A309 Porting manual

목차

1. 개발 환경

- **운영 체제:** Ubuntu 22.04 LTS (AWS EC2)
- **프레임워크:** Spring Boot 3.4.2
- **데이터베이스:** MySQL 8.0.41 (AWS RDS), MongoDB 2.3.9
- **메시지 브로커:** Apache Kafka 2.8.1
- **캐시:** Redis 7.4.2
- **파일(이미지) 저장소:** AWS S3
- **CI/CD:** Jenkins + Docker
- **프로그래밍 언어:** Java 17, Python 3.10, JavaScript (React Native, Expo)
- **버전 관리:** Git (GitLab), Jira

---

2. 라이브러리 버전

백엔드 (Spring Boot)

- Spring Boot: 3.4.2
- Spring Security: 3.4.2
- JPA (Hibernate): 3.4.2
- Redis: 7.4.2
- Kafka Client: 3.8.1
- Lombok: 1.18.36
- JWT: 0.11.5
- WebSocket : 10.1.34

Python 서버 (FastAPI)

- FastAPI: 0.115.8
- Google Cloud Vision API

모바일 클라이언트 (React Native & Expo)

- React Native: 0.76.6

- Expo : 52.0.35
- Axios: 1.7.9

---

3. 프로젝트 설정 파일 (application, env)

백엔드 (`application.properties`)

```
spring.application.name=opt-back
spring.data.redis.port=6379
spring.data.redis.host=redis
spring.datasource.url=jdbc:mysql://i12a309.p.ssafy.io:3307/opt?useSSL=tr
ue&characterEncoding=UTF-8&serverTimezone=UTC
spring.datasource.username=opt
spring.datasource.password=rkddlrlarladhcjs309
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.data.mongodb.uri=mongodb://localhost:27017/admin
spring.data.mongodb.database=admin
jwt.secretKey="fightingA309fightingghkdlxlddpdltkarhdrnQhkdlxld"
spring.security.user.name=user
spring.security.user.password=1111
logging.level.org.springframework.security=DEBUG
#S3
cloud.aws.credentials.accessKey=AKIAZN5YNYZ3VW5HG7GL
cloud.aws.credentials.secretKey=WB2Zs/THjqfFnwaVuDiykGYSzAaavPWG
J7Uefwab
cloud.aws.region.static=ap-northeast-2
cloud.aws.stack.auto-=false
spring.servlet.multipart.max-file-size=10MB
spring.servlet.multipart.max-request-size=10MB
profile.image.bucket.name=opt-profile-images
exercise.media.bucket.name=opt-exercise-media
review.image.bucket.name=opt-review-images
meal.image.bucket.name=opt-meal-images
#jpa
spring.jpa.hibernate.ddl-auto=update
#gpt
```

```
openai.api.key=sk-svcacct-SDbf_09UZfXr-SvcA0bi8ZJx7nFTbcqjDz_EJnJz
qoufa2P6bGJ1jwSkxsWyTKuco_XTT3BlbkFJPxMkn47sK79fn2XvBuWerA4-
4IbvW9fMoK6X8qckRndvc_wL7N4t5jFGfNVSV1uKpfYA
openai.model=gpt-3.5-turbo
openai.api.url=https://api.openai.com/v1/chat/completions
#kakao
kakao.auth.client=b74e72417a62d1b6d0fc3b1e25087c3d
kakao.auth.redirect=https://i12a309.p.ssafy.io/auth/kakao
kakao.auth.token-uri=https://kauth.kakao.com/oauth/token
kakao.auth.member-info-uri=https://kapi.kakao.com/v2/user/me
kakao.auth.logout=https://kapi.kakao.com/v1/user/logout
kakao.auth.unlink=https://kapi.kakao.com/v1/user/unlink

# kafka
spring.kafka.bootstrap-servers=kafka:9092
# kafka consumer
spring.kafka.consumer.group-id=spring-group
spring.kafka.consumer.auto-offset-reset=latest
spring.kafka.consumer.enable-auto-commit=true
spring.kafka.consumer.max-poll-records=10
# kafka consumer additional
spring.kafka.consumer.properties.isolation.level=read_committed
spring.kafka.consumer.properties.fetch.min.bytes=1
fcm.project.id=ssafy-opt
fcm.secret.file=ssafy-opt-firebase-adminsdk-fbsvc-ba5665439a.json

business.license.bucket.name=opt-business-license
certificate.bucket.name=opt-certificate
challenge.image.bucket.name=opt-challenge-images
```

