



## C201: 취미 공유 SNS - 모꼬지

삼성**SW**청년아카데미 광주캠퍼스 7기

공동프로젝트 (22.07.05 - 22.08.19)

### 포팅 매뉴얼

담당 컨설턴트: 김성재

조성민(팀장), 박세호, 박상현, 임윤희, 정호진, 허재영

## 목차

<b>0. 모꼬지란?</b>	<b>4</b>
<b>1. 프로젝트 기술스택</b>	<b>4</b>
가. 이슈관리	4
나. 형상관리	4
다. 커뮤니케이션	4
라. 개발 환경	5
백엔드 기술스택	5
프론트엔드 기술스택	5
<b>2. 프론트엔드 빌드 방법</b>	<b>6</b>
가. 빌드 파일 생성	6
나. 서버 실행	6
다. Dockerfile	7
<b>3. 백엔드 빌드 방법</b>	<b>8</b>
가. AWS EC2 DB 세팅	8
나. AWS EC2 Docker 세팅	9
다. Docker Jenkins 세팅	9
다. Nginx Default 값	11
<b>4. EC2 세팅 및 Jenkins</b>	<b>13</b>
가. Jenkins Git 연동	14

나. Jenkins Build	15
<b>5. 외부 서비스 - SNS 로그인</b>	<b>18</b>
가. 카카오	18
나. 구글	19
다. 네이버	21
<b>6. 외부서비스 - 파이어베이스</b>	<b>23</b>
가. 파이어베이스를 활용한 이유	23
나. 애플리케이션 추가	23
다. React 프로젝트에서 Firebase 사용법	23
다. 사진 저장 및 활용	24

## 0. 모꼬지란?

취미 공유 SNS인 모꼬지는 “여러 사람이 함께 즐기다”라는 순 우리말로 많은 사람들이 다양한 취미를 과정을 밟으며 쉽게 즐길 수 있도록 돕는 웹 서비스입니다. 사용자는 모꼬지에서 여러 취미를 접할 수 있고 성공 스토리를 쌓을 수 있습니다. 챌린지와 각 챌린지를 구성하는 스테이지를 통해 차근차근 과정을 밟아나갈 수 있습니다. 사용자는 챌린지를 등록하여 타 유저들과 취미를 즐기는 방법도 공유할 수 있습니다. 스테이지에 성공하면 포스팅을 남길 수 있고 타 포스팅에 댓글을 달며 서로의 성공 스토리를 공유합니다.

## 1. 프로젝트 기술스택

가. 이슈관리

- Jira

나. 형상관리

- Gitlab

다. 커뮤니케이션

- Mattermost
- Notion

## 라. 개발 환경

- OS: Windows 10
- 사용 **IDE**:
  - IntelliJ IDEA 2022.1.3
  - Visual Studio Code : 1.70.2v
  - UI/UX: Figma
- 백엔드 기술스택
  - Springboot : 2.6.9
  - MariaDB : mariadb Ver 15.1 Distrib 10.3.34-MariaDB, for debian-linux-gnu (x86\_64) using readline 5.2
  - AWS : ubuntu 20.04.4 LTS
  - Jenkins : 2.346.2
  - Docker : 20.10.17
  - Openjdk : 11.0.16
  - spring: gradle
  - nginx : nginx/1.18.0 (Ubuntu)
- 프론트엔드 기술스택
  - node.js : 16.15.0v 64bit (LST 버전 사용)
  - npm : 8.5.5v
  - react : 18.2.0v
  - react-router-dom : 6.3.0v
  - react-redux : 8.0.2v

- redux Toolkit : 1.8.3v
- TypeScript : 4.7.4v
- Sass : 1.54.4v

## 2. 프론트엔드 빌드 방법

### 가. 빌드 파일 생성

- 실행 환경을 조성하기 위하여 `npm run build`를 통해 빌드 파일을 생성합니다.

```
npm run build
```

### 나. 서버 실행

- 해당 명령어는 'serve라는 웹 서버를 다운받아 실행시킬 때 build라는 디렉토리를 document root'로 하겠다는 뜻으로, 한 번 만 실행시킬 웹 서버를 실행시키는 명령어입니다.

```
npx serve -s build
```

## 다. Dockerfile

```
FROM node:16.16.0 as builder

# 작업 폴더를 만들고 npm 설치

RUN mkdir /usr/src/app

WORKDIR /usr/src/app

ENV PATH /usr/src/app/node_modules/.bin:$PATH

COPY package.json /usr/src/app/package.json

RUN npm install --silent

RUN npm install react-scripts@5.0.1 -g --silent


# 소스를 작업폴더로 복사하고 앱 실행

COPY . /usr/src/app

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

### 3. 백엔드 빌드 방법

#### 가. AWS EC2 DB 세팅

##### 1) MariaDB 패키지 저장소 추가

