

**싸브리타임**

23.01.02-23.02.17

A602 (Go)-IDLE

고병진, 김태영, 안준홍, 정원영, 최재희, 황진태

# 개요

## 프로젝트 개요

SSAFY의 다양한 정보들 및 요구사항들을 교육생들에게 알림과 자동화를 통해 제공해준다면 더 편리한 싸피생활이 되지 않을까 하는 생각으로 기획하게 되었습니다.   
SSaveryTime은 SSAFY + EVERY TIME 의 합성어로서, SSAFY의 모든 시간을 효율적으로 이용할 수 있도록 싸피 생활에 도움을 줄 수 있는 다양한 기능들을 제공해주겠다는 의미를 지니고 있습니다.  
SSAFY관련 정보들은 SSaveryTime에게 맡기고, 교육생들은 배움에만 더 전념할 수 있도록 도움을 주는 애플리케이션입니다.

### 프로젝트 사용 도구

|  |  |
| --- | --- |
| 목록 | 이름 |
| 이슈 관리 | JIRA, Notion |
| 형상 관리 | GitLab |
| 문서 관리 | GitLab, Notion |
| 커뮤니케이션 | Notion, Mattermost |
| 디자인 | Figma |
| UCC | 모바비 |
| CI/CD | Jenkins |
| 가상화 | Docker |
| IDE | IntelliJ, VSCode |

### 개발환경

|  |  |
| --- | --- |
| 이름 | 버전 |
| IntelliJ | Community 2022.3.1 / Ultimate 2021.2.4 |
| VSCode | 1.75.1 |
| JDK | Openjdk 11.0.18+10 |
| Spring Boot | 2.7.X |
| Python | 3.11.2 |
| Flask | 2.2.2 |
| Dart | 2.18.6 / DevTools 2.15.0 |
| Flutter | 3.3.10 |
| Jenkins | 2.375.2 |
| Docker | 20.10.23 |
| Server | Ubuntu 20.04 LTS |
| MySQL | 8.0.30 |
| OpenVidu | 2.25.0 |

### 외부 서비스

|  |  |
| --- | --- |
| 이름 | 내용 |
| FCM(Firebase Cloud Message) | 애플리케이션 알림 전송  자세한 설정 파일은 google-services.json 참고 |

### 확인사항

#### BackEnd

##### Spring Boot

<app\_route>\src\main\resources\application.properties

* 해당 파일의 spring.datasource.url의 mysql host가 database 인지 확인
* Docker를 사용하므로 mysql 서비스명으로 접근
* 해당 파일의 OPENVIDU 관련 설정 확인

##### Flask

* DB 주소가 맞는지 확인
* Flask는 외부 접근 MySQL 접속

#### FrontEnd

* repo폴더의 파일들의 baseUrl이 DB주소가 맞는지 확인

## 환경설정

### 도커

#### 도커 설치를 위한 필수 패키지 설치

sudo apt-get update  
sudo apt-get install \  
 ca-certificates \  
 curl \  
 gnupg \  
 lsb-release

#### GPG 키 및 저장소 등록

sudo mkdir -m 0755 -p /etc/apt/keyrings  
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

echo \  
 "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \  
 $(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

#### 도커 엔진 설치

sudo apt-get update  
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

### Jenkins (해당 과정은 CI/CD를 위해 수행하므로 선택사항)

#### Dockerfile 작성

**FROM** jenkins/jenkins:lts  
  
**USER** root  
  
**RUN** apt-get update && \  
 apt-get -y install apt-transport-https \  
 ca-certificates \  
 curl \  
 gnupg2 \  
 zip \  
 unzip \  
 software-properties-common && \  
 curl -fsSL https://download.docker.com/linux/$(. /etc/os-release; echo "$ID")/gpg > /tmp/dkey; apt-key add /tmp/dkey && \  
 add-apt-repository \  
 "deb [arch=amd64] https://download.docker.com/linux/$(. /etc/os-release; echo "$ID") \  
 $(lsb\_release -cs) \  
 stable" && \  
 apt-get update && \  
 apt-get -y install docker-ce

#### docker-compose.yml 작성

# docker-compose.yml  
version: '3.7'  
  
services:  
 jenkins:  
 build:  
 context: .  
 container\_name: jenkins  
 user: root  
 ports:  
 - 8080:8080  
 - 50000:50000  
 container\_name: jenkins  
 volumes:  
 - ./jenkins\_home:/var/jenkins\_home  
 - //var/run/docker.sock:/var/run/docker.sock