백엔드 (`project_root/.env`)

```
HUGGINGFACE_TOKEN=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
```

Python 서버 환경 변수 (`opt-fast/.env`)

```
DATABASE_URL=mysql+aiomysql://opt:rkddlrlarladhcjs309@i12a309.p.ssafy.io:3307/opt
# GCP Document AI 정보
PROJECT_ID=opt-ocr
LOCATION=us
PROCESSOR_ID=bfe9513a7655ad9e
# open api
SERVICE_KEY=i3ftXJDy4R3wwGYkuUy29G0cG0kG5MZdfGeQl4elR8Naw7K5j28aFHB65n80Y%2B4VAZj2dwzlWiCCgRplrl%2BNLg%3D%3D
GOOGLE_APPLICATION_CREDENTIALS=/app/opt-ocr-938590945c20-wk51sbg4tbgs9g83j6hid1×4kw.json
UVICORN_HOST=localhost
UVICORN_PORT=8000
UVICORN_RELOAD=True
HUGGINGFACE_TOKEN=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
```

모바일 (`.env`)

```
EXPO_PUBLIC_BASE_URL=https://i12a309.p.ssafy.io
EXPO_PUBLIC_API_KEY=b74e72417a62d1b6d0fc3b1e25087c3d
EXPO_PUBLIC_KAKAO_API_KEY=66d3dcfbe76e5757180675235bdde36f
EXPO_PUBLIC_KAKAO_REST_API_KEY=2298aef09ede9d8a34308bbb415ee6fb
```

---

4. 사용된 외부 API

- **Google Cloud Vision API**: OCR 기능 제공
- **Kakao Map API**: 트레이너의 헬스장 위치 표시
- **AWS S3**: 이미지 및 파일 저장
- **ExerciseDB** : 약 1300여 개의 운동 데이터 제공
- 모션 인식 및 운동 측정
  - **ML Kit (Google Firebase ML Kit)**
- 키워드 추출
  - **Hugging Face Transformers API** → `jhgan/ko-sroberta-multitask` 모델 로딩
  - **Sentence-BERT (SBERT)** → 키워드와 토픽 간 유사도 비교
  - **PyTorch** → 모델 실행 및 유사도 계산
  - **Okt (Open Korean Text)** → 형태소 분석 및 토큰 병합 (자연어 처리)

---

5. 빌드 및 배포 방법

docker-compose를 이용한 빌드

docker 설치

```
sudo apt update
sudo apt install -y docker.io
```

docker-compose 설치 및 권한 부여

```
sudo curl -L "https://github.com/docker/compose/releases/latest/download/d
sudo chmod +x /usr/local/bin/docker-compose
```

