

4차 2기 Project



3세대 스마트 화분

2018년 06월 04일

지능형 IoT 전문가 육성과정

3조

김신재

이해주

전승용

주상훈

황원준

목 차

1. 프로젝트 개요	4
1.1 프로젝트 기획 배경 및 목표	4
1.2 구성원 및 역할	6
1.3 프로젝트 추진 일정	7
2. 프로젝트 현황	9
2.1 시장 분석	9
2.2 경쟁 제품 장단점 분석	11
2.2.1 Parrot Pot	11
2.2.2 RoPot	11
2.2.3 Planty	12
2.3 차별화 핵심 전략	13
3. 프로젝트 개발 결과	14
3.1 시스템 전체 구성도	14
3.2 시스템 배치 구성도	15
3.2.1 챗팻 단말	15
3.2.2 서버	15
3.2.3 채팅엔진	15
3.2.4 안드로이드	16
3.3 시스템 프로세스/스레드 구성도	18
3.3.1 챗팻 단말	18
3.3.2 서버	18
3.3.3 안드로이드	19

3.4 네트워크 중심의 시스템 구성도	20
3.4.1 일반 데이터 및 채팅 데이터 흐름도	20
3.4.2 프로토콜 설계	20
3.5 전체 & 플랫폼별 기능 목록 및 동작 시나리오	22
3.5.1 플랫폼별 기능 목록 및 시스템 동작 시나리오	22
3.5.2 주요 기능의 통신 및 시스템 흐름도	24
3.5.2.1 주요 기능 목록	24
3.5.2.2 흐름과 순서도에 대한 설명	24
3.5.2.3 주요 기능의 흐름도	25
3.6 클래스 설명 및 기능 정의	32
3.6.1 챗봇 단말	32
3.6.2 서버	36
3.6.3 채팅엔진	37
3.6.4 안드로이드	38
3.7 데이터베이스 관계도 / 테이블 설계	40
3.7.1 서버 데이터베이스 설계도	40
3.7.2 플랫폼별 데이터베이스 사용 다이어그램	40
3.8 화면 설계	41
3.8.1 화면 전체 구성도	41
3.8.2 상세 화면 구성	42
4. 기대 효과	46
4.1 향후 개선 사항	46
4.2 기대 효과	47
5. 개발 후기	50
6. 강사 의견	54
※ 별첨 1. 업무 일지	별첨 1_3 조_업무일지.pdf

1. 프로젝트 개요

1.1 프로젝트 기획 배경 및 목표

(1) 홈가드닝의 수요와 성장

실내에서 식물을 기르는 것은 인류의 오랜 전통이자 훌륭한 인테리어의 수단이었다. 예전부터 인간과 소통하며 교감하는 반려동물에 더불어 최근에는 반려식물이라는 단어가 등장하며 그 관심도가 높아지고 있는 추세다. 2015 년 한국정원산업의 현황과 전망 심포지엄에서 현재 정원 관련 산업의 규모는 약 3300 억 원으로 10 년 뒤에 1 조 원까지 증가할 것이라고 전망하였다.

정원을 통한 식물과 조경에 대한 관심 뿐만 아니라 실내에서 기를 수 있는 홈가드닝 제품의 수요 또한 증가하고 있다. 2017 년 롯데마트에서는 화분 매출이 전년동기대비 128.1%로 증가하였고, 화초는 52%, 원예재료 매출도 33.3% 증가하였다. 또한, 신세계몰에서는 홈 가드닝의 대표 제품이라고 할 수 있는 스투키 매출이 전년대비 591%나 늘었으며 이어 금전수(270%), 뱅갈고무나무(130%) 등 높은 신장률을 보였다.

실제 홈 가드닝이 인기를 끌자 식물의 종류나 기르는 법 등을 묻는 온라인 상담건수도 부쩍 늘었다. 경기도 농업기술원에서 운영하는 사이버 식물병원에 따르면 지난해 404 건이었던 식물 관련 온라인 상담건수가 2017 년 5 월에만 1100 여 건으로 급증했다. 사기업에서 밝힌 통계 수치를 읽으면 사람들의 식물에 대한 전체적인 관심과 수요가 늘어나는 것을 통해 홈가드닝의 시장 규모 또한 꾸준히 상승할 것으로 보인다.

(2) 스마트 화분 시장의 현재

실내 화분 산업에 뛰어드는 국내외 IoT 기업들이 있다. 대표적인 국내기업으로는 엔씨의 플랜티가 글로벌 시장에 진출했으며, 해외 기업으로는 샤오미의 로팟과 패롯의 패롯팟 등이 있다.

스마트 화분 시장 대표주자들 각각의 시장 진입 시점을 살펴보면 엔씨가 2013 년도에 개발을 시작하여 스타트업으로 최우수상 수상 후 개발을 시작하였다. 패롯팟은 2015 년도 CES 에 출품하였고, 샤오미 로팟은 1 년 동안의 개발 후 2016 년 출시로 가장 늦게 발매되었다. 앞서 언급한 회사들의 제품들은 대부분 동일한 기능을 수행하는데, 1) 화분 내 토양의 수분과 영양상태 측정 2) 어플을 통해 해당 정보를 전달 3) 재배하는 식물의 종을 입력할 경우 권장되는 재배 방법 추천 등의 기능을 제공한다. 또한 고가형 모델의 경우 자동 급수 시스템을 갖춘 경우도 있다.

선발주자들이 시장의 영역을 확장하고 있지만 스마트 화분 시장이 형성된 지 채 5 년이 되지 않았으며 그 기능은 화분을 조금 더 쉽게 관리해주는 것에 그치고 있다. 결정적으로 위의 스마트 화분들은 식물의 전통적인 성격인 '수동성'에는 어떠한 변화도 주지 않았다.

(3) 화분의 진화와 '햇팟'의 가능성 - 세대의 관점에서

화분으로써의 최소한의 기능을 수행하는 '1 세대 화분'에서 더 나아가, 위에서 소개한 것처럼 시중에는 이미 스마트 화분, 스마트 화분 관리기를 통해 화분에서 식물의 상태를 체크함으로써 해당

반려식물에게 도움이 될 수 있도록 상태를 보여주는 '2 세대 화분'이 있다. '챗팻'은 이러한 기능에 더하여 사용자로부터 식물의 일방적 소통이 아닌 식물과 소통을 원하는 사용자를 위한 프로젝트이다. '챗팻'은 식물에 대한 사용자의 애정과 관심을 받아 **대화**의 형태로 상호작용을 하는 '3 세대 화분'으로, 사용자와 반려식물의 교감을 증대시키는 것에 초점을 맞추었다.

3 세대 스마트 화분의 시작, 챗팻

1 세대 화분



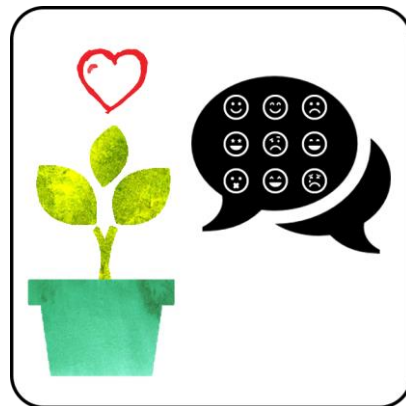
전통적인 의미의 화분. 식물이 자랄 수 있는 바탕을 제공

2 세대 화분



각종 센서들을 통해 기온, 조도, 토양 수분 함량을 포함한 식물의 상태를 볼 수 있게 만들어 줌

3 세대 화분



1 세대, 2 세대 화분의 기능을 계승하며, 화분에서 자라는 식물이 감정을 표현할 수 있게 해주는 채널(채팅)을 통해 사용자와 상호작용

(4) 3 세대 스마트 화분 '챗팻'의 핵심기능 소개

- 1) 주변의 환경과 사용자의 관심을 감지, 반려식물의 기분을 추측하여 사용자와 대화한다.
- 2) 반려식물과의 상호작용으로 쌓은 친밀도 포인트에 따라 식물과의 대화가 점차 변화
 - 반려식물 관찰 시 식물이 반응
 - 앱을 통해 반려식물의 사진을 찍을 때마다 식물이 반응하여 사용자가 식물을 관찰하고 키우는 재미를 더할 수 있는 경험 제공. (추후 업데이트 예정)
 - 찍은 사진을 통해 관리하는 식물별 재배일지 제공 (추후 업데이트 예정)
- 3) 온도, 습도, 조도 등 기본 환경 정보와 식물의 영양상태 실시간 정보 조회 및 자동 급수기능

1.2 구성원 및 역할

이름 (가나다 순)	전공	역할	구현 부분
김신재	정보통신공학	팀원	채팅 엔진 개발 식물 정보 리서치
이해주	융합전자	팀원	서버 개발
전승용	전자공학	팀원	챗봇 단말 개발
주상훈	환경과학	팀원	안드로이드 앱 개발 UI 디자인
황원준	건축학	팀장	전체 시스템 구성 및 설계, 일정 조율, 플랫폼별 기술지원, 시연용 챗봇 단말 모델 제작, 시연 동영상 편집, 결과보고서 작성, 발표자료 제작

1.3 프로젝트 추진 일정

구분	기간	활동	비고
사전 기획	5/4(금) ~ 5/10(목)	프로젝트 기획 및 팀 구성	3~7인/팀
	5/11(금) ~ 5/17(목)	팀(PM/팀원) 구성 및 프로젝트 아이디어 선정	
	5/18 (금)	팀별 착수 보고 (Kick-off)	
PJT 수행 / 완료	5/18(금) ~ 6/4(월)	프로젝트 수행	
	5/25(금)	팀별 1 차 중간 보고	
	5/30(수)	중간 점검	
	6/1(금)	팀별 2 차 중간 보고	
	6/4(월)	팀별 최종 발표 (구축 완료 보고)	최우수팀 선발

일일단위계획표

	~5/17	5/18 Day-1 D-9	5/23 Day-2 D-8	5/24 Day-3 D-7	5/25 Day-4 D-6	5/28 Day-5 D-5	5/29 Day-6 D-4	5/30 Day-7 D-3	5/31 Day-8 D-2	6/1 Day-9 D-1	6/4 Day-10 D-Day
공식 일정	사전구성	프로젝트 시작			1차 시연			중간 점검		2차 시연	프로젝트 종료
팀 목표	아이디어 선정	아이디어 구체화 및 사업성과 타당성 조사	전체 시스템 설계 조건에 따른 개발환경 구축	필요 최소 기능 구현 후 기술적 가능성 검토	피드백에 따른 기능 조정 및 강화 시작	핵심 기능 보완 및 강화	회원구성 및 사용자 시나리오에 따른 기능 확장	피드백에 따른 기능 강화 시작 및 마무리 시작	UX 강화 완료된 플랫폼부터 문서화 시작	최종 결과물에 따른 보고서 및 발표자료 제작	결과물 제출 및 발표
0. 요구분석											
1. 분석설계											
1-1. DB설계											
1-2. 프로토콜 설계											
1-3. 시연 후 보완											
2. 개발코딩											
3. 최종 점검											
업무별 계획 1) 서버											
1-1. 통신 프로세스 작성											
1-1-1. 챗봇 단말과 통신 테스트											
1-1-2. AVD와 통신 테스트											
1-1-3. 실제 데이터 송수신 테스트											
1-1-4. 프로토콜에 맞추어 전송 테스트											
1-1-5. 챗봇, AVD에게 서비스 제공											
1-2. DB 적용											
1-2-1. DB 파싱 모듈 적용											
1-2-2. DB Save/Load											
1-2-3. DB에 맞춰 통신 테스트											
1-3. 채팅엔진 탑재											
1-4. 최종 시스템 보완											
업무별 계획 2) 안드로이드 (휴대전화)											
2-1. 통신 쓰레드 작성											
2-1-1. 서버와 통신 테스트											
2-1-2. 실제 데이터 송수신 테스트											
2-1-3. 프로토콜에 맞추어 전송 테스트											
2-2. UI 프로토타입 개발											
2-2-1. 최소 필요화면 산정											
2-2-2. 핵심기능 표시를 위한 UI제작											
2-2-3. 필요화면 구체화											
2-3. DB 적용											
2-3-1. 챗봇클래스, 챗봇데이터베이스 생성											
2-3-2. 서버에서 수신한 데이터 저장											
2-3-3. 테이블데이터에서 UI 표시											
2-4. UI 최종 개발											
2-4-1. 데이터 UI 표시											
2-5. 최종 시스템 보완											
업무별 계획 3) 챗봇 단말											
3-1. 센서데이터 수집 프로세스 작성											
3-2. 통신 프로세스 작성											
3-2-1. 서버와 통신 테스트											
3-2-2. 실제 데이터 송수신 테스트											
3-2-3. 프로토콜에 맞추어 전송 테스트											
3-3. DB 적용											
3-3-1. 수집한 데이터 저장 및 송신											
3-4. 2개 이상의 기기에 대해서 실험											
3-5. 최종 시스템 보완											
업무별 계획 4) 채팅 엔진											
4-1. 리서치											
4-1-1. 오픈API 적용 가능여부 조사											
4-1-2. 문장 구조 및 문법 조사											
4-2. 엔진 핵심 스트럭처 작성											
4-3. 엔진 강화											
4-3-1. 문장에 영향을 미치는 요소 정의											
4-3-2. 문장 각 구성성분 강화											
4-4. 최종 시스템 보완											

2. 프로젝트 현황

2.1 시장분석

실내에서 기르는 식물은 실내원예, 홈 가드닝, 플랜테리어(Planterior, Plant + Interior의 합성어), 반려식물 등 다양한 단어로 이용되고 있다. 현재 어느 기관에서도 국내 화분 시장의 규모는 정확하게 제공하고 있지 않다. 한 편 이를 간접적으로 추정해볼 수 있는 정원산업의 규모가 2015년 '한국정원산업의 현황과 전망 심포지엄'에서 처음으로 발표되었으므로 이에 기반하여 시장을 분석하였다.

위 심포지엄에서 발표된 '정원산업 현황 조사와 전망에 관한 연구'에 따르면 정원 산업 전체 규모는 12,792 억원 규모로 추산되고 있다. 또한 정원 산업에 사용되는 식물소재 부분은 8,676 억원이며, 원예자재, 정원도구, 정원용품이 포함된 정원자재 부분은 747 억원으로 추산된다.

표 43. 국내 정원산업 규모(산업부문 기준)

정원 산업 부문	2014년도 (억 원)	비중 %
합계	12,792	100.0
식물소재	8,676	67.8
정원자재 ⁵⁴⁾	747	5.8
정원시설·가구	339	2.7
소비유통	1,709	13.4
교육서비스 ⁵⁵⁾	92	0.7
전시문화관광	378	3.0
설계시공	709	5.5
정원관리	142	1.1

출처) 정원산업 현황 조사와 전망. 95p. 서울시립대학교 산학협력단. 김완순.

국가기관의 통계에서는 화분 산업만의 규모를 추산할 수 없지만 일반 사용자들이 홈 가드닝 용품을 쉽게 구매가능한 대형 마트에서 원예부문 매출 증가비율을 뉴스에서 보도하고 있다. 2014년 기준 이마트에서는 원예 부문 매출이 전년 대비 65% 증가하였다. 롯데마트는 2014년 기준 홈 가드닝 상품군의 매출이 전년 대비 31.2% 증가하였으며, 2017년 기준 128.1% 증가하였다. 신세계몰의 경우 홈 가드닝 대표 제품인 스투키 매출이 전년 대비 591% 증가하였다.

국내 화분시장의 규모가 정확하게 추산된 결과는 없다. 하지만, 1) 홈 가드닝의 상위 분류군인 정원 산업 규모가 2014 년 12,792 억 원이며, 10 년 뒤에는 1 조원 규모로 증가할 것이라고 전망되는 점 2) 대형 유통 판매처에서 홈 가드닝 관련 제품 판매량이 지속적으로 늘어나고 있다는 점을 통해 홈 가드닝 시장은 꾸준한 증가 추세를 이룰 것이라고 예측된다.