#### Jenkins 실행 (DockerFile과 docker-compose.yml이 함께 있는 위치에서 실행)

* 관리자 권한 필요 시

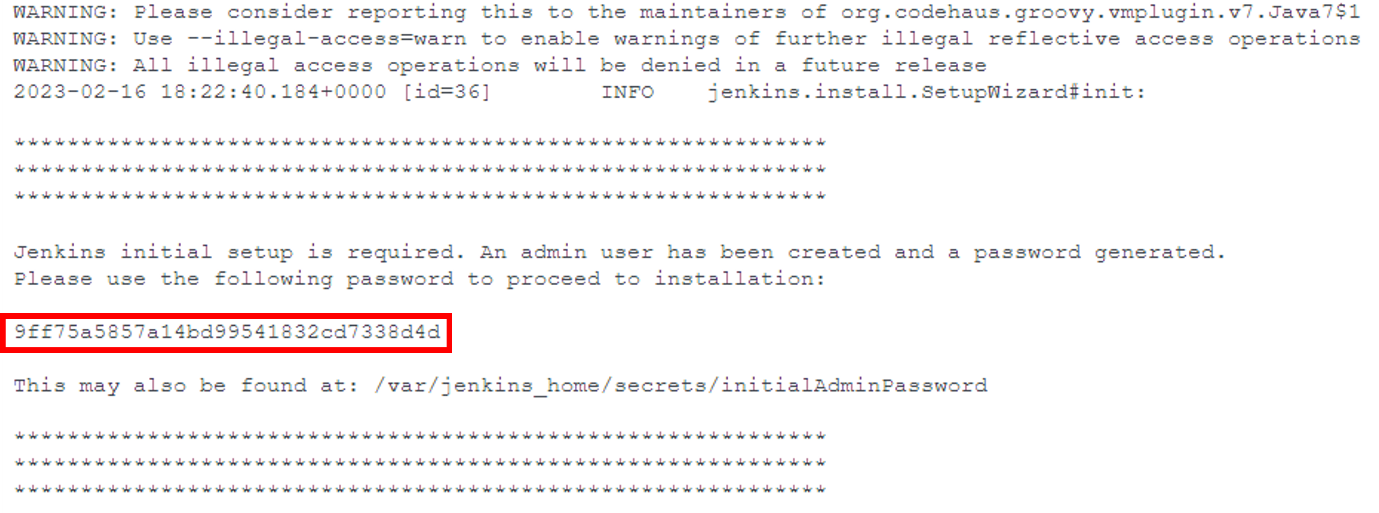
sudo docker compose up -d

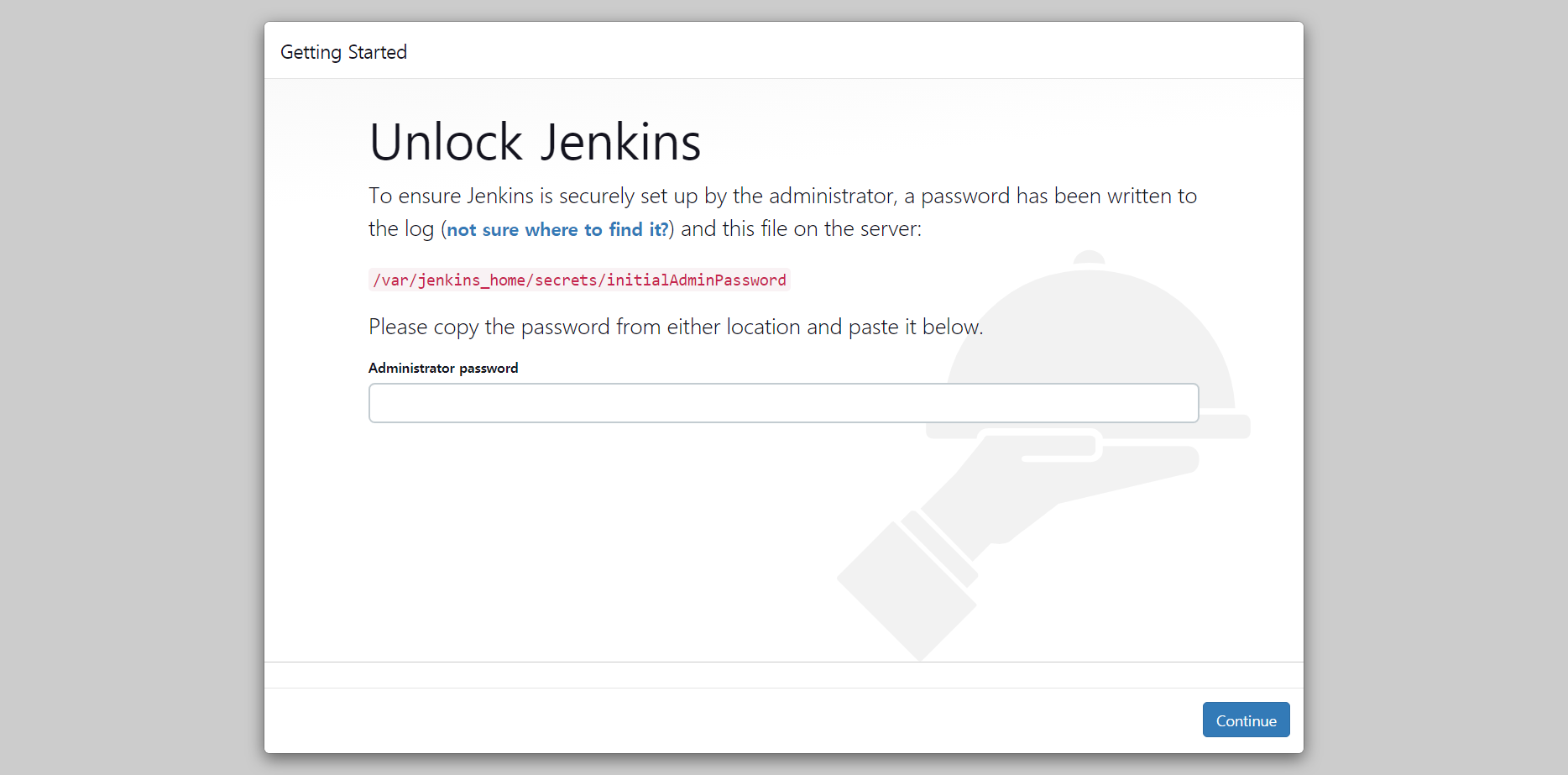
* 일반 권한

docker compose up -d

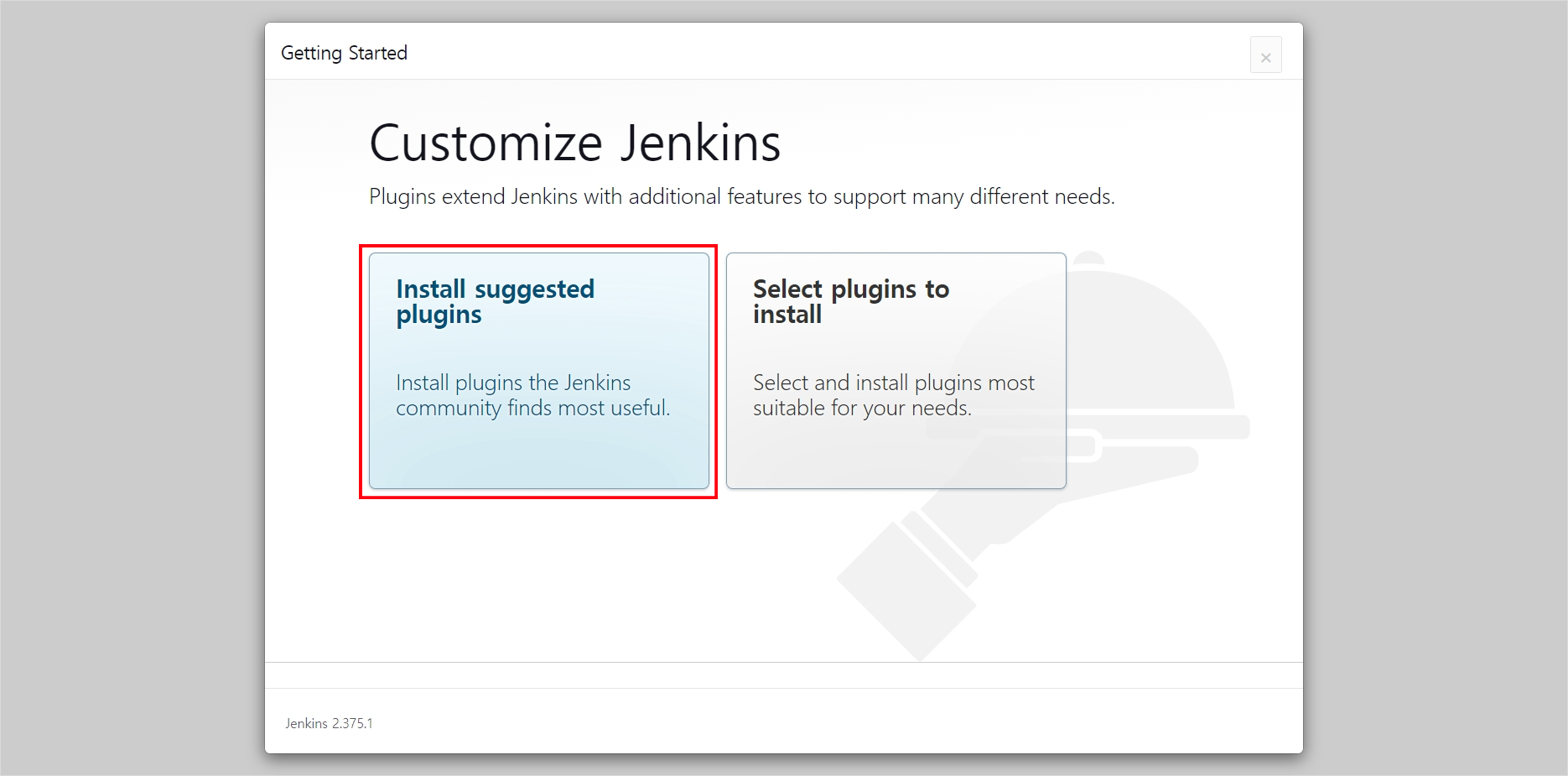
#### Jenkins 환경설정

* 초기 비밀번호 입력

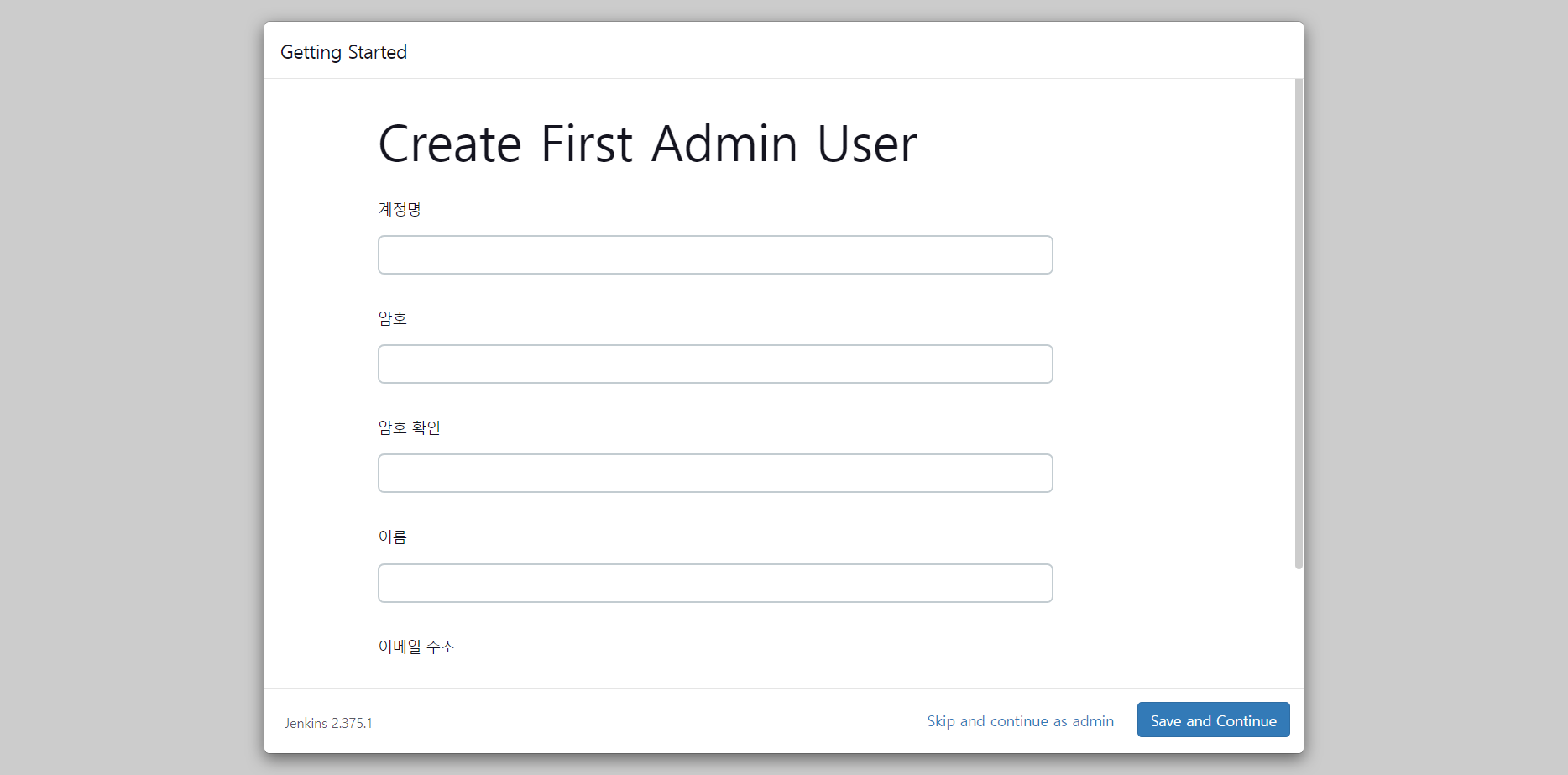




* 기본으로 설치



* Admin 계정 및 다른 서버와 동일 URL을 사용하기 때문에 Instance Configuration은 패스

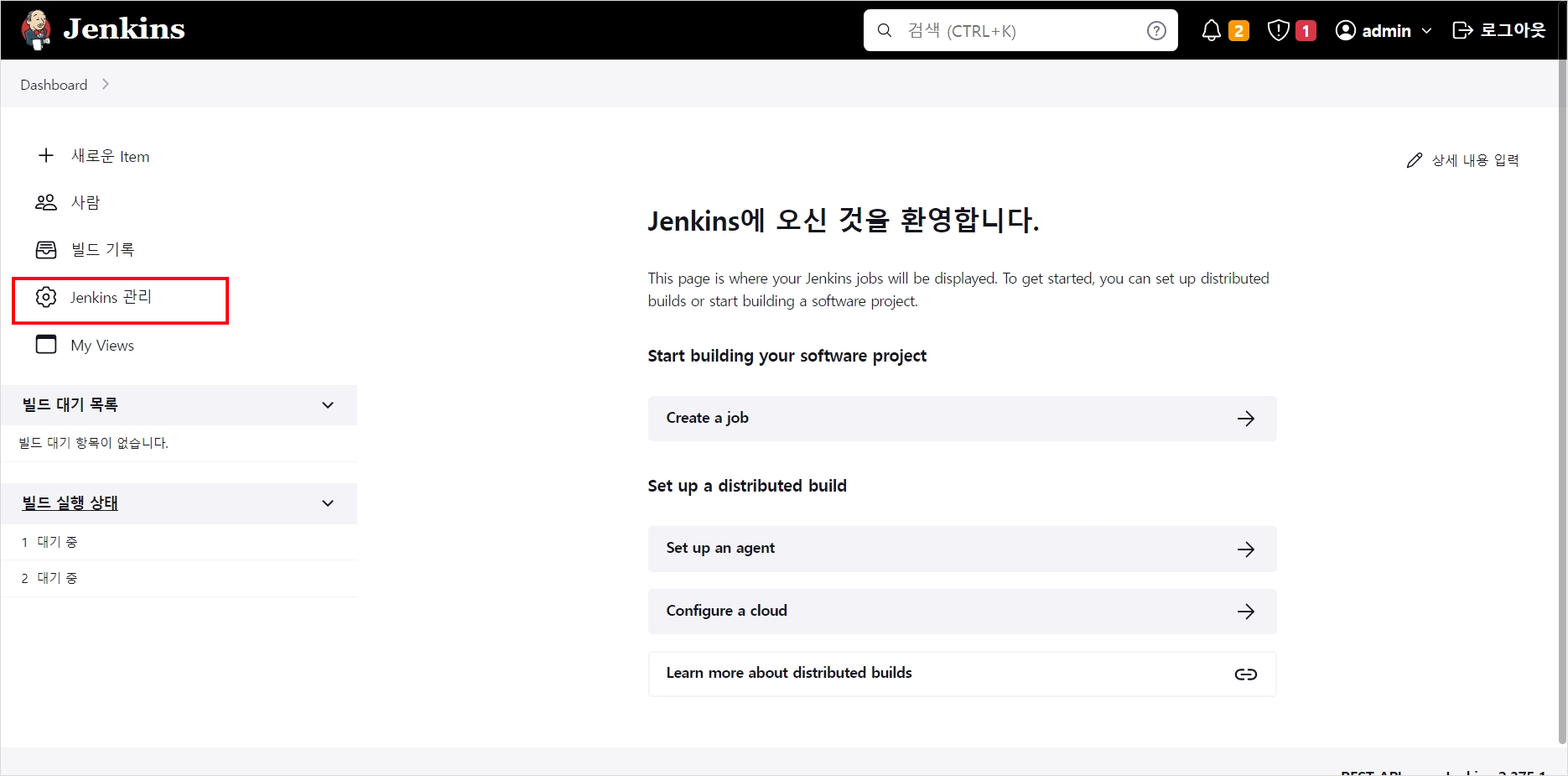


텍스트이(가) 표시된 사진

자동 생성된 설명

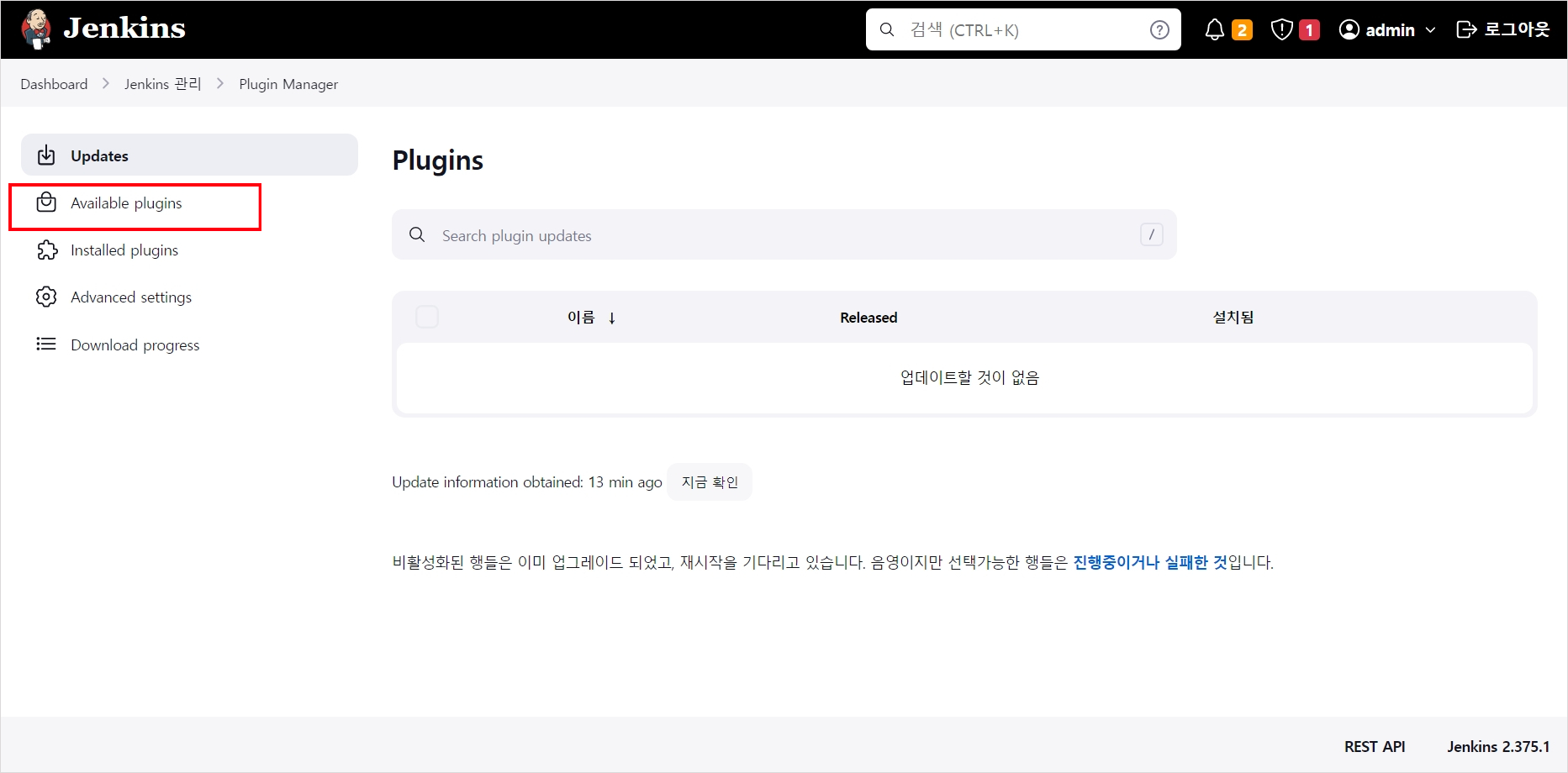
#### Jenkins 플러그인 설치

* 플러그인 설치 페이지로 이동



텍스트이(가) 표시된 사진

자동 생성된 설명



* Docker 플러그인 설치

텍스트이(가) 표시된 사진

자동 생성된 설명

* GitLab 플러그인 설치

텍스트이(가) 표시된 사진

자동 생성된 설명

