

설계 보고서

-Planty(플랜티): LLM과 IoT를 활용한
개인 맞춤형 식물 관리 솔루션-

과목명: 실무중심산학협력프로젝트 1

교수님: 박소현 교수님

팀 명: BIoTy

팀 원: 구선주(32220207)

김민지(32200588),

민유진(32221598)

최예림(32224684)

목차

1. 프로젝트 개요	1
연구 주제	1
연구 개요	2
기존 연구와의 차별점	3
연구 목표	4
기대효과	5
2. 프로젝트 개념 설계	6
정보 구조도	6
어플리케이션	6
IoT	13
LLM	15
3. 프로젝트 개발을 위한 환경 구축	18
협업도구	18
개발환경	19
4. 구현 방법	22
IoT	22
어플리케이션	23
LLM	27
5. 개발 진행 상황	30
역할 분담	30
개발 일정	31
개발 진행 상황	31
추후 계획	34
참고문헌	36

연구 필요성

반려식물에 대한 수요가 증가하면서, 누구나 쉽게 활용할 수 있는 효율적인 식물 관리 도구의 필요성이 커지고 있다. 그러나 식물 관리에 대한 경험과 지식이 부족한 초보자들에게는 식물 상태의 이상을 인지하거나 적절한 조치를 취하는 것 자체가 쉽지 않은 일이다.

따라서, 누구나 직관적으로 사용할 수 있으면서도 고도화된 기술이 접목된 식물 관리 솔루션의 필요성이 대두된다. 이에 따라 본 프로젝트의 플랜티는 단순히 정보를 보여주는 수준을 넘어, 센서 기반의 자동화된 데이터 수집 및 원격 제어 기능, 그리고 대화형 AI 기술을 통한 실시간 맞춤형 조언을 결합한 형태의 식물 관리 방식을 제시한다.

또한, 기술적 편의성뿐만 아니라 사용자와 식물 간의 감성적 연결을 중심 가치로 둔다. 사용자의 반려식물 성격을 반영한 페르소나 기반 식물 에이전트와 대화를 나눌 수 있도록 함으로써, 단순한 관리가 아닌 정서적 교감이 가능하다. 이는 기존의 어플리케이션들과 확연하게 구분되는 정서적 반려 경험을 제공하는 핵심 기능이며, 기술과 자연, 사람 간의 새로운 연결 방식을 제시한다.

연구 개요

PlanTy (플랜티)

Planty는 Plant와 Buddy의 합성어로 당신의 반려 식물과 함께 하는 공간이라는 의미를 가지며, 사용자가 식물과 대화를 나누고, 감정을 공유하며, 건강하게 키울 수 있도록 돕는 스마트 반려 식물 관리 어플리케이션이다. 식물의 상태를 감성적인 표현으로 전달하고, 실시간 IoT 모니터링과 AI 기반 챗봇을 통해 보다 친근하고 풍성한 가드닝 경험을 제공한다. 식물의 상태를 실시간으로 모니터링하고, 맞춤형 관리 솔루션을 제공하여 편리한 식물 관리를 가능하게 한다.

1. IoT 통합 자동화

센서 기반으로 식물 데이터를 수집하고, 와이파이 기반 통신을 통해 원격에서 식물 상태를 실시간으로 확인하고 자동 제어가 가능하다.

2. LLM 기반 챗봇

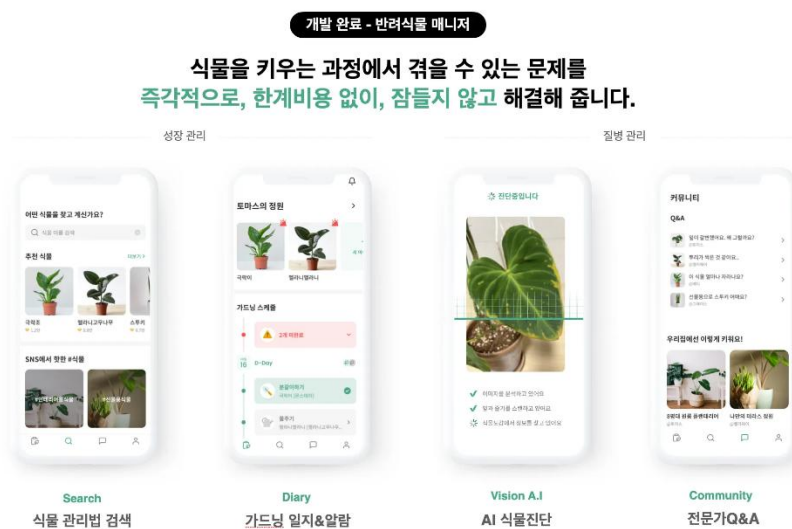
기존의 일방향 알림이나 텍스트 정보 제공 방식과 달리, 센서 데이터를 기반으로 즉각적이고 상황에 맞는 대화형 응답을 제공하며, 식물의 상태에 따른 사용자 맞춤형 피드백을 제공한다.

3. 에이전트 기반 정서적 교감

식물의 성격을 반영한 에이전트를 부여하여 사용자가 마치 식물과 대화하듯 상호작용을 할 수 있는 정서적 유대감 중심의 인터페이스를 구현한다.

기존 연구와의 차별점

기존 연구



[그림 2. 어플리케이션 그루우 groo(출처: 구글 플레이스토어)]

현재 시장에 출시된 식물 관리 어플리케이션들은 식물 정보 제공, 일정 알림, 커뮤니티 기능 등 기초적인 관리의 편의성 향상에 초점을 맞추고 있다. 대표적인 어플리케이션으로 플랜톡 Plantalk, 그루우 groo, 플랜트그램 plantgram, 풀박 fuleaf이 존재한다.

플랜톡의 경우 IoT 기기와 연동하여 센서 수치를 모니터링하고 환경 리포트를 제공하며, 그루우의 경우 스마트 스케줄링과 이미지 기반 식물 진단 기능을 포함한다. 플랜트그램과 풀박 역시 다양한 식물 정보와 사용자 커뮤니티를 제공하며, 알림 기능이나 추천 콘텐츠로 사용자의 관리를 지원하고 있다.

하지만 이들 서비스는 공통적으로 식물을 관리 대상으로만 다루고 있으며, 사용자와 식물 간의 정서적 관계 형성이나 쌍방향 소통에 대한 기능은 미흡한 상태이다.

기존 어플리케이션과의 차별점

PlanTy는 단순한 관리 기능을 넘어서, 식물을 소통 가능한 반려 존재로 인식하고 있는 점이 기존 어플리케이션들과의 가장 큰 차별점이다.

IoT 기반 식물 관리 어플리케이션과 차별화된 기능을 제공한다. 기존 어플리케이션(예: 플랜톡)에서 사용하는 IoT 기기는 블루투스 기반 통신을 활용하기 때문에 근거리에서만 정보 수신이 가능하다는 한계가 존재한다. 반면, PlanTy는 와이파이 기반 통신 방식을 채택하여 장시간 외출하거나 식물과 물리적으로 멀리 떨어져 있는 경우에도 실시간으로 식물 상태 정보를 받아볼 수 있으며, 원격지에서도 식물 관리를 지속할 수 있다. 또한, 워터펌프, 팬, LED 등 다양한 식물 관리 모듈을 추가로 연결하여 보다 고도화된 식물 관리 기능을 제공함으로써 사용자의 관리 편의성과 자동화를 동시에 향상시킨다.

LLM 기반 기능 측면에서도 PlanTy는 기존의 시나리오에 따른 고정 답변을 제공하는 방식에서 벗어나, 실시간 응답이 가능한 식물 특화 챗봇 서비스를 구현하였다. 이를 위해 검색 기반 생성(RAG) 기법을 활용하여 식물에 특화된 정보를 바탕으로 보다 정확하고 풍부한 답변을 제공하며, 멀티 에이전트 시스템을 적용하여 각 식물의 성격을 반영한 차별화된 상호작용을 구현하였다. 아울러, 매일 식물의 상태를 분석하여 식물의 기분을 한 줄로 요약해 제공함으로써, 사용자와 식물 간의 정서적 유대감을 강화하는 기능 또한 지원한다.

연구 목표

본 연구에서는 각 식물의 상태 및 관리 요구에 따라 실시간으로 사용자에게 알림을 제공하고, 적절한 조치를 안내함으로써 개인 맞춤형 알림 및 조언 서비스를 제공하고자 한다. 이를 위해 IoT 센서와 이미지 분석 모델을 활용하여 사용자가 직접 식물의 상태를 확인하지 않더라도 자동으로 진단 및 관리가 가능하도록 하여, 스마트 기술 기반의 관리 효율화를 추구한다. 또한, 대규모 언어 모델(LLM)과 검색 기반 생성(RAG) 기법을 적용하여 다양한 식물에 대한 정보와 구체적인 관리 방법을 사용자 맞춤형으로 제공함으로써, 사용자의 관리 편의성과 전문성을 동시에 향상시키고자 한다. 아울러, 본 시스템은 식물 관리 경험이 없는 초보자부터 숙련된 전문가에 이르기까지 다양한 수준의 사용자가 모두 활용할 수 있는 범용적 솔루션을 지향하여, 폭넓은 사용자층을 위한 통합적 도구를 제공하는 것을 목표로 한다.

기대효과

1. 초보자도 쉽게 사용할 수 있는 스마트 식물 관리 솔루션 제공

사용자 친화적인 UI/UX와 챗봇 기반 맞춤형 가이드를 통해, 식물에 대한 지식이 없는 초보자도 손쉽게 식물을 등록하고 관리가 가능하다.

2. IoT 기술을 활용한 식물 관리 자동화 및 최적화

센서 데이터를 기반으로 물주기, 조도 조절 등의 작업을 자동화하여, 사용자 개입 없이도 식물의 건강을 안정적으로 유지할 수 있다.

3. AI 기반 챗봇을 통한 사용자 편의성 향상

LLM 기반 챗봇이 실시간으로 식물 상태에 대한 피드백을 제공하여, 사용자의 불확실성을 줄이고 전반적인 관리 효율을 높인다.

4. 스마트 홈가드닝 트렌드에 맞춘 혁신적인 서비스 제공

1인 가구 및 도시 거주자들의 실내 식물 관리 수요에 대응하며, 기술과 자연이 조화를 이루는 새로운 홈가드닝 경험을 제시한다.

2. 프로젝트 개념 설계

정보 구조도



[그림 3. 정보 구조도]

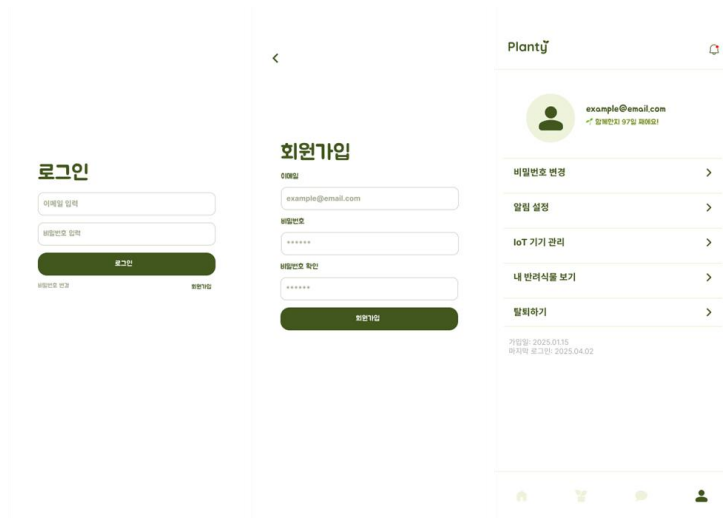
위의 도식은 어플리케이션의 주요 기능과 사용자 흐름을 체계적으로 시각화한 정보 구조도이다. 초기 단계에서는 회원가입, 로그인, 비밀번호 찾기와 같은 기본 인증 절차를 통해 사용자의 접근을 제어하며, 이후 홈 화면을 기점으로 다양한 서비스 기능에 자연스럽게 연결될 수 있도록 설계하였다.

