



AirLingo 포팅매뉴얼

메뉴

1. 기술 스택

2. 아키텍처 구조

3. 포트 설정

4. 배포 방법

- EC2접속 툴 설치 및 설정
- 서버 시간 설정
- 도커 설치
- 자바,깃 설치
- Mysql 설치
 - Mysql Workbench연결
- Redis 설치
- Nginx 설치
 - Nginx 설정
- 젠킨스 설치
 - GITLAB 연동
 - WEBHOOK 설정
 - 빌드 쉘 스크립트 작성
- RabbitMQ 설치
- NODE 설치 && YJS 시그널링 서버 설치
- 지급받은 EC2에 Openvidu CE 설치
- OPENVIDU PRO버전 배포

5. 외부 서비스 소개

- AWS Lambda
- AWS S3
- NAVER CLOVA Speech API
- PAPAGO API

기술 스택

형상 관리

- Gitlab

이슈 관리

- Jira

Communication

- Mattermost
- WebEx
- Notion

UI / UX

- Figma

IDE

- IntelliJ Ultimate
- VSCode

Server

- AWS-EC2 - t2.xlarge
 - Ubuntu - 20.04 LTS
 - Docker - 24.0.4
 - Nginx - 1.18.0
 - Jenkins
 - Java - 17.0.7
 - Openvidu - 2.28.0
- AWS-S3
- AWS-Lambda

ETC

- Mobaxterm
- Postman
- ChatGPT

DataBase

- MySql - 8.0.33
- Redis - 7.0.12

API

- PAPAGO API
- CLOVA Speech API

/\'를 입력해 명령어 사용

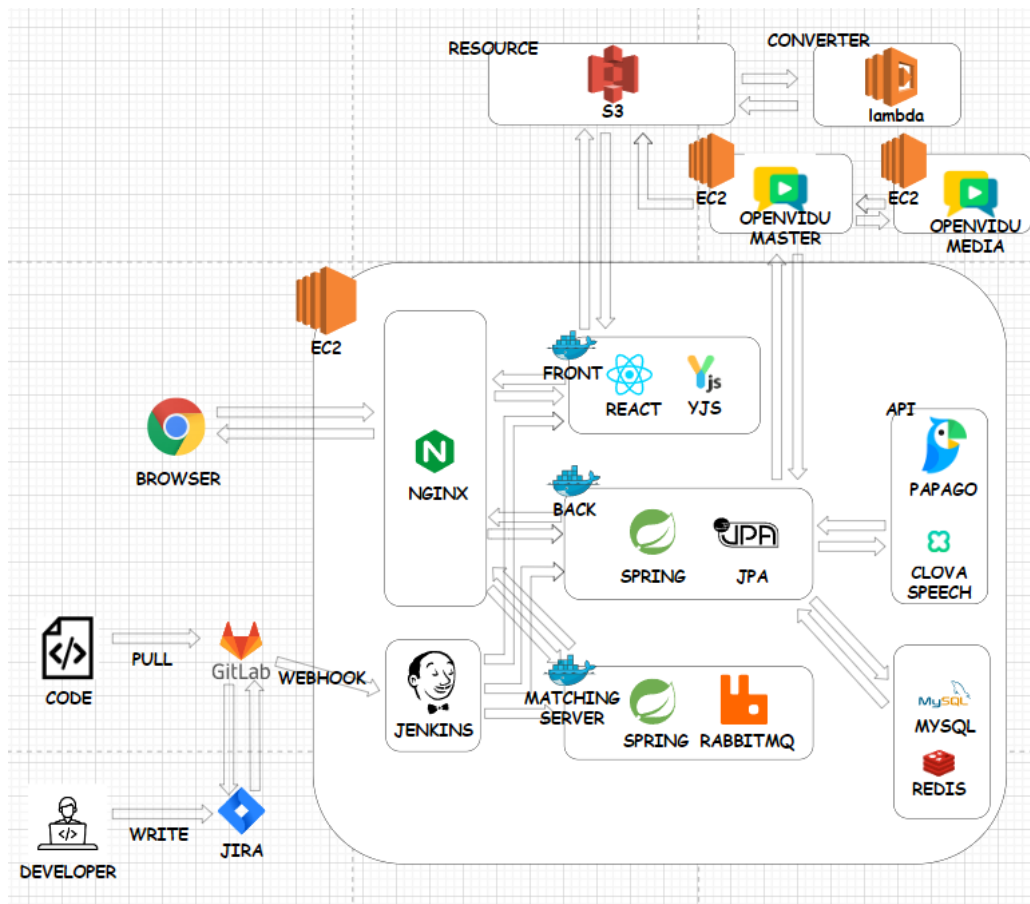
Back-End

- Java-17.0.7
- Spring boot - 3.1.1
- Gradle - 8.1.1
- Spring Data Jpa - 3.1.1
- Spring Data Redis - 3.1.1
- Spring Boot Websocket - 3.1.1
- Spring AMQP(RabbitMq) - 3.1.1
- Lombok - 1.18.28
- Swagger - 2.0.2

Front-End

- React - 18.2.0
- Node.js - 18.17.0
- Redux - 8.1.1
- Yjs - 13.6.7
- Axios - 1.4.0
- React-calendar - 4.6.0
- React-chartjs-2 - 4.3.1
- React-quill - 2.0.0
- React-speech-kit - 3.0.1
- Sockjs-client - 1.6.1
- Stompjs - 2.3.3
- y-webrtc - 10.2.5
- y-websocket - 1.5.0

아키텍처 구조



포트 설정

App	EC2 Port	컨테이너 Port
Spring	8081	8081
Matching	8082	8082
React	5173	5173
MySQL	3306	X
nginx http	80	X
nginx https	443	X
Jenkins	8080	X
Redis	6379	X
RabbitMQ	5432	X
Openvidu	5443	5443
Coturn	3478	3478
Kurento	8888	8888
y-webrtc	3333	X
y-websocket	3334	X

1. MobaXterm 설치(원격 접속 툴)

설치후 우측 상단 SSH버튼 클릭후 설정 후 접속

Session settings ×

SSH Telnet Rsh Xdmcp RDP VNC FTP SFTP Serial File Shell Browser Mosh Aws S3 WSL

Basic SSH settings

Remote host * ☐ Specify username Port

Advanced SSH settings Terminal settings Network settings Bookmark settings

☒ X11-Forwarding ☒ Compression Remote environment:

Execute command: ☐ Do not exit after command ends

SSH-browser type: ☐ Follow SSH path (experimental)

☒ Use private key

Execute macro at session start:

2. 서버 시간 설정

```
#현재 시간 설정 확인
date

#목록에 서울 시간 있는지 확인
timedatectl list-timezones | grep Seoul

#서울 시간으로 변경
sudo timedatectl set-timezone Asia/Seoul
```

3. 도커 설치

```
#APT 업데이트 , 다양한 패키지들 시스템에 설치
sudo apt update
sudo apt install -y ca-certificates curl software-properties-common apt-transport-https gnupg lsb-release

#도커 Repository 접근을 위한 gpg 키 다운 및 설정
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 $(lsb_re
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu "$(. /

#도커(관련 패키지) 설치
sudo apt update
```

```
sudo apt install docker-ce docker-ce-cli containerd.io docker-compose
docker --version
```

4. 자바 , 깃 설치

```
sudo apt-get update
sudo apt-get install openjdk-17-jdk
sudo apt-get install -y git
java -version
```

5. Mysql 설치

```
#MySQL APT Repository 추가 & 패키지 다운로드
sudo wget https://dev.mysql.com/get/mysql-apt-config_0.8.13-1_all.deb
sudo dpkg -i mysql-apt-config_0.8.13-1_all.deb

#MySQL 설치
sudo apt-get update
sudo apt-get install mysql-server

#방화벽 허용(Workbench 쓰기 위해서)
sudo ufw allow mysql

#DB 설치 및 유저 생성 후 권한 설정
create user 'airlingo'@'%' identified by 'airlingo1579';
grant all privileges on *.* to 'airlingo'@'%';
flush privileges;
```

Workbench 에서 new connection

Connection Name:

Connection Method: Method to use to connect to the RDBMS

Parameters ☒ SSL ☐ Advanced

SSH Hostname: SSH server hostname, with optional port number.

