

포팅 매뉴얼



D207: Just Move It

삼성SW청년아카데미 구미 캠퍼스 7기

공통 프로젝트

담당 컨설턴트: 유상진

팀원: 최영진(팀장), 강동관, 김효선, 정성우, 정종일, 조경수

목차

1. 프로젝트 개요	1
2. 프로젝트 기술 스택	1
3. 빌드 상세내용	2
4. 배포 특이사항	2
5. 서버 환경 세팅	3
6. 외부 서비스	8

=====

1. 프로젝트 개요

영화관에서 일할 당시 키오스크를 사용하기 어려워하는 분들이 많았습니다. 또한 코로나 팬데믹 이후 직접적인 키오스크 조작에 있어 거부감을 느끼는 분들도 많았습니다.

이에 조금 더 편리한 키오스크를 제작해 사용자들에게 보다 나은 경험을 주고자 Just Move It 을 만들어보게 되었습니다.

2. 프로젝트 기술 스택

가. 사용툴

- A. 이슈 관리: Jira
- B. 형상 관리: GitLab
- C. 커뮤니케이션: MatterMost, Notion
- D. 목업 관리: Figma
- E. 기타: miro (플로우 차트 생성)

나. 개발 환경

- A. OS: Window 10
- B. Server: AWS, NginX
- C. FE: HTML5, CSS, ES6, React 18.2.0

- D. BE: Spring Boot, JPA
- E. Android: Java, Retrofit2
- F. Database: MySQL
- G. AI: Teachable Machine, TensorFlow
- H. IoT: Rasbian

3. 빌드 상세내용

가. BE

```
C:\WS07P12D207\WBEWCommonPJT> ./gradlew build
```

나. FE

node_modules 를 위한 기본 install. 버전 문제로 강제 설치를 진행해야 합니다.

```
C:\WS07P12D207\WFEWnewPjt> npm i --force
```

설치가 완료되면 프로젝트를 시작합니다.



```
C:\WS07P12D207\WFEWnewPjt> npm run start
```

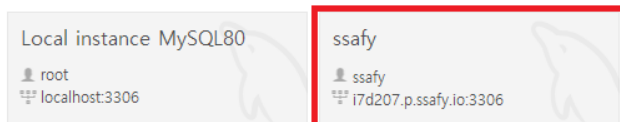
4. 배포 특이사항

가. DB 계정

- 1) MySQL, Workbench 추가하기

[Browse Documentation >](#)

MySQL Connections  



2) EC2 계정 정보 등록

Setup New Connection

Connection Name: Type a name for the connection

Connection Method: Method to use to connect to the RDBMS

Parameters SSL Advanced

Hostname: Port: Name or IP address of the server host - and TCP/IP port.

Username: Name of the user to connect with.

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

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

username: ssafy, password: 1234

5. 서버 환경 세팅

가. EC2 세팅

- 1) AWS EC2 접속을 위한 MobaXterm 설치
- 2) 싸피에서 지급받은 pem 키를 이용하여 세션 등록
- 3) EC2 서버에 Java 8, mySql workbench 8.0.29, nodeJs 18.7.0 설치
- 4) https 인증을 위한 certbot letsencrypt 에서 ssl 인증
- 5) nginx 설치 후 /etc/nginx/sites-available/default 파일을 수정

```
server{
#HTTP 기본 경로인 80 포트로 접속
    listen 80 default_server;
    server_name i7d207.p.ssafy.io;
    if ($host = i7d207.p.ssafy.io) {
```

```
        #Nginx 로 들어온 http 주소를 https 주소로 변환시켜준다
        return 301 https://$host$request_uri;
    }
}
```

```
# managed by Certbot
```

```
}
```

```
server{
```

```
#HTTPS 기본 포트인 443 포트로 접속하여 해당 경로에 맞게 리버스 프록싱 시켜준다
```

```
    listen 443 ssl;
```

```
    listen [::]:443;
```

```
    server_name i7d207.p.ssafy.io;
```

```
    proxy_hide_header Access-Control-Allow-Origin;
```

```
    add_header 'Access-Control-Allow-Origin' '*';
```

#웹 소켓 경로

```
    location /ws/socket {
```

```
        proxy_pass <http://localhost:8081>;
```

```
        proxy_set_header Host $host;
```

```
        proxy_http_version 1.1;
```

```
        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 Origin "";
```

```
        proxy_set_header Connection "Upgrade";
```

```
    }
```

