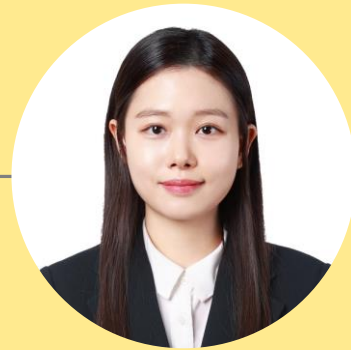


# Portfolio

책임감 있는 개발자  
이지수

# 책임감 있는 개발자

이지수



## 개인 정보

학교 / 전공 건국대학교 / 미디어커뮤니케이션학과

생년월일 1995.05.07

이메일 cloudsea1106@naver.com

깃허브 [깃허브 바로가기](#)

## 프로그래밍 역량

Python



웹 백엔드 개발, 데이터 수집 및 분석 경험  
알고리즘 문제 풀이 가능

Django



DRF를 통한 RESTful API 구현 가능  
세 번의 백엔드 프로젝트 진행 경험

DB



MySQL과 Redis를 사용하여 프로젝트 진행  
DB를 설계하고 SQL문을 통해 데이터 추출 가능

CI/CD



Jenkins와 Docker를 사용한 서버 환경 구축  
AWS EC2 서버 배포 경험

## 교육

삼성 청년 SW 아카데미 (SSAFY) 7기

2022.01 ~ 2022.12

- 공통 프로젝트 'PLANTinum' 우수상
- 특화 프로젝트 '꽃마리' 우수상
- 기업연계 프로젝트 'BOM' 최우수상

## 자격사항

SQLD

2022.09

컴퓨터활용능력1급

2018.10

## 프로젝트



기간 2022.10 ~ 2022.11

파트 백엔드, DB, 배포

내용

BMS 및 환자 건강 상태 모니터링 IoT 웹 서비스: **BOM(봄)**

- Jenkins와 Docker를 사용한 AWS EC2 서버 배포
- BMS 정보 기간 별 조회 및 엑셀 다운로드 기능
- 환자 건강 정보 기간 별 조회 및 엑셀 다운로드 기능
- 보호자 용 실시간 환자 건강 정보 조회 기능

#Python3 #Django(DRF) #MySQL



기간 2022.08 ~ 2022.10

파트 백엔드, DB, 배포

내용

빅데이터를 활용한 사용자 취향 기반 꽃 추천 서비스: **꽃마리**

- Jenkins와 Docker를 사용한 AWS EC2 서버 배포
- 하이브리드 필터링을 활용한 대상 별 꽃 추천 기능
- 협업 필터링을 활용한 좋아요 기반 게시물 추천 기능
- 꽃말과의 유사도를 활용한 편지 내용 기반 꽃 추천 기능

#Python3 #Django(DRF) #MySQL #Redis #Selenium



기간 2022.07 ~ 2022.08

파트 백엔드, DB

내용

IoT 기반 식물 자동 관리 및 웹 기반 식물 거래 서비스: **PLANTinum(플랜티넘)**

- 회원관리 기능
- 내 식물 정보 등록, 수정 및 토양 습도, 관수 날짜 정보 조회 기능
- 닉네임 자동 생성 및 프로필 조회, 수정 기능
- 내 식물 IoT 기기와 웹 연결 기능
- 식물 거래 글 작성 및 조회, 페이지네이션, 필터 기능