```
sudo apt-get install apt-transport-https curl
sudo curl -o /etc/apt/trusted.gpg.d/mariadb_release_signing_key.asc
'https://mariadb.org/mariadb_release_signing_key.asc'
sudo sh -c "echo 'deb https://tw1.mirror.blendbyte.net/mariadb/repo/10.3/ubuntu focal
main'
```

##### 2) MariaDB 설치

```
sudo apt-get update
sudo apt-get install mariadb-server
```

##### 3) root 계정으로 사용자 생성 및 권한 부여

```
sudo mysql -u root -p
CREATE USER '아이디'@'%' IDENTIFIED BY '비밀번호';
GRANT ALL PRIVILEGES ON 데이터베이스.* TO '아이디'@'%';
FLUSH PRIVILEGES;
exit
```

##### 4) 원격 접속을 위한 설정 변경

```
sudo vi /etc/mysql/mariadb.conf.d/50-server.cnf
bind-address = 0.0.0.1 로 변경
```



5) 설정 변경 적용을 위한 재시작

```
sudo systemctl restart mysql
```

6) default값 확인

```
mysqld --print-defaults
```

나. AWS EC2 Docker 세팅

1) apt 업데이트

```
sudo apt-get update  
sudo apt-get upgrade
```

2) Docker의 Official GPG Key 및 저장소 추가

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o  
/usr/share/keyrings/docker-archive-keyring.gpg
```

3) Docker Engine 설치

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

다. Docker Jenkins 세팅

1) Docker Jenkins 이미지 설치

```
sudo docker pull jenkins:its
```

## 2) Docker Jenkins Volume 생성

```
sudo docker volume create jenkins-volume
```

## 3) Docker Jenkins Image 실행

```
sudo docker run -d \
-p 8085:8080 \
-v jenkins-volume:/var/jenkins_home \
-v /var/run/docker.sock:/var/run/docker.sock:ro \
-v /var/lib/docker/containers:/var/lib/docker/containers:ro \
--name jenkins \
--network my-network \
jenkins/jenkins:its
```

## 4) Jenkins 접속 및 기본 설정

- sudo docker logs jenkins 에서 Jenkins 비밀번호를 복사하여

sudo ufw allow 8085 # aws 방화벽 설정합니다.

```
sudo ufw allow 8085          # aws 방화벽 설정
```

- 8085 포트로 접속해 복사한 비밀번호 입력 후 설치한 다음 Admin 계정 생성합니다.

