

목차

I. 빌드 및 배포

1. 개발 환경
2. 설정 파일 목록
3. 설정 파일 상세
4. Ubuntu 패키지 설치
5. FE / BE 빌드
6. OpenVidu 배포
- 7.

II. 외부 서비스

1. 구글 로그인
2. 네이버 로그인
3. Teachable Machine
4. 네이버 이메일 발송

I. 빌드 및 배포

1. 개발 환경

- Server: AWS EC2 Ubuntu 20.04 LTS
- JAVA JDK: 11.0.15.1
- MySQL: 8.0.29
- Redis: 3.0.504
- SpringBoot: 2.7.1
- Node.js: 16.16.0 LTS
- IntelliJ Ultimate: 2022.1.3
- VSCode: Stable Build
- Hibernate: 5.6.9 Final
- Vue.js: 2.7.7
- Vuetify: 2.6.7
- OpenVidu: 2.22.0
- Nginx: 1.18.0
- Docker: 20.10.17
- Teachable Machine

2. 설정 파일 목록

<SpringBoot>

- application.properties
- application-oauth.properties

<Nginx>

/etc/nginx/sites-available/default

3. 설정 파일 상세

<SpringBoot>

- application.properties

Port

server.port=8082

Redis

spring.redis.host=localhost

spring.redis.port=6379

Swagger

spring.mvc.pathmatch.matching-strategy=ant_path_matcher

MySQL

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

DB url

spring.datasource.url=jdbc:mysql://i7a802.p.ssafy.io:3306/jpa?characterEncoding=UTF-8&serverTimezone=UTC&

#spring.datasource.url=jdbc:mysql://localhost:3306/jpa?characterEncoding=UTF-8&serverTimezone=UTC&

DB username

spring.datasource.username=계정명

DB password

spring.datasource.password=비밀번호

JPA options

spring.jpa.show-sql=true

spring.jpa.hibernate.ddl-auto=none

spring.jpa.properties.hibernate.format_sql=true

spring.jpa.hibernate.database=mysql

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect

spring.jpa.properties.hibernate.default_batch_fetch_size = 1000

multipart - file upload

#spring.servlet.multipart.location=C:\Temp\upload

spring.servlet.multipart.max-file-size=10MB

spring.servlet.multipart.max-request-size=50MB

file.upload.dir.window=C:\Temp\upload

file.upload.dir.linux=/home/ubuntu/upload

#mail

spring.mail.host=smtp.naver.com

spring.mail.port=465

spring.mail.username=kimsm9822

spring.mail.password=FSS9R491S7UF

spring.mail.properties.mail.smtp.auth=true

spring.mail.properties.mail.smtp.ssl.enable=true

spring.mail.properties.mail.smtp.ssl.trust=smtp.naver.com

#oauth

spring.profiles.include=oauth

- application-oauth.properties

#OAuth

spring.security.oauth2.client.registration.google.client-id=531257147697-
gedu73e4t7ugg9dt0vhd5k9ddla98akg.apps.googleusercontent.com

spring.security.oauth2.client.registration.google.client-secret=GOCSPX-
CdmpYzJ68t6tloCEZulfxNR7Burn

spring.security.oauth2.client.registration.google.scope=profile,email

#spring.security.oauth2.client.registration.google.redirect-
uri=http://localhost:8082/api/auth/login/google/callback

#spring.security.oauth2.client.registration.google.redirect-
uri=http://localhost:8080/auth/google/redirect

spring.security.oauth2.client.registration.google.redirect-
uri=https://i7a802.p.ssafy.io/auth/google/redirect

spring.security.oauth2.client.registration.naver.client-id=YTUxbIXLw0BuEFXiYUFV

spring.security.oauth2.client.registration.naver.client-secret=DlnbN38WFt

#spring.security.oauth2.client.registration.naver.redirect-
uri=http://localhost:8080/auth/naver/redirect

spring.security.oauth2.client.registration.naver.redirect-
uri=https://i7a802.p.ssafy.io/auth/naver/redirect

