



# campinity

## 포팅 메뉴얼

구미 1반 D101



# 목차

**1. 개발환경**

**2. 배포서버 환경 구축(CI/CD)**

**3. DB 환경 설정**

**4. 빌드 및 로컬환경 테스트 방법**

**5. Kakao API 설정**

**6. 안드로이드 앱 apk 빌드 및 테스트 방법**

**7. 시연 시나리오**



# 1. 개발 환경

## 형상 관리

- Gitlab

## 이슈 관리

- Jira

## CI/CD

- jenkins(2.375.2)
- Docker

## IDE

- IntelliJ 2022 1.3
- Android Studio

## Server

- AWS EC2
  - Ubuntu 20.04LTS
  - Docker 20.10.18

## Front-End

## Communication

- Notion
- Discord
- Google Meet

## UI/UX

- Figma

## OS

- Window 10
- MacOS Monterey

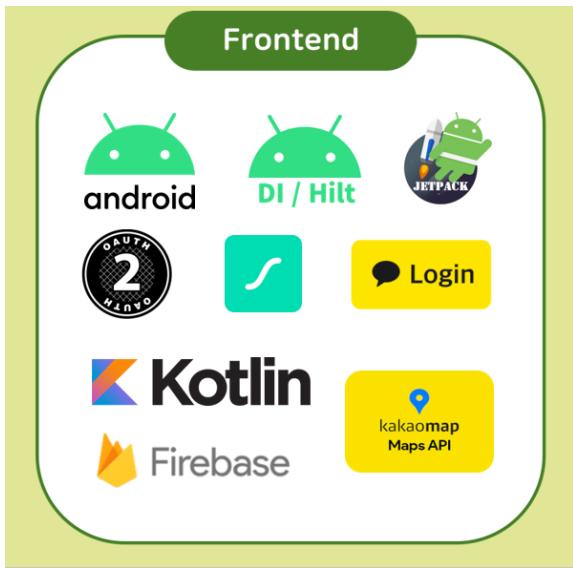
## DataBase

- Ver 15.1 Distrib 10.10.2-MariaDB
- MySQL WorkBench 6.X
- MongoDB
- Redis

## 기타 편의 툴

- PostMan 9.28.1
- Wireshark

- LiveData
- ViewModel



- DataBinding
- ViewPager2
- OkHttp3
- Retrofit2
- Coroutine
- Navigation
- Glide
- Hilt
- Version catalog

## Back-End



- Java Open-JDK 11.0.15
- Gradle 8.0
- Spring Boot 2.7.7
  - Spring Data JPA
  - Lombok
  - Swagger 3.0.0
- Spring Security 5.7.6
- JUnit5 5.8.1
- Firebase 9.1.1
- Websocket 2.6.2
- Spring Batch



## 2. 배포서버 환경 구성(CI/CD)

### 목차

- [Docker 및 docker-compose 설치](#)
- [도커 실행](#)
- [jenkins 설치 및 실행](#)
- [jenkins 초기 설정](#)
- [Jenkins 플러그인 추가 설치](#)
- [jenkins Credential등록](#)
- [jenkins 세부 설정](#)
- [Gitlab WebHook설정](#)
- [Jenkins 설정 후 서버 빌드 및 배포 방법](#)

### ▼ Docker 및 docker-compose 설치

#### 1. 패키지 업데이트 진행

```
sudo apt update & apt upgrade
```

#### 2. Docker 설치에 필요한 필수 패키지 설치

```
sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent software-properties-common
```

#### 3. Docker의 GPC key 인증

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

#### 4. Docker Repository등록

```
sudo add-apt-repository \
"deb [arch=amd64] https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) \
stable"
```

#### 5. Docker 설치

```
sudo apt-get update && sudo apt-get install docker-ce docker-ce-cli containerd.io
```

#### 6. docker-compose 설치

```
sudo curl -L \
"https://github.com/docker/compose/releases/download/1.28.5/dockercompose-$(uname -s)-$(uname -m)" \
-o /usr/local/bin/docker-compose
# 이 명령어는 외부에서 그대로 파일을 가져와서 현재의 시스템에 옮겨 놓는 것이다.
# 결과적으로 "/usr/local/bin/" 경로에 "docker-compose"라는 이름의 파일로 다운된다.
# 참고) https://github.com/docker/compose/releases 에서 최신 버전 확인이 가능하다.
# 최신 버전을 설치하고 싶다면 위 명령어에 보이는 1.28.5라는 버전 숫자를 바꿔주면 된다!
sudo chmod +x /usr/local/bin/docker-compose # chmod 를 통해서 실행이 가능하게 세팅
docker-compose -v # docker-compose 명령이 제대로 먹히는지 확인한다.
```

### ▼ 도커 실행

```
sudo systemctl enable docker
```

```
sudo service docker start
```

## ▼ 젠킨스 설치 및 실행

1. Dockerfile을 만들어서 jenkins 이미지를 활용해 docker.sock을 활용해 통신가능하도록 컨테이너를 띄워준다.

```
FROM jenkins/jenkins:lts
USER root
RUN apt-get update \
    && apt-get -y install lsb-release \
    && curl -fsSL https://download.docker.com/linux/debian/gpg | gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
    && echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/debian $ \
    && apt-get update \
    && apt-get -y install docker-ce docker-ce-cli containerd.io
RUN usermod -u 1000 jenkins && \
    groupmod -g 998 docker && \
    usermod -aG docker jenkins
```

- 이때 주의할점은 아래의 옵션들이다. usermod -u {호스트의사용자아이디} groupmod -g {호스트의 도커 그룹 아이디}
- ubuntu host에서 호스트의 사용자 아이디를 가져오려면 `id -u` 명령어를 활용하고, 도커 그룹 아이디를 가져오고 싶다면 `cat /etc/group | grep docker` 를 사용하면 된다.

2. 이미지를 만든다

```
docker build -t my-jenkins:0.1 .
```

2. 이미지 확인

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
my-jenkins	0.1	d37227db069f	32 minutes ago	985MB

2. volume 만들기

```
docker volume create jenkins
```

3. volume 확인 (docker volume ls)

DRIVER	VOLUME NAME
local	jenkins

6. jenkins 컨테이너 실행

```
docker run -d --name jenkins \
-v /var/run/docker.sock:/var/run/docker.sock \
-v jenkins:/var/jenkins_home \
-p 8080:8080 my-jenkins:0.1
```

## ▼ 젠킨스 초기 설정

1. <http://서버아이피:8080>으로 접속 한다.

2. 젠킨스에 처음 접속하면 초기 관리자 계정의 비밀번호를 입력하라고 나온다.

### Getting Started

## Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

```
/var/jenkins_home/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

Continue

3. 우선 jenkins 컨테이너에 접속한다

```
docker exec -it jenkins /bin/bash
```

4. 초기 관리자 비밀번호를 확인한다

```
cat /var/jenkins_home/secrets/initialAdminPassword
```

5. 비밀번호를 입력하면 아래와 같은 화면이 나오는데, Install suggested plugins 클릭

### Getting Started

## CUSTOMIZE JENKINS

Plugins extend Jenkins with additional features to support many different needs.

Install suggested  
plugins

Install plugins the Jenkins community finds most useful.

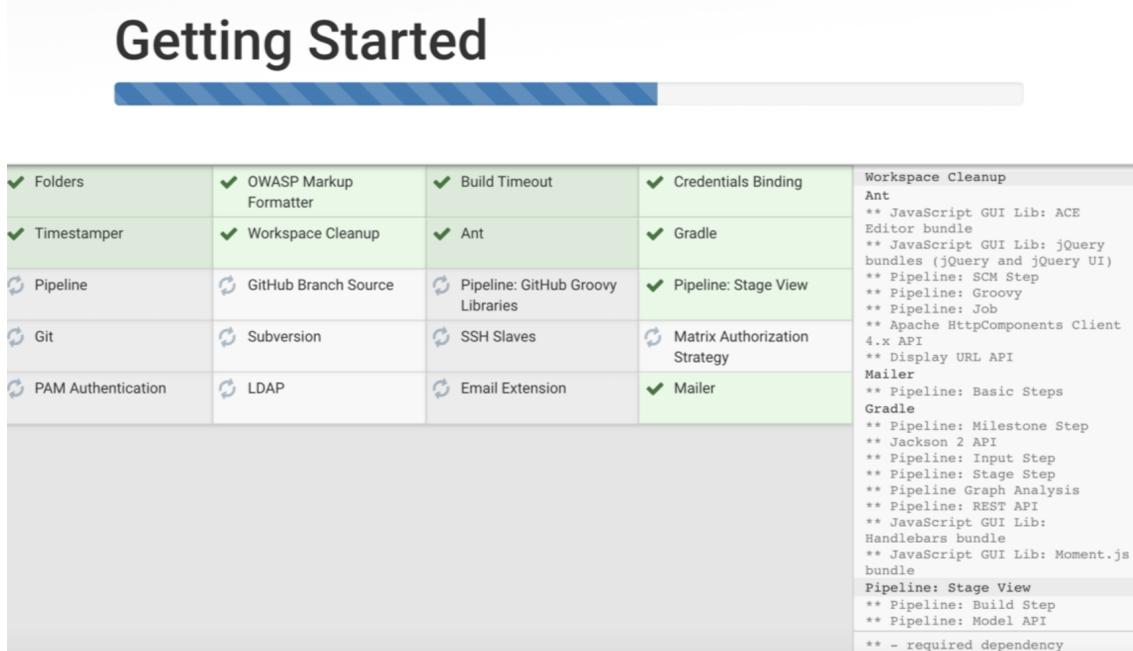
Select plugins  
to install

Select and install plugins most suitable for your needs.

Jenkins 2.289

6. 그대로 쪽 설치진행 하면 아래와 같이 진행이 된다, 설치가 모두 완료되면 관리자의 아이디 패스워드 설정하는창이 나오고 본인이 설정하고싶은 패스워드로 설정하면된다.

## Getting Started



## ▼ Jenkins 플러그인 추가 설치

The screenshot shows the Jenkins 'Plugin Manager' page. It has a header 'Dashboard > Jenkins 관리 > Plugin Manager'. Below the header, there are tabs: 'Updates' (disabled), 'Available plugins' (selected), 'Installed plugins', and 'Advanced settings'. There's also a search bar 'search available plugins'. At the bottom, there are buttons for 'Install', 'Name ↓', and 'Released'.

### 설치할 플러그인 목록

- Gitlab
- Gradle(이미 설치되어 있다면 설치 안해줘도 됨)

플러그인 검색창에서 Gitlab, Gradle 검색 후 설치해준 뒤 jenkins를 재시작 시켜준다.

설치완료가 되면 jenkins admin페이지 하단에 restart jenkins라는 버튼이 생기고, 눌러주면 자동으로 재시작이 완료된다.

그러나 가끔 재시작이 정상적으로 되지 않는 경우가 생기는데 그때는 EC2에 원격접속 후 jenkins 컨테이너를 아래와 같은 명령어를 사용해 재실행해준다.