SSH Username: Name of the SSH user to connect with.

SSH Password: SSH user password to connect to the SSH tunnel.

SSH Key File: Path to SSH private key file.

MySQL Hostname: MySQL server host relative to the SSH server.

MySQL Server Port: TCP/IP port of the MySQL server.

Username: Name of the user to connect with.

Password: The MySQL user's password. Will be requested later if not set.

Default Schema: The schema to use as default schema. Leave blank to select it later.

- Hostname: EC2(서버)의 public IP 주소/도메인
- Port: 3306
- Username: airlingo
- Password: airlingo1579

6. Redis 설치

```
#JWT Token을 저장하기 위한 저장소로 이용
#Redis 서버 설치 및 버전 확인
sudo apt-get install redis-server
redis-server --version

#Redis 설정 파일 수정
sudo vi /etc/redis.conf

#원격 액세스 가 가능하도록 서버를 열어줌
bind 0.0.0.0 ::1

#메모리 최대 사용 용량 및 메모리 초과시 오래된 데이터를 지워 메모리 확보하도록 정책 설정
maxmemory 2g
maxmemory-policy allkeys-lru

sudo systemctl restart redis-server

#Redis 설정 확인
sudo systemctl status redis-server
redis-cli ping

#port 4321로 바꿔줌

#명령어를 이용해 Redis 서버 접속 가능
redis-cli
```

7. Nginx 설치

```
sudo apt-get install nginx

#실행전 80번 포트를 사용하는 nginx를 위해 방화벽 허용
sudo ufw allow 80

#nginx 실행
sudo systemctl start nginx
sudo systemctl stop nginx
sudo systemctl status nginx

#인증서 발급
sudo snap install certbot --classic
sudo certbot --nginx
#이메일 쓰고 agree
#뉴스레터 no
#도메인 입력
#인증서 발급 성공 , 인증서 보관하기

#인증서 접근권한 문제 발생 -> 밑에 명령어로 해결
sudo chmod 755 /etc/letsencrypt/live/

#ssl 설정후 왜 안되나 했더니 443포트를 아직 안열어 줬었다...
sudo ufw allow 443

#심볼릭 링크를 확인하기 위한 명령어
readlink /etc/nginx/sites-enabled/default
```

```
# cd etc/nginx/conf.d
# sudo vim default.conf
# etc/nginx/conf.d/default.conf

server{
    # 80 port로 서버 오픈
    listen 80;
    #IPv6 주소에서 들어오는 요청을 처리
    listen [::]:80;
    #서버 이름
    server_name i9a308.p.ssafy.io;
    #HTTP 요청을 받으면 모두 HTTPS로 리디렉션(301 Redirect)
    return 301 https://$server_name$request_uri;
}
```

```

server{
    #포트 443과 IPv4 주소, 그리고 포트 443과 IPv6 주소에서 들어오는 요청을 처리
    listen 443 ssl;
    listen [::]:443 ssl;

    #서버 이름
    server_name i9a308.p.ssafy.io www.i9a308.p.ssafy.io;

    #HTTPS 요청을 받으면 /var/www/html 디렉토리에 위치한 정적 파일들을 서비스
    #root /var/www/html;
    #index index.html index.htm index.nginx-debian.html;

    #2가지 키가 발급
    ssl_certificate /opt/openvidu/certificates/live/i9a308.p.ssafy.io/fullchain.pem;
    ssl_certificate_key /opt/openvidu/certificates/live/i9a308.p.ssafy.io/privkey.pem;

    location / { # 프론트엔드
        proxy_pass http://localhost:5173;
        proxy_redirect off;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
    }

    location /api { # 백엔드
        proxy_pass http://localhost:8081;
        proxy_redirect off;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "Upgrade";
        proxy_set_header Host $host;
    }

    location /matching { #매칭서버
        proxy_pass http://localhost:8082;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "Upgrade";
        proxy_set_header Host $host;
    }

    location /ws { #매칭 웹소켓
        proxy_pass http://localhost:8081/ws/;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
    }

    location /chat { #채팅 웹소켓
        proxy_pass http://localhost:8081/chat/;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
    }

    location /openvidu {
        proxy_pass https://i9a308.p.ssafy.io:8443;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
    }

    # y-webrtc 웹소켓 연결 설정 for 스크립트 동시편집
    location /ywebrtc {
        proxy_pass http://localhost:3333;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $host;
    }

    # y-websocket 웹소켓 연결 설정 for 화이트보드
    location /websocket {
        proxy_pass http://localhost:3334;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $host;
    }

    proxy_set_header Host $http_host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    add_header 'Access-Control-Allow-Origin' '*';
}

```



```
proxy_http_version 1.1;
proxy_set_header Upgrade $http_upgrade;
proxy_set_header Connection "upgrade";
```