spring.security.oauth2.client.registration.naver.authorization_grant_type=authorization_code

spring.security.oauth2.client.registration.naver.scope=name,email,profile_image

spring.security.oauth2.client.registration.naver.client-name=Naver

spring.security.oauth2.client.provider.naver.authorization_uri=https://nid.naver.com/oauth2.0/authorize

spring.security.oauth2.client.provider.naver.token_uri=https://nid.naver.com/oauth2.0/token

spring.security.oauth2.client.provider.naver.user-info-uri=https://openapi.naver.com/v1/nid/me

spring.security.oauth2.client.provider.naver.user_name_attribute=response

<Nginx>

Default server configuration

```
server {  
    server_name i7a802.p.ssafy.io;  
    location / {  
        root /home/ubuntu/dist;  
        try_files $uri $uri/ /index.html =404;  
    }  
    location /api {  
        proxy_pass http://localhost:8082/api;  
    }  
    location /gan {  
        proxy_pass http://localhost:8083/gan;  
    }  
  
    listen [::]:443 ssl ipv6only=on; # managed by Certbot  
  
    listen 443 ssl; # managed by Certbot
```

```

ssl_certificate /etc/letsencrypt/live/i7a802.p.ssafy.io/fullchain.pem; # managed by Certbot

ssl_certificate_key /etc/letsencrypt/live/i7a802.p.ssafy.io/privkey.pem; # managed by Certbot

include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot

ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

}

server {

    if ($host = i7a802.p.ssafy.io) {

        return 301 https://$host$request_uri;

    } # managed by Certbot


    listen 80 default_server;

    listen [::]:80 default_server;


    server_name i7a802.p.ssafy.io;

    return 404; # managed by Certbot

}

```

4. 우분투 패키지 설치

본 프로젝트는 수동배포로 진행하여 과정이 다소 복잡합니다.

<JDK>

```
$ sudo apt-get update
```

```
$ sudo apt-get install openjdk-11-jdk
```

설치 후

```
$ java -version
```

<nginx>

```
$ sudo apt install nginx
```

설치 후 3 번의 설정 파일처럼 서버 이름 설정 후 인증서 발급 필요

<certbot>

repository 에 certbot 설치

```
$ sudo add-apt-repository ppa:certbot/certbot
```

python-certbot-nginx 설치

```
$ sudo apt-get install python-certbot-nginx
```

SSL 인증서 생성하고 적용하기

```
$ sudo certbot --nginx
```

<docker>

업데이트 및 HTTP 패키지 설치 Permalink

```
$ sudo apt update
```

```
$ sudo apt-get install -y ca-certificates ₩
```

```
curl ₩
```

```
software-properties-common ₩
```

```
apt-transport-https ₩
```

```
gnupg ₩
```

```
lsb-release
```

GPG 키 및 저장소 추가 Permalink

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
```

<redis>

#설치


```
$ sudo docker pull redis:latest
```

```
#도커에 네트워크 생성
```

```
$ sudo docker network create redis-net
```

```
# 볼륨 잡아서 컨테이너 실행
```

```
$ sudo docker run --name redis -p 6379:6379 --network redis-net -v /home/ubuntu/redisvolume -  
d redis:latest redis-server --appendonly yes
```

```
<mysql>
```

```
$ sudo docker pull mysql:8.0.29
```

```
$ sudo docker run -d -p 3306:3306 -v /home/ubuntu/mysql_data:/var/lib/mysql -e  
MYSQL_ROOT_PASSWORD=루트비밀번호 --name mysql-container mysql:8.0.29
```

```
# mysql 컨테이너 접속
```

```
$ sudo docker exec -it mysql-container bash
```

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