```
docker restart {container_id 또는 container name}
```

## ▼ 젠킨스 Credential등록

1. Jenkins관리 → Manage Credentails에 들어간다

## Security

The screenshot shows the Jenkins Security section with three main options:

- Configure Global Security**: Secure Jenkins; define who is allowed to access/use the system.
- Manage Credentials**: Configure credentials (highlighted with a red box).
- Configure Credential Providers**: Configure the credential providers and types.

## 2. System 선택

# Credentials

The screenshot shows the Jenkins Credentials page with the following interface elements:

- Header: T P Store ↓ Domain
- Stores: User, System (highlighted in yellow), Global
- Domain: (global)

## 3. 우측 상단에 Add Credentials 클릭

The screenshot shows the Jenkins Global credentials (unrestricted) page with the following details:

- Header: Global credentials (unrestricted) + Add Credentials
- Text: Credentials that should be available irrespective of domain specification to requirements matching.
- Table:

ID	Name	Kind	Description
uberamen5ch1308	uberamen5ch1308/***** (gitlab)	Username with password	gitlab

## 4. 우선 Gitlab 프로젝트 Repository에 들어가서 accesstoken을 발급받는다

The screenshot shows the GitLab Access Tokens creation page for the project s08-webmobile4-sub2. The left sidebar shows project navigation. The right panel contains the following fields:

- Project Access Tokens**: Add a project access token. Enter the name of your application, and we'll return a unique project access token.
- Token name**: A text input field.
- Expiration date**: A date picker set to 2023-03-18.
- Select a role**: A dropdown menu set to Guest.
- Select scopes**: A section for setting permission levels. Options include:
  - api**: Grants complete read and write access to the scoped project API, including the Package Registry.
  - read\_api**: Grants read access to the scoped project API, including the Package Registry.
  - read\_repository**

## 5. 발급 받은 accessToken을 Credential을 만들때 password로 입력해주고 나머지 부가 정보들도 입력해주고 Create해준다

New credentials

Kind: Username with password

Scope: Global (Jenkins, nodes, items, all child items, etc)

Username: übermen5ch1308 Gitlab username

Treat username as secret:

Password: project accessstoken 입력

ID: Gitlab ID 입력

Description:

**Create**

## ▼ 젠킨스 세부 설정

1. 젠킨스 로그인을 완료하고 새로운 Item추가를 클릭한다.
2. item이름을 입력하고 Freestyle project를 선택한다.

Enter an item name

campinity-develop<sup>2</sup>

» Required field

**Freestyle project**  
이것은 Jenkins의 주요 기능입니다. Jenkins은 어느 빌드 시스템과 어떤 SCM(형상관리)으로 묶인 당신의 프로젝트를 빌드할 것이고, 소프트웨어 빌드보다 다른 어떤 것에 자주 사용될 수 있습니다.

**Pipeline**

3. 소스 코드 관리 목록중 Git을 선택하고 Repository URL에 Gitlab프로젝트 URL을 입력하고 이전에 설정해둔 Credential로 선택해준다.

### 소스 코드 관리

None

Git [?](#)

#### Repositories [?](#)

Repository URL [?](#)

<https://lab.ssafy.com/s08-webmobile4-sub2/S08P12D101.git> (깃랩 repository URL)

! Please enter Git repository.

#### Credentials [?](#)

- none - GitLab repository token을 기반으로 만든 Credential등록

[+ Add](#)

[고급...](#)

4. 바로 아래에 어떤 브랜치에서 commit하고와서 Integration 시킬지 branch를 명시해준다. (deploy-be로 설정)

Branches to build ?

Branch Specifier (blank for 'any') ?

X

Add Branch

5. 빌드 유발 설정 부분을 아래와 같이 설정해준다 .

### 빌드 유발

빌드를 원격으로 유발 (예: 스크립트 사용) ?

Build after other projects are built ?

Build periodically ?

Build when a change is pushed to GitLab. GitLab webhook URL: http://i8d101.p.ssafy.io:8080/project/campinity-develop2 ?

Enabled GitLab triggers

Push Events

Push Events in case of branch delete

Opened Merge Request Events

Build only if new commits were pushed to Merge Request ?

Accepted Merge Request Events

Closed Merge Request Events

Rebuild open Merge Requests

Never

Approved Merge Requests (EE-only)

Comments

Comment (regex) for triggering a build ?

Jenkins please retry a build

6. Build Steps설정은 다음과 같다

## Build Steps

The screenshot shows the Jenkins Pipeline configuration page. It displays two build steps:

- Invoke Gradle script**:
  - Gradle Version**: gradle 8.0
  - Tasks**: build -p /var/jenkins\_home/workspace/campinity-develop/Server -x test
- Execute shell**:
  - Command**:

```
cd Server
docker-compose up --build -d
```

At the bottom, there are two buttons: **저장** (Save) and **Apply**.

## ▼ Gitlab WebHook설정

1. Gitlab WebHook 탭에 들어가서 설정값들을 입력해준다.

2. 배포서버 환경 구성(CI/CD)

**Webhook**

[Webhooks](#) enable you to send notifications to web applications in response to events in a group or project. We recommend using an [integration](#) in preference to a webhook.

**URL** `jenkins-server-url/jenkins-item-name`

URL must be percent-encoded if it contains one or more special characters.

**Secret token**

Used to validate received payloads. Sent with the request in the `X-Gitlab-Token` HTTP header.

**Trigger**

Push events  
  
Push to the repository.

Tag push events  
A new tag is pushed to the repository.

Comments  
A comment is added to an issue or merge request.

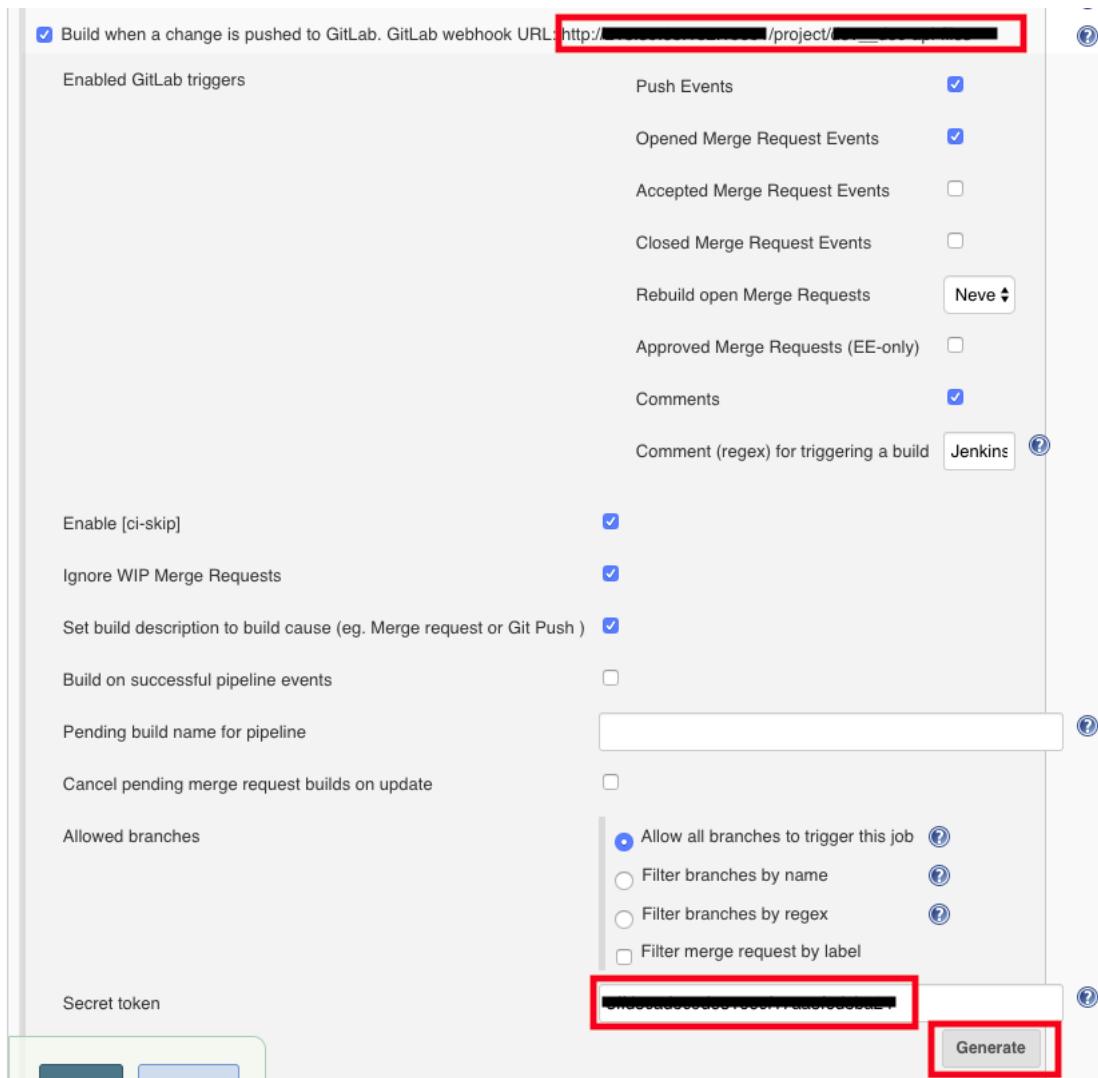
Confidential comments  
A comment is added to a confidential issue.

Issues events  
An issue is created, updated, closed, or reopened.

Confidential issues events  
A confidential issue is created, updated, closed, or reopened.

Merge request events  
A merge request is created, updated, or merged.

- 설정값중 Secret token은 jenkins서버에 접속해서 item(campinity-develop) → 구성 → 빌드유발 → 고급 → Secret token → generate 클릭해서 토큰을 발급받고 입력해준다.



### 3. Gitlab WebHook으로 다시 돌아가서 WebHook test

**SSL verification**

Enable SSL verification

**Save changes** **Test** **Delete**

---

Status	Trigger	Elapsed time	Request time
200	Merge Request Hook	0.02 sec	just now
<a href="#">View details</a>			

4.

## ▼ Jenkins 설정 후 서버 빌드 및 배포 방법

### 자동으로 빌드 및 배포

Jenkins에 등록한 project URL에 Merge Request가 accept되면 자동으로 빌드 및 배포가 된다.

### 수동으로 빌드 및 배포

The screenshot shows the Jenkins dashboard for the 'campinity-develop' project. At the top, there's a breadcrumb navigation: Dashboard > campinity-develop >. Below it is a horizontal menu bar with several items: 상태 (Status), 변경 사항 (Changes), 작업공간 (Workspaces), **지금 빌드** (Build Now) which is highlighted in yellow, 구성 (Configure), Project 삭제 (Delete Project), and Rename. Underneath the menu bar, there's a list of recent builds: Last build, (#225), 25 min 전; Last stable build, (#225), 25 min 전; Last successful build, (#225), 25 min 전; Last failed build, (#131), 5 days 15 hr 전; Last unsuccessful build, (#155), 3 days 3 hr 전; and Last completed build, (#225), 25 min 전.

지금 빌드 버튼을 누르면 item 설정된 URL 및 branch를 jenkins가 인식하고 commit들을 fetch후 build 및 deploy를 진행한다.

## 참고 사항

위의 과정을 그대로 따라서 거쳐온다면, 서버가 제대로 작동하지 않을것이다. 그 이유는 MariaDB 컨테이너에 Database가 생성되지 않은 상태이기 때문이다.

따라서 EC2에 원격접속 후 DB관련 환경설정을 해주어야한다.

[3. MariaDB 환경설정](#)



## 3. MariaDB 환경설정



MariaDB Container에 접속 후 접속할 유저 생성

### 1. EC2 접속

```
ssh -i I8D101.pem ubuntu@i8d101.p.ssafy.io
```

### 2. MariaDB container 접속

```
docker exec -it database /bin/bash
```

### 3. MariaDB 서버에 root계정으로 접속

```
mysql -u root -p
```

### 4. User 등록

```
create user userid@접속할 host 외부아이피 identified by '비밀번호';
```

### 5. 권한 부여

```
GRANT ALL PRIVILEGES ON DB명.테이블 TO 계정아이디@host IDENTIFIED BY '비밀번호';
```

### 6. 권한 반영

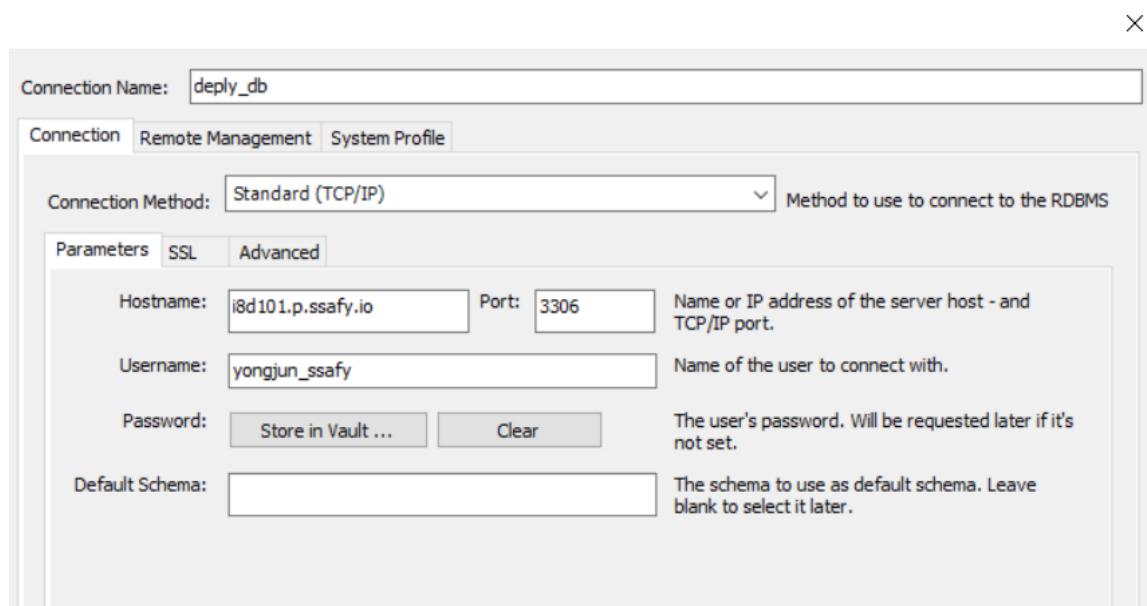
```
flush privileges;
```



## MariaDB 원격접속 후 Database 생성 및 데이터 삽입

### 1. MySQL Workbench로 접속

- MariaDB서버에서 root권한을 생성한 user의 username과 password를 설정해주고, hostname은 host의 아이피(Domain Name)를 입력해준다. 포트는 3306



### 2. 접속 후 project repository/exec 폴더에있는 backup.sql을 다운받아서 sql을 execute 해준다.



## 4. 빌드 및 로컬환경 테스트방법



### 환경 구성

- 운영체제 : Window 10
- 준비물 : IntelliJ, Docker Desktop, Git



### 빌드 및 테스트 방법

#### 1. Repository clone

