삵음삵음 포팅 매뉴얼

생성자	hyuntall
생성 일시	@2023년 8월 17일 오후 10:30
태그	

1. 기술 스택

Front-end

- IDE : Visual Studio Code v1.81.1
- Front-end Framework : react v18.2.0
- Web Server : nginx/1.18.0 (Ubuntu)

Back-end

- IDE : STS v4.19.1, IntelliJ 2023.1.4
- JVM : JavaSE-17
- Back-end Framework : Spring Boot v3.1.1
- Database : MySql v8.0.34

3rd Party Library

- 화상통화 서비스 : openvidu/openvidu-dev:2.28.0
- Message Broker : redis v6.2.4
- MediaPipe : @mediapipe/task-vision v0.10.3
- FACE-API : face-api.js v0.22.2

2. 배포 방법 및 EC2 설정

배포 시 주의사항

- **Free ports inside the server**: OpenVidu platform services will need the following ports to be available in the machine: 80, 443, 3478, 5442, 5443, 6379 and 8888. If some of these ports is used by any process, OpenVidu platform won't work correctly. It is a typical error to have an NGINX process in the system before installing OpenVidu. Please uninstall it.

⚠️ **openvidu 자체적으로 redis, nginx 등을 사용하기 때문에, openvidu를 먼저 설치하지 않으면 포트 충돌이 발생하여 실행이 제대로 되지 않을 수 있습니다.**

URL 설정

```
// src/apiConfig.js.에 적절한 도메인 주소를 입력합니다.

export const serverUrl = "https://i9a702.p.ssafy.io";
export const clientUrl = "i9a702.p.ssafy.io";
```

포트 개방

App	EC2 Port	컨테이너 Port
React	3000	3000

App	EC2 Port	컨테이너 Port
Spring	8081	8081
MySQL	3306	x
Redis	4000	x
openvidu	8443	8443
coturn	3478	3478
Kurento	8888	8888
Jenkins	8000	8000
nginx http	80	x
nginx https	443	x

```
sudo su
#포트 설정
ufw allow 80/tcp
ufw allow 443/tcp
ufw allow 3000
ufw allow 4000
ufw allow 8080
ufw allow 8081
ufw allow 8442
ufw allow 8443
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
```

서버 시간 설정

```
date # 현재 시간 확인
timedatectl list-timezones # 시간 설정 목록 확인
sudo timedatectl set-timezone Asia/Seoul # 서울 시간으로 변경
```

Java, git 설치

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

openvidu 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

#실행
sudo ./openvidu start
```

openvidu .env 설정

```
DOMAIN_OR_PUBLIC_IP="도메인주소"

OPENVIDU_SECRET="패스워드"

CERTIFICATE_TYPE=letsencrpyt

LETSENCRYPT_EMAIL="이메일주소"

HTTP_PORT="http 접근에 사용할 포트번호"

HTTPS_PORT="https 접근에 사용할 포트번호"
```

```
# OpenVidu configuration
# ----------------------
# Documentation: https://docs.openvidu.io/en/stable/reference-docs/openvidu-config/

# NOTE: This file doesn't need to quote assignment values, like most shells do.
# All values are stored as-is, even if they contain spaces, so don't quote them.

# 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=i9a702.p.ssafy.io

# OpenVidu SECRET used for apps to connect to OpenVidu server and users to access to OpenVidu Dashboard
OPENVIDU_SECRET=sarkemsarkem

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

# If CERTIFICATE_TYPE=letsencrypt, you need to configure a valid email for notifications
LETSENCRYPT_EMAIL=hyunchul1998@naver.com

# Proxy configuration
# If you want to change the ports on which openvidu listens, uncomment the following lines

# Allows any request to http://DOMAIN_OR_PUBLIC_IP:HTTP_PORT/ to be automatically
# redirected to https://DOMAIN_OR_PUBLIC_IP:HTTPS_PORT/.
# WARNING: the default port 80 cannot be changed during the first boot
# 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
```

.env 설정 시 주의사항

⚠️ **openvidu 시작 전에,**
HTTP_PORT=80
HTTPS_PORT=443
으로 설정해야 정상적으로 letsencrypt 인증서를 발급받을 수 있습니다.
인증서 발급이 완료된 이후에 포트 번호를 변경해주세요.

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 'sarkem'@'%'identified by 'sarkem';
grant all privileges on *.*to 'sarkem'@'%';
flush privileges;
```

docker 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 /ect/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
```

Redis 설치

```
#게임 플로우 실시간 통신을 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

# Redis 재실행
sudo systemctl restart redis-server

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

#기본 포트인 6379에서 실행되고 있는지 확인
netstat -nlpt | grep 4000

#(netstat 명령어가 없어서 net-tools 패키지 설치함)
sudo apt install net-tools

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

redis.conf

```
# openvidu 내에서 개별적으로 레디스를 사용중이므로 포트번호 변경
port 4000

#원격 액세스 가 가능하도록 서버 개방
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
```

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

#ssl 설정후 443포트 허용
sudo ufw allow 443

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

default.conf

```
# 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 i9a702.p.ssafy.io; # 사용할 url로 변경
  #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;

  proxy_connect_timeout 1d;
  proxy_send_timeout 1d;
  proxy_read_timeout 1d;

  #서버 이름
        server_name i9a702.p.ssafy.io www.i9a702.p.ssafy.io; # 사용할 url로 변경

  #2가지 키가 발급 (openvidu 설치 시 발급받은 인증서 사용)
  ssl_certificate /opt/openvidu/certificates/live/i9a702.p.ssafy.io/fullchain.pem;
  ssl_certificate_key /opt/openvidu/certificates/live/i9a702.p.ssafy.io/privkey.pem;

  location / {   # 프론트엔드
          proxy_pass http://localhost:3000;
    proxy_redirect off;
    proxy_http_version 1.1;
            proxy_set_header Upgrade $http_upgrade;
            proxy_set_header Connection "upgrade";
        }
  location /ws {
          proxy_pass http://localhost:3000;
          proxy_http_version 1.1;
          proxy_set_header Upgrade $http_upgrade;
          proxy_set_header Connection "upgrade";
          proxy_set_header Host $host;
      }
        location /api { # 백엔드
    proxy_pass http://localhost:8081;
    proxy_redirect off;
  }

  location /ws-stomp { # 백엔드 웹소켓
    proxy_pass http://localhost:8081/ws-stomp/;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header HOST $http_host;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "Upgrade";
  }

  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' '*';
}
```

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 생김
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
#결국 /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","./Sarkem-0.0.1-SNAPSHOT.jar"]
```

Front

```
FROM node:16

WORKDIR /

COPY package.json .

RUN npm install

COPY . .

ENV WDS_SOCKET_PORT 0

EXPOSE 3000

CMD ["npm", "start"]
```

Spring boot : Application.yml

```
server.port: 8081
server.ssl.enabled: false

# openvidu setting
# ${openvidu의 .env에서 설정한 URL + 포트번호}
OPENVIDU_URL: "https://localhost:8443/"
# ${openvidu의 .env에서 설정한 SECRET}
OPENVIDU_SECRET: "sarkemsarkem" # (jenkins script 사용 시 주석처리)

# sarkem
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://localhost:3306/sarkem?useSSL=false&useUnicode=true&serverTimezone=Asia/Seoul
    username: ${username} # sarkem
    password: ${password} # sarkem
  data:
    redis:
      host: localhost
      port: 4000
  jpa:
    show-sql: true
    hibernate:
      ddl-auto: update
    properties:
      hibernate:
        format_sql : true

logging:
  level:
    com.a702.sarkem: debug
```

참고: Jenkins

💡 **jenkins를 통해 CI/CD를 구축한다면 아래와 같은 스크립트를 사용합니다.**

OPENVIDU_SECRET=”openvidu의 .env에서 설정한 패스워드”

```
BE_CONTAINER="sarkem_back_container"
BE_IMAGE="sarkem_back_image"
FE_CONTAINER="sarkem_front_container"
FE_IMAGE="sarkem_front_image"

docker build -t $BE_IMAGE /var/lib/jenkins/workspace/sarkem/Sarkem/backend
if [ "$(docker ps -a -q -f name=$BE_CONTAINER)" ]; then
  echo "hi3"
    docker stop $BE_CONTAINER
    docker rm $BE_CONTAINER
fi
docker run -d --network="host" -p 8081:8081 -e OPENVIDU_SECRET=sarkemsarkem --name $BE_CONTAINER $BE_IMAGE

docker build -t $FE_IMAGE /var/lib/jenkins/workspace/sarkem/Sarkem/front_ex
if [ "$(docker ps -a -q -f name=$FE_CONTAINER)" ]; then
  echo "hi2"
    docker stop $FE_CONTAINER
    docker rm $FE_CONTAINER
fi
docker run -d --network="host" --name $FE_CONTAINER $FE_IMAGE
docker image prune -f
```