주차장의 모든것 배포 매뉴얼

본 문서는 안드로이드로 개발된 주차장의 모든것을 사용하기 위한 가이드를 프론트엔드(안드로이드)와 백엔드 스프링, 레디스 서버를 배포하는 과정에 대해 서술하고 있습니다.

# **1. 안드로이드 배포**

## 1.1. 안드로이드 개발 환경

- Android Studio Dolphin (2021.3.1 Patch1)

- targetSDK 32

- minSDK 21

- Kotlin

## 1.2. AOS 기술 스택

- Retrofit 2.9.0 (통신 라이브러리)

- Coroutines(비동기 데이터 처리 라이브러리)

- Glide 4.13.2 (이미지 로드 라이브러리)

- ViewModel-ktx 2.6.0-alpha01

- livedata-ktx 2.6.0-alpha01

- Naver Login(네이버 로그인 라이브러리)

- Kakao Login(카카오 로그인 라이브러리)

- Room (내부 데이터 베이스 라이브러리)

- Naver Map(네이버 지도 SDK)

- Firebase Cloud Messaging(알림을 위한 라이브러리)

- Hilt(종속성 주입 라이브러리)

- Jetpack Navigation(화면 이동 라이브러리)

- Bootpay(결제 라이브러리)

- DrawerLayout(사이드바 라이브러리)

- CoordinatoryLayout(하단 바 라이브러리)

## 1.3. 앱 빌드를 위한 과정

**1.3.1. Android Studio 빌드 방법**

1.3.1.1. local.properties 값 설정

1. Gradle Scripts > local.properties 에

BASE\_URL="http://k8d108.p.ssafy.io:"

NAVER\_CLIENT\_KEY=""

SEARCH\_BASE\_URL = "https://dapi.kakao.com/"

KAKAO\_CLIENT\_KEY=" "

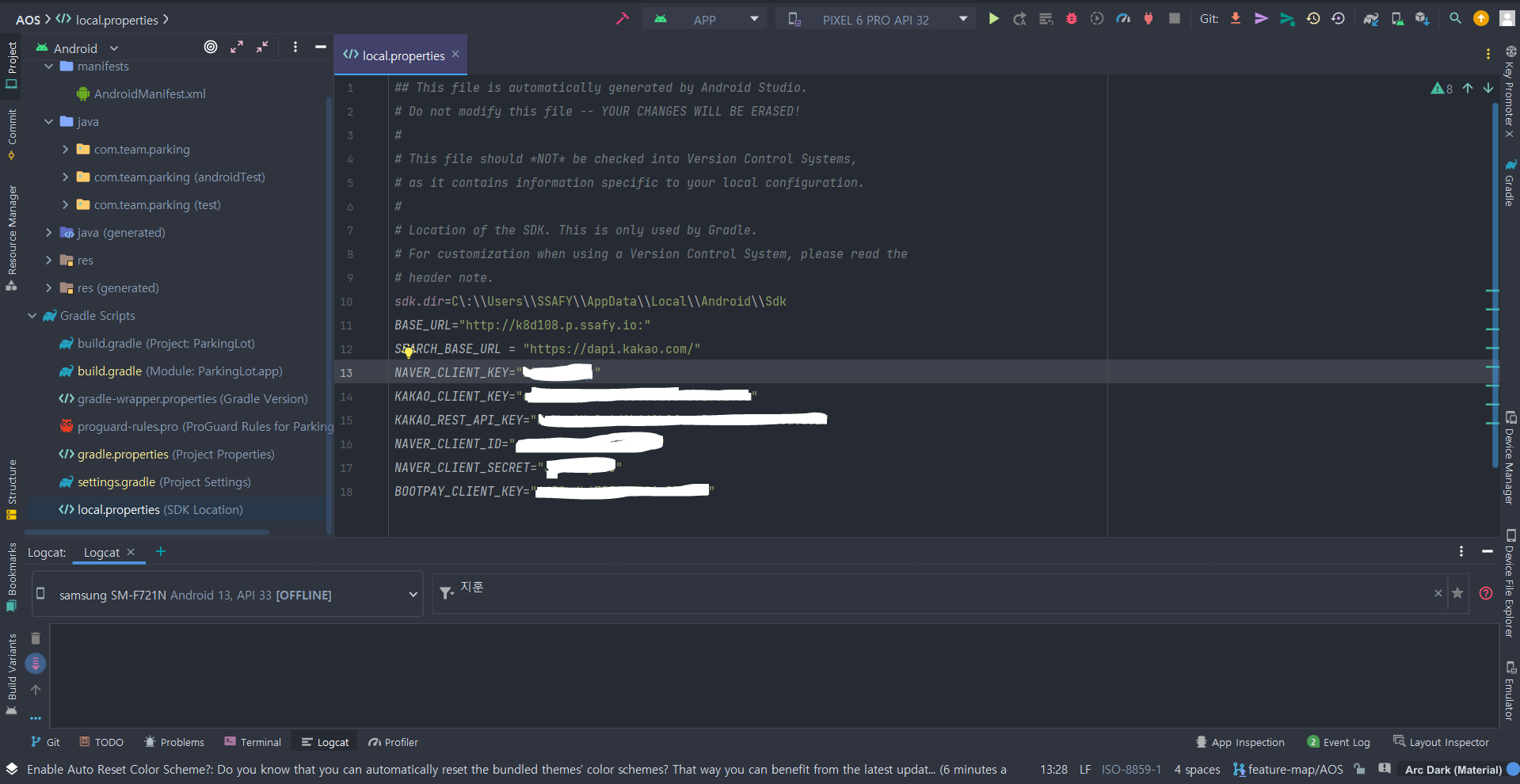
KAKAO\_REST\_API\_KEY=" "