```
$ 패스워드 입력
```

```
# 접속 후 계정 생성
```

```
$ mysql> CREATE USER 계정명@'%' identified by '비밀번호';
```

```
$ mysql> GRANT ALL PRIVILEGES ON *.* to 계정명@'%';
```

```
$ mysql> FLUSH PRIVILEGES;
```

```
$ mysql> exit;
```

```
# 이후 workbench 로 접속하여 'jpa' 스키마 생성, dump.sql 실행
```

```
<GAN 가상 시창 기능용 서버를 위한 패키지 설치>
```

```
# 이 기능을 사용하려면 설치해야 하지만 실험적인 기능이고 설치가 다소 복잡하여 필수사항은  
아닙니다.
```

```
<아나콘다 설치>
```

```
$ wget https://repo.anaconda.com/archive/Anaconda3-2019.10-Linux-x86\_64.sh
```

```
# 배쉬 진입 후 설치 진행
```

```
$ bash Anaconda3-2019.10-Linux-x86_64.sh
```

```
# 배쉬로 진입
```

```
$ vi ~/.bashrc
```

```
# 진입 후 아래 내용 추가
```

```
$ export PATH="/home/username/anaconda3/bin:$PATH"
```

```
# 이후 .bashrc 실행
```

```
$ source ~/.bashrc
```

```
<fastapi>
```

```
$ pip install uvicorn
```

```
$ pip install fastapi
```

```
<pytorch>
```

```
$ conda install pytorch torchvision torchaudio cpuonly -c pytorch
```

```
$ pip install scikit-image
```

```
$ pip install dominate
```

```
$ pip install pandas
```

```
$ pip install visdom
```

```
$ pip install --upgrade pip setuptools wheel
```

```
$ pip install opencv-python
```

```
# python 서버 가동 준비 완료
```

5. FE / BE 빌드 및 배포

```
<FE>
```

```
npm install --legacy-peer-deps
```

```
npm run build
```

생성된 dist 폴더는 /home/ubuntu/ 아래에 위치

<BE-java>

gradle bootJar 로 빌드하여 jar 파일 생성

```
➤ nohup java -jar -Duser.timezone=Asia/Seoul 파일이름.jar &
```

백그라운드로 서버 가동

<BE-python>

Gitlab /backend-python/ 아래에 있는 /pose-transfer 폴더를 /home/ubuntu/ 에 배포

pose-transfer 디렉토리에 있는 main.py 백그라운드 실행

```
$ nohup python main.py &
```

6. OpenVidu 배포

- 기존 OpenVidu 가 있다면 삭제

```
$ sudo docker ps -a
```

#openvidu, kurento media server 등의 컨테이너가 존재한다면 삭제한다.

```
$ sudo docker rm #컨테이너 모두 삭제를 원할 경우
```

```
$ sudo docker rm $(docker ps -a) # 이미지도 삭제
```

```
$ sudo docker rmi # 이미지 전체 삭제를 원할 경우
```

```
$ sudo docker rmi $(docker images)
```

- OpenVidu 설치

```
$ sudo su # 관리자 권한
```

```
$ cd /opt # openvidu 가 설치되는 경로
```

```
# openvidu on promises 설치
```

```
$ curl https://s3-eu-west1.amazonaws.com/aws.openvidu.io/install_openvidu_latest.sh | bash
```

```
$ exit
```

- OpenVidu 설정 및 SSL 인증서 적용

```
$ cd /opt/openvidu
```

```
$ vi .env
```

```
DOMAIN_OR_PUBLIC_IP=
```

```
OPENVIDU_SECRET=MY_SECRET
```

```
CERTIFICATE_TYPE=letsencrypt
```

```
LETSencrypt_EMAIL=인증서 발급 시 입력한 이메일
```

```
HTTP_PORT=8442 HTTPS_PORT=8443 ...
```

```
# ESC 입력 후 :wq! 로 저장 후 나가기
```

- OpenVidu 포트 개방

```
$ sudo apt update
```

```
$ sudo apt install netfilter-persistent
```

```
$ sudo apt install iptables-persistent
```

```
$ sudo service iptables start
```

```
$ sudo iptables -A INPUT -p udp --match multiport --dports 40000:65535 -j ACCEPT
```