홈 화면에서는 반려식물 등록 및 관리, 식물 도감 조회, 스마트 챗봇 사용, 개인 정보 관리 등 핵심 기능을 직관적으로 탐색할 수 있도록 구분하였다. 각 기능별로 세부 메뉴를 명확히 나누어 정보 접근성을 높였으며, 사용자가 필요한 기능을 최소한의 탐색 경로로 이용할 수 있도록 최적화하였다.

어플리케이션

회원 정보 관리

회원 관리 기능은 사용자의 계정 생성부터 정보 수정 및 탈퇴까지의 과정을 지원한다.

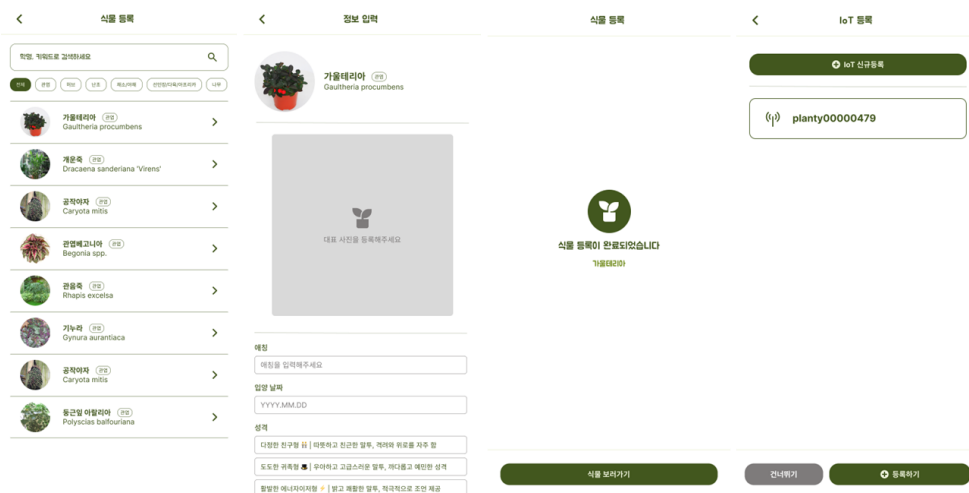


[그림 4. 회원 정보 관리]

앱을 실행하면 애플리케이션 로고와 함께 슬로건이 표시되어 서비스를 소개한다. 로그인 화면에서는 이메일과 비밀번호를 입력하여 기존 계정에 접속할 수 있으며, 계정이 없는 사용자는 회원가입 화면으로 이동하여 이메일과 비밀번호를 입력해 새로운 계정을 생성할 수 있다. 회원가입이 완료된 사용자는 마이페이지를 통해 계정 정보를 확인하고, 비밀번호 변경, 알림 설정, IoT 기기 관리, 반려식물 목록 조회, 탈퇴 기능을 이용할 수 있다.

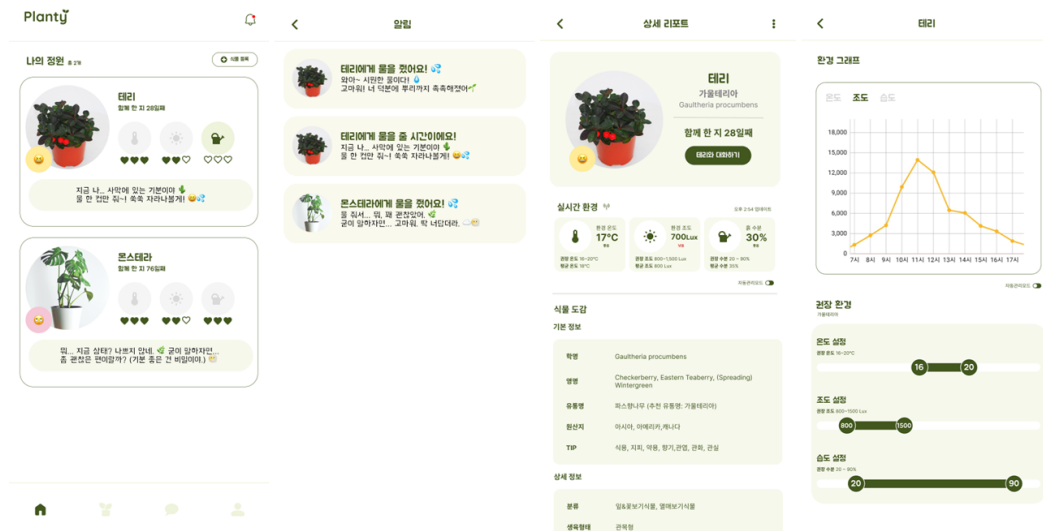
개인 식물 관리

사용자가 자신만의 반려식물을 등록하고, 상태를 모니터링하며 개개인의 반려식물을 관리할 수 있도록 지원한다.



[그림 5. 식물 및 IoT 등록]

식물 등록 화면에서는 다양한 식물 도감 리스트를 제공하며, 사용자는 원하는 식물을 검색하여 선택할 수 있다. 선택한 식물에 대해 사진 등록, 애칭 설정, 입양일 입력, 성격 선택 등을 통해 나만의 반려식물 정보를 입력할 수 있다. 식물 등록을 완료하면 IoT 기기 연결을 통해 식물의 환경 데이터를 실시간으로 모니터링할 수 있으며, 신규 기기 등록 또한 지원한다.

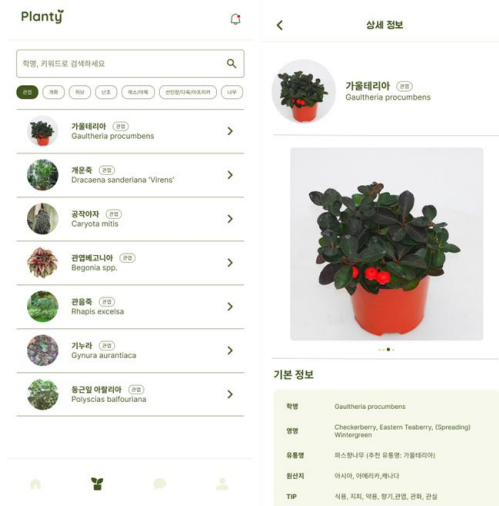


[그림 6. 개인 식물 관리]

나의 정원, 즉 홈 화면에서는 등록한 반려식물들의 상태를 한눈에 확인할 수 있다. 각 식물별로 필요한 액션 알림, 현재 상태와 상태 메시지 등이 표시되어 사용자가 식물의 건강 상태를 쉽게 파악할 수 있다. 홈 화면에서 제공하는 액션 버튼을 통해 물주기, 바람 조절, 팬 켜기 등의 기능을 직접 실행할 수 있으며, 해당 작업은 IoT 기기와 연동되어 즉시 반영된다. 상세 리포트 화면에서는 선택한 식물의 실시간 온도, 조도, 습도 데이터를 확인할 수 있으며, 환경 그래프를 통해 시간별 변화를 시각적으로 분석할 수 있다.

식물 도감

식물 도감 기능은 다양한 식물 정보를 제공하여 사용자가 반려식물을 선택하는 데 도움을 준다.

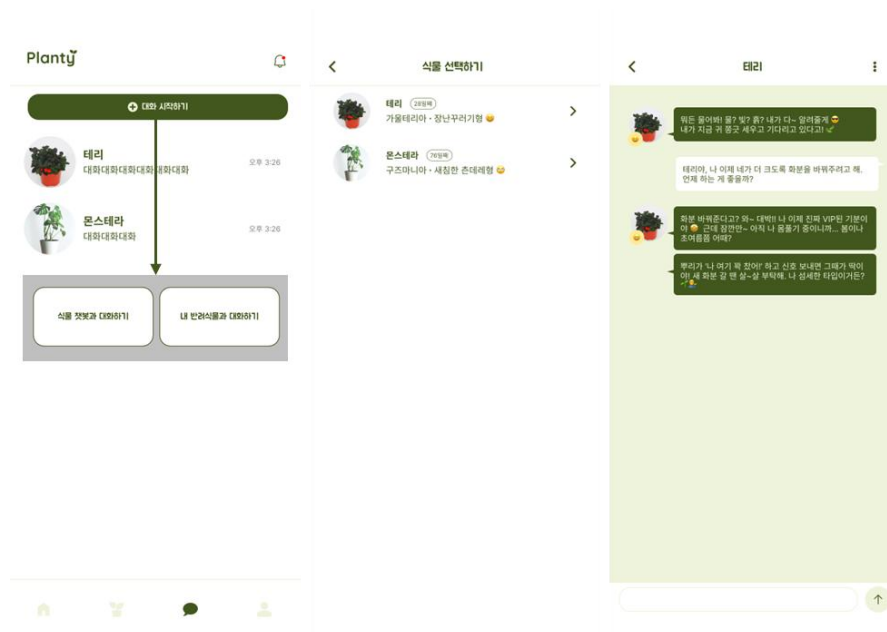


[그림 7. 식물 도감]

도감 목록 화면에서는 식물의 학명, 영문명, 이미지를 함께 표시하며, 사용자는 검색 기능과 필터를 통해 원하는 식물을 쉽게 찾을 수 있다. 식물 항목을 선택하면 상세 정보 화면으로 이동하며, 식물의 사진과 함께 학명, 영명, 주요 특성, 원산지, 관리 팁 등 구체적인 정보를 확인할 수 있다. 이를 통해 사용자는 식물의 특성을 사전에 이해하고, 자신에게 맞는 반려식물을 선택할 수 있다.

스마트챗봇

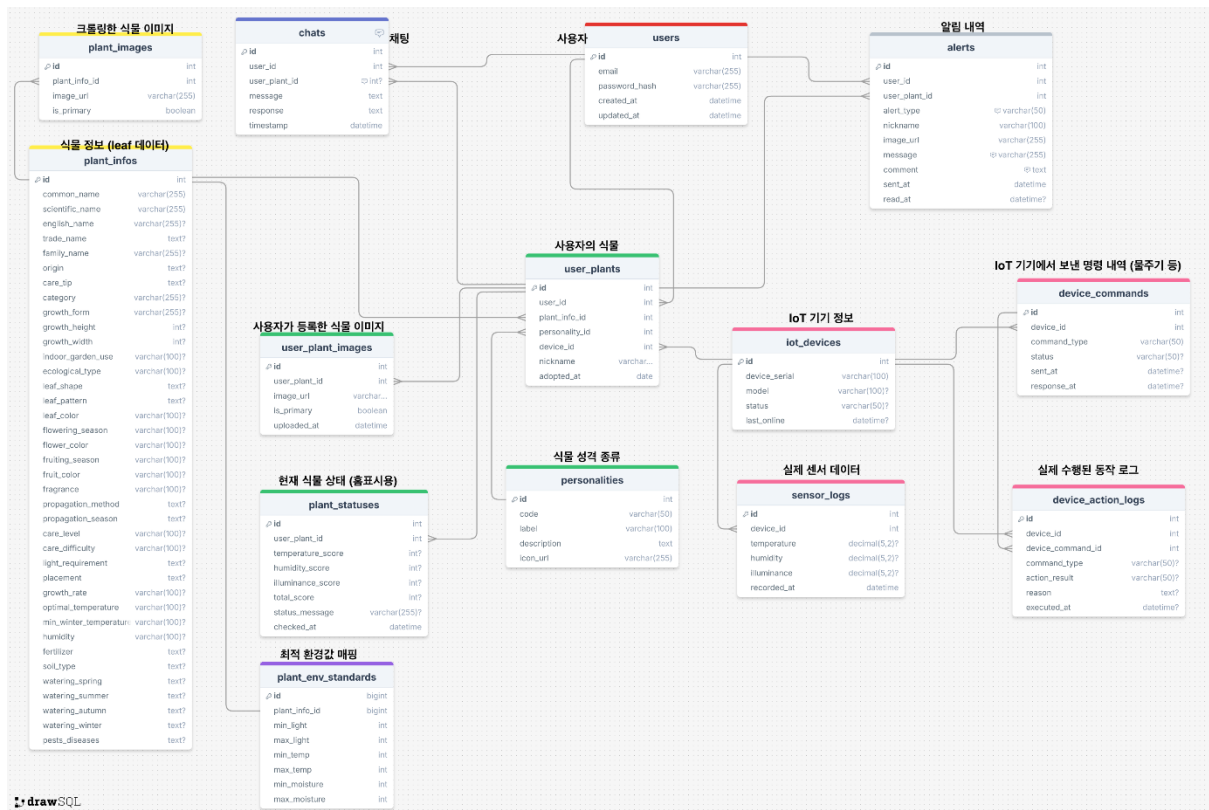
스마트 챗봇 기능은 사용자가 식물과 관련된 다양한 질문을 하거나, 등록된 반려식물과 대화를 나눌 수 있도록 지원한다.



