

# 포팅 매뉴얼



## A603: YouLangMe

삼성SW 청년아카데미 서울캠퍼스 7기

공통프로젝트 7주, 2022/7/4~2022/08/19

## 포팅 매뉴얼

담당 컨설턴트- 박찬국

신규진(팀장), 고은민, 노용래, 김무중, 이민호, 최규섭

### <<목차>>

1. 기술 스택 -----
2. 빌드 상세내용 -----
3. 배포 특이사항 -----
4. DB 계정 -----
5. 프로퍼티 정의 -----
6. 외부 서비스 -----

코로나 바이러스로 인한 팬데믹 시대의 여파로 외국인과의 대면하여 외국어를 배울 수 있는 어학연수와 교환 학생의 기회가 줄었습니다. 또한 외국인과의 대화하는 것에 대한 울렁증이 심해 언어 교류를 두려워하는 사람들이 많습니다. 저희는 이러한 지점을 고려하여 웹 RTC 기술을 통해서 언제 어디서나 외국인과 자유롭게 편하게 언어 교류를 할 수 있는 YOULANGME를 개발하게 되었습니다.

## 1. 프로젝트 기술 스택

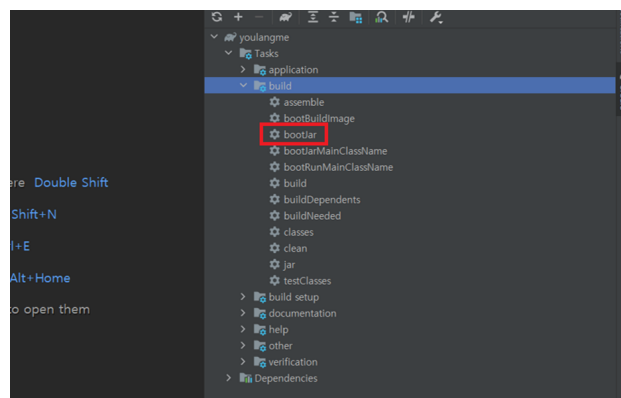
- 이슈관리 : Jira
- 형상관리: GitLab
- 커뮤니케이션: Mattermost, Notion, Discord
- 개발환경
  - OS: Windows 10
  - IDE
    - IntelliJ IDEA Community Edition 2022.1.3
    - Visual Studio Code 1.69.1

- Database: MySql WorkBench 8.0 CE
- Server : AWS EC2
- 상세 사용
  - Backend
    - JAVA 8
    - Spring boot Gradle
    - lombok, openvidu, Swagger
  - Frontend
    - React 17.2.0
    - React-router-dom 5.3.0
    - reduxjs/toolkit, redux-persist
  - AWS EC2
    - Nginx
    - Docker

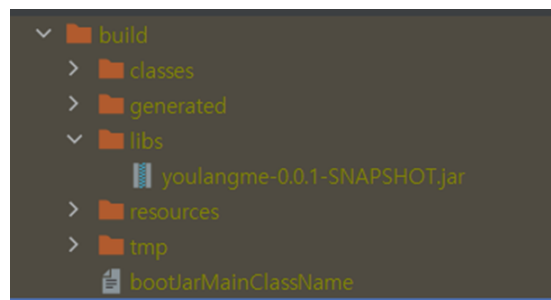
## 2. 빌드 상세 내용

### Spring build

- 우측의 Gradle -> youlangme -> Tasks -> build -> bootJar를 더블 클릭



- build 폴더가 생성되고, 그 아래에 libs 폴더 아래에 youlangme-0.0.1-SNAPSHOT.jar 가 빌드되었는지 확인



- 실행 방법 (백그라운드)

```
nohup java -jar youlangme-0.0.1-SNAPSHOT.jar &
```

### 3. 배포 특이사항

#### docker engine 설치

<https://docs.docker.com/engine/install/ubuntu/>

- Repository 설정

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

- GPG 키

```
sudo mkdir -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 설치

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

- Redis 설치 (docker 이미지 pull)

```
sudo docker pull redis
```

- Redis 실행 및 Redis-cli 실행

```
sudo docker run --name ylm -d -p 6379:6379 redis
sudo docker exec -it ylm redis-cli
```

#### 3 - 1 프론트엔드 배포

- 가) 클론받은 프로젝트 폴더에서 프론트엔드 폴더로 이동합니다.
- 나) 종속성을 모두 설치 후, 빌드를 진행합니다.
- 다) 도커 이미지 생성에 필요한 Dockerfile과 nginx.conf 파일은 다음과 같습니다.

