포팅메뉴얼

사용한 jvm, webserver

- **azul/zulu-openjdk:17 , nginx , intellij , Visual Studio Code**

빌드시 사용되는 환경변수

```
spring:
  datasource:
    url: jdbc:mysql://j10C102.p.ssafy.io:3306/fullerting?serv
    username: fulldba
    password: full202403110344
  jackson:
    time-zone: Asia/Seoul

  data:
    redis:
      host: j10C102.p.ssafy.io
      port: 6379

  security:
    oauth2:
      client:
        registration:
          google:
            client-id: 648265936949-kqbendno25msh4hklurcvtpkv
            client-secret: GOCSPX-Mm16KeYp85dNktb8DCOy9BL2QAE
            redirect-uri: https://j10c102.p.ssafy.io/login/oa
#            redirect-uri: http://localhost:8080/login/oauth2
            scope: profile, email
```

```
  jwt:
    access: whatwehavetowantinalifeismentalsatisfactionnotmat
    refresh: youforgetthisyoualwaysfeelfailevenifyouwillgetev
    accesstime: 360000000  # 10H -> 0하나 더 붙임...
    refreshtime: 604800000 # 7Days

  servlet:
    multipart:
      max-file-size: 10MB
      max-request-size: 110MB

  jpa:
    show-sql: false  # jpa 쿼리문 출력 설정
    properties:
      hibernate:
        format_sql: true
    open-in-view: false  # SSE 설정시 필요 -> DB Connection 닫아


cloud:
  aws:
    credentials:
      access-key: AKIAW3MEAWFWGVMIB3VX
      secret-key: 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
    region:
      static: ap-northeast-2
    s3:
      bucket: fullerting-s3-v2
      stack:
        auto: false

springdoc:
  swagger-ui:
    path: /api/swagger-ui.html
```

```
logging:
  level:
    org:
      springframework:
        security:
          web:
          FilterChainProxy: DEBUG


fcm:
  service-account-file: key/fullerting-fcm-firebase-adminsdk-
  topic-name: alarm
  project-id: fullerting-fcm

#      hibernate:
#        type:
#          descriptor:
#            sql: trace


server:
  port: 8080
  servlet:
    context-path: /
  websocket:
    port: 9090

#  tomcat:
#    remoteip:
#      protocol-header: x-forwarded-proto
```

- **datasource: 데이터베이스 연결 정보를 설정합니다. MySQL 데이터베이스에 대한 URL, 사용자 이름 및 암호가 포함되어 있습니다.**
- **jackson: Jackson 라이브러리 설정으로, 시간대를 Asia/Seoul로 지정합니다.**

- **redis: Redis 데이터베이스 연결 정보를 설정합니다. 호스트 및 포트가 여기에 포함되어 있습니다. security.oauth2: OAuth 2.0 클라이언트 등록 정보를 설정합니다.**
- **구글 클라이언트 ID, 클라이언트 비밀키 및 리디렉션 URI가 포함되어 있습니다.**
- **jwt: JWT(JSON Web Token) 설정으로, 액세스 토큰, 리프레시 토큰 및 만료 시간이 포함되어 있습니다.**
- **servlet.multipart: 서블릿 멀티파트 파일 업로드 제한을 설정합니다.**
- **jpa: JPA 설정으로, SQL 출력 여부와 Hibernate 설정이 포함되어 있습니다.**
- **cloud.aws: AWS(Amazon Web Services) 자격 증명 및 지역 정보를 설정합니다. S3 버킷 이름과 스택 설정이 여기에 포함됩니다.**
- **springdoc.swagger-ui: Swagger UI 경로를 설정합니다.**
- **logging.level: 로깅 레벨을 설정합니다. 여기에서는 Spring Security의 FilterChainProxy에 대한 로그 레벨을 DEBUG로 설정합니다.**
- **fcm: Firebase Cloud Messaging(Firebase 알림) 설정으로, 서비스 계정 파일, 토픽 이름 및 프로젝트 ID가 포함되어 있습니다. server: 서버 설정으로, 포트 번호 및 서블릿 컨텍스트 경로를 설정합니다.**

```
VITE_REACT_APP_WSS_URL=wss://j10c102.p.ssafy.io/api/ws
VITE_REACT_APP_SSE_URL=https://j10c102.p.ssafy.io/api/v1/noti
VITE_REACT_APP_API_URL=https://j10c102.p.ssafy.io/api/v1

# VITE_REACT_APP_API_URL=http://localhost:8080/v1
# VITE_REACT_APP_WSS_URL=ws://localhost:8080/ws
# VITE_REACT_APP_SSE_URL=http://localhost:8080/v1/noti/pub
```

VITE_REACT_APP_WSS_URL=wss://j10c102.p.ssafy.io/api/ws

Websocket 통신을 위한 url 입니다.