NAVER\_CLIENT\_ID=" "

NAVER\_CLIENT\_SECRET=" "

BOOTPAY\_CLIENT\_KEY=""

를 추가한다.



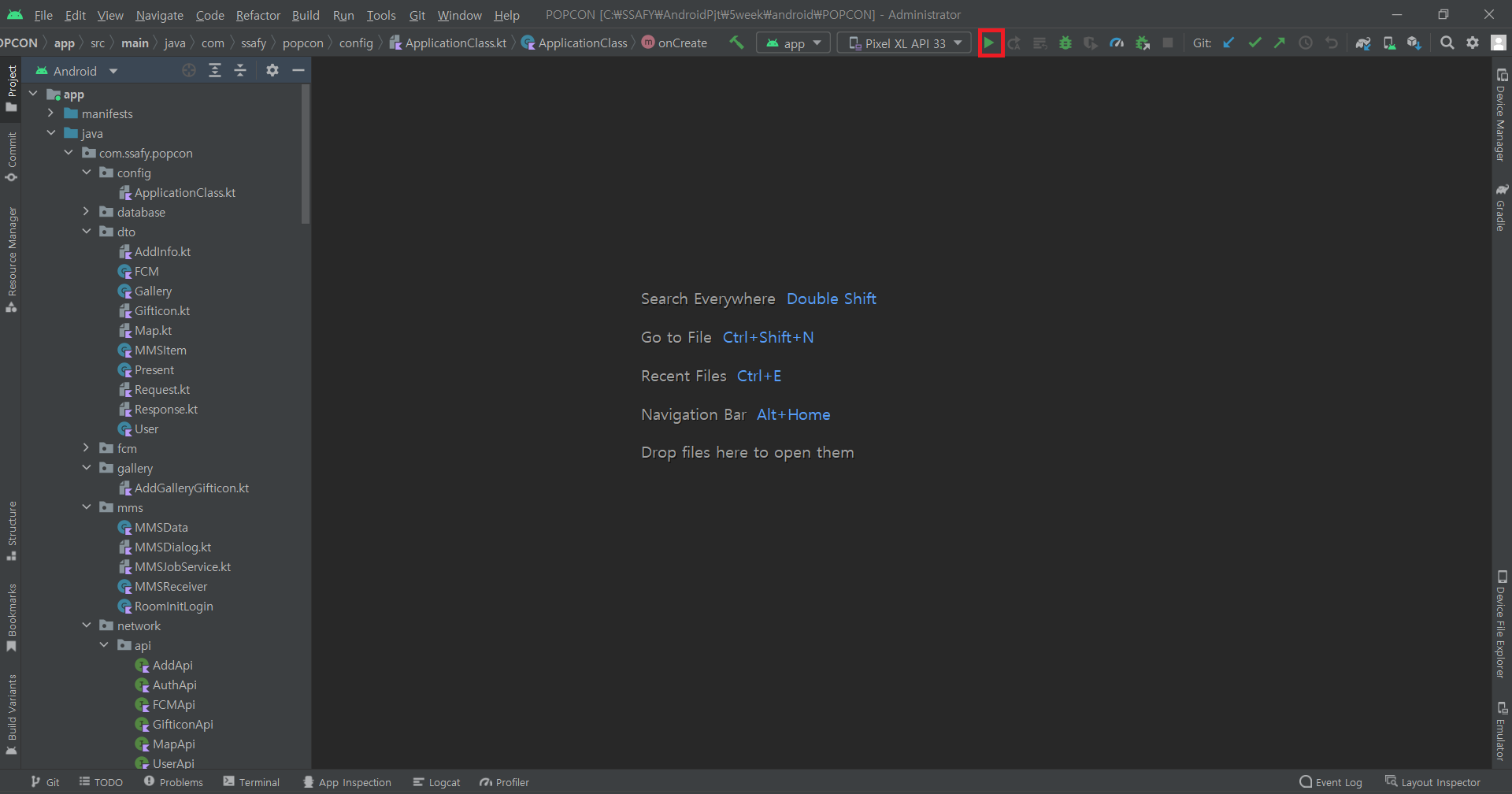
1. BASE\_URL을 제외한 나머지 값들은 [ 외부 서비스 설정 ]을 참고한다.