→ 웹 서버에서 WebSocket 등의 프로토콜을 사용할 때 필요한 설정

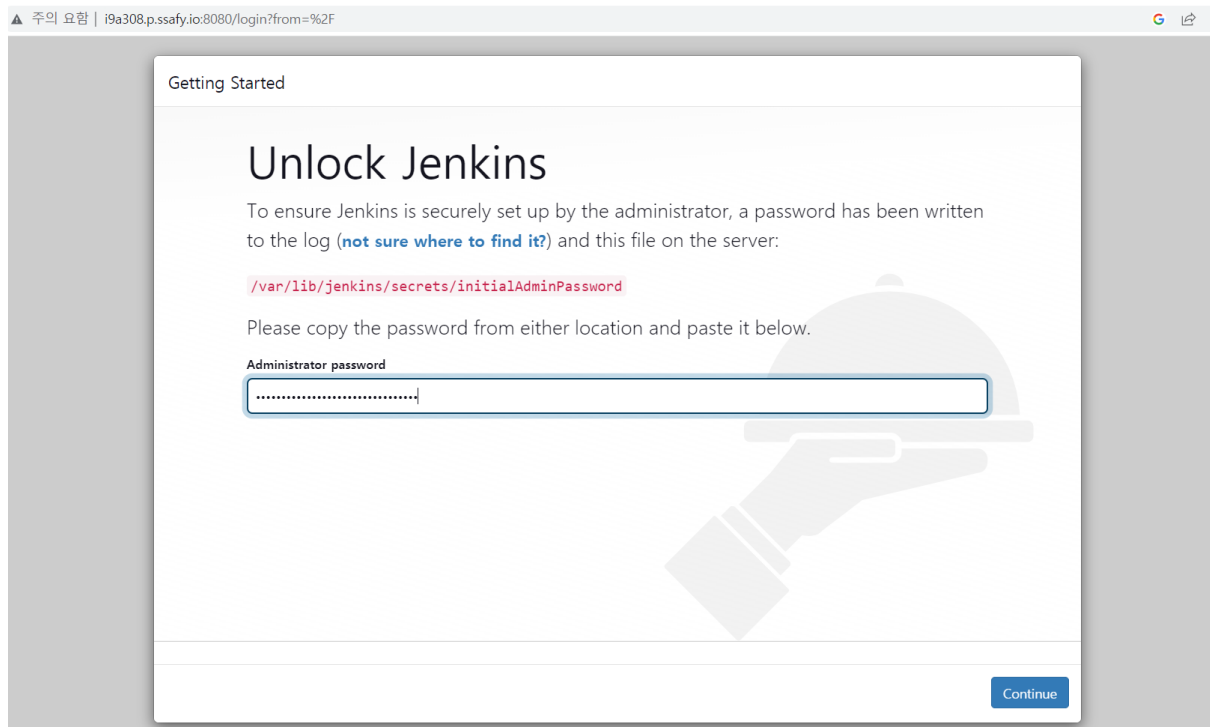
8. 젠킨스 설치

```
#밑에 명령어로 실행할것
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
  /usr/share/keyrings/jenkins-keyring.asc > /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null

sudo apt-get update
sudo apt-get install jenkins

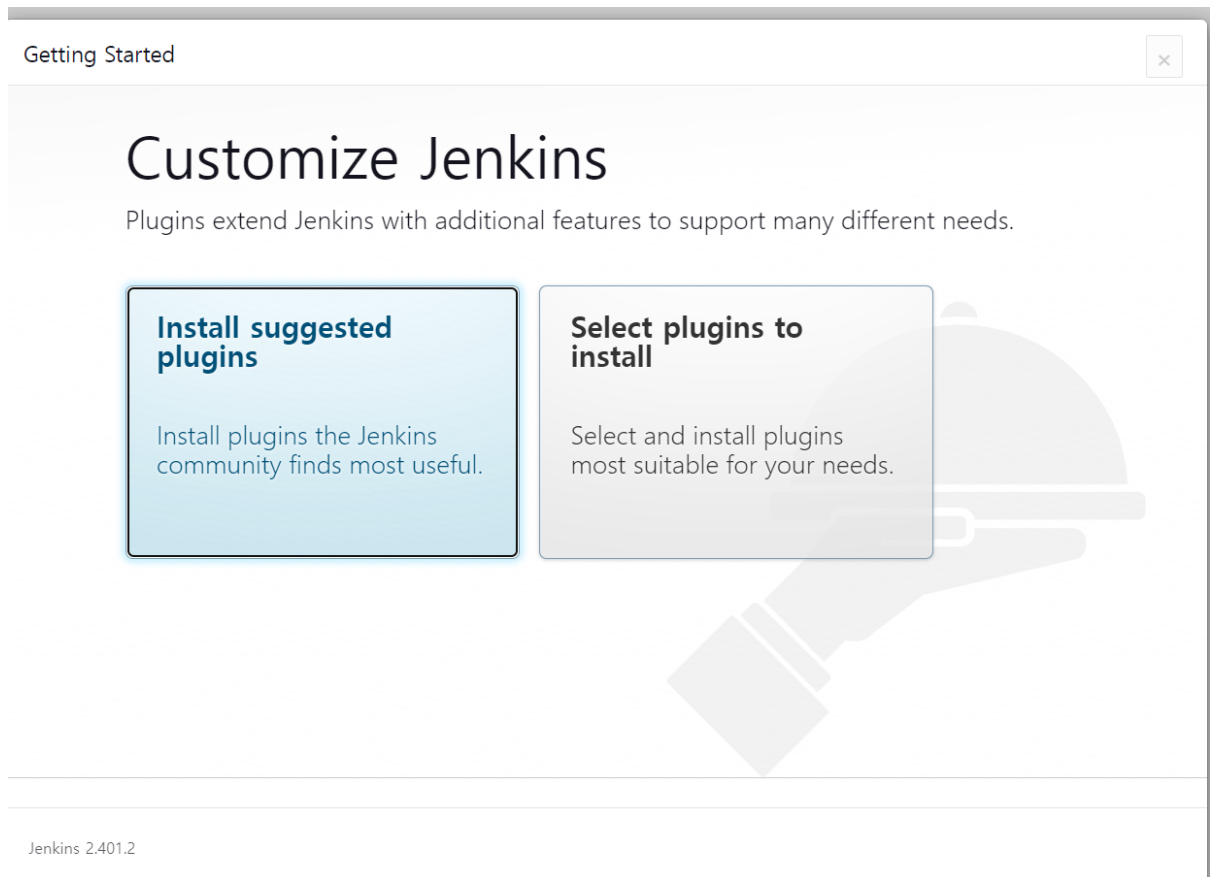
# 확인
sudo systemctl status jenkins
sudo systemctl start jenkins

# 초기 비밀번호 확인
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
# a49dc39585b14255a8d4ec8e1d512590
```