### Flutter

#### 해당 환경 설정은 윈도우 환경에서 설치 페이지를 참고하여 진행

* + - 1. Flutter <https://docs.flutter.dev/get-started/install/windows>

### FCM (Firebase Cloud Messaging)

* Firebase 콘솔에서 프로젝트 생성
* 프로젝트 메인 => 앱 추가 => Flutter 플랫폼 선택
* 지시사항에 따라 설정 진행

## 빌드 및 배포

### BackEnd

#### Spring Boot & MySQL

##### Spring Boot 빌드

* App 루트에서 진행

./gradlew clean build -x test

##### Docker volume

* 관리자 권한

sudo docker volume create ssafytime\_db

* 일반 권한

docker volume create ssafytime\_db

##### Dockerfile 작성

**FROM** openjdk:11  
**ARG** JAR\_FILE="build/libs/ssafytime-\*-SNAPSHOT.jar"  
**COPY** ${JAR\_FILE} build.jar  
**ENTRYPOINT** ["java","-jar","-Duser.timezone=Asia/Seoul","build.jar"]

##### docker-compose.yml 작성

version: "3.7"  
volumes:  
 ssafytime\_db:  
 external: true  
 name: ssafytime\_db  
services:  
 database:  
 image: mysql:8.0.30  
 environment:  
 MYSQL\_DATABASE: ssafy\_web\_db  
 MYSQL\_ROOT\_HOST: "%"  
 MYSQL\_ROOT\_PASSWORD: ssafy  
 TZ: Asia/Seoul  
  
 volumes:  
 - ssafytime\_db:/var/lib/mysql  
  
 ports:  
 - 3306:3306  
  
 command:  
 - "--character-set-server=utf8mb4"  
 - "--collation-server=utf8mb4\_unicode\_ci"  
  
 restart: always  
  
 application:  
 build: .  
 ports:  
 - 9090:9090  
  
 restart: always

##### 실행

* 관리자 권한

sudo docker compose up -d

* 일반 권한

docker compose up -d

#### Flask

##### Dockerfile 작성

**FROM** python:3.11.2-slim  
**COPY** . .  
**RUN** pip install --no-cache-dir -r requirements.txt  
**EXPOSE** 5000  
**CMD** ["python", "./SSAFYTIME\_lunch.py"]

##### docker-compose.yml 작성

version: "3.7"  