1.3.1.2. App

1. 스마트폰을 연결한다.
2. 왼쪽 상단에 Run/Debug Configurations이 app으로 선택되어 있는지 확인 후, 스마트폰이 연결 되어있는지 확인한다.



1. 녹색 버튼을 눌러 앱을 설치 및 실행한다.

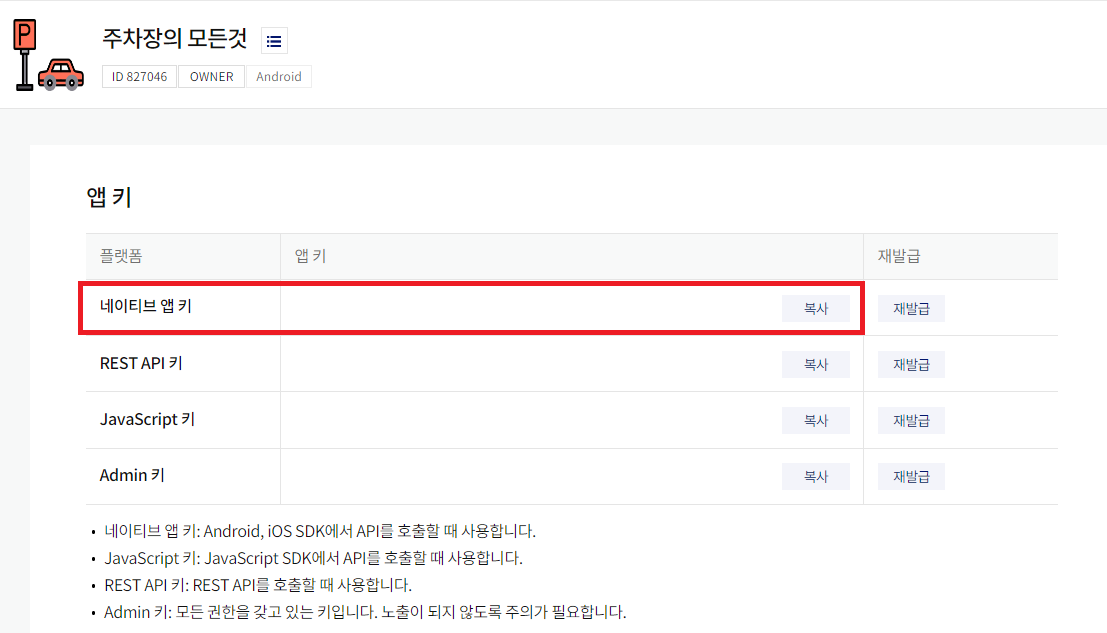


**1.3.2. 외부 서비스 설정**

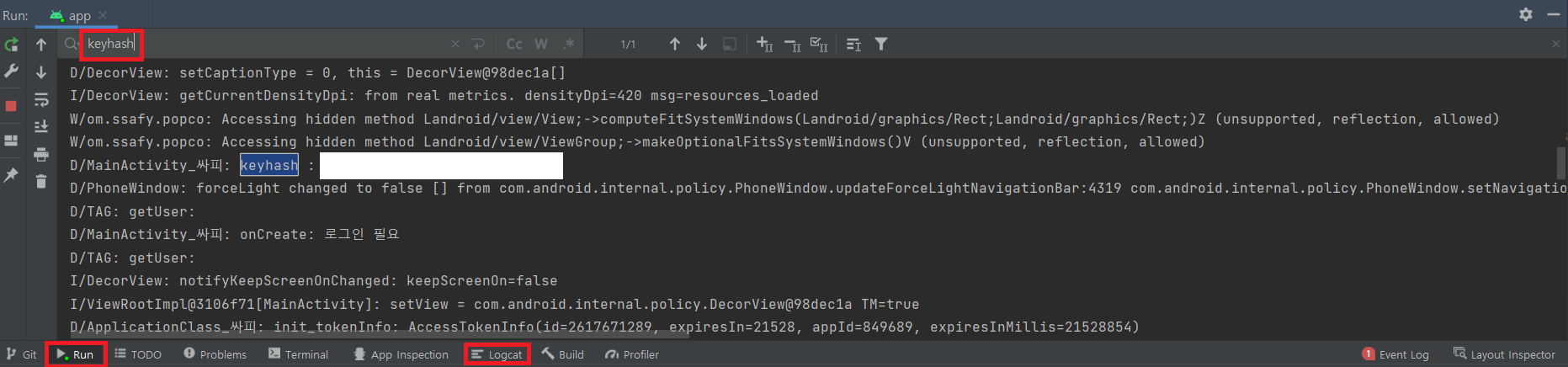