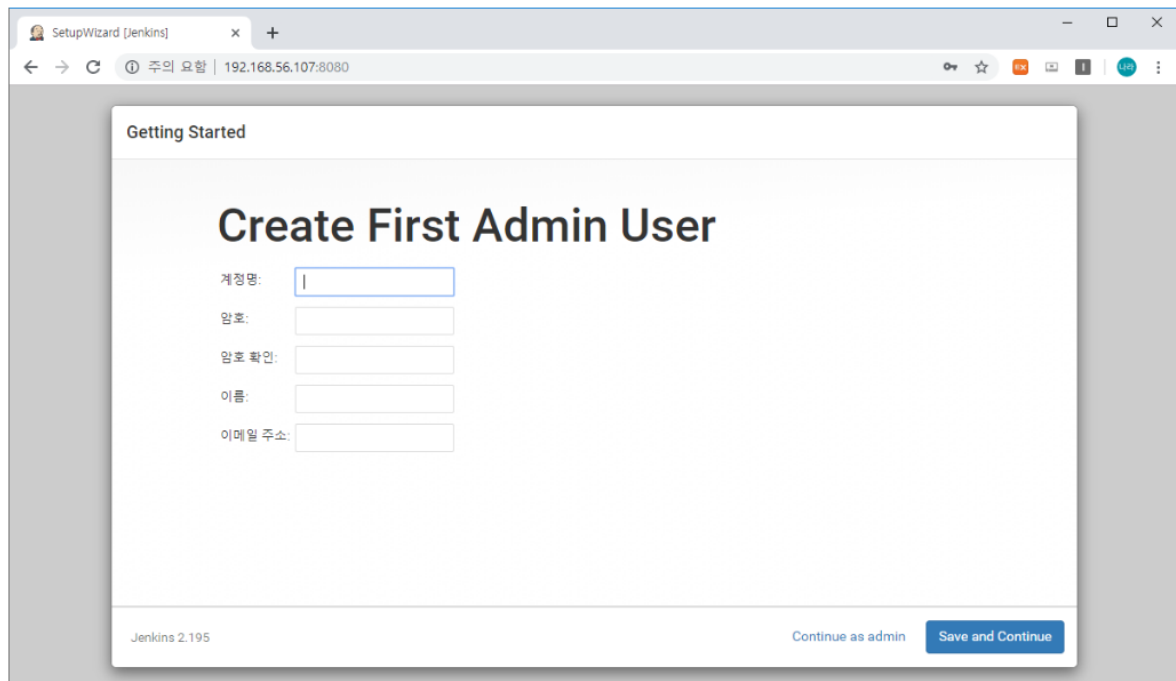


비밀번호 : a49dc39585b14255a8d4ec8e1d512590

플러그인 설치



관리자 계정 생성

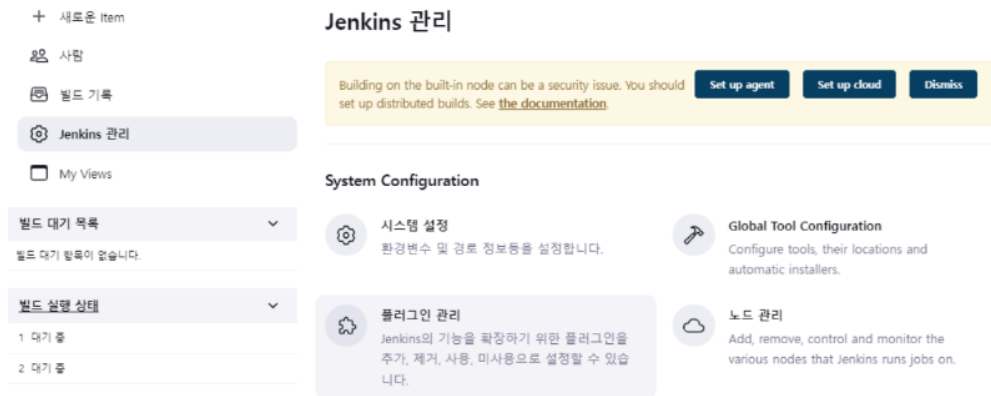


- 계정명 : ailingo
- 암호 : ailingo1234
- 이름 : psg

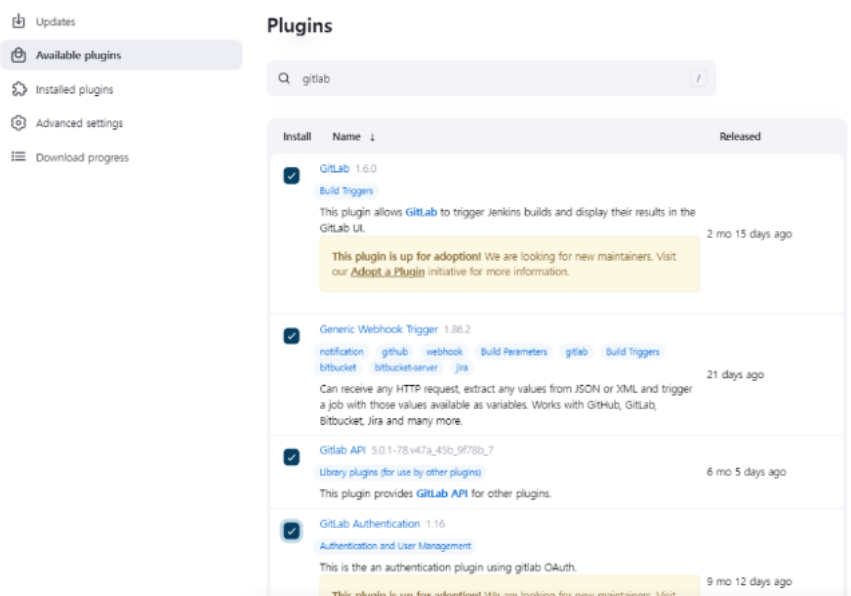
- 이메일 : psg980331@naver.com

Jenkins 프로젝트 설정

1. Jenkins 관리 > 플러그인 관리를 누른다.



Docker, Gitlab 관련 플러그인 설치





새로운 ITEM 생성 → Freestyle project


Enter an item name


airlingo


» A job already exists with the name 'airlingo'

 **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.

 **Maven project**
Maven 프로젝트를 빌드합니다. Jenkins은 POM 파일의 이점을 가지고 있고 급격히 설정을 줄입니다.

 **Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

 **Multi-configuration project**
다양한 환경에서의 테스트, 플러그인 특성 빌드, 기타 등등처럼 다수의 서로다른 환경설정이 필요한 프로젝트에 적합함.

 **Folder**
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder is a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

OK

- airlingo 미리 생성해놔서 빨간색 오류가 뜨는것!

Gitlab 연결

Kind

Username with password

Scope ?

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

Username ?

psg980331@naver.com

☐ Treat username as secret ?

Password ?

ID ?

airlingo

UserName	gitlab 계정 아이디
Password	gitlab 계정 비밀번호
id	Credential을 식별하는 아이디
Description	Credential에 대한 설명

- password 잘 확인...깃랩 패스워드 틀린줄 모르고 3시간 삼질...ah...

Configure

- General
- 소스 코드 관리**
- 빌드 유발
- 빌드 환경
- Build Steps
- 빌드 후 조치

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://i9a308.p.ssafy.io:8080/project/airlingo> ?

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 ?

- Advanced (고급) 클릭 - **Secret token** 메뉴의 Generate 클릭
- 토큰 복사 후 저장

secret token : 8c7a8dd659158efd7a24723d98f5dc03

- Build when a change is pushed to GitLab. 옆에 있는 GitLab webhook URL: 뒤의 **url**을 복사

GitLab repository에 입력

Webhooks

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

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

☒ Show full URL

☐ Mask portions of URL

Do not show sensitive data such as tokens in the UI.

Secret token

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

Trigger

☒ Push events

☐ Tag push events

A new tag is pushed to the repository

- Jenkins와 연결할 gitlab의 repository에서 Settings - Webhooks
- URL : 위에서 복사해두었던 url
- Secret token : 위에서 복사해두었던 토큰값
- Trigger - Push events : push event를 발생시킬 branch (비워두면 전체 branch)

GitLab Access Token 생성

Project Access Tokens

Generate project access tokens scoped to this project for your applications that need access to the GitLab API. You can also use project access tokens with Git to authenticate over HTTP(S). [Learn more.](#)

Add a project access token

Enter the name of your application, and we'll return a unique project access token.

Token name

For example, the application using the token or the purpose of the token. Do not give sensitive information for the name of the token, as it will be visible to all project members.

Expiration date

Select a role

Select scopes

Scopes set the permission levels granted to the token. [Learn more.](#)

- ☒ **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**
Grants read access (pull) to the repository.
- ☒ **write_repository**
Grants read and write access (pull and push) to the repository.

[Create project access token](#)

- GitLab의 연결할 repository - Settings - Access Tokens 생성
- access token이 새로 생성된 것을 확인하고, 복사

Access Token으로 GitLab에 연결

Jenkins 관리 - 시스템 설정 - Gitlab - Credentials - Add 클릭

Dashboard > 환경설정

Usage Statistics

☒ Help make Jenkins better by sending anonymous usage statistics and crash reports to the Jenkins project. ?

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain: Global credentials (unrestricted)

Kind: GitLab API token

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

API token:

ID ? : gitlab

Description ? :

Add Cancel

저장 Apply

Test Connection

- Kind : GitLab API token 선택

- API token : 위에서 생성한 GitLab Access Token
- ID : Credentials를 구분하는 고유값 (아무거나 상관 X)

GitLab

☒ Enable authentication for '/project' end-point ?

GitLab connections

Connection name ?

A name for the connection

airlingo

GitLab host URL ?

The complete URL to the GitLab server (e.g. http://gitlab.mydomain.com)

https://lab.ssafy.com

Credentials ?

API Token for accessing GitLab

GitLab API token

Add ▾

고급 ▾

- Connection name : Gitlab repository를 구분할 수 있는 고유값 (아무거나 상관 X)
- global host URL : 도메인 주소까지만 (ex : https://lab.ssafy.com)
- Credentials : 위에서 Access Token을 이용해 만든 credential 선택

General

설명

[Plain text] [미리보기](#)

- ☐ Commit agent's Docker container ?
- ☐ Run the build inside Docker containers
- ☐ Define a Docker template
- ☐ GitHub project

GitLab Connection

airlingo

- ☐ Use alternative credential

- 위에서 GitLab Access Token으로 Jenkins 환경설정에 등록해둔 GitLab Connection으로 선택

셸 스크립트 작성

Build Steps

≡ Execute shell ?