[그림 8. 스마트 챗봇]

대화 시작 화면에서는 두 가지 모드를 선택할 수 있다. 첫 번째는 "식물 챗봇과 대화하기"로, 식물에 대한 일반적인 지식이나 정보를 질문할 수 있는 모드이다. 두 번째는 "내 반려식물과 대화하기"로, 사용자가 등록한 반려식물 중 하나를 선택하여 친근한 대화를 나눌 수 있는 모드이다. 반려식물과의 대화 모드에서는 사용자가 등록한 성격에 따라 식물의 말투와 사용하는 이모티콘이 달라진다. 또한, 선택한 반려식물에 특화된 답변을 제공하여, 보다 몰입감 있고 현실감 있는 대화 경험을 지원한다.

ERD



[그림 9. ERD]

ERD는 사용자, 반려식물, IoT 기기, 센서 데이터 등 주요 엔터티를 중심으로 구성되어 있다. 전체 구조는 사용자가 반려식물을 등록하고, 이를 IoT 기기를 통해 관리하며, 식물의 상태 변화를 모니터링하고 알림이나 챗봇 기능으로 상호작용하는 일련의 흐름을 지원하는 데 중점을 두고 설계되었다.

users 테이블은 시스템의 모든 사용자 정보를 관리하는 중심 역할을 수행한다. 사용자는 user_plants를 통해 자신의 반려식물을 등록할 수 있으며, 각 반려식물은 plant_infos(식물 도감 정보) 및 personalities(식물 성격 정보)와 연결되어 있다. 등록된 반려식물은 선택적으로 IoT 기기(iot_devices)와 연동할 수 있으며, 이를 통해 실시간으로 수집되는 환경 데이터는 sensor_logs 테이블에 기록된다. 수집된 데이터를 기반으로 식물의 현재 상태는 plant_statuses 테이블에 저장되며, 사용자는 이를 통해 식물의 건강 상태를 지속적으로 확인할 수 있다. 또한, 사용자는 chats 테이블을 통해 식물 챗봇과의 대화 기록을 저장하고 관리할 수 있으며, 식물의 상태나 관리 알림은 alerts 테이블을 통해 제공된다.

본 ERD 구조를 통해 Planty는 반려식물 관리의 자동화, 실시간 모니터링, 사용자와 식물 간의 인터랙션을 통합적으로 지원할 수 있도록 설계되었다.

API 명세서

구분	Controller	API 기능	API Path	HTTP 메서드	Request	Response	비고
Auth	AuthController	회원가입	/auth/signup	POST	SignupRequestDto	200 OK	사용자를 새로 등록하고 비밀번호를 해시 처리하여 저장
Auth	AuthController	로그인	/auth/login	POST	LoginRequestDto	200 OK	이메일과 비밀번호를 검증하고 JWT 토큰 발급
Auth	AuthController	토큰 유효성 검사	/auth/validate	GET	-	200 OK	토큰의 유효성 검사
User	UserController	비밀번호 변경	/users/password	PUT	PasswordChangeRequestDto	200 OK	현재 비밀번호로 확인 후 새 비밀번호로 갱신
User	UserController	비밀번호 찾기	/users/reset	POST	ResetRequestDto	200 OK	비밀번호 재설정
Alert	AlertController	알림 등록	/alerts	POST	AlertRequestDto	200 OK	사용자 정보 및 관련 데이터 비동기적으로 불러옴
Alert	AlertController	알림 조회	/alerts/{alertId}	GET	-	200 OK	사용자별 알림 목록을 최신순으로 불러옴
Plant	PlantController	식물 도감 조회	/plants	GET	-	200 OK	식물 ID로 plant_info에서 상세 정보 조회
Plant	PlantController	식물 상세 정보 조회	/plants/{plantId}	GET	-	200 OK	식물 ID로 plant_info에서 상세 정보 조회
UserPlant	UserPlantController	사용자의 반려식물 목록 조회	/user-plants	GET	-	200 OK	로그인 사용자의 user_plants 목록 조회 (총 10개 이하)
UserPlant	UserPlantController	반려식물 상세 정보	/user-plants/{userPlantId}	GET	-	200 OK	내 반려식물의 상세 정보 + 실시간 IoT 상태 + 식물 도감 정보
UserPlant	UserPlantController	사용자 식물 기증 등록	/user-plants	POST	UserPlantCreateRequestDto	200 OK	사용자가 식물 도감에서 선택한 식물용 기증으로 반납
UserPlant	UserPlantController	반려식물에 IoT 기기 연결	/user-plants/{userPlantId}/device	POST	ConnectDeviceRequestDto	200 OK	등록된 반려식물에 IoT 기기를 연결함
UserPlant	UserPlantController	반려식물 삭제	/user-plants/{userPlantId}	DELETE	-	200 OK	사용자 식물 삭제 (user_plants에서 제거)
Personality	PersonalityController	성격 조회	/personalities	GET	-	200 OK	식물 성격 조회

[그림 10. API 명세서]

API 명세서는 Planty 애플리케이션의 클라이언트와 서버 간 통신을 위해 정의된 엔드포인트 목록과 요청/응답 규격을 체계적으로 정리한 문서이다. 주요 기능 단위별로 컨트롤러를 구분하고 있으며, 각 API의 HTTP 메서드, 요청 경로, 요청(Request) 및 응답(Response) 데이터 형식을 명확히 정의하여, 일관된 통신 흐름을 지원하도록 설계하였다.

API는 다음과 같은 기능 그룹으로 구성되어 있다.

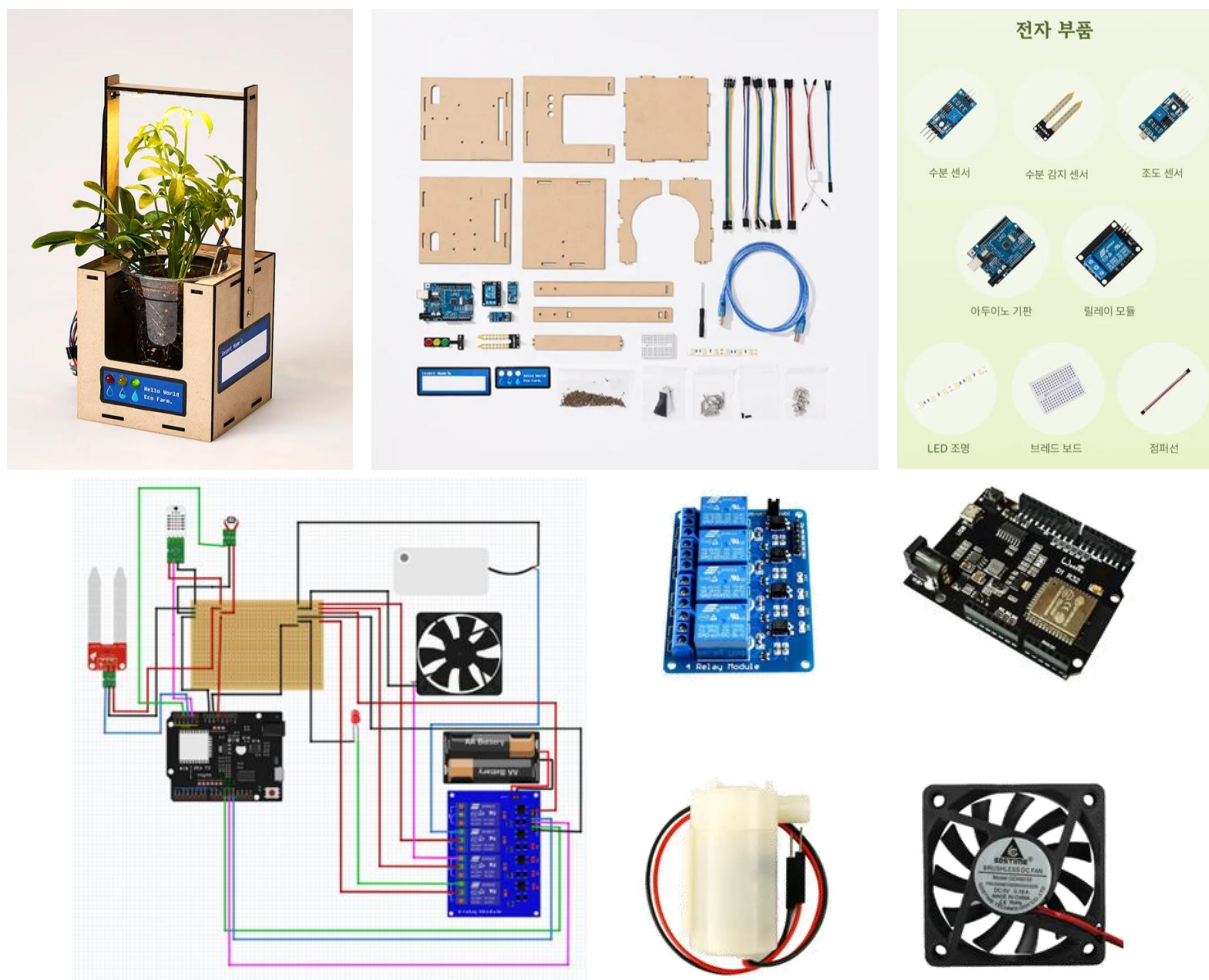
1. 회원 관리: 회원가입, 로그인, 비밀번호 변경/초기화 등의 사용자 인증 및 계정 관리 기능
2. 식물 관리: 식물 도감 조회, 반려식물 등록 및 조회, 반려식물 상태 확인 기능
3. IoT 기기 연동: IoT 기기 등록, 제어 명령 전송(물주기, 조명 제어 등), 센서 데이터 수집 기능
4. 알림 기능: 반려식물 상태 변화에 따른 사용자 알림 조회 및 알림 설정 변경 기능
5. 챗봇 기능: 식물 챗봇과의 대화를 시작하고, 대화 기록을 조회 및 저장하는 기능

RESTful 아키텍처 원칙을 준수하여 리소스를 명확하게 표현하고, HTTP 메서드를 기능에 맞게 사용하였다. 또한, 요청 결과에 대해 HTTP 상태 코드를 일관되게 적용함으로써, 클라이언트와 서버 간 통신의 명확성을 높였다. 해당 API 명세서는 팀원 간 협업, 클라이언트 앱 개발, 테스트 및 유지보수 과정에서 참조 자료로 활용되며, 기능 추가 및 수정 시 지속적으로 업데이트한다.

클라이언트와 서버 간 데이터 송수신 시에는 각 요청과 응답에 대해 전용 DTO(Data Transfer Object)를 사용하여 데이터 구조를 명확히 분리하였다. 요청(Request)과 응답(Response) 데이터 모델을 명확히 구분함으로써, 데이터 일관성과 확장성을 확보하고, 서비스 계층과 컨트롤러 계층 간의 책임을 명확히 분리하였다. 모든 주요 API 엔드포인트에 대해 별도의 Request DTO와 Response DTO를 정의하여 사용하고 있으며, 이를 통해 유지보수성과 코드 가독성을 높였다.