1.3.2.1. Kakao Dev 설정 (소셜 로그인)

1. 내 애플리케이션 > 앱 설정 > 앱 키에 들어가서 네이티브 앱 키를

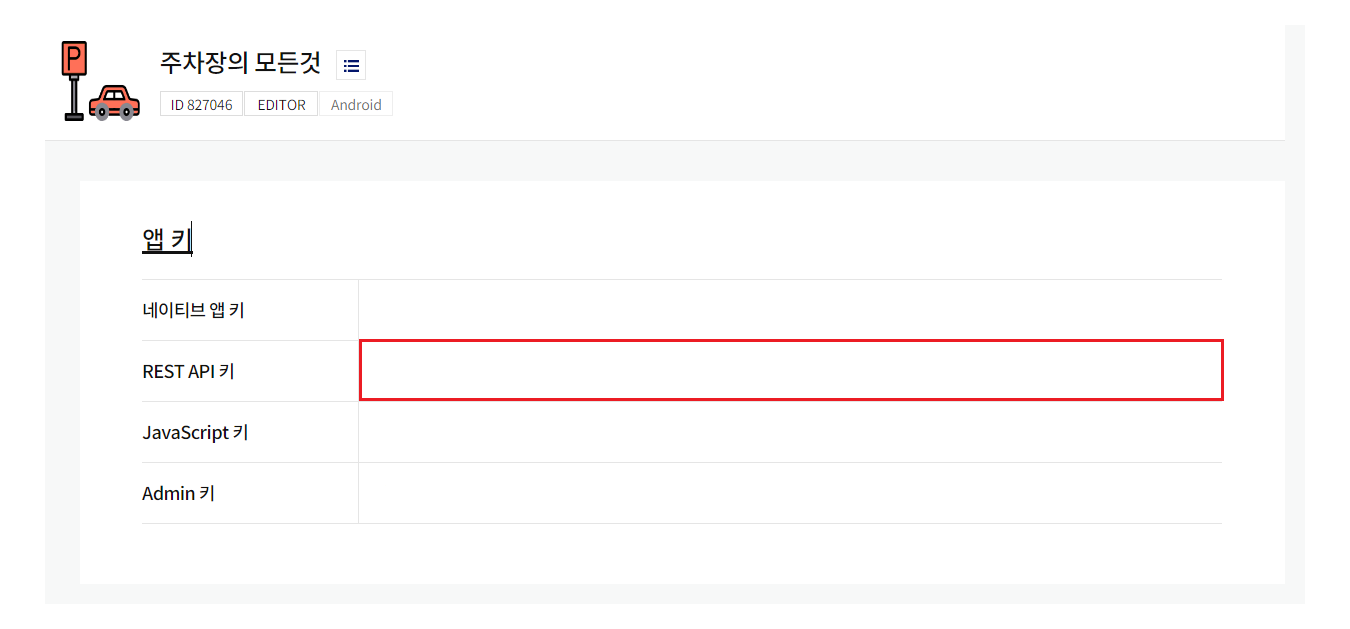
안드로이드 스튜디오의 Gradle Scripts > local.properties의 ‘KAKAO\_CLIENT\_KEY ‘에 추가한다.