```
# Dockerfile
# nginx image 사용
FROM nginx

# WORKDIR /app을 만들고 지정
RUN mkdir /app

WORKDIR /app

RUN mkdir ./build

ADD ./build ./build

# conf.d에 있는 default.conf를 삭제
RUN rm /etc/nginx/conf.d/default.conf

# 현재 위치에 있는 nginx.conf를 /etc/nginx/conf.d로 복사
COPY ./nginx.conf /etc/nginx/conf.d

# 포트 설정
EXPOSE 80

# 실행시 nginx daemon off
CMD ["nginx", "-g", "daemon off;"]
```

```
# nginx.conf
server {
    listen 80;
    location / {
        root    /app/build;
        index  index.html;
        try_files $uri $uri/ /index.html;
    }
}
```

라) 도커 이미지를 생성한 후, 도커 컨테이너를 통해 프론트엔드를 배포합니다.

```
# 저장소 클론
git clone https://lab.ssafy.com/s07-webmobile1-sub2/S07P12A603.git

# 프론트엔드 폴더로 이동
cd frontend

# 종속성 설치
npm install --legacy-peer-deps

# 빌드 파일 생성
CI=false npm run build

# 도커 이미지 빌드
docker build -t nginx-react:0.1 .

# 도커 컨테이너를 이용한 프론트엔드 배포
docker run --name nginx_react -d -p 3000:80 nginx-react:0.1
```

### 3 - 2 Spring 배포

#### Dockerfile 내용 확인

```
FROM openjdk:8

EXPOSE 8080

ADD build/libs/youlangme-0.0.1-SNAPSHOT.jar app.jar

ENTRYPOINT ["java", "-jar", "/app.jar"]
```

## Ubuntu 환경에서 docker build

```
// build 전에 기존에 돌아가는 이미지가 있다면 종료하기

sudo docker stop ylm_back // 컨테이너 종료

sudo docker rm ylm_back // 컨테이너 삭제

sudo docker rmi ylm_back // 이미지 삭제

// backend 폴더 내부에서 아래 명령 실행

sudo docker build -t ylm_back . // 현 위치에서 ylm_back 이름으로 도커 이미지 빌드

sudo docker run --name ylm_back -d -p 8080:8080 ylm_back // 8080 포트, 백그라운드로 실행
```

### 3 - 3 Django 배포

가) 작성한 Django 프로젝트의 폴더 내부에 Dockerfile을 생성합니다.

```
FROM python:3.8

WORKDIR /usr/src/youlangme

COPY . .

RUN pip install --upgrade pip
RUN pip install -r requirements.txt

EXPOSE 8000

CMD ["python3", "manage.py", "runserver", "0:8000", "--noreload"]
```

나) 생성한 경로에서 Dockerfile을 다음 명령어로 빌드합니다.

```
sudo docker build . -t django-docker --network=host
```

다) 생성된 Docker image 확인 후 Docker run 명령어로 Django 서버를 실행합니다.

```
sudo docker run --net=host --name ylm_matching -d -p 8000:8000 ylm_matching
```

### 3 - 4 SSL 인증서 및 인증서 적용

- 오픈비두를 배포하기 위해 root 권한을 얻은 후, 문서에서 권장되는 경로로 이동합니다

```
sudo su

cd /opt
```

- 오픈비두 설치 후 설치된 경로로 이동합니다.

```
curl <https://s3-eu-west-1.amazonaws.com/aws.openvidu.io/install_openvidu_latest.sh> | bash
```

```
cd openvidu
```

- 오픈비두 환경 설정 파일인 .env를 다음과 같이 설정합니다.

```
nano .env

# OpenVidu configuration
# -----
# 도메인 또는 퍼블릭IP 주소
DOMAIN_OR_PUBLIC_IP=i7a603.p.ssafy.io

# 오픈비두 서버와 통신을 위한 시크릿
OPENVIDU_SECRET=YOULANGME

# Certificate type
CERTIFICATE_TYPE=letsencrypt

# 인증서 타입이 letsencrypt일 경우 이메일 설정
LETSencrypt_EMAIL=mjgh0135@gmail.com

# HTTP port
HTTP_PORT=8442

# HTTPS port(해당 포트를 통해 오픈비두 서버와 연결)
HTTPS_PORT=8443 # 이미 할당된 포트를 사용해서는 안됨, 자체 nginx로 인해 포트 충돌 생김
```

- 설정 후 오픈비두 서버 실행합니다.