```
$ sudo iptables -A INPUT -p tcp --match multiport --dports 40000:65535 -j ACCEPT
```

```
$ sudo iptables -I INPUT 1 -p tcp --dport 22 -j ACCEPT
```

```
$ sudo iptables -I INPUT 1 -p tcp --dport 80 -j ACCEPT
```

```
$ sudo iptables -I INPUT 1 -p tcp --dport 443 -j ACCEPT
```

```
$ sudo iptables -I INPUT 1 -p tcp --dport 3478 -j ACCEPT
```

```
$ sudo iptables -I INPUT 1 -p udp --dport 3478 -j ACCEPT
```

```
$ sudo iptables -I INPUT 1 -p tcp --dport 8082 -j ACCEPT
```

```
$ sudo iptables -I INPUT 1 -p tcp --dport 8443 -j ACCEPT
```

```
$ sudo iptables -A INPUT -p tcp --dport 8080 -j ACCEPT
```

```
$ sudo service iptables save
```

```
$ sudo service iptables restart sudo
```

```
$ sudo netfilter-persistent save
```

```
$ sudo netfilter-persistent start
```

- OpenVidu 서비스(관련 컨테이너) 실행

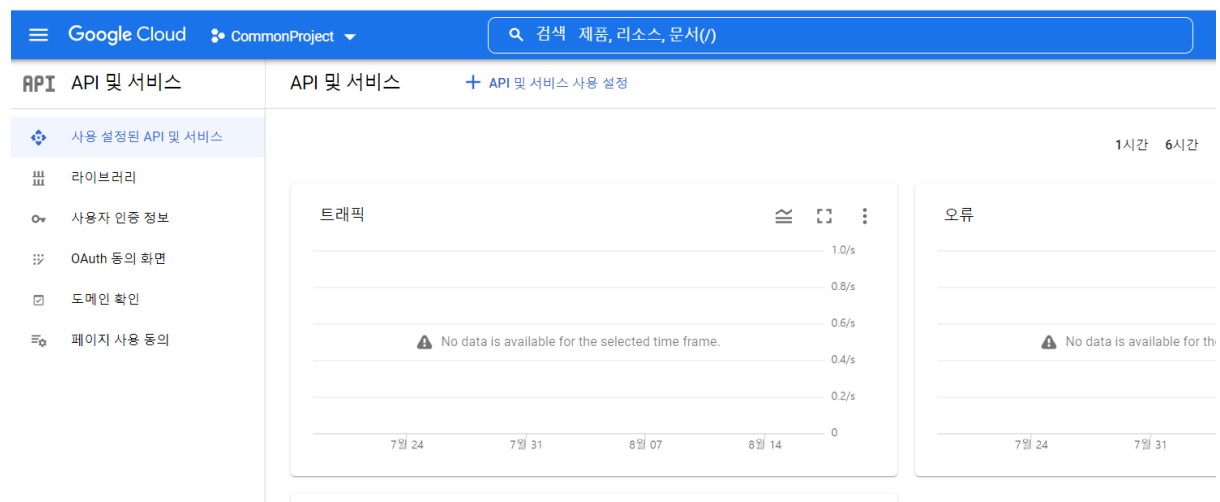
```
$ sudo ./openvidu start      # /opt/openvidu 위치에서
```