1. 앱을 실행한다.
2. Run 또는 Logcat에 keyhash를 검색하여 복사한다.



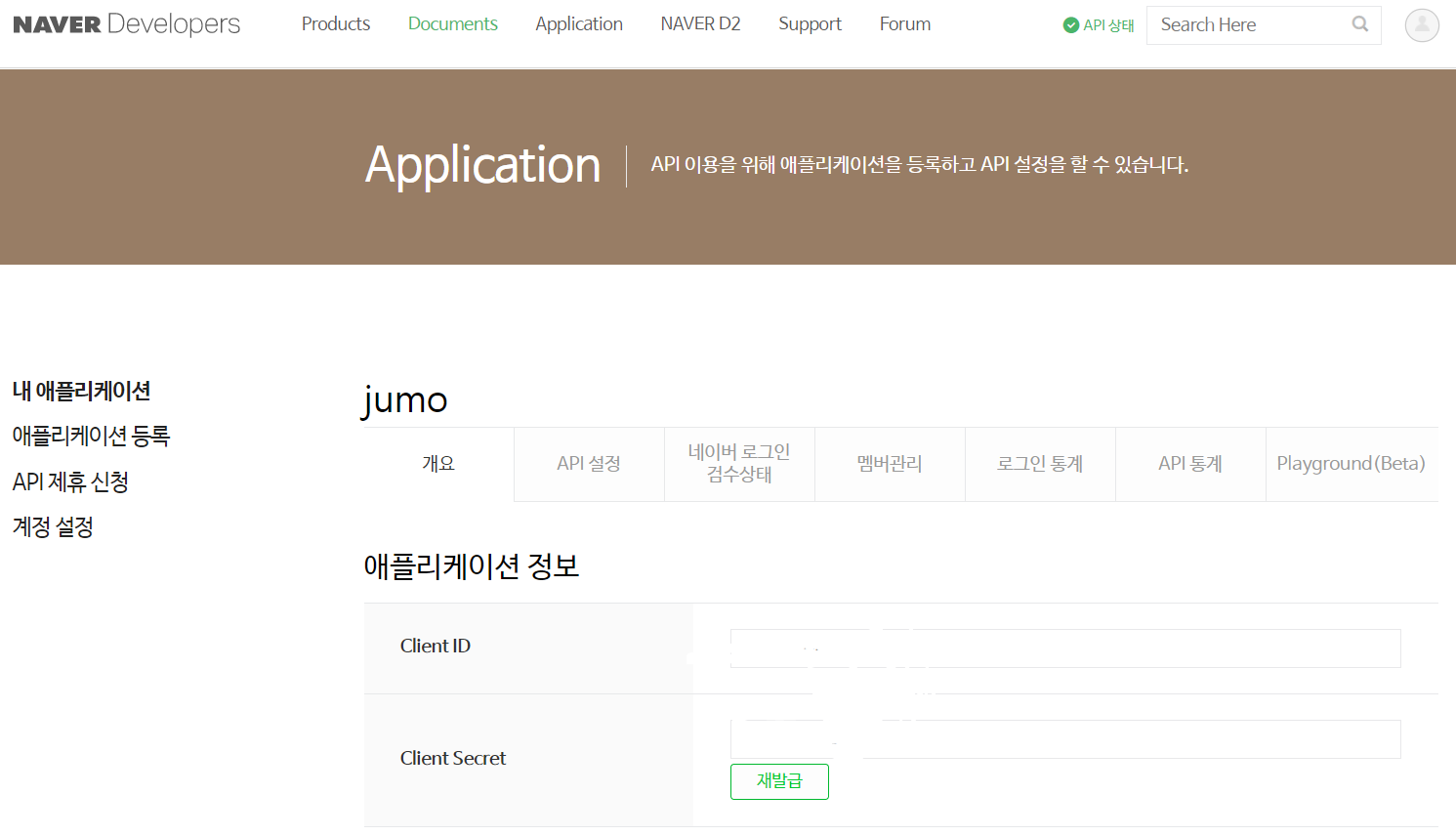
1. 내 애플리케이션 > 앱 설정 > 플랫폼에 들어가서 Android의 수정 버튼을 클릭 후 키 해시에 (3)의 값을 붙여넣는다.



1. 안드로이드 스튜디오의 Gradle Scripts > local.properties의 ‘KAKAO\_REST\_API\_KEY ‘에 추가한다.
2. 추가할때는 “KakaoAK {REST API 키}”형식으로 한다.