#Python3 #Django(DRF) #MySQL #Redis

# BOM

SAMSUNG SDI 기업연계 프로젝트

BMS 및 환자 건강 상태 모니터링 서비스



## 정보

기간 2022.10 ~ 2022.11

도메인 삼성SDI 기업 연계 프로젝트

참가 인원 HW 2, FE 2, BE 1

담당 파트 백엔드, DB, 배포

개발 환경 Python3, Django(DRF), MySQL, Jenkins, Docker, S3

수상 SSAFY 자율 PJT 최우수상

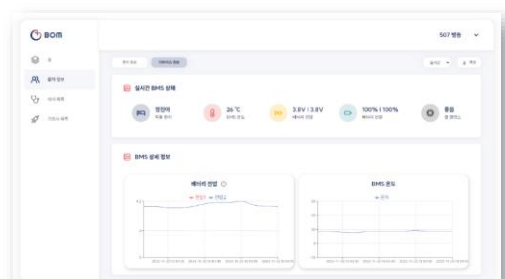
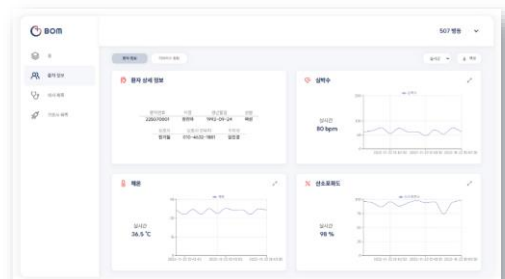
## 프로젝트 소개

**BOM**은 BMS 및 환자 건강 상태 모니터링 IoT 웹 서비스입니다.

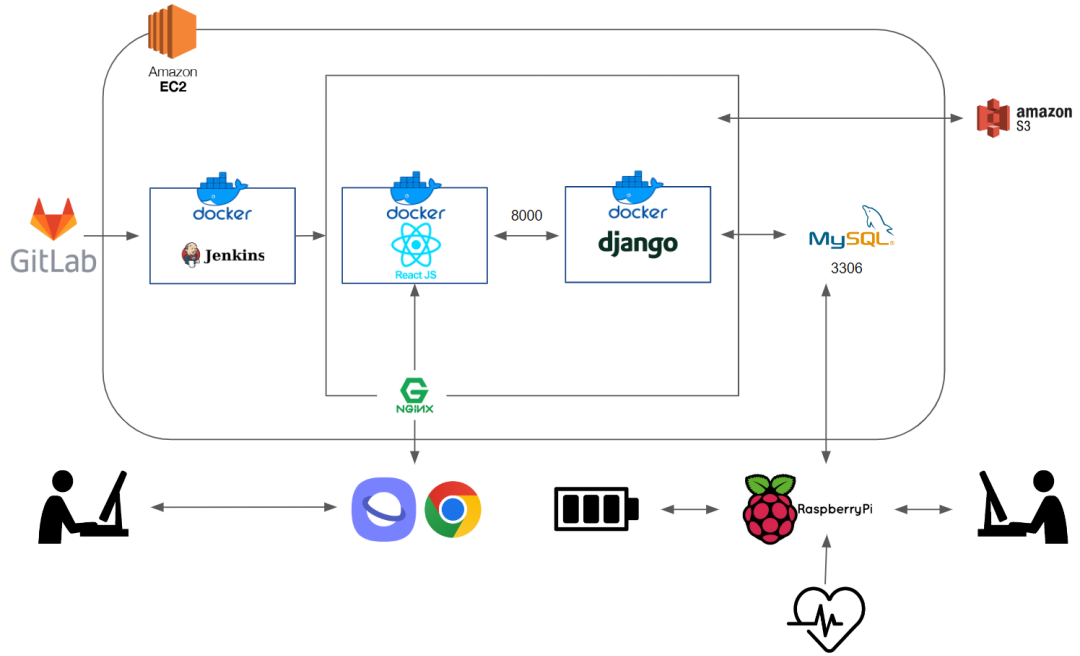
BMS와 배터리 상태를 체크하고, 산소포화도 및 온도 센서를 통해 환자의 건강 정보를 수집합니다. 병동에서는 수집된 데이터들을 시계열로 파악할 수 있고, 엑셀 파일로 다운로드 할 수 있습니다. 환자의 보호자는 실시간으로 환자의 건강 상태를 확인할 수 있습니다.