IoT

IoT 설계



[그림 11. IoT 구성 및 회로 설계]

PlanTy의 IoT를 위해 에코드인의 스마트 테라리움 키트를 기반으로 다른 부품들을 추가하였다. ESP32 보드를 중심으로, 식물의 정보를 수집하기 위한 토양습도센서,

온습도센서(DHT), 조도센서(CDS) 등의 환경 감지 센서와 식물 관리 자동화를 위한 워터펌프, 팬, LED, 릴레이 모듈이 포함된다.

시스템의 안정화와 효율성 향상을 위해 다음 세가지 원칙에 따라 회로를 설계하였다.

1. 전원 분리

릴레이 모듈을 통해 워터펌프, 팬, LED의 전원을 개별적으로 제어할 수 있도록 구성하였다. 특히 워터펌프 및 팬과 같이 높은 전류가 필요한 부하에는 릴레이 모듈 자체에 별도의 건전지를 연결하여, ESP32의 전력 부족으로 인한 동작 불능을 방지하였다.

2. IO 핀 분리

ESP32의 각 IO 핀은 센서 및 제어 모듈별로 독립적으로 연결하였다. 이를 통해 센서로부터 수집된 데이터를 안정적으로 수신하고, 각 제어 모듈에 정확한 제어 신호를 전달할 수 있도록 하였다.

3. 불필요한 연결 최소화

초기 회로 구상 단계에서 각 부품의 동작 전압과 전류를 고려하여 배선을 단순화하였고, 중복되거나 비효율적인 연결을 제거하여 회로의 안정성과 유지보수성을 확보하였다.

IoT 구성요소 연결 방식

- 토양 습도 센서는 ESP32의 아날로그 입력 핀(A0 등)에 연결하여 토양 내 수분량을 실시간으로 측정하며, 측정값을 기반으로 워터펌프의 동작 여부를 제어한다.
- DHT 온습도 센서는 디지털 핀을 통해 연결되며, 온도와 습도 데이터를 측정하여 팬 동작의 기준 값으로 활용한다.
- 조도 센서(CDS)는 아날로그 입력을 통해 실내 조도 상태를 감지하며, 일정 기준 이하의 조도일 경우 LED를 자동 점등하도록 설계하였다.
- 릴레이 모듈은 워터펌프, 팬, LED를 각각 제어하는 릴레이 채널 3개를 사용하였다. 릴레이 제어 신호는 ESP32의 디지털 출력 핀에서 전달되며, 모듈 자체는 별도 전원(건전지)을 통해 구동되어 전류 안정성을 확보하였다.

작동 흐름

각 센서에서 수집된 데이터는 실시간으로 ESP32에 전달되며, 사전에 정의된 임계값과 비교하여 다음과 같은 제어 로직을 수행한다:

- 토양 수분이 부족하면 릴레이를 통해 워터펌프를 작동
- 실내 온도 또는 습도가 높으면 팬 작동
- 조도가 낮으면 LED 조명 점등

이러한 방식으로 식물의 성장 환경을 자동으로 유지하며, 사용자가 개입하지 않아도 기본적인 환경 관리가 가능하도록 설계되었다.

MQTT 통신

MQTT(Message Queueing Telemetry Transport)는 발행-구독(Publish-Subscribe) 기반의 메시지 송수신 프로토콜이다. IoT와 같은 제한된, 혹은 대규모 트래픽 전송을 위해 만들어진 프로토콜이다. IoT 기기에서 수집한 데이터를 실시간으로 처리하는 데 사용한다.

LLM

PlanTy는 사용자가 식물과 정서적인 교감을 나눌 수 있도록 하는 챗봇 기능을 중심으로 설계되었다. 이 챗봇은 단순한 정보 제공을 넘어 감성적인 응답까지 제공해야 한다. SLM 모델을 사용하여 사용자가 데이터를 외부 서버로 전송하지 않고 모바일 기기 내에서 대화를 처리할 수 있도록 하여, 안정적인 답변이 가능하도록 할 예정이다. 파인튜닝과 RAG를 통해 SLM모델이 식물에 특화된 답변을 할 수 있도록 하고, Agent를 통해 사용자가 지정한 식물의 페르소나를 반영하여 정서적인 반응이 가능하도록 할 것이다.

SLM

대규모 언어 모델 (Large Language Model, LLM)은 방대한 양의 데이터를 바탕으로 자연어를 이해하고 생성하는 모델이다. LLM 모델은 계산 리소스를 많이 소모하고, 클라우드 기반의 연산 환경이 필요하며, 개인 정보 유출 등의 보안 문제가 동반될 수 있다. 반면, SLM(Small Language Model)은 LLM을 경량화한 모델로, 제한된 환경에서도 작동 가능하도록 최적화되어 있다. 파라미터 수는 줄었지만 특정 도메인이나 목적에 적합하도록 학습되면, 작은 크기에도 불구하고 매우 유용한 성능을 낼 수 있다. 특히, 모바일 기기나 임베디드 시스템에서도 온디바이스로 실행이 가능하다. 아직은 기술이 부족하여 모바일

상에서 서비스는 현실적으로 불가능하지만, 이번에 SLM 모델을 사용하여 어플리케이션이 오픈소스로 배포되었을 때 더 발전할 수 있는 가능성을 열어두고자 한다.

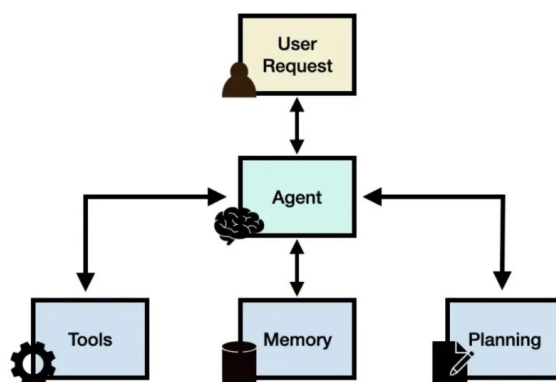
파인튜닝

파인튜닝은 사전 학습된 언어 모델을 특정 도메인이나 태스크에 맞게 추가 학습하는 과정이다. 사전 학습된 언어 모델은 식물에 대해 전문적으로 답변하거나 정서적으로 반응하는 능력이 부족할 수 있다. 이를 보완하기 위해, 식물과 관련된 데이터셋을 기반으로 모델을 추가 학습시켜 사용자가 원하는 모델에 부합하도록 모델을 조정한다.

RAG

RAG는 '검색 증강 생성(Retrieval-Augmented Generation)'이라고 번역할 수 있으며, 대규모 언어 모델에 외부 지식 베이스를 결합해 보다 정확하고 신뢰할 수 있는 답변을 생성하는 기술이다. 사용자의 질문에 대해 관련 문서를 검색하고, 그 결과를 바탕으로 LLM이 응답을 생성하는 방식으로 작동한다. 이를 통해 사실 오류와 맥락 부족 문제를 보완할 수 있으며, 최신 정보를 반영하고 출력 결과에 대한 근거를 제시함으로써 설명 가능성과 신뢰성을 높일 수 있다.

Agent

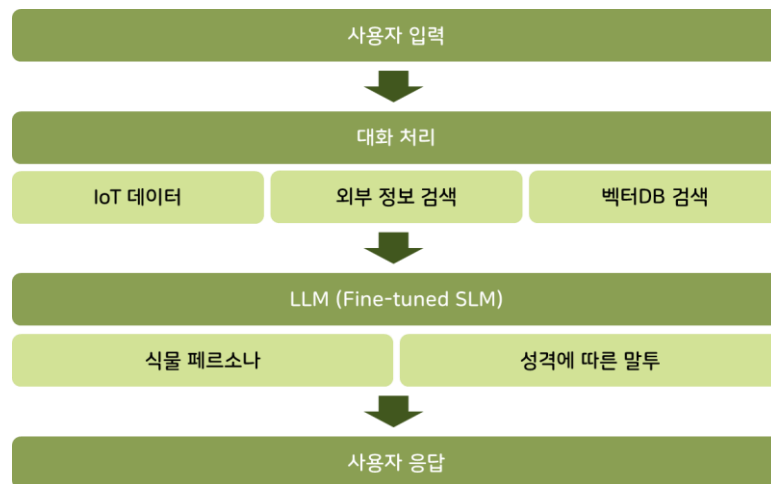


[그림 12. Agent 구조]

대규모 언어 모델을 기반으로 작동하는 자율적 인공지능 시스템으로, 단순한 질의응답을 넘어 복잡한 작업 수행 능력을 갖춘 고급 AI 도구이다. 사용자 요청을 이해하고, 필요한 정보를 스스로 탐색하거나 계획을 수립하여 문제 해결까지 이어지는 일련의 과정을 능동적으로 처리할 수 있다. 자연어 이해뿐만 아니라 의사 결정, 외부 도구 활용, 환경과의

상호작용 등 복합적인 기능을 수행하며, 특히 작업 흐름을 설계하고 다양한 시스템과 연동할 수 있는 유연성과 확장성을 지닌다는 점에서 기존 AI와 구별된다

Agent 설계



[그림 13. Agent 설계]

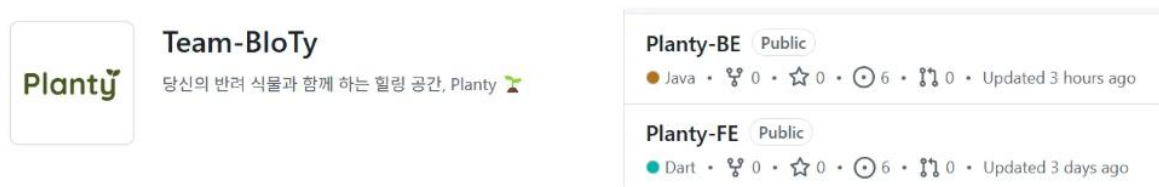
이번 프로젝트에서 설계한 LLM 기반 Agent 시스템은 식물과 사용자의 감성적 상호작용을 중심으로 구성된 대화형 구조이다. 사용자의 입력이 들어오면 대화 처리 모듈이 이를 분석하고, 외부 지식 소스(RAG)와 IoT 센서 데이터를 활용해 문맥을 보완한다. 이렇게 수집된 정보를 종합하여 파인튜닝된 LLM에 전달하면, 모델은 사용자 질문에 적합한 응답을 생성한다. 이때 사용자가 지정한 식물의 성격에 따라 해당 페르소나가 적용된 에이전트가 활성화되며, 각기 다른 말투와 표현 방식을 기반으로 답변이 이루어진다. 생성된 응답은 미리 설정된 성격별 말투를 유지하도록 설계되어 있으며, 기능적인 정보 제공을 넘어서 정서적인 교감까지 가능하도록 구성되어 있다.

3. 프로젝트 개발을 위한 환경 구축

협업도구

본 프로젝트의 협업 도구로 GitHub, Notion, Google Drive 및 Colab을 사용한다.