2.2 경쟁 제품 장단점 분석

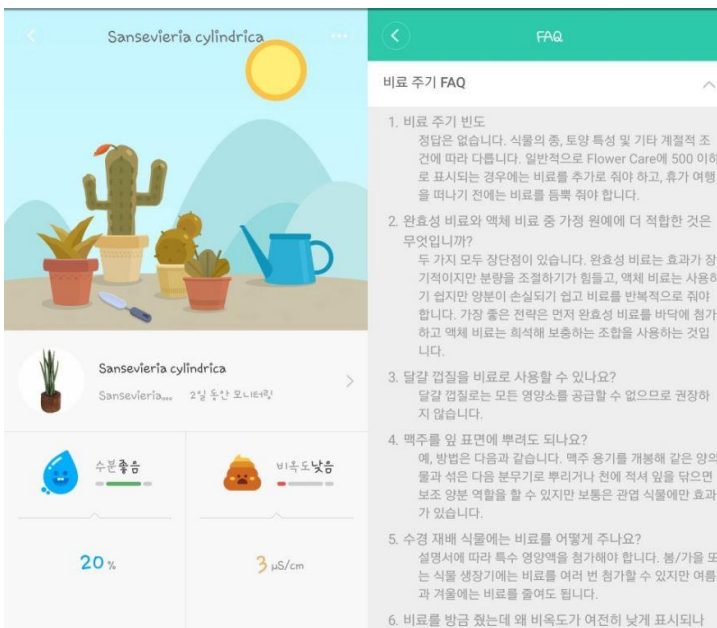
2.2.1 Parrot Pot

- 적시에 정확한 양의 물 공급
- 최대 1개월간 무인 자동 급수를 제공
- 4개의 센서(햇빛, 비료 수준, 온도 및 토양 수분) 실시간 데이터 측정
- 'FreeFlight Jumping(앱)' 제공하여 실시간으로 제어, 데이터 확인
- 전원 : 건전지



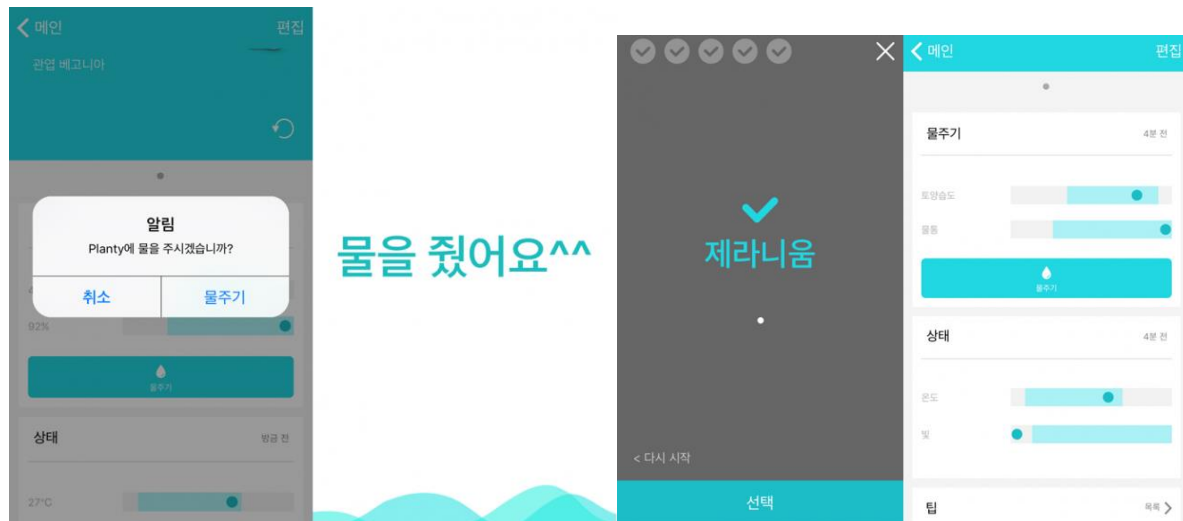
2.2.2 RoPot

- 수분 및 영양 상태 표시
- '플라워 케어' 어플리케이션 사용
- 스마트폰을 통해 실시간으로 수분 및 영양상태 확인
- 화분 전면의 LED 색을 통해서 수분, 영양 상태 확인
- 전원 : 충전방식

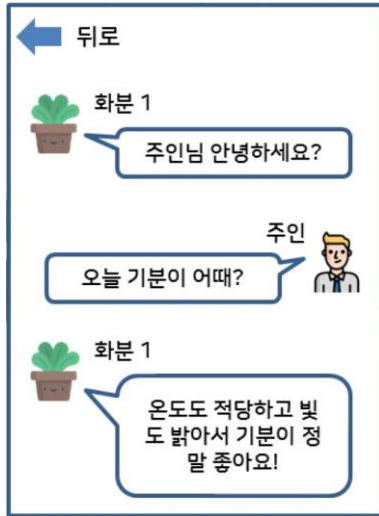


2.2.3 Planty

- 온도, 습도, 양분, 조도센서 등 사용
- 스마트폰으로 화분상태 모니터링
- 와이파이를 통해 원격으로 화분에 물 공급
- SNS를 활용한 친구 초대 그리고 공유 기능 제공
- 물이 부족하거나 온도가 맞지 않으면 사용자에게 알림 전송
- 물이 계속해서 나오거나 센서 오류가 발생하는 경우가 있음.
- 충전방식



2.3 차별화 핵심 전략



[그림 2.3.1 채팅 예시 화면]



[그림 2.3.2 찻팟 정보 보기]



[그림 2.3.3 찻팟 모아보기]

1) 채팅

기존 일방향의 수동적 반려식물 키우기에서 양방향의 능동적인 반려식물 키우기. 다양한 요인들에 의하여 변하는 반려식물의 응답은 실제 식물과 대화하고 있다는 기분을 들게 할 것이다.

이에 더하여 찻팟에게 물어볼 경우 현재 온도나 습도와 같은 실내 환경 정보 뿐만 아니라 외부 날씨 정보도 제공, 타 채팅 앱과 차별화 된 사용자 개개인에게 맞춤 정보를 이용할 수 있다.

이 모든 내용은 모두 내가 기르는 반려식물을 통해서 알게 되는 것이므로 사용자가 반려식물에게 느끼는 친밀감 또한 더욱 커질 것이다.

2) 찻팟 정보 보기

찻팟 별 상태조회 및 관리기능. 실내 기온 및 토양 수분까지 다양한 센서 데이터 값을 실시간으로 확인할 수 있으며, 사용자가 갱신하지 않아도 시간 별로 센서데이터는 서버에 저장하여 사용자의 요청에 따라 조회할 수 있다.

또한 자동 급수기능 활성화/비활성화, 이름과 품종 변경하기 기능이 있다. 이 때 등록된 품종에 따라 회사 측에서 제공하는 '올바르게 키우는 방법' 기능이 추후 업데이트 예정이다.

3) 찻팟 모아보기

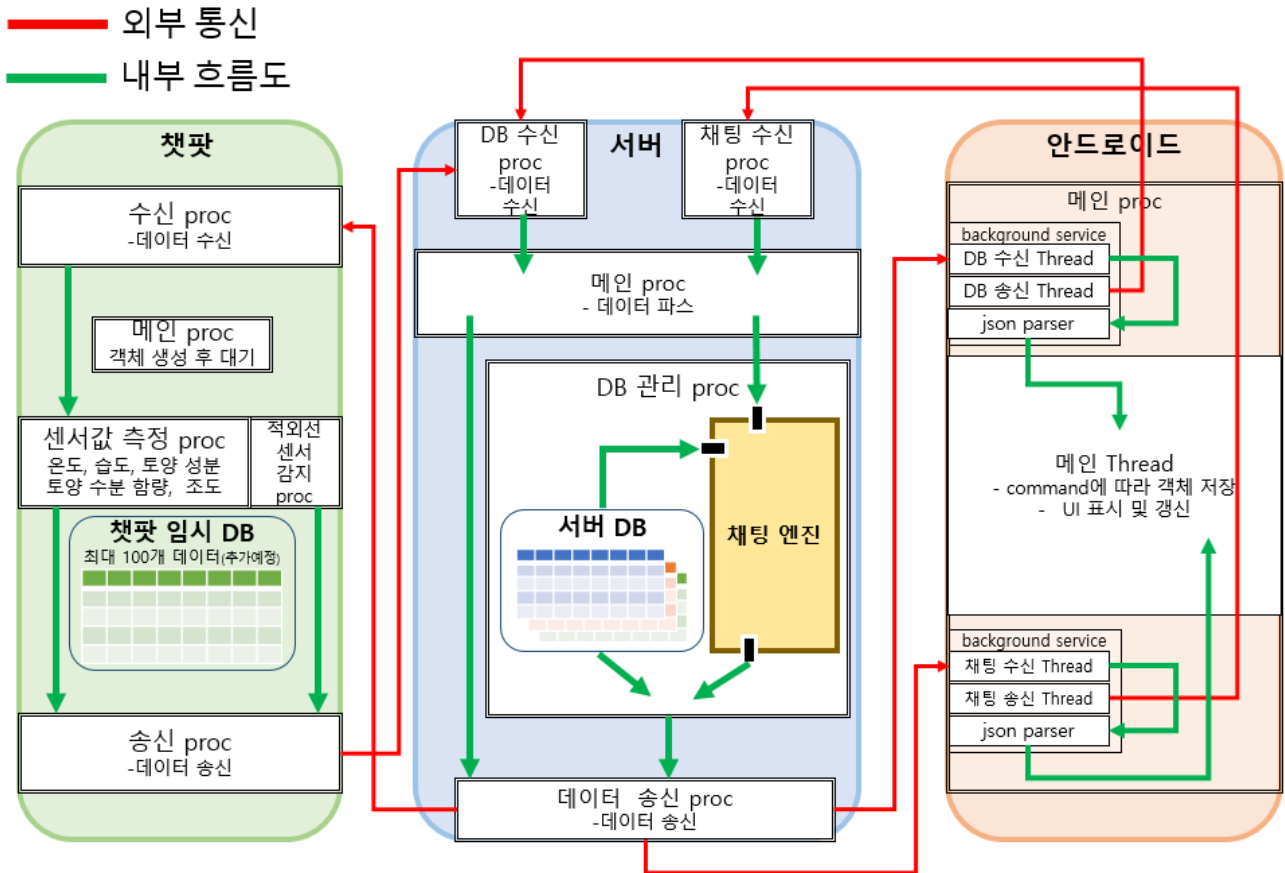
내가 등록한 찻팟을 모아서 한 눈에 볼 수 있는 모아보기 기능. 사용자는 모아보기에서 각 찻팟의 기본 정보를 볼 수 있고, 여기에서 '찻팟별 정보 관리' 혹은 '채팅'을 시작할 수 있다.

새로운 찻팟 단말을 구입한 경우 여기서 등록하여 관리할 수 있다.

* 본 화면은 예시 화면으로, 실제 완료 내용과 다를 수 있음.

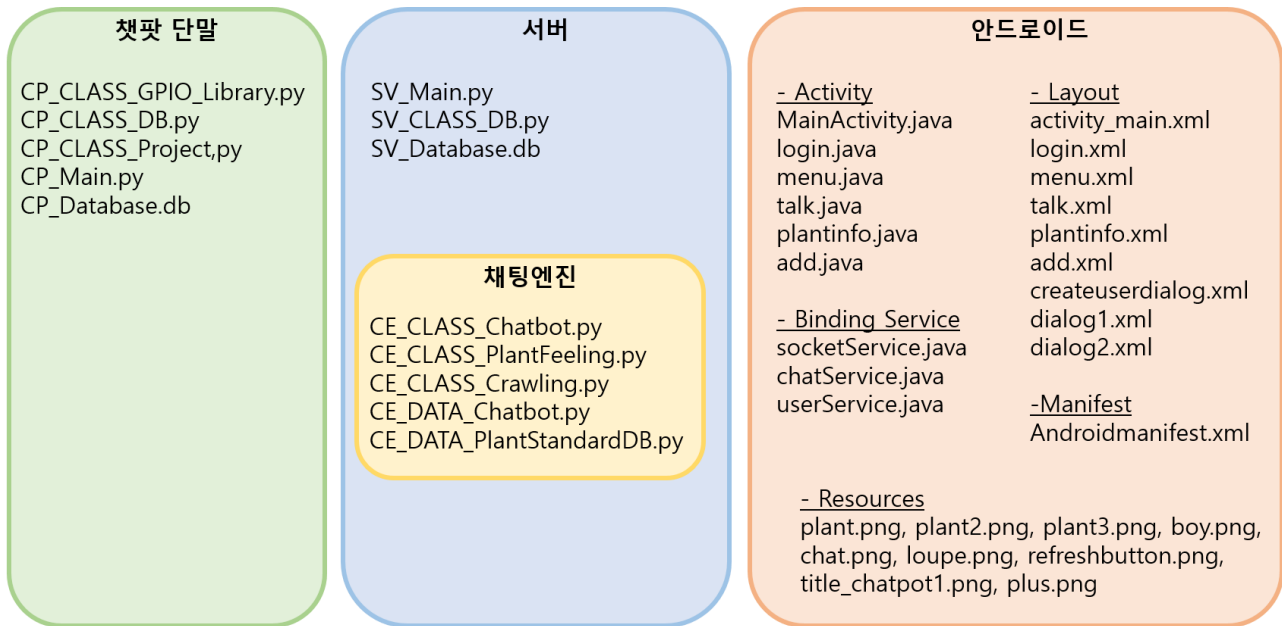
3. 프로젝트 개발 결과

3.1 시스템 전체 구성도



[그림 3-1. 시스템 구성도 - 전체]

3.2 시스템 배치 구성도



[그림 3.5.1. 시스템 구성도 - 파일 배치]

3.2.1. 챗봇 단말

- 소스 파일

CP_CLASS_GPIO_Library.py: 센서들(온도, 습도, PIR 등)의 클래스를 저장한 파일
 CP_CLASS_DB.py : 데이터베이스(테이블) 생성 및 저장. 최신 데이터 100 건만 저장
 CP_CLASS_Project.py : 각각의 프로세스의 클래스. 3 개의 프로세스 구동 및 5 개의 큐를
 인수로 받음
 CP_Main.py : 화분의 ID, 큐 생성, 프로세스 생성 및 구동 후 종료시까지 대기

- 데이터베이스 파일

CP_Database.db : 챗봇 단말의 임시 데이터베이스

3.2.2. 서버

- 소스 파일

SV_Main.py : 단말기 및 안드로이드와 통신이 가능한 서버 코드
 SV_Database.py : 서버에서 필요한 데이터베이스를 관리하고 저장하는 기능.
 채팅엔진의 객체를 객체를 생성 후 메소드 호출.

- 데이터베이스 파일

SV_Database.db : 서버의 메인 데이터베이스

3.2.3. 채팅 엔진 (서버의 SV_Database.py 에서 객체 생성되어 실행)

- 소스 파일

CE_CLASS_Chatbot.py : chatbot 메인 클래스

CE_CLASS_PlantFeeling.py : 종합 정보를 판단하여 식물의 기분을 결정하는 클래스
 CE_CLASS_Crawling.py : 날씨를 크롤링하는 클래스. BeautifulSoup 사용
 CE_DATA_Chatbot.py : 대화를 위한 문장, 단어들 저장된 파일
 CE_DATA_PlantStandardDB.py: 식물재배 정보

3.2.4. 안드로이드

- 소스 파일

- 액티비티

MainActivity.java : 앱 시작하기 버튼을 통해 socketService.java (바인딩서비스) 실행
 Login.java : ID 및 Password 입력으로 사용자 정보 전달 및 서버로부터 사용자 정보 수신 / 사용자 아이디 생성
 Menu.java : 서버로부터 수신된 사용자 정보를 바탕으로 menu 화면 UI 갱신 / 챗봇과 대화기능 연결 / 챗봇의 상세보기 기능 연결 / 챗봇 추가기능 연결
 talk.java : chatService.java(바인딩 서비스)를 통해 8181 포트로 챗봇과의 대화기능 구현 / 대화한 데이터는 chatService.java의 ArrayList chatstorage에 저장하여 chatHistory 버튼으로 저장한 대화 불러오기 기능 구현
 plantinfo.java : 챗봇의 상세 센서 데이터값 실시간 갱신 기능 구현 / 챗봇의 이름, 품종 변경기능 구현 / 챗봇의 자동 물주기 기능 구현
 add.java : 챗봇 추가기능 구현 - 이름, 품종, 챗봇 ID, 날짜 입력

- 바인딩서비스

socketService.java : socket을 연결하는 clientThread를 통해 sendThread, receiveThread 구현. 이를 통해 사용자 정보 송수신
 chatService.java : socket을 새로 연결하여 화분과 chatting하는 service 제공
 userService.java : 서버로부터 받은 json 타입 명령어를 분할 / 챗봇 정보값은 class potIDInfoClass에 담아 ArrayList potlist_potIDInfo에 담음 / 챗봇 센서값은 class potSensorDataClass에 담아 ArrayList potsensorlist_potSensorData에 담음

- 레이아웃

activity_main.xml : 진입 화면
 login.xml : 로그인 화면
 menu.xml : 챗봇 모아보기 화면
 talk.xml : 채팅 화면
 plantinfo.xml : 챗봇 상세보기 화면
 add.xml : 챗봇 등록하기 화면
 createUserDialog.xml : 유저 등록 다이얼로그
 dialog1.xml : 챗봇 이름 변경 다이얼로그
 dialog2.xml : 챗봇 품종 변경 다이얼로그

- 매니페스트

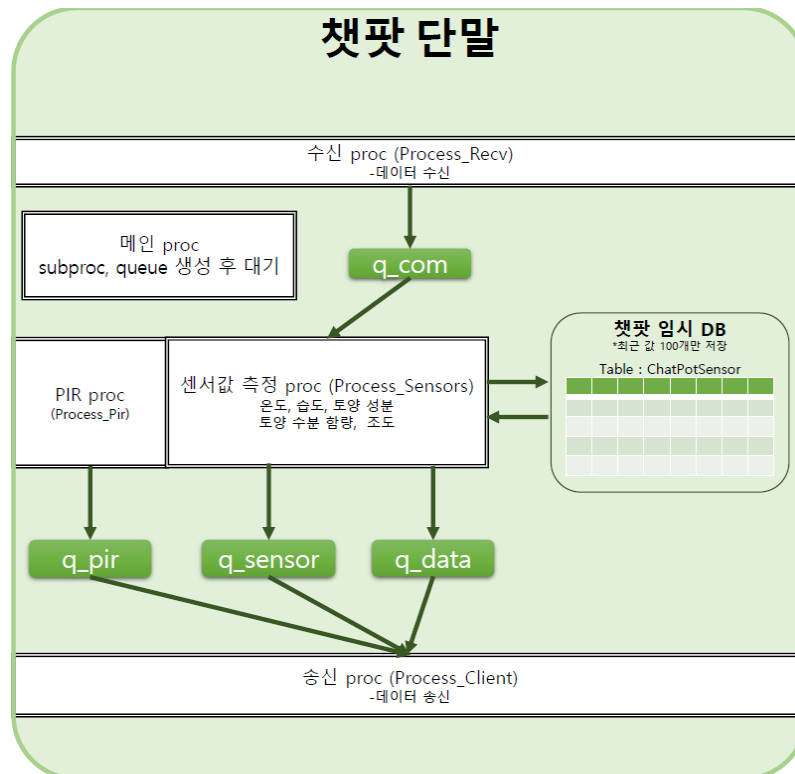
AndroidManifest : chatService, socketService, userService 등록 / Internet permission

