빌드 및 배포 정리 문서

1. 사용한 JVM, 웹서버, WAS 제품 등의 종류와 설정값, 버전(IDE 버전 포함) 기재

종류

웹서버 - Nginx

WAS - Tomcat

버전

Backend

Java - OpenJDK 11.0.14.1

spring-boot - 2.6.3

spring 내장 tomcat - 2.6.3

gradle - 7.4.1

Frontend

node - 14.18.1

npm - 6.14.15

nginx - 1.18.0(Ubuntu)

IDE

intelliJ IDEA - 2021.3.1

Visual Studio Code - 1.66.0

2. 빌드 시 사용되는 환경변수 등의 주요 내용 상세 기재

Frontend

frontend-vue/`.env` 파일 추가

```
APP_PORT=3001
CHOKIDAR_USEPOLLING=true
SKIP_PREFLIGHT_CHECK=true

VUE_APP_ENV=production
VUE_APP_BACKEND_HOST_URL=http://localhost:3000/
VUE_APP_ETHEREUM_RPC_URL=http://20.196.209.2:8545/

VUE_APP_SALE_FACTORY_CA=0xDAc66969f7031a92F48BACd1A8907bbf78bC2253
VUE_APP_NFT_CA=0x5F2D9F51c1453C708700e46FD400A3a6A9b7A5F0
VUE_APP_ERC20_CA=0xd7D2b2859e8485eF94dA0B11c421c77c48311981

VUE_APP_ADMIN_ADDRESS=0x4135f8fD42c98cAb53883863b6b80A7AA806e0E9
VUE_APP_ADMIN_PRIVKEY=0xe20c61798cad4c6edb27c902e3592d9c0f92983239fef77f334a3701e7bad767

VUE_APP_API_BASE_URL=http://j6b209.p.ssafy.io/api
```

AWS에서 빌드 과정

1. Git 설치 및 연동(**/home/ubuntu 경로에서**)

```
apt-get install git
git clone https://lab.ssafy.com/s06-blockchain-nft-sub2/S06P22B209.git
# 로그인
```

2. build

```
cd frontend-vue 폴더로
npm install
npm run build
```

Backend

spring-backend/src/main/resources/`application.yml` 파일 추가

```
spring:
  datasource:
    url: jdbc:mysql://j6b209.p.ssafy.io:5000/handtohand
```

```
    username: ssafy
    password: ssafy
    driver-class-name: com.mysql.cj.jdbc.Driver
  mvc:
    pathmatch:
      matching-strategy: ant_path_matcher
jpa:
  generate-ddl: false
  open-in-view: false
  hibernate:
    ddl-auto: validate
  properties:
    hibernate:
      format_sql: true
logging.level:
  org.hibernate.SQL: debug
  org.hibernate.type: trace

cloud:
  aws:
    credentials:
      access-key: AKIAXB5B4COU24P663OM
      secret-key: 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
    s3:
      bucket: handtohand
    region:
        static: ap-northeast-2
```

AWS에서 빌드 과정

1. Java 설치

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

2. build

```
cd backend-spring 폴더로
sudo chmod 777 ./gradlew  # 폴더나 파일에 권한부여(777 - 모든 권한)
sudo ./gradlew build      # build
```

3. 배포 시 특이사항 기재

Frontend

방법 - AWS에서 nodejs로 빌드 후, NGINX로 실행

빌드전

1. NGINX 설치

```
#root계정이나 sudo 명령어 사용
apt-get update
apt-get upgrade
apt-get install nginx
```

2. NGINX 환경설정

```
cd /etc/nginx/sites-available
sudo vi default
```

3. default 파일 수정

```
server {
  listen 80 default_server;
  listen [::]:80 default_server;

  root /home/ubuntu/S06P22B209/frontend-vue/dist;  # 실행시킬 dist 폴더 위치

  index index.html index.htm index.nginx-debian.html;

  server_name _;
# 80/ 으로 접근 시 반응
  location / {
    try_files $uri $uri/ =404;
  }

# reverse proxy 설정 - 경로 재설정해줌(nginx 제공 기능 중 하나)
# 실제 8081 포트에 접근하지 않아도 8081 포트에 접근한 것과 동일한 효과
  location /api/ {  # 서버에 /api로 시작하는 path로 접근할 시 8081으로 돌려줌
    proxy_pass http://j6b209.p.ssafy.io:8081/;
    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 $schem;
    proxy_set_header X-NginX-Proxy true;
}
```

빌드후

1. NGINX로 실행

```
#root계정이나 sudo 명령어 사용
service nginx start   # 시작
service nginx restart # 재시작
service nginx status  # 상태확인
service nginx stop    # 중지
```

Backend

방법 - AWS에서 Gradle로 빌드 후, Docker에서 Spring 내장 tomcat로 실행

1. Spring-backend 폴더 내에 Dockerfile 파일 생성

   Dockerfile

   ```
   FROM openjdk:11-jdk
   EXPOSE 8081
   ARG JAR_FILE=build/libs/handtohand-0.0.1-SNAPSHOT.jar  # jar파일 경로
   COPY ${JAR_FILE} app.jar    # docker 내의 app.jar로 복사
   ENTRYPOINT ["java","-jar","/app.jar"]
   ```

2. Docker image 생성(빌드)

   ```
   # Dockerfile 있는 곳에서
   docker build . -t {설정할 이미지 이름}  # -t : 태그
   ```

3. Docker Container 실행

   ```
   docker run --name back -d -p 8081:8081 {설정한 이미지 이름}
   ```

4. DB 접속 정보 등 프로젝트(ERD)에 활용되는 주요 계정 및 프로퍼티가 정의된 파일 목록

DB명

handtohand

관리자 계정

사용자 : handtohand

암호 : handtohand

해당 db에 대한 권한만 가진 사용자 계정

사용자 : ssafy

암호 : ssafy

프로퍼티

--character-set-server=utf8mb4

--collation-server=utf8mb4_unicode_ci