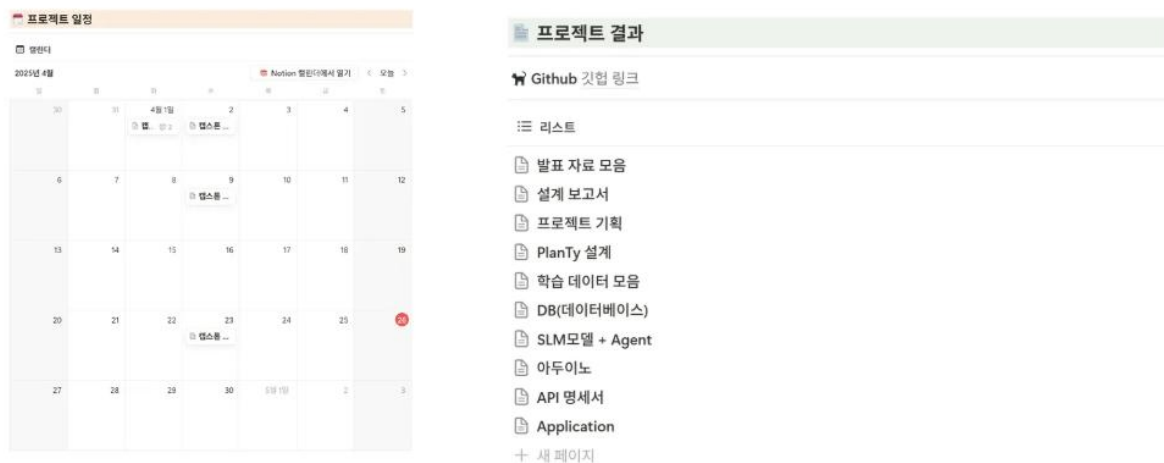
GitHub



[그림 14. BloTy Github]

GitHub는 프로젝트의 소스 코드 저장소로 사용한다. Issue를 사용하여 주요 작업, 버그 보고 및 기능 개선 요청을 관리하고, Git을 이용한 버전 관리를 통해 변경 이력을 추적할 수 있다. 개발을 완료한 코드는 자신의 branch에서 작업 후 pull request를 통해 main 브랜치에 코드를 반영한다.

















Notion



[그림 14. BloTy Notion]

Notion은 프로젝트의 통합 정보 관리 및 협업 워크 스페이스로 활용한다. 캘린더에는 프로젝트의 전반적인 일정을 기록하고, 회의록을 작성한다. 프로젝트를 기획 및 설계하는 과정에서 요구사항 정의, 설계 문서 등 프로젝트 관련 문서 들을 공동 작성한다. 또한 개발 과정에서 필요한 기술 자료, 스터디 내용 등을 팀원 간에 공유하고, github을 통해 공유한 코드들에 대한 설명을 notion을 통해 공유한다.

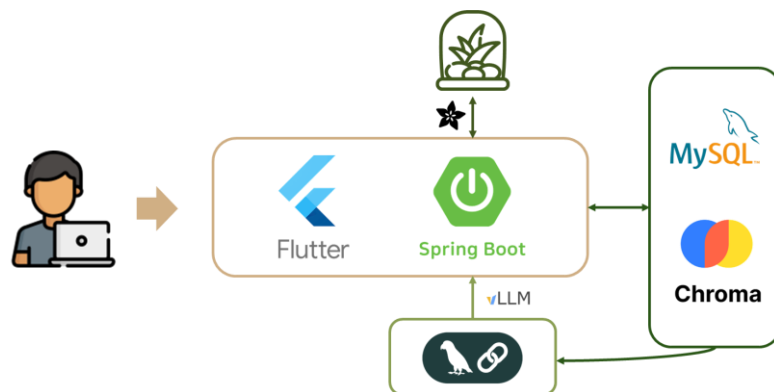
Google Drive & Colab

이름 	소유자	수정 날짜	파일 크기
 SFT 데이터터.ipynb 	 나	3월 25일 나	198KB
 SFT data generation.ipynb 	 나	3월 9일	327KB
 SLM 파인튜닝 모델 테스트.ipynb 	 나	4월 3일 나	580KB
 SLM 파인튜닝_1.ipynb 	 나	4월 1일 나	166KB
 SLM 파인튜닝_2.ipynb 	 나	4월 1일 나	263KB
 SLM 파인튜닝_3.ipynb 	 나	4월 1일 나	217KB

[그림 14. BioTy Google Drive]

Google Colab은 인공지능 모델 개발 및 실험을 위한 클라우드 기반 실행 환경으로 사용된다. SLM 모델을 파인튜닝 할 때 Colab의 A100 GPU를 사용하기 때문에, Colab을 사용하여 코드를 공유하고 공동 작성하였다. Colab을 사용할 경우 데이터셋을 Google Drive에서 가지고 오기 때문에, Google Drive에 저장된 데이터도 함께 공유하였다.

개발환경



[그림 14. PlanTy 개발환경]

IoT

하드웨어

- **아두이노:** IoT 기기
- **센서:** 온습도, 조도, 토양습도
- **모듈:** LED, 워터펌프, 쿨링팬

개발 언어

- **C++:** DHT.h, Wifi.h 등의 라이브러리로 IoT 기기 설정
- **Adafruit IO:** MQTT 통신

데이터베이스

- **MySQL:** IoT에서 수집한 데이터 및 매핑할 식물 데이터 저장

Application

디자인툴

- **Figma:** UI/UX 디자인, 프로토타입 작성

개발 언어, 프레임워크

- **Dart / Flutter (프론트):** 사용자 UI, 챗봇, 알림, IoT 정보 표시 기능 개발
- **Java / Springboot (백엔드):** DB 연동, IoT 센서 데이터 수신, 비즈니스 로직 처리 및 API 서버 구축

API 인증 및 보안

- **JWT (Json Web Token):** 사용자 인증 및 세션 관리를 위한 토큰 기반 보안 시스템 적용
- **Spring Security:** 서버 측 인증(Authentication) 및 권한 부여(Authorization) 적용하여 API 접근 제어 강화

데이터베이스

- **MySQL:** 사용자, 식물, IoT 데이터 등을 저장하는 관계형 데이터베이스 관리 시스템(RDBMS)으로 사용

개발 환경 (IDE/툴)

- **Visual Studio Code + Xcode 시뮬레이터**: 프론트엔드 개발 및 iOS 환경 테스트
- **IntelliJ IDEA**: 백엔드 개발을 위한 통합 개발 환경

Test 도구

- **Swagger (OpenAPI 3.0)**: API 명세 자동화 및 통합 테스트

LLM

개발 언어 및 프레임워크

- **Python**: Python을 사용하여 챗봇 API 구현
- **Transformers**: 식물에 특화된 모델로 파인튜닝
- **LangChain**: RAG를 통한 식물 정보 검색
- **LangGraph**: 식물의 성격에 따라 답변 제공하는 에이전트 구현

데이터베이스

- **MySQL**: 챗봇 대화 기록 저장
- **ChromaDB**: RAG에 사용할 식물 및 농업 관련 데이터 저장

4. 구현 방법

IoT

회로설계

센서 및 제어 모듈을 ESP32 보드에 연결한 후, 아두이노 IDE를 통해 각 구성요소가 정상 작동하는지 테스트하였다. 회로는 모듈이 독립적으로 작동할 수 있도록 구성하되, 각 구성요소의 제어 및 데이터 핸들링은 소프트웨어적으로 통합되도록 설계하였다. 실제 데이터 흐름과 제어 조건은 구현 단계에서 코드로 처리하였다.

MQTT통신

센서에서 수집한 데이터를 LLM과 어플리케이션으로 전달하기 위해 MQTT 통신을 통해 Adafruit IO를 중간 데이터 저장소로 사용한다. 센서에서 수집한 토양습도, 조도, 온도 데이터를 저장하고, 어플리케이션에서 보낸 명령을 IoT에 전달하기 위한 매개체로 사용된다.

센서 데이터 매핑

센서에서 수집된 데이터는 애플리케이션 및 LLM과 연동하기 위해 다음과 같이 **표준화 및 매핑** 과정을 거친다:

- **온도:** 센서에서 섭씨 단위로 출력되며, 외부 데이터(기상 API, 크롤링 기준 등)도 동일 단위를 사용하므로 추가 변환 없이 사용된다.
- **습도:** 토양 습도 센서에서 출력되는 값은 아날로그 입력으로 0~4095 범위의 상대값이다. 이를 퍼센트 단위(%)로 변환하기 위해, 다양한 수분 상태의 토양에서 센서 값을 직접 수집하고, 물리적으로 관찰 가능한 기준(예: "건조", "촉촉함", "젖음")과 대응되는 값으로 매핑하였다. 이 기준은 모델 학습이나 판단 기준으로 사용된다.
- **조도:** 조도 센서 또한 0~4095 범위의 상대조도 값을 출력하므로, 실제 환경(예: 실내 형광등, 햇빛, 어두운 방)에서 측정한 lux 값을 참조하여 상대값과의 매핑 기준을 수립하였다. 이를 통해 외부 크롤링 데이터에서 사용되는 lux 단위와 연동할 수 있도록 하였다.

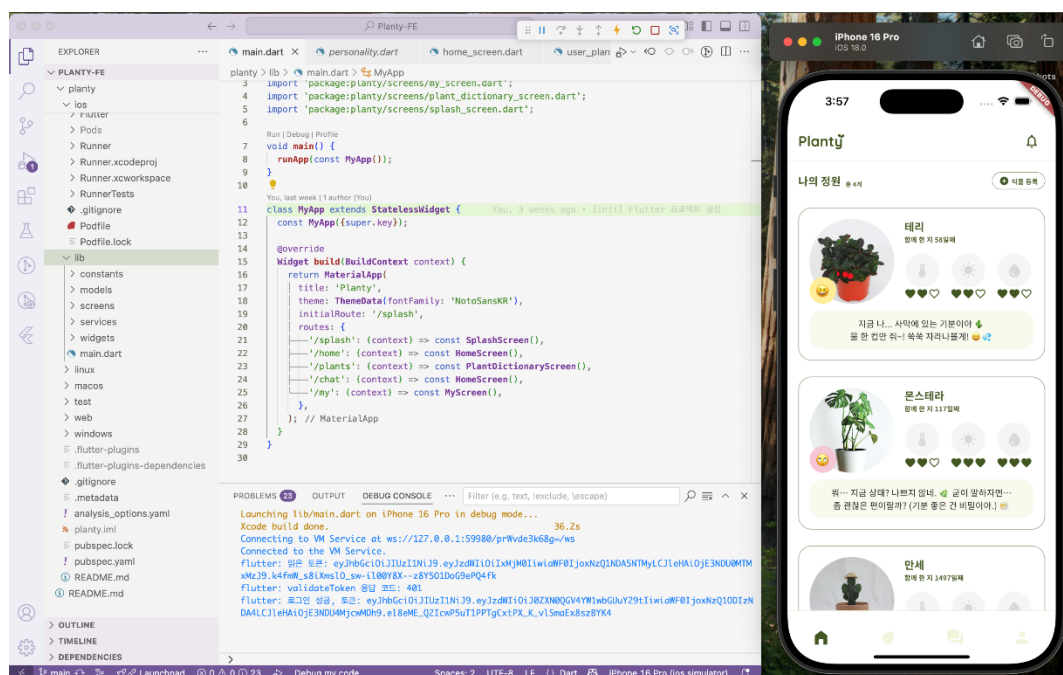
로직 및 제어 방식

센서 데이터를 기반으로 각 모듈의 작동 여부를 판단하는 로직은 ESP32 내부에서 구현되었다. 만약 토양 수분이 임계값 이하라면 워터 펌프를 켜다. 온습도가 기준을 초과하면 팬을 켜다. 조도가 기준치 이하라면 LED를 켜다. 이러한 설정은 어플리케이션에서 직접 제어하거나, Adafruit IO 피드를 통해 명령을 수신 받아 작동하게 할 수도 있다. 이를 통해 자동 제어와 수동 제어가 모두 가능한 하이브리드 제어 시스템을 구현하였다.

어플리케이션

Planty 어플리케이션은 Flutter 기반 프론트엔드와 Spring Boot 기반 백엔드 구조로 구성한다.

