

**November 18, 2022**

**SSAFY 7th TEAM Qhoto**

팀장: 박명규

팀원: 김상현, 김정아, 박영준, 유경훈, 정형진

목차

[개발 동기 2](#_Toc111773023)

[기술 스택 2](#_Toc111773024)

[빌드 상세내용 3](#_Toc111773025)

[백엔드 빌드 3](#_Toc111773026)

[빌드 4](#_Toc111773027)

[프론트 엔드 빌드 5](#_Toc111773028)

[배포 특이사항 6](#_Toc111773029)

[DB설정 7](#_Toc111773030)

[1. MySQL WorkBench 추가하기 7](#_Toc111773031)

[2. 계정 정보 입력 7](#_Toc111773032)

[프로퍼티 정의 8](#_Toc111773034)

[스프링부트 HTTPS 설정 10](#_Toc111773035)

[외부서비스 10](#_Toc111773036)

[TMap API 10](#_Toc111773037)

# 개발 동기

요즘 소비가 많이 되는 틱톡, 유튜브 쇼츠, 인스타그램 릴스 와 같은 짧은 영상. 간단하고 편하게 '나'의 일상을 공유할 수 있어 사용자들 사이에서 더욱 각광 받고 있습니다. 하지만 이들 모두 대체로 화려하거나, 재미있거나, 누구나가 찍어서 공유하기에는 어려움이 있습니다.

자신의 소소한 일상을 짧은 콘텐츠로 공유하면 어떨까 생각하며 **'Qhoto'**를 기획했습니다. 매일 다른 퀘스트를 수행하면서 지루한 일상에 소확행을 주고, 자신의 취미나 관심사를 공유하여 짧은 시간 동안 재미를 줄 수 있도록 하였습니다.

또한, 환경 카테고리에서는 환경 보존에 대한 미션을 수행하면서 환경 보존에 대한 기대도 할 수 있습니다. 이는 사용자가 늘어난다면 사회를 변화시키는 공익 프로젝트로 자리 잡을 수 있을 것으로 예상됩니다. 단순히 사용자의 성취감과 소통을 위한 것이 아닌 더 나은 사회를 위한 미션들을 통해 의미 있는 프로젝트가 될 것이라고 예상합니다.