services:  
 ssafytime\_lunch:  
 build: .  
 ports:  
 - 5000:5000  
  
 restart: always

##### 실행

* 관리자 권한

sudo docker compose up -d

* 일반 권한

docker compose up -d

#### OpenVidu (해당 과정부터 관리자 권한으로 진행(Super User) su)(공식문서 권장)

##### OpenVidu on premises 버전 설치

sudo su  
cd /opt

curl https://s3-eu-west-1.amazonaws.com/aws.openvidu.io/install\_openvidu\_2.25.0.sh | bash

cd openvidu

##### OpenVidu 설정

* .env 파일 수정

DOMAIN\_OR\_PUBLIC\_IP= #서버 주소  
OPENVIDU\_SECRET= # openvidu에서 활용할 OPENVIDUAPP과 매치되는 value  
  
# Certificate type:  
# - selfsigned: Self signed certificate. Not recommended **for** production use.  
# Users will see **an** ERROR when connected **to** web page.  
# - owncert: Valid certificate purchased in **a** Internet services company.  
# Please **put** the certificates **files** inside folder ./owncert  
# with names certificate.key and certificate.cert  
# - letsencrypt: Generate **a** **new** certificate using letsencrypt. Please **set** the  
# required contact email **for** Let's Encrypt in LETSENCRYPT\_EMAIL  
# variable.

# ssl 인증 방식  
CERTIFICATE\_TYPE=letsencrypt  
  
LETSENCRYPT\_EMAIL=user@example.com

##### 실행

./openvidu start

### FrontEnd

#### 배포용이 아니므로 단순 release를 빌드를 진행

* Sound null safety

flutter build apk --release --target-platform=android-arm64

* No sound null safety

flutter build apk --release --no-sound-null-safety --target-platform=android-arm64