VITE_REACT_APP_SSE_URL=https://j10c102.p.ssafy.io/api/v1/noti/pub

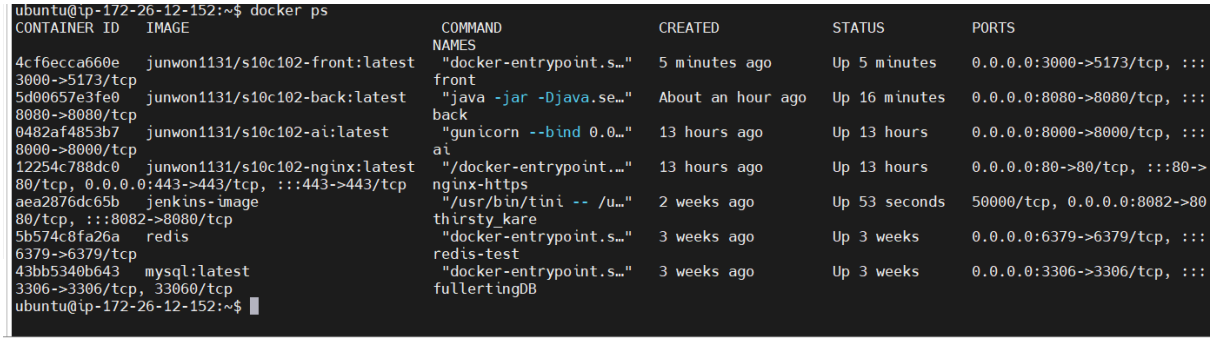
알림 서비스를 위한 sse url

VITE_REACT_APP_API_URL=https://j10c102.p.ssafy.io/api/v1 nginx 에서 설정해둔 백엔드 컨테이너에 내부 접근을 위한 url 입니다.

배포시 특이사항

	port 번호
Jenkins	8082
front	3000→5173
back	8080
redis	6379
AI	8000
Nginx	80

젠킨스 실행시 -v 로 볼륨매핑 하고 실행.




```
version: '3.8'

services:

  nginx:
    image:   junwon1131/s10c102-nginx:latest
    container_name: nginx-https
    build:
      context: nginx
      dockerfile: Dockerfile

    ports:
      - "80:80"
```

```
      - "443:443"
    volumes:
#      - ./nginx/nginx.conf:/etc/nginx/nginx.conf
      - /etc/letsencrypt/live/j10c102.p.ssafy.io/fullchain.pe
      - /etc/letsencrypt/live/j10c102.p.ssafy.io/privkey.pem:
    restart: unless-stopped
    depends_on:
      - back
      - front
    networks:
      - jenkins-network


  ai:
    container_name: ai
    image: junwon1131/s10c102-ai:latest
    build:
      context:  ../A.I
      dockerfile: Dockerfile
    restart: unless-stopped
    ports:
      - "8000:8000"
    networks:
      - jenkins-network


  front:
    container_name: front
    image: junwon1131/s10c102-front:latest
    build:
      context: ../frontend
      dockerfile: Dockerfile
    restart: unless-stopped
    ports:
      - "3000:5173"
    networks:
      - jenkins-network
    depends_on:
```

```
      - back

  back:
    container_name: back
    image: junwon1131/s10c102-back:latest
    build:
      context: .
      dockerfile: Dockerfile
    restart: unless-stopped
    ports:
      - "8080:8080"
    networks:
      - jenkins-network
#    logging:
#      driver: "json-file"
#      options:
#        max-size: "10m"



networks:
  jenkins-network:
    external: true
```

docker compose . yml 을 활용하여 Jenkins build 과정에서 한번에 필요한 파일들의 이미지를 빌드후 compose down, up 을 통해 컨테이너 실행하였습니다.