## 다. Nginx Default 값

```
server {  
  
    listen 443 ssl default_server;  
  
    listen [::]:443 ssl default_server;  
  
    ssl_certificate /etc/letsencrypt/live/i7c201.p.ssafy.io/fullchain.pem;  
    ssl_certificate_key /etc/letsencrypt/live/i7c201.p.ssafy.io/privkey.pem;  
  
    server_name i7c201.p.ssafy.io www.i7c201.p.ssafy.io;  
  
    location / {  
        proxy_pass http://localhost:3000;  
    }  
  
    location /api {  
        proxy_pass http://localhost:8080;  
        proxy_http_version 1.1;  
        proxy_set_header Upgrade $http_upgrade;  
        proxy_set_header Connection "Upgrade";  
        proxy_set_header Host $host;  
    }  
}  
  
server {  
  
    listen 443 ssl default_server;  
  
    listen [::]:443 ssl default_server;  
  
    ssl_certificate /etc/letsencrypt/live/i7c201.p.ssafy.io/fullchain.pem;  
    ssl_certificate_key /etc/letsencrypt/live/i7c201.p.ssafy.io/privkey.pem;
```

```
server_name i7c201.p.ssafy.io www.i7c201.p.ssafy.io;

location / {
    proxy_pass http://localhost:3000;
}

location /api {
    proxy_pass http://localhost:8080;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "Upgrade";
    proxy_set_header Host $host;
}
}

server {

    listen 80;

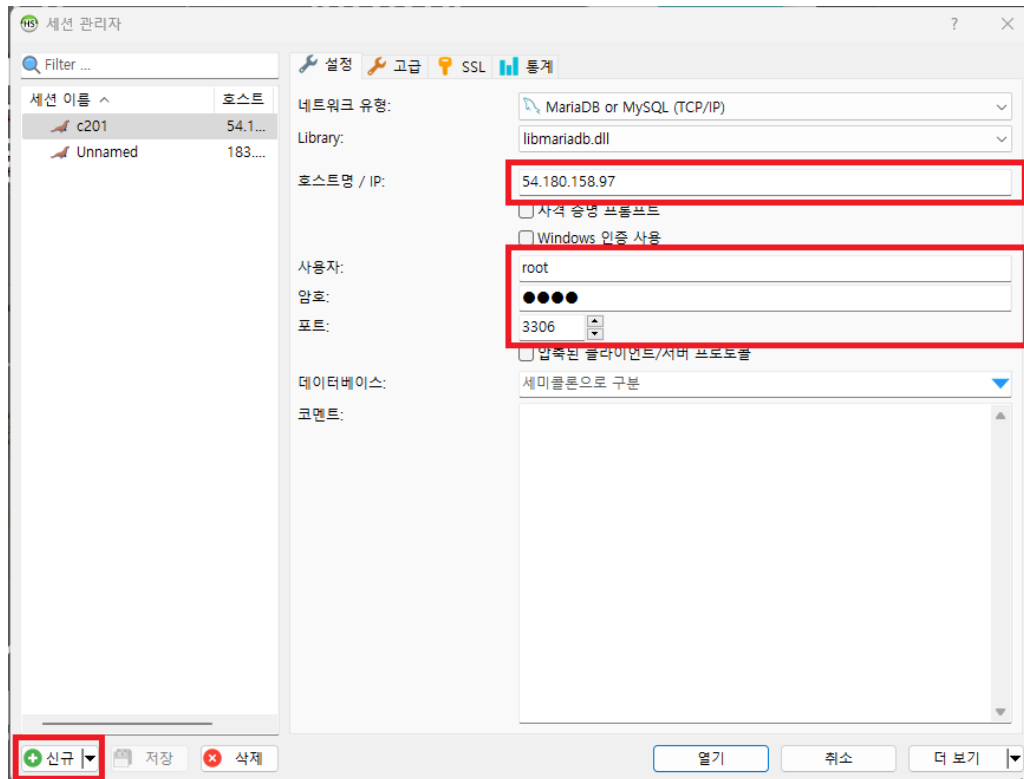
    listen [::]:80;

    server_name i7c201.p.ssafy.io;

    # // 서버네임으로 들어오면 https로 연결해줌
    return 301 https://$host$request_uri;
}
```

## 4. EC2 세팅 및 Jenkins

- 서버 ip와 설정한 db 계정과 암호로 접속해 사용합니다.



## 가. Jenkins Git 연동

### 소스 코드 관리

☐ None

☒ Git ?

#### Repositories ?

##### Repository URL ?

https://lab.ssafy.com/s07-webmobile2-sub2/moggozi.git

##### Credentials ?

nomzaxs@gmail.com/\*\*\*\*\*

+ Add

고급...

Add Repository

#### Branches to build ?

##### Branch Specifier (blank for 'any') ?

develop

Add Branch

#### Repository browser ?

(자동)

#### Additional Behaviours

Add ▾

☒ Build when a change is pushed to GitLab. GitLab webhook URL: http://54.180.158.97:8085/project/moggozi ?

#### 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

## 나. Jenkins Build

```
echo "*****Back End Build Start*****"

cd backend/restapiserver

chmod +x gradlew

./gradlew clean build

echo "*****Back End Build Done*****"


echo " Execute shell start"


cd ../../frontend


# 기존에 있는 이미지를 삭제합니다.

docker rmi -f react || true


# 도커 빌드

docker build -t react .


echo " Execute shell end"


echo " 빌드 후 조치 start"


# 기존에 있는 컨테이너를 중지합니다.

docker stop react-container || true


# 기존에 있는 컨테이너를 삭제합니다.

docker rm -f react-container || true
```

# 컨테이너를 설치하고 실행합니다.

```
docker run -d -p 3000:3000 --name react-container react
```

```
echo " 빌드 후 조치 end"
```

- 빌드 후 프론트는 도커 이미지로, 백은 jar로 실행합니다.

#### 빌드 후 조치

Send build artifacts over SSH ?

SSH Publishers

SSH Server

Name ?

aws

고급...

Transfers

Transfer Set

Source files ?

backend/restapiserver/build/libs/\* jar

Remove prefix ?

backend/restapiserver/build/libs

Remote directory ?

/

Exec command ?

sudo sh /home/ubuntu/webserver/startserver.sh

All of the transfer fields (except for Exec timeout) support substitution of [Jenkins environment variables](#)

고급...