- 리소스 파일 목록

plant.png, plant2.png, plant3.png, boy.png, chat.png, loupe.png, refreshbutton.png,
title_chatpot1.png, plus.png

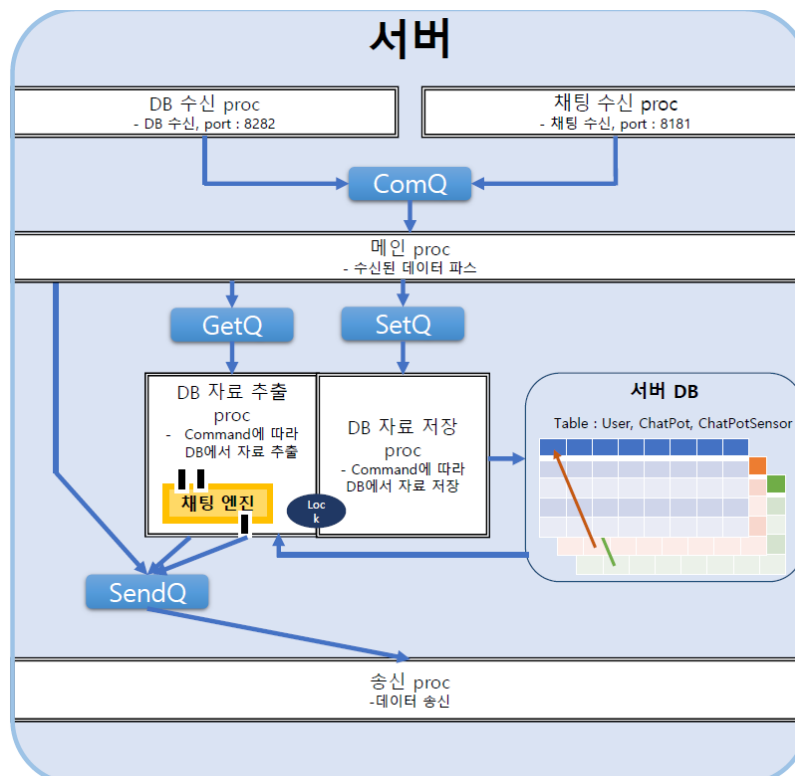
3.3 시스템 프로세스/스레드 구성도

3.3.1 챗팟 단말



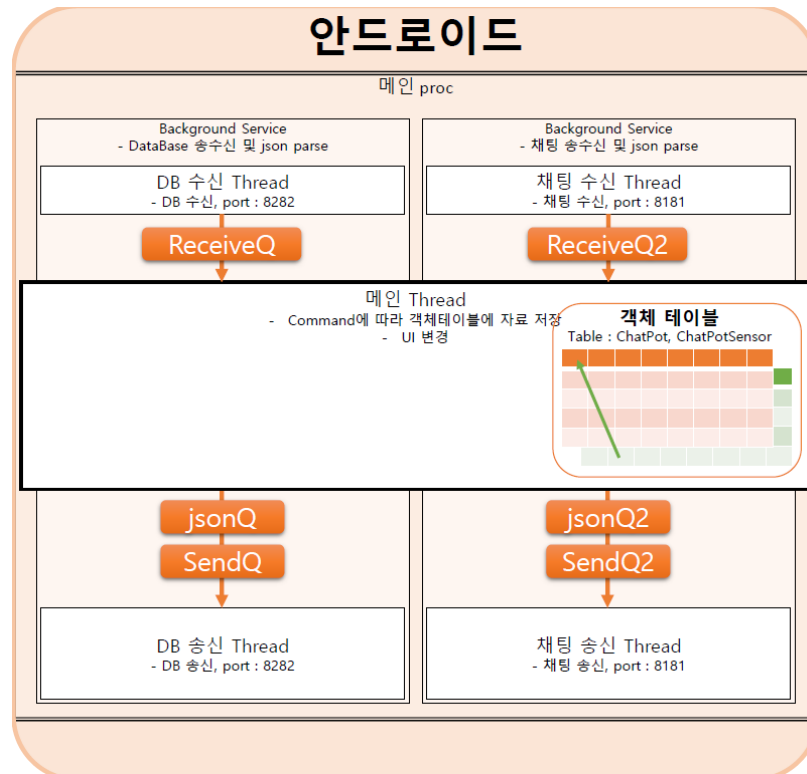
[그림 3-3-1. 시스템 프로세스/스레드 구성도 : 챗팟 단말]

3.3.2 서버



[그림 3-3-2. 시스템 프로세스/스레드 구성도 : 서버]

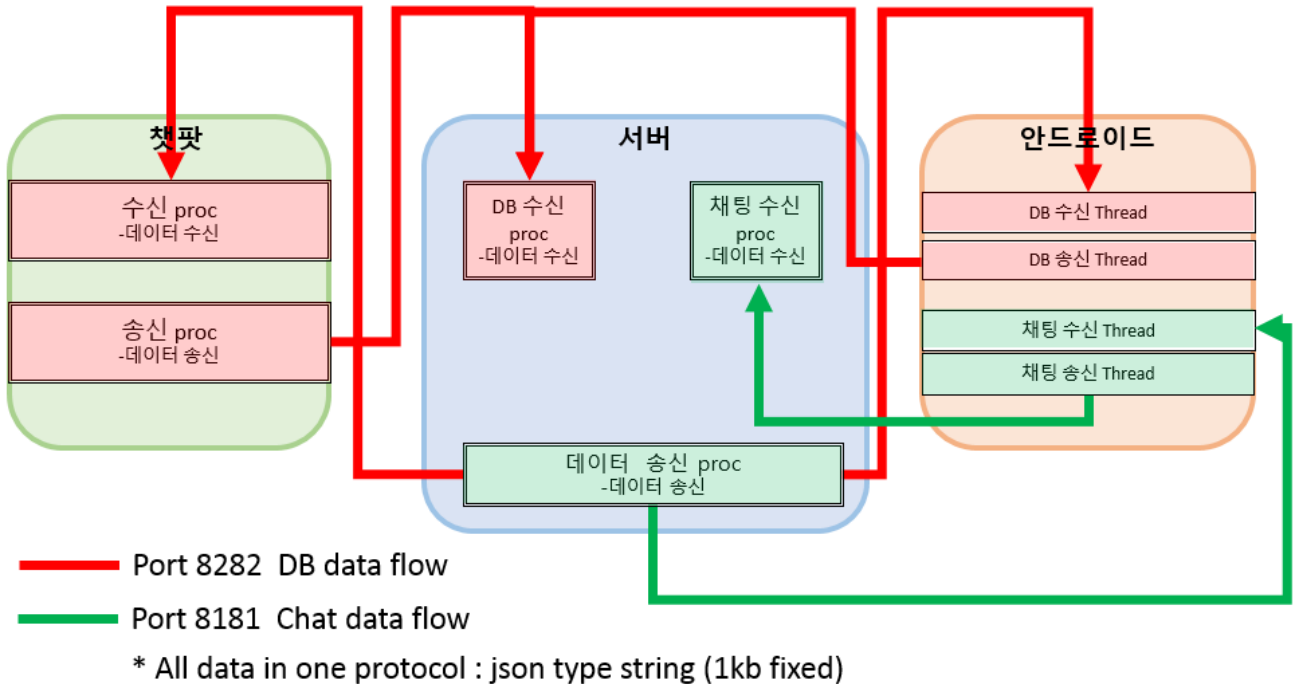
3.3.3 안드로이드



[그림 3-3-3. 시스템 프로세스/스레드 구성도 : 안드로이드]

3.4 네트워크 중심의 시스템 구성도

3.4.1 일반 데이터 및 채팅 데이터 흐름도



3.4.2 프로토콜 설계

프로토콜 형식 : 1kb 의 json 형식 문자열

사용 구간 : 통신 전 구간

기본 형식 :

```
{
  "header" : {
    "self" : "챗팟ID" || "유저ID" || "server" ...
    "command" : "set" || "login" || "refresh" ...
    "option" : {
      "해당 command에 대한 option"
    } //option 끝
  } //header 끝
  "data" : {
    "해당 command에 대한 data"
  } //data 끝
} // 전송 끝
```

* || 는 '또는'의 의미임
 밑줄 친 값은 해당되는 값이며 변동 가능
 // 이후의 문장은 설명하기 위함임

사용 예시 :

사례 1) 챗팻에서 서버로 갱신된 센서 데이터 값 갱신 전송

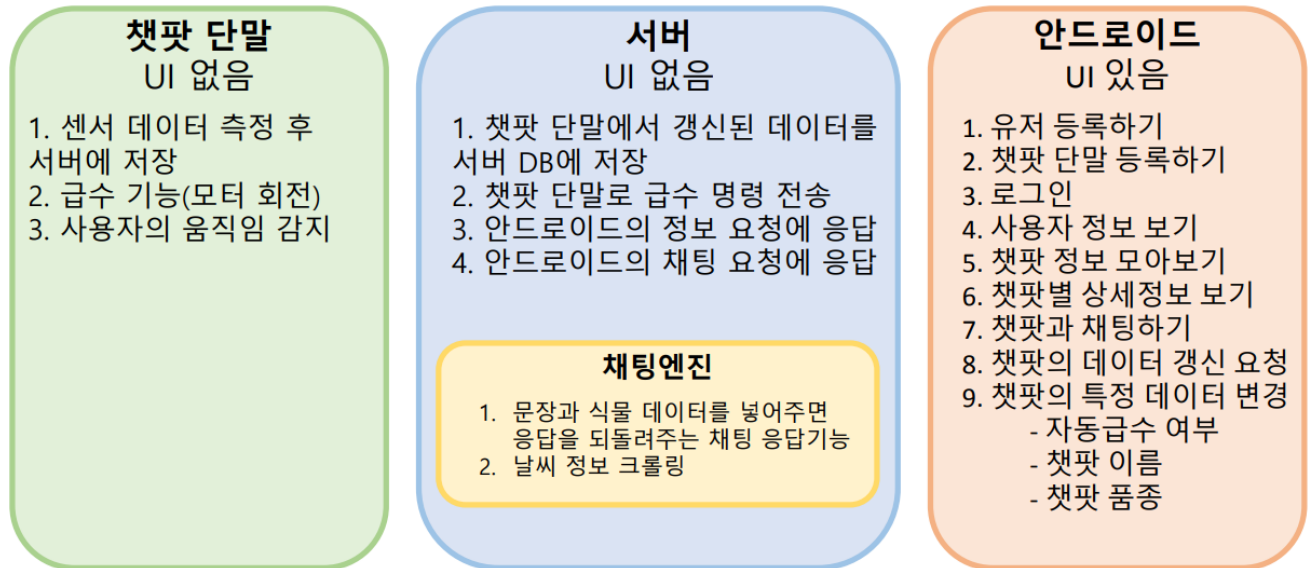
```
{
    "header"      : {
        "self"      : "100" // 챗팻 단말 ID가 100이라는 의미
        "command"   : "set"  // 서버의 DataBase를 set 하겠음
        "option"    : {
            "all"     // DataBase의 모든 값에 대해 갱신하겠음
        } //option 끝
    } //header 끝
    "data"        : {
        "Temp" : "23" //온도 값
        "Humi" : "50" //습도 값
        "Ligh" : "250" //조도 값
        "Nutr" : "1.1" //토양 염류도(토양의 영양 상태) 값
        "Grou" : "50" //토양 수분 함량 값
    } //data 끝
} // 전송 끝
```

사례 2) 안드로이드에서 서버로 채팅 요청

```
{
    "header"      : {
        "self"      : "10" // 유저 ID가 10이라는 의미
        "command"   : "chat" // 서버로 채팅을 요청하겠음
        "option"    : {
            "100"    // 대상 챗팻 단말의 ID가 100임
        } //option 끝
    } //header 끝
    "data"        : {
        "Hello. How are you today?" // 채팅 내용
    } //data 끝
} // 전송 끝
```

3.5 전체 & 플랫폼별 기능 목록 및 동작 시나리오

3.5.1 플랫폼별 기능 목록 및 시스템 동작 시나리오



[그림 3.5.1 플랫폼별 기능 목록]

기능에 따른 시스템 동작 시나리오		
플랫폼	기능	시스템 동작 시나리오
챗봇 단말	센서 데이터 측정 후 서버에 저장	1. 센서 데이터 측정 2. 측정 값을 데이터 송신 프로세스로 보냄 3. 데이터 송신 프로세스에서 서버로 센서 데이터 송신
	급수 기능 (모터 회전)	1. 데이터 수신 프로세스에서 '급수 명령' 수신 2. 급수 (모터 회전)
	사용자의 움직임 감지	1. 사용자 감지(PIR) 프로세스에서 움직임 감지 2. 감지 시 데이터 송신 프로세스로 감지 신호 보냄 3. 데이터 송신 프로세스에서 서버로 감지 신호 송신
서버	챗봇 단말에서 갱신된 데이터를 서버 DB에 저장	1. 데이터 수신 프로세스에서 데이터 수신 2. 메인 프로세스에서 데이터 요청의 종류와 값 파싱 3. 파싱된 데이터를 DB저장 프로세스로 보냄 4. DB저장 프로세스에서 DB에 저장
	챗봇 단말로 급수 명령 전송	1. 챗봇 단말로부터 데이터 수신 시 토양의 수분 함량도를 체크 2. 토양의 수분 함량도가 기준치보다 미달일 경우 DB에서 해당 챗봇의 자동급수 ON/OFF여부를 체크 3. 해당 챗봇의 자동급수가 ON일 경우 데이터 송신 프로세스로 '급수 명령' 보냄 4. 데이터 송신 프로세스에서 챗봇 단말로 '급수 명령' 송신

		안드로이드의 정보 요청에 응답	1. 데이터 수신 프로세스에서 데이터 수신 2. 메인 프로세스에서 데이터 요청의 종류와 값 파싱 3-1. DB에 값을 저장하는 경우 : DB저장 프로세스로 파싱된 데이터를 보내어 DB에 저장. 필요에 따라 갱신된 값을 회신함 3-2. DB에서 값을 읽는 경우 : DB추출 프로세스로 요청된 데이터를 보내어 값을 읽음. 읽은 값을 요청된 기기로 회신
		안드로이드의 채팅 요청에 응답	1. 채팅 수신 프로세스에서 채팅 수신 2. 메인 프로세스에서 채팅 요청이 들어온 기기를 식별, 해당 기기의 Device ID와 채팅 내용을 DB추출 프로세스로 보냄 3. 해당 기기에 대한 정보와 채팅 내용을 채팅엔진에 입력 4. 채팅엔진으로부터 응답 값을 받아서 데이터 송신 프로세스로 보냄 5. 데이터 송신 프로세스에서 안드로이드 기기의 채팅 수신 프로세스로 송신
	채팅엔진	채팅 응답기능	1. 서버로부터 채팅 요청된 기기의 데이터 값과 사용자가 요청한 채팅 내용을 받음 2. 요청된 채팅에 대해 의도 분석 3. 분석된 의도에 따라 응답 생성 4. 생성된 응답을 서버로 리턴
		날씨 정보 크롤링	1. '채팅 응답기능 : 시나리오 2'에서 해석된 의도가 '오늘의 날씨'일 경우 네이버에서 날씨 정보 크롤링 2. 응답에 날씨 정보 첨부
	안드로이드	유저 등록	1. 등록될 유저의 ID와 Password를 입력 받아 데이터 송신 스레드로 보냄 2. 데이터 송신 스레드에서 서버로 유저 등록 요청
		챗봇 단말 등록	1. 등록될 챗봇 단말의 고유 식별번호를 입력 받아 현재 로그인 된 유저의 ID와 함께 데이터 송신 스레드로 보냄 2. 데이터 송신 스레드에서 서버로 챗봇 단말 등록 요청
		로그인	1. 입력된 유저의 ID와 Password를 데이터 송신 스레드로 보냄 2. 데이터 송신 스레드에서 서버로 로그인 요청 3. 로그인 성공 시 해당 유저가 보유한 모든 정보를 데이터 수신 스레드에서 수신하여 내부 객체에 저장
		사용자 정보 보기	1. 객체에 저장된 정보를 UI에 표시
		챗봇 정보 모아보기	1. 객체에 저장된 정보를 UI에 표시
		챗봇별 상세정보 보기	1. 객체에 저장된 정보를 UI에 표시
		챗봇과 채팅	1. 사용자로부터 입력 받은 채팅 내용과 해당 챗봇 단말의 식별번호를 채팅 송신 스레드로 보냄

		2. 해당 내용을 서버로 송신 3. 서버로부터의 채팅 응답을 채팅 수신 스레드에서 수신 4. 수신된 채팅 내용을 객체에 저장 5. 객체에 저장된 정보를 UI에 표시
	채팅의 데이터 갱신 요청	1. 데이터를 갱신할 채팅 단말의 식별번호를 데이터 송신 스레드로 보냄 2. 서버로 해당 채팅 단말의 데이터 갱신 요청 송신 3. 서버에서 갱신된 데이터를 수신하여 내부 객체에 저장 4. 객체에 저장된 정보를 UI에 표시
	채팅의 특정 데이터 변경	1. 데이터를 수정할 채팅 단말의 식별번호와 함께 어떤 데이터를 수정할 것인지 데이터 송신 스레드로 보냄 2. 서버로 해당 채팅 단말의 데이터 수정 요청 송신

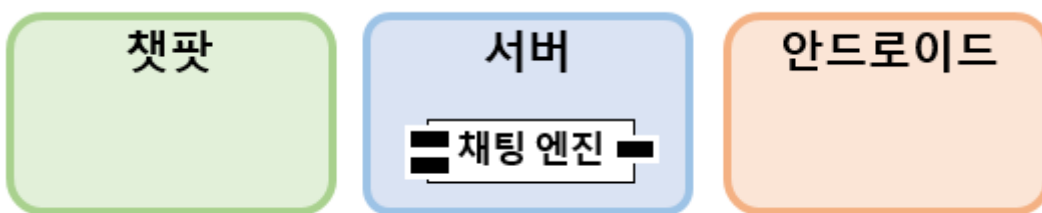
3.5.2 주요 기능의 통신 및 시스템 흐름도

3.5.2.1 주요 기능 목록

- (1) 채팅 단말에서 서버로의 주기적인 센서 데이터 값 저장
- (2) 안드로이드에서 서버로 특정 유저 로그인 요청 후 성공 시 해당 유저의 전체 데이터 수신
- (3) 안드로이드에서 서버로 특정 채팅 단말에 대한 센서 데이터 값 갱신요청 후 해당 값 수신
- (4) 안드로이드에서 서버로 특정 채팅 단말에 대한 채팅 요청

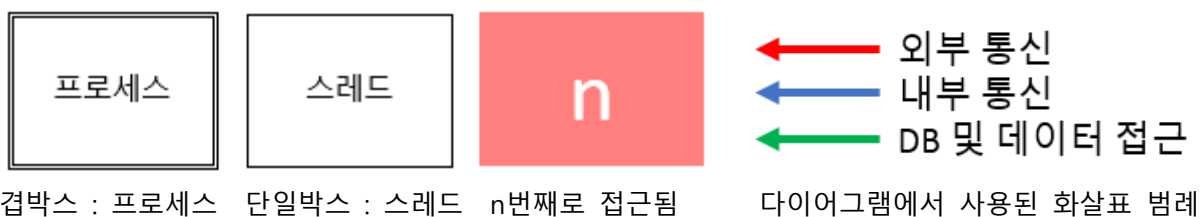
3.5.2.2 흐름과 순서도에 대한 설명

3.5.2.2.1 색상 별 플랫폼 구분



* 채팅엔진은 캡슐화 되어 서버에 탑재된다.