```
git clone --single-branch --branch develop-be https://lab.ssafy.com/s08-webmobile4-sub2/S08P12D101.git
```

#### 2. batch application.yml 수정

```
~/S08P12D101/Server/demo-batch-server/src/main/resources/application.yml
```

위 경로에 있는 파일을 열어서 spring:profiles:active: local로 수정한다

```
server:
  port: 8002

spring:
  profiles:
    active: local # 기본 환경을 prod로 셋팅 local 개발할때는 local로 바꾸세요.

  batch-db:
    datasource:
      driver-class-name: org.mariadb.jdbc.Driver
  campinity-db:
    datasource:
      driver-class-name: org.mariadb.jdbc.Driver
```

#### 3. core application.yml 수정

```
~/S08P12D101/Server/demo-core/src/main/resources/application.yml
```

위 경로에 있는 파일을 열어서 spring:profiles:active: local로 수정한다

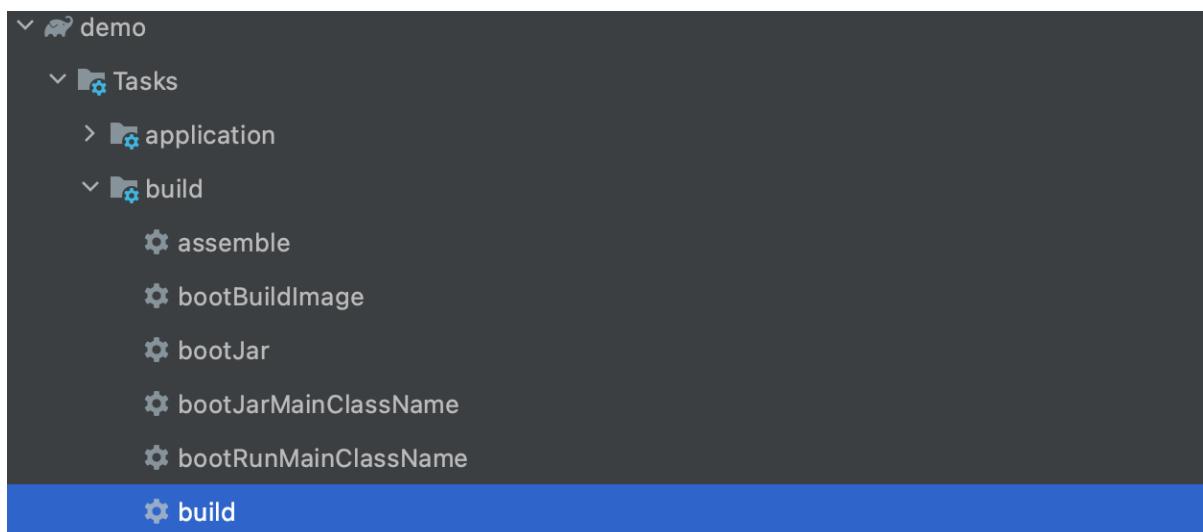
```
server:
  port: 8003

spring:
  profiles:
    active: local # 기본 환경을 prod로 셋팅 local 개발할때는 local로 바꾸세요.

  hikari:
    connectionTimeout : 30000
    maximumPoolSize : 30
    maxLifeTime : 97
    poolName : HikariCP
    readOnly : false
```

## 4. Gradle을 이용해 빌드(jar파일 생성)

IntelliJ의 Gradle을 켜면 빌드



## 5. docker-compose로 container띄우기

```

# ~/SSAFY/common_project/clone/S08P12D101/Server/ [develop-be*] docker-compose up -d
Creating network "server_default_bridge" with the default driver
Creating redis_boot ... done
Creating mongo ... done
Creating database ... done
Creating server_application_1 ... done
Creating server_batch-server_1 ... done
# ~/SSAFY/common_project/clone/S08P12D101/Server/ [develop-be*] docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
1151c3c57e3a server_application "java '-Duser.timezone=' 3 seconds ago Up 2 seconds 0.0.0.0:8003->8003/tcp, :::8003->8003/tcp server_application_1
3f8530af788 server_batch-server "java -Duser.timezone=" 3 seconds ago Up 2 seconds 0.0.0.0:8002->8002/tcp, :::8002->8002/tcp server_batch-server_1
6febbedf20ec redis:alpine "docker-entrypoint.s..." 5 seconds ago Up 3 seconds 0.0.0.0:6379->6379/tcp, :::6379->6379/tcp redis_boot
5ad3d14bc70e mariadb "docker-entrypoint.s..." 5 seconds ago Up 3 seconds 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp database
3a62c67df135 mongo:4.4.3 "docker-entrypoint.s..." 5 seconds ago Up 3 seconds 0.0.0.0:27017->27017/tcp, :::27017->27017/tcp mongo

```

## 6. docker-compose로 container 띄우기

접속이 잘되는지 테스트해보기



## 5. Kakao API 설정

### ▼ 카카오 로그인

**Kakao Developers**

카카오 API를 활용하여 다양한 어플리케이션을 개발해보세요. 카카오 로그인, 메시지 보내기, 친구 API, 인공지능 API 등을 제공합니다.

 <https://developers.kakao.com/>

kakao developers

### 1. 카카오 Developers에서 애플리케이션 추가

#### 애플리케이션 추가하기

앱 아이콘

이미지 업로드

파일 선택

JPG, GIF, PNG  
권장 사이즈 128px, 최대 250KB

앱 이름

내 애플리케이션 이름

사업자명

사업자 정보와 동일한 이름

• 입력된 정보는 사용자가 카카오 로그인을 할 때 표시됩니다.  
• 정보가 정확하지 않은 경우 서비스 이용이 제한될 수 있습니다.

서비스 이용이 제한되는 카테고리, 금지된 내용, 금지된 행동 관련 운영정책을 위반하지 않는 앱입니다.

### 2. 카카오 로그인 활성화



### 3. Redirect URI 등록

#### Redirect URI

##### Redirect URI

카카오 로그인에서 사용할 OAuth Redirect URI를 설정합니다.

여러개의 URI를 줄바꿈으로 추가해주세요. (최대 10개)

REST API로 개발하는 경우 필수로 설정해야 합니다.

예시: (O) <https://example.com/oauth> (X) <https://www.example.com/oauth>