- 빌드 후 쉘 스크립트를 이용해 백 실행합니다.

```
echo "PID Check..."

CURRENT_PID=$(ps -ef | grep java | grep restapiserver* | awk '{print $2}')

echo "Running PID:${CURRENT_PID}"

if "$CURRENT_PID" [ -z CURRENT_PID ];then

    echo "Project is not running" <200b>

else

    kill -9 $CURRENT_PID sleep 10

if

echo "Deploy Project...."

nohup    java    -jar    /home/ubuntu/webserver/restapiserver-0.0.1-SNAPSHOT.jar    >
/home/ubuntu/webserver/nohup.out 2>&1 &
```

## 5. 외부 서비스 - SNS 로그인

### 가. 카카오

- 서비스의 회원가입/로그인을 위해 카카오 API를 이용하였습니다. 간소한 절차를 통해 서비스를 이용할 수 있어 이용자의 편의성을 높였습니다.
- 카카오 인증 서비스를 이용하기 위해서는 카카오 계정을 통해 Developer로 등록하여야 합니다.
- 사용할 서비스(카카오 로그인)를 선택한 뒤, 기본 정보를 등록합니다.

- 카카오 로그인 시 사용할 도메인과, **redirect** 도메인을 등록합니다.

### Web

[삭제](#)[수정](#)

사이트 도메인

<https://i7c201.p.ssafy.io>

- 카카오 로그인 사용 시 Redirect URI를 등록해야 합니다. [등록하러 가기](#)

- 리다이렉트 도메인은 카카오로 로그인한 사용자의 인가코드를 받아 구글 서버와 통신하여 액세스토큰, 리프레쉬 토큰을 발급받기 위해 필요한 도메인입니다.

## Redirect URI

삭제

수정

Redirect URI	https://i7c201.p.ssafy.io/api/login/oauth2/code/kakao
<ul style="list-style-type: none"> <li>• 카카오 로그인에서 사용할 OAuth Redirect URI를 설정합니다. (최대 10개)</li> <li>• REST API로 개발하는 경우 필수로 설정해야 합니다.</li> </ul>	

## 나. 구글

- 서비스의 회원가입/로그인을 위해 구글 API를 이용하였습니다.
- 프로젝트 등록을 한 뒤, OAuth 클라이언트 ID를 등록합니다.
- 승인된 도메인을 사용하지 않을 경우 API는 테스트 버전만 사용가능합니다.
- 이를 위해 도메인을 https로 등록하였습니다.

Google Cloud Moggozi

API API 및 서비스 사용자 인증 정보 + CREATE CREDENTIALS DELETE

사용 설정된 API 라이브러리 사용자 인증 정보 OAuth 동의 화면 도메인 확인 페이지 사용 동의

API 키  
할당량과 액세스 권한을 확인하기 위해 간단한 API 키로 프로젝트를 확인합니다. [자세히 알아보기](#)

OAuth 클라이언트 ID  
앱에서 사용자 데이터에 액세스할 수 있도록 사용자 동의를 요청합니다.

서비스 계정  
로봇 계정을 사용하여 서버 간의 앱 수준 인증을 사용 설정합니다. [제한사항](#)

사용자 인증 정보 선택 도움말  
사용할 사용자 인증 정보의 유형을 결정할 수 있도록 몇 가지 질문을 합니다.

<input type="checkbox"/>	이름	생성일 ↓	유형	클라이언트 ID
<input type="checkbox"/>	Moggozi2	2022. 8. 10.	웹 애플리케이션 리케이션	948844328868-ot3h...

- 승인된 자바스크립트 원본은 프론트 서버의 주소를 의미합니다.
- 승인된 리다이렉션 URI란, 웹에서 로그인한 사용자의 인가코드를 통해 구글 인증 서버와 액세스 토큰, 리프레시 토큰을 주고받을 수 있는 주소를 의미합니다.

이름 \*  
Moggozi2

OAuth 2.0 클라이언트의 이름입니다. 이 이름은 콘솔에서 클라이언트를 식별하는 용도로만 사용되며 최종 사용자에게 표시되지 않습니다.

*i*
아래에 추가한 URI의 도메인이 승인된 도메인으로 OAuth 등의 화면에 자동으로 추가됩니다.

**승인된 자바스크립트 원본** *?*

브라우저 요청에 사용

URI 1 \*  
https://i7c201.p.ssafy.io

+ URI 추가

**승인된 리디렉션 URI** *?*

웹 서버의 요청에 사용

URI 1 \*  
https://i7c201.p.ssafy.io/api/login/oauth2/client/google

+ URI 추가

다. 네이버

- 서비스의 회원가입/로그인을 위해 네이버 **API**를 이용하였습니다.
- 타 **API**를 이용하는 것과 절차는 같으나 테스트가 아닌 실사용 모드를 위해서는 네이버의 인가가 필요합니다.
- 상세 이용방법과 네이버 아이콘이 어디에 배치되는지에 관한 정보를 작성하여 승인 절차를 거쳐야 합니다.