#### 생성된 apk를 스마트폰에 설치

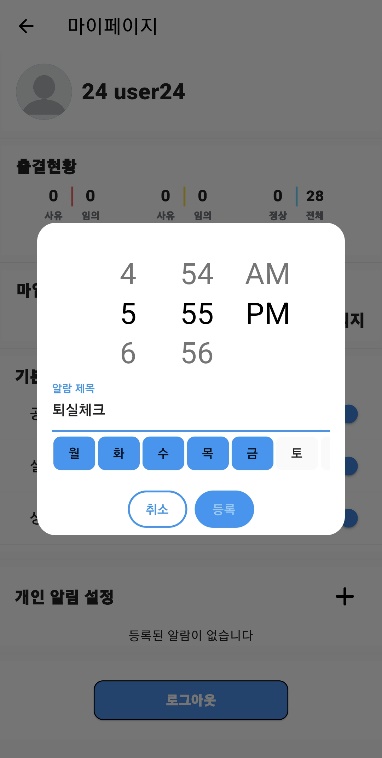
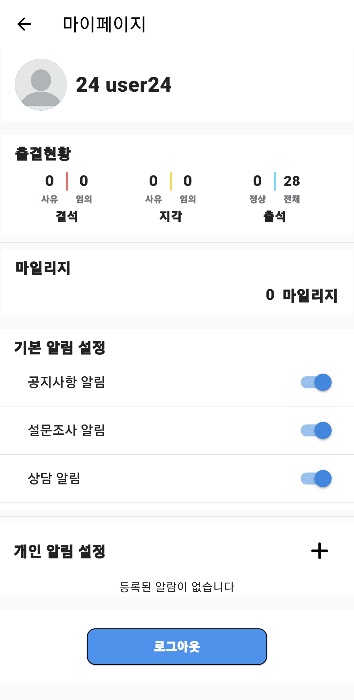
<app\_route>\build\app\outputs\flutter-apk\app-release.apk

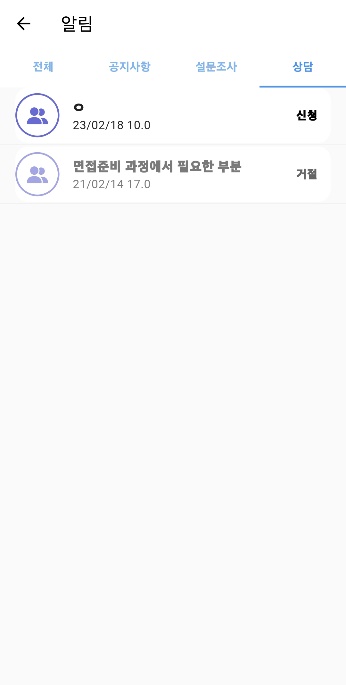
* 해당 위치에 생성된 apk를 설치 후 테스트 진행

# 시연 시나리오

로그인 페이지 메인 페이지

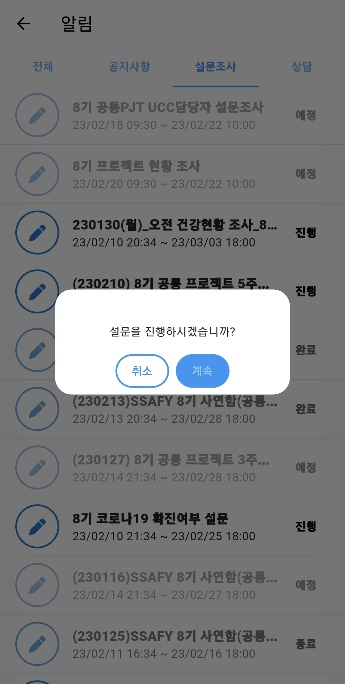
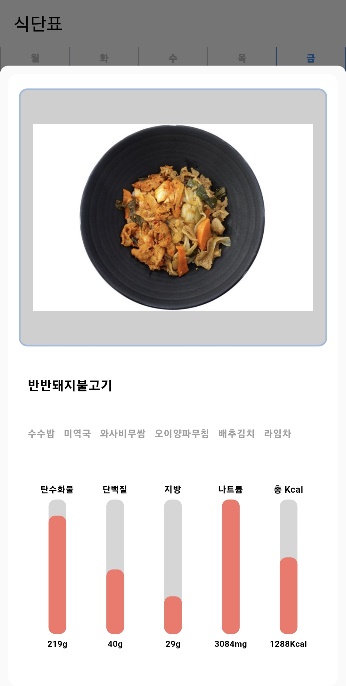


마이 페이지 마이페이지 알람기능

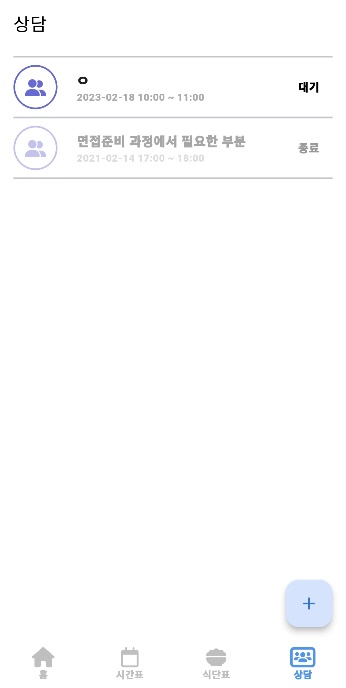
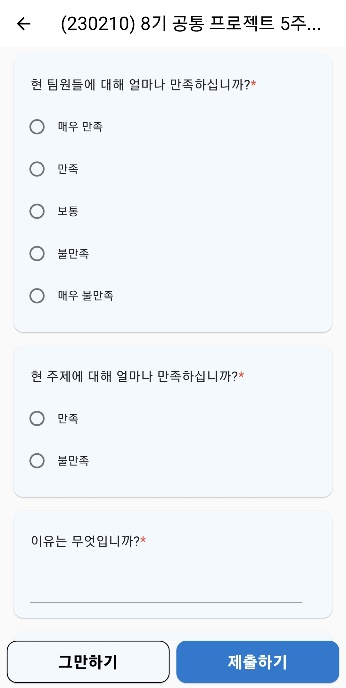
알림: 공지사항 알림: 상담

