A307 : PLEX

SSAFY 서울캠퍼스 7기

공통프로젝트 (2022.07.11 ~ 2022.08.19)

포팅 매뉴얼

담당 컨설턴트 : 박세영

김용환(팀장), 김지훈, 박원창, 정예원, 한하평, 홍인호

<< 목차 >>

1. 프로젝트 기술 스택

가. 이슈관리 : Jira

나. 형상관리 : Gitlab

다. 커뮤니케이션 : Mattermost, Notion, Webex

라. 개발 환경

1) OS : Windows 10

2) IDE : IntelliJ (2022.1.3), Visual Studio Code (1.69.0)

3) Database : MySQL (5.7.35), Redis (7.0.4)

4) Server : AWS EC2 Ubuntu(20.04 LTS), nginx(1.18.0), Docker(20.10.17)

마. 기술 스택

1) Backend : Java (OpenJDK 1.8.0.332), JPA(5.0.0), spring boot(2.4.5), spring security(2.4.5), OpenVidu(2.22.0)

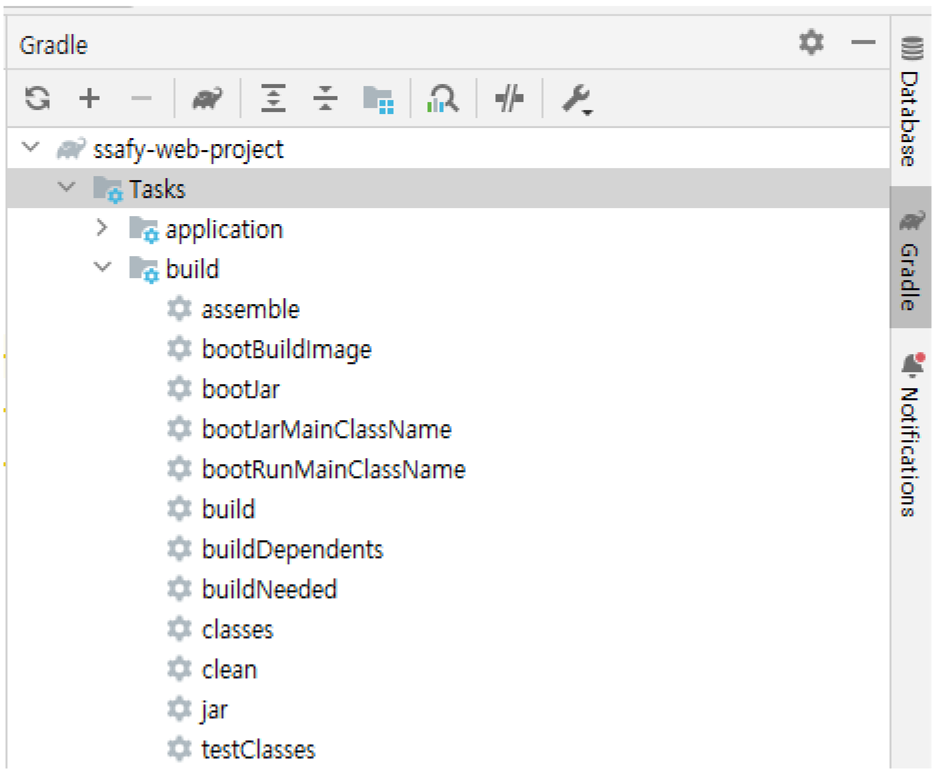
2) Frontend : Vue 2, Vuex, Node.js (v16.16.0), Vuetify, Teachable Machine, Phaser,

2. 빌드 상세내용

가. 백엔드

1. 인텔리제이의 Gradle을 사용한다.
2. 오른쪽에 있는 Gradle 탭을 누른다.

3. build 폴더로 이동
4. bootJar를 누르면 설정한 경로(backend/build/libs)에 jar 파일이 만들어진다.