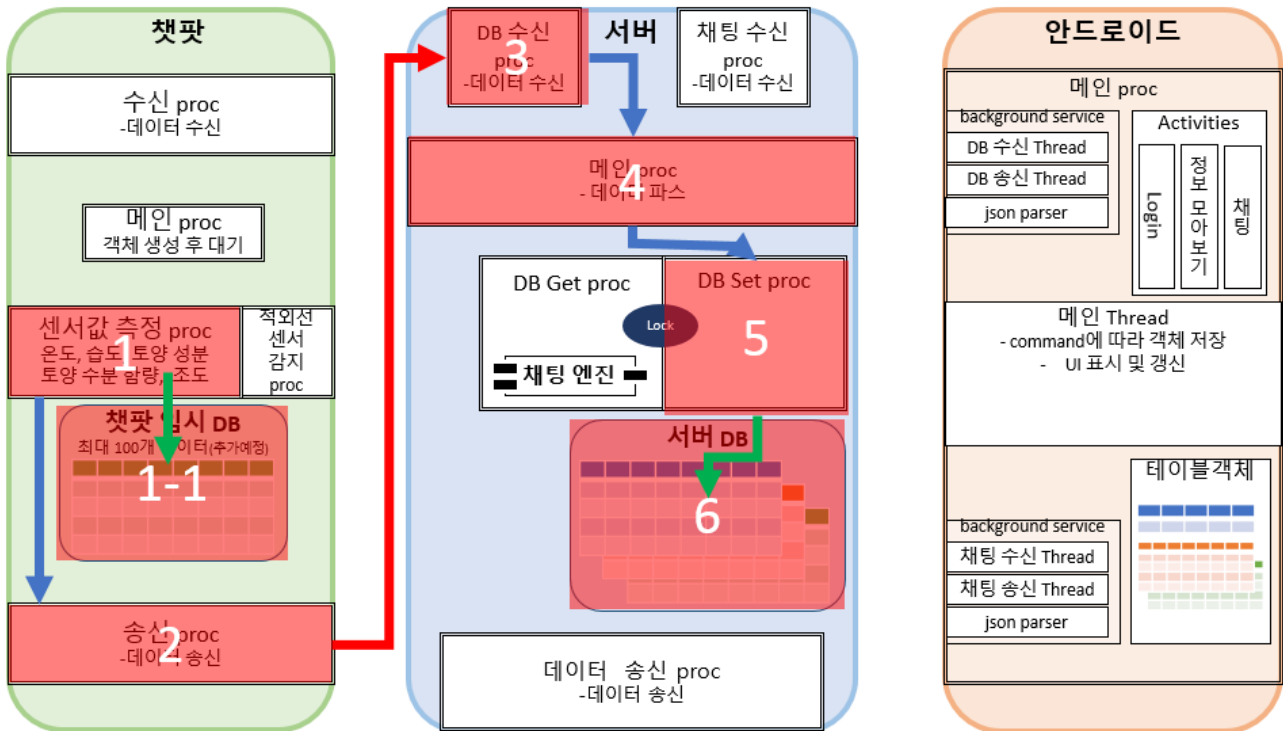
3.5.2.2.2 다이어그램 범례



3.5.2.3 주요 기능의 흐름도

3.5.2.3.1 주요 기능 (1) :

챗봇 단말에서 서버의 DB에 주기적으로 센서 데이터 값을 저장하는 경우



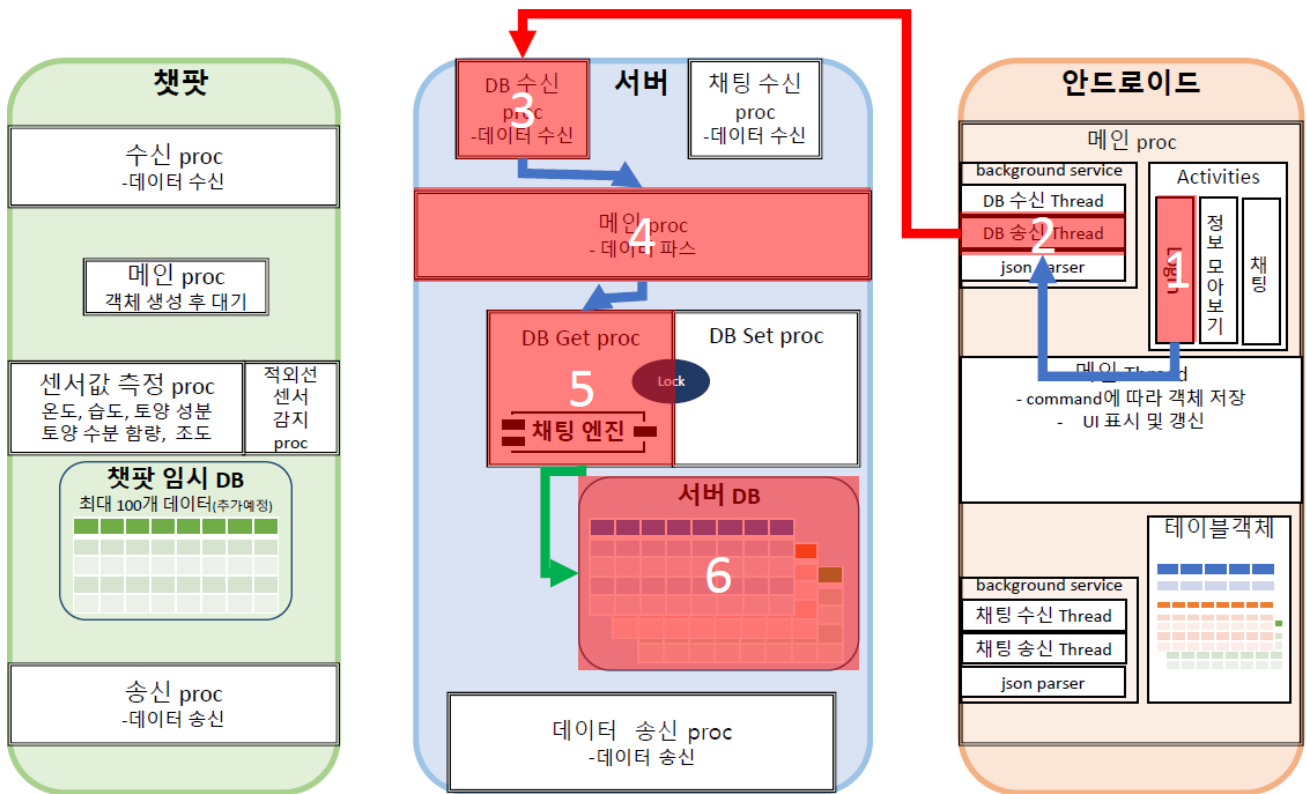
[그림 3.5.2.3 주요 기능 (1) 전체 흐름도]

설명 :

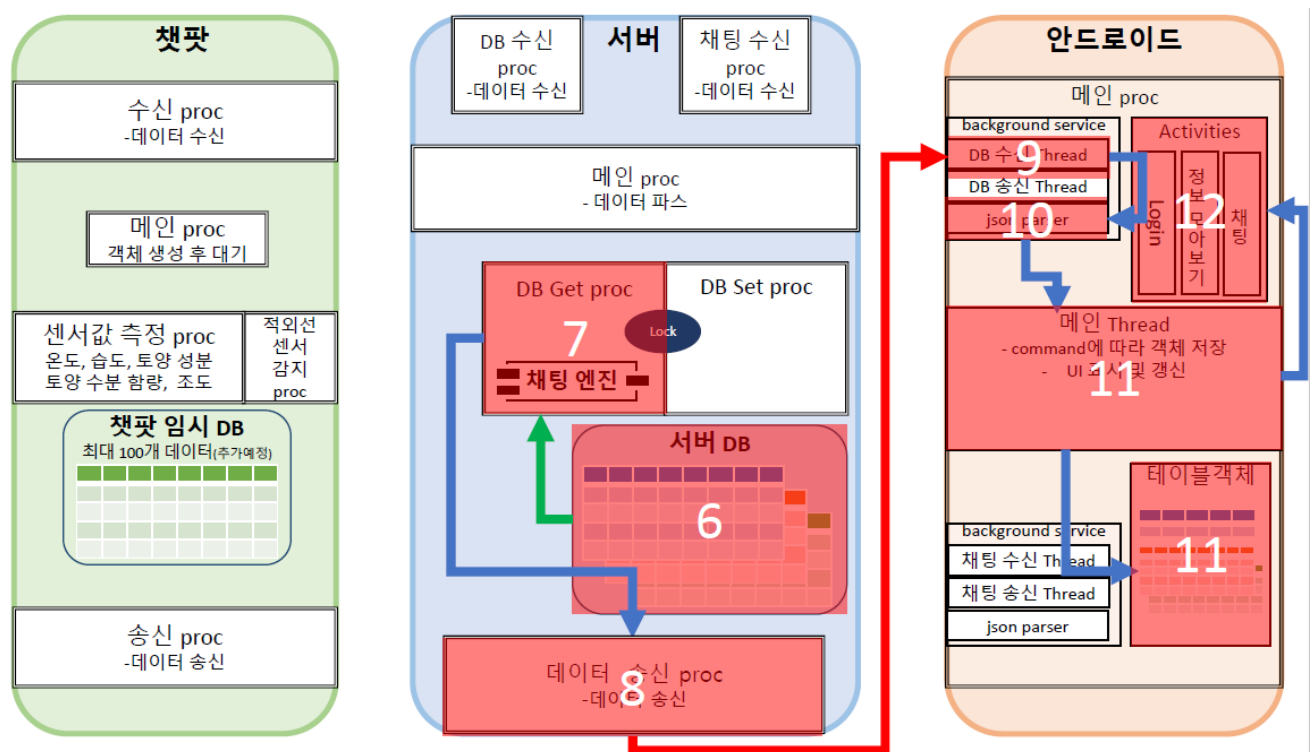
- 1) '센서값 측정 proc'에서 주기적으로 데이터 측정
- 1-1) 측정된 데이터를 임시 DB에 저장
- 2) 측정된 데이터 값을 '송신proc'에서 서버로 전송
- 3) 서버에서 수신한 데이터를 파싱하기 위해 '메인 proc'으로 전송, 시나리오 1의 상황 인지
- 4) 프로토콜에 맞추어 수신한 데이터를 파싱하여 'DB Set proc'으로 전송
- 5) 'DB Set proc'에서 수신한 데이터를 sqlite 쿼리로 변환
- 6) 서버 DB에 저장

3.5.2.3.2 주요 기능 (2) :

안드로이드에서 서버로 특정 유저 로그인 요청. 성공 시 해당 유저의 전체 데이터를 수신하는 경우



[그림 3.5.2.3.2.1 주요 기능 (2) 흐름도 1~6]



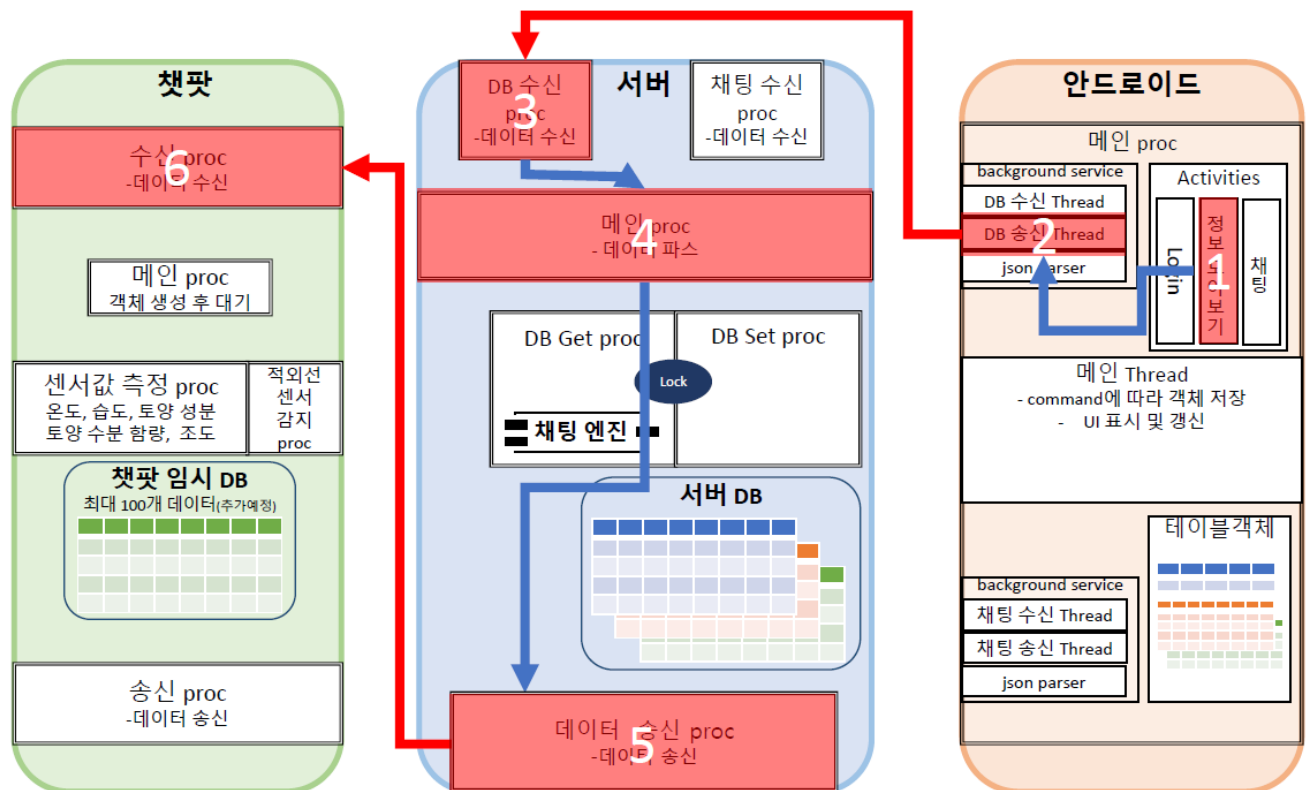
[그림 3.5.2.3.2.2 주요 기능 (2) 흐름도 7~12]

설명 :