Erd 에 활용되는 주요계정 및 프로퍼티가 정의된 파일 목록

	URL	`jdbc:mysql://j10C102.p.ssafy.io:3306/fullerting?serverTimezone=Asia/Seoul`
	`username`	`fulldba`
	`password`	`full202403110344`

프로젝트에서 사용하는 외부 서비스 정보를 정리한 문서

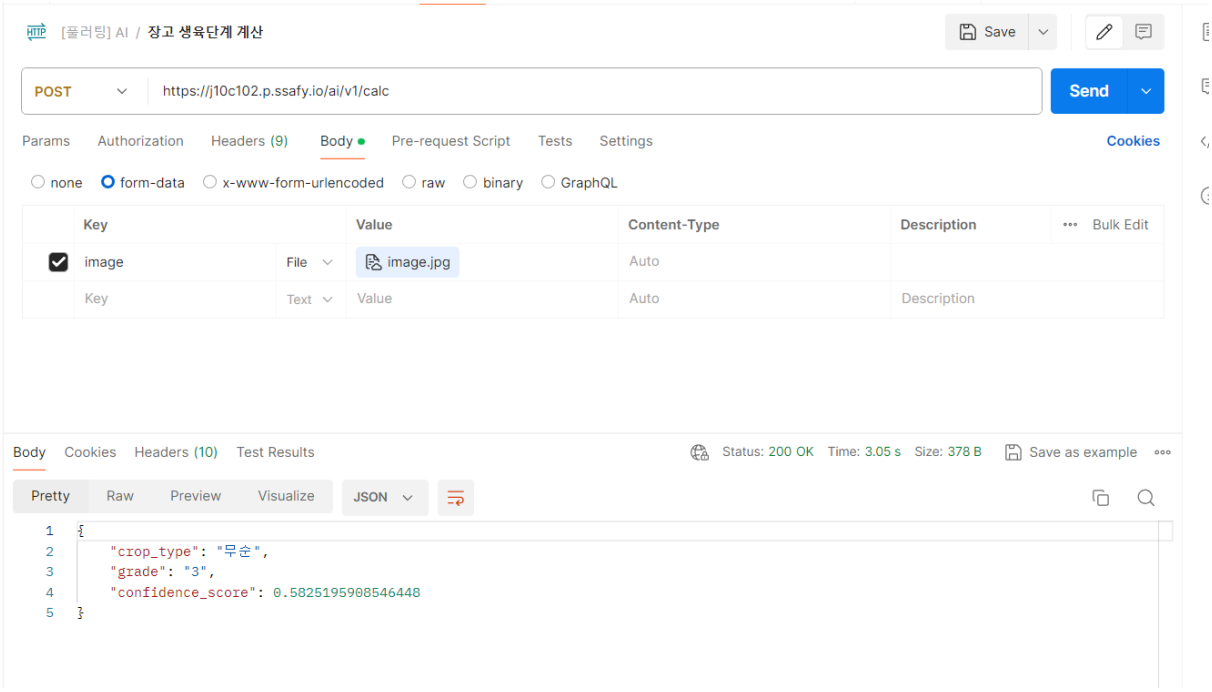
AI (장고서버)

```
from django.shortcuts import render
from django.http import JsonResponse
from django.views.decorators.csrf import csrf_exempt
from keras.models import load_model  # TensorFlow is required
from PIL import Image, ImageOps  # Install pillow instead of P
import numpy as np
import os


cur_dir = os.path.dirname(os.path.abspath(__file__))
model_path = os.path.join(cur_dir, '..', 'AImodel','keras_mod
label_path = os.path.join(cur_dir, '..', 'AImodel','labels.tx
# image_path = os.path.join(cur_dir, '..', 'AImodel','moosoon

@csrf_exempt
def calc_ai(request):
    if request.method == 'POST' and request.FILES['image']:
        # Return To Spring

        # Disable scientific notation for clarity
        np.set_printoptions(suppress=True)

        # Load the model
        model = load_model(model_path, compile=False)
        # Load the labels
        class_names = open(label_path, "r", encoding="utf-8")

        # Create the array of the right shape to feed into th
        # The 'length' or number of images you can put into t
        # determined by the first position in the shape tuple
        data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.fl
```

```
        # 이미지 파일 받기
        image_file = request.FILES['image']

        # Replace this with the path to your image
        image = Image.open(image_file).convert("RGB")

        # resizing the image to be at least 224x224 and then c
        size = (224, 224)
        image = ImageOps.fit(image, size, Image.Resampling.LA

        # turn the image into a numpy array
        image_array = np.asarray(image)

        # Normalize the image
        normalized_image_array = (image_array.astype(np.float

        # Load the image into the array
        data[0] = normalized_image_array

        # Predicts the model
        prediction = model.predict(data)
        index = np.argmax(prediction)
        class_name = class_names[index]
        confidence_score = prediction[0][index]

        # Print prediction and confidence score
        # print("Class:", class_name[2:], end="")
        # print("Confidence Score:", confidence_score)

        response_data = {
            'crop_type': class_name.split()[1],
            'grade': class_name.split()[2],
            'confidence_score': float(confidence_score)
        }

        return JsonResponse(response_data)

    else:
```

```
        return JsonResponse({'error': 'This method is not all
```

post 형식을 통해 이미지를 입력하면 그의 결과로 작물의 종류와 생육단계를 파악할수 있습니다.