Command

See [the list of available environment variables](#)

pwd

고급 ▾

Add build step ▾

셸 스크립트 코드

```
BE_CONTAINER="airlingo_back_container"
BE_IMAGE="airlingo_back_image"
FE_CONTAINER="airlingo_front_container"
FE_IMAGE="airlingo_front_image"
MATCH_CONTAINER="airlingo_match_container"
MATCH_IMAGE="airlingo_match_image"

docker build -t $BE_IMAGE /var/lib/jenkins/workspace/airlingo/backend/airlingo
if [ "$(docker ps -a -q -f name=$BE_CONTAINER)" ]; then
    docker stop $BE_CONTAINER
    docker rm $BE_CONTAINER
fi
docker run -e TZ=Asia/Seoul -d --network="host" --name $BE_CONTAINER $BE_IMAGE
```

```

docker build --no-cache -t $FE_IMAGE /var/lib/jenkins/workspace/airlingo/frontend/airlingo
if [ "$(docker ps -a -q -f name=$FE_CONTAINER)" ]; then
    docker stop $FE_CONTAINER
    docker rm $FE_CONTAINER
fi
docker run -e TZ=Asia/Seoul --env-file /var/lib/jenkins/workspace/airlingo/frontend/airlingo/.env.production -d --network="host" --name=$FE_CONTAINER $FE_IMAGE

docker build -t $MATCH_IMAGE /var/lib/jenkins/workspace/airlingo/matching/airlingo
if [ "$(docker ps -a -q -f name=$MATCH_CONTAINER)" ]; then
    docker stop $MATCH_CONTAINER
    docker rm $MATCH_CONTAINER
fi
docker run -e TZ=Asia/Seoul -d --network="host" --name $MATCH_CONTAINER $MATCH_IMAGE

docker image prune -f

```

9. RabbitMQ 설치



```

sudo apt update
sudo apt install -y erlang
sudo apt install rabbitmq-server

```

```

#rabbitmq-env.conf 파일 포트 수정 -> 5432
sudo systemctl restart rabbitmq-server

```

```

#유저 추가 및 권한 부여
sudo rabbitmqctl add_user admin admin
sudo rabbitmqctl set_user_tags admin administrator
sudo rabbitmqctl set_permissions -p / admin ".*" ".*" ".*"

```

10. NODE 설치 && YJS 시그널링 서버 설치



#NodeSource 저장소 설정

```
curl -sL https://deb.nodesource.com/setup\_11.x | sudo -E bash -
```

#Node.js 설치

```
sudo apt-get install -y nodejs
```

#설치 확인

```
node -v
```

```
npm -v
```

#y-webrtc설치

```
cd /etc
```

```
mkdir ywebrtc-server && cd ywebrtc-server
```

```
npm init -y
```

```
npm install y-webrtc
```

#y-webrtc 서버 실행

```
cd /etc/ywebrtc-server
```

```
nohup env PORT=3333 node ./node_modules/y-webrtc/bin/server.js &
```

#서버 끄기

```
ps aux | grep y-webrtc
```

```
kill -9 YOUR_PID
```

#y-websocket설치

```
cd /etc
```

```
mkdir websocket-server && cd websocket-server
```

```
npm init -y
```

```
npm install y-websocket
```

#y-websocket server 실행

```
cd /etc/websocket-server
```

```
nohup env PORT=3334 HOST=localhost node ./node_modules/y-websocket/bin/server.js &
```

#서버 끄기

```
lsof -t -i:3333
```

```
lsof -t -i:3334
```

```
kill -9 pid
```

11. 지급받은 EC2에 Openvidu CE 설치

```
#OpenVidu 배포하기 위해 루트 권한 필요
```

```
sudo su
```

```
#OpenVidu 설치 권장 폴더 /opt 이동
```

```
cd /opt
```

```
#OpenVidu 설치
```

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

```
#설치 후 openvidu 폴더로 이동
```

```
cd openvidu
```

```
#도메인 설정과 OpenVidu 통신을 위한 Secret 값이 담긴 설정 파일 작성
```

```
nano .env
```

```

#포트 설정
ufw allow 80/tcp
ufw allow 443/tcp
ufw allow 3478/tcp
ufw allow 3478/udp
ufw allow 40000:57000/tcp
ufw allow 40000:57000/udp
ufw allow 57001:65535/tcp
ufw allow 57001:65535/udp
ufw enable

#실행
sudo ./openvidu start

#그렇지만 안되서 이것저것 만지다가 openvidu,redis,nginx 다 삭제
#redis,nginx 깔기전에 openvidu 먼저 배포!

#OpenVidu 배포하기 위해 루트 권한 필요
sudo su

#OpenVidu 설치 권장 폴더 /opt 이동
cd /opt

#OpenVidu 설치
curl https://s3-eu-west-1.amazonaws.com/aws.openvidu.io/install_openvidu_latest.sh | bash

#설치 후 openvidu 폴더로 이동
cd openvidu

#도메인 설정과 OpenVidu 통신을 위한 Secret 값이 담긴 설정 파일 작성
nano .env

HTTP_PORT=8442
HTTPS_PORT=8443
이렇게 설정하고
./openvidu start 하니까
ssl인증서 발급을 못받음
다시 삭제하고
첫 시작은
HTTP_PORT=80
HTTPS_PORT=443
으로 실행 시켜주었더니 인증서를 잘 발급 받았다.

#NGINX 다시 설치
#OPENVIDU 내에 인증서를 설치한 인증서에 적용시켜주었음

```

```
# Domain name. If you do not have one, the public IP of the machine
# For example: 198.51.100.1, or openvidu.example.com
DOMAIN_OR_PUBLIC_IP=i9a308.p.ssafy.io

# OpenVidu SECRET used for apps to connect to OpenVidu server and
OPENVIDU_SECRET=airlingo

# Certificate type:
# - selfsigned: Self signed certificate. Not recommended for production
# Users will see an ERROR when connected to web page
# - owncert: Valid certificate purchased in a Internet service
# Please put the certificates files inside folder
# with names certificate.key and certificate.cert
# - letsencrypt: Generate a new certificate using letsencrypt. Please
# required contact email for Let's Encrypt in LETSENCRYPT_EMAIL
# variable.
CERTIFICATE_TYPE=letsencrypt

# If CERTIFICATE_TYPE=letsencrypt, you need to configure a valid
LETSENCRYPT_EMAIL=psg980331@naver.com

# Proxy configuration
# If you want to change the ports on which openvidu listens, uncomment
# Allows any request to http://DOMAIN_OR_PUBLIC_IP:HTTP_PORT/ to
# redirected to https://DOMAIN_OR_PUBLIC_IP:HTTPS_PORT/.
# WARNING: the default port 80 cannot be changed during the first
# if you have chosen to deploy with the option CERTIFICATE_TYPE=letsencrypt
HTTP_PORT=8442

# Changes the port of all services exposed by OpenVidu.
# SDKs, REST clients and browsers will have to connect to this port
HTTPS_PORT=8443

# Old paths are considered now deprecated, but still supported by
```

12. OPENVIDU PRO버전 배포

CloudFormation 작성

사전 조건 - 템플릿 준비

템플릿 준비

모든 스택은 템플릿을 기반으로 합니다. 템플릿은 JSON 또는 YAML 텍스트 파일로, 스택에 포함하려는 AWS 리소스에 대한 구성 정보가 들어 있습니다.

☒ 준비된 템플릿

☐ 샘플 템플릿 사용

☐ Designer에서 템플릿 생성

템플릿 지정

템플릿은 스택의 리소스와 속성을 설명하는 JSON 또는 YAML 파일입니다.

템플릿 소스

템플릿을 선택하면 템플릿이 저장될 Amazon S3 URL이 생성됩니다.

☒ Amazon S3 URL

☐ 템플릿 파일 업로드

Amazon S3 URL

Amazon S3 템플릿 URL

S3 URL: https://s3-eu-west-1.amazonaws.com/aws.openvidu.io/CF-OpenVidu-Pro-latest.yaml

Designer에서 보기

상세 정보 입력

AirLingo 포팅매뉴얼

19

Domain and SSL certificate configuration

Certificate Type

[selfsigned] Self signed certificate. Not recommended for production use. [owncert] Valid certificate purchased in a Internet services company. [letsencrypt] Generate a new certificate using Let's Encrypt.

letsencrypt ▼

AWS Elastic IP (EIP)

Previously created AWS Elastic IP to associate it to the OpenVidu EC2 instance. If certificate type is 'selfsigned' this value is optional. If certificate type is 'owncert' or 'letsencrypt' this value is mandatory. Example 13.33.145.23.

airlingo.site

Domain Name pointing to Elastic IP

Valid domain name pointing to previous IP. If certificate type is 'selfsigned' this value is optional. If certificate type is 'owncert' or 'letsencrypt' this value is mandatory. Example: openvidu.company.com

1.2.3.4

URL to the CRT file (owncert)