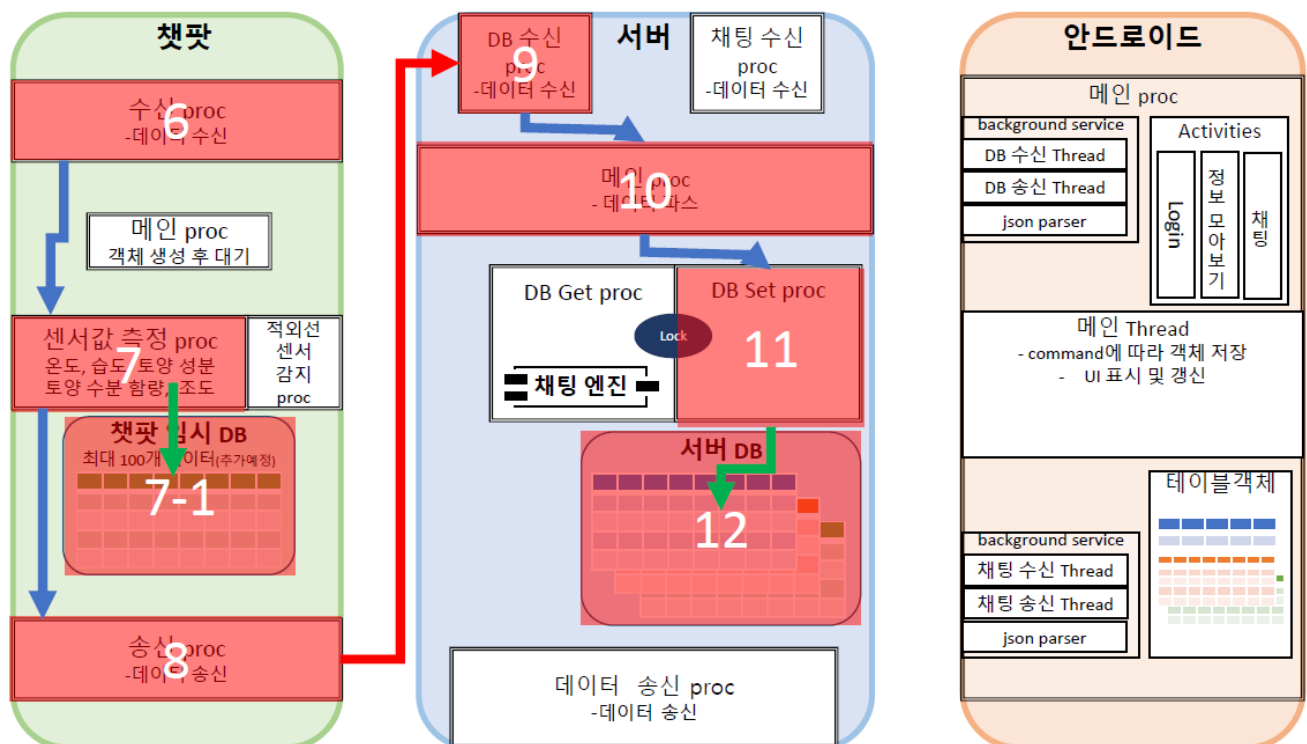
- 1) '정보 모아보기' 액티비티로부터 요청을 받음
- 2) 'DB송신Thread'에서 데이터 갱신 요청을 프로토콜에 맞추어 서버로 전송
- 3) 서버에서 수신한 값을 '메인 proc'으로 전송
- 4) '메인 proc'에서 해당 값을 파스, 로그인 요청 인지하여 'DB Get proc'으로 전송
- 5) 'DB Get proc'에서 수신한 데이터를 sqlite 쿼리로 변환
- 6) 서버 DB에 접근하여 해당 정보가 있는지 추출
- 7) 서버 DB에서 추출한 받은 유저 값과 로그인 요청이 일치하는지 확인
일치하는 경우 보유 챗봇의 정보를 안드로이드로 함께 전송함
- 8) 'DB송신proc'에서 프로토콜에 맞추어 안드로이드로 전송
- 9) 'DB수신Thread'에서 받은 값을 해독하기 위해 json parser로 전송
- 10) 'json parser'에서 로그인 시도에 대한 결과를 확인
- 11) 로그인 성공 시 함께 들어온 챗봇 데이터를 객체테이블과 UI객체에 대입
- 12) 화면 출력

3.5.2.3.3 주요 기능 (3) :

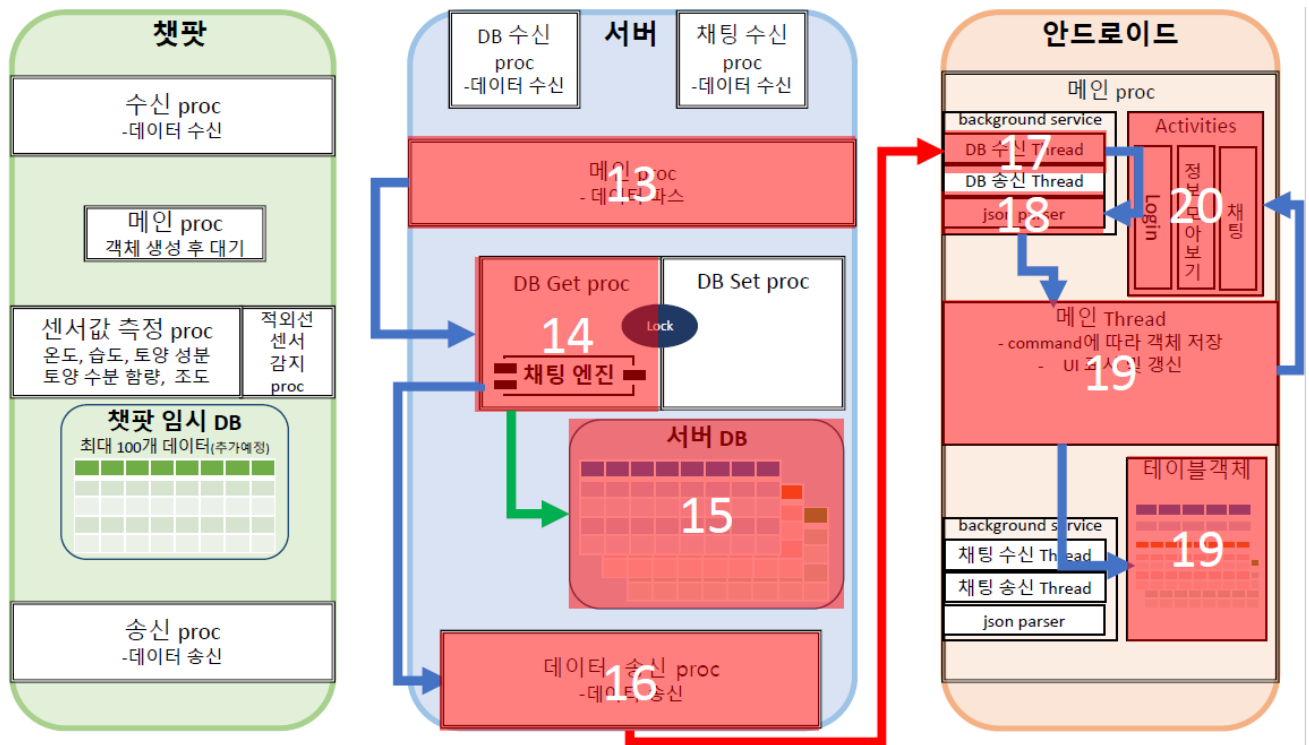
안드로이드에서 서버로 특정 챗봇 단말에 대한 센서 데이터 값 갱신요청 후 값을 수신하는 경우



[그림 3.5.2.3.3.1 주요 기능 (3) 흐름도 1~6]



[그림 3.5.2.3.3.2 주요 기능 (3) 흐름도 6~12]



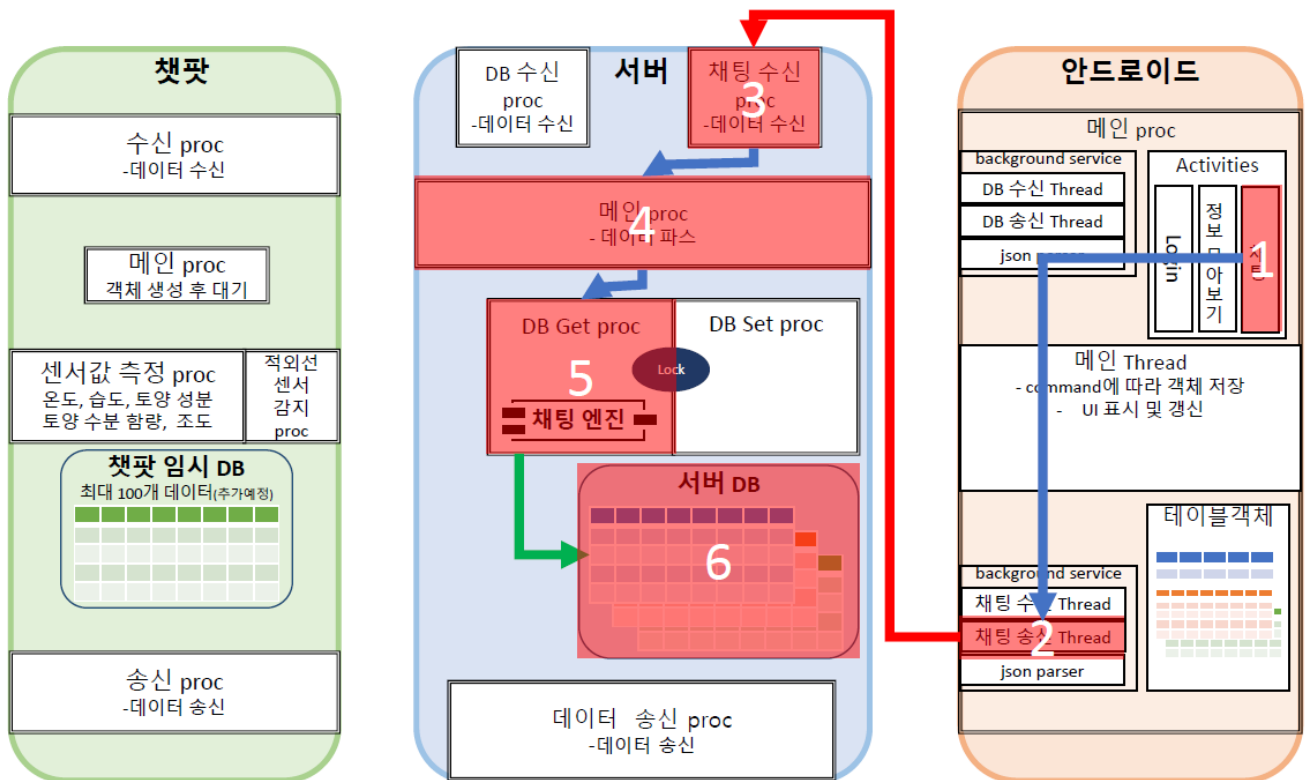
[그림 3.5.2.3.3.2 주요 기능 (3) 흐름도 13~20]

설명 :

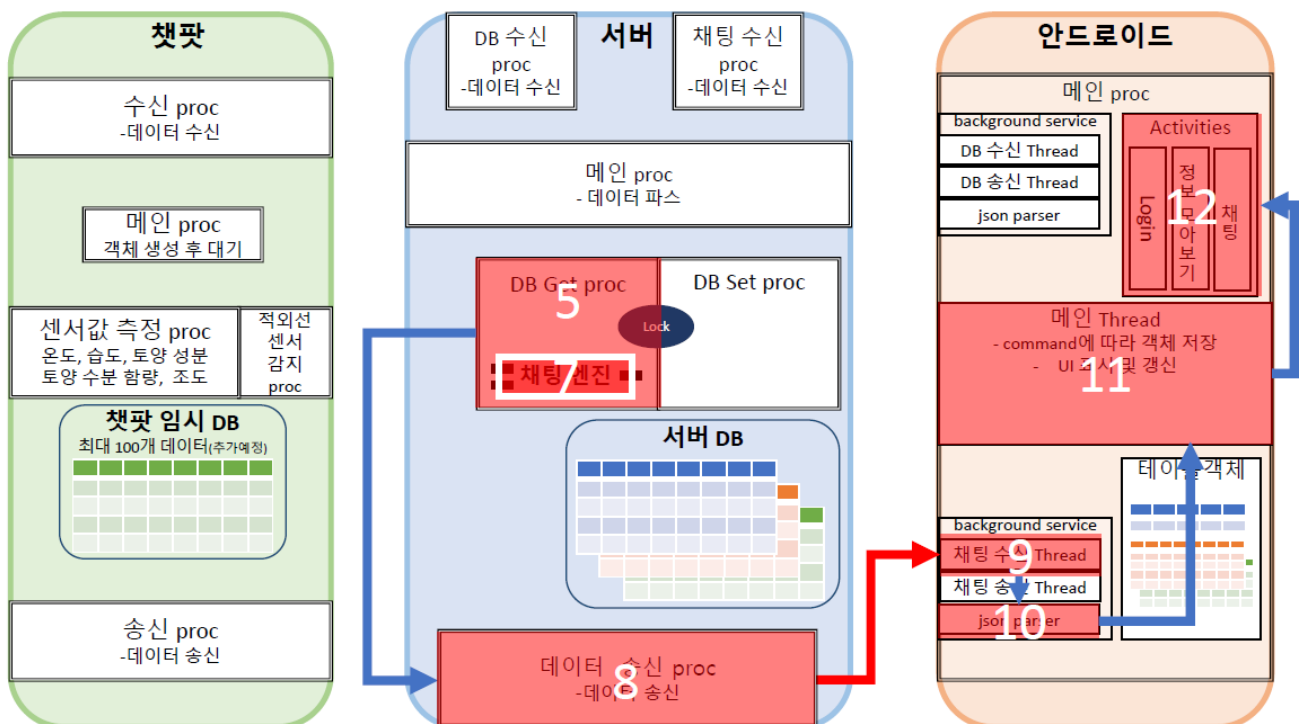
- 1) '정보 모아보기' 액티비티로부터 데이터 갱신 요청을 받아 'DB송신 Thread'로 전송
- 2) 'DB송신Thread'에서 해당 값을 프로토콜에 맞추어 서버로 전송
- 3) 서버에서 수신한 값을 '메인 proc'으로 전송
- 4) '메인 proc'에서 해당 값을 파스, 데이터 갱신 요청 인지하여 'DB 송신 proc'으로 전송
- 5) 'DB 송신 proc'에서 챗팻으로 데이터 갱신 요청 전송
- 6) 챗팻에서 데이터 갱신 요청 수신
- 7) '센서값 측정 proc'에서는 주기 간 sleep 중 데이터 갱신 요청이 들어오면 sleep에서 깨어나 센서 값 측정
- 8 ~ 12) 과정은 '시나리오 1'과 동일. 하지만 명령이 '데이터 저장'이 아닌 '데이터 갱신'이므로 DB 저장 후 '메인 proc'으로 돌아옴
- 13) 챗팻으로부터 들어온 데이터를 서버 DB에 저장 후 다시 해당 값에 접근하기 위해 'DB Get proc'으로 이동
- 14 ~ 15) 'DB Get proc'에서 최신 데이터 값을 읽어서 'DB 송신 proc'으로 전송
- 16) 'DB 송신 proc'에서 해당 값을 프로토콜에 맞추어 안드로이드로 송신
- 17) 안드로이드의 'DB 수신 Thread'에서 수신
- 17~20) 본 과정은 '시나리오2'의 과정 9~12)와 동일함

3.5.2.3.4 주요 기능 (4) :

안드로이드에서 서버로 특정 챗봇 단말에 대한 채팅 요청



[그림 3.5.2.3.4.1 주요 기능 (4) 흐름도 1~6]



[그림 3.5.2.3.4.2 주요 기능 (4) 흐름도 7~12]

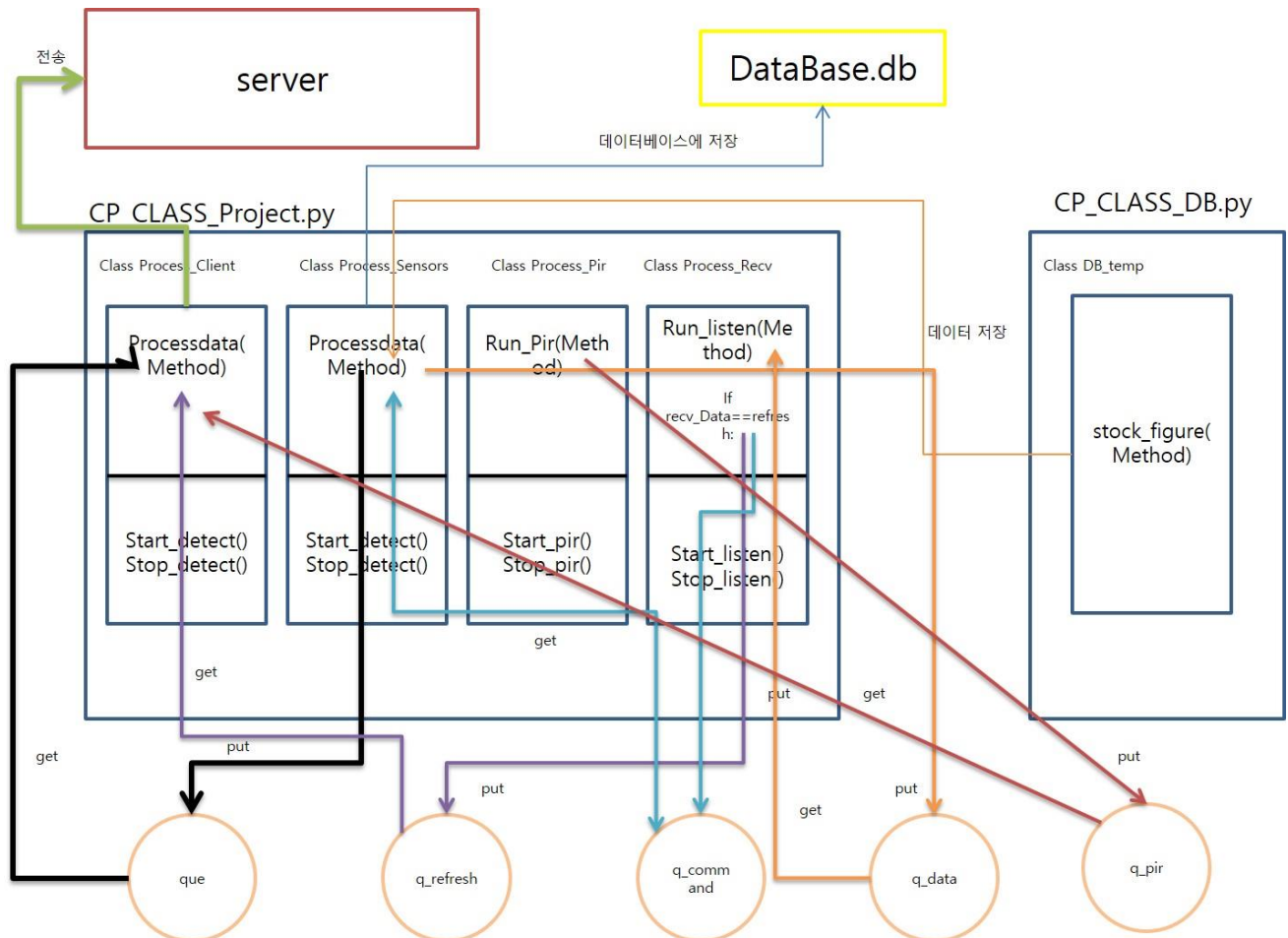
설명 :