```
$ sudo ./openvidu stop      # 종료할 때는 같은 경로에서
```

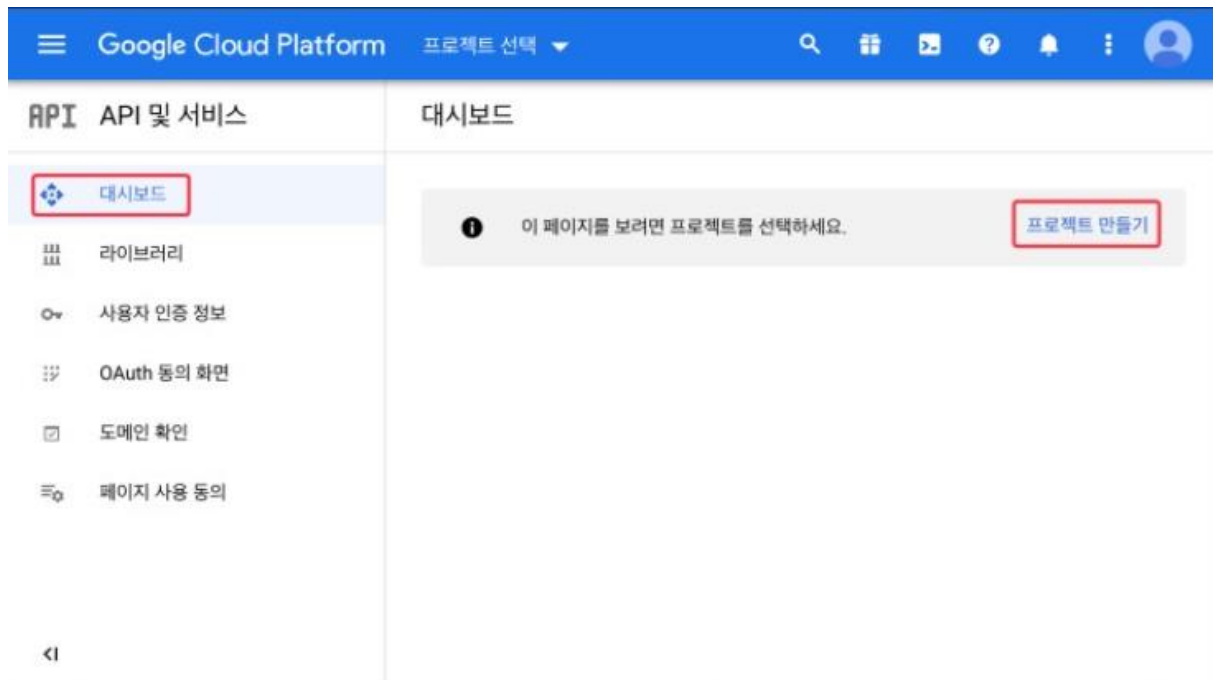
II. 외부 서비스

1. 구글 로그인 설정

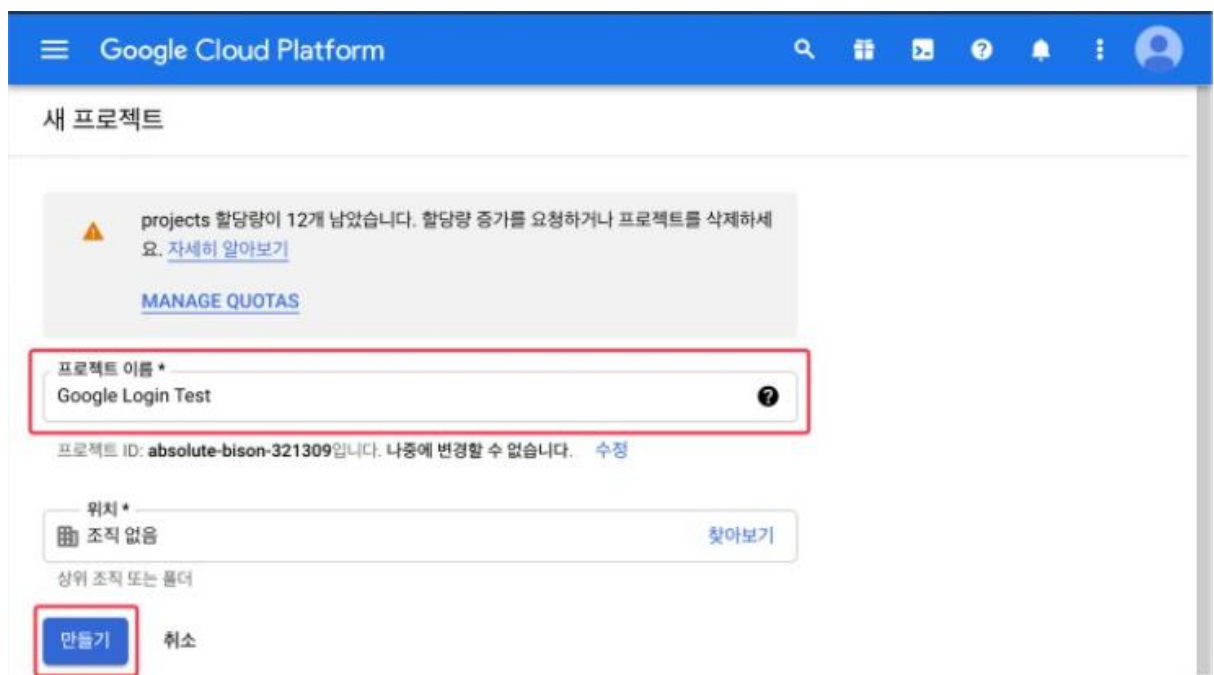
1) <https://console.cloud.google.com/apis/dashboard> 에 접속한다.



2) 프로젝트 생성하기



3) 프로젝트 이름 설정



4) OAuth 동의 화면

☰ Google Cloud
FAFFY
🔍 검색 제품, 리소스, 문서(/)

API API 및 서비스

- 사용 설정된 API 및 서비스
- 라이브러리
- 사용자 인증 정보
- OAuth 동의 화면
- 도메인 확인
- 페이지 사용 동의

앱 등록 수정

1 **OAuth 동의 화면** —
 2 범위 —
 3 테스트 사용자 —
 4 요약

앱 정보

동의 화면에 표시되어 최종 사용자가 개발자를 확인하고 문의할 수 있습니다.

앱 이름 *
 FAFFY - Find All Fashions For You
동의를 요청하는 앱의 이름

사용자 지원 이메일 *
 kimsm9822@gmail.com ▼
사용자가 동의 관련 질문을 위해 문의할 때 이용합니다.

앱 로고
[찾아보기](#)

사용자가 앱을 알아보는 데 도움이 되도록 동의 화면에 대한 이미지(1MB 이하 크기)를 업로드합니다. 허용되는 이미지는 JPG, PNG, BMP입니다. 최적의 결과를 위해서는 로고가 120x120픽셀 크기의 정사각형이어야 합니다.

Find All Fashions For You

5) 범위 설정

Google Cloud

FAFFY

검색 제품, 리소스, 문서(/)

API 및 서비스

사용 설정된 API 및 서비스

라이브러리

사용자 인증 정보

OAuth 동의 화면

도메인 확인

페이지 사용 동의

앱 등록 수정

✓ OAuth 동의 화면

2 범위

3 테스트 사용자

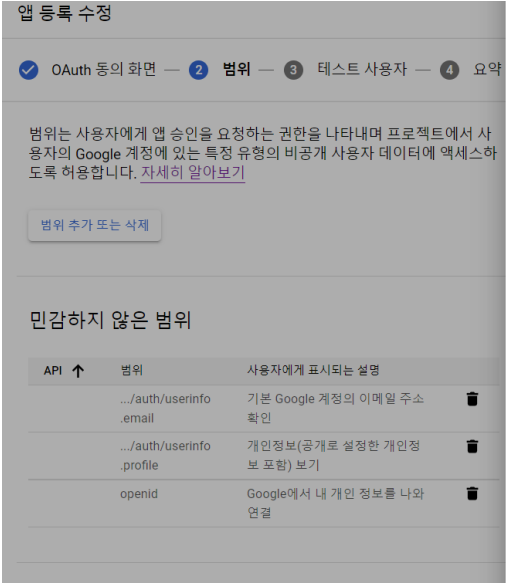
4 요약

범위는 사용자에게 앱 승인을 요청하는 권한을 나타내며 프로젝트에서 사용자의 Google 계정에 있는 특정 유형의 비공개 사용자 데이터에 액세스하도록 허용합니다. [자세히 알아보기](#)

범위 추가 또는 삭제

민감하지 않은 범위