#REACT 경로 설정

```
    location / {
```

```
        proxy_pass <http://localhost:3000>;
```

```
        proxy_set_header Host $host;
```

```
        proxy_http_version 1.1;
```

```
        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";
    }

#Swagger 경로
location ~ ^/(swagger|webjars|configuration|swagger-resources|v2|csrf) {
    proxy_pass <http://localhost:8081>;

    proxy_set_header Host $host;
    proxy_http_version 1.1;
    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";
}

#스프링 부트 (컨트롤러) 경로
location /api {
    proxy_pass <http://localhost:8081>;

    proxy_set_header Host $host;
    proxy_http_version 1.1;
    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";
}

ssl_certificate /etc/letsencrypt/live/i7d207.p.ssafy.io/fullchain.pem;
ssl_certificate_key /etc/letsencrypt/live/i7d207.p.ssafy.io/privkey.pem;
}

```

나. 배포 환경 세팅

1) Docker-compose 설정

```

version: '3'
services:
  mysql:
    image: mysql:latest

```

container_name: mysql_container

ports:

- 3306:3306 # HOST:CONTAINER

environment:

MYSQL_ROOT_PASSWORD: Ssafyd207

MYSQL_PASSWORD: Ssafyd207

MYSQL_USER: ssafy

MYSQL_PASSWORD: Ssafyd207

command:

- --character-set-server=utf8mb4
- --collation-server=utf8mb4_unicode_ci

volumes:

- /etc/mysql/

networks:

- backend-network

application:

build:

context: ./app

dockerfile: Dockerfile

image: springboot

ports:

- "8081:8081"

depends_on:

- mysql

container_name: app_container

environment:

SPRING_DATASOURCE_URL:

jdbc:mysql://mysql:3306/ssafy?useSSL=false&serverTimezone=UTC&useLegacyDatetimeCode=false

SPRING_DATASOURCE_USERNAME: ssafy

SPRING_DATASOURCE_PASSWORD: Ssafyd207

networks:

- backend-network

react:

image: react

ports:

- "3000:3000"

container_name: node_container

volumes:

```

        - /S07P12D207/FE/newPjt/:/var/www
networks:
    - frontend-network
build:
    context: /S07P12D207/FE/newPjt/
    dockerfile: Dockerfile

networks:
    backend-network:
    frontend-network:

```

① react docker file

경로: / S07P12D207/FE/newPjt/Dockerfile

```

FROM node:18.2.0-alpine
WORKDIR /app
COPY package*.json ./
COPY . ./
RUN npm install -f
EXPOSE 3000
CMD ["npm", "run", "start"]

```

② spring boot docker file

빌드된 자바 파일을 cp 명령어를 이용하여 /etc/nginx/app 폴더 안에 넣어주었다.

```

FROM openjdk:8-jdk-alpine
ARG JAR_FILE=CommonPJT-0.0.1-SNAPSHOT.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java","-jar","/app.jar"]

```

③ MySQL Docker

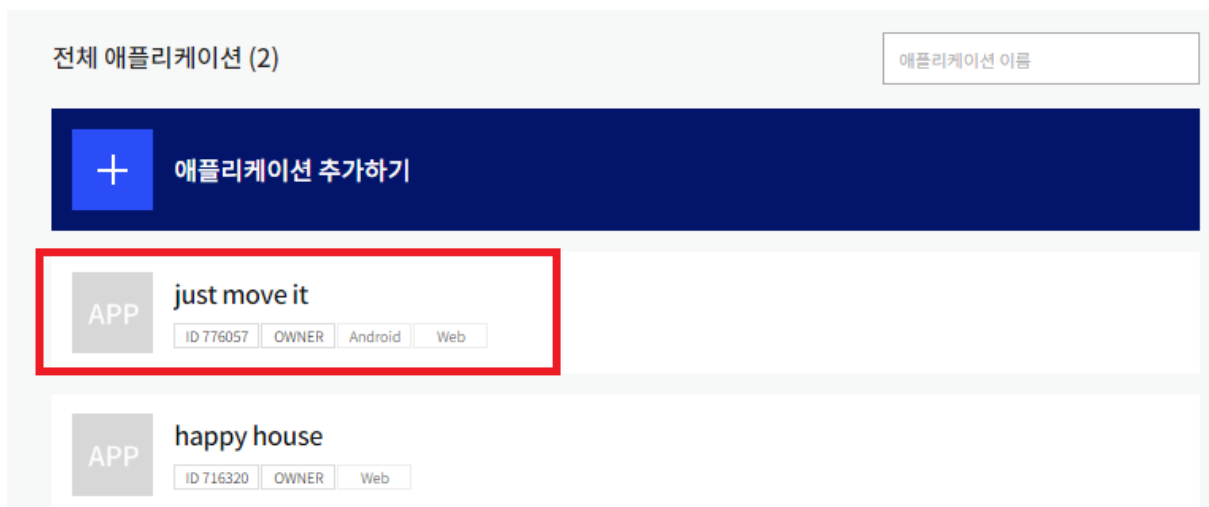
- 1) mySql 도커 이미지 다운로드
- 2) mySql 도커 컨테이너 생성 후 아이디 생성
- 3) docker-compose up -d 를 통하여 이미지를 컨테이너에 올림

6. 외부 서비스

가. 카카오 로그인

안드로이드 앱 서비스의 사용자 로그인 구현, 사용자의 성별과 나이 수집을 목적으로 사용하였습니다.