프론트엔드 개발 (Flutter)



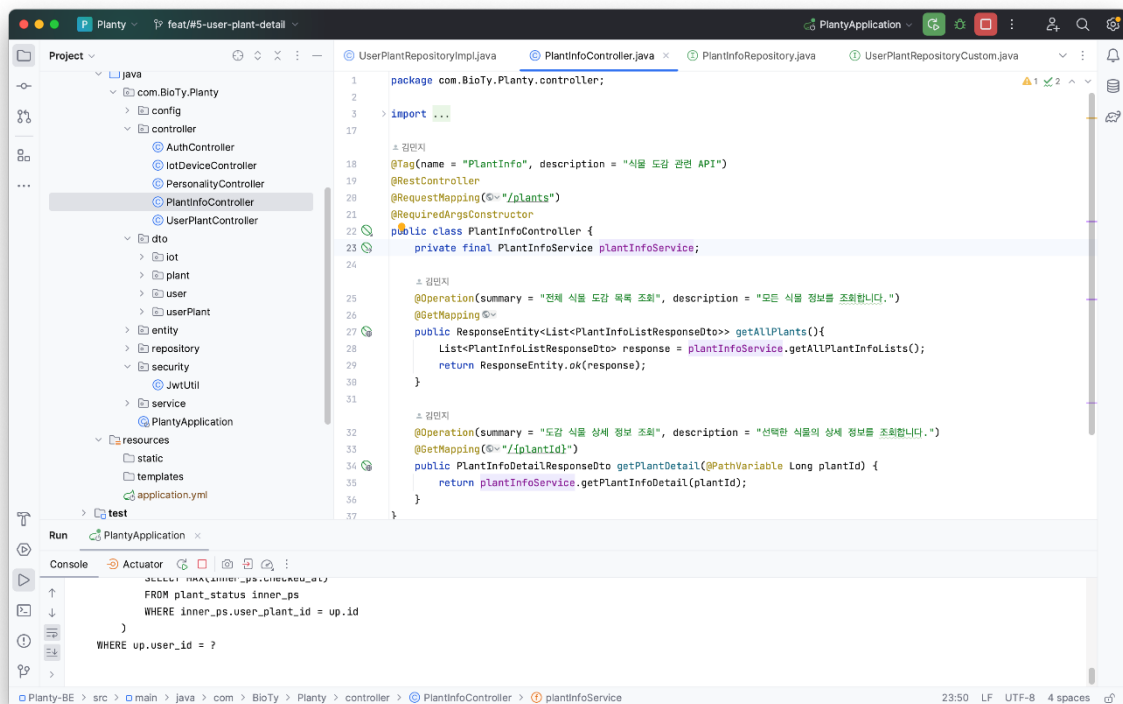
[그림 15. 프론트엔드 개발 (Flutter)]

Flutter를 이용하여 회원가입/로그인, 나의 정원, 식물 등록, 식물 도감, 스마트 챗봇 등 주요 화면을 개발한다. HTTP 통신 라이브러리를 통해 서버 API와 연동하며, JWT 기반 인증 토큰을 저장하여 세션을 관리한다. 반려식물 등록 및 관리 기능은 사용자의 선택에 따라 화면이 동적으로 업데이트되며, IoT 기기와 연동하여 물주기, 바람 조절, 조도 조절 등의 기능을 제공한다.

프로젝트는 개발 기능별로 디렉토리를 나누어 체계적으로 관리한다. screens, models, services, widgets, utils 등의 폴더를 분리하여 각 기능의 역할을 명확히 구분한다. 이러한 구조를 통해 코드 가독성과 유지보수성을 높이며, 협업 과정에서 코드 충돌을 최소화하고 개발 효율성을 향상시킨다.

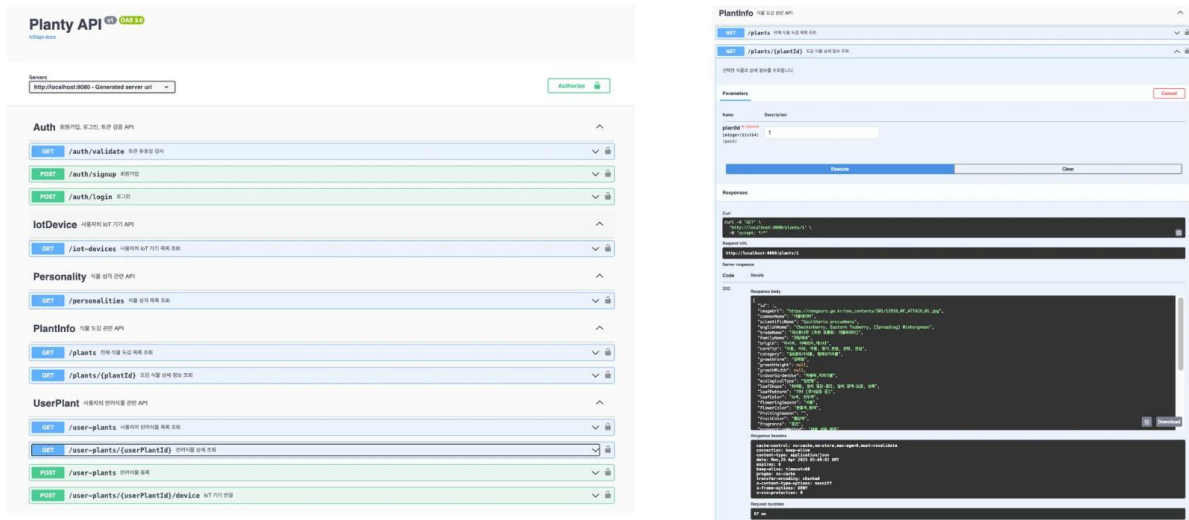
앱 테스트는 Xcode iOS Simulator를 이용하여 iOS 환경에서 진행한다.

백엔드 개발 (Spring Boot)



[그림 16. 백엔드 개발 (Spring Boot)]

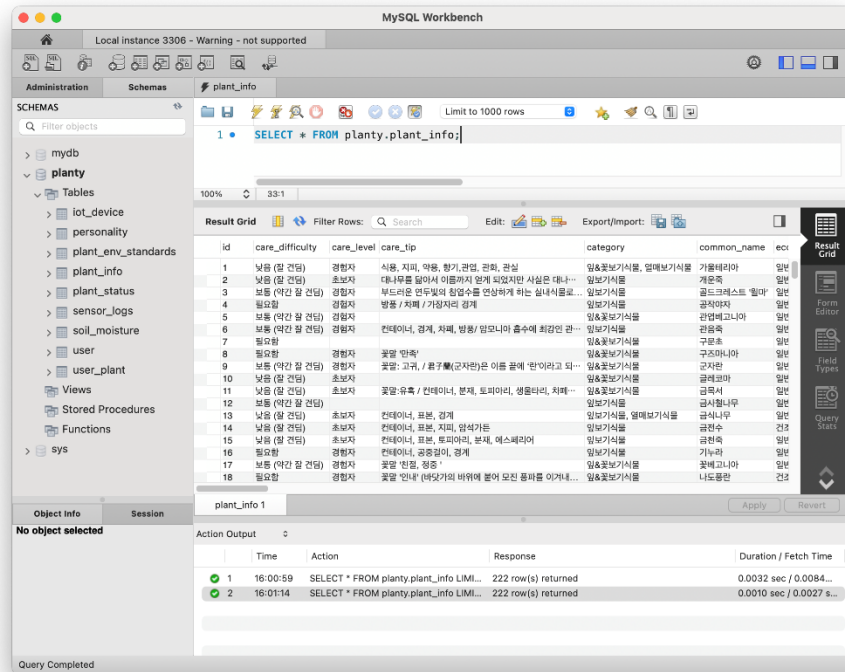
Spring Boot를 사용하여 RESTful API 서버를 구축한다. 요청 및 응답 데이터는 DTO(Data Transfer Object) 패턴으로 정의하여 계층 간 데이터 흐름을 명확히 분리한다. Spring Security와 JWT를 적용하여 사용자 인증 및 권한 관리를 처리하고, Swagger(OpenAPI 3.0)를 연동하여 API 명세를 자동화하고 테스트를 지원한다. IoT 기기에서 수집된 센서 데이터는 HTTP 통신을 통해 수신하며, 데이터베이스에 저장하고, 식물의 상태 모니터링 및 알림 기능에 활용한다.



[그림 17. API 개발 및 테스트]

API 개발과 테스트는 Swagger UI를 적용하여 API 명세와 테스트 환경을 통합 관리한다. Spring Boot 프로젝트에 Swagger(OpenAPI 3.0)를 연동하여, 구현된 API 엔드포인트들을 자동으로 문서화하고 있으며, 각 API 요청/응답 구조를 실시간으로 확인하고 테스트할 수 있도록 구성하였다. Swagger UI를 통해 API 호출 파라미터 입력, 응답 결과 확인, 오류 검증 등을 손쉽게 수행할 수 있어, 개발 중 API 검증 및 디버깅 과정의 효율성을 크게 향상시킨다. 현재는 로컬 개발 환경에서 테스트용으로 운영하고 있으며, 이후 버전 관리 및 공식 문서화에도 활용할 예정이다.

프론트엔드-백엔드 연동 테스트

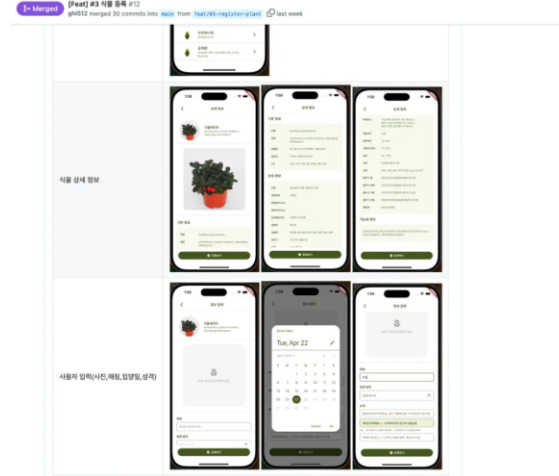
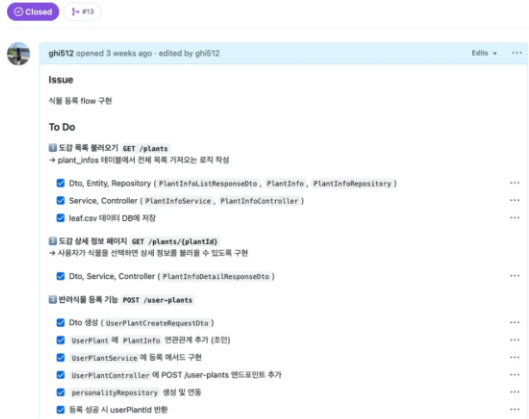


[그림 18. 프론트엔드-백엔드 연동 테스트]

개발이 완료된 기능에 대해서는 Flutter 프론트엔드와 Spring Boot 백엔드를 로컬 환경(localhost)에서 연결하여 통합 테스트를 진행한다. 프론트엔드 화면에서 사용자 액션(회원가입, 로그인, 식물 등록, IoT 제어 등)을 수행하면, 해당 요청이 실시간으로 백엔드 서버에 전달되고, 서버 로직에 따라 데이터베이스(MySQL)에 즉시 반영되는 과정을 검증한다. 이를 통해 API 통신 정상 여부와 데이터 흐름을 직접 확인하고, 프론트-백엔드-DB 간의 통합 동작을 실시간으로 검증한다.

협업 및 버전 관리

[Feat] 식물 등록 #4



[그림 19. 어플리케이션 협업 및 버전 관리]

GitHub를 활용하여 프론트엔드(Planty-FE)와 백엔드(Planty-BE) 레포지토리를 분리하여 관리한다. 프론트엔드와 백엔드는 각각 독립적인 브랜치를 운영하며, 기능 단위로 브랜치를 생성하여 개발을 진행한다.

Issues 기능을 활용하여 개발 이슈를 등록하고 관리한다. 각 이슈에는 개발할 기능, 세부 To-Do 리스트, 구현 방법 등을 자세히 기록하여 개발 과정을 체계적으로 관리한다. 마일스톤(Milestone)을 설정하여 기간 내 목표를 설정하고, 이슈들을 마일스톤에 연결하여 진행 상황을 관리한다.

Pull Request(PR)를 생성할 때는 개발 내용을 명확히 기록하며, 팀원들이 쉽게 이해할 수 있도록 설명과 함께 이미지나 GIF를 첨부한다. 구현한 기능의 흐름이나 화면 동작은 녹화하여 시각적으로 제공하며, 코드 변경 사항과 기능 변경 사항을 모두 상세히 작성하여 나머지 과정이 원활하게 이루어지도록 한다.