## Moggozi

개요	API 설정	멤버관리	로그인 통계	API 통계	Playground(Beta)
----	--------	------	--------	--------	------------------

### 애플리케이션 정보

Client ID	<input type="text" value="sxvllGBRak3r0RWC1QNV"/>
Client Secret	<div>*****</div> <div>보기</div>

### 네이버 로그인

로고 이미지	
개발 상태	<div>서비스 적용</div>

### API 호출 안내

지도 API 인증 실패나 네이버 로그인 이용 제한이 걸렸다면 [API 설정] 탭에서 URL 관련 설정을 수정하시면 정상 이용 가능합니다!!!

### 비로그인 오픈 API 당일 사용량

API 호출량 / 일일 허용량

'네이버 로그인' 애플리케이션 검수 결과,  
**'승인'이 완료되었습니다.**

안녕하세요.

'네이버 로그인'에서 요청하신 애플리케이션 검수 결과를 안내 드립니다.

'승인'이 되었으나, 수정이 필요한 부분이 있습니다.

#### 애플리케이션 정보

---

Client ID                **svllGBRak3z0iWC1QNv**

애플리케이션 이름 **Moggozi**

검수 결과                **승인**

#### 수정 요청 사항

1. 최초 사용자는 로그인 페이지뿐 아니라, 회원 가입 페이지를 통해서 첫 진입하기도 합니다. 로그인 페이지 뿐 아니라, 회원 가입 페이지에도 '네이버 아이디로 로그인' 버튼이 노출되도록 수정해 주세요.

- [네이버 아이디로 로그인 적용 가이드](#)를 참고하세요.

## 6. 외부서비스 - 파이어베이스

### 가. 파이어베이스를 활용한 이유

- 사진 저장을 외부 스토리지에 저장하여 사용하기 위해 파이어베이스의 **Storage**를 사용합니다.
- **Storage**의 폴더 경로를 활용해 사진 이미지 경로를 받아올 수 있어, **DB** 없이 사진을 업로드하고 불러와 사용할 계획이었으나 불러오는 속도가 느려 백엔드 **DB**에 사진 **url**을 저장하여 사용합니다.

### 나. 애플리케이션 추가

- **Firebase**의 새 프로젝트 생성합니다.
- **Google** 애널리틱스 사용 설정을 **OFF**로 설정하여야 합니다.
- 빌드 **Storage - Storage** 추가합니다.
- 프로덕션 모드에서 시작합니다.
- 지역을 **asia-northeast3**으로 설정합니다.

### 다. **React** 프로젝트에서 **Firebase** 사용법

- **FE** 폴더 **root** 경로에 있는 **.env** 파일에 제품의 **SDK**를 등록합니다.
- 타입스크립트를 사용하니 **npm i @types/firebase firebase**로 설치합니다.
- '**.env**' 파일에 저장된 **key**를 불러올 설정파일을 생성합니다.
  - **src/fbase/fbase.ts** : **firebase**라는 이름으로 사용하면 안되니 주의하여야 합니다.
  - 설정 코드는 다음과 같습니다.

```
fbase.ts x
src > fbase > fbase.ts > ...
1 import { initializeApp } from "firebase/app";
2 import * as firebaseAuth from "firebase/auth";
3 import { getStorage } from "firebase/storage";
4
5 const firebaseConfig = {
6   apiKey: process.env.REACT_APP_API_KEY,
7   authDomain: process.env.REACT_APP_AUTH_DOMAIN,
8   projectId: process.env.REACT_APP_PROJECT_ID,
9   storageBucket: process.env.REACT_APP_STORAGEBUCKET,
10  messagingSenderId: process.env.REACT_APP_MESSAGING_SENDER_ID,
11  appId: process.env.REACT_APP_APP_ID,
12 };
13
14 const app = initializeApp(firebaseConfig);
15
16 export const firebaseInstance = firebaseAuth;
17 export const storageService = getStorage(app, firebaseConfig.storageBucket);
18
```

#### 다. 사진 저장 및 활용

- storage에 저장할 스냅샷 레퍼런스 : `ref(storageService, <Storage 경로/파일 명>)`
- 이미지 업로드 : `uploadBytes`(위에서 뽑은 레퍼런스, image)
- 파일 다운로드 할 수 있는 url : `getDownloadURL`(레퍼런스)
- 이미지 제거 : `deleteObject`(레퍼런스)