구글 로그인

callbackurl 을

`HttpServletResponse` 값에 넣어줘서 해당 url 에 접근이 가능하게 하였고,

디비에 있는 사용자의 이메일이 리소스 서버에서 제공한 oauth2user 의 이름과 같으면

쿠키에 인증객체를 통해 만든 jwt 도 같이 담아서 보내줍니다.

```
package com.ssafy.fullerting.security.handler;

import com.fasterxml.jackson.databind.ObjectMapper;
import com.ssafy.fullerting.global.utils.MessageUtils;
import com.ssafy.fullerting.security.model.dto.response.Issue
```

```
import com.ssafy.fullerting.security.service.TokenService;
import com.ssafy.fullerting.user.model.entity.CustomUser;
import com.ssafy.fullerting.user.repository.UserRepository;
import com.ssafy.fullerting.user.service.UserService;
import jakarta.servlet.ServletException;
import jakarta.servlet.http.Cookie;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.security.core.Authentication;
import org.springframework.security.oauth2.core.user.OAuth2Us
import org.springframework.security.web.authentication.Simple
import org.springframework.stereotype.Component;

import java.io.IOException;
import java.net.URLEncoder;
import java.nio.charset.StandardCharsets;

@Slf4j
@RequiredArgsConstructor
@Component
public class OAuthSuccessHandler extends SimpleUrlAuthenticat
    private final UserRepository userRepository;
    private final UserService userService;
    private final TokenService tokenService;
    private final ObjectMapper objectMapper;

    @Override
    public void onAuthenticationSuccess(HttpServletRequest re
        //http://localhost:8080/login 에서 확인
        OAuth2User oAuth2User = (OAuth2User) authentication.g
        log.info("Oauth 사용자 정보 확인 : {}", oAuth2User.toStr

        // 리디렉션할 URL과 토큰 발급
        String redirectUrl = "https://j10c102.p.ssafy.io/auth
        try {
            userRepository.findByEmail(oAuth2User.getName()).
```

```
                    user -> {
                        // 사용자가 이미 존재하는 경우
                        try {
//                            redirectToCallbackWithToken(res
                            addTokenToCookies(response,tokenS
                            response.sendRedirect(redirectUrl
                        } catch (IOException e) {
                            throw new RuntimeException(e);
                        }
                    },
                    () -> {
                        // 새로운 사용자 등록
                        registerNewUser(oAuth2User);
                        try {
//                            redirectToCallbackWithToken(res
                            addTokenToCookies(response,tokenS
                            response.sendRedirect(redirectUrl
                        } catch (IOException e) {
                            throw new RuntimeException(e);
                        }
                    }
            );
        } catch (Exception e) {
            log.error("인증 성공 후 처리 중 오류 발생", e);
        }
    }

    private void registerNewUser(OAuth2User oAuth2User) {
        // 새 사용자 등록 로직
        CustomUser customUser = CustomUser.of(oAuth2User);
        customUser.setPassword("DummyPasswordeoar!@3");
        userService.registOauthUser(customUser);
    }

    private void redirectToCallbackWithToken(HttpServletRespo
        String urlWithToken = String.format("%s?accessToken=%
                redirectUrl,
                URLEncoder.encode(issuedToken.getAccessToken(
```

```
                URLEncoder.encode(issuedToken.getRefreshToken
        response.sendRedirect(urlWithToken);
    }

    private void addTokenToCookies(HttpServletResponse respon
        // 액세스 토큰을 쿠키에 저장
        Cookie accessTokenCookie = new Cookie("accessToken",
        accessTokenCookie.setDomain("j10c102.p.ssafy.io"); //
        accessTokenCookie.setPath("/"); // 쿠키를 전체 도메인에서
        accessTokenCookie.setHttpOnly(httpOnly); // JavaScrip
        accessTokenCookie.setSecure(true); // HTTPS를 사용하는
        accessTokenCookie.setMaxAge(1 * 24 * 60 * 60); // 쿠키

        // 리프레시 토큰을 쿠키에 저장
        Cookie refreshTokenCookie = new Cookie("refreshToken"
        refreshTokenCookie.setDomain("j10c102.p.ssafy.io"); /
        refreshTokenCookie.setPath("/"); // 쿠키를 전체 도메인에서
        refreshTokenCookie.setHttpOnly(httpOnly); // JavaScri
        refreshTokenCookie.setSecure(true); // HTTPS를 사용하는
        refreshTokenCookie.setMaxAge(1 * 24 * 60 * 60); // 쿠키

        // 응답에 쿠키 추가
        response.addCookie(accessTokenCookie);
        response.addCookie(refreshTokenCookie);
    }


    // JSON 생성
    private void writeResponse(HttpServletResponse response,
        try {
            String json = objectMapper.writeValueAsString(tok
            response.setContentType("application/json;charset
            response.setStatus(HttpServletResponse.SC_OK);
            response.getWriter().write(json);
        } catch (IOException e) {
            log.error("Response write error", e);
        }
```

```
    }
}
```