## 담당 기능

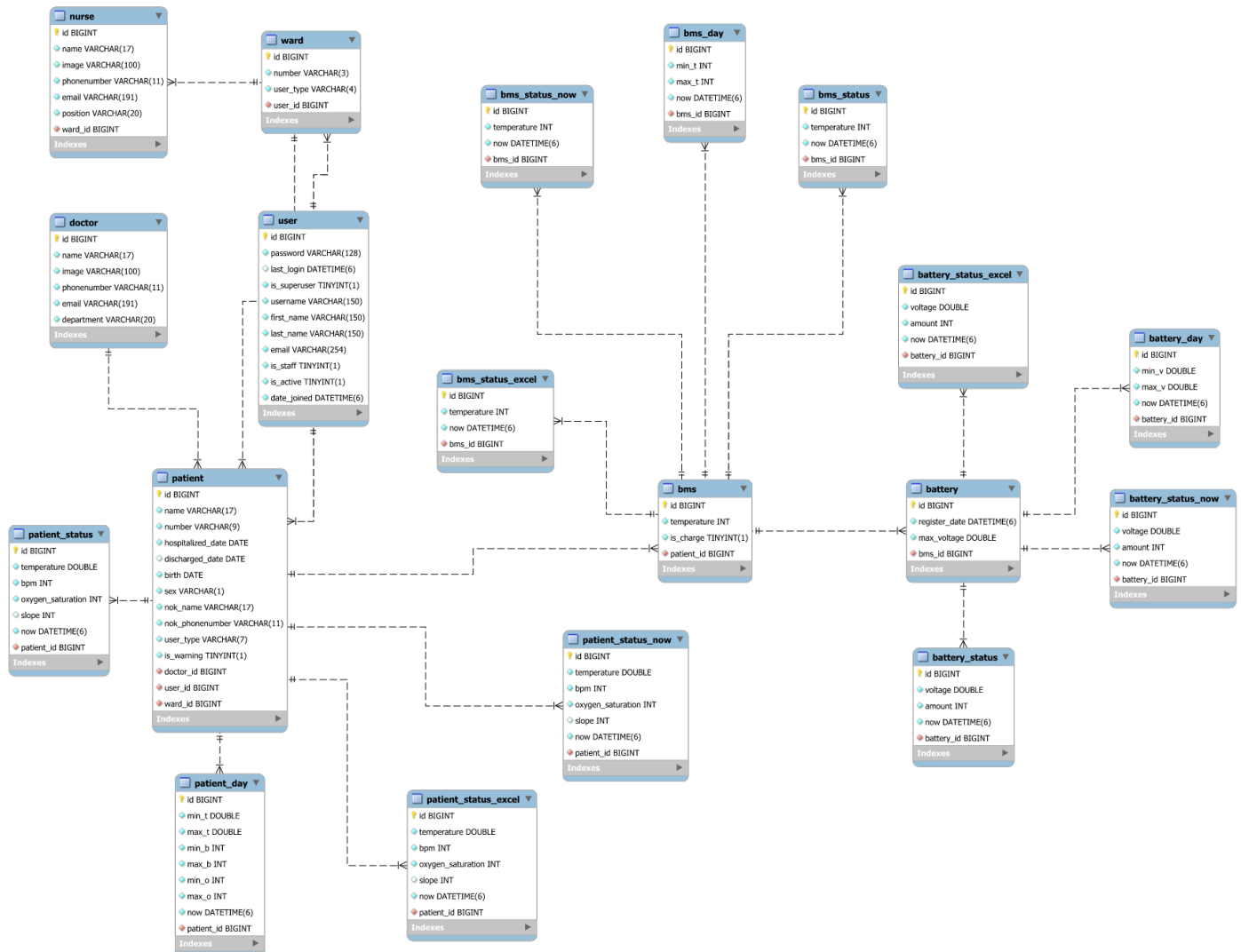
- 병동과 보호자를 구분한 JWT 로그인
  - access token 기간 2일, refresh token 기간 14일로 설정
- 환자 입원 추이, 병상 가동률 등 병동 정보 조회
- 이름, 생년월일, 담당 주치의 등 환자 상세 정보 조회
- 실시간 및 기간 별 환자 건강 정보 조회, 엑셀 다운로드
  - 체온, 심박수, 산소 포화도 조회 가능
  - 실시간은 현재와 최근 1분 동안의 정보 조회
  - 기간 별은 하루, 일주일, 한 달로 세분화하여 각각 1시간, 12시간, 1일을 기준으로 최댓값과 최솟값 조회
  - 엑셀은 매 정각에 해당하는 값을 기간 별로 조회
  - 매 정각마다 1시간 내 최댓값과 최솟값, 정각에 해당하는 값을 테이블에 새롭게 저장하고 기존 값은 삭제하여 응답 속도 보장
- 실시간 및 기간 별 BMS, 배터리 정보 조회, 엑셀 다운로드
  - BMS 온도, 배터리 전압, 배터리 잔량 조회
- 이름, 이메일, 연락처 등 의사, 간호사 정보 조회




