") Long id) {
    eventService.deleteEvent(id);
    return "삭제 완료";
}
private List<List<LocalDate>> generateCalendarDays(LocalDate firstDayOfMonth, LocalDate lastDayOfMonth) {

```

```

    List<List<LocalDate>> calendarDays = new ArrayList<>();

    LocalDate firstDayOfWeek =
firstDayOfMonth.withDayOfMonth(1).minusDays(firstDayOfMonth.getDayOfWeek().getValue() % 7);
    LocalDate lastDayOfWeek = lastDayOfMonth.plusDays(6 - lastDayOfMonth.getDayOfWeek().getValue() %
7);

    for (LocalDate date = firstDayOfWeek; !date.isAfter(lastDayOfWeek); date = date.plusDays(7)) {
        List<LocalDate> week = IntStream.range(0, 7)
            .mapToObj(date::plusDays)
            .collect(Collectors.toList());
        calendarDays.add(week);
    }
    return calendarDays;
}
}

```

5.3 편의시설 예약 (Web)

□ 클래스 명 : ReservationController

□ 클래스 개요 :

- 아파트 내 편의시설 예약 관리를 처리하는 컨트롤러 클래스
- 사용자는 헬스장, 독서실 등 특정 시설을 선택하여 날짜와 시간 예약을 할 수 있음
- 예약 내역 조회, 수정, 삭제를 지원하며, 예약 완료 후 상세 정보 확인 가능

□ 클래스 구조도

```
ReservationController
├─ @GetMapping("/")
│   └─ 예약 가능한 모든 시설 목록 조회
├─ @GetMapping("/reservation/{facilityType}")
│   └─ 선택된 시설의 예약 페이지로 이동
├─ @PostMapping("/reservation/{facilityType}/add")
│   └─ 새로운 예약 추가
├─ @PostMapping("/reservation/{facilityType}/edit/{id}")
│   └─ 예약 수정
├─ @PostMapping("/reservation/{facilityType}/delete/{id}")
│   └─ 예약 삭제
├─ @GetMapping("/confirmation/{id}")
│   └─ 예약 완료 상세 정보 표시
├─ @GetMapping("/download")
│   └─ 예약 데이터를 Excel 파일로 다운로드
├─ @PostMapping("/upload")
│   └─ Excel 파일을 업로드하여 예약 데이터 관리
```


□ 소스 분석

@Controller

```
public class ReservationController {
    private final ReservationService_Fitness fitnessService;
    private final ReservationService_Study studyService;

    public ReservationController(ReservationService_Fitness fitnessService, ReservationService_Study studyService) {
        this.fitnessService = fitnessService;
        this.studyService = studyService;
    }

    // 커뮤니티 센터 페이지
    @GetMapping(value = "/resident/community")
    public String communityCenter() {
        return "Community/community";
    }

    // 피트니스 센터 페이지 (달력 포함)
    @GetMapping(value = "/resident/communityCenter/fitnessCenter")
    public String fitnessCenter(@RequestParam(value = "year", required = false) Integer year,
                                @RequestParam(value = "month", required = false) Integer month, Model
                                model) {
        YearMonth yearMonth = getCurrentOrSpecifiedMonth(year, month);
        model.addAttribute("yearMonth", yearMonth);
        model.addAttribute("prevMonth", yearMonth.minusMonths(1));
        model.addAttribute("nextMonth", yearMonth.plusMonths(1));
        LocalDate firstDayOfMonth = yearMonth.atDay(1);
        LocalDate lastDayOfMonth = yearMonth.atEndOfMonth();
        int firstDayOfWeek = firstDayOfMonth.getDayOfWeek().getValue() % 7;
        firstDayOfWeek = firstDayOfWeek == 0 ? 7 : firstDayOfWeek;
        model.addAttribute("daysInMonth", firstDayOfMonth.datesUntil(lastDayOfMonth.plusDays(1)).toList());
        model.addAttribute("firstDayOfWeek", firstDayOfWeek);
        return "Community/FitnessCenter";
    }