나. 프론트

1. node_modules를 위한 기본 install

   - npm install

2. Nginx 배포를 위한 배포파일 빌드

   - npm run build

   - 빌드 파일 생성 경로(backend/src/main/resources)에 dist 폴더 생성

다. 서버(도커 설치)

1. apt 패키지 업데이트 및 HTTPS 활성화

```
sudo apt-get update

sudo apt-get install \

    ca-certificates \

    curl \
    gnupg \
    lsb-release
```

2. 도커에 GPG 키 추가

```
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

3. 레포지토리 세팅

```
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
```

4. 도커 설치

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

5. 도커 컴포즈 설치

```
sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
docker-compose -version
```

라. 서버(MySQL 설치)

1. 도커 허브에 있는 이미지 가져오기

```
docker pull mysql:latest
```

2. MySQL 컨테이너 생성 및 실행

```
docker run --name mysql-container MYSQL_ROOT_PASSWORD=<password> -v /opt/lib/mysql:/var/lib/mysql -d -p 3306:3306 mysql:latest
```

3. 컨테이너 상태 확인

```
docker ps -a
```

3. 배포 특이사항

1. 현재 구동 중인 nginx 확인

   ```
   ps -ef | grep nginx
   ```

2. 현재 사용중인 포트 확인

   ```
   netstat -ano
   ```

3. Nginx 재시작

   ```
   service nginx restart
   ```

4. 구동 중인 프로세스 종료

   ```
   kill -9 <PID>
   ```

5. 도커로 컨테이너 실행

   ```
   docker run <image>
   ```

6. 도커 컨테이너 상태 확인

   ```
   docker ps
   ```

7. 도커 허브에서 이미지 받아오기

   ```
   docker pull <image>:<version>
   ```

8. Dockerfile로부터 이미지 만들기

   ```
    touch Dockerfile
   // Dockerfile에 내용 넣기
   docker build -t <image-name> <[절대|상대]경로>
   ```

9. docker compose 를 사용한 컨테이너 배포

   ```
   docker compsoe up --build -d
   ```

10. docker compose 를 사용한 컨테이너 종료

    ```
    docker compose down
    ```

11. 도커를 통한 쉘 접속

    ```
    docker exec -it <container-name or id> /bin/bash
    ```

4. DB 계정

- username: root
- password: tkvlclfxla7

5. 프로퍼티 정의

가. Nginx Default 값

1. 정적 파일을 올리기 위한 nginx

```
server{
        server_name localhost;

        location / {
                root    /usr/share/nginx/html/dist;
                index   index.html;

        }

        listen [::]:443 ssl ipv6only=on; # managed by Certbot
        listen 443 ssl; # managed by Certbot
        ssl_certificate /etc/letsencrypt/live/i7a307.p.ssafy.io/fullchain.pem; # managed
by Certbot
        ssl_certificate_key /etc/letsencrypt/live/i7a307.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


}
```

2. 프록시 기능을 위한 nginx

```
server{
                        root /var/www/html;

        # Add index.php to the list if you are using PHP
        index index.html index.htm index.nginx-debian.html;
    server_name i7a307.p.ssafy.io; # managed by Certbot


        location / {
                proxy_pass https://i7a307.p.ssafy.io:3000;

                #proxy_pass https://host.docker.internal:3000;
                # First attempt to serve request as file, then
                # as directory, then fall back to displaying a 404.
                #try_files $uri $uri/ =404;
        }

        location /api {
                proxy_pass https://i7a307.p.ssafy.io:5000/api;
                #proxy_pass https://host.docker.internal:5000/api;
                proxy_http_version 1.1;
                proxy_redirect off;
                charset utf-8;

                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;
                proxy_set_header Upgrade $http_upgrade;
                proxy_set_header Connection "upgrade";

        }
}
```

나. 도커 파일

1. 프론트 Dockerfile

```
FROM nginx
COPY dist /usr/share/nginx/html
COPY nginxconf/default.conf /etc/nginx/conf.d/default.conf
EXPOSE 443
```

2. 백엔드 Dockerfile

```
FROM openjdk:11
COPY . /usr/src/myapp
WORKDIR /usr/src/myapp
EXPOSE 8080
CMD ["java","-jar","backend.jar"]
```

3. docker-compose.yml