# 아키텍처



# ERD





# 꽃마리

빅데이터를 활용한  
사용자 취향 기반 꽃 추천 서비스



## 정보

기간 2022.08 ~ 2022.10

도메인 빅데이터 추천

참가 인원 FE 3, BE 3

담당 파트 백엔드(추천), DB, 배포

개발 환경 Python3, Django(DRF), MySQL, Redis, Selenium, Jenkins, Docker

수상 SSAFY 특화 PJT 우수상

## 프로젝트 소개

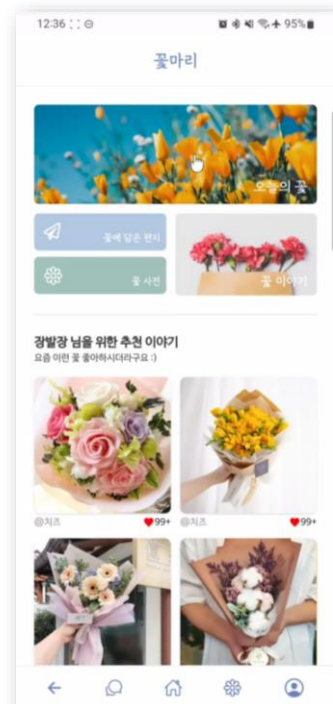
**꽃마리**는 빅데이터를 활용한 사용자 취향 기반 꽃 추천 서비스입니다.

협업 필터링(Collaborative Filtering)과 하이브리드 필터링(Collaborative Filtering & Content Based Filtering)을 사용하여 꽃 이야기와 꽃을 추천합니다.

작성한 편지를 OCR로 텍스트를 인식하고 편지 내용과 유사한 꽃말을 지닌 꽃을 추천합니다.

## 담당 기능

- 하이브리드 필터링을 활용한 가족, 연인, 친구 등 대상 별 꽃 추천
  - 유저 간 태그를 비교한 협업 필터링과 화훼 유통 판매량을 비교한 콘텐츠 기반 필터링을 조합
  - Scikit-learn의 코사인 유사도를 활용하여 유저 간 태그 유사도가 0.3 이상인 경우에만 꽃 분석
  - 화훼 유통 판매량에 따라 꽃을 품목 별로 분류
  - 판매량에 따라 올해:작년:재작년 판매 점수를 6:3:1 비율로 부여
  - 유저 간 태그 유사도와 판매 점수를 7:3 비율로 계산
  - 이미 태그에 담은 꽃, 싫어하는 꽃을 제외하여 18개 꽃 추천
  - 추천한 꽃이 18개 미만인 경우 인기 꽃을 상위부터 추천
- 협업 필터링을 활용한 좋아요 기반 꽃 이야기 게시물 추천
- 편지 내용과 유사한 꽃말을 지닌 꽃 추천
  - KoNLPy를 사용하여 편지 텍스트를 형태소 단위로 분류
  - Scikit-learn의 TF-IDF를 활용하여 중요 단어 추출
  - 편지 텍스트와 유사도가 가장 높은 꽃말을 지닌 꽃 추천