    // 피트니스 센터 예약 처리 및 확인 페이지로 이동
    @PostMapping("/resident/communityCenter/fitnessCenter/reserve")
    public String reserveFitnessCenter(@RequestParam("location") String location,
                                        @RequestParam("startDate") String startDateStr,
                                        @RequestParam("months") int months, Model model) {
        LocalDate startDate = LocalDate.parse(startDateStr);
        fitnessService.createReservation(location, startDate, months);
    }
}
```

```

        model.addAttribute("location", location);
        model.addAttribute("startDate", startDate);
        model.addAttribute("months", months);
        return "Community/FitnessCenterConfirmation";
    }

    // 독서실 페이지 (달력 포함)
    @GetMapping(value = "/resident/communityCenter/studyRoom")
    public String studyRoom(@RequestParam(value = "year", required = false) Integer year,
                           @RequestParam(value = "month", required = false) Integer month, Model model) {
        YearMonth yearMonth = getCurrentOrSpecifiedMonth(year, month);
        model.addAttribute("yearMonth", yearMonth);
        model.addAttribute("prevMonth", yearMonth.minusMonths(1));
        model.addAttribute("nextMonth", yearMonth.plusMonths(1));
        LocalDate firstDayOfMonth = yearMonth.atDay(1);
        LocalDate lastDayOfMonth = yearMonth.atEndOfMonth();
        int firstDayOfWeek = firstDayOfMonth.getDayOfWeek().getValue() % 7;
        firstDayOfWeek = firstDayOfWeek == 0 ? 7 : firstDayOfWeek;
        model.addAttribute("daysInMonth", firstDayOfMonth.datesUntil(lastDayOfMonth.plusDays(1)).toList());
        model.addAttribute("firstDayOfWeek", firstDayOfWeek);
        return "Community/StudyRoom";
    }

    // 독서실 예약 처리 및 확인 페이지로 이동
    @PostMapping("/resident/communityCenter/studyRoom/reserve")
    public String reserveStudyRoom(@RequestParam("location") String location,
                                   @RequestParam("date") String dateStr,
                                   @RequestParam("seatNumber") int seatNumber,
                                   @RequestParam("entryTime") String entryTime,
                                   @RequestParam("exitTime") String exitTime, Model model) {
        LocalDate reservationDate = LocalDate.parse(dateStr);
        Reservation_Study reservation = new Reservation_Study(location, reservationDate, seatNumber, entryTime,
                                                                exitTime);
        studyService.createReservation(reservation);
        // 예약 확인 정보를 모델에 추가
        model.addAttribute("location", location);
        model.addAttribute("reservationDate", reservationDate);
        model.addAttribute("seatNumber", seatNumber);
        model.addAttribute("entryTime", entryTime);
        model.addAttribute("exitTime", exitTime);
        return "Community/StudyRoomConfirmation";
    }


```