- 1) '채팅' 액티비티로부터 채팅 요청을 받아 '채팅 송신 Thread'로 전송
- 2) '채팅 송신Thread'에서 해당 값을 프로토콜에 맞추어 서버로 전송
- 3) 서버에서 수신한 값을 '메인 proc'으로 전송
- 4) '메인 proc'에서 해당 값을 파스, 채팅 요청 인지하여 'DB Get proc'으로 전송
- 5) 채팅 엔진에 해당 챗봇의 데이터 넘겨주기 위해 'DB Get proc'에서 해당 챗봇의 데이터 추출
- 6) DB 접근하여 데이터 추출
- 7) DB에서 추출한 해당 챗봇의 데이터에 기반한 문자열을 채팅 엔진으로부터 받아 '채팅 송신 proc'으로 전송
- 8) 서버로부터 안드로이드의 '채팅 수신 Thread'로 프로토콜에 맞추어 전송
- 9) 전송받은 데이터를 파서에게 전송
- 10) 파스한 데이터를 '메인 Thread'에 전송
- 11) '메인 Thread'에서 해당 챗봇의 UI객체에 채팅 문자열 대입
- 12) 화면 출력

3.6 클래스 설명 및 기능 정의

3.6.1 챗봇 단말

3.6.1.1 챗봇 단말 클래스 관계도



3.6.1.2 챗봇 단말 기능 설명

3.6.1.2.1 CP_CLASS_GPIO_Library.py

class DCMOTOR : 자동 급수 표현

필드:

Ports : 사용하는 포트 번호
Portname : 사용하는 포트의 이름

메소드 :

DCMOTOR_forward() : 모터를 시계 방향으로 구동
DCMOTOR_backward() : 모터를 반시계 방향으로 구동

DCMOTOR_stop() : 모터 정지

class TempHumi : 온,습도 측정

필드 :

Temp : 측정한 온도 값

Humi : 측정한 습도 값

메소드 :

measure_tmp() : 온도 측정

measure_humi() : 습도 측정

class Light : 조도 측정

메소드 :

readLight() : 조도 값 측정

class SPI : ADC 를 통한 센서 값 측정

메소드 :

Printcds() : ADC 를 통한 조도 값 측정 (땅의 영양 상태를 대신함)

Printvtr() : ADC 를 통한 가변저항의 값 측정 (땅의 수분 함량 상태를 대신함)

class PIR : Passive Infra-red Sensor 측정

필드 :

Ports : 사용하는 포트 번호

direction: 사용하는 포트의 I/O

메소드 :

PIR_motionDetection() : 인체감지 시 True 리턴

3.6.1.2.2 CP_CLASS_DB.py

class DB_temp

필드 :

__dataBasePath : 저장되는 데이터베이스 이름

메소드 :

__init__() : 데이터베이스 생성

Stock_figure() : 데이터 저장 (최대 100 개까지)

3.6.1.2.3 CP_CLASS_Project.py

class Process_Client : 각 센서로부터 받은 값을 서버로 보내는 프로세스

필드 :

Id : 화분의 Id 값

Que : 측정한 센서 값을 전달하는 큐
q_pir : PIR 센서 값을 전달하는 큐
q_refresh : refresh 커맨드가 들어왔을 때, 센서 값을 전달하는 큐

메소드 :

ProcessData() : 각 센서의 값을 서버로 전송
Start_client() : 프로세스 시작
Stop_client() : 프로세스 종료

class Process_Sensors : 센서 값들을 측정하여 큐에 넣는 프로세스

필드 :

th : 온도 측정 값
light : 조도 측정 값
spi : ADC 를 통해 측정한 조도와 가변저항 값
id : 화분의 ID
que : 센서 값들을 전달하는 큐
dataque : refresh 커맨드가 들어왔을 때 임시로 센서 값을 전달하는 큐
comque : 커맨드를 전달하는 큐
db : 데이터베이스 클래스 객체 생성

메소드 :

ProcessSensor() : 각 센서의 값을 큐에 전달하고 데이터베이스에 저장
Start_detect() : 프로세스 시작
Stop_detect() : 프로세스 종료

class Process_Pir : 움직임을 감지하면 True 를 큐에 넣는 프로세스

필드 :

Id : 화분의 ID 값
Que : 측정한 센서값을 전달하는 큐
q_pir : pir 센서 측정 값을 전달하는 큐
q_refresh : refresh 커맨드가 들어왔을 때 센서 값을 전달하는 큐
comq : 커맨드를 전달하는 큐
count : 친밀도를 위한 빈도수 저장

메소드 :

run_pir() : 사용자가 감지되는 빈도수에 따라 친밀도 상승, 하강하는 것을 큐에 전달
Start_pir() : 프로세스 시작
Stop_pir() : 프로세스 종료

class Process_Recv : 커맨드를 수신하여 해당 커맨드를 수행하는 프로세스

필드 :

Id : 화분의 ID 값
Que : 측정한 센서값을 전달하는 큐

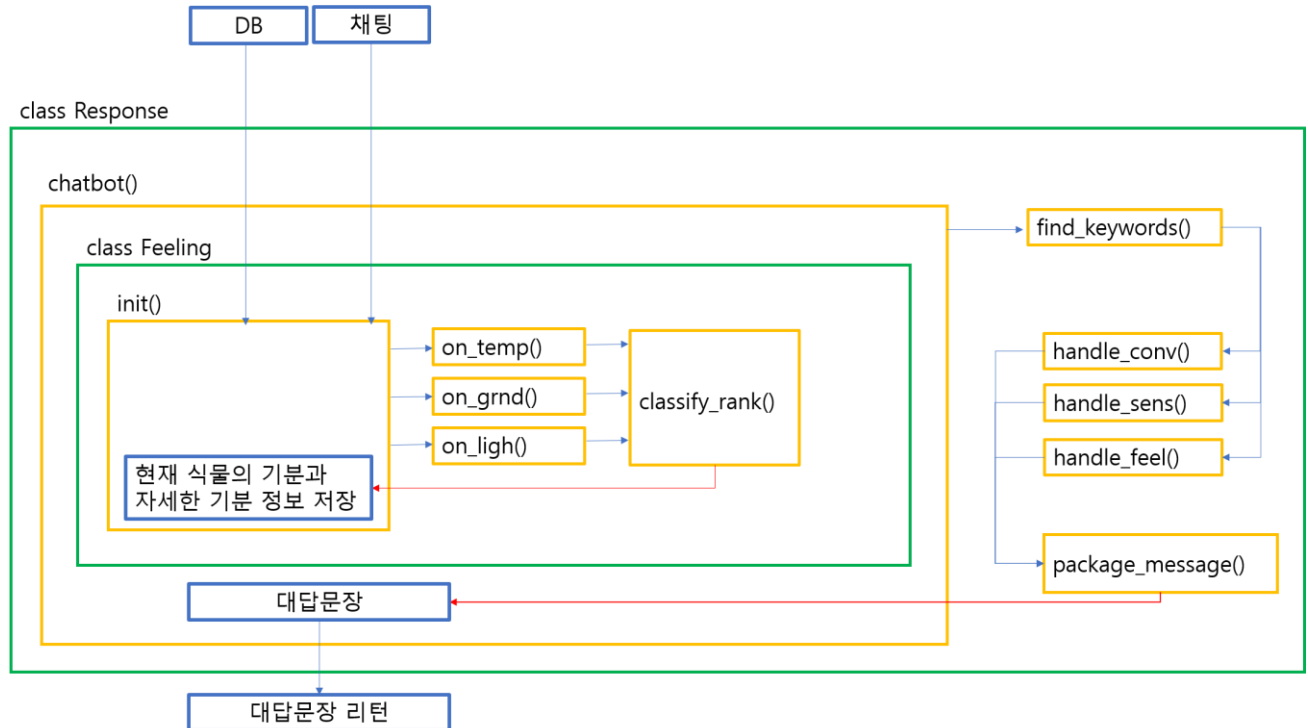
q_pir : pir 센서 측정 값을 전달하는 큐
dataque : 센서 값이 전달되는 임시 큐
comque : 커맨드를 전달하는 큐
q_refresh : 커맨드가 들어왔을 때 센서 값을 전달하는 큐

메소드 :

run_listen() : 서버로부터의 커맨드를 수신
Start_listen() : 프로세스 시작
Stop_listen() : 프로세스 종료

3.6.3 채팅엔진

3.6.3.1 채팅엔진 클래스 관계도



3.6.3.2 채팅엔진 기능 설명

class Response : 채팅엔진 전체를 관장하는 클래스

메소드 :

- find_keywords() : 키워드를 추출하여 1) 기본 대화 2) 센서 값이 필요한 대화 3) 식물의 기분을 묻는 대화로 분류
- handle_conv() : 친밀도에 따른 분류
- handle_sens() : DB의 센서값을 문장에 포함
- handle_feel() : 식물 기분에 따른 분류
- package_message() : 전체적인 문장 구성

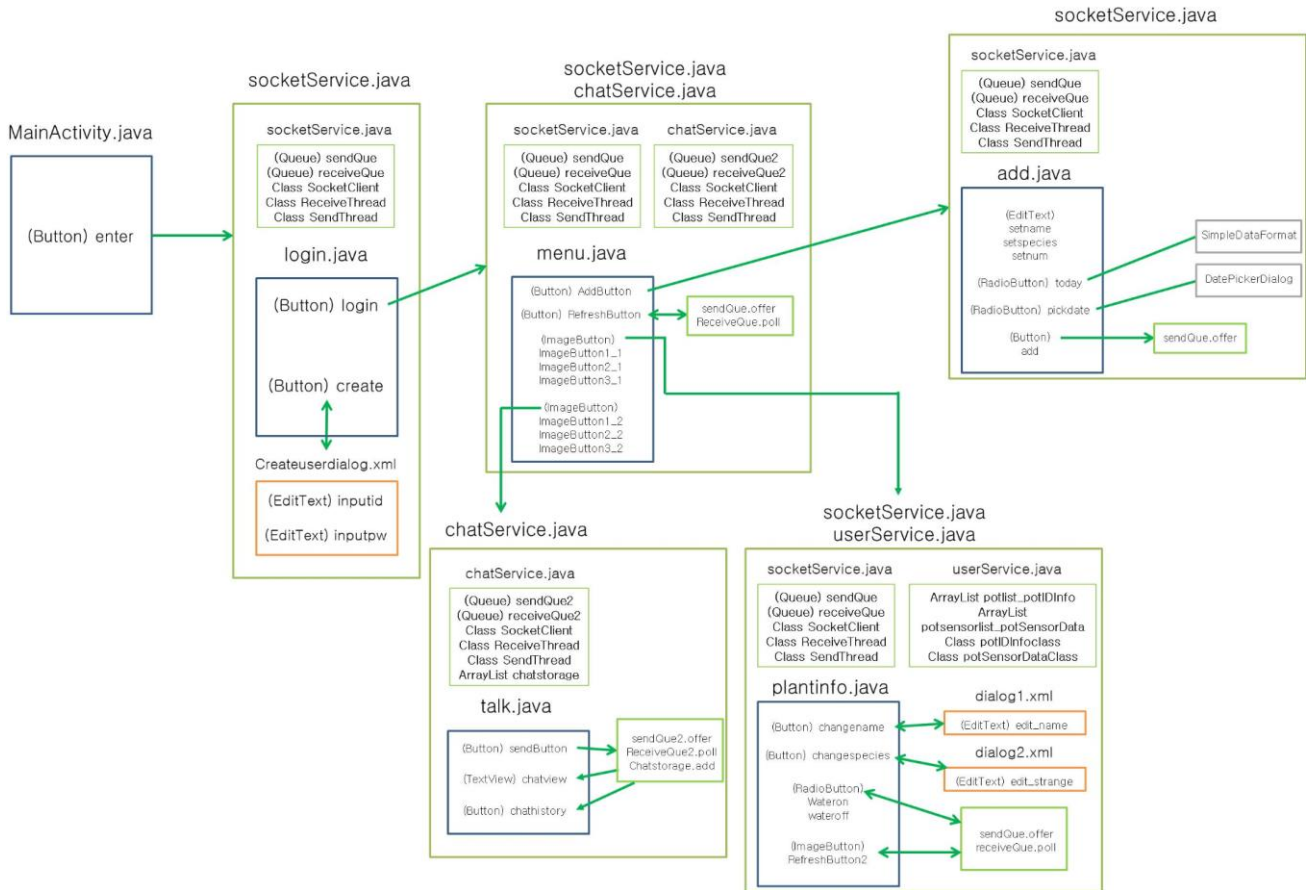
class Feeling : Response 내부 클래스이며 응답을 위한 식물의 기분을 결정하는 메소드로 이루어짐

메소드 :

- on_temp() : 해당 종의 재배정보와 비교하여 온도에 따른 기분을 등급으로 분류
- on_grnd() : 해당 종의 재배정보와 비교하여 토양 내 수분에 따른 기분을 등급으로 분류
- on_ligh() : 해당 종의 재배정보와 비교하여 조도에 따른 기분을 등급으로 분류
- classify_rank() : 각 등급을 전부 더하고 기분을 5 단계로 최종 분류

3.6.4 안드로이드

3.6.3.1 안드로이드 클래스 관계도



Activity :

- MainActivity.java : 앱 시작하기 버튼을 통해 socketService.java(바인딩 서비스) 실행
- login.java : ID 및 Password 입력으로 사용자 정보 전달 및 서버로부터 사용자 정보 수신 / 사용자 아이디 생성
- menu.java : 서버로부터 수신된 사용자 정보를 바탕으로 menu 화면 UI 갱신 / 챗팟과 대화기능 연결 / 챗팟의 상세보기 기능 연결 / 챗팟 추가기능 연결
- talk.java : chatService.java(바인딩 서비스)를 통해 8181 포트로 챗팟과의 대화기능 구현 / 대화한 데이터는 chatService.java의 ArrayList chatstorage에 저장하여 chathistory 버튼으로 저장한 대화 불러오기 기능 구현
- plantinfo.java : 챗팟의 상세 센서 데이터값 실시간 갱신 기능 구현 / 챗팟의 이름, 품종 변경기능 구현 / 챗팟의 자동 물주기 기능 구현
- add.java : 챗팟 추가기능 구현 - 이름, 품종, 챗팟 ID, 날짜 입력

Binding Service :

socketService.java : socket 을 연결하는 clientThread 를 통해 sendThread, receiveThread 구현.

이를 통해 사용자 정보 송수신

chatService.java : socket 을 새로 연결하여 화분과 chatting 하는 service 제공

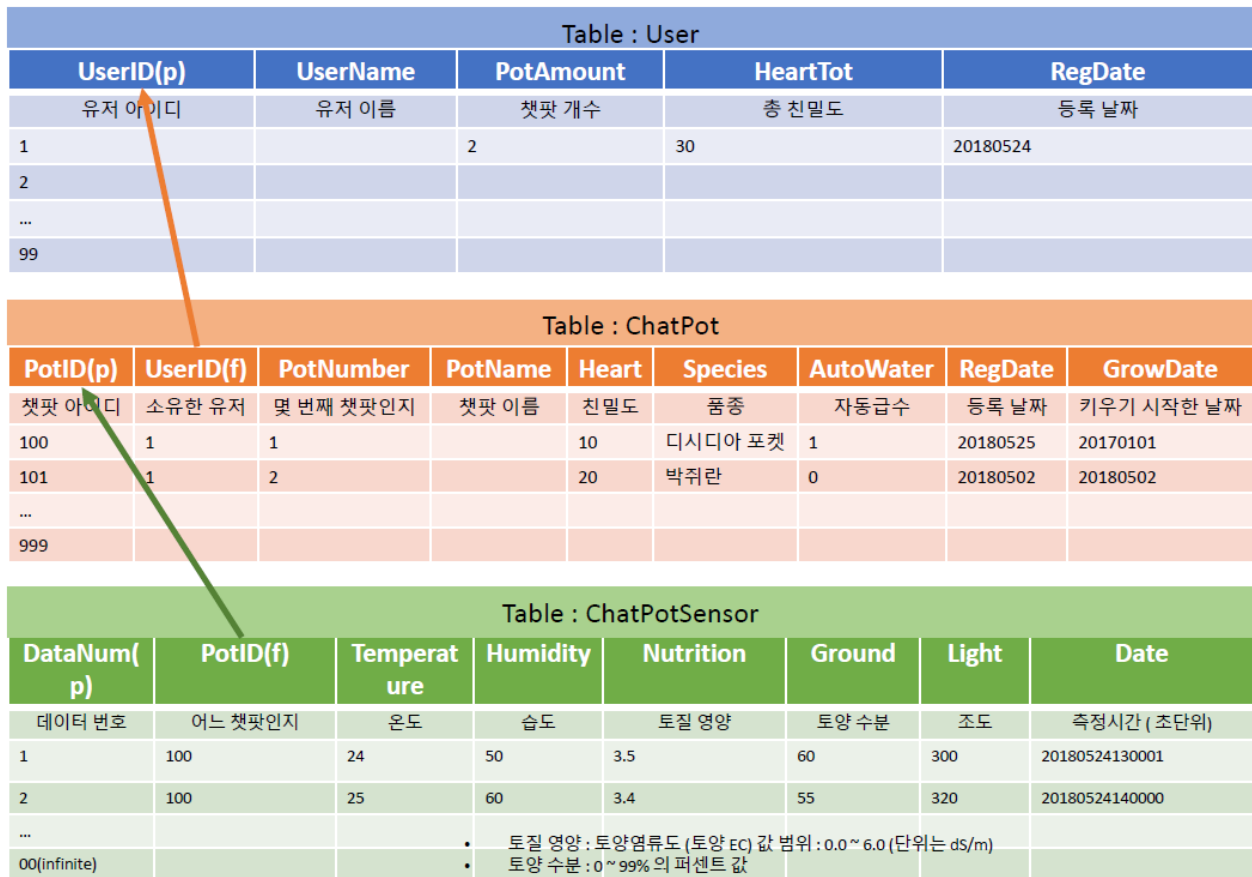
userService.java : 서버로부터 받은 json 타입 명령어를 분할 / 챗봇 정보값은 class