```
./openvidu start
```

- SSL 인증서를 적용하기 위해 다음과 같은 명령어를 입력합니다.

```
sudo apt-get install certbot python3-certbot-nginx

sudo service stop nginx

sudo certbot certonly --standalone -d i7a603.p.ssafy.io
```

- 이 경우 /etc/letsencrypt/live/i7a603.p.ssafy.io에 ssl 인증서가 설치됩니다.
- /etc/nginx/sites-available로 이동, 아래와 같은 파일을 생성합니다.

```
server {
    location /{
        proxy_pass <http://localhost:3000>;
    }

    location /api{
        proxy_pass <http://localhost:8080>;
    }
}

listen 443 ssl;
server_name i7a603.p.ssafy.io;
ssl_certificate /etc/letsencrypt/live/i7a603.p.ssafy.io/fullchain.pem; # managed by Certbot
ssl_certificate_key /etc/letsencrypt/live/i7a603.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 ($host = i7a603.p.ssafy.io) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    listen 80;
    server_name i7a603.p.ssafy.io;
    return 404; # managed by Certbot
}
```

- 80 port로 접근할 경우 ssl인증서가 적용된 443 port 로 리다이렉트됩니다.
- 또한 443 port의 /api로 접근 시에는 localhost:8080로 분기 처리됩니다.
- 이후 다음 명령어를 입력하면 ssl 인증서가 적용됩니다

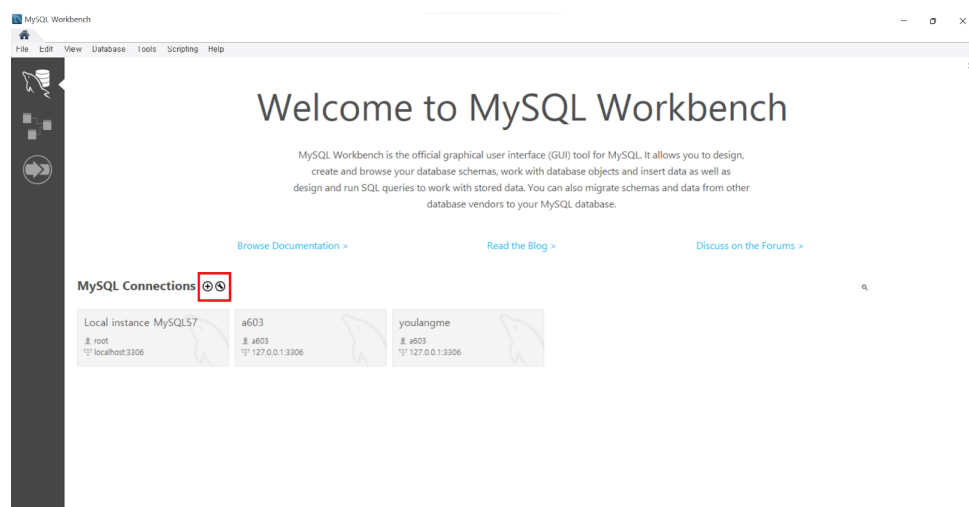
```
sudo ln -s /etc/nginx/sites-available/[파일명] /etc/nginx/sites-enabled/[파일명]
# 필자의 경우 sslyoulangme.conf