덤프 파일 최신

시연 시나리오

1. 메인페이지

네비게이션 바(화면 하단에 위치)

- 첫 번째 버튼을 누르면 메인페이지로 이동
- 두 번째 버튼을 누르면 작물 거래 페이지로 이동
- 세 번째 버튼을 누르면 커뮤니티로 이동
- 네 번째 버튼을 누르면 작물 일지 페이지로 이동
- 다섯 번째 버튼을 누르면 마이페이지로 이동

이외 기능

- 오른쪽 상단에 있는 버튼 중 왼쪽 버튼을 클릭시 채팅방 목록 페이지로 이동
- 오른쪽 상단에 있는 버튼 중 오른쪽 버튼을 클릭시 알림함으로 이동
- 왼쪽 상단에 작물 일지 박스를 클릭시 작물일지 페이지로 이동한
- 단, 작성 해놓은 작물일지가 없을 시 이 박스는 랜더링 되지 않는다.
- 작물 거래하기 아래 있는 박스를 클릭시 작물 거래하기 페이지로 이동
- 박스 옆 화살표 클릭시 다음 게시물 확인 가능
- 작물 거래하기 옆에 있는 캐릭터 클릭시 작물 거래하기 페이지로 이동
- 커뮤니티 바로가기 옆 캐릭터 클릭시 커뮤니티 페이지로 이동
- 각각의 게시글 클릭시 커뮤니티 페이지로 이동
- 게시글을 오른쪽 혹은 왼쪽으로 드래그해서 다른 게시물 확인 가능
- 지도 보기 버튼을 클릭하면 전국의 텃밭 정보를 담은 지도 페이지로 이동

2. 로그인 및 회원가입 페이지

- 로그인
  - 아이디 : test1@test.com
  - 비밀번호 : abcd1234!
- 회원가입 버튼을 누르면 회원가입 페이지로 이동
  - 이메일, 비밀번호, 비밀번호 확인, 닉네임을 입력 후 확인을 누르면 회원가입 성공
- 구글 로고를 클릭시 구글 계정으로 로그인

3. 작물거래

3.1 작물거래 페이지

- 동네인증
  - 동네인증 버튼을 누르면 동네인증 페이지로 이동
  - 현재 위치를 기반으로 위치 추적
  - 주소 검색으로 동네인증하기 버튼을 클릭시 주소 검색 페이지로 이동
  - 확인 버튼 클릭시 동네 저장 완료
- 네비게이션 바
  - 전체 클릭시 모든 카테고리의 작물 거래 게시글 확인 가능
  - 제안 클릭시 제안 카테고리의 작물 거래 게시글 확인 가능
  - 거래 클릭시 거래 카테고리의 작물 거래 게시글 확인 가능
  - 나눔 클릭시 나눔 카테고리의 작물 거래 게시글 확인 가능
  - 관심 클릭시 관심 카테고리(좋아요 눌러진 게시글)의 작물 거래 게시글 확인 가능
- 각각의 게시글 클릭시 각 게시글에 해당하는 상세조회 페이지로 이동
  - 각 카테고리 별 다른 화면이 랜더링 됩니다.
- 게시글 사진 오른쪽 하단 하트를 클릭하면 '좋아요' 활성화/비활성화
- 오른쪽 하단에 +버튼 클릭하면 작물거래 게시글 작성하는 페이지로 이동

3.2 작물거래 게시글 작성 페이지

- 필수항목(제목, 거래 방법, 거래 단위, 시작가, 거래 희망 장소, 내용, 사진 등록을 작성하여 등록하기 버튼을 누르면 작물 거래 게시갈 작성이 된다
- 책 그림의 버튼을 클릭하면 작물 일지를 선택할수 있다
- 카메라 그림의 버튼을 클릭하면 사진을 선택할 수 있다

3.3 작물거래 상세페이지

- 작물거래 방식이 '제안' 일 때(구매자)
  - 화면 왼쪽 상단에 화살표 클릭 시 이전페이지로 이동
  - 제목 옆 하트 클릭 시 '좋아요' 활성화/비활성화