```
http://localhost:8003/api/v4/members/login-kakao  
http://i8d101.p.ssafy.io:8003/api/v4/members/login-kakao
```

취소

저장

## ▼ 카카오 맵

### 1. 키해시 등록

## Android

삭제

수정

패키지명	com.ssafy.campinity
마켓 URL	market://details?id=com.ssafy.campinity
키 해시	7zjiK1SEvaYOU5dZ6lQ6WhBOuoA= D2MXzfSh+Q7reHO4KSnrPA7U7DQ= FyrvyqP6pYXnrK4Gie+9tEELwis= HVwVpXUZ7Kg/qD3myZlrlZCU94Q= MrufKUQLImOf1U3G7K5nowqh3Ec= U9lXIQNNKsxV0iPA1eHxykCifg4= WRgq4Y+c5mqy2o/D2xRCQhyLnOo= aKAtmSZc0nPrjfPqUkaINQi2lU= Ujezl3wCKT3+R+FIL4lWbvB15WA= xB7mB49E3ud24oVuQ0/Rqjtdd0I=

## 2. 앱 키 등록(네이티브 앱키)

### 앱 키

플랫폼

네이티브 앱 키

REST API 키

JavaScript 키

Admin 키



## 6. 안드로이드 앱 apk 빌드 및 테스트 방법



### 환경 구성

- 운영체제 : Window 10(MAC Monterey도 동일)
- 준비물 : Android Studio, Git

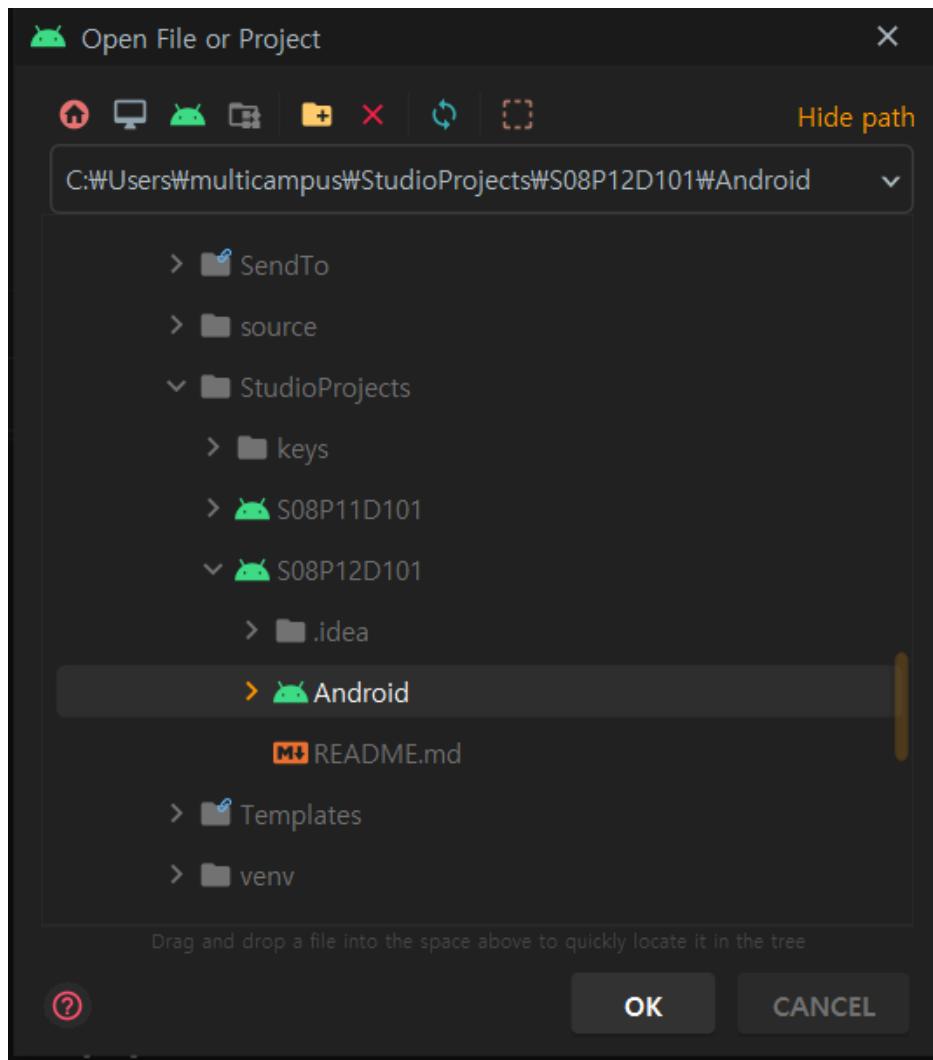


### 빌드 및 테스트 방법

#### 1. Repository clone

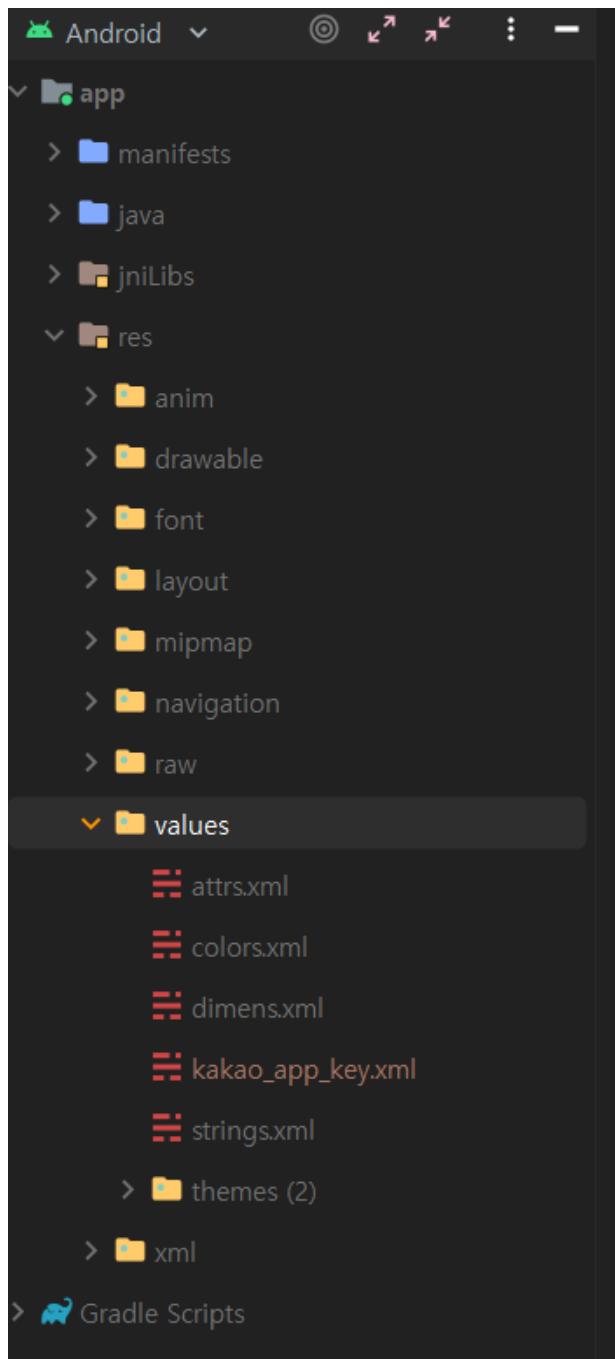
```
git clone --single-branch --branch develop-fe https://lab.ssafy.com/s08-webmobile4-sub2/S08P12D101.git
```