sudo nginx -t

sudo service restart nginx
```

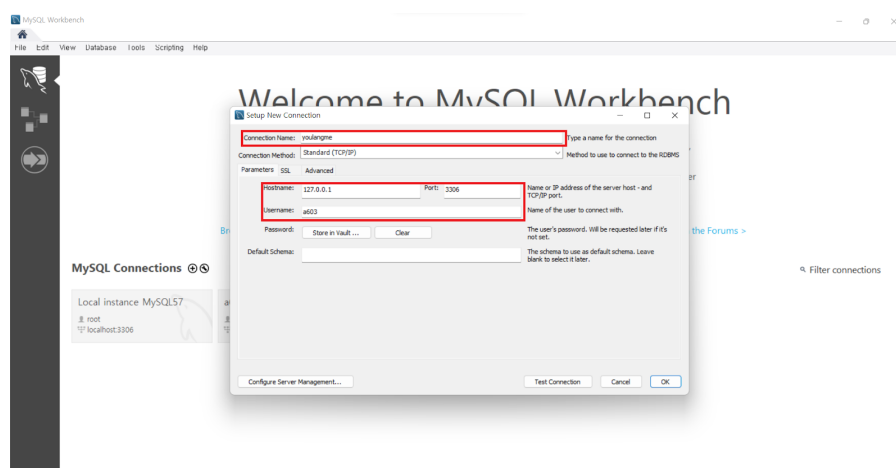
## 4. DB 계정

가. MySql workbench 추가하기



MySql Workbench를 열어서 새로운 내용을 추가하기 위해 “+” 버튼을 눌러줍니다.

나. EC2 계정 정보 설정



username은 a603, password는 youlangme를 사용하였습니다.

기존 root 계정이 아닌 별도의 계정을 생성하여 프로젝트를 진행하였습니다.

## 5. 프로퍼티 정의

가) nginx 세팅

- Docker 사용 시에는 /etc/nginx/sites-available로 이동한 후 아래와 같은 파일을 생성합니다.

```
server {
    location /{
        proxy_pass <http://localhost:3000>;
    }

    location /api{
        proxy_pass <http://localhost:8080>;
    }

listen 443 ssl;
    server_name i7a603.p.ssafy.io;
    ssl_certificate /etc/letsencrypt/live/i7a603.p.ssafy.io/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/i7a603.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 ($host = i7a603.p.ssafy.io) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    listen 80;
    server_name i7a603.p.ssafy.io;
    return 404; # managed by Certbot
}
```

- 다음 명령어를 입력합니다.

```
sudo ln -s /etc/nginx/sites-available/[파일명] /etc/nginx/sites-enabled/[파일명]
# 필자의 경우 sslyoulangme.conf

sudo nginx -t

sudo service restart nginx
```

- Git으로부터 클론 받은 후 프론트엔드로 이동하면 다음과 같은 nginx.conf가 있습니다