1) 어플리케이션 추가



2) 플랫폼 등록

3) 개인정보 동의 항목

개인정보

항목 이름	ID	상태
닉네임	profile_nickname	● 필수 동의 설정
프로필 사진	profile_image	● 필수 동의 설정
카카오계정(이메일)	account_email	● 선택 동의 설정
성별	gender	● 선택 동의 [수집] 설정
연령대	age_range	● 선택 동의 [수집] 설정

나. 카카오 페이

앱 서비스 내 구매 기능을 구현하기 위해 사용하였습니다. 그러나 실제 결제를 위한 제휴 신청은 하지 않았기 때문에 테스트 결제로 남겨두었습니다.

1) 결제 준비

① Request Url

```
POST /v1/payment/ready HTTP/1.1
Host: kapi.kakao.com
Authorization: KakaoAK ${APP_ADMIN_KEY}
Content-type: application/x-www-form-urlencoded;charset=utf-8
```

② Request Parameter

```
curl -v -X POST "<https://kapi.kakao.com/v1/payment/ready>" \WW
-H "Authorization: KakaoAK ${APP_ADMIN_KEY}" \WW
--data-urlencode "cid=TCOONETIME" \WW
--data-urlencode "partner_order_id=partner_order_id" \WW
--data-urlencode "partner_user_id=partner_user_id" \WW
--data-urlencode "item_name=MOVIE_TITLE" \WW
--data-urlencode "quantity=1" \WW
--data-urlencode "total_amount=TICKET_PRICE" \WW
--data-urlencode "vat_amount=200" \WW
--data-urlencode "tax_free_amount=0" \WW
--data-urlencode "approval_url=https://developers.kakao.com/success" \WW
--data-urlencode "fail_url=https://developers.kakao.com/fail" \WW
--data-urlencode "cancel_url=https://developers.kakao.com/cancel"
```

③ Response

```
HTTP/1.1 200 OK
Content-type: application/json;charset=UTF-8
{
  "tid": "T1234567890123456789",
  "next_redirect_app_url": "<https://mockup-pg-web.kakao.com/v1/xxxxxxxxx/alInfo>",
  "next_redirect_mobile_url": "<https://mockup-pg-web.kakao.com/v1/xxxxxxxxx/mlInfo>",
  "next_redirect_pc_url": "<https://mockup-pg-web.kakao.com/v1/xxxxxxxxx/info>",
  "android_app_scheme": "kakaotalk://kakaopay/pg?url=https://mockup-pg-
web.kakao.com/v1/xxxxxxxxx/order",
```

```
"ios_app_scheme": "kakaotalk://kakaopay/pg?url=https://mockup-pg-  
web.kakao.com/v1/xxxxxxxxxx/order",  
"created_at": "2016-11-15T21:18:22"  
}
```

2) 결제 요청

결제 준비 API 응답으로 받은 **next_redirect_mobile_url** 을 웹뷰로 띄워 진행.

3) 결제 승인

① Request URL

```
POST /v1/payment/approve HTTP/1.1  
Host: [kapi.kakao.com](http://kapi.kakao.com/)  
Authorization: KakaoAK ${APP_ADMIN_KEY}  
Content-type: application/x-www-form-urlencoded;charset=utf-8
```

② Request Param

```
curl -v -X POST  
"[<https://kapi.kakao.com/v1/payment/approve>](<https://kapi.kakao.com/v1/payment/approve>)"  
₩  
-H "Authorization: KakaoAK ${APP_ADMIN_KEY}" ₩  
--data-urlencode "cid=TC0ONETIME" ₩  
--data-urlencode "tid=T1234567890123456789" ₩  
--data-urlencode "partner_order_id=partner_order_id" ₩  
--data-urlencode "partner_user_id=partner_user_id" ₩  
--data-urlencode "pg_token=xxxxxxxxxxxxxxxxxxxxxx"
```

③ Response

```
HTTP/1.1 200 OK  
Content-type: application/json;charset=UTF-8  
{  
  "aid": "A5678901234567890123",  
  "tid": "T1234567890123456789",  
  "cid": "TC0ONETIME",  
  "partner_order_id": "partner_order_id",  
  "partner_user_id": "partner_user_id",
```

```
"payment_method_type": "MONEY",  
"item_name": "MOVIE_TITLE",  
"quantity": 1,  
"amount": {  
  "total": 2200,  
  "tax_free": 0,  
  "vat": 200,  
  "point": 0,  
  "discount": 0,  
  "green_deposit": 0  
},  
"created_at": "2016-11-15T21:18:22",  
"approved_at": "2016-11-15T21:20:47"  
}
```