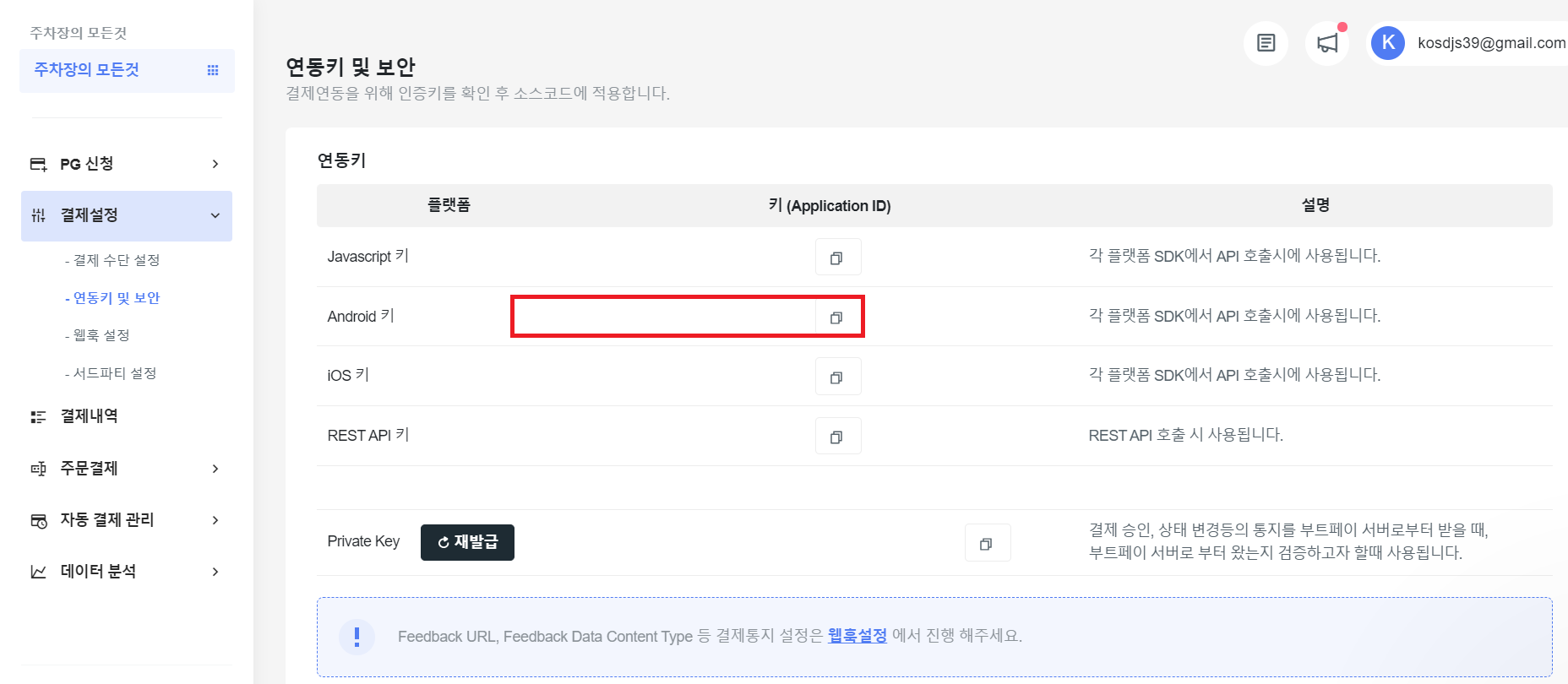
1.3.2.2. Naver Dev 설정 (소셜 로그인)

1. Application > 내 애플리케이션 > 개요 > 애플리케이션 정보에 들어간다.



1. Client ID를 복사하여 안드로이드 스튜디오의 Gradle Scripts > local.properties의 ‘NAVER\_CLIENT\_ID’에 붙여 넣는다.
2. Client Secret를 복사하여 안드로이드 스튜디오의 Gradle Scripts > local.properties의 ‘NAVER\_CLIENT\_SECRET’에 붙여 넣는다.

1.3.2.3. BootPay 설정 (결제 시스템)

1. 프로젝트 > 결제설정 > 연동키 및 보안에 들어간다.  
   