    - 제안하기 버튼 클릭 시 제안하기 페이지로 이동
- 작물거래 방식이 '제안' 일 때(판매자)
    - 화면 왼쪽 상단에 화살표 클릭 시 이전페이지로 이동
    - 화면 오른쪽 상단에 쓰레기통 클릭시 게시글 삭제 confirm창 랜더링
    - 제목 옆 하트 클릭 시 '좋아요' 활성화/비활성화
    - 제안목록 버튼 클릭 시 제안목록 확인 페이지로 이동
- 작물거래 방식이 '거래' 일 때(구매자)
    - 화면 왼쪽 상단에 화살표 클릭 시 이전페이지로 이동
    - 제목 옆 하트 클릭 시 '좋아요' 활성화/비활성화
    - 채팅하기 클릭 시 채팅방 활성화 및 해당 게시글의 채팅 페이지로 이동
- 작물거래 방식이 '거래' 일 때(판매자)
    - 화면 왼쪽 상단에 화살표 클릭 시 이전페이지로 이동
    - 화면 오른쪽 상단에 쓰레기통 클릭 시 게시글 삭제 confirm창 랜더링
    - 제목 옆 하트 클릭 시 '좋아요' 활성화/비활성화
    - 채팅방으로 이동하기 클릭 시 자신의 채팅방 목록 페이지로 이동
- 작물거래 방식이 '나눔' 일 때(구매자)
    - 화면 왼쪽 상단에 화살표 클릭 시 이전페이지로 이동
    - 제목 옆 하트 클릭 시 '좋아요' 활성화/비활성화
    - 채팅하기 클릭 시 채팅방 활성화 및 해당 게시글의 채팅 페이지로 이동
- 작물거래 방식이 '나눔' 일 때(판매자)
    - 화면 왼쪽 상단에 화살표 클릭 시 이전페이지로 이동
    - 화면 오른쪽 상단에 쓰레기통 클릭 시 게시글 삭제 confirm창 랜더링
    - 제목 옆 하트 클릭 시 '좋아요' 활성화/비활성화
    - 채팅하기 클릭 시 자신의 채팅방 목록 페이지로 이동

3.4 제안 목록 확인 페이지

- 왼쪽 상단에 화살표 클릭 시 이전페이지로 이동
- 제안 목록 중 하나를 클릭하면 제안 당사자와의 채팅방 생성 및 채팅 화면으로 이동

3.5 제안하기 페이지

- 왼쪽 상단에 화살표 클릭 시 이전페이지로 이동
- 화면 하단의 입력란에 최고가 보다 높은 금액 작성 후 종이비행기 버튼을 누르면 제안 성공

3.6 채팅 페이지

- 왼쪽 상단에 화살표 클릭 시 이전페이지로 이동
- 화면 하단의 입력란에 채팅 작성 후 종이비행기 버튼을 누르면 채팅 보내기 성공

3.7 채팅 목록 페이지

- 왼쪽 상단에 화살표 클릭 시 이전페이지로 이동
- 채팅 목록 중 한개를 클릭하면 해당하는 채팅방으로 이동

4. 작물일지

4.1 작물일지

4.1.1 작물일지 생성

- 4번째 새싹 모양의 메뉴를 선택한다.
- 우측 하단 + 버튼을 클릭한다.
- 작물(무순, 토마토, 상추) 선택한다.
- 시작일 선택한다.
- 작물 닉네임(8글자 제한)
- 확인버튼을 누르면 작물일지가 생성된다.
- 작물일지 페이지에서 작물 사진, 닉네임, 단계, 기간을 볼 수 있다.

4.1.2 작물일지 수정

- 작물일지를 클릭한다.
- 우측 상단 연필 모양의 수정 버튼을 클릭한다.
- 작물 종류는 수정 불가하다.
- 시작일을 수정할 수 있다.
- 작물 닉네임을 수정할 수 있다.

- 확인버튼을 누르면 작물일지가 수정된다.

4.1.3 작물일지 삭제

- 작물일지를 클릭한다.
- 우측 상단 연필 모양의 휴지통 버튼을 클릭한다.
- 정말로 삭제하시겠습니까? 모달이 뜨고 확인 버튼을 누르면 작물일지가 삭제된다.
- 정말로 삭제하시겠습니까? 모달이 뜨고 취소 버튼을 누르면 작물일지가 유지된다.