# 아키텍처

```

graph LR
    GitLab --> Jenkins
    Jenkins --> Docker1[Docker Jenkins]
    Docker1 --> Docker2[Docker NGINX]
    Docker2 -- 8080 --> Docker3[Docker Spring Boot]
    Docker3 <--> |8000| Docker4[Docker Django]
    Docker3 --> S3[amazon S3]
    Docker3 <--> |6379| Redis[redis]
    Docker4 <--> |3306| MySQL[MySQL]
    NGINX[NGINX] --> Browser[User]
    Browser --> Docker2
  
```

# ERD

```

    erDiagram
        flower ||--o{ picture : "has"
        flower ||--o{ article : "has"
        flower ||--o{ popular : "has"
        flower ||--o{ hashtag : "has"
        flower ||--o{ subject : "has"
        flower ||--o{ comment : "has"
        flower ||--o{ kind : "has"
        flower ||--o{ flower_like : "has"
        flower ||--o{ tag : "has"
        picture ||--o{ article : "has"
        article ||--o{ popular : "has"
        article ||--o{ hashtag : "has"
        article ||--o{ comment : "has"
        article ||--o{ kind : "has"
        article ||--o{ flower_like : "has"
        article ||--o{ tag : "has"
        popular ||--o{ hashtag : "has"
        popular ||--o{ subject : "has"
        popular ||--o{ comment : "has"
        popular ||--o{ kind : "has"
        popular ||--o{ flower_like : "has"
        popular ||--o{ tag : "has"
        hashtag ||--o{ subject : "has"
        hashtag ||--o{ comment : "has"
        hashtag ||--o{ kind : "has"
        hashtag ||--o{ flower_like : "has"
        hashtag ||--o{ tag : "has"
        subject ||--o{ comment : "has"
        subject ||--o{ kind : "has"
        subject ||--o{ flower_like : "has"
        subject ||--o{ tag : "has"
        comment ||--o{ kind : "has"
        comment ||--o{ flower_like : "has"
        comment ||--o{ tag : "has"
        kind ||--o{ flower_like : "has"
        kind ||--o{ tag : "has"
        flower_like ||--o{ tag : "has"
        tag ||--o{ kind : "has"
        tag ||--o{ flower_like : "has"
        user ||--o{ article : "has"
        user ||--o{ comment : "has"
        user ||--o{ flower_like : "has"
        user ||--o{ tag : "has"
        user ||--o{ kind : "has"
        user ||--o{ flower : "has"
        user ||--o{ popular : "has"
        user ||--o{ hashtag : "has"
        user ||--o{ subject : "has"
        user ||--o{ comment : "has"
        user ||--o{ kind : "has"
        user ||--o{ flower_like : "has"
        user ||--o{ tag : "has"
        kind ||--o{ flower_like : "has"
        kind ||--o{ tag : "has"
        flower_like ||--o{ tag : "has"
        tag ||--o{ kind : "has"
        tag ||--o{ flower_like : "has"
    
```

The ERD illustrates the following tables and their attributes:

- flower**: flower\_id BIGINT, content VARCHAR(255), date VARCHAR(255), image VARCHAR(255), language VARCHAR(255), name VARCHAR(255).
- picture**: picture\_id BIGINT, image VARCHAR(255), article\_id BIGINT.
- article**: article\_id BIGINT, content TEXT, date DATETIME(6), title VARCHAR(255), user\_id BIGINT.
- popular**: popular\_id BIGINT, popular\_date DATE, article\_id BIGINT.
- hashtag**: hashtag\_id BIGINT, article\_id BIGINT, subject\_id BIGINT.
- subject**: subject\_id BIGINT, flower\_language VARCHAR(255), subject\_name VARCHAR(255).
- comment**: comment\_id BIGINT, content TEXT, date DATETIME(6), article\_id BIGINT.
- user**: user\_id BIGINT, age INT, birthday DATE, email VARCHAR(255), is\_active BIT(1), login\_count BIGINT, name VARCHAR(255), profile\_image VARCHAR(255), sex BIT(1).
- kind**: kind\_id BIGINT, flower\_image VARCHAR(255), kind\_name VARCHAR(255), subject\_id BIGINT.
- flower\_like**: flower\_like\_id BIGINT, kind\_id BIGINT, tag\_id BIGINT, user\_id BIGINT.
- tag**: tag\_id BIGINT, dear VARCHAR(255).

Relationships are defined as follows:

- flower** to **picture**, **article**, **popular**, **hashtag**, **subject**, **comment**, **kind**, **flower\_like**, and **tag**: One-to-many (1 to N).
- picture** to **article**: One-to-many (1 to N).
- article** to **popular**, **hashtag**, **comment**, **kind**, **flower\_like**, and **tag**: One-to-many (1 to N).
- popular** to **hashtag**, **comment**, and **kind**: One-to-many (1 to N).
- hashtag** to **subject**, **comment**, and **kind**: One-to-many (1 to N).
- subject** to **comment**, **kind**, **flower\_like**, and **tag**: One-to-many (1 to N).
- comment** to **kind**, **flower\_like**, and **tag**: One-to-many (1 to N).
- kind** to **flower\_like** and **tag**: One-to-many (1 to N).
- flower\_like** to **tag**: One-to-many (1 to N).
- tag** to **kind**: One-to-many (1 to N).
- user** to **article**, **comment**, **flower\_like**, **tag**, **kind**, **flower**, **popular**, **hashtag**, **subject**, **comment**, **kind**, **flower\_like**, and **tag**: One-to-many (1 to N).

# Plantinum

---

스마트 화분 및 식물 거래 플랫폼





## 정보

기간 2022.07 ~ 2022.08

도메인 웹 IoT

참가 인원 HW 2, FE 2, BE 1

담당 파트 백엔드, DB

개발 환경 Python3, Django(DRF), MySQL, Redis, S3

수상 SSAFY 공통 PJT 우수상

## 프로젝트 소개

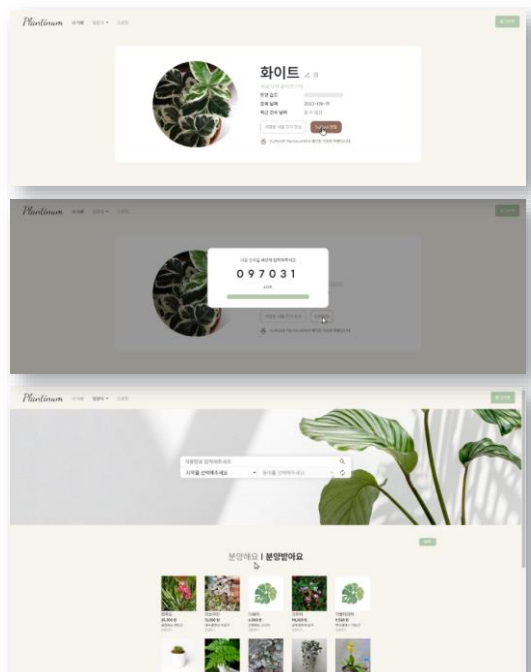
**PLANTinum**은 IoT 기반 식물 자동 관리 및 웹 기반 식물 거래 서비스입니다.