2. Android 키를 복사하여 안드로이드 스튜디오의 Gradle Scripts > local.properties의 ‘BOOTPAY\_CLIENT\_KEY’에 붙여 넣는다.

# **2. API 서버 배포 과정**

## 2.1. 개발환경

- JAVA : Oracle Java 11.0.16

- Spring Boot 2.7.9

- IDE : Intelli J 2022.3

-빌드 도구 : (스프링) Jenkins 와 docker

Build .gradle 주요 종속 라이브러리

classpath "org.springframework.boot:spring-boot-gradle-plugin:$**{**springBootVersion**}**"  
classpath "io.spring.gradle:dependency-management-plugin:1.0.11.RELEASE"  
implementation 'com.google.firebase:firebase-admin:9.1.1'  
implementation 'org.springframework.boot:spring-boot-starter-webflux'  
implementation group: 'org.springframework.cloud', name: 'spring-cloud-gcp-starter', version: '1.2.5.RELEASE'  
implementation group: 'org.springframework.cloud', name: 'spring-cloud-gcp-storage', version: '1.2.5.RELEASE'  
runtimeOnly 'com.mysql:mysql-connector-j'  
implementation group: 'com.googlecode.json-simple', name: 'json-simple', version: '1.1.1'  
implementation group: 'com.google.code.geocoder-java', name: 'geocoder-java', 'version': '0.16'  
api 'org.springframework.boot:spring-boot-starter-data-redis'  
implementation 'org.springframework.boot:spring-boot-starter-jdbc'  
implementation 'org.springframework.boot:spring-boot-starter-validation'  
implementation 'org.springframework.boot:spring-boot-starter-web'  
compileOnly 'org.projectlombok:lombok'  
developmentOnly 'org.springframework.boot:spring-boot-devtools'

## 2.2. 배포 환경

배포 서버 : Ubuntu 20.04.4 LTS

Docker : 23.0.1

DB : Mysql - latest

Nosql : redis alpine

Docker Image 목록

**Portainer : 9000 / 컨테이너 이미지 gui 관리**

**Jenkins : 8080 / 자동화 및 배포**

**Jumo-notification : 8083 / 알림**

**Jumo-parking : 8082 / 주차장 정보 및 지도 정보**

**Jumo-user : 8081/ 유저 및 프로필**

**Mysql-container : 3306/ RDB**

**Nginx : 80/ 무중단 배포를 위해 로드 밸런서 역할로 사용하려고 했으나 x**

**Redis : 6379 / 알림 및 token 정보 관리를 위한 nosql**

## 2.3. 서버 배포를 위한 과정

**스프링**

1. https://lab.ssafy.com/s08-final/S08P31D108.git 를 clone 한다.

2. develop/BE 브랜치에 머지 리퀘스트를 보낸다.

3. 머지 되는 순간 주요 웹 서버들이 빌드 된다.

4. 수동 배포를 위해선 각 소스코드의 gradlew 페이지가 있는 위치의 각 이미지 (모듈 ) 별로 다음과 같은 명령어를 입력하면 된다.