4.2 다이어리

4.2.1 다이어리 생성

- 우측 하단 + 버튼을 클릭한다.
- 버튼 위의 아래 책 모양 버튼을 클릭한다.
- 날짜를 선택한다.(작물일지 시작일부터 오늘 날짜까지만 선택 가능)
- 제목을 작성한다.(20글자 제한)
- 내용을 작성한다.(300글자 제한)
- 사진을 선택한다.(최대 5장 제한) 사진은 선택 사항으로 필수 항목이 아니다. x 버튼을 누르면 사진을 삭제할 수 있다.
- 확인버튼을 누르면 다이어리가 생성된다.
- 다이어리 메뉴를 선택하면 모든 다이어리 목록을 볼 수 있다.

4.2.2 다이어리 상세보기

- 다이어리를 클릭하면 작성 날짜, 제목, 사진, 내용을 볼 수 있다.
- 돌아가기 버튼을 클릭하면 해당 작물일지의 작물일기 화면으로 돌아간다.

4.2.3 다이어리 수정

- 다이어리 상세보기를 한다.
- 우측 상단 연필 모양의 수정 버튼을 클릭한다.
- 날짜를 수정할 수 있다.
- 제목을 수정할 수 있다.
- 내용을 수정할 수 있다.
- 사진을 수정할 수 있다.
- 확인버튼을 누르면 다이어리가 수정된다.

4.2.4 다이어리 삭제

- 다이어리 상세보기를 한다.
- 우측 상단 휴지통 모양의 수정 버튼을 클릭한다.
- 정말로 삭제하시겠습니까? 모달이 뜨고 확인 버튼을 누르면 다이어리가 삭제된다.
- 정말로 삭제하시겠습니까? 모달이 뜨고 취소 버튼을 누르면 다이어리가 유지된다.

4.3 작물꿀팁

- 작물 꿀팁 메뉴를 선택하면 해당 작물의 단계 별 꿀팁을 볼 수 있다.

4.4 물주기

4.4.1 물주기 생성

- 우측 하단 + 버튼을 클릭한다.
- 버튼 위의 아래 물방울 모양 버튼을 클릭한다.
- 날짜를 선택한다.(작물일지 시작일부터 오늘 날짜까지만 선택 가능)
- 확인버튼을 누르면 물주기가 생성된다.

4.4.2 물주기 삭제

- 다이어리에서 삭제하고 싶은 날짜의 물주기 박스를 클릭한다.
- 삭제하시겠습니까? 모달이 뜨고 확인 버튼을 누르면 물주기 박스가 삭제된다.
- 삭제하시겠습니까? 모달이 뜨고 취소 버튼을 누르면 물주기 박스가 유지된다.

4.5 작물 인식하기

- 작물일기에서 작물인식하기 버튼을 클릭한다.
- 박스를 클릭한다.
- (모바일)카메라로 사진을 촬용한다. (웹)사진 파일을 선택한다.
- x 버튼을 클릭하면 사진을 삭제하고 재등록할 수 있다.
- 확인 버튼을 누르면 로딩중 화면이 나온다.
- 약 5초 후 작물인식 성공 시 {작물 종류} {작물 단계}단계로 업그레이드 완료!!!! 모달이 뜨고 작물 단계가 업그래이드 된다.
- 작물인식 실패 시 해당 작물이 아니거나 단계 및 정확도가 낮습니다. 다시 촬영해 주세요. 모달이 뜬다.

- 작물의 마지막 단계일 경우 [{뱃지이름}] 뱃지를 획득하였습니다! 모달이 뜨고 뱃지를 획득한다. 획득한 뱃지는 마이페이지에서 볼 수 있다.
- 작물인식 성공 시 작물 프로필의 수확까지 남은 일 수가 업데이트 된다.

4.6 수확하기

- 작물일기에서 수확하기 버튼을 클릭한다.
- 정말로 수확하시겠습니까? 모달이 뜨고 확인 버튼을 누르면 수확이된다. 수확하기 버튼이 회색으로 바뀌고 클릭할 수 없게된다.
- 정말로 수확하시겠습니까? 모달이 뜨고 취소 버튼을 누르면 수확이 되지 않는다.
- 수확한 경우 작물 프로필에 수확 표시와 함께 작물을 키운 날짜가 표시된다. 작물일지의 작물 프로필에서도 수확 표시를 확인할 수 있다.
- 수확한 경우 우측 하단의 +버튼을 클릭하면 책 모양 버튼의 다이어리 작성만 가능하다. (물주기 기능 불가)

4.7 작물 프로필

- 작물일기 메인 상단에 작물 사진, 닉네임, 작물 단계, 기간, 수확까지 남은 일 수 정보가 출력된다.
  수확까지 남은 일 수는 작물 인식 후 단계 별로 계산되어 출력된다.