API ↑	범위	사용자에게 표시되는 설명	
	.../auth/userinfo.email	기본 Google 계정의 이메일 주소 확인	
	.../auth/userinfo.profile	개인정보(공개로 설정한 개인정보 포함) 보기	
	openid	Google에서 내 개인 정보를 나와 연결	

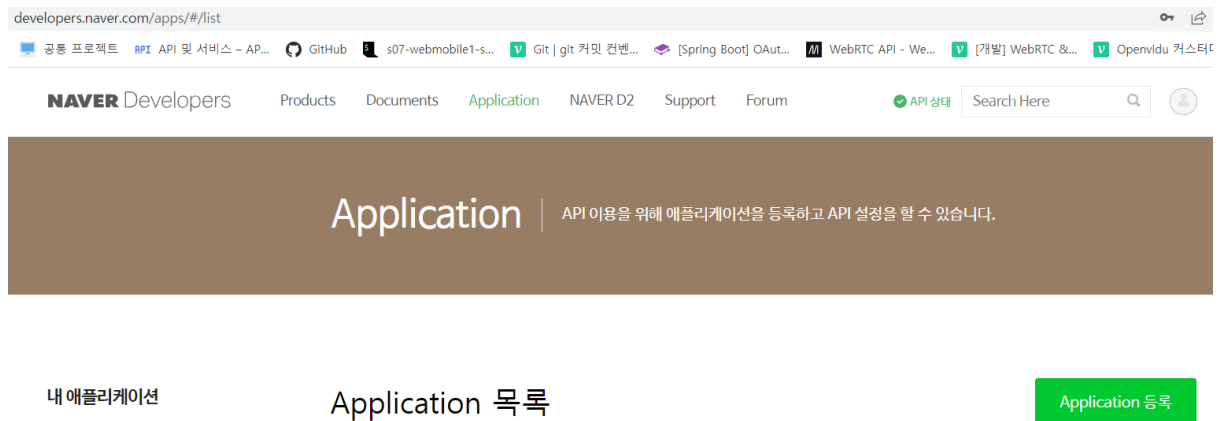


아래에는 사용 설정된 API의 범위만 나와 있습니다. 이 화면에 누락된 범위를 추가하려면 [Google API 라이브러리](#)에서 API를 찾아 사용 설정하거나 아래의 '붙여넣은 범위' 텍스트 상자를 사용하세요. 라이브러리에서 사용 설정한 새 API를 확인하려면 페이지를 새로고침하세요.

필터	속성 이름 또는 값 입력	
API	범위	사용자에게 표시되는 설명
<input checked="" type="checkbox"/>	.../auth/userinfo.email	기본 Google 계정의 이메일 주소 확인
<input checked="" type="checkbox"/>	.../auth/userinfo.profile	개인정보(공개로 설정한 개인정보 포함) 보기
<input checked="" type="checkbox"/>	openid	Google에서 내 개인 정보를 나와 연결
<input type="checkbox"/>	BigQuery API .../auth/bigquery	View and manage your data in Google BigQuery and see the email address for your Google Account
<input type="checkbox"/>	BigQuery API .../auth/cloud-platform	Google Cloud 데이터 확인, 수정, 구성, 삭제 및 Google 계정의 이메일 주소 확인
<input type="checkbox"/>	BigQuery API .../auth/bigquery.readonly	Google BigQuery에서 데이터를 봅니다.
<input type="checkbox"/>	BigQuery API .../auth/cloud-platform.read-only	Google Cloud 서비스 전체의 데이터 조회 및 Google 계정의 이메일 주소 확인