docker-compose 파일

```
version: "3.9"

services:
  mysql:
    image: mysql:8.0
    container_name: mysql
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: rkddlrlarladhcjs309
      MYSQL_DATABASE: opt
      MYSQL_USER: opt
      MYSQL_PASSWORD: rkddlrlarladhcjs309
    ports:
      - "3307:3306"
    volumes:
      - /var/lib/mysql:/var/lib/mysql
    networks:
      opt:

  redis:
```

```
  image: redis:latest
  container_name: redis
  restart: always
  ports:
    - "6380:6379"
  volumes:
    - ./redis_data:/data
  command: ["redis-server", "--appendonly", "yes"]
  networks:
    opt:

mongo:
  image: mongo:latest
  container_name: mongo
  restart: always
  environment:
    MONGO_INITDB_ROOT_USERNAME: opt
    MONGO_INITDB_ROOT_PASSWORD: rkddlrlarladhcjs309
  ports:
    - "27018:27017"
  volumes:
    - ./mongo_data:/data/db
    - /etc/mongod.conf:/etc/mongod.conf
  networks:
    opt:

backend:
  build:
    context: /home/ubuntu/docker/backend-src
    dockerfile: Dockerfile  # Spring 설정에서 가져옴
  container_name: backend
  restart: always
  ports:
    - "8080:8080"
  environment:
    - DB_HOST=mysql
    - DB_USER=opt
    - DB_PASSWORD=rkddlrlarladhcjs309
```

```
    - REDIS_HOST=redis
    - REDIS_PORT=6379
    - SPRING_KAFKA_BOOTSTRAP_SERVERS=kafka:9092
    - MONGO_HOST=mongo
    - MONGO_PORT=27017
    - MONGO_DATABASE=admin
    - MONGO_USERNAME=opt
    - MONGO_PASSWORD=rkddlrlarladhcjs309
  depends_on:
    - mysql
    - redis
    - mongo
    - kafka  # Spring 설정에서 가져옴
  platform: linux/amd64  # 아키텍처 강제 설정
  networks:
    opt:

nginx:
  image: nginx:latest
  container_name: nginx
  restart: always
  ports:
    - "80:80"
    - "443:443"
  volumes:
    - /home/ubuntu/docker/nginx/nginx.conf:/etc/nginx/nginx.conf:ro
    - /home/ubuntu/docker/nginx/sites-enabled:/etc/nginx/sites-enabled
    - /etc/letsencrypt:/etc/letsencrypt:ro  # SSL 인증서 마운트
    - /var/www/html:/var/www/html  # 웹 루트 마운트
  depends_on:
    - backend
  networks:
    opt:

jenkins:
  #    image: jenkins/jenkins:jdk21
  container_name: jenkins
  build:
```

```
    context: jenkins-dockerfile
    dockerfile: Dockerfile
  restart: always
  user: root
  ports:
    - "9090:8080"  # Jenkins 웹 인터페이스 포트
    - "50000:50000"  # Jenkins 에이전트 포트
  volumes:
    - ./jenkins_home:/var/jenkins_home  # Jenkins 데이터 저장 볼륨
    - /var/run/docker.sock:/var/run/docker.sock  # Jenkins가 Docker 실행 가능하
  networks:
    opt:

zookeeper:
  image: wurstmeister/zookeeper:latest
  container_name: zookeeper
  # platform: linux/amd64  # 강제 아키텍처 설정
  ports:
    - "2181:2181"
  environment:
    ZOOKEEPER_CLIENT_PORT: 2181
  networks:
    opt:

kafka:
  image: wurstmeister/kafka:latest
  container_name: kafka
  ports:
    - "9092:9092"
  environment:
    KAFKA_BROKER_ID: 1
    KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
    KAFKA_LISTENERS: PLAINTEXT://kafka:9092
    # KAFKA_ADVERTISED_HOST_NAME: 127.0.0.1
    # KAFKA_ADVERTISED_LISTENERS: PLAINTEXT://kafka:9092
    KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR: 1
    # SPRING_KAFKA_BOOTSTRAPSERVERS: kafka:9092
    KAFKA_CREATE_TOPICS: "business_license_request:1:1,business_license_
```

```
    depends_on:
      - zookeeper
    networks:
      opt:

  fastapi:
    build:
      context: /home/ubuntu/docker/opt-fast
      dockerfile: Dockerfile
      args:
        - HUGGINGFACE_TOKEN=${HUGGINGFACE_TOKEN}
    env_file:
      - /home/ubuntu/docker/.env
    container_name: fastapi
    ports:
      - "8000:8000"
      - "5000:5000"
    depends_on:
      - kafka
    environment:
      - BOOTSTRAP_SERVERS=kafka:9092
    # entrypoint: ["uvicorn main:app --host 0.0.0.0 --port 8000"]
    volumes:
      - /home/ubuntu/docker/opt-fast:/app
    networks:
      opt:

networks:
  opt:
    driver: bridge

volumes:
  mysql_data:
  redis_data:
  jenkins_home:
```

Jenkins에서 CI/CD Pipeline 구성

```
pipeline {
    agent any

    environment {
        JAVA_HOME = "/var/jenkins_home/java-21-openjdk"
        PATH = "${JAVA_HOME}/bin:${PATH}"
        GIT_REPO = 'https://lab.ssafy.com/s12-webmobile2-sub1/S12P11A309.git
        BRANCH = 'BackEnd'
    }

    stages {

        stage('Clean Workspace') {
            steps {
                sh 'rm -rf ${WORKSPACE}