potIDInfoClass 에 담아 ArrayList potlist_potIDInfo 에 담음 / 챗봇 센서값은

class potSensorDataClass 에 담아 ArrayList potsensorlist_potSensorData 에 담음

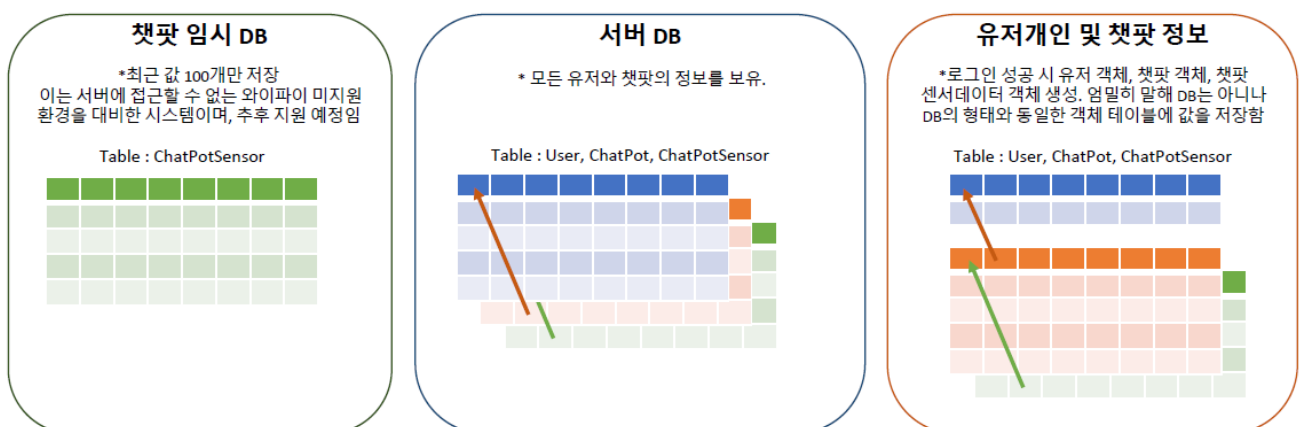
3.7. 데이터베이스 관계도 / 테이블 설계

3.7.1 서버 데이터베이스 설계도



[그림 3.7.1. 서버 데이터베이스 설계도]

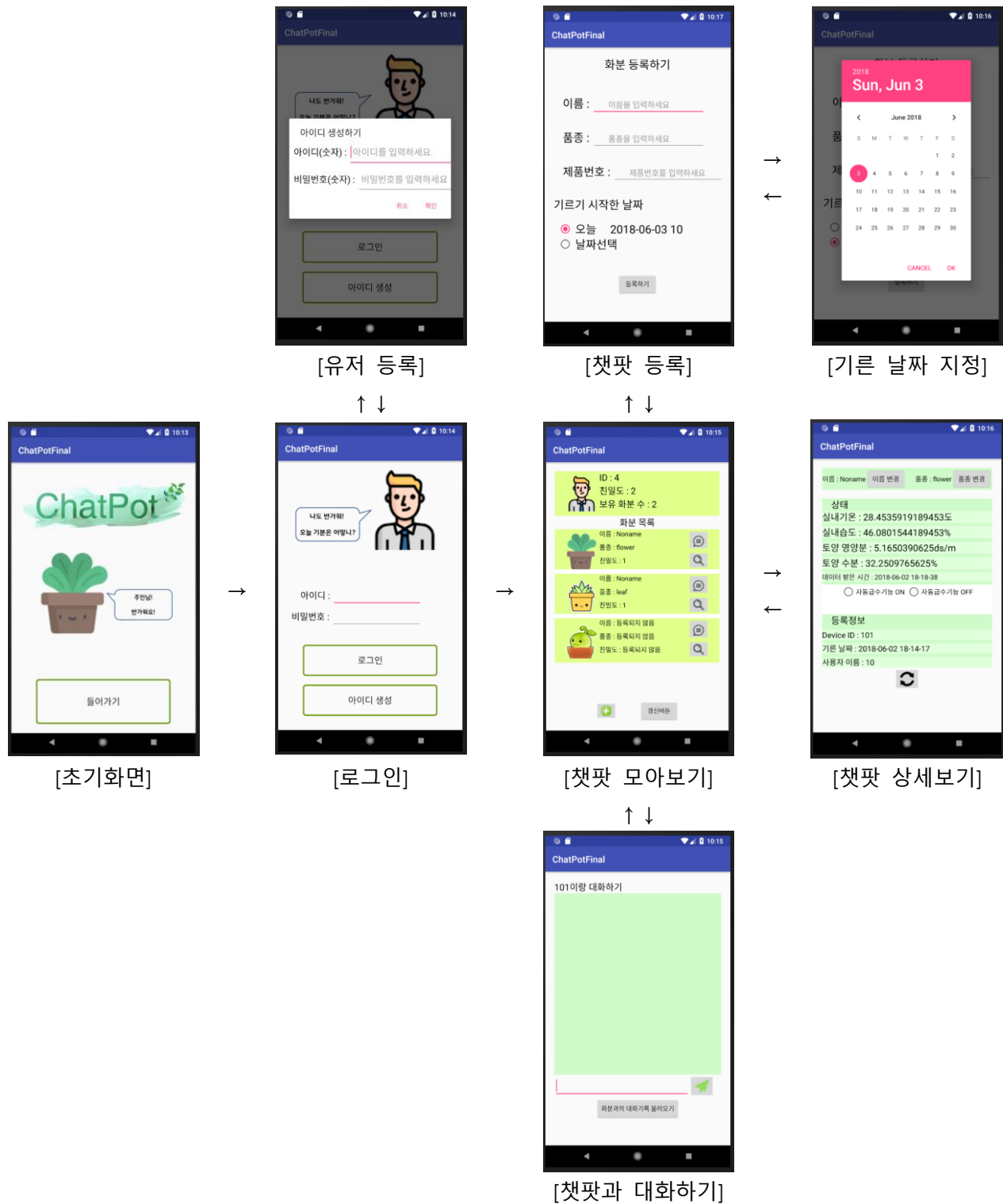
3.7.2 플랫폼별 데이터베이스 사용 다이어그램



[그림 3.7.2. 플랫폼별 데이터베이스 사용]

3.8 화면 설계

3.8.1 화면 전체 구성도



[그림 3.8.1 화면 전체 구성도]

3.8.2. 상세 화면 구성

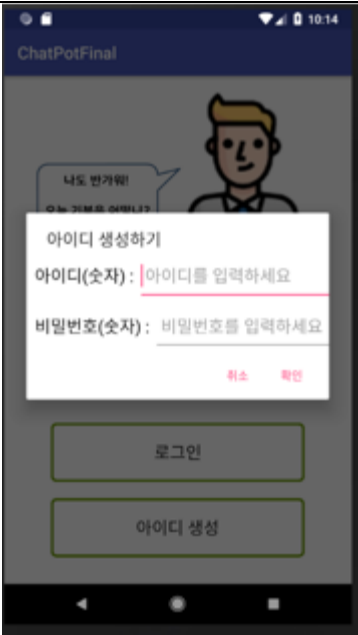
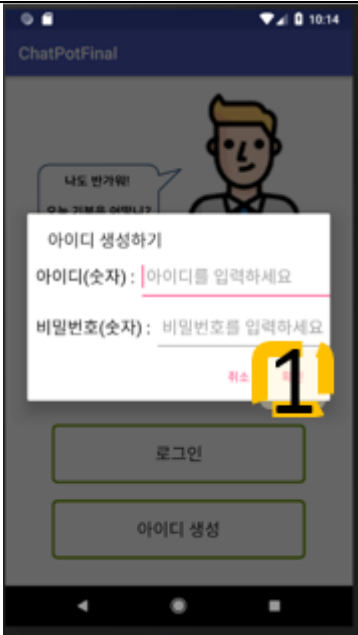
3.8.2.1 초기화면

실제 화면	상호작용이 가능한 부분
	 <p>1. 버튼. 소켓 생성 후 [로그인] 화면으로 진입</p>

3.8.2.2 로그인

실제 화면	상호작용이 가능한 부분
	 <p>1. 입력받은 필드값으로 로그인 시도. 성공 시 [챗팟 모아보기] 진입 2. [유저 등록] 화면 진입</p>

3.8.2.3 유저 등록

실제 화면	상호작용이 가능한 부분
	

1. 입력받은 필드값으로 유저 등록 시도

3.8.2.4 (로그인 성공 후) 챗팟 모아보기


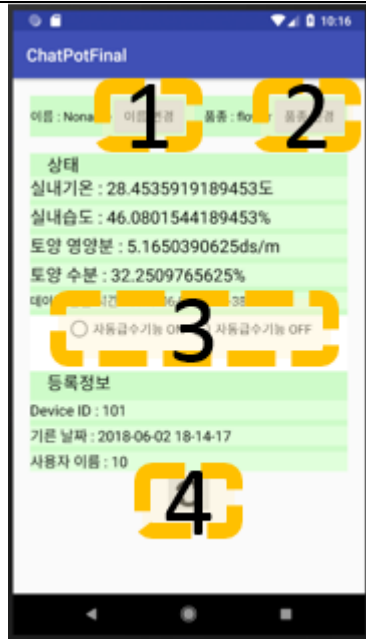
실제 화면	상호작용이 가능한 부분
	

1. [챗팟과 대화하기] 진입
2. [챗팟 상세보기] 진입
3. [챗팟 등록] 진입
4. 서버에서 최신 데이터를 수신받아 UI에 표시

3.8.2.5 챗봇과 대화하기

실제 화면	상호작용이 가능한 부분
	 <ol style="list-style-type: none"> 1. 채팅 입력필드의 값을 서버로 송신, 회신을 받으면 UI에 표시 2. 객체에서 지난 대화 내역 불러와 UI에 표시

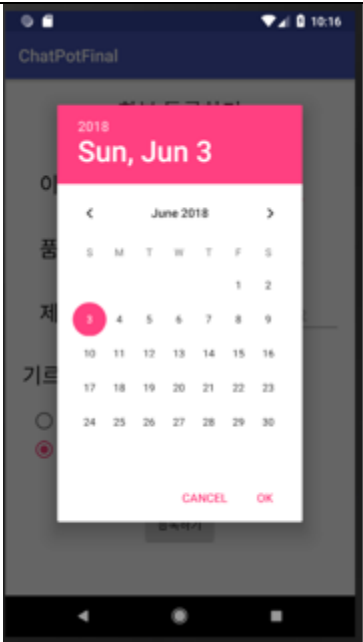
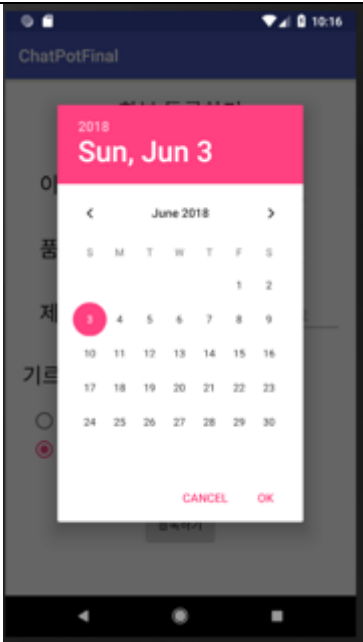
3.8.2.6 챗봇 상세보기

실제 화면	상호작용이 가능한 부분
	 <ol style="list-style-type: none"> 1. 챗봇의 이름 변경 (다이얼로그 생성 후 필드값 입력받음) 2. 챗봇의 품종 변경 (다이얼로그 생성 후 필드값 입력받음) 3. 챗봇의 자동급수기능을 켜거나 끄 4. 챗봇으로 최신 데이터 갱신 요청 후 회신된 내용 UI 표시

3.8.2.7 챗팟 등록하기

실제 화면	상호작용이 가능한 부분
	 <ol style="list-style-type: none"> 1. '오늘' 선택 시 당일 날짜로 챗팟을 등록 기르기 시작한 날짜가 등록 날짜와 다를 경우 실제 기르기 시작한 날짜를 고를 수 있는 [기른 날짜 지정] 화면으로 진입 2. 필드값을 입력받아 서버에 챗팟 등록 요청

3.8.2.8 기른 날짜 지정

실제 화면	상호작용이 가능한 부분
	 <p>* 날짜를 고른 후 OK 입력 시 이전 화면으로 해당 날짜 값을 리턴</p>

4. 기대 효과 및 향후 개선 사항

4.1 기대 효과



[그림 4.1 기대 효과]

1) 새로운 대안으로써의 첫팟

- 플랜테리어에 대한 범지구적 관심도가 증가하고 있다. 이는 국내도 예외가 아니며 이에 대한 지표는 각종 대형 마트에서의 홈가드닝 용품 판매 증가율에서 증명되고 있다. 첫팟은 급격하고도 꾸준히 성장하는 플랜테리어 시장의 새로운 대안으로써 부상할 것이다.

2) 동반자로서의 첫팟

- 가파르게 증가하는 1인 가구는 전체 가구의 27%까지 올라섰으며, 1인 가구의 36%는 20~30대로 이루어져 있다. 화분을 키우고 싶어도 반려식물을 돌보기 어려운 1인 가구의 주요 구성원이자 기술에 익숙한 20~30대에게 첫팟은 최적의 반려 식물 동반자가 될 것이다.

3) 사용자에게 따라 변해가는 첫팟

- 대화하는 3세대 스마트 화분 첫팟은 채팅을 통해 현재 실내의 기온과 습도, 그리고 외부의 날씨까지도 제공한다. 이러한 사용자 맞춤형 정보 제공은 반려식물의 정보와 연계되어 '나만의 재배일지' 등 지속적인 서비스 업데이트를 통해 이루어질 예정이다.

4.2 향후 개선 사항

4.2.1 서비스 측면에서의 개선 사항

1) 반려식물 품종에 따른 최적의 재배 환경 정보 제공

- 사용자가 챗팟 단말에 등록한 품종에 따라 방대한 식물 도감 DB에 따라 식물 키우기 Tip을 제공할 예정이다.

2) 반려식물 각각에 대해 '사진으로 기록하는 재배일지' 기능

- 현재 경쟁사인 엔씨의 플랜티는 식물의 재배 활동을 기록하는 모바일 앱 라이프(Life)를 서비스고 있으며 2014년 서비스 제공 10개월 만에 월간 액티브 유저수 1500만 명을 돌파하였다. 챗팟의 경우 재배일지를 작성하는 것에서 그치는 것이 아니라 자주 관찰하고 사진을 찍어줄 수록 친밀도가 올라가 식물의 응답이 변하는 채팅 시스템 도입, 이용자들이 먼저 찾는 서비스를 제공할 예정이다.

3) 친밀도 마일리지에 따른 앱 내 사용 아이템 상점

- 친밀도는 사용자가 꾸준히 반려식물을 관찰하고 재배일지를 기록하고 채팅을 통해 지속적인 관심을 보여야만 올라가는 사용자만의 자부심이다. 따라서 이를 통해 채팅으로만 식물이 반응하는 것이 아니라 해당 유저가 얼마나 식물 키우기에 관심이 있고 사랑하는 지를 시각적으로 보여주는 칭호와 휘장 등을 상점을 통해 친밀도 마일리지로 구매 가능하게 할 예정이다.

4) 다수의 사용자가 정보를 교류할 수 있는 커뮤니티 개설

- 같은 관심사를 가진 사람들과 교류하고 싶은 것은 당연한 욕구이지만 식물의 경우 위치가 고정되어 있어 온라인 커뮤니티를 통해 교류와 정보 나눔이 쉽지 않다. 이에 따라 챗팟 서비스에서는 1) 지역 기반 커뮤니티 2)품종 기반 커뮤니티를 동시에 제공하여 이용토록 할 예정이다. 친밀도 마일리지를 통해 구입한 칭호와 휘장 등으로 자신이 얼마나 식물에 관심이 있는지 자랑할 수도 있으며 서로의 재배일지를 보며 해당 품종의 재배 정보를 자유롭게 교류할 수 있는 커뮤니티 서비스가 업데이트 예정이다.

5) 더욱 스마트한 채팅엔진과 보이스 엔진의 탑재

- 인공지능 챗봇은 현 ICT산업의 가장 뜨거운 감자 중 하나이다. 따라서 구글, 아마존과 삼성을 비롯한 국내외 유수의 기업들이 이를 위한 다방면의 자연어 처리에 힘을 기울이고 있는 형국이며 자연어 처리 기술은 향후 몇 년간 비약적으로 발달할 것으로 전망된다.

챗팟이 실제 출시되어 서비스를 지속하고 사용자들의 이용 성향과 식물 재배에 대한 데이터베이스를 늘리는 동안 우리의 채팅 엔진을 더욱 똑똑하게 만들어 줄 자연어 처리 전문업체

와의 서비스·기술 협력 계약을 통해 더욱 스마트한 채팅엔진과 챗봇에서 직접 사용자에게 말을 걸 수 있도록 보이스 엔진이 탑재되어, 완전히 새로운 반려식물 재배의 장을 열기를 기대한다.