#### 2. 안드로이드 스튜디오 열기

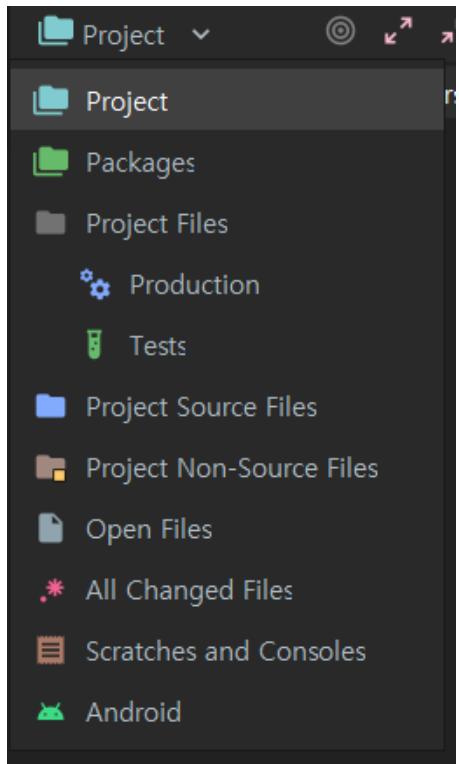


### 3. res의 values 폴더 클릭 후 kakao\_api\_key.xml 파일 붙여넣기

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/753a1f39-9c19-4ea0-adfb-4f295acc3393/Untitled.xml>

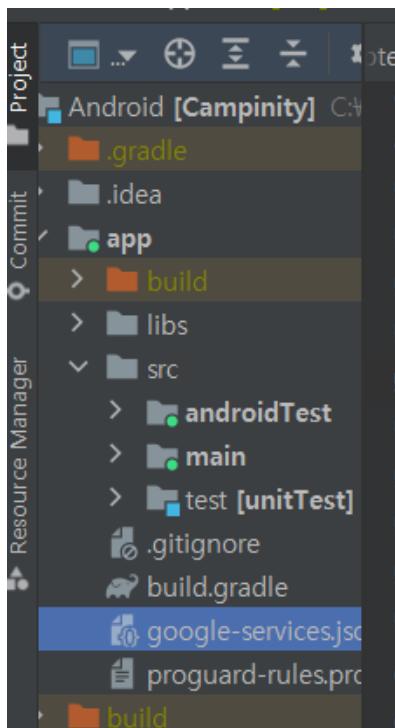


#### 4. 왼쪽 파일 계층 구조의 상단의 Android 클릭 후 Project로 계층 보기 변경

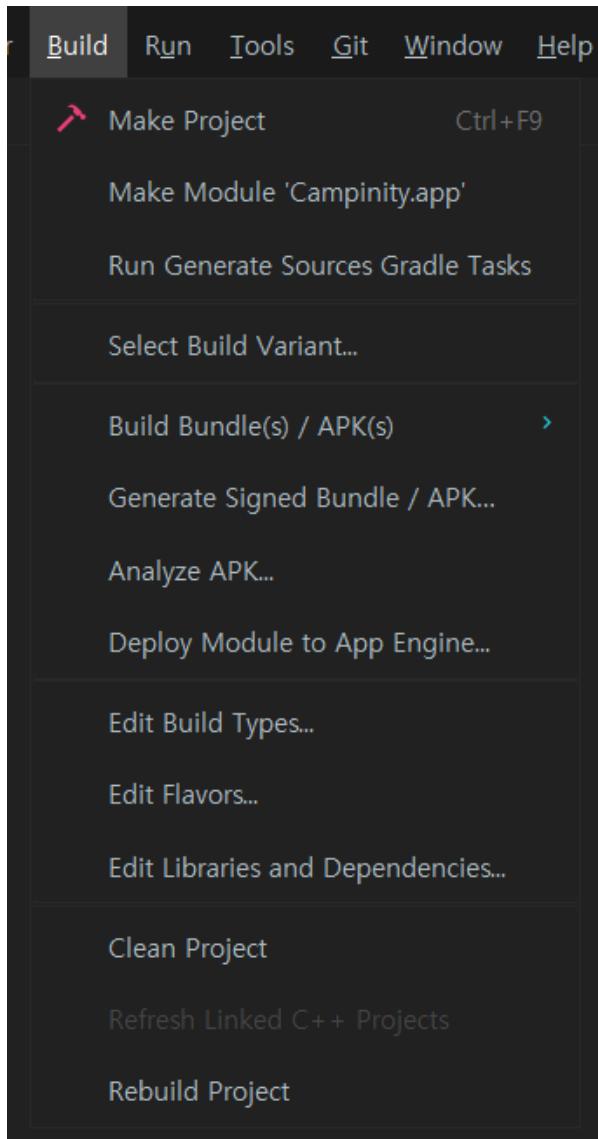


## 5. app 폴더 클릭 후 google-services.json 불여넣기

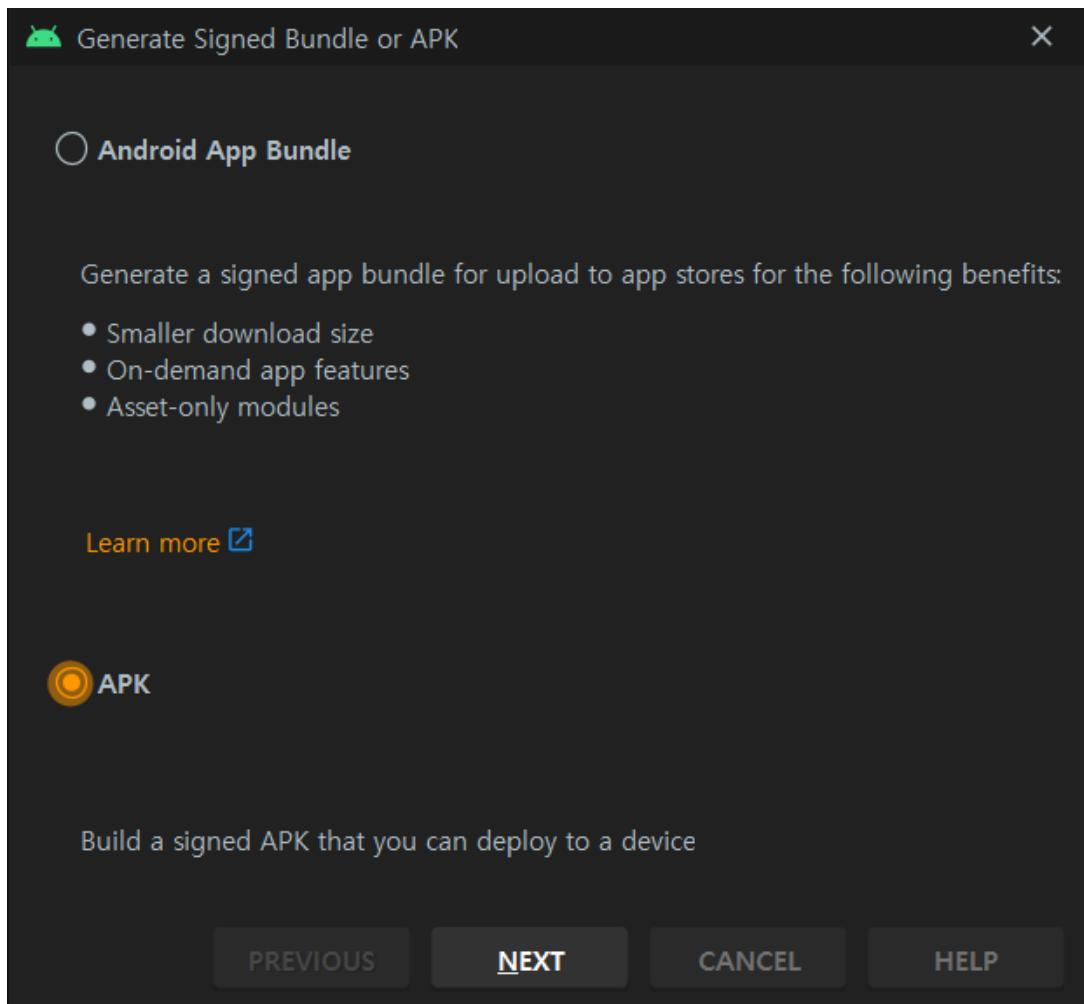
<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/48d32072-2203-4f36-a52b-fdba4caf2ba2/Untitled.json>



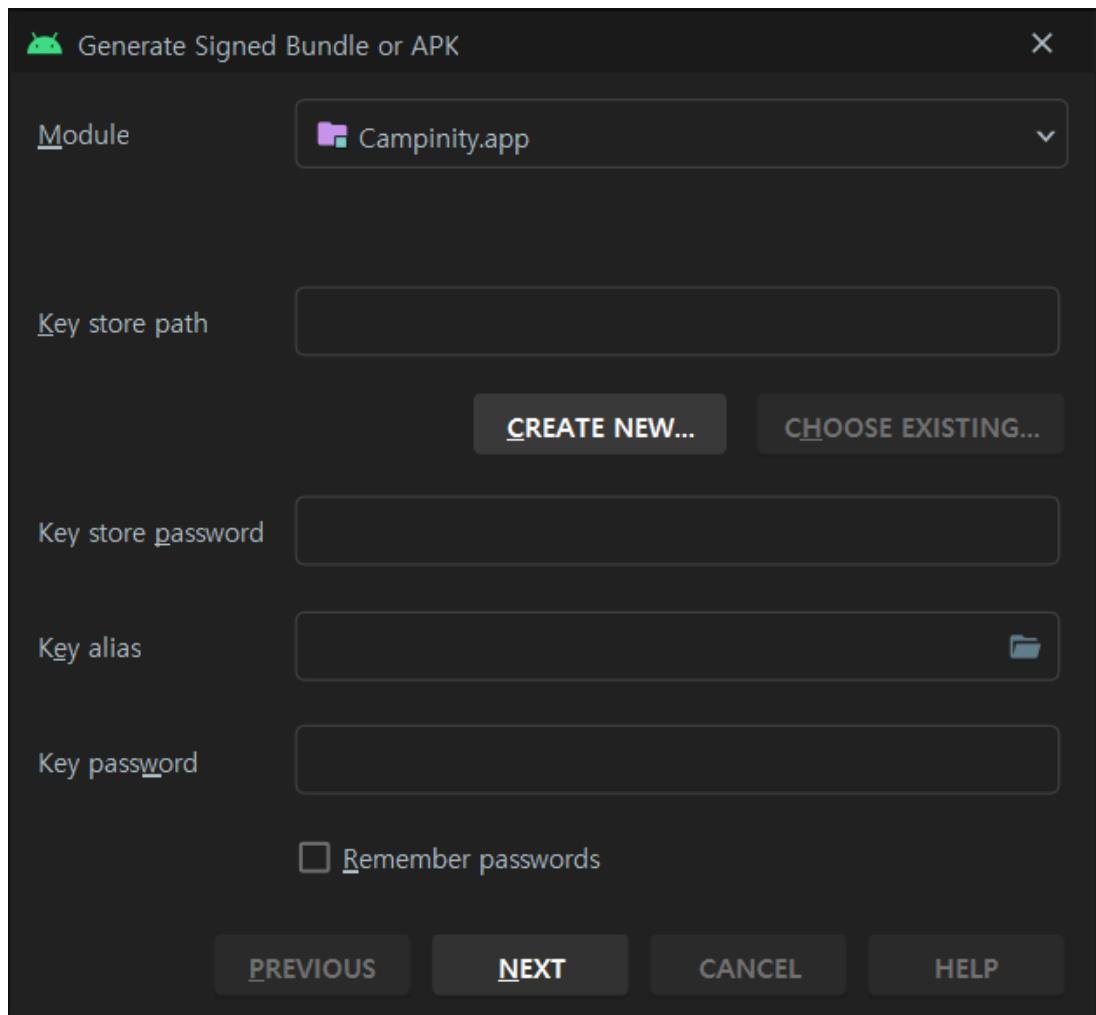
## 6. 상단 빌드 탭의 Generate Signed Bundle / APK 클릭



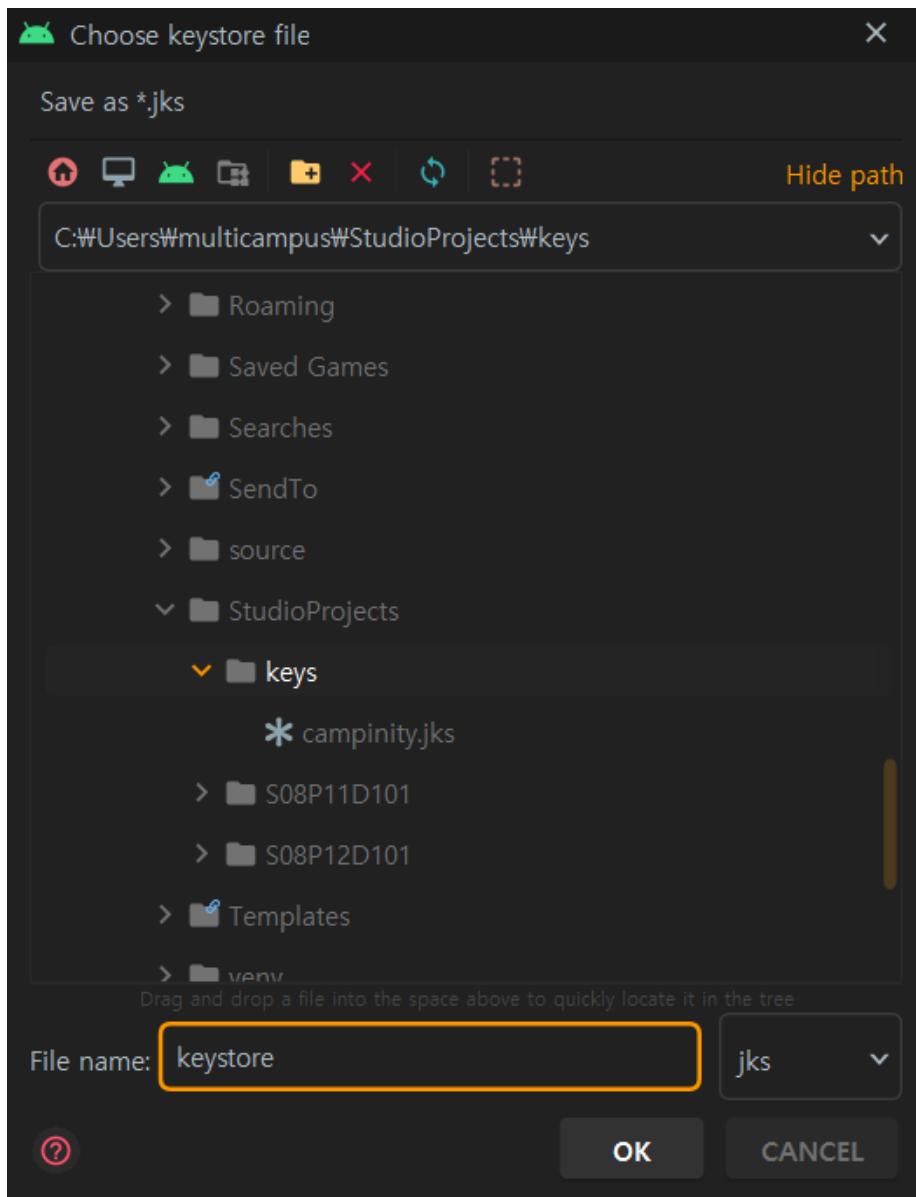
## 7. APK 선택, NEXT 클릭



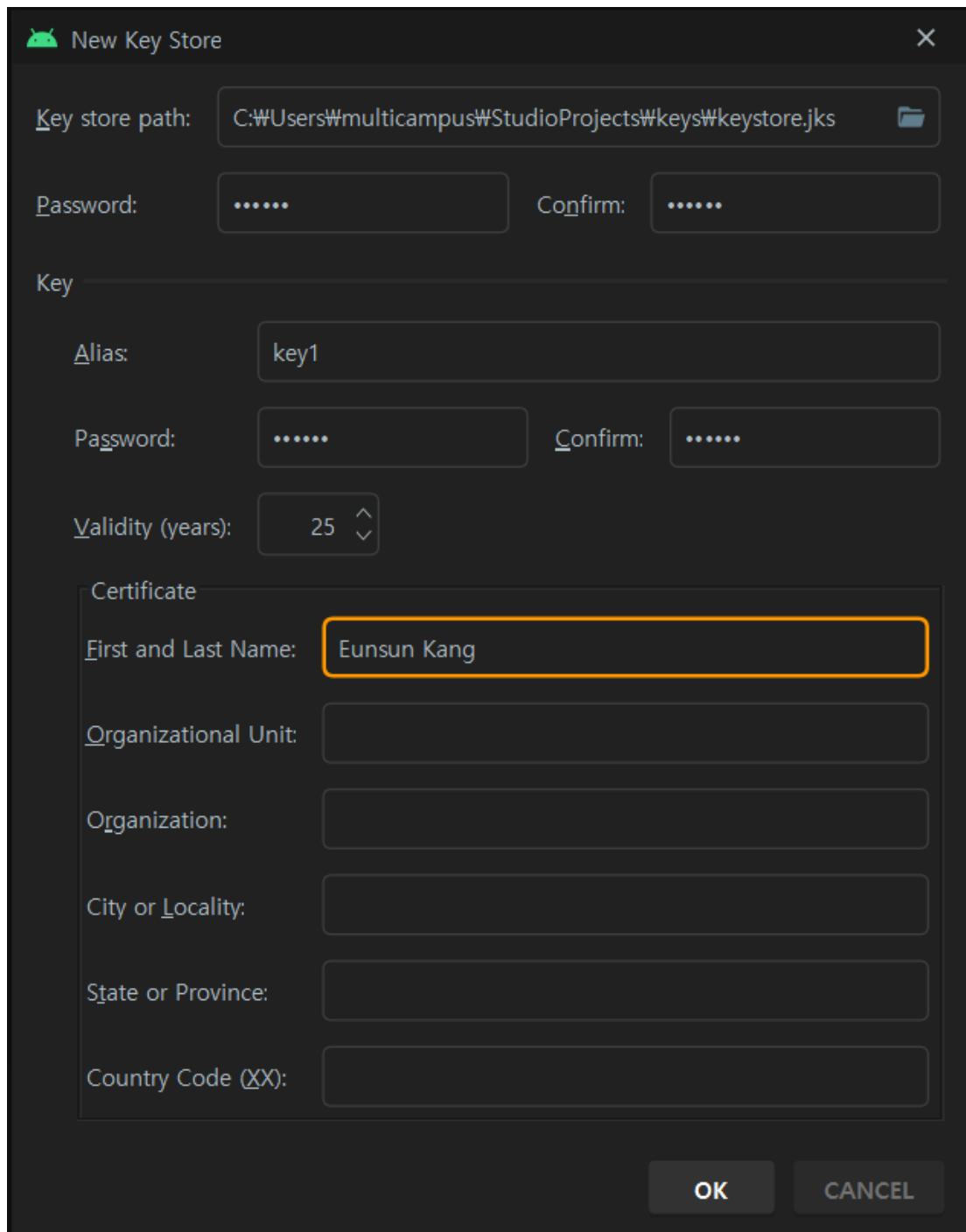
## 8. Key store path의 CREATE NEW 클릭



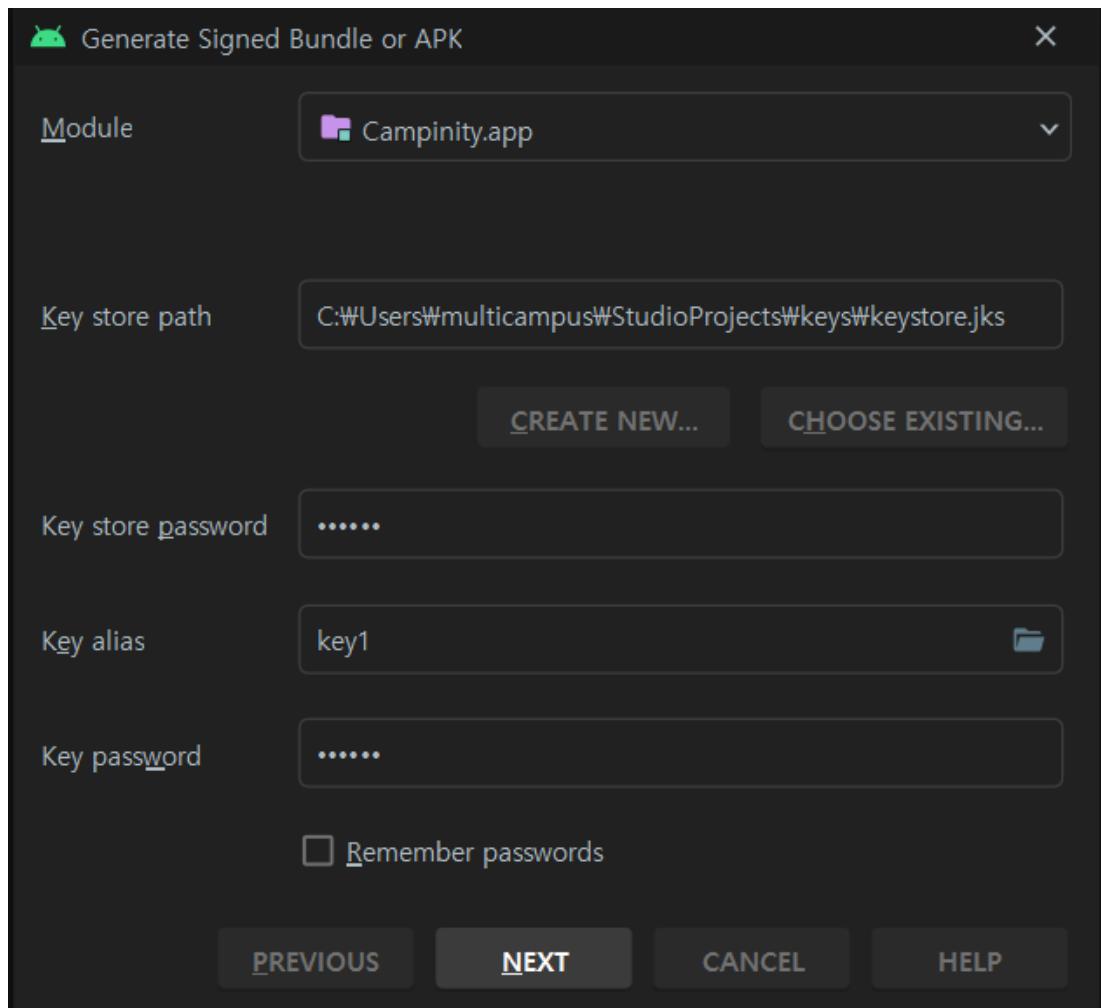
## 9. 저장 위치와 이름 설정 후 OK 클릭



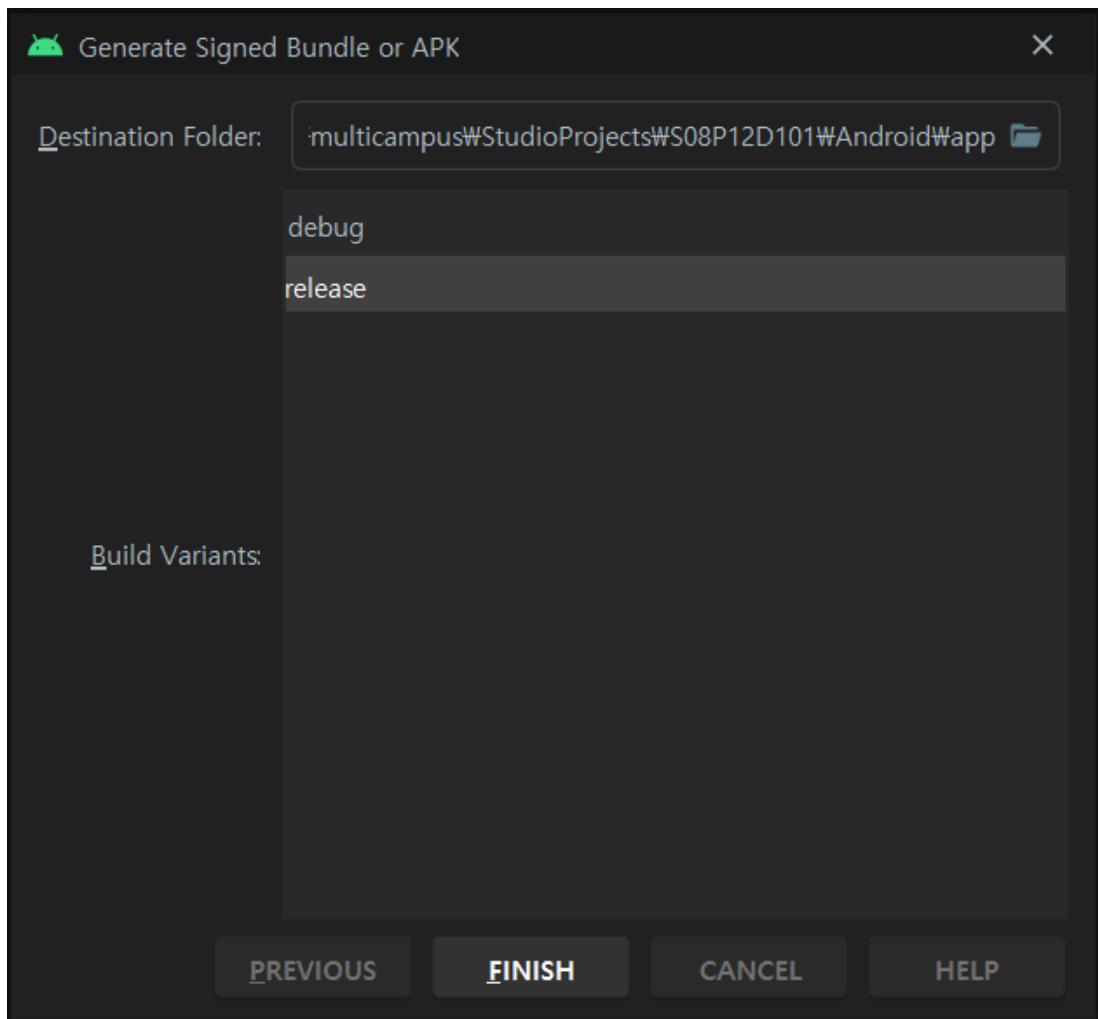
## 10. 나머지 정보 입력 후 OK 클릭



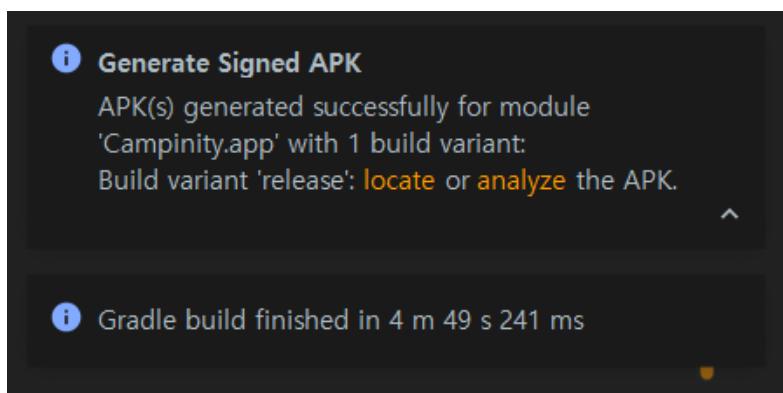