```
server {
    listen 80;
    location / {
        root    /app/build;
        index   index.html;
        try_files $uri $uri/ /index.html;
    }
}
```

- 빌드 시 conf.d/default.conf가 삭제되고, 다음과 같은 conf로 설정됩니다.

## 6. 외부 서비스

### 가. 네이버

번역 기능을 추가하기 위해 네이버 OpenApi인 Papago를 사용하였습니다.

이를 통해 채팅 중 상대방의 언어를 빠르게 번역할 수 있게 만들었습니다.



## 애플리케이션 등록 (API 이용신청)

애플리케이션의 기본 정보를 등록하면, 좌측 내 애플리케이션 메뉴의 서브 메뉴에 등록하신 애플리케이션 이름으로 서브 메뉴가 만들어집니다.

애플리케이션 이름 ⇄	<div> <input type="text" value="YouLangMe"/> <span>✓</span> </div> <p>애플리케이션 이름은 사용자에 표시되는 이름으로 서비스 브랜드를 대표할 수 있는 이름으로 가급적 10자 이내로 간결하게 설정해주세요.</p> <ul style="list-style-type: none"> <li>40자 이내의 영문, 한글, 숫자, 공백문자, 쉼표(,), "/" , "-" , "_" 만 입력 가능합니다.</li> </ul>
사용 API ⇄	<div> <div> <div>선택하세요. ▼</div> <span>✓</span> </div> <div> <input type="text" value="Papago 번역"/> <span>✕</span> </div> </div>
비로그인 오픈 API 서비스 환경	<div> <div> <div>환경 추가 ▼</div> <div> <div>WEB 설정 ✕ ^</div> <div> <div>웹 서비스 URL (최대 10개)</div> <div> <input type="text" value="https://localhost:8080/api"/> <div> <span>+</span> <span>✓</span> </div> </div> </div> </div> </div> <p>텍스트 폼 우측 끝의 '+' 버튼을 누르면 행이 추가되며, '-' 버튼을 누르면 행이 삭제됩니다.</p> <ul style="list-style-type: none"> <li>http와 https는 구분하지 않습니다.</li> <li>www는 빼고 입력해 주세요. 예) http://naver.com</li> <li>서브 도메인이 있으면 대표 도메인명만 입력해 주세요. (예: http://naver.com)</li> <li>하이브리드 앱은 location.href 객체 출력 값을 입력하면 됩니다. (예: file://로컬 URI)</li> </ul> </div>

등록하기

취소

### 1. 어플리케이션 추가

어플리케이션 이름을 등록해 줍니다.

#### 1. 사용 API 선택

많은 API들 중에 지금 필요한 Papago 번역을 사용 API로 추가합니다