5. 커뮤니티

5.1 커뮤니티

5.1.1 게시물 목록

- 3번째 새싹 말풍선 메뉴를 선택한다.
- 상단의 메뉴 버튼을 클릭하면 전체, 자유게시판, 작물소개, 꿀팁공유, 텃밭요리 별로 필터링하여 게시물 목록이 출력된다.

5.1.2 게시물 작성

- 우측 하단 + 버튼을 클릭한다.
- 카테고리(자유게시판, 작물소개, 텃밭요리, 꿀팁공유)를 선택한다.
- 제목을 작성한다.(20글자 제한)
- 본문을 작성한다.(500글자 제한)
- 사진을 선택한다.(최대 5장 제한) x 버튼을 누르면 사진을 삭제할 수 있다.

- 확인버튼을 누르면 게시물이 생성된다.
- 커뮤니티 페이지에서 게시물 목록을 확인할 수 있다.

5.1.3 게시물 상세보기

- 게시물을 클릭하면 제목, 본문, 작성자, 시간, 좋아요 개수, 댓글 개수, 댓글을 확인할 수 있다.
- 본문 하단의 하트 버튼을 클릭하면 하트가 초록색으로 바뀌고 관심 게시물로 등록된다.
- 관심 게시물인 경우 하트 버튼을 클릭하면 관심 게시물이 해제된다.

5.1.4 게시물 삭제

- 내가 작성한 게시물일 경우 게시물 상세보기에서 우측 상단의 휴지통 버튼을 클릭하면 정말로 삭제하시겠습니까? 문구가 출력되고 확인 버튼을 누르면 게시물이 삭제된다.
- 게시물이 삭제되면 게시물 페이지로 이동한다.

5.1.5 댓글 작성

- 게시물 상세보기에서 하단의 댓글을 입력하고 등록 버튼을 누르면 댓글을 작성할 수 있다.

5.1.6 댓글 삭제

- 내가 쓴 댓글일 경우 댓글 삭제 버튼이 출력된다.
- 댓글 삭제 버튼 클릭 시 정말로 삭제하시겠습니까? 모달이 뜨고 확인 버튼을 누르면 댓글이 삭제된다.

6. 마이페이지

6.1 계정

6.1.1 내 정보 확인

- 5번째 사람 메뉴를 클릭한다.
- 맨 위의 프로필 메뉴를 클릭한다.
- 닉네임, 이메일, 비밀번호 정보를 확인할 수 있다.

6.1.2 내 정보 수정

- 프로필 수정에서 사진을 클릭하고 이미지 파일을 선택하면 프로필을 수정할 수 있다.
- 닉네임란에 수정할 닉네임을 작성후 하단의 수정 버튼 클릭 시 닉네임이 수정된다.

6.1.3 로그아웃

- 로그아웃 메뉴를 클릭하면 정말로 로그아웃하시겠습니까? 모달이 뜨고 확인 버튼 클릭 시 로그아웃 할 수 있다.
- 로그아웃 성공 시 메인 페이지로 이동한다.

6.2 마이페이지 메뉴

6.2.1 뱃지

- 보유 뱃지 메뉴를 클릭하면 내가 보유한 뱃지의 개수와 종류가 출력된다.

6.2.2 나의 제안 목록

- 해당 메뉴를 클릭하면 나의 제안 목록을 모아 볼 수 있다.

6.2.2 관심 게시글

- 해당 메뉴를 클릭하면 관심 게시글을 모아 볼 수 있다.

6.2.2 거래 완료 게시글

- 해당 메뉴를 클릭하면 거래 완료 게시글을 모아 볼 수 있다.

7. 텃밭 정보

7.1 텃밭 지도

7.1.1 텃밭 정보 모아보기

- 메인페이지 메인 하단의 지도보기 버튼을 클릭하면 텃밭 정보 페이지로 이동한다.
- 텃밭정보를 제공하는 지도와 클러스터링된 마커가 표시된다.
- 지도 하단의 지역 버튼을 클릭하면 지역 별로 텃밭 정보를 볼 수 있다.

7.1.2 텃밭 정보 상세보기

- 지도를 확대하면 텃밭이 있는 위치에 삽 모양으로 마커가 표시된 것을 볼 수 있다.
- 삽 모양의 마커를 클릭하면 텃밭의 상세 정보를 볼 수 있다.
- 텃밭 상세 정보는 텃밭 명, 유형, 주소, 기타 시설이 포함되어 있다.