## 11. Next 클릭



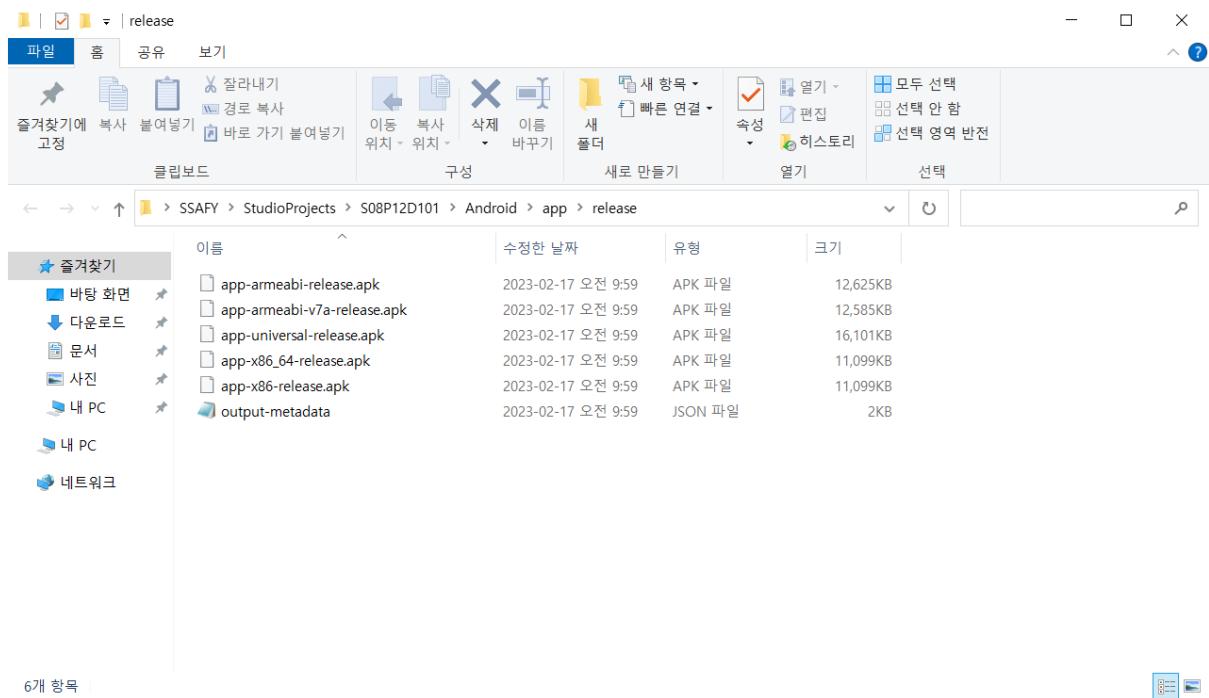
**12. release로 선택 후 FINISH 클릭**



13. 빌드가 완료 된 후 하단에 나타난 popup을 확장하면 빌드된 파일이 있는 locate로 바로 이동 할 수 있다.

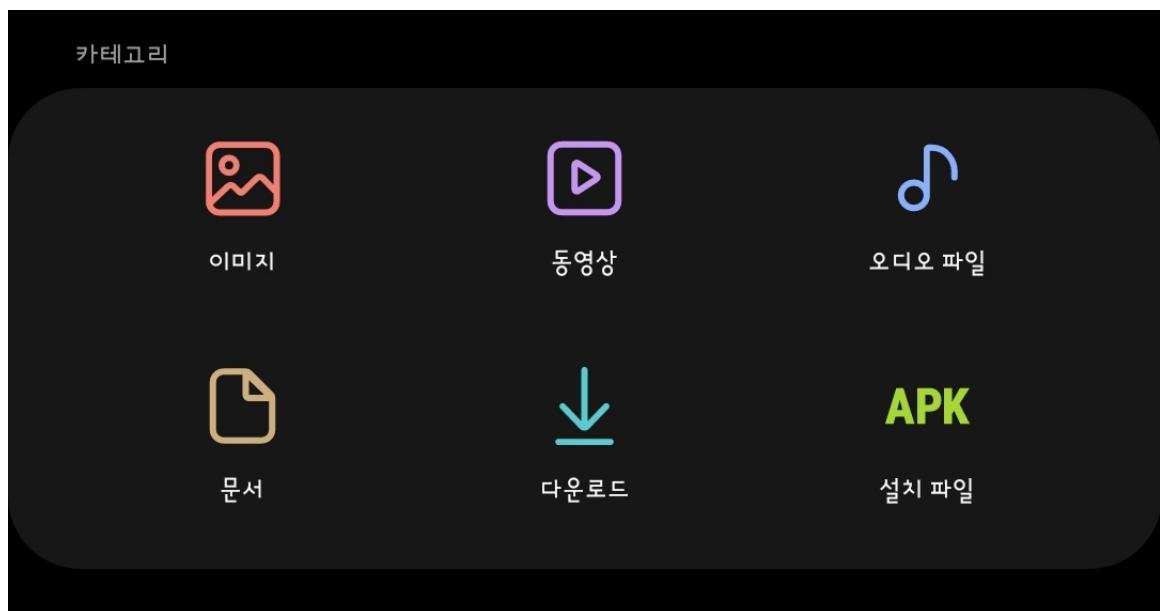


14. 빌드된 파일은 해당 프로젝트의 Android/app/release 폴더 내부에 위치하며, app-x86-release.apk 파일을 이용한다.

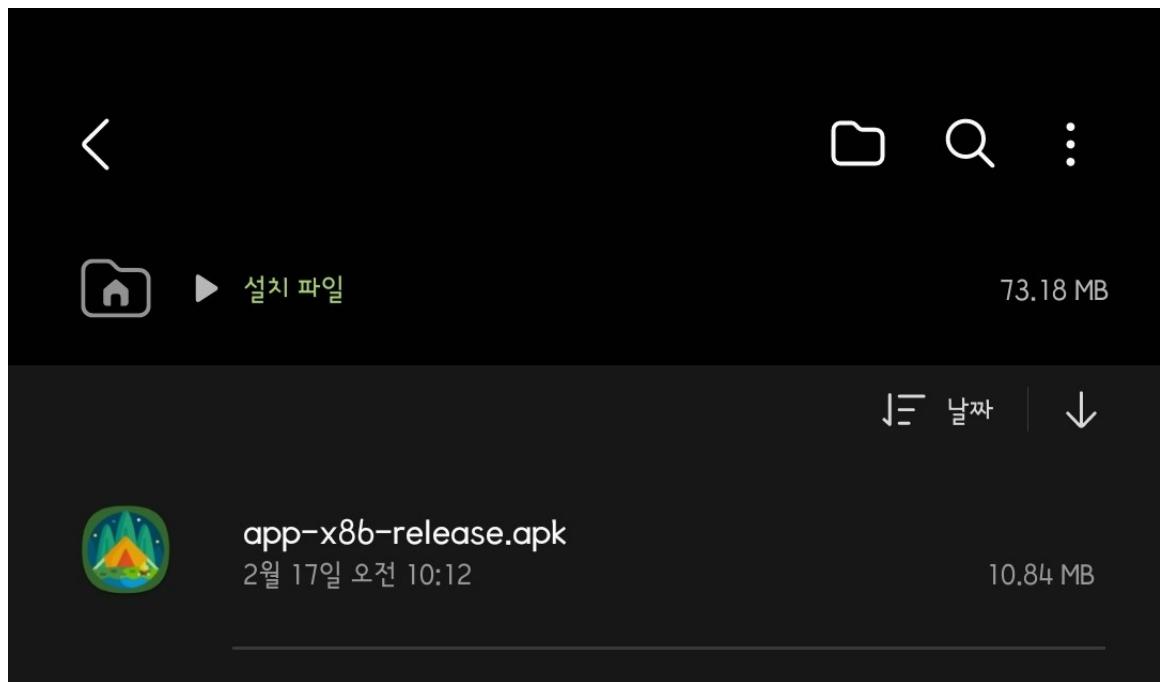


## 15. apk파일을 핸드폰으로 옮긴 후 설치한다.(개인 메신저나 usb 파일 공유 이용)

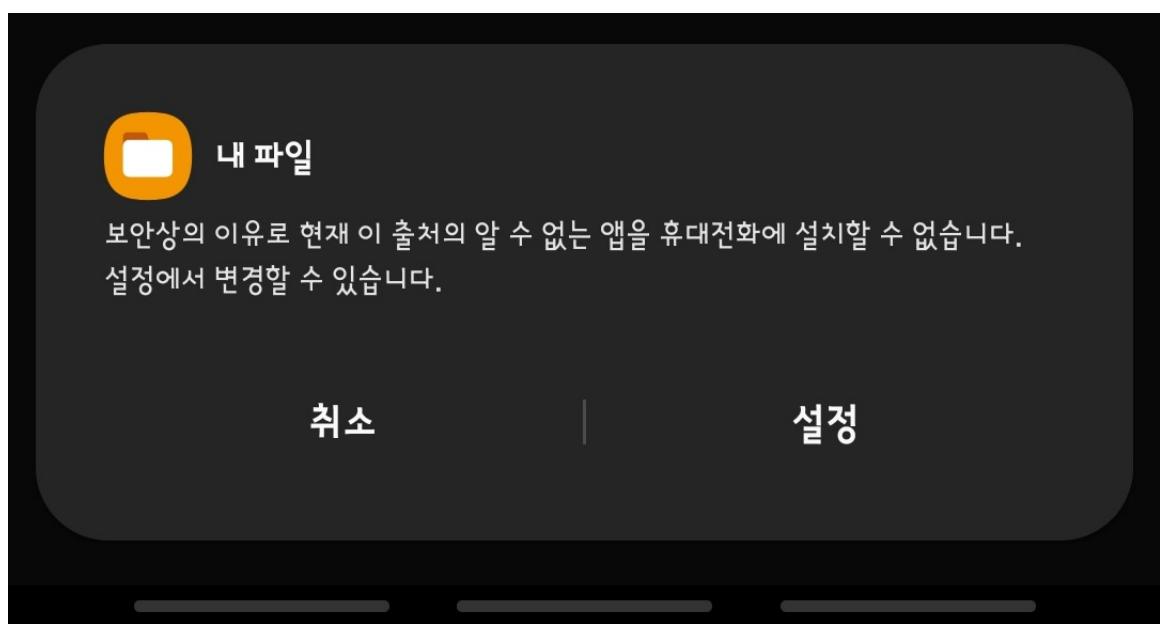
### 1. 폴더의 설치파일 카테고리 선택



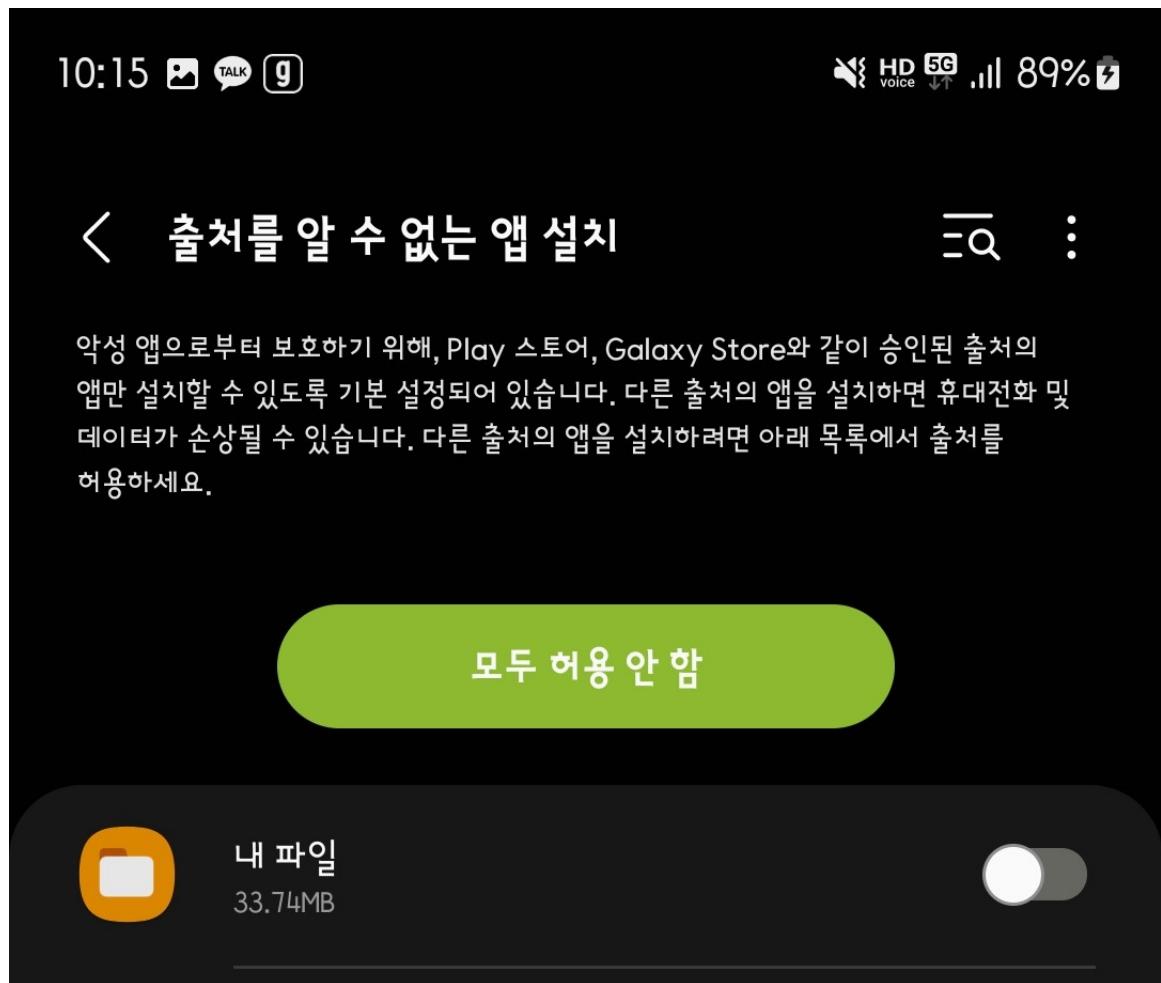
### 2. 설치할 apk 클릭



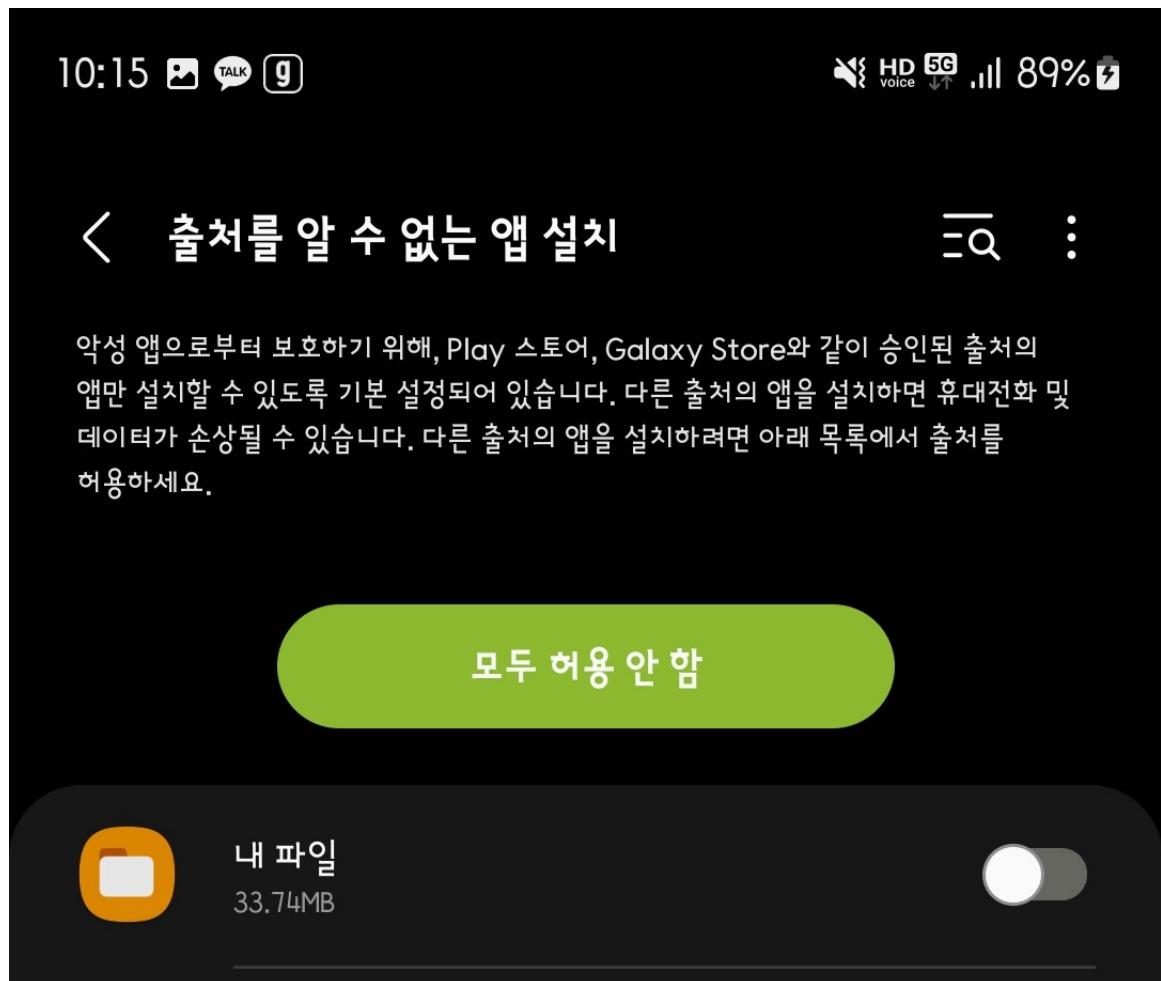
### 3. 출처를 알 수 없는 앱 설치 설정 변경



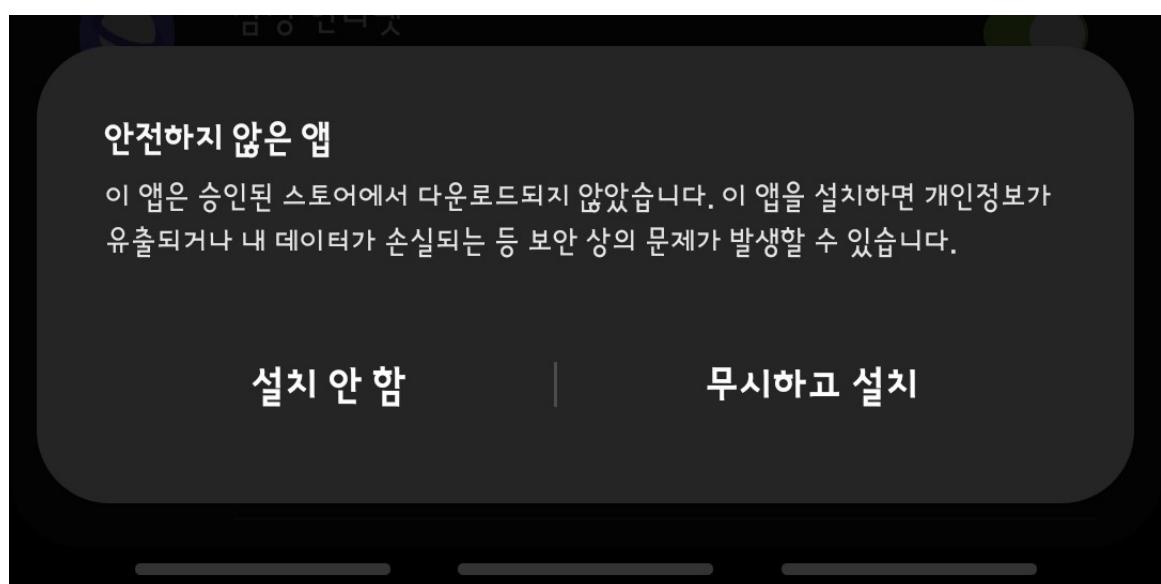
### 4. 내 파일의 설치를 허용(토글 버튼)



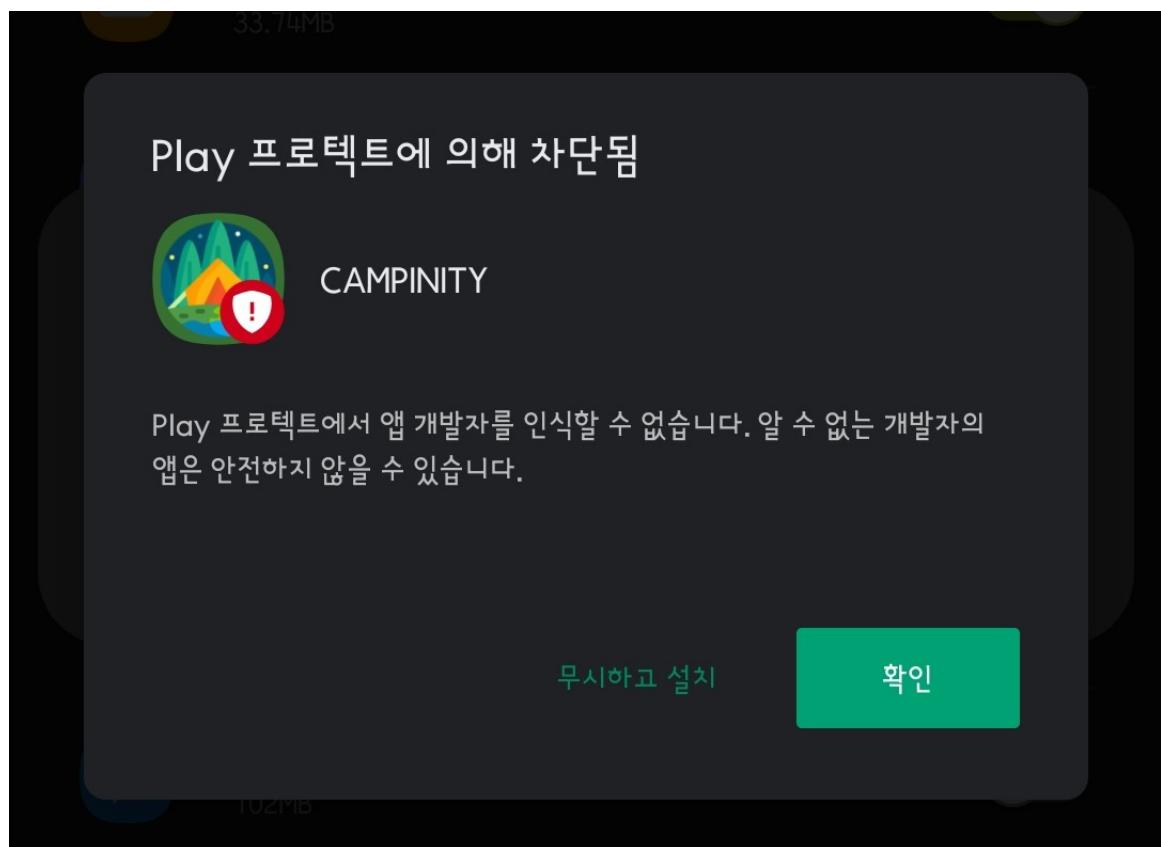
5. 앱 설치 클릭



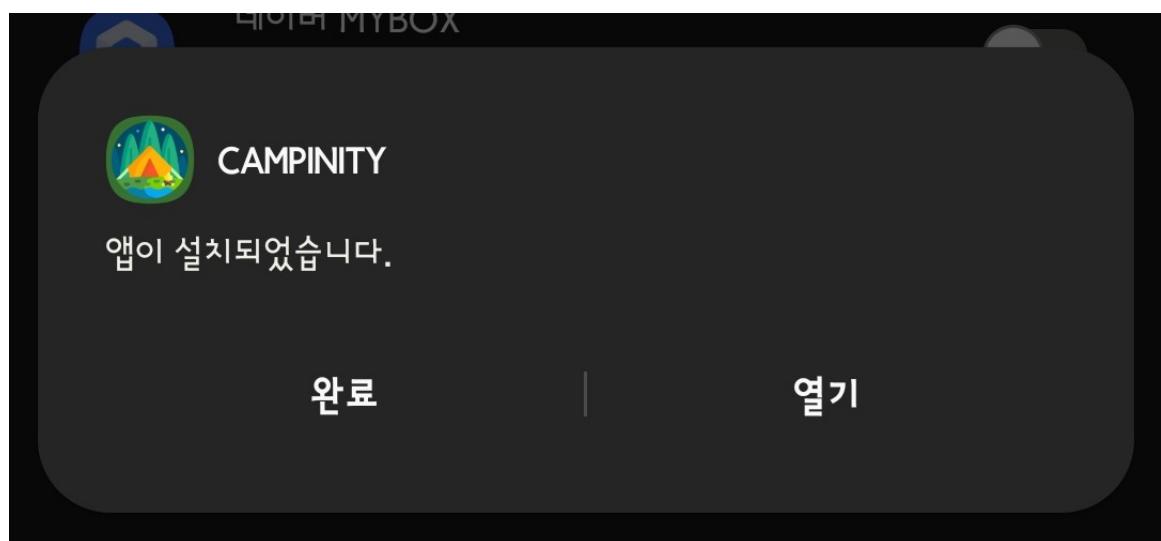
6. 안전하지 않은 앱 경고 무시하고 설치 클릭



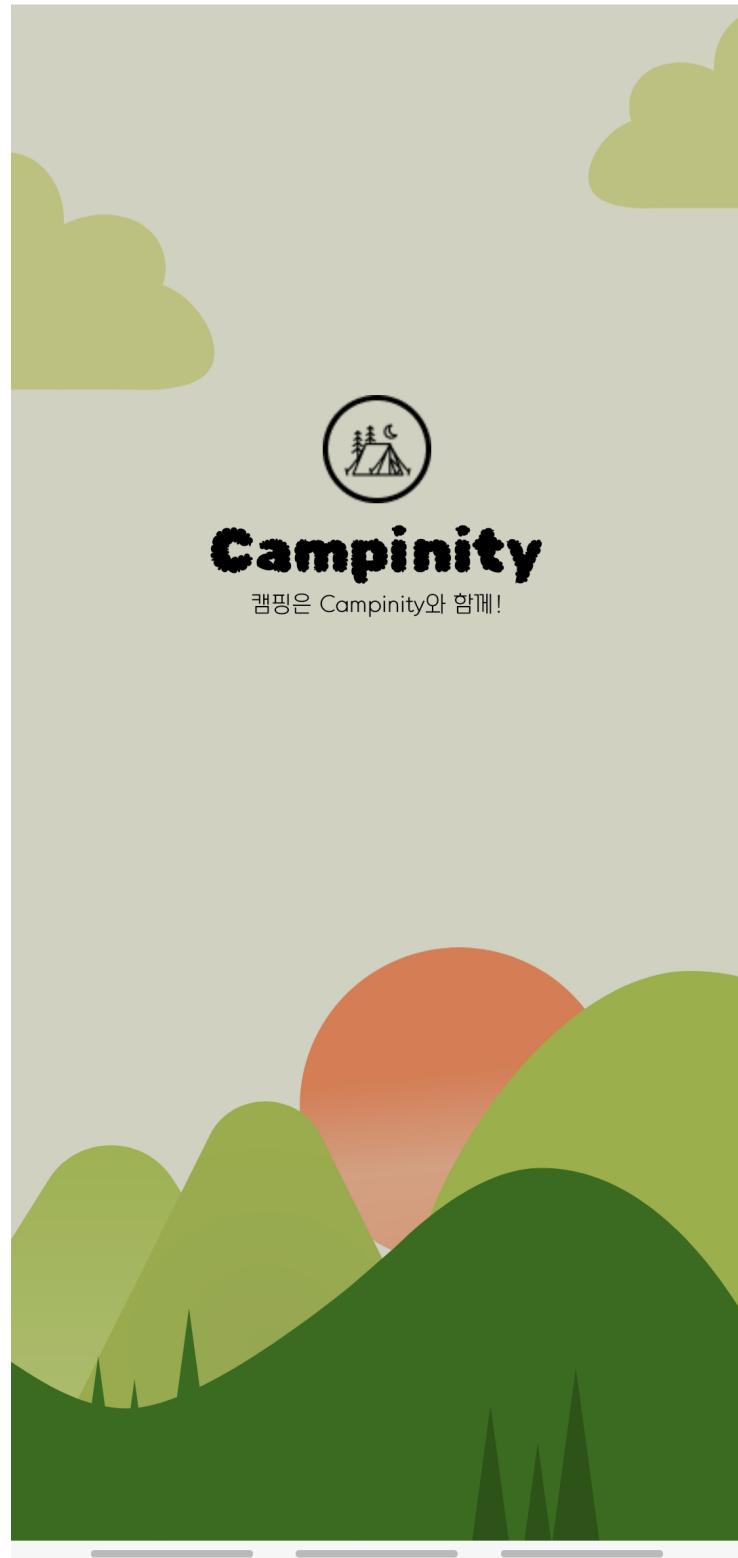
7. 무시하고 설치 클릭



8. 설치가 완료 된 후 앱 열기 클릭



9. 끝





## 7. 시연 시나리오

- 
1. 스플래시
  2. 로그인
    - a. 기존 회원X → 회원가입
    - b. 기존 회원O → 자동 로그인