# 기술 스택

1. 이슈관리 : Jira

2. 형상관리 : Gitlab

3. 커뮤니케이션 : mattermost, notion, webex, ERD Cloud

4. 개발 환경

1. OS : window 10

2. IDE

1. Intellij IDEA 2022.1.3

2. Visual Studio Code

3. UI/UX : Figma

3. Database : MySQL Workbench

4. Server : AWS EC2 (MobaXterm)

1. Ubuntu 20.04.4 LTS

5. 상세 사용

1. Backend

1. Java 11

2. Spring Boot gradle

3. Spring Boot 2.7.1

4. Lombok, Jpa

2. Frontend

1. React Native (0.70.3)

2. HTML5, CSS3, JavaScript(ES6)

3. Redux(8.0.4)

4. TypeScript

3. AWS

1. RDS(mysql)

2. EC2

3. Cloud Front

4. S3

4. Google Cloud Platform

1. compute engine

5. CI/CD

1. Jenkins

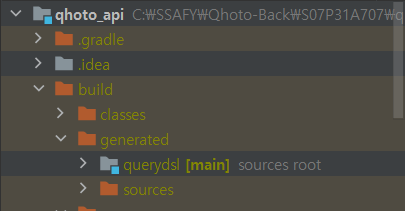
2. docker

6. web server

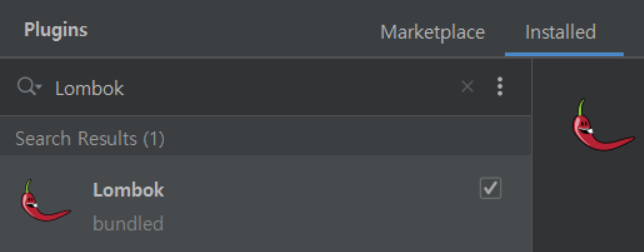
1. nginx

# 빌드 상세내용

## 백엔드 빌드

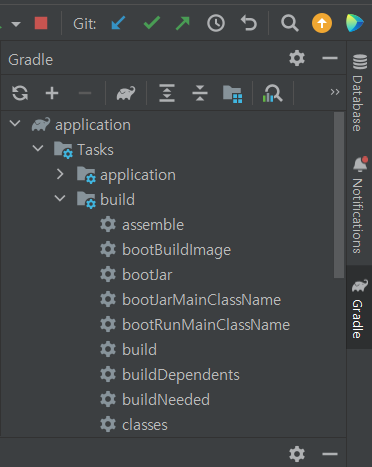


Intellij 에서 qhoto\_api를 gradle으로 import 합니다. import 이후, Lombok의 플러그인 설치를 해야합니다. Intellij의 File → Settings → Plugins 로 들어갑니다. Lombok을 검색한 뒤 설치 해줍시다.



installed에 Lombok이 설치되면 intellij를 재실행 해줍니다.

### 빌드



intellij 우측 상단에 Gradle 버튼을 클릭합니다.

위와 같이 뜨는 화면에 clean, validate, compile, install 순서대로 클릭을 하고 각각 실행이 완료 될 때까지 기다려 줍니다.

후에 qhoto\_api/target 폴더에 들어가면jar 파일이 생성된 것을 확인할 수 있습니다.

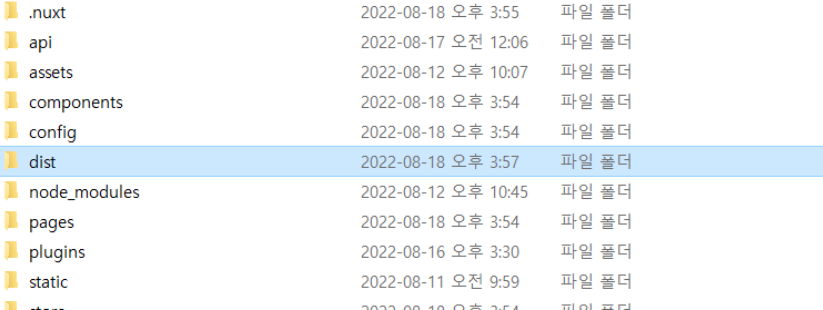
## 프론트 엔드 빌드

1. node\_module를 위한 기본 install

npm i

1. 빌드 명령어

npm run generate



dist 폴더 생성

## Jenkins 설정

Jenkins id\_rsa.pub

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDP4c5K9hqyESZy0brji1jGrpKeKZb2y7gfGVtZgjuZ/J3gUKsKdk/YDT1Q6SIRqystF8CAwnQLWOgwKC4ltZJodPb8yIzQrxSCmqXJPEnzjYBDCHwlbwAI/VRWxnnQjm9U/AdSB0Oel+BzT1gGqGqBXcKWDaYfpYXgRtQGh337ILZEMnpCKF3cPCAbe3mq3a2jvEy1WEnofJ08Y9sarnKU60S1kCPOCXwrc3wAekaikxerM01KJVeoDEYF1N5n/favJmZUVfv6rdh/s2N+Qw3MojCi/oYwuUM2WpEYP4KqXAk0mp4fencwwgM7zx6PbP1u2kbZUjJgQ9LYoK2Zc7bV audrb96@qhoto-jenkins