LLM

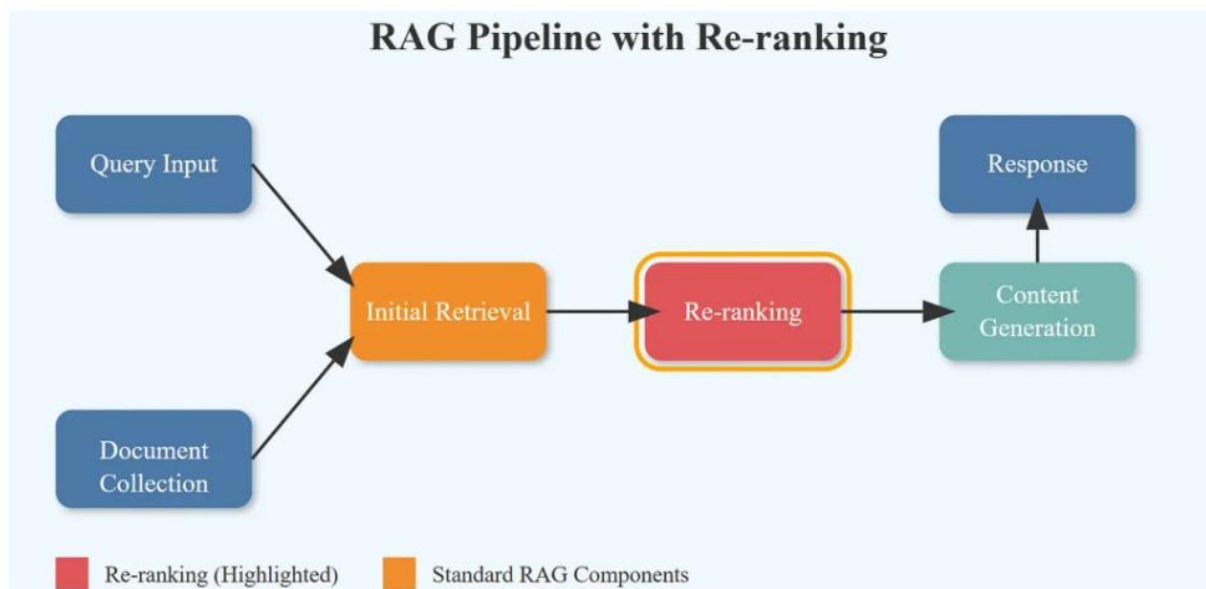
파인튜닝

파인튜닝을 진행할 SLM 모델을 찾기 위해 9b 이하의 파라미터를 가지는 모델 중 답변 시간이 짧게 걸리고, 식물에 대한 답변이 가장 좋은 모델을 선정하였다. 약 9개의 모델을 테스트하였고, 최종적으로 Gemma-3-4b-it을 기반으로, 식물 관리 및 정서적 상호작용에 최적화된 파인튜닝을 수행하였다. 파인튜닝은 SFT(Supervised Fine-Tuning) 방식으로 진행되었으며, QLoRA와 IA를 사용한 모델을 여러 개 만든 뒤 평가를 통해 하나의 모델을 선정하였다.

파인튜닝 모델 평가 방식

파인튜닝 데이터는 SFT 방식으로 파인튜닝되며, 학습 데이터는 [question: answer] 형식의 JSON 파일이다. 해당 데이터들은 사용자가 자주 묻는 식물 관련 질문 및 농업에 대한 정보를 포함하고 있다. 학습 모델의 성능을 검증하기 위해 학습 데이터와 같은 형식의 테스트 데이터셋을 수집하였다. 각 모델들의 테스트 데이터셋에 대한 응답을 저장한뒤, 모범 답안과 모델의 답변을 GPT-4o 모델에 전달하여 평가하였다. 모델은 식물 관련 정보의 정확성, 작문 실력 2가지를 기준으로 점수를 매겼고, 그 외에도 모델의 사이즈, 평균 응답 시간 등을 고려하여 종합적인 성능 비교 지표를 활용하였다.

RAG



[그림 20. RAG Pipeline with Re-ranking]

식물과 관련된 답변 성능을 높이고, 최신 정보를 반영하기 위해 RAG(Retrieval Augmented Generation)를 사용한다. 식물과 관련된 답변을 위해 대량의 식물 정보 PDF 파일을 로드하여 텍스트를 추출한 후, 텍스트를 문장 단위로 분할하여 벡터 임베딩을 수행하였다. 임베딩된 벡터는 벡터 데이터베이스에 저장되고, 벡터화된 문장들을 기반으로 검색 엔진을 통해 사용자의 질의와 관련된 정보를 빠르게 찾는다. 이후, 추출된 정보를 바탕으로 생성 모델이 자연어로 답변을 생성하여 사용자에게 전달하게 된다. 이러한 RAG 파이프라인 구축에는 LangChain 프레임워크를 활용하였다.

하지만 일반적으로 출시되는 어플리케이션의 경우 참고하는 파일의 크기가 방대하여, 위와 같은 기초적인 방식으로는 사용자의 질문에 대해 정확한 답변을 제공하기 어려운 문제가 발생한다. 본 프로젝트에서도 PDF 파일의 크기와 내용의 복잡성으로 인해, 검색된 문서가 질의와 정확히 일치하지 않거나 중요한 정보가 상위에 노출되지 않는 문제가 발생하였다.

이를 해결하기 위해, Reranker 기법을 도입하여 검색 기능을 개선하였다. 쿼리와 문서 간의 관련성 점수를 계산하여 검색된 문서들을 재정렬하는 두 단계의 프로세스를 수행하게 된다. 첫 번째 단계에서는 임베딩 기반 검색으로 관련 문서를 추출하고, 두 번째 단계에서 Reranker가 더욱 관련성 높은 문서를 상위로 끌어올려 정확한 정보를 제공한다. 이 방식은 대용량 문서에서 더 효율적이고 정확한 검색을 가능하게 하여, 최종 답변의 정확도를 높였다.

Agent

사용자의 질문에 대해 답변을 제공할 때, 사용자가 선택한 식물의 성격을 반영하기 위해 LangGraph를 활용하여 멀티 에이전트를 구현한다. LangGraph는 state를 기반으로 여러 에이전트와 도구 간의 상호작용을 그래프 형태로 유연하게 정의하고 관리할 수 있다. LangGraph로 구현된 워크플로우에서 여러 에이전트들은 고유한 페르소나를 가지며, 사용자는 인터페이스를 통해 원하는 에이전트 페르소나를 선택할 수 있다. 선택된 페르소나에 따라 해당 에이전트는 학습된 특징이 반영된 특정 말투, 어조, 표현 방식을 사용하여 최종 답변을 생성한다.

5. 개발 진행 상황

역할 분담

프로젝트의 효율적인 개발 및 영역별 전문성 확보를 위해 프로젝트의 핵심 구성 요소인 IoT, Application, LLM 세 부분으로 나누어 역할을 분담하였다. Application의 경우 IoT 및 LLM과 통신하는 데이터가 존재하거나, 협업이 필요한 부분은 공동 개발을 진행할 예정이다.

IoT

- 담당 팀원: 구선주
- 주요 역할 및 책임:
 - IoT 장치 설계 및 구현
 - 수집된 식물 데이터 전처리 및 가공
 - 센서로부터 수집하는 데이터 테스트 및 식물 데이터와 매핑
 - 장치와 백엔드 시스템 간의 통신 프로토콜 정의 및 구현

Application

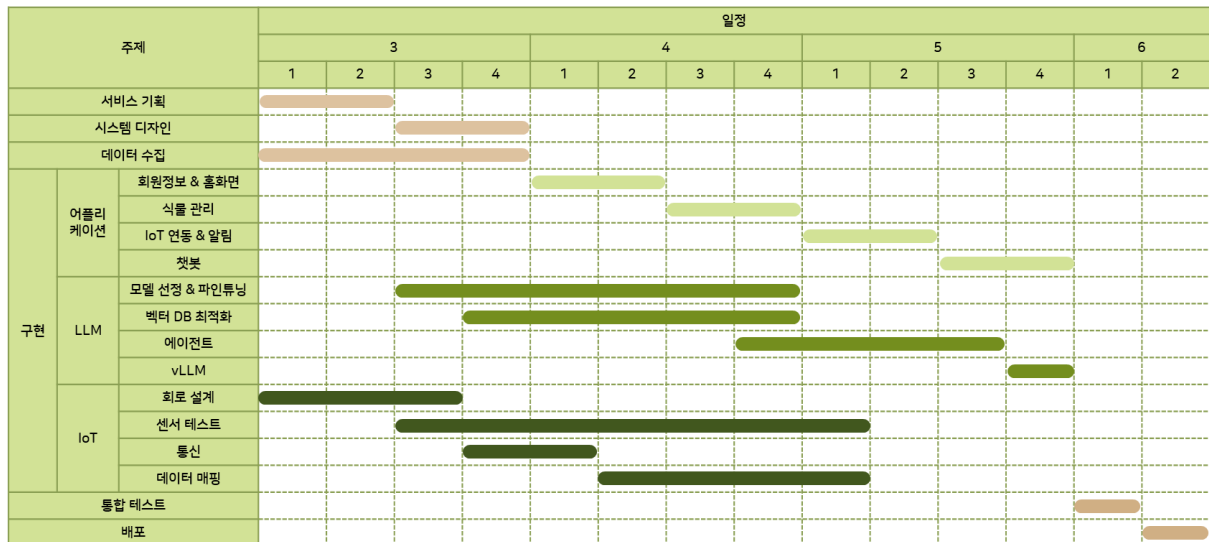
- 담당 팀원: 김민지
- 주요 역할 및 책임:
 - 사용자 인터페이스 (UI/UX) 설계 및 개발
 - 백엔드 서버 구축 및 관리
 - IoT 영역 및 LLM 영역과의 데이터 연동
 - 데이터 흐름 관리
 - 데이터베이스 설계 및 운영
 - 전체 시스템 통합 및 배포

LLM

- 담당 팀원: 민유진, 최예림
- 주요 역할 및 책임:
 - 식물 정보 및 농업 데이터 수집

- 파운데이션 모델 선정
- SLM 모델 파인튜닝
- 벡터 데이터베이스 및 RAG 최적화
- 에이전트 설계 및 프롬프트 엔지니어링
- 모델 응답 해석 및 최적화

개발 일정



[그림 21. 개발 일정]

프로젝트의 개발 일정표는 위의 사진과 같다. 2월 중 프로젝트 구상을 시작하여, 3월 중 프로젝트 기획을 완료하였다. 3월부터 개발을 시작하여, 5월 중으로 어플리케이션 개발을 마치고 6월 초에 어플리케이션 테스트 및 배포를 하는 것으로 계획 중이다.

개발 진행 상황

IoT

모듈 작동

설계한 회로대로 센서와 모듈을 연결한 뒤, 아두이노가 명령에 따라 정상 작동하는 것을 확인하였다. 아직 어플리케이션과 연결되는 부분이 완벽하게 구현되지 않아, 우선적으로 토양습도센서-워터펌프, 조도센서-LED, 온도센서-팬 간의 상호작용을 위주로 구현을 완료하였다.