  3. 홈 화면
    - a. 마이페이지
      - i. 회원정보 수정
      - ii. 내 쪽지 목록
        1. 리뷰
        2. 자유
      - iii. 내 스크랩 목록
        1. 캠핑장
        2. 큐레이션
      - iv. 로그아웃
    - b. 채팅 목록
      - i. 이벤트 성사 되면 채팅방 자동 생성
      - ii. 채팅방 진입
        1. 웹소켓으로 채팅 구현
    - c. 큐레이션 배너
      - i. 아이템 클릭 시 상세 화면으로 이동
        1. 스크랩
      - ii. 큐레이션 전체 목록 화면
        1. 전체

2. 튜토리얼
  3. 레시피
  4. 장소 추천
- d. 검색 모드
- i. 클러스터링
    1. 시, 도 수준
      - a. 캠핑장 개수
    2. 시, 군, 구 수준
      - a. 캠핑장 개수
    3. 상세 수준
      - a. 캠핑장 조회 결과
  - ii. 검색 기능
    1. 이름 검색
    2. 지역 검색
      - a. 각각 선택
      - b. 전체 선택
      - c. 초기화
    3. 필터 검색
      - a. 각각 선택
      - b. 초기화
    4. 현 지도에서 검색
  - iii. 검색 결과 보기 모드
    1. 목록보기
      - a. 페이지네이션
      - b. 캠핑장 간략 정보
        - i. 상세페이지
        - ii. 우체통

### iii. 스크랩

#### 2. 지도보기

##### a. 캠핑장 마커

###### i. 간략 정보 dialog

1. 상세 페이지

2. 우체통

3. 스크랩

