**HIFES**

**I. 개요**

**1. 프로젝트 개요**

**코로나가 종식되면서 이제는 축제도 다시 활성화되는 추세인데, 축제에 참가하는데 불편한 점들이 많습니다. 예를 들면, 부스의 위치를 알기 힘들다거나, 같이 온 친구들과 소통이 잘 안된다거나, 실시간 정보를 받기가 힘들다는 점 등입니다. 그래서 이런 불편한 점들을 해결하기 위해서 HiFes앱을 기획했습니다.**

**2. 프로젝트 사용도구**

* 이슈 관리 : Jira
* 형상 관리 : Gitlab
* 커뮤니케이션 : Mattermost, Notion
* 디자인 : Figma
* UCC :
* CI/CD : Jenkins

**3. 개발환경**

* Java 11
* Spring Boot 2.7.14
* IntelliJ Ultimate
* MariaDB 10.11
* Server : AWS EC2 Ubuntu 20.04.3 LTS

**4. 외부 서비스**

* Naver Map API
* Kakao Login API
* Firebase Cloud Messaging V1

**5. Gitignore 처리한 핵심 키들**

# Gradle files

.gradle/

build/

### oauth ###

/src/main/resources/application-oauth.properties

# Local configuration file (sdk path, etc)

local.properties

# Log/OS Files

\*.log

# IntelliJ

\*.iml

.idea/

misc.xml

deploymentTargetDropDown.xml

render.experimental.xml

# Google Services (e.g. APIs or Firebase)

google-services.json

### Java ###

# Compiled class file

\*.class

# Log file

# BlueJ files

\*.ctxt

# Mobile Tools for Java (J2ME)

.mtj.tmp/

# Package Files #

\*.jar

\*.war

\*.nar

\*.ear

\*.zip

\*.tar.gz

\*.rar

### Gradle ###

.gradle

\*\*/build/

!src/\*\*/build/

# Ignore Gradle GUI config

gradle-app.setting

# Avoid ignoring Gradle wrapper jar file (.jar files are usually ignored)

!gradle-wrapper.jar

# Avoid ignore Gradle wrappper properties

!gradle-wrapper.properties

# Cache of project

.gradletasknamecache

# IntelliJ IDEA

\*.iws

/out/

# User-specific configurations

.idea/caches/

.idea/libraries/

.idea/shelf/

.idea/workspace.xml

.idea/tasks.xml

.idea/.name

.idea/compiler.xml

.idea/copyright/profiles\_settings.xml

.idea/encodings.xml

.idea/misc.xml

.idea/modules.xml

.idea/scopes/scope\_settings.xml

.idea/dictionaries

.idea/vcs.xml

.idea/jsLibraryMappings.xml

.idea/datasources.xml

.idea/dataSources.ids

.idea/sqlDataSources.xml

.idea/dynamic.xml

.idea/uiDesigner.xml

.idea/assetWizardSettings.xml

.idea/gradle.xml

.idea/jarRepositories.xml

.idea/navEditor.xml

# Legacy Eclipse project files

.cproject

.settings/

# virtual machine crash logs (Reference: <http://www.java.com/en/download/help/error\_hotspot.xml>)

## Plugin-specific files:

# mpeltonen/sbt-idea plugin

.idea\_modules/

!/gradle/wrapper/gradle-wrapper.jar

**II 빌드**

**1. 환경변수**

[application.properties](http://application.properties) :

# DATABASE

spring.jpa.database-platform=org.hibernate.dialect.MariaDBDialect

#spring.datasource.url=jdbc:mariadb://localhost:3306/board

#spring.datasource.username=admin

#spring.datasource.password=dodo1234

spring.jpa.show-sql=true

spring.jpa.hibernate.ddl-auto=update

spring.mvc.converters.preferred-json-mapper=gson

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

# jwt key

jwt.secretKey=dksfajdk38jdfjf920fj920f2j9f02j0f3j39jfidkjf932j4kjdfd902sfdsijdif9

jwt.access.expiration=21600000

jwt.access.header=Authorization

jwt.refresh.expiration=1209600000

jwt.refresh.header=Authorization

springdoc.version=1.6.9

spring.jpa.hibernate.naming.implicit-strategy=org.hibernate.boot.model.naming.ImplicitNamingStrategyLegacyJpaImpl

spring.jpa.hibernate.naming.physical-strategy=org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl

spring.main.allow-bean-definition-overriding=true

# fcm key

fcm.key.path=gjgs-fcm.json

fcm.key.scope=https://www.googleapis.com/auth/cloud-platform

build.gradle :

plugins {

id 'java'

id 'org.springframework.boot' version '2.7.14'

id 'io.spring.dependency-management' version '1.0.15.RELEASE'

}

group = 'hiFes'

version = '0.0.1-SNAPSHOT'

java {

sourceCompatibility = '11'

}

configurations {

compileOnly {

extendsFrom annotationProcessor

}

}

repositories {

mavenCentral()