<input type="checkbox"/>	BigQuery API .../auth/devstorage.full_control	Manage your data and permissions in Cloud Storage and see the email address for your Google Account
<input type="checkbox"/>	BigQuery API .../auth/devstorage.read_only	Google 클라우드 저장소에서 데이터 조회
<input type="checkbox"/>	BigQuery API .../auth/devstorage.read_write	Cloud Storage의 데이터 관리 및 Google 계정의 이메일 주소 확인

6) 사용자 인증 정보 및 리디렉션 URI

1) naver developers의 Application 탭에서 새로운 앱 등록



2) 어플리케이션의 이름과 네이버 로그인 API를 선택 후, 제공 정보 선택

애플리케이션 이름

FAFFY

- 네이버 로그인할 때 사용자에게 표시되는 이름이므로 서비스 브랜드를 대표할 수 있는 이름으로 가급적 10자 이내로 간결하게 설정해주세요.
- 40자 이내의 영문, 한글, 숫자, 공백문자, 쉼표(,), "/" , "-" , "_" , 만 입력 가능합니다.

사용 API

선택하세요.

네이버 로그인

제공 정보 선택(이용자 식별자는 기본 정보로 제공)

필수 항목은 개인정보보호법 제3조 제1항, 제16조 제1항 등에 따라 서비스 제공을 위해 필요한 최소한의 개인정보를 선택해야 합니다.

권한	필수	추가
회원이름	<input checked="" type="checkbox"/>	<input type="checkbox"/>
이메일 주소	<input checked="" type="checkbox"/>	<input type="checkbox"/>
별명	<input checked="" type="checkbox"/>	<input type="checkbox"/>
프로필 사진	<input type="checkbox"/>	<input type="checkbox"/>
성별	<input type="checkbox"/>	<input checked="" type="checkbox"/>
생일	<input type="checkbox"/>	<input checked="" type="checkbox"/>