##### b. 리뷰 쪽지 마커

###### i. 상세보기

#### iv. 캠핑장 상세 보기 페이지

##### 1. 우체통

###### a. 질문 작성

###### b. 내 질문 조회

##### 2. 스크랩

##### 3. 사진, 설명 등 정보

##### 4. 리뷰

###### a. 작성

###### b. 삭제

###### c. 전체 목록 조회

##### e. 커뮤니티 모드

###### i. 현재 위치 캠핑장 등록

###### 1. 이전 등록 캠핑장 기록 잔여

###### 2. 검색

###### a. 이름으로 검색

###### b. 현재 위치로 검색

###### 3. 캠핑장 구독 토글 버튼으로 구독 설정, 해제 가능.

###### a. 구독 해제시 fcm 관련 기능 전부 사용 불가

ii. 캠핑장 우체통

1. 질문함

a. 상세 조회

b. 답변 작성

2. 내 편지함

a. 상세 조회

b. 답변 작성

iii. 쪽지

1. 리뷰 쪽지

a. 전체 조회 가능

b. 작성

2. 자유 쪽지

a. 100m 거리만 조회 가능

i. 조회 가능 쪽지는 애니메이션 효과 보유

b. 작성

iv. 도움 주기

1. 이벤트 생성

a. 생성 시 해당 캠핑장 내부의 사람들에게 fcm 알림 발신

2. 이벤트 수락

a. 생성된 이벤트 알림 수신자가 선착순으로 수락

b. 수락하면 생성자와 수락자 모두에게 fcm 알림 발신

c. 채팅방 자동 생성 → 채팅 목록에서 확인 가능

v. 사용자 위치 추적 모드

1. ON

a. 사용자 위치를 중심으로 지도가 움직임

b. 사용자가 임의로 지도를 움직여도 다시 위치 중심으로 돌아옴.

2. OFF

a. 사용자 위치가 표시되지만 임의로 지도를 움직여도 위치 중심으로 돌아오지 않음.

f. 나의 컬렉션

i. 전체 목록

1. 생성
2. 상세 조회
  - a. 수정
  - b. 삭제

ii. 보기 모드

1. 카드 뷰 모드
2. 그리드 모드

g. 차트

i. 지금 가장 핫한 캠핑장 TOP 5

1. 캠핑장 클릭 시 상세페이지로 이동

ii. 평점 좋은 캠핑장 TOP 5

1. 캠핑장 클릭 시 상세 페이지로 이동