If certificate type is 'owncert' this is the URL where CRT file will be downloaded

String 입력

URL to the key file (owncert)

If certificate type is 'owncert' this is the URL where KEY file will be downloaded

String 입력

Email for Let's Encrypt (letsencrypt)

If certificate type is 'letsencrypt', this email will be used for Let's Encrypt notifications

example@qwe.com

OpenVidu configuration

OpenVidu Pro License key

Visit <https://openvidu.io/account>

Which OpenVidu Edition you want to deploy

Visit <https://docs.openvidu.io/en/stable/deployment/#openvidu-editions>

pro ▼

Openvidu Secret

Secret to connect to this OpenVidu Platform. Cannot be empty and must contain only alphanumeric characters [a-zA-Z0-9], hyphens ('-') and underscores ('_')

Initial number of Media Node in your cluster

How many Media Nodes do you want on startup (EC2 instances will be launched)

1

OpenVidu Recording Configuration

OpenVidu Recording

If 'disabled', recordings will not be active. If 'local' recordings will be saved in EC2 instance locally. If 's3', recordings will be stored in a S3 bucket*

s3 ▼

S3 Bucket where recordings will be stored

S3 Bucket Name

bucketname

Elasticsearch and Kibana configuration

Enable Elasticsearch and Kibana

Choose if you want OpenVidu to use Elasticsearch.

false ▼

Elasticsearch URL

If you have an external Elasticsearch service running, put here the url to the service. If empty, an Elasticsearch service will be deployed next to OpenVidu. ('ElasticSearch Enabled' must be true)

String 입력

Kibana URL

If you have an external Kibana service running, put here the url to the service. If empty, a Kibana service will be deployed next to OpenVidu. ('ElasticSearch Enabled' must be true)

String 입력

Elasticsearch and Kibana username

Username for Elasticsearch and Kibana. ('ElasticSearch Enabled' must be true)

elasticadmin

Elasticsearch and Kibana password

Password for Elasticsearch and Kibana ('ElasticSearch Enabled' must be true)

String 입력

EC2 Instance configuration

Instance type for Openvidu Server Pro Node

Specifies the EC2 instance type for your OpenVidu Server Pro Node

t2.large ▼

Instance type for Media Nodes

Specifies the EC2 instance type for your Media Nodes

t2.large ▼

SSH Key

Name of an existing EC2 KeyPair to enable SSH access to the instance. It is mandatory to perform some administrative tasks of OpenVidu.

AWS::EC2::KeyPair::KeyName 선택 ▼

작성 후 배포!

도커 파일 - Back

```
FROM gradle:jdk17 as builder
#working directory 경로
WORKDIR /usr/src/app
#ARG key=value , 여러번 사용되는 문자열이나 숫자 등을 변수로 만들어주는 속성
#ARG JAR_PATH=./build/libs/*.jar
#COPY ( 복사할 파일[컨테이너 외부] ) ( 복사될 위치[컨테이너 내부] )
COPY . /usr/src/app
#빌드 결과가 /usr/src/app/build/libs/project-name.version.jar 로 생성
RUN gradle build --no-daemon -x test

FROM openjdk:17-alpine
EXPOSE 8081
WORKDIR /usr/src/app
#--from=builder /usr/src/app/build/libs/*.jar 이전 임시 컨테이너에 생성된 파일
#결과 /usr/src/app 밑에 airlingo-0.0.1-SNAPSHOT.jar파일을 복사!
COPY --from=builder /usr/src/app/build/libs/*.jar ./
#환경에따라 빌드 해주는 명령어
CMD ["java", "-jar", "-Dspring.profiles.active=prod", "./airlingo-0.0.1-SNAPSHOT.jar"]
```

도커 파일 - Front

```
#베이스 이미지를 도커 허브에서 가져옴
FROM node:lts
#해당 어플의 소스코드들이 들어가게 됨
WORKDIR /usr/src/app
#소스코드가 바뀔때마다 종속성까지 다시 복사를 해주는 수고를 덜기위해 종속성 목록을 담고있는
#package.json 복사
COPY package*.json .
# .env.production 파일을 .env로 복사
COPY .env.production ./env
#종속성 다운
RUN npm install --no-optional
RUN npm install websocket
#모든 소스코드들 WORKDIR로 복사
COPY . .
# VITE_OPEN 환경변수 설정
ENV VITE_OPEN=false
EXPOSE 5173
CMD ["npm", "run", "dev"]
```

도커 파일 - Matching

```
FROM gradle:jdk17 as builder
#working directory 경로
WORKDIR /usr/src/app
#ARG key=value , 여러번 사용되는 문자열이나 숫자 등을 변수로 만들어주는 속성
#ARG JAR_PATH=./build/libs/*.jar
#COPY ( 복사할 파일[컨테이너 외부] ) ( 복사될 위치[컨테이너 내부] )
COPY . /usr/src/app
#빌드 결과가 /usr/src/app/build/libs/project-name.version.jar 로 생성
RUN gradle build --no-daemon -x test

FROM openjdk:17-alpine
```

```

EXPOSE 8082
WORKDIR /usr/src/app
#--from=builder /usr/src/app/build/libs/*.jar 이전 임시 컨테이너에 생성된 파일
#결국 /usr/src/app 밑에 airlingo-0.0.1-SNAPSHOT.jar파일을 복사!
COPY --from=builder /usr/src/app/build/libs/*.jar ./
#환경에 따라 빌드 해주는 명령어
CMD ["java","-jar","-Dspring.profiles.active=prod","./airlingo-0.0.1-SNAPSHOT.jar"]

```

메인서버 설정파일(application.yml)

```

spring:
  profiles: # profiles 설정
    active: dev # 다른 설정이 없을 때 default 환경 값
  # jpa설정
  jpa:
    properties: # property 사용 설정
      hibernate: # hibernate property 설정
        format_sql: true # 보여지는 쿼리를 예쁘게 포맷팅 -> 사용하지 않으면 긴 줄 형태로 출력됨
    servlet:
      multipart:
        enabled: true
        max-file-size: 20MB
        max-request-size: 100MB

server:
  port: 8081

# 로그 레벨 설정
logging:
  level:
    # hibernate 가 남기는 모든 로그가 debug모드로 설정
    # jpa hibernate가 생성하는 sql이 로그를 통해서 찍히도록 하는 설정
    org.hibernate.SQL: debug
    org.hibernate.orm.jdbc.bind: trace
    org.springframework.web.socket: trace

# Swagger springdoc-ui Configuration
springdoc:
  packages-to-scan: com.ssafy.airlingo
  default-consumes-media-type: application/json;charset=UTF-8
  default-produces-media-type: application/json;charset=UTF-8
  swagger-ui:
    path: demo-ui.html
    tags-sorter: alpha # alpha: 알파벳 순 태그 정렬, method: HTTP Method 순 정렬
    operations-sorter: alpha # alpha: 알파벳 순 태그 정렬, method: HTTP Method 순 정렬
  api-docs:
    path: /api-docs/json
    groups:
      enabled: true
  cache:
    disabled: true

#AWS
cloud:
  aws:
    s3:
      bucket: ${AIRLINGO_BUCKET}
    credentials:
      accessKey: ${AIRLINGO_ACCESS-KEY}
      secretKey: ${AIRLINGO_SECRET-KEY}
    region:
      static: ap-northeast-2
      auto: false
    stack:
      auto: false

```

메인서버 설정파일(application-dev.yml)

```

spring:
  jpa:
    hibernate:
      ddl-auto: create #create update none
    datasource:

```

```

driver-class-name: com.mysql.cj.jdbc.Driver
url: jdbc:mysql://localhost:3306/airlingo?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=Asia/Seoul&characterEncoding=UTF-8
username: ${AIRLINGO_USERNAME}
password: ${AIRLINGO_PASSWORD}
rabbitmq:
  host: localhost
  port: 5672
  username: ${AIRLINGO_ADMIN_USERNAME}
  password: ${AIRLINGO_ADMIN_PASSWORD}

# redis 설정
server:
  data:
    redis:
      host: localhost
      port: 6379
      timeout: 5000 # 서버와의 연결 시도 및 응답 대기 시간 최대 5초

#openvidu
openviduUrl: ${OPENVIDU_URL}
openviduSecret: ${OPENVIDU_SECRET}