#### 1. 사용 서비스 환경 등록

사용할 환경을 등록해줍니다. WEB에서 사용할 것이므로 WEB설정으로 추가해주고 사용할 URL을 등록해줍니다.

## YouLangMe

개요	API 설정	멤버관리	로그인 통계	API 통계	Playground(Beta)
----	--------	------	--------	--------	------------------

### 애플리케이션 정보

Client ID	daX_iDO_hwyxJ7HnaOg_
Client Secret	..... <a href="#">보기</a>

### 파파고 NMT API 가이드

가이드에 따라, cURL을 이용해 동작을 확인해 보세요.

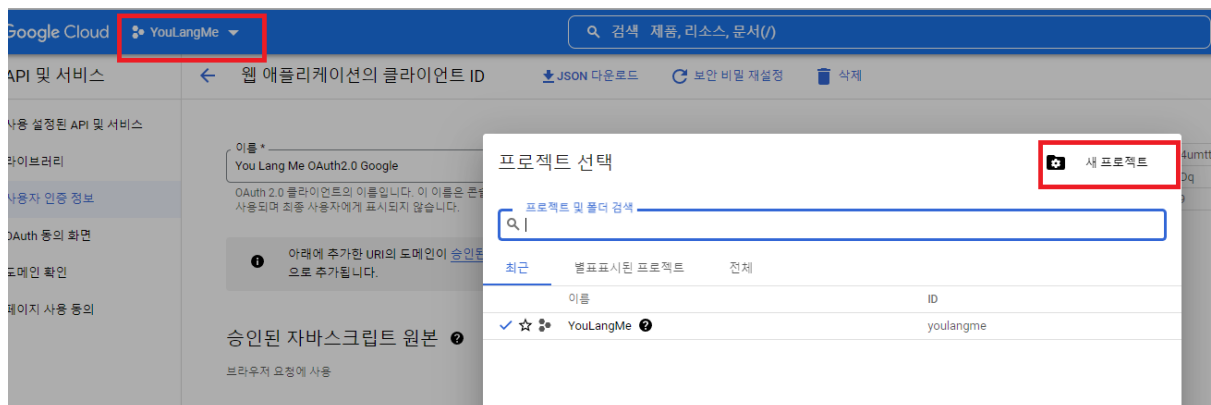
```
curl "https://openapi.naver.com/v1/papago/n2mt" \
-H "Content-Type: application/x-www-form-urlencoded; charset=UTF-8" \
-H "X-Naver-Client-Id: daX_iDO_hwyxJ7HnaOg_" \
-H "X-Naver-Client-Secret: m2chYsXQYb" \
-d "source=ko&target=en&text=만나서 반갑습니다." -v
```

#### 4) 등록된 어플리케이션 확인

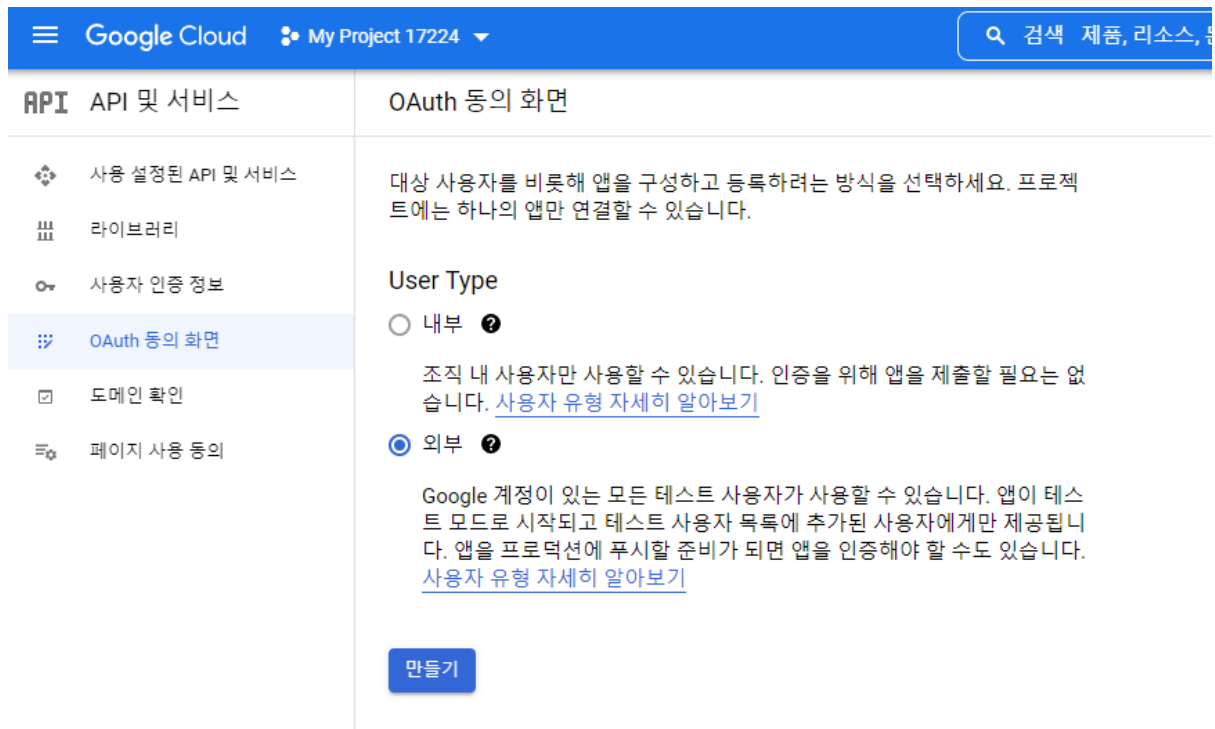
등록된 어플리케이션을 통해 발급 받은 Client ID와 Client 비밀번호를 통해 코드에 작성하여 어플리케이션과 연결해주면 Papago 사용 설정이 마무리 됩니다.

### 나. 구글 OAuth 2.0

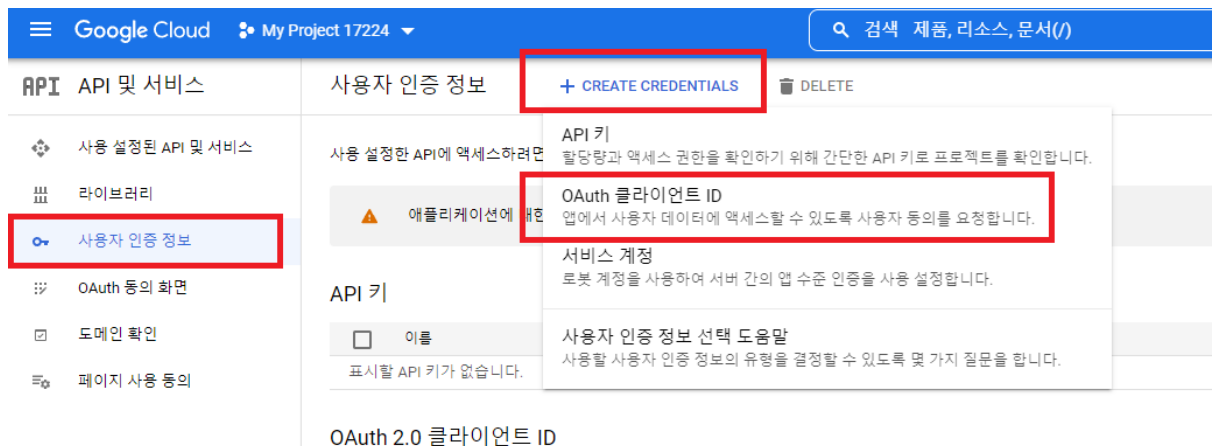
소셜 로그인 기능을 구현하기 위해 구글 OAuth를 사용합니다.



1. 구글 클라우드 소셜 로그인 기능을 구현하기 위해 구글 OAuth를 사용합니다. 플랫폼에 접속하여 좌측 상단에 드롭다운 클릭 후 나오는 팝업 창의 새 프로젝트 버튼을 클릭합니다.



2. 생성한 프로젝트 접속 후 OAuth 동의 화면에서 사용자 타입을 외부로 설정합니다.



3. 앱 정보, 앱 도메인, 사용자 정보를 입력한 후 사용자 인증 정보 탭에서 OAuth 클라이언트 ID를 선택합니다

클라이언트 ID는 Google OAuth 서버에서 단일 앱을 식별하는 데 사용됩니다. 앱이 여러 플랫폼에서 실행되는 경우 각각 자체 클라이언트 ID가 있어야 합니다. 자세한 내용은 [OAuth 2.0 설정을 참조하세요](#). OAuth 클라이언트 유형을 [자세히 알아보세요](#).

애플리케이션 유형 \*

웹 애플리케이션

이름 \*

Youlangme

OAuth 2.0 클라이언트의 이름입니다. 이 이름은 콘솔에서 클라이언트를 식별하는 용도로만 사용되며 최종 사용자에게 표시되지 않습니다.

아래에 추가한 URI의 도메인이 [승인된 도메인](#)으로 [OAuth 동의 화면](#)에 자동으로 추가됩니다.

승인된 자바스크립트 원본

브라우저 요청에 사용