알림 : 설문조사 알림 : 전체

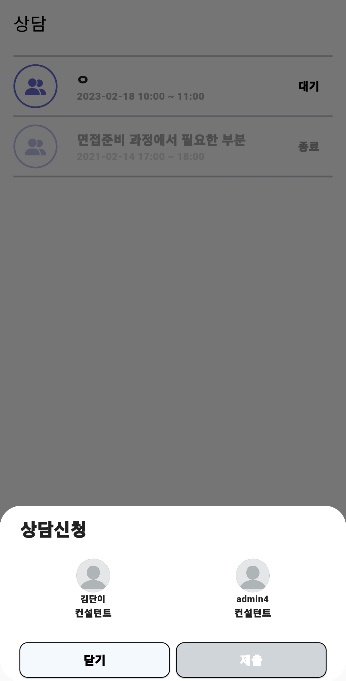
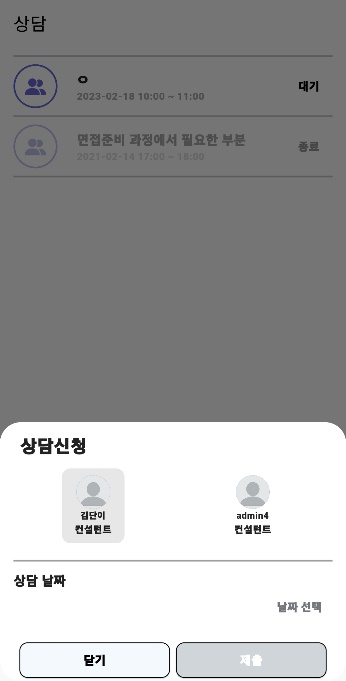
시간표 페이지 식단 페이지

메뉴상세보기 설문조사 선택

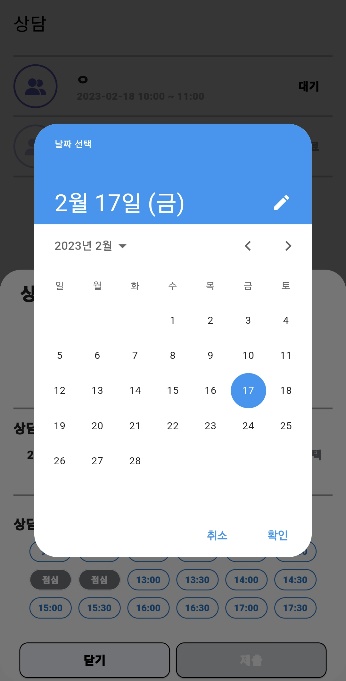
설문조사 진행 교육생 상담화면



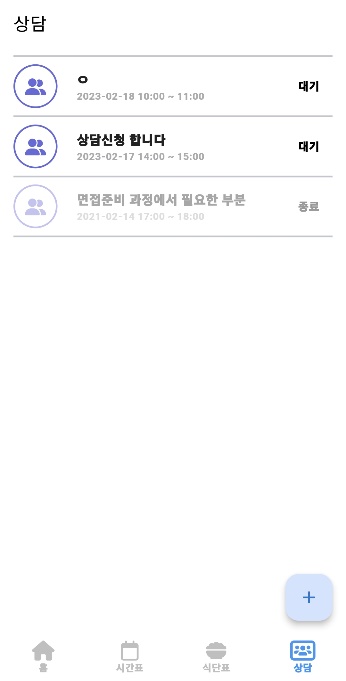
상담 신청 초기 상담신청 관리자 선택

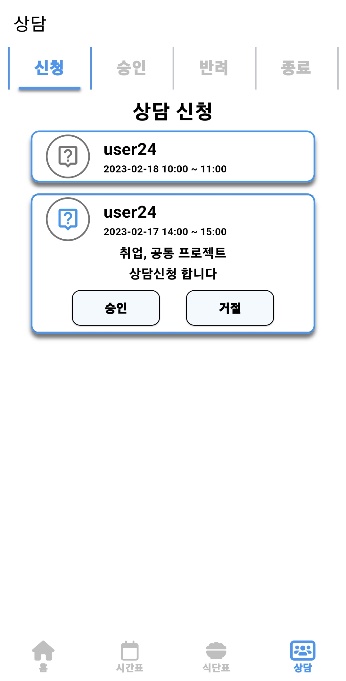
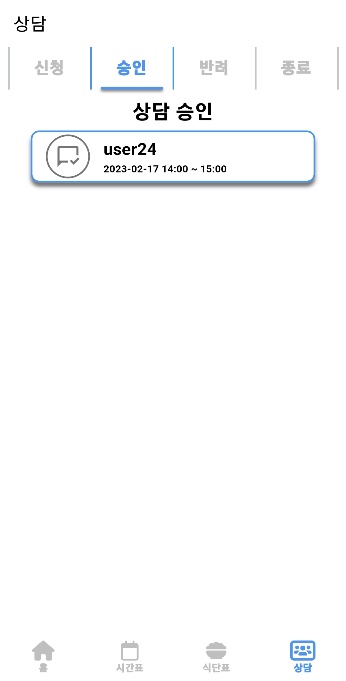


상담신청 날짜선택 시간 선택



상담 신청 작성 완료 교육생 상담화면 신청 완료



관리자 상담 승인 페이지 상담 승인

관리사 상담 승인 완료

