# 개요

## 프로젝트 개요

## 프로젝트 사용 도구

이슈 관리 : JIRA

형상 관리 : Gitlab

커뮤니케이션 : Notion, Mattermost

디자인 : Figma

UCC : 모바비

CI/CD : mobax, nginx, Jenkins

## 개발환경

VS Code : 1.70.2

IntelliJ : 11.0.13+7-b1751.21 amd64

JVM : 16.0.1 (스프링은 11로 빌드)

SERVER : AWS EC2 Ubuntu 20.04.3 LTS

DB : Mysql

## 외부 서비스

Kako API :

AWS s3 service:

## Gitgnore 처리한 핵심 키들

react :

Spring :

# 빌드

## 환경변수 형태

.env:

.application.properties:

# DB Source UR

#spring.datasource.url=jdbc:mysql://172.17.0.1:3306/hoydia\_demo?serverTimezone=UTC

spring.datasource.url=jdbc:mysql://127.0.0.1:3306/hoydia\_demo?serverTimezone=UTC

spring.datasource.username=root

spring.datasource.password=ssafy

# JPA

#spring.jpa.database-platform=org.hibernate.dialect.MySQL5InnoDBDialect // ddl-auto create

spring.jpa.hibernate.ddl-auto=update

spring.jpa.hibernate.properties.hibernate.format\_sql=true

spring.jpa.show-sql=true

# logging

logging.level.org.hibernate.SQL = debug

# Swagger

spring.mvc.pathmatch.matching-strategy=ant\_path\_matcher

app.auth.tokenSecret = temporarysecrettemporarysecrettemporarysecret

app.auth.tokenExpiry = 3333333333

spring.profiles.include = aws

##AWS S3

# AWS Account Credentials (AWS access key)

cloud.aws.credentials.accessKey=${S3.ACCESSKEY}

cloud.aws.credentials.secretKey=${S3.SECRETKEY}

# AWS S3 bucket Info

cloud.aws.s3.bucket=${S3.BUCKETNAME}

cloud.aws.region.static=${S3.REGION}

cloud.aws.stack.auto=false

# file upload max size

spring.servlet.multipart.max-file-size=5MB

spring.servlet.multipart.max-request-size=5MB

# AWS S3 bucket URL

cloud.aws.s3.bucket.url=${S3.URL}

.application-aws:

#AWS EC2

#EC2.DOMAIN = i7a708.p.ssafy.io ## ec2 ??? ?????? ???? ??? ??

EC2.DOMAIN = localhost

#MYSQL

MYSQL.USERNAME=root

MYSQL.PASSWORD=ssafy

#AWS S3

S3.URL = https://hoydia-bucket.s3.ap-northeast-2.amazonaws.com/

S3.BUCKETNAME = hoydia-bucket

S3.REGION = ap-northeast-2

S3.ACCESSKEY = AKIARZXTFJ4GELYHSWQN

S3.SECRETKEY = 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

# Security OAuth

# JWT 시크릿 키

jwt:

secret: tEa MaKDEV fOC usonP oweR fulSec Ur itY sER viCe

## 빌드하기

1) Front

npm I

npm run dev

or

npx vite

2) Back-spring

Gradle 실행

Bootjar 실행

3) Back-django

Python -m venv venv

Source venv/Scripts/activate

Pip install -r requirements.txt

Python manage.py runserver

## 배포하기

Nginx 설정

map $http\_upgrade $connection\_upgrade {

default upgrade;

'' close;

}

server {

listen 80 default\_server;

listen [::]:80 default\_server;

server\_name tupli.kr;

return 301 https://$server\_name$request\_uri;

}

server {

listen 443 ssl;

listen [::]:443 ssl;

server\_name tupli.kr;

ssl\_certificate /etc/letsencrypt/live/tupli.kr/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/tupli.kr/privkey.pem;

root /var/www/html/dist;

index index.html;

location / {

try\_files $uri $uri/ /index.html;

}

location /api/v1 {

proxy\_pass http://localhost:8080;

proxy\_redirect off;

charset utf-8;

rewrite /api/v1/(.\*) /$1 break;

proxy\_set\_header Host $host;

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 X-NginX-Proxy true;

}

location /api/v1/ws-stomp {

proxy\_pass http://localhost:8080;

rewrite /api/v1/(.\*) /$1 break;

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection "Upgrade";

proxy\_set\_header Host $host;

}

}