스마트 화분 'Supool'을 통해 자동으로 식물을 관리하고 웹 페이지와 연동하여 식물 정보를 조회할 수 있습니다. 실내 정원 식물의 관수 방법, 토양 정보, 특별 관리 방법을 조회할 수 있습니다.

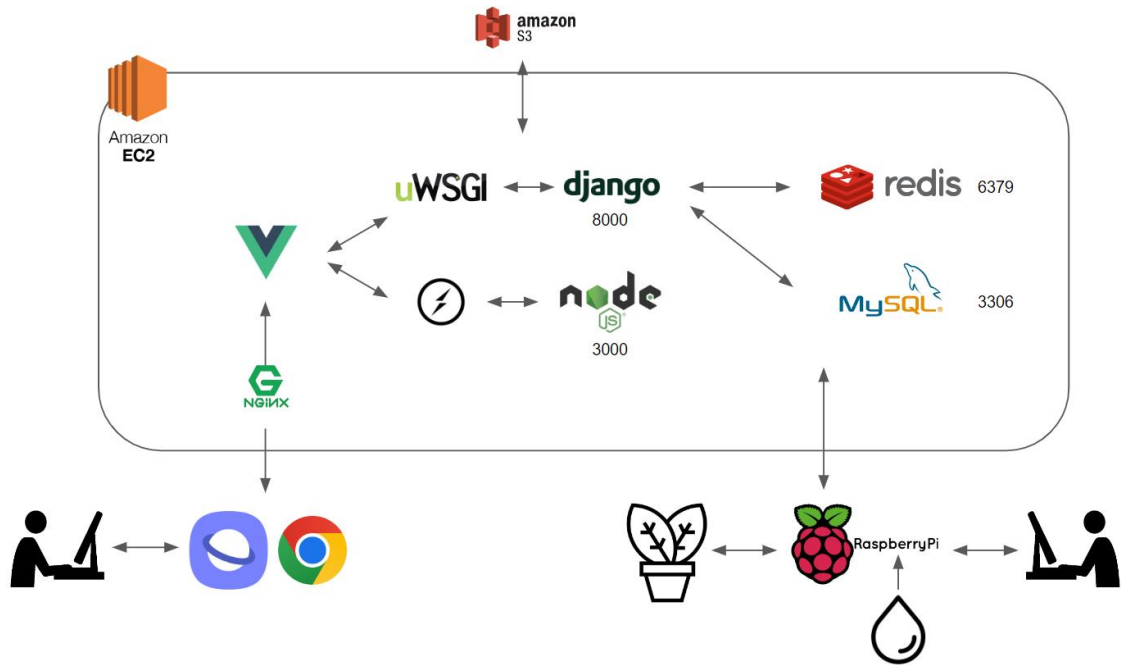
식물 거래 플랫폼 '잎팔이' 서비스를 통해 유저들 간에 채팅이 가능하며 식물을 거래할 수 있습니다.

## 담당 기능

- 가입 및 로그인 등의 회원 관리
- 회원 가입 시 유니크한 닉네임 자동 생성
  - 리스트에서 랜덤으로 추출한 단어들과 랜덤 숫자를 조합
- 사용자 프로필 조회 및 수정
- 내 식물 정보 등록, 수정 및 토양 습도, 관수 날짜 정보 조회
- 내 식물 IoT 기기와 웹 연결
  - Redis TTL을 활용
  - DB에 존재하지 않는 OTP(일회용 비밀번호)를 생성하고 60초 뒤에 자동으로 삭제
  - 생성 후 60초 내에 발급받은 OTP 재확인 가능
- 실내 정원용 식물의 관리 정보 조회
  - 농촌진흥청 농사로의 공공 데이터를 활용하여 관수 방법, 토양 정보, 특별 관리 방법 수집
- 식물 거래 글 작성 및 조회, 페이지네이션
- 식물 이름, 지역, 동네 별 거래 글 필터



# 아키텍처



# ERD