```
}  
  
private YearMonth getCurrentOrSpecifiedMonth(Integer year, Integer month) {  
    LocalDate today = LocalDate.now();  
    if (year == null) year = today.getYear();  
    if (month == null) month = today.getMonthValue();  
    return YearMonth.of(year, month);  
}  
}
```

6. 구현 화면

6-1 Web 구현화면

□ 회원가입(세대주)



회원가입

회원가입

회원 유형

세대주

아파트명

장안대

동

291

호수

204

이름

양하늘

닉네임

YHN

아이디

gksmf20211304


비밀번호

전화번호

010-2024-1118

회원가입

□ 회원가입(세대원)


회원가입

회원 유형	세대원
아파트명	<input type="text"/>
동	<input type="text"/>
호수	<input type="text"/>
이름	<input type="text"/>
닉네임	<input type="text"/>
아이디	<input type="text"/>
비밀번호	<input type="password"/>
전화번호	<input type="text"/>
세대주 이름	<input type="text"/>
승인 코드	<input type="text"/>
<input type="button" value="회원가입"/>	

□ 관리자 로그인 승인 화면


계시판 편의시설 예약 관리비▼ 캘린더 주변 정보 전화번호부 어서오세요, 관리자님! 로그아웃

승인 완료


승인코드	회원유형	닉네임	아이디	이름	전화번호	아파트명	동	호수	승인
202410216786	head	dudals	123	김영민	134654646	장안대	101	101	승인


승인 완료 사용자 목록

승인코드	회원유형	닉네임	아이디	이름	전화번호	아파트명	동	호수	상태	삭제
202410241602	head	test39393	222	양하늘	111	장안대	202	202	승인 완료	삭제
202410383175	head	jioh	333	김지오	46546546465	장안대	203	203	승인 완료	삭제
202410383175	household	JIOH	444	김지오2	010-1111-2222	장안대	203	203	승인 완료	삭제
202410241602	household	ji	888	지	454646	장안대	202	202	승인 완료	삭제
202410384524	head	YHN	gksmf20211304	양하늘	010-2024-1118	장안대	201	204	승인 완료	삭제
202410241602	household	영	1212	김	54656	장안대	202	202	승인 완료	삭제

승인 버튼을 클릭하여 사용자를 승인할 수 있습니다.

□ 로그인 후 메인화면


계시판 편의시설 예약 관리비▼ 캘린더 주변 정보 전화번호부 어서오세요, 김지오님! 마이페이지 로그아웃



□ 게시판

[게시판](#)
[편의시설 예약](#)
[관리비](#)
[캘린더](#)
[주변 정보](#)
[전화번호부](#)
환영합니다, 양하늘님!
[마이페이지](#)
[로그아웃](#)

전체 카테고리

[공지 사항](#)
[자유 게시판](#)
[민원 게시판](#)
[정보 공유](#)
[나눔 / 장터](#)

글 등록

작성자, 제목으로 검색

검색

카테고리	제목	작성자	작성일	조회	좋아요	댓글
민원 게시판	밤에는 조금만 조용히 해주세요~	YHN (201)	2024.11.17	20	2	2
나눔 / 장터	에어팟 2세대 나눔할게요	YHN (201)	2024.11.17	1	0	0
자유 게시판	헬로키티가 좋아요~	YHN (201)	2024.11.17	0	0	0
자유 게시판	헬로키티 좋아하시나요?	영 (202)	2024.11.17	2	1	0
나눔 / 장터	글 나눔해요~	영 (202)	2024.11.17	12	1	2