`

Worker instance authorized\_keys

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCOLrJwg0PFrruWT3mZ/1B2HGsfbIkPMfRpqx8AvMxgMFMAX4XiUEzontueuCq62qHygjyUXU/zTAUln4H1z55QBd7KMgTPqTNJEvVNZNogaKJ0KxVFqtoBywFvX7crcXLbpCahoMHJ2pluUZmij2uDkoJTwUjlcmrWBnxMXjum/MvN+xDSdO1kNEcTj+P+vRQaeNDuDO2Hqf70kXzJsC4CY0geLiLemxx+fFDJhhuHCDud0LyjZxkzno/U11YlmDjsGXxBcEfyzMOf/FAzB/kN26h2PN/g3JEWCb5AbJwAH944UGtE/6MGGAlSItNOcnCQHrOwGux+xbXaxp4dNmbT K7A707T ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDP4c5K9hqyESZy0brji1jGrpKeKZb2y7gfGVtZgjuZ/J3gUKsKdk/YDT1Q6SIRqystF8CAwnQLWOgwKC4ltZJodPb8yIzQrxSCmqXJPEnzjYBDCHwlbwAI/VRWxnnQjm9U/AdSB0Oel+BzT1gGqGqBXcKWDaYfpYXgRtQGh337ILZEMnpCKF3cPCAbe3mq3a2jvEy1WEnofJ08Y9sarnKU60S1kCPOCXwrc3wAekaikxerM01KJVeoDEYF1N5n/favJmZUVfv6rdh/s2N+Qw3MojCi/oYwuUM2WpEYP4KqXAk0mp4fencwwgM7zx6PbP1u2kbZUjJgQ9LYoK2Zc7bV audrb96@qhoto-jenkins

백엔드 excute shell

cd /var/lib/jenkins/workspace/qhoto-api-instance/qhoto\_api

chmod 544 gradlew

./gradlew clean build

./gradlew build

docker image prune

docker build -t audrb96/qhoto-api .

sudo docker push audrb96/qhoto-api

빌드 후 조치

Send build artifacts over SSH

docker images -f "dangling=true" -q

docker rmi -f $(docker images -f "dangling=true" -q)

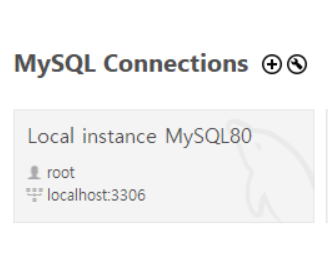
docker rm -f qhoto-api

docker pull audrb96/qhoto-api

docker run -d -p 7070:7070 --name qhoto-api audrb96/qhoto-api

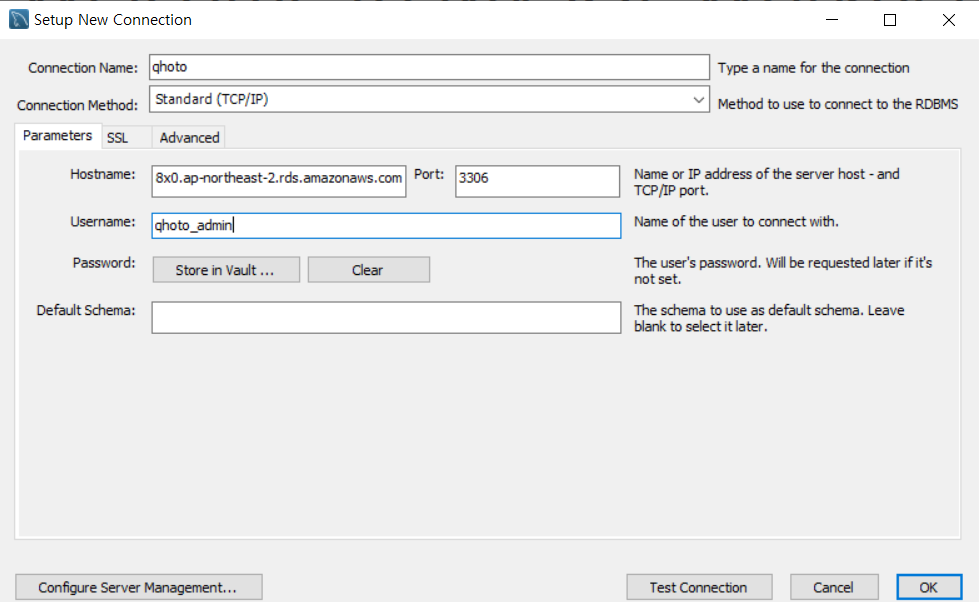
## DB설정

### 1. MySQL WorkBench 추가하기



MySQL WorkBench를 열어서 + 버튼을 눌러줍니다.