+ URI 추가

승인된 리디렉션 URI

웹 서버의 요청에 사용

URI 1 \*

https://17a603.p.ssafy.io/api/oauth2/callback/google

4. 정보 입력 후 사용자 인증 정보 탭에서 OAuth 클라이언트 ID를 선택해서 프로젝트명, 로그인 후 리다이렉션할 URI를 입력합니다.

### 다. 구글 이메일

Google 계정

Google 계정 검색

홈

개인 정보


데이터 및 개인 정보 보호

보안

사용자 및 공유

결제 및 구독


정보



A 603님, 환영합니다

정보, 개인 정보 보호 및 보안 설정을 관리하여 나에게 맞는 방식으로 Google을 사용할 수 있습니다.
[자세히 알아보기](#)


개인 정보 보호 및 맞춤설정



Google 계정에 저장된 데이터를 확인하고 Google 사용 환경을 맞춤설정하기 위해 어떤 활동을 저장할지 선택합니다.

데이터 및 개인 정보 보호 관리

보안 권장사항 확인




보안 진단에서 권장 조치를 확인할 수 있습니다.

계정 보호


개인정보 보호를 위한 추천 설정 이용 가능

개인정보 보호 진단을 실행하고 나에게 맞는 설정을 선택하세요.



1. 구글 계정에 접속해서 보안 카테고리를 눌러줍니다.

Google에 로그인



비밀번호

최종 변경일: 7월 28일

>

2단계 인증

☒ 사용

>

앱 비밀번호

비밀번호 1개

>

1. 2단계 인증을 사용으로 바꿔주고 앱 비밀번호를 SMTP로 설정해줍니다.

이후 발급받은 앱 비밀번호를 이용하여 application.properties에 설정해줍니다

## 라. Openvidu 적용

- 오픈비두를 배포하기 위해 root 권한을 얻은 후, 문서에서 권장되는 경로로 이동합니다.

```
sudo su
cd /opt
```

- 오픈비두 설치 후 설치된 경로로 이동합니다.

```
curl <https://s3-eu-west-1.amazonaws.com/aws.openvidu.io/install_openvidu_latest.sh> | bash
cd openvidu
```

- 오픈비두 환경 설정을 변경합니다.

```
nano .env

# OpenVidu configuration
# -----
# 도메인 또는 퍼블릭IP 주소
DOMAIN_OR_PUBLIC_IP=i7a603.p.ssafy.io

# 오픈비두 서버와 통신을 위한 시크릿
OPENVIDU_SECRET=YOULANGME

# Certificate type
CERTIFICATE_TYPE=letsencrypt

# 인증서 타입이 letsencrypt일 경우 이메일 설정
LESENCRYPT_EMAIL=mjgh0135@gmail.com

# HTTP port
HTTP_PORT=8442

# HTTPS port(해당 포트를 통해 오픈비두 서버와 연결)
HTTPS_PORT=8443 # 이미 할당된 포트를 사용해서는 안됨, 자체 nginx로 인해 포트 충돌 생김
```

- 설정 후 오픈비두 서버를 실행합니다.

```
./openvidu start
```