민원 게시판	어제(11/16) 202동 5층분들	영 (202)	2024.11.17	6	0	0
자유 게시판	예쁜 꽃 보고가세요	영 (202)	2024.11.17	10	0	3
정보 공유	아이들 영양제 추천	jioh (203)	2024.11.17	22	1	3
정보 공유	OO마트 세일한대요	jioh (203)	2024.11.17	5	1	4
자유 게시판	아이들 계단 이용	jioh (203)	2024.11.17	11	0	0
자유 게시판	갑자기 왜이렇게 추워!	jioh (203)	2024.11.17	3	0	2

□ 게시판 글 등록

[게시판](#)
[편의시설 예약](#)
[관리비](#)
[캘린더](#)
[주변 정보](#)
[전화번호부](#)
환영합니다, 양하늘님!
[마이페이지](#)
[로그아웃](#)

글 등록

카테고리: 자유 게시판

제목: 내일 OO마트 세일 공구하실분?

내용:

저희는 많이 못 먹어서요
내일 OO마트 세일하면 조금 많이 살 것 같은데
저희처럼 많이 못 먹는 분

파일 첨부:

파일 선택

선택된 파일 없음

등록

□ 게시판 글 상세


[게시판](#)
[편의시설 예약](#)
[관리비▼](#)
[캘린더](#)
[주변 정보](#)
[전화번호부](#)
[환영합니다, 양하늘님!](#)
[마이페이지](#)
[로그아웃](#)

[돌아가기 >](#)

밤에는 조금만 조용히 해주세요~

작성자: YHN (201) | 작성일: 2024-11-17 | 조회수: 21

새벽 3시에 고래고래 소리를 지르시더라고요ㅠㅠ
어떤 분인지는 못 봤지만

함께 사는 곳에서 예의를 조금만 지켜주세요!

❤️ 좋아요 3

댓글

jioh (203)

맞아요 저도 너무 시끄러워서 잠을 설치어요

2024-11-17 10:14


영 (202)

다음부터는 신상 확인해서 주의를 줘야 한다고 생각합니다

2024-11-17 10:15

댓글을 입력하세요

□ 편의시설 예약 첫 화면


[게시판](#)
[편의시설 예약](#)
[관리비▼](#)
[캘린더](#)
[주변 정보](#)
[전화번호부](#)
[환영합니다, 양하늘님!](#)
[마이페이지](#)
[로그아웃](#)

커뮤니티 센터 예약

헬스장

위치: 지하 1층 (105동 부근)

운영 시간: 06:00 - 23:00

예약 하기

독서실

위치: 107동 옆

운영 시간: 06:00 - 익일 05:00

예약 하기


그룹 스터디 룸

위치: 독서실 맞은편

운영 시간: 07:00 - 익일 05:00

예약 하기

□ 편의시설 예약(헬스장)


[게시판](#)
[편의시설 예약](#)
[관리비▼](#)
[캘린더](#)
[주변 정보](#)
[전화번호부](#)
[환영합니다, 양하늘님!](#)
[마이페이지](#)
[로그아웃](#)

피트니스 센터 예약

<
2024-11
>

일	월	화	수	목	금	토
					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

개월 수 선택

개월 수: 3개월 ▼

예약 확정

□ 편의시설 예약(독서실)


[게시판](#)
[편의시설 예약](#)
[관리비▼](#)
[캘린더](#)
[주변 정보](#)
[전화번호부](#)
[환영합니다, 양하늘님!](#)
[마이페이지](#)
[로그아웃](#)

독서실 예약

<
2024-11
>

일	월	화	수	목	금	토
					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

좌석 선택


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

시간 선택

입실 시간: 8:00 ▼ 퇴실 시간: 17:00 ▼

□ 입주민 개별 캘린더

<div>  게시판 편의시설 예약 관리비▼ 캘린더 주변 정보 전화번호부 환영합니다, 김지오님! 마이페이지 로그아웃 </div>						
<div> < 2024년 12월 > </div>						
일	월	화	수	목	금	토
1	2	3	4	5	6	7
8	9 * 헬스장 이용 *	10	11	12 엘리베이터 점검	13	14