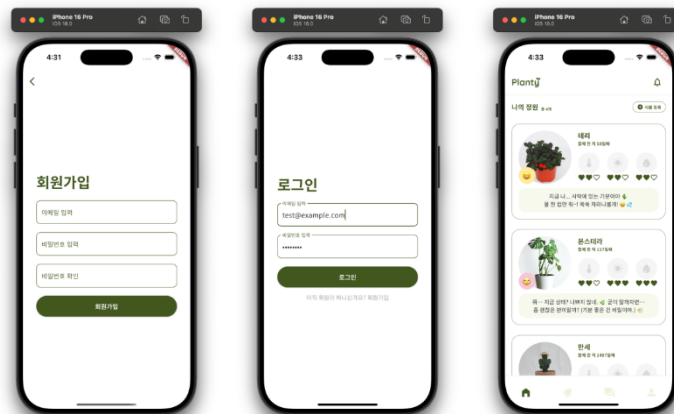
Planty			
Feed Name	Key	Last value	Recorded
<input type="checkbox"/> LightIntensity	planty.lightintensity	1728	1 minute ago
<input type="checkbox"/> SoilMoisture	planty.soilmoisture	249	1 minute ago
<input type="checkbox"/> Temperature	planty.temperature	19.40	1 minute ago

[그림 22. IoT 통신 결과]

어플리케이션과 통신의 경우 Wifi와 MQTT통신을 이용하여 Adafruit IO에 센서에서 수집한 데이터를 업로드하는 과정까지 진행하였다.

어플리케이션

회원가입 및 로그인 기능



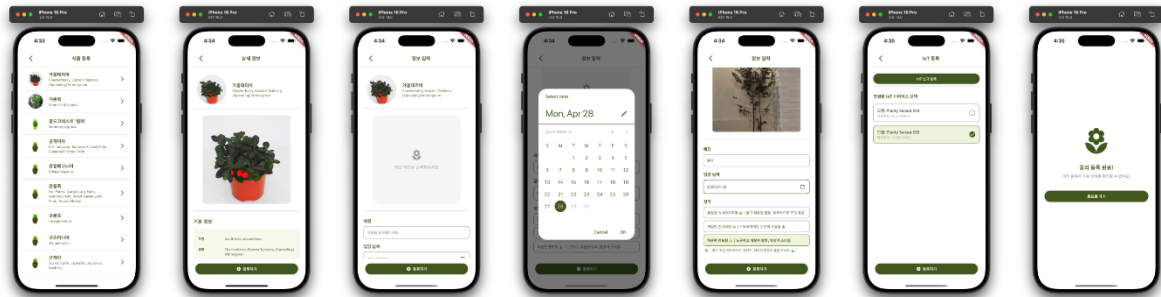
[그림 23. 회원가입 및 로그인 구현]

사용자는 이메일과 비밀번호로 회원가입 및 로그인을 할 수 있도록 구현하였다. 로그인 시 발급된 JWT 토큰을 Flutter Secure Storage에 저장하여 자동 로그인을 지원한다. Spring Boot 백엔드에서는 /auth/signup, /auth/login, /auth/validate API를 제공하고, Spring Security를 적용하여 인증 처리를 수행하였다. 프론트엔드에서는 공통 입력 필드와 버튼 위젯을 사용하여 로그인 및 회원가입 화면을 구성하였으며, Postman과 Flutter 연동 테스트를 통해 기능을 검증하였다.

내 식물 목록 (홈 화면) 기능

사용자는 홈 화면에서 등록한 반려식물 목록을 카드 형태로 확인할 수 있도록 구현하였다. 각 카드에는 식물 이름, 상태 메시지, 감정 이모티콘 등이 표시된다. Flutter에서는 API 응답을 기반으로 UserPlantCard를 동적으로 렌더링하고, 상단바와 하단바 UI도 함께 구성하였다. 백엔드에서는 사용자 토큰을 기반으로 반려식물 목록을 조회하는 /user-plants API를 구현하였으며, Native Query를 활용하여 사용자별 최신 식물 상태를 함께 조회할 수 있도록 하였다.

반려식물 등록 기능



[그림 23. 반려식물 등록 구현]

사용자는 홈 화면에서 등록 버튼을 클릭하여 식물 등록 플로우를 시작할 수 있도록 구현하였다. 식물 도감 목록과 상세 정보를 확인한 뒤, 애칭, 입양일, 식물 성격 선택, IoT 기기 연결까지 포함하여 반려식물을 등록할 수 있다. 백엔드에서는 /plants(식물 목록 조회), /plants/{plantId}(도감 상세 조회), /user-plants(반려식물 등록), /personalities(성격 목록 조회), /iot-devices(IoT 기기 목록 조회), /user-plants/{id}/device(IoT 기기 연결) API를 구현하였다.

LLM

Base Model 선택 및 파인튜닝

본 프로젝트의 LLM base model로 Gemma-3-4b-it를 선택하였으며, 식물 관리 및 정서적 상호작용에 최적화된 파인튜닝을 진행하였다. 파인튜닝은 SFT(Supervised Fine-Tuning) 방식으로 수행되었으며, QLoRA와 IA를 사용해 여러 모델을 제작하고, 성능 평가를 통해 최적의 모델을 선택하였다. 이 파인튜닝된 모델은 최종적으로 GPT-4o 모델에 전달되어 현재 반복적인 성능 테스트 중에 있다.

RAG 시스템 구축

대용량 PDF 파일을 활용한 RAG 시스템을 구축하였으며, 파일 크기와 내용의 복잡성으로 인해 발생한 문제를 해결하기 위해 Reranker 기법을 도입하여 검색 성능을 향상시켰다. 이 방식으로 관련성 높은 문서를 상위에 배치하고, 최종적으로 보다 정확한 정보를 제공할 수 있도록 성능 테스트를 완료하였다.

대화 기록 처리 설계

사용자와 챗봇 간의 대화 기록을 효율적으로 반영할 수 있도록 설계를 마무리하였다. 대화 기록은 MySQL 데이터베이스에 저장되며, 사용자가 이전에 진행한 대화와 관련된 정보를 쉽게 조회할 수 있도록 user_id가 일치하는 경우 해당 대화 내용을 대화 화면에 출력하는 방식으로 구현되었다.

추후 계획

IoT

현재 IoT 시스템의 기본적인 센서 연결과 모듈 제어는 완료된 상태이며, 향후 다음과 같은 작업을 통해 시스템의 정밀도와 완성도를 높일 예정이다.

우선 온습도 센서 데이터의 정규화 및 매핑 작업이 필요하다. 현재 센서에서 출력되는 값은 상대적인 수치로, 외부에서 수집한 식물 데이터와 형식이 상이하다. 이를 해결하기 위해 다양한 온도 및 습도 환경에서 센서 값을 직접 수집하고, 이를 같은 기준으로 매핑하는 실험을 진행할 예정이다.

다음으로 애플리케이션과의 통신 연결을 완성하고, 센서로부터 수집된 데이터를 실시간으로 전달하는 구조를 안정화할 계획이다. 이를 통해 사용자 애플리케이션에서 매핑된 데이터를 바탕으로 명령을 내릴 수 있으며, 해당 명령에 따라 워터펌프, 팬, LED 등 제어 모듈이 정상적으로 작동하는지 확인하고 최종 테스트를 거칠 예정이다.

어플리케이션

남은 기간 동안 반려식물 상세 화면, 스마트 챗봇, IoT 기기 제어, 알림 기능, 환경 리포트, 마이페이지 화면을 추가 개발할 예정이다. AWS를 활용한 배포도 준비 중으로, Spring Boot 서버는 EC2에 배포하고, 데이터베이스는 RDS를 통해 운영할 예정이다. 이미지 업로드 및 관리는 S3를 활용하는 방향으로 검토하고 있으며, 전체 인프라를 안정적으로 구성하는 것을 목표로 하고 있다.

또한, 현재는 아직 연동하지 않은 IoT 센서 데이터 수집과 챗봇 AI 대화 기록 저장 기능도 원활히 연결할 수 있도록 추가 작업을 진행할 예정이다. IoT 기기와 서버 간 데이터 송수신이 실시간으로 이뤄지도록 개선하고, 챗봇 대화 데이터는 사용자별로 자연스럽게 이어질 수 있도록 데이터 흐름을 다듬을 계획이다.

전체적으로는 기능별 성능 테스트를 추가로 진행하고, 속도나 안정성 측면에서 개선이 필요한 부분은 수정하여 전체 서비스 품질을 높여갈 예정이다.

LLM

RAG

현재 RAG 시스템은 자연어 생성 모델을 파인튜닝한 모델로 교체할 계획이다. 해당 모델은 성능 테스트 중이므로, 우선 Groq를 이용하여 RAG 시스템의 벡터 DB에서 정보를 가져와 답변을 제공하는 방식으로 운영하고 있다. 향후 파인튜닝 작업이 최종적으로 완료되면, 생성 모델을 새로운 파인튜닝 모델로 교체할 수 있도록 시스템 코드를 수정할 계획이다. 또한, 벡터 DB에서 관련 정보를 더욱 정교하게 추출하고, 생성된 답변의 품질을 향상시키기 위해 지속적인 성능 개선 작업을 진행할 예정이다.

Agent

LangGraph 프레임워크를 활용하여 Agent를 구현할 예정이다. 사용자가 식물의 성격을 선택하면, RAG의 데이터를 기반으로 답변을 내고, Agent가 주어진 성격을 기반으로 답변을 제공한다. 성격별로 다른 Agent를 만들어 페르소나를 부여하고, 페르소나에 어울리는 말투의 답변이 사용자에게 제공된다.

참고문헌

- [공지] (Plantgram) 식물 물주기 알람 앱 추천 . (2021).
<https://m.blog.naver.com/plantgram/222319436699>.
- [트렌드] '윌'으로 소통하는 반려식물이 대세 . (2023).
<https://www.spcmagazine.com/%EC%9E%8E%EC%9C%BC%EB%A1%9C-%EC%86%8C%ED%86%B5%ED%95%98%EB%8A%94-%EB%B0%98%EB%A0%A4%EC%8B%9D%EB%AC%BC%EC%9D%B4-%EB%8C%80%EC%84%B8/>.
- '반려식물' 인지도 1년 전보다 높아져, 농촌진흥청 . (2023).
https://rda.go.kr/board/boardfarminfo.do?mode=view&prgld=day_farmprmninfoEntry&dataNo=100000784485&CONTENT1=#script.
- 2030세대의 새로운 트렌드 '식집사'... 이제는 반려 식물 시대 . (2023).
<http://www.civicnews.com/news/articleView.html?idxno=35830>.
- AI가 상태진단, 집 비울 땐 '플랜트 호텔'...'쑥쑥' 크는 반려식물 벤처 . (2023).
<https://www.hankyung.com/article/2023020182721>.
- Plantgram(플랜트그램) 식물 물주기 알람 어플, 식물관리앱 추천 [출처]
Plantgram(플랜트그램) 식물 물주기 알람 어플, 식물관리앱 추천|작성자 Plantgram . (2020).
<https://blog.naver.com/plantgram/222094640988>.
- Z세대의 식물 키우기가 4050과 다른 점은?, 대학내일 20대연구소 . (2022).
<https://www.20slab.org/Archives/38217>.
- 검색 데이터로 확인한 반려 식물을 찾는 세대 별 특징 . (2024).
<https://kr.listeningmind.com/case-study/all-about-the-pet-plants/>.
- 그루우! - 글로벌 식물 앱 & 초개인화 식물쇼핑 혁신을 함께 만드실 분을 찾습니다. . (n.d).
<https://official-groo.notion.site/5df408b827654a6bb42eb9ac41751ffa>.
- 빅데이터로 알아보는 반려식물 . (2021).
<https://www.nongsaro.go.kr/portal/ps/psv/psvr/psvre/curationDtl.ps?menuId=PS03352&srchCurationNo=1696>.
- 식물도 '식구'... 쑥쑥 크는 '홈가드닝' . (2024).
https://www.chosun.com/economy/tech_it/2024/01/25/KE5K6T3DZ5C3JOLBOURZCPNZTE/

플랜톡 IoT 식물센서와 함께 하는 스마트한 식집사 생활 [출처] 플랜톡 IoT 식물센서와 함께 하는 스마트한 식집사 생활|작성자 행파고 . (2023).

<https://blog.naver.com/sjk5000/223137222493>.

홈가드닝을 위한 식물 키우기 어플 소개, groo 그루우 . (2022).

<https://blog.naver.com/sarah6612/222918650269>.