연령대	<input type="checkbox"/>	<input type="checkbox"/>
출생연도	<input type="checkbox"/>	<input checked="" type="checkbox"/>

3) 로그인 오픈 API 환경 설정 – PC 웹으로 추가하고, 서비스 URL, callback URL 입력

로그인 오픈 API
서비스 환경 ②

환경 추가 ▼

PC 웹

서비스 URL

https://i7a802.p.ssafy.io

서비스 URL 예시: (O) http://naver.com (X) http://www.naver.com
서비스 URL 값이 잘못 입력되어 있으면 정확한 값으로 수정하실 때 까지 네이버 로그인 사용이 일시적으로 제한됩니다.
불법/음란성 사이트 등 이용약관에 위배되는 사이트의 경우, 이용이 제한될 수 있습니다.
서비스하려는 사이트 URL과 동일한 사이트 URL로 해주셔야 **네이버 로그인 배지**가 노출됩니다.

네이버 로그인
Callback URL (최대 5개)

https://i7a802.p.ssafy.io/auth/naver/redirect

텍스트 폼 우측 끝의 '+' 버튼을 누르면 행이 추가되며, '-' 버튼을 누르면 행이 삭제됩니다.
Callback URL은 네이버 로그인 후 이동할 페이지 URL입니다. Callback URL 값이 잘못 입력되어 있으면 정확한 값으로 수정하실 때 까지 네이버 로그인 사용이 일시적으로 제한됩니다.
입력한 주소와 다른 Callback URL로 리다이렉트 될 경우, 이용이 제한될 수 있습니다.

4) 클라이언트 id/secret 확인

NAVER Developers

Products

Documents

Application

NAVER D2

Support

Forum

API 상태

Search Here

Application

API 이용을 위해 애플리케이션을 등록하고 API 설정을 할 수 있습니다.

내 애플리케이션

FAFFY

애플리케이션 등록

API 재휴 신청

계정 설정

FAFFY

개요

API 설정

네이버 로그인
검수상태

멤버관리

로그인 통계

API 통계

Playground (Beta)

애플리케이션 정보

Client ID

YTUxbIXLw0BuEFXiyUFV

Client Secret

.....

보기