### 2. 계정 정보 입력



Hostname: qhoto-db.c2k2xt6qu8x0.ap-northeast-2.rds.amazonaws.com

Username: qhoto\_admin

Password: ssafyA707

# 프로퍼티 정의

1. nginx 세팅
   1. ec2에서 /etc/nginx/sites-available 파일로 접근

sudo apt-get update

* 1. nginx.conf 파일 편집

sudo vi /etc/nginx/nginx.conf

//로드 밸런싱 설정

upstream cpu-bound-app {

server k7A707.p.ssafy.io:7070 weight=100 max\_fails=3 fail\_timeout=3s;

server 34.64.51.121:7070 weight=100 max\_fails=3 fail\_timeout=3s;

server 34.64.127.65:7070 weight=100 max\_fails=3 fail\_timeout=3s;

}

server {

server\_name qhoto-api.com;

root /usr/share/nginx/html;

//requestBody max size

client\_max\_body\_size 50M;

# Load configuration files for the default server block.

include /etc/nginx/default.d/\*.conf;

//리버스 프록시 설정

location / {

proxy\_pass http://cpu-bound-app;

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection 'upgrade';

proxy\_set\_header Host $host;

proxy\_cache\_bypass $http\_upgrade;

}

error\_page 404 /404.html;

location = /404.html {

}

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

}

// https 설정

listen [::]:443 ssl ipv6only=on; # managed by Certbot

listen 443 ssl; # managed by Certbot

ssl\_certificate /etc/letsencrypt/live/qhoto-api.com/fullchain.pem; # managed by Certbot

ssl\_certificate\_key /etc/letsencrypt/live/qhoto-api.com/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

## 스프링부트 HTTPS 설정

SSL 인증서 없이 웹 애플리케이션을 서버에 올리면 http로 접속이 되고 경고 메시지가 나타납니다.

application.yaml 파일에 위의 코드를 넣어주어 해결합니다.

ssl:

key-store: classpath:keystore.p12

key-store-type: PKCS12

key-password: ssafyA702

# Redis 설정

$ sudo apt-get update

$ sudo apt-get upgrade

$ sudo apt-get install redis-server

$ redis-server –version #Redis 설치 확인

$ sudo vi /etc/redis.conf

bind 127.0.0.1 >> bind 0.0.0.0 으로 변경

port 6379

$ sudo systemctl restart redis-server.service

application.yaml

spring:

redis:

host: j7a702.p.ssafy.io

port: 6379

timeout: 6

# S3

application.yaml

cloud:

aws:

s3:

bucket: qhoto

region:

static: ap-northeast-2

credentials:

accessKey: AKIAVVSSTXJT4QWHBS7T

secretKey: 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

stack:

auto: false

# 외부서비스

# 1. KaKao Auth Login

# (1) kakao login 이란?

# - OAuth 2.0 기반의 소셜 로그인 서비스

- 사용자의 카카오톡 또는 카카오계정으로 서비스에 로그인 가능

# (2) kakao developers 추가 및 설정

- kakao developers의 내 애플리케이션에서 추가하기

- 내 애플리케이션 > 앱 설정 > 플랫폼 Web 사이트 도메인 수정

- kakao 애플리케이션에서 접속할 수 있게 사용할 도메인을 추가해줍니다.

- 내 애플리케이션 > 제품 설정 > 카카오 로그인 설정

- 활성화 설정을 on으로 해준 다음, redirect URL에 로그인 버튼 누른 후 이동할 Front page의 URL을 적어줍니다.

- 내 애플리케이션 > 제품 설정 > 카카오 로그인 > 동의항목

페이지 가입 시 사용자에게 동의 후 받을 정보를 설정합니다.