4.2.2 시스템 측면에서의 개선 사항

1) 와이파이 미지원 환경을 위한 블루투스 지원

- 챗봇이 와이파이 망에 연결되어 있지 않다면 휴대전화 단말기와 직접 블루투스로 통신하여 별도의 와이파이 망 없이도 챗봇 관리 및 재배일지 기록이 가능하도록 지원할 예정이다.

2) 실내 온습도를 비롯한 공기질 등을 감지하는 종합 센서 허브로써의 챗봇

- 반려식물이 잘 자라도록 하기 위해 필요한 센서 데이터들은 다른 곳에서도 유용하게 사용될 수 있다. 예를 들면 최근 큰 이슈가 되고 있는 미세먼지와 공기질에 대한 문제가 있다. 추후 출시 될 챗봇 2세대에는 공기질 감지 센서를 장착, '아마존 에코' 등 스마트 홈 허브 디바이스와의 연계를 통해 공기청정기 가동 등의 명령을 내리도록 할 수 있다.

3) 화분 형태가 아닌 스틱 형태의 챗봇 Lite 개발

- 화분 모양인 챗봇은 크기가 정해져 있으므로 실내용 작물에는 적합하지만 흙의 양이 많이 필요한 큰 식물에 대해서는 서비스를 제공하기 힘들다. 따라서 핵심 센서만을 응축하여 스틱 형태의 챗봇 Lite를 개발, 기존에 기르고 있는 크고 무거운 화분을 교체할 필요가 없도록 하면서도 원활한 재배에 꼭 필요한 정보는 제공한다. 뿐만 아니라 챗봇 앱 서비스와 연동되므로 재배일지를 비롯한 각종 서비스는 동일하게 이용 가능하다.

4) 스마트 화분을 넘어서 스마트 실내 경작으로 - 자동화된 소규모 실내 경작

- 유기농 채소에 대한 관심이 급증한 것은 불과 몇 년 되지 않았지만, 미세먼지나 공기질에 대한 문제 또한 대두되면서 깨끗한 채소에 대한 대중들의 관심은 지속되고 있다. 실내 경작은 위생 뿐만 아니라 생산성에서도 전통적인 방식에 비해 압도적인 생산량을 보여주고 있다. 세계 최대의 실내 농장인 일본의 Shimamura 농장은 25,000 평방 피트에서 하루에 10,000 개의 상추를 채취하고 있으며, 이를 전통적인 방식으로 생산할 경우 100 배의 경작 규모가 필요하다. 요구 에너지의 측면에서도 40% 적은 전력 소모, 80% 적은 비료와 99% 적은 수분이 필요하다. 이러한 일본의 실험에서 증명된 실내 경작의 가치는 세계적으로 인정받아 동일한 시스템의 실내 농장이 홍콩, 몽골, 러시아와 중국 본토로까지 확장 건설되고 있다.

이와 같은 사회적 관심 속에서 대규모의 기업형 실내 농장이 성공하는 동안 우리는 더 작은 단위를 위한 실내 경작에 주목한다. 국내의 인구 1000 명당 숙박음식점 업체 수는 13.5 개로, 일본의 2 배 이상, 미국과 영국의 5 배 이상의 수치이다. 이토록 높은 음식점 비율에서 재료의 신선도라는 경쟁력과 도매가보다 저렴한 직접 생산의 단가, 최종적으로 생산의 A to

Z 까지 관여하지 않아도 되는 자동화된 소규모 실내 경작의 이점은 분명 존재하며 경제적 이점 뿐만 아니라 직접 기른 깨끗한 채소가 제공된다는 소비자들의 믿음 또한 살 수 있을 것이다.

규모를 조금 더 확장한다면 자체 사원 식당을 구비하고 있는 대규모 기업들과 호텔 대상으로도 사업의 확장 가능성이 있다. 특히 호텔의 경우 최고급 식사를 제공받기 위해 높은 가격을 지불하는 손님들이 많기 때문에 직접 재배한 깨끗한 채소를 제공한다는 호텔 레스토랑의 이미지 제고와 더불어 유기농 채소를 구매하는 것보다 싸게 재배할 수 있는 방식을 선택할 것이라는 측면에서 사업성이 있다고 판단된다.

이를 위해서는 우선 실내에서 경작되는 식물이 어떤 환경에서 최적의 상태로 자라는 지에 대한 충분한 데이터베이스를 보유하고 있어야 할 것이며 소프트웨어의 개발 뿐만 아니라 실내 경작에 적합한 하드웨어에 대한 연구·개발 또한 이루어져야 할 것이다. 따라서 챗팻 서비스가 본 궤도에 오르기 시작하면 기존 챗팻의 서비스 측면 강화 뿐만 아니라 실내 경작 전반에 대한 트렌드 리서치 및 자체 하드웨어 개발에 착수하는 등 스마트 화분을 넘어선 새로운 스마트 실내 경작의 한 축으로 자리매김할 수 있도록 사업의 규모를 확장할 예정이다.

5. 개발 후기



성명	후기
김신재 (채팅엔진 및 리서치)	<p>프로젝트 시작당시 선뜻 채팅엔진을 맡겠다고 했지만, 채팅엔진이 이렇게 어렵고 복잡한 개념인 줄 몰랐다. Intent, Entity 등 필수개념을 공부해야 했고 문장의 구조도 분석해야만 했다.</p> <p>단기간에 범용적인 채팅엔진을 개발하기에는 무리가 있다고 판단해서 API 를 적용시키고자 했지만, 프로젝트에 적용할 만한 API 를 찾지 못했다. SVM 이나 임의숲같은 분류기를 사용할 생각도 해보았지만 학습시킬 데이터베이스 문제도 있었고, 상시 변화하는 센서값을 알려주는 대화도 있어야 했기에 해당 개념을 제외시키기도 했다. 그리고 영어 대화가 이루어져야 했기 때문에 언어적 어려움도 있었다.</p> <p>부족한 코딩실력으로 간단한 채팅엔진 알고리즘을 구현했다. 개인적으로 어디 내놓기는 완성도가 낮은 채팅엔진이라 생각한다. 하지만 이번 프로젝트를 통해 채팅엔진에 대해 자세히 알게 되었고, 자체적인 분류알고리즘 개발과 데이터 관리, 복잡한 알고리즘의 메소드화와 캡슐화 작업이 예전에 해왔던 프로젝트와는 다른 성격의 프로젝트였기 때문에 값진 경험이라 생각한다. 그리고 나보다 더 고생한 팀원들에게 고생 많았다고 전하고 싶다.</p>

<p>이해주 (서버)</p>	<p>지난 2 주간 팀원들과 함께 머리를 맞대고 아이디어를 구상하고, 코드를 짜고, 그리고 결과물을 만들어 내는 데까지 정말 많은 경험을 얻을 수 있었다. 수료 과정 약 6 개월간 배운 것을 토대로, 우리만의 힘으로 프로젝트를 완수하는 것은 정말로 특별한 경험이었다. 과정을 이수하기전에는 서버의 구동 원리조차 몰랐었던 나였지만, 이번 프로젝트를 통하여 프로토콜과 서버의 역할과 필요성을 몸소 체득할 수 있었고, 안드로이드와 PC, 그리고 라즈베리파이 간의 통신이 되고 우리가 원하고자 하던 기능이 동작하는 것을 보면서 많은 감정이 교차하였다.</p> <p>언뜻 보면 간단해 보이지만, 저렇게 단순해 보이는 통신마저 수많은 어려움이 있었기 때문이다. 프로토콜을 구성하는 것에서부터, 송/수신 데이터 크기로 인한 통신 문제, 명령어 두개가 합쳐져서 들어오는 문제 등 한 발짝 나아갈 때마다 우리를 가로막는 문제점과 싸워야 했다. 그러한 문제를 해결하기 위해서 팀원들과 고민을 하며 해결책을 찾아 나가면서 프로젝트를 진행함에 있어서 팀원과의 소통이 얼마나 중요한지 깨달았다. 또한 문제가 생기거나 기능을 추가할 때마다 임시방편적인 코드를 넣어가며 일을 진행하다 보니, 후반부에 가니 코드가 너무 복잡하고 길어지는 문제가 발생하였었다. 코드를 구성하는데 있어서, 큰 그림을 먼저 보고 그에 따라 확장성 있고 유연한 코드를 구성해야 함을 절실하게 느꼈고, 미숙한 코딩 경험이 절실하게 드러났음을 알 수 있었다.</p> <p>처음에 우리가 계획했던 목표를 100% 달성하진 못하였다. 포기하였던 기능도 있고, 시간이 없어서 건드려 보지도 못했던 기능도 있다. 하지만 컴퓨터 전공이 한 명도 없는 우리 팀원들이 모두 코드를 구성하고, 이를 연동하여 소기의 목적을 달성한 것이 개인적으로 자랑스럽고 프로젝트를 진행하며 많은 것을 깨닫고 배우게 해 줄 수 있었던 팀원들에게 고맙고 미안한 감정이 든다. 고생한 만큼 팀원들 모두에게 앞으로의 인생에 좋은 경험이자 좋은 추억이 되길 빌며 소감을 마친다.</p>
<p>전승용 (챗봇 단말)</p>	<p>6 개월 간의 과정을 통해서 가장 많이 다뤘던 언어인 Python 언어를 사용하여 프로젝트의 단말을 구현하였다. 그 과정 동안에 배웠던 센서를 다루는 법과 socket 통신을 통해서 이 단말을 구성하였다. 이 프로젝트 과정에서 팀 프로젝트이다 보니 서로 간의 프로토콜, 데이터 수신 범위 등 신경 써야 할 부분이 많았다. 이 부분을 신경 쓰지 못해서 예기치 못하게 시간을 많이 썼었다. 이 부분이 미숙하여 프로토콜과 데이터 수신 범위 등을 많이 수정하면서 안드로이드, 서버 파트의 팀원과 소통과 협의를 하였다. 이렇게 소스 코드를 수정하면서 팀원과의 소통이 중요하다는 것을 다시 한번 깨달았다.</p> <p>또한 여러 센서들을 사용했는데, 미숙한 코딩으로 여러가지가 발생하였었다. 특히, PIR 센서를 사용할 때 이러한 문제가 두드러지게 발생하였다. 이 센서를 포기할까도 많이 생각하였었지만, 끊임없이 코딩을 수정한 결과 구현에</p>

	<p>성공하였다. 이 과정에서 포기하지 않은 자신에게 뿌듯함과 성취감을 얻었다. 또한 단말을 일부 구성한 뒤에 서버와의 연결을 시도하고, 그 뒤에 안드로이드와의 연결을 시도하였다. 안드로이드를 통해서 단말을 제어하고 데이터를 얻어오는 것을 성공했을 때, 나 뿐만 아니라 팀원 모두 큰 성취감을 얻을 수 있었다.</p> <p>처음에 계획했던 프로젝트를 한번 처음부터 다시 하고, 기능들을 수정하면서 시간을 소비했다는 점이 우리가 미숙했다지만 조금 아쉽다. 다음 프로젝트에는 이러한 시간을 줄이면서 더욱 완성된 작품을 만들 수 있을 것이라고 확신이 든다.</p> <p>이 프로젝트를 통해서 팀원으로써 책임감과 협업 소통을 배울 수 있었고, 개인적으로는 프로그래밍 역량이 한발 앞으로 나아갔다고 생각한다. 마지막으로 프로젝트를 하면서 미숙했던 부분이 많아서 팀원들에게 도움을 많이 요청했었다. 팀원들은 자신들의 일도 바쁨에도 불구하고 많은 도움을 주어서 내가 맡은 역할을 끝낼 수 있었다고 생각한다. 팀원들에게 모두 고맙고 나와 팀원 모두에게 뿌듯하면서도 힘들었지만 즐거운 추억이 될 수 있을 것이라고 생각한다.</p>
<p>주상훈 (안드로이드)</p>	<p>2017년 12월 경 코딩의 세계에 문외한이었던 나는 약 5개월이 넘는 기간 동안 밀도 높은 코딩 세계에 뛰어들었다. 이번 프로젝트는 제대로 된 팀프로젝트 경험이 없었던 나에게 귀중한 시간이었다. 5개월의 기간동안 배운 것을 응축하여 결과물을 만들어내는 것에 묘한 기대감이 가득했다.</p> <p>팀프로젝트는 강사님께서 결코 쉽지 않을 프로젝트라며 단언하셨던 것이 현실화되며 시작부터 난항을 겪었다. 좋은 아이디어라고 생각했던 제안은 번번히 장애물에 가로막혔다. 주말에 모여 새로운 아이디어를 짜기 위해 하루종일 머리를 맞대어 식물과 대화하는 '챗팻'이라는 프로젝트를 시작하게 되었다.</p> <p>포부 좋게 시작한 프로젝트는 희로애락의 정수였다. 특히 나는 안드로이드를 중점으로 맡게 되면서 더욱 그랬었다. 버벅거리면서 원하는 것을 구현할 때의 기쁨. 에러 하나 쉽게 찾기 어려운 구조의 안드로이드에게 느껴지는 환멸감. 팀원의 결과물들이 함께 구동되었을 때의 즐거움. 정말 말도 안되는 소켓통신의 어려움으로 머리 끝까지 오르는 분노. 다양한 감정이 뒤섞이며 누구를 탓할 것도, 나를 원망할 것도 아닌 것이 팀프로젝트라는 것을 느끼게 되었다.</p> <p>이번 프로젝트를 통해 많은 것을 배울 수 있었다. 정말 코딩을 나의 것으로 만들기 위해서는 프로젝트나 실전처럼 만들고 조립하고 구현해보는 것이 중요하다는 것. 이를 위해서는 기초적인 수업을 잘 듣는 것이 많은 도움이 된다는 것. 협업에 있어서는 파일 이름과 같은 사소한 것 하나까지라도 맞춰서 쌓아올려야 한다는 것. 이런 사소한 것에 대해 상대방에게 짜증을 내는 것이 아니라 이해하고 상대를 격려해줘야 한다는 것.</p>

	<p>2 주 남짓 넘는 프로젝트 기간에서 코딩의 세계라는 작은 머리와 함께 다독이며 나아갈 수 있어야 한다는 어린 심장을 얻었다. 비전공자라는 이름표를 떼어버리고 우리의 프로젝트인 챗팻을 기반으로 식물과 대화하듯 새로이 얻은 작은 머리와 어린 심장을 견고히 성장시키며 전공자 못지 않은 실력을 갖추고 싶다.</p> <p>2 주 동안 너무 고생했을 우리 팀에게 많은 것을 알려주고 배움을 얻게 해준 것에 깊은 감사를 표하고 싶다. 다들 고생했다!</p>
<p>황원준 (PM)</p>	<p>전체 계획에 있어서 조금의 틈이라도 있으면 시스템 전체가 흔들리는 것을 실감. 사용자 입장에서 바라본 전체 사용 시나리오, 그에 요구되는 핵심 기능과 이에 따른 시스템 시나리오 설계, 그리고 마지막으로 이를 뒷받침 할 치밀한 DB 와 프로토콜 설계가 모두 선행되어야만 납기 기일에 맞출 수 있는 원활한 프로젝트 진행이 가능하다는 점을 절감하게 되었음.</p> <p>뿐만 아니라 팀원 간의 소통이란 단순히 말로 설명하는 것도 아니고 '팀원 간의 합의된 형식에 따른 적절하고 간소화된 문서'의 작성에 있는 것도 아니며, 이를 '팀원 전체가 함께 작성하고, 읽고, 이해하고, 최종적으로 그 규칙에 따름'에 있음을 깨달음. 그리고 모든 프로젝트에서 공통적으로 반드시 필요한 문서들에 대해서는 PM 이 미리 형식을 완전히 정하여 프로젝트 착수 전 모두에게서 공감을 이끌어 내는 것이 중요함을 알게 되었음. 예를 들어 일반 문서작업과는 다르게 파일의 이름과 내부에서 사용되는 이름을 바꿀 경우 예기치 못한 에러를 발생시켜 시스템 자체의 결함을 초래할 수 있기 때문에 이를 개인의 재량에 맡기면 최종 릴리즈 시 무의미한 작업량이 늘어남과 더불어 감수하지 않아도 될 위험을 무릅쓰게 됨. 이 외에도 본인의 코드 구조화 및 파일과 클래스에 대한 구조도식화 양식도 포함됨. 프로젝트 구조의 도식화와 프로그램 목록화&문서화의 중요성에 대해 미리 인지하여 매일 마지막 30 분을 할애하여 작성해 달라고 요청했지만 세세한 양식을 공유하지 않아 결과적으로 개인마다 다른 문서 양식을 최종 보고서에 신게 되었음. 이처럼 모든 프로젝트에서 반드시 필요한 부분에 대해 관리자의 입장에서 바라볼 수 있는 기회를 얻게 되는 값진 경험을 함.</p> <p>프로젝트 자체에 대해 자평하자면 핵심 기능과 UI 의 측면에서 시간 관계상 설계에 미치지 못하는 아쉬운 점이 다소 있고 결과물이 예상 목표치에 비해 밀돌지만, PM 으로서는 팀원 개개인의 능력을 한계치까지 발휘시킨 성공적인 프로젝트였음.</p>

6. 강사 의견

평 가 요 소	배점	평
아이디어 : 유사한 서비스 존재 유무 및 체계성	/20	
2. 개발 : 실제 구현 정도 및 배포 유무, 코드의 무결성 및 난이도, 현업적용도, 실무기술 반영정도	/30	
3. PJT 수행력 : 일정관리 및 역할분담, 목표 일정 달성도, 팀내 참여도 등	/30	
4. 준비도 : 프리젠테이션 및 프로젝트 준비 정도	/20	
계	/100	