```

메인서버 설정파일(application-prod.yml)

```

spring:
  jpa:
    hibernate:
      ddl-auto: none #create update none
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://domain/airlingo?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=Asia/Seoul&characterEncoding=UTF-8
    username: ${AIRLINGO_USERNAME}
    password: ${AIRLINGO_PASSWORD}
  rabbitmq:
    host: localhost
    port: 5432
    username: ${AIRLINGO_ADMIN_USERNAME}
    password: ${AIRLINGO_ADMIN_PASSWORD}

# redis 설정
server:
  data:
    redis:
      host: localhost
      port: 6379
      timeout: 5000 # 서버와의 연결 시도 및 응답 대기 시간 최대 5초

#openvidu
openviduUrl: ${OPENVIDU_URL}
openviduSecret: ${OPENVIDU_SECRET}

```

매칭서버 설정파일(application.yml)

```

spring:
  profiles: # profiles 설정
    active: dev # 다른 설정이 없을 때 default 환경 값

server:
  port: 8082

# Swagger springdoc-ui Configuration
springdoc:
  packages-to-scan: com.ssafy.airlingo
  default-consumes-media-type: application/json;charset=UTF-8
  default-produces-media-type: application/json;charset=UTF-8
  swagger-ui:
    path: demo-ui.html
    tags-sorter: alpha # alpha: 알파벳 순 태그 정렬, method: HTTP Method 순 정렬
    operations-sorter: alpha # alpha: 알파벳 순 태그 정렬, method: HTTP Method 순 정렬
  api-docs:
    path: /api-docs/json
    groups:
      enabled: true

```

```
cache:
  disabled: true
```

매칭서버 설정파일(application-dev.yml)

```
spring:
  rabbitmq:
    host: localhost
    port: 5672
    username: ${AIRLINGO_ADMIN_USERNAME}
    password: ${AIRLINGO_ADMIN_PASSWORD}
```

매칭서버 설정파일(application-prod.yml)

```
spring:
  rabbitmq:
    host: localhost
    port: 5432
    username: ${AIRLINGO_ADMIN_USERNAME}
    password: ${AIRLINGO_ADMIN_PASSWORD}
```

프론트 .env.production파일

```
VITE_SERVER_URL=https://i9a308.p.ssafy.io/
VITE_SOCKET_URL=https://i9a308.p.ssafy.io/ws
VITE_CHAT_SOCKET_URL=https://i9a308.p.ssafy.io/chat
```

외부서비스 사용

1. AWS Lambda
2. Naver Clova Speech API
3. Papago API

AWS Lambda 생성



S3에 저장된 .WEBM파일의 음성을 .MP4로 바꾸기 위해
S3에 .WEBM파일이 저장될때 실행될 Lambda 작성

IAM 역할 생성

신뢰할 수 있는 엔터티 선택 정보

신뢰할 수 있는 엔터티 유형

<input checked="" type="radio"/> AWS 서비스 EC2, Lambda 등의 AWS 서비스가 이 계정에서 작업을 수행하도록 허용합니다.	<input type="radio"/> AWS 계정 사용자 또는 서드 파티에 속한 다른 AWS 계정의 엔터티가 이 계정에서 작업을 수행하도록 허용합니다.	<input type="radio"/> 웹 자격 증명 지정된 외부 웹 자격 증명 공급자와 연동된 사용자가 이 역할을 맡아 이 계정에서 작업을 수행하도록 허용합니다.
<input type="radio"/> SAML 2.0 연동 기업 디렉터리에서 SAML 2.0과 연동된 사용자가 이 계정에서 작업을 수행할 수 있도록 허용합니다.	<input type="radio"/> 사용자 지정 신뢰 정책 다른 사용자가 이 계정에서 작업을 수행할 수 있도록 사용자 지정 신뢰 정책을 생성합니다.	

사용 사례

EC2, Lambda 등의 AWS 서비스가 이 계정에서 작업을 수행하도록 허용합니다.

일반 사용 사례

- ☐ **EC2**
Allows EC2 instances to call AWS services on your behalf.
- ☒ **Lambda**
Allows Lambda functions to call AWS services on your behalf.

다른 AWS 서비스의 사용 사례:

사용 사례를 조회할 서비스 선택

AWS - 서비스 - IAM - 역할 만들기 클릭

1. 신뢰할 수 있는 유형의 개체 선택 - AWS 서비스사용 사례 선택 - Lambda
2. 권한 정책 연결S3, CloudWatch Logs 권한이 필요함 (상황에 따라 strict 하게 변경)

AmazonS3FullAccess

CloudWatchLogsFullAccess

1. 태그 추가 (선택 사항) 건너뛰
2. 검토
 - a. 역할 이름: lambda-ffmpeg-role

FFmpeg 계층 생성

계층을 생성하여 bin 파일을 실행할 수 있다

FFmpeg 다운

1. <https://johnvansickle.com/ffmpeg/> 홈페이지로 이동
2. release: 4.4 - ffmpeg-release-amd64-static.tar.xz 다운 및 압축 해제
3. ffmpeg-release-amd64-static 풀더 ffmpeg.zip 으로 압축

S3 업로드

ffmpeg.zip 파일을 S3 에 업로드

Lambda 계층 생성

AWS - Lambda - 추가 리소스 - 계층 - 계층 생성 클릭

1. Amazon S3에서 파일 업로드 클릭하여 ffmpeg.zip 파일 경로 입력


계층 구성

이름

설명 - 선택 사항

☐ .zip 파일 업로드
☒ Amazon S3에서 파일 업로드


Amazon S3 링크 URL
함수 코드 .zip에 S3 링크 URL을 붙여넣습니다.


 This field is too short. The minimum length is 10 characters.


호환 아키텍처 - 선택 사항 정보
계층에 대해 호환 명령 세트 아키텍처를 선택합니다.

☒ x86_64
☐ arm64

호환 런타임 - 선택 사항 정보
최대 15개의 런타임을 선택합니다.





Python 3.8 

라이선스 - 선택 사항 정보

Lambda 함수 생성

AWS - Lambda - 함수 - 함수 생성

1. 런타임 설정
2. 기존 역할 사용 - lambda-ffmpeg-role 설정
3. 함수 생성

함수 이름

함수의 용도를 설명하는 이름을 입력합니다.

webmToMp3

공백 없이 문자, 숫자, 하이픈 또는 밑줄만 사용합니다.

런타임 정보

함수를 작성하는 데 사용할 언어를 선택합니다. 콘솔 코드 편집기는 Node.js, Python 및 Ruby만 지원합니다.

Python 3.8



아키텍처 정보

함수 코드에 대해 원하는 명령 세트 아키텍처를 선택합니다.

- ☒ x86_64
☐ arm64

권한 정보

기본적으로 Lambda는 Amazon CloudWatch Logs에 로그를 업로드하는 권한을 가진 실행 역할을 생성합니다. 이 기본 역할은 나중에 트리거를 추가할 때 사용자 지정할 수 있습니다.

▼ 기본 실행 역할 변경

실행 역할

함수에 대한 권한을 정의하는 역할을 선택합니다. 사용자 지정 역할을 생성하려면 [IAM 콘솔](#)로 이동하십시오.

- ☐ 기본 Lambda 권한을 가진 새 역할 생성
☒ 기존 역할 사용
☐ AWS 정책 템플릿에서 새 역할 생성

기존 역할

생성한 기존 역할 중에 이 Lambda 함수와 함께 사용할 역할을 선택합니다. 이 역할에는 Amazon CloudWatch Logs에 로그를 업로드할 수 있는 권한이 있어야 합니다.

lambda-ffmpeg-role



IAM 콘솔에서 [lambda-ffmpeg-role](#) 역할을 확인합니다.