15	16	17	18 건강검진	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4

□ 관리비 목록(관리자)

<div>  게시판 편의시설 예약 관리비▼ 캘린더 주변 정보 전화번호부 어서오세요, 관리자님! 로그아웃 </div>						
관리비 업로드						<div> 역설등록 개별등록 </div>
번호	부과연월	작성자	메모	등록일	수정	삭제
53	2025년 1월	관리자	테스트2	2024-11-12	수정	삭제
54	2025년 2월	관리자	테스트3	2024-11-12	수정	삭제
55	2025년 4월	관리자	테스트4	2024-11-12	수정	삭제
52	2025년 1월	관리자	테스트1	2024-11-12	수정	삭제
61	2024년 12월	관리자	헬스장 이용비 (2025-02까지)	2024-11-18	수정	삭제
60	2024년 12월	관리자	일반관리비	2024-11-18	수정	삭제
65	2024년 12월	관리자	헬스장 이용비 (2025-02까지)	2024-11-18	수정	삭제
66	2024년 12월	관리자	일반 관리비	2024-11-18	수정	삭제

□ 관리비 등록(관리자)


게시판 편의시설 예약 관리비▼ 캘린더 주변 정보 전화번호부 로그인▼ 회원가입

관리비 등록

부과연도

2024

부과월

12

관리비 형태

전기요금

금액

11

메모

동

101

호

101

□ 관리비 목록(입주민)


게시판 편의시설 예약 관리비▼ 캘린더 주변 정보 전화번호부 환영합니다, 김지오님! 마이페이지 로그아웃

관리비 내역

번호	부과연도	부과월	관리비 형태	금액	등록일	동	호수	납부여부
54	2025	2	일반	546456	2024-11-12	203	203	완납
52	2025	1	일반	1000000	2024-11-12	203	203	완납
71	2025	1	전기요금	23450	2024-11-18	203	203	미납

□ 아파트 내 주요 전화번호 목록



[게시판](#)

[편의시설 예약](#)

[관리비▼](#)

[캘린더](#)

[주변 정보](#)

[전화번호부](#)

[환영합니다, 김지오님!](#)


[마이페이지](#)

[로그아웃](#)

전화번호 목록

시설명	번호	이용가능시간
관리사무소	031-589-6656	08:00 - 22:00
헬스장	031-789-6656	06:00 - 23:00
독서실	031-689-6656	06:00 - 23:00

□ 마이페이지(입주민)

	게시판	편의시설 예약	관리비	캘린더	주변 정보	전화번호부	환영합니다, 김지오님!	마이페이지	로그아웃
마이페이지									
회원 정보 이름: 김지오 아이디: 333 동: 203 호수: 203 닉네임: jioh 전화번호: 46546546465 개인정보 수정					세대주 정보 이름: 김지오 승인 코드: 202410383175 세대원 목록 이름: 김지오2				
작성한 게시글									
OO마트 세일한대요 정보 공유 작성자: jioh (203) L O R 자세히 보기					아이들 영양제 추천 정보 공유 작성자: jioh (203) 아이들 영양제 추천해요 OO키즈, OO키커 아이들이 맛있게 잘 먹어요 ^^ 자세히 보기				

7. 프로젝트 후기

1. 프로젝트 목표 및 성과

이번 프로젝트는 아파트 입주민과 관리자에게 필요한 종합적인 웹 서비스를 구현하는 것을 목표로 했다.

주요 성과:

기능적인 완성도: 관리비 조회 및 납부, 편의시설 예약, 게시판 등 핵심 기능이 정상적으로 작동.

사용자 친화적 설계: 입주민과 관리자의 요구를 반영한 UI/UX 디자인.

확장 가능성 확보: Oracle Database, Spring Boot, Thymeleaf를 기반으로 확장 가능한 아키텍처 설계.

2. 팀원 별 후기

김영민: 초기 설계 단계에서는 각 기능의 요구사항을 정의하고, 필요한 기술 스택을 선택했으며, 일정 관리와 진행 상황 점검을 주도했습니다. 팀장으로서 팀원들이 어려움을 겪을 때는 적극적으로 도와 문제를 해결했으며, 팀 분위기를 긍정적으로 유지해 프로젝트를 성공적으로 마무리할 수 있도록 노력했습니다. 이 과정을 통해 리더십과 문제 해결 능력을 키웠고, 팀워크의 중요성을 깊이 느꼈습니다.