# user 모듈 배포 예시

**./gradlew :jumo-user:build**

**Sudo docker stop user-develop**

**Sudo docker rm user-develop**

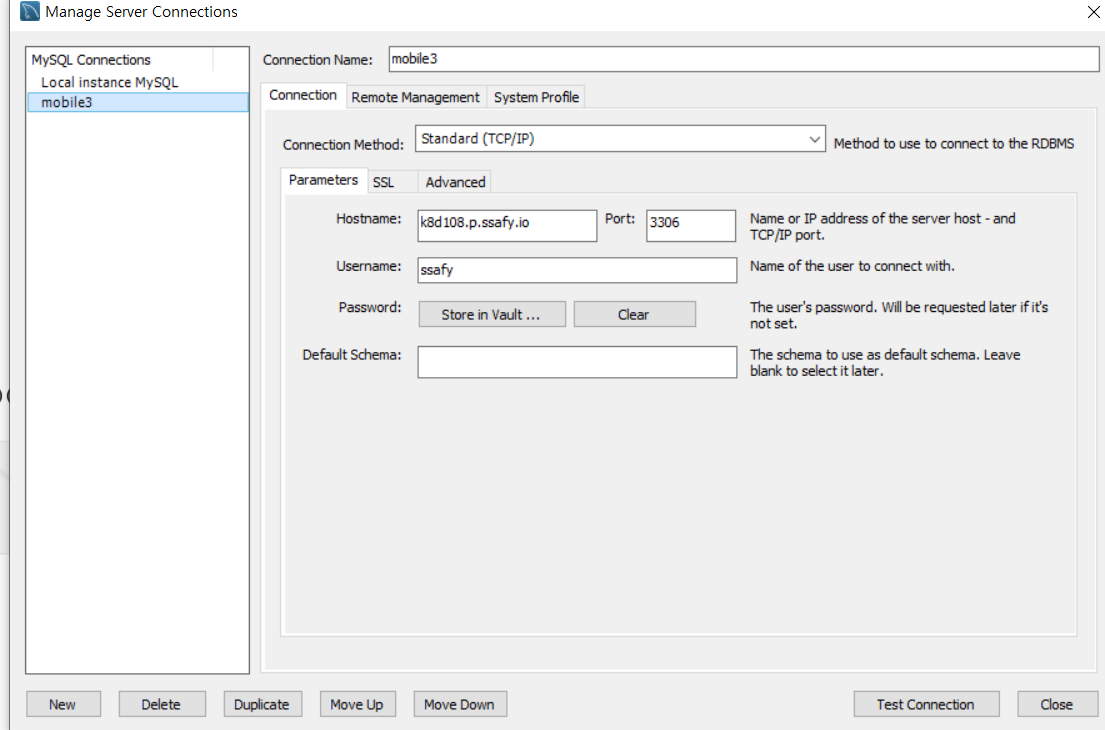
**Sudo docker build -t jumo:user.**

**Sudo docker run -d -p 8081:8081 --name user-develop jumo:user**

4. 관리자 페이지는 따로 없다.

## 2.4. DB 접속을 위한 세팅 정보

* Mysql 최신버전 이미지
* Workbench를 통한 connection 정보

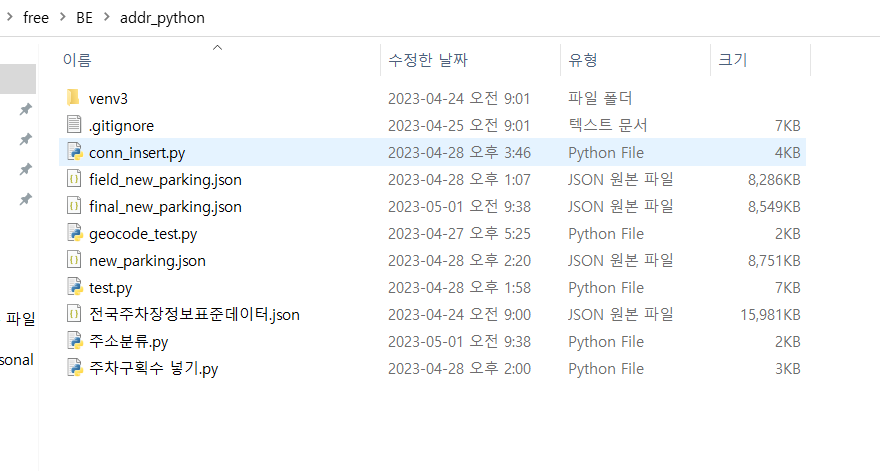


* 계정정보는 ID : ssafy PW: saffy
* Default Database 는 jumo

## 2.5. 배포시 특이사항

- DB 에 1만 개의 주차장 데이터를 insert 하여야 정상적으로 앱을 사용해볼 수 있다.

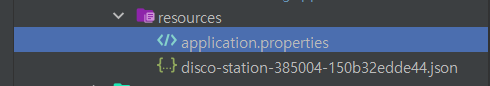
- 관련 DB 전처리 및 주입을 위한 코드 들은 다음과 위치에 있다.



- http://k8d108.p.ssafy.io:8081/swagger-ui.html 등 8081 8082 8083 포트를 통해 swagger 를 확인 할 수 있다.

- 커뮤니티 이미지 저장을 위해 구글 cloud storage 를 저장공간으로 사용하고 있다.

- 해당 연동을 위한 계정 정보 파일의 소스 코드에서의 위치는 다음과 같다.

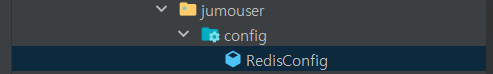


Application.properties 내용

spring.cloud.gcp.storage.credentials.location=classpath:disco-station-385004-150b32edde44.json  
spring.cloud.gcp.storage.project-id=disco-station-385004

-user 및 알림과 관련하여 redis 관련 설정이 존재한다.

spring.cache.type= *redis*spring.cache.redis.time-to-live=3600  
spring.cache.redis.cache-null-values = true  
spring.cache.redis.host = k8d108.p.ssafy.io  
spring.cache.redis.port = 6379



소스코드의 Config 와 application property 에 해당 설정이 존재한다