}

dependencies {

implementation 'org.springframework.boot:spring-boot-starter-validation'

implementation 'org.springframework.boot:spring-boot-starter-data-jpa'

implementation 'org.springframework.boot:spring-boot-starter-oauth2-client'

implementation 'org.springframework.boot:spring-boot-starter-security'

implementation 'org.springframework.boot:spring-boot-starter-thymeleaf'

implementation 'org.springframework.boot:spring-boot-starter-web'

implementation 'commons-io:commons-io:2.11.0'

implementation 'org.projectlombok:lombok'

developmentOnly 'org.springframework.boot:spring-boot-devtools'

runtimeOnly 'org.mariadb.jdbc:mariadb-java-client'

compileOnly 'org.projectlombok:lombok'

annotationProcessor 'org.projectlombok:lombok'

developmentOnly 'org.projectlombok:lombok'

testImplementation 'org.springframework.boot:spring-boot-starter-test'

testImplementation 'org.springframework.security:spring-security-test'

implementation 'io.jsonwebtoken:jjwt-api:0.11.5'

implementation 'io.jsonwebtoken:jjwt-impl:0.11.5'

implementation group: 'com.google.code.gson', name: 'gson', version: '2.8.5'

implementation 'com.auth0:java-jwt:4.2.1'

implementation group: 'org.apache.poi', name: 'poi', version: '4.1.2'

implementation group: 'org.apache.poi', name: 'poi-ooxml', version: '4.1.2'

implementation group: 'org.apache.tika', name: 'tika-core', version: '2.3.0'

implementation group: 'org.springframework.boot', name: 'spring-boot-starter-webflux'

implementation 'commons-fileupload:commons-fileupload:1.4'

implementation 'org.springdoc:springdoc-openapi-ui:1.6.9'

implementation 'com.google.firebase:firebase-admin:6.8.1'

implementation group: 'com.squareup.okhttp3', name: 'okhttp', version: '4.2.2'

implementation 'org.modelmapper:modelmapper:2.4.4'

}

tasks.named('test') {

useJUnitPlatform()

}

**2. 빌드하기**

Back-Spring Boot

* Gradle 실행
* Bootjar 실

**3. 배포하기**

pipeline {

agent {

docker {

// 젠킨스에서 사용할 Docker 이미지 설정

// 여기서는 Gradle과 Java가 설치된 이미지를 사용합니다.

image 'gradle:jdk11'

args '--network=host' // Docker 컨테이너를 호스트의 네트워크와 공유

}

}

environment {

// 빌드 결과물이 저장될 경로

BUILD\_DIR = "/var/www/backend"

DOCKER\_CONTAINER\_NAME = "hifes\_develop.app"

}

stages {

stage('Git Clone') {

steps {

git branch: 'develop', credentialsId: '45c98653-ac5e-4ec6-ac6f-9fb104c5dbc9',

url: '<https://lab.ssafy.com/s09-webmobile4-sub2/S09P12D104>'

}

}

stage('Build') {

steps {

sh 'pwd'

sh 'ls -al'

// 스프링 프로젝트 빌드

sh 'chmod +x backend/hiFes/gradlew'

sh 'cd backend/hiFes && ./gradlew clean build'

}

}

stage('Deploy') {

steps {

// 기존 서버 중지

sh "docker stop $DOCKER\_CONTAINER\_NAME || true" // 기존 컨테이너가 없을 경우를 위해 무시

sh "docker rm $DOCKER\_CONTAINER\_NAME || true" // 기존 컨테이너가 없을 경우를 위해 무시

// 빌드된 파일을 원하는 위치로 배포

// 예: 서버에 배포하는 경우

sh "mkdir -p $BUILD\_DIR"

sh "cp backend/hiFes/build/libs/hiFes-0.0.1-SNAPSHOT.jar $BUILD\_DIR"

sh "cd $BUILD\_DIR"

sh "pwd"

sh "ls -al"

// 새로운 서버 실행

sh "docker run -d -p 8001:8001 --name $DOCKER\_CONTAINER\_NAME --network=host -v $BUILD\_DIR:/app -v /home/ubuntu/images:/home/ubuntu/images gradle:jdk11 java -jar /app/hiFes-0.0.1-SNAPSHOT.jar"

}

}

}

post {

//success {

//echo '빌드 성공! 서버가 구동되었습니다.'

// 추가 작업 수행 가능

//}

//failure {

//echo '빌드 실패...'

// 실패 시 추가 작업 수행 가능

//}

//mattermost 연동 하고 싶을 시 mattermost plugin 설치 후 작성

success {

script {

def Author\_ID = sh(script: "git show -s --pretty=%an", returnStdout: true).trim()

def Author\_Name = sh(script: "git show -s --pretty=%ae", returnStdout: true).trim()

mattermostSend (color: 'good',

message: "배포 성공!!!: ${env.JOB\_NAME} #${env.BUILD\_NUMBER} by ${Author\_ID}(${Author\_Name})\\n(<${env.BUILD\_URL}|Details>)",

endpoint: '<https://meeting.ssafy.com/hooks/dokezizx7pfw7d89iftxd6mqww>',

channel: 'legend'

)

}

}

failure {

script {

def Author\_ID = sh(script: "git show -s --pretty=%an", returnStdout: true).trim()

def Author\_Name = sh(script: "git show -s --pretty=%ae", returnStdout: true).trim()

mattermostSend (color: 'danger',

message: "배포 실패!!!: ${env.JOB\_NAME} #${env.BUILD\_NUMBER} by ${Author\_ID}(${Author\_Name})\\n(<${env.BUILD\_URL}|Details>)",

endpoint: '<https://meeting.ssafy.com/hooks/dokezizx7pfw7d89iftxd6mqww>',

channel: 'legend'

)

}

}

}

}

**4. 서비스 이용방법**

1. 카카오톡 로그인을 한다.