Layers 추가

계층 - [Add a layer] 클릭

1. 사용자 지정 계층 - ffmpeg 선택

Lambda > 계층 > 계층 추가

계층 추가

계층 선택 Info

호환되는 런타임이 포함된 계층을 선택하거나 계층 버전의 Amazon 리소스 이름(ARN)을 지정합니다.

☐ AWS 계층

AWS에서 제공하는 계층 목록에서 계층을 선택합니다.

☒ 사용자 지정 계층

AWS 계층 또는 조직에서 생성한 계층 목록에서 계층을 선택합니다.

☐ ARN 지정

ARN을 제공하여 계층을 지정합니다.

사용자 지정 계층

함수의 런타임과 호환되는 AWS 계층 또는 조직 생성 계층입니다.

ffmpeg

버전

2

취소 **추가**

트리거 추가

1. 버킷 - 트리거 적용할 버킷 설정
2. 이벤트 유형 설정
3. 접미사 - .webm 파일이 업로드 될때만 트리거되도록 한다
4. 재귀 호출 발생할 수 있는지 한번 더 생각해보기

**S3**

aws asynchronous storage



Bucket

Please select the S3 bucket that serves as the event source. The bucket must be in the same region as the function.



Bucket region: ap-northeast-2

Event types

Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.

All object create events

Prefix - 선택 사항

Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters.

Suffix - 선택 사항

Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters.

Recursive invocation

If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. [Learn more](#)

- ☒ I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.

Lambda 코드 작성

```

import subprocess
import boto3
import os

def lambda_handler(event, context):

    # S3 정보 추출
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = event['Records'][0]['s3']['object']['key']
    print(bucket)
    print(key)

    # S3에서 파일 다운로드
    s3_client = boto3.client('s3')
    download_path = '/tmp/{}'.format(key)
    os.makedirs(os.path.dirname(download_path), exist_ok=True) # 중간 디렉터리 생성
    s3_client.download_file(bucket, key, download_path)

    # 출력 경로 설정
    output_path = '/tmp/{}'.format(key.replace('.webm', '.mp3'))
    os.makedirs(os.path.dirname(output_path), exist_ok=True)

    # ffmpeg 실행
    command = "/opt/ffmpeg-4.4-amd64-static/ffmpeg -i {} -acodec libmp3lame {}".format(download_path, output_path)
    result = subprocess.call(command, shell=True)
    if result != 0:
        print("ffmpeg command failed with error code", result)
    else:
        print("ffmpeg command succeeded")

    # 결과 파일 S3에 업로드

```

```
s3_client.upload_file(output_path, bucket, key.replace('.webm', '.mp3'))

return {
    'statusCode': 200,
    'body': 'Conversion successful!'
}
```

NAVER CLOVA Speech 사용법

이용 신청

1. 네이버 클라우드 플랫폼 콘솔에 접속
2. **Region** 메뉴에서 이용 중인 리전을 클릭
3. **Platform** 메뉴에서 이용 중인 플랫폼을 클릭
4. **Services > AI Services > CLOVA Speech** 메뉴를 차례대로 클릭
5. **Subscription** 메뉴를 클릭
6. **[이용 신청]** 버튼을 클릭
7. **[이용 신청]** 버튼을 클릭
8. 알림 팝업 창이 나타나면 **[확인]** 버튼을 클릭.

도메인 생성

< 도메인 생성

도메인 정보

도메인의 기본 정보를 입력합니다. (필수 입력 항목입니다.)

도메인 이름

airlingo

도메인 코드

airlingo

코드 사용 가능

유형 선택

☐ Free
 ☒ Basic

Storage 설정

인식 대상 파일이 저장된 Object Storage 경로와 인식 결과물을 저장할 Object Storage 경로를 입력합니다. (필수 입력 항목입니다.)

결과 파일 저장 경로

airlingo/

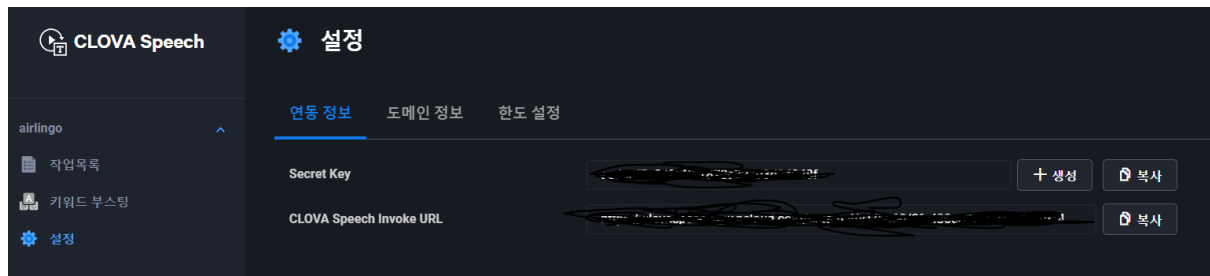
Object Storage 바로가기

인식 대상 저장 경로

airlingo/

Object Storage 바로가기

도메인 들어가서 설정의 키 사용!



PAPAGO API

애플리케이션 등록 [↗](#)

네이버 개발자 센터에서 애플리케이션을 등록하는 방법은 다음과 같습니다.

1. 네이버 개발자 센터의 메뉴에서 [Application](#) > [애플리케이션 등록](#)을 선택합니다.
2. 이용약관 동의 단계에서 [이용약관에 동의합니다.](#)를 선택한 다음 [확인](#)을 클릭합니다.
3. 계정 정보 등록 단계에서 휴대폰 인증을 완료하고 회사 이름을 입력한 다음 [확인](#)을 클릭합니다. 휴대폰 인증은 담당자 연락처 확인을 위해 필요한 과정이며, 애플리케이션을 처음 등록할 때 한 번만 인증받으면 됩니다.
4. 애플리케이션 등록 (API이용신청) 페이지에서 [애플리케이션 등록 세부 정보](#)를 입력한 다음 등록하기를 클릭합니다.

애플리케이션 등록 세부 정보 [↗](#)

애플리케이션 등록 (API이용신청) 페이지에서 애플리케이션 세부 정보를 입력하는 방법은 다음과 같습니다.

1. 등록하려는 애플리케이션의 이름을 [애플리케이션 이름](#)에 입력합니다. 최대 40자까지 입력할 수 있습니다.
2. [사용 API](#)에서 Papago 번역을 선택해 추가합니다.
3. [비로그인 오픈 API 서비스 환경](#)에서 애플리케이션을 서비스할 환경을 추가하고 필요한 상세 정보를 입력합니다.

애플리케이션 등록

애플리케이션 이름	<input type="text" value="Papago-NMT-TEST"/> ✓ <ul style="list-style-type: none"> • 네이버 아이디로 로그인할 때 사용자에게 표시되는 이름이므로 가급적 10자 이내의 간결한 이름을 사용해주세요. • 40자 이내의 영문, 한글, 숫자, 공백문자, "-", "_" 만 입력 가능합니다.
사용 API ↗	<div> <input type="text" value="선택하세요."/> ▼ ✓ </div> <div> <input type="text" value="Papago 번역"/> ✕ </div>
비로그인 오픈 API 서비스 환경	<input type="text" value="환경 추가"/> ▼ ①

애플리케이션 키 확인

내 애플리케이션

Papago-NMT-TEST

Papago-detectLangs-TEST

Romanization-TEST

애플리케이션 등록

Clova Platform Console *β*

API 재휴 신청

계정 설정

Papago-NMT-TEST

개요

API 설정

멤버관리

로그인 통계

API 통계

Playground(Beta)

애플리케이션 정보

Client ID

8a3d64a2mG7m9Cc4dDdP

Client Secret

.....

보기