사용자 경험을 중시한 UI/UX 디자인과 프론트엔드 구현을 담당했습니다. Thymeleaf를 활용해 반응형 웹 페이지를 제작했습니다. 팀원들과 협업하며 사용자 편의성을 극대화하는 데 집중했고, 이를 통해 프론트엔드 개발 역량을 한 단계 성장시킬 수 있었습니다. 특히, 사용자 친화적인 UI 설계가 얼마나 중요한지 깨닫게 된 값진 경험이었습니다.

또, 회원가입과 인증 시스템 개발을 전담하며 입주민 승인 절차와 세대주 및 세대원 간의 관계를 고려한 데이터 모델링을 설계했습니다. Oracle Database와 Spring Boot를 활용해 복잡한 승인 로직을 구현하며, 데이터 무결성과 보안을 유지하는 데 중점을 두었습니다. 이를 통해 데이터베이스 설계와 백엔드 로직 구현에 대한 실력을 크게 향상시킬 수 있었습니다.

양하늘: 주로 캘린더 기능과 편의시설 예약 로직 구현을 담당했습니다. 사용자별 개인화된 캘린더와 관리자 캘린더 기능을 분리하여, 각 사용자의 예약 정보가 정확히 반영되도록 데이터 흐름을 설계했습니다. Spring Boot와 Oracle Database를 활용해 복잡한 예약 비즈니스 로직을 구현하며, 성능 최적화를 위해 다양한 방법을 고민했습니다. 이를 통해 대규모 데이터를 처리하는 로직 설계와 성능 최적화의 중요성을 깊이 이해하게 되었고, 문제 해결 능력과 협업의 가치를 크게 배울 수 있었습니다.

캘린더 화면의 사용자 경험을 향상시키기 위한 프론트엔드 개발도 맡아 JavaScript와 Thymeleaf를 사용해 월간/일간 캘린더 UI를 설계하고, 날짜 선택 및 이벤트 표시 기능을 구현했습니다. 특히, 사용자와 관리자의 캘린더가 직관적으로 동기화되는 인터랙션 설계에 중점을 두었으며, 지속적인 피드백을 통해 디자인을 개선하는 과정을 경험했습니다. 반응형 웹 디자인을 적용하여 다양한 디바이스에서도 일관된 사용자 경험을 제공하는 데 성공했습니다.

이 프로젝트를 통해 동적으로 데이터를 처리하고 상호작용하는 UI 설계 기술을 익혔으며, 사용자 피드백을 반영하며 점진적으로 개선해 나가는 방법의 중요성을 배웠습니다.

김지오: 백엔드 개발과 데이터베이스 설계를 맡아 Spring Boot와 Oracle DB를 기반으로 다양한 기능을 구현했습니다. 특히, 사용자 관리 시스템의 API 설계와 데이터베이스 연동 작업에 주력했으며, 그중 관리비 청구 및 납부 관리 기능 개발에 깊이 관여했습니다. 관리비 청구 데이터를 효율적으로 처리할 수 있는 API를 설계하고, 사용자별 맞춤형 관리비 정보를 제공하는 시스템을 구현했습니다. 이 과정에서 데이터를 정확하게 반영하기 위해 납부 상태와 연체 여부를 실시간으로 처리하는 로직을 구축하며, 데이터의 일관성과 신뢰성을 보장했습니다.

Oracle SQL을 최적화하기 위해 효율적인 쿼리 구조를 설계하며 대규모 데이터 처리의 성능 문제를 해결하는 데 집중했습니다. 특히, 복잡한 데이터 간의 관계를 고려한 설계와 쿼리 최적화를 통해 성능 향상을 이루어냈습니다. 또한, 관리비 청구 주기와 연체 처리 로직을 자동화함으로써 시스템의 효율성을 높였습니다.

프로젝트를 통해 대규모 데이터 구조와 백엔드 로직 설계의 중요성을 체감했으며, 실시간 데이터 처리와 시스템 안정성을 확보하는 데 필요한 기술 역량을 키울 수 있었습니다. 이번 경험은 백엔드 개발자로서 한층 더 성장할 수 있는 계기가 되었으며, 데이터 설계와 최적화, 그리고 협업의 중요성을 깊이 깨닫는 기회가 되었습니다.

8. 참고 자료 및 소스

- 1) <https://www.aptner.com/>
- 2) <https://aptner.oopy.io/e1a0a543-5a85-4c1f-83fb-b6e1895ff36e>
- 3) <https://www.apti.co.kr/apti/>
- 4) <https://aptree.co.kr/home/>