```
version: "3"
services:
  backend:
    build:
      context: ./backend
      dockerfile: Dockerfile
    links:
      - "backredis"
    ports:
      - "5000:8080"
    volumes:
      - ./backend:/usr/src/myapp

  backredis:
    image: redis

  frontend:
    build:
      context: ./frontend
      dockerfile: Dockerfile
    ports:
      - "3000:443"
    volumes:
      - ./frontend/volfront:/usr/share/nginx/html
      - /etc/letsencrypt:/etc/letsencrypt
      - /etc/ssl:/etc/ssl
```

다. application.properties

```
#it will be set build date by gradle. if this value is @build.date@, front-end is development mode
build.date=@build.date@
server.port=8080
#server.address=localhost
server.servlet.contextPath=/
# Charset of HTTP requests and responses. Added to the "Content-Type" header if not set explicitly.
server.servlet.encoding.charset=UTF-8
# Enable http encoding support.
server.servlet.encoding.enabled=true
# Force the encoding to the configured charset on HTTP requests and responses.
server.servlet.encoding.force=true

# for SPA
spring.resources.static-locations=classpath:/dist/
spa.default-file=/dist/index.html
spring.mvc.throw-exception-if-no-handler-found=true
spring.resources.add-mappings=false

# Swagger
springfox.documentation.swagger.use-model-v3=false

#database
```

```
spring.jpa.hibernate.naming.implicit-strategy=org.springframework.boot.orm.jpa.hibernate.S
pringImplicitNamingStrategy
spring.jpa.hibernate.naming.physical-strategy=org.springframework.boot.orm.jpa.hibernate.
SpringPhysicalNamingStrategy
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL57Dialect
spring.data.web.pageable.one-indexed-parameters=true
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true

spring.datasource.url=jdbc:mysql://i7a307.p.ssafy.io:3306/ssafy_web_db?useUnicode=true&
characterEncoding=utf8&serverTimezone=Asia/Seoul&zeroDateTimeBehavior=convertToNul
l&rewriteBatchedStatements=true
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.hikari.username=root
spring.datasource.hikari.password=tkvlclfxla7
#spring.datasource.url=jdbc:mysql://localhost:3306/ssafy_web_db?useUnicode=true&charac
terEncoding=utf8&serverTimezone=Asia/Seoul&zeroDateTimeBehavior=convertToNull&rewr
iteBatchedStatements=true
#spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
#spring.datasource.hikari.username=ssafy
#spring.datasource.hikari.password=ssafy



# redis
spring.redis.host=backredis
spring.redis.port=6379

# jwt
jwt.secret=dyAeHubOOc8KaOfYB6XEQoEj1QzRlVgtjNL8PYs1A1tymZvvqkcEU7L1imkKHe
Da
jwt.refreshsecret=alsdjflkajsdlkfjasljLKJLKSDJFsdfjsldkjf
# unit is ms. 15 * 24 * 60 * 60 * 1000 = 15days
#jwt.expiration=1800000
jwt.expiration=1296000000
jwt.refreshexpiration=1296000000

#logging
logging.file.name=./ssafy-web.log
logging.level.root=INFO
logging.level.com.samsung.security=DEBUG
logging.level.org.springframework.web=DEBUG
logging.level.org.apache.tiles=INFO
logging.level.org.sringframework.boot=DEBUG
logging.level.org.sringframework.security=DEBUG

spring.devtools.livereload.enabled=true
```

```
#gzip compression
server.compression.enabled=true
server.compression.mime-types=application/json,application/xml,text/html,text/xml,text/plain,
application/javascript,text/css

#for health check
management.servlet.context-path=/manage
management.health.db.enabled=true
management.health.default.enabled=true
management.health.diskspace.enabled=true

server.ssl.enabled=true
server.ssl.key-store=classpath:keystore.p12
server.ssl.key-store-password=squid
server.ssl.key-store-type=PKCS12
server.ssl.key-alias=tomcat

#server.ssl.enabled: true
#server.ssl.key-store: classpath:openvidu-selfsigned.jks
#server.ssl.key-store-password: openvidu
#server.ssl.key-store-type: JKS
#server.ssl.key-alias: openvidu-selfsigned

openvidu.url=https://i7A307.p.ssafy.io:4443
openvidu.secret=SQUID
#openvidu.url=https://localhost:4443
#openvidu.secret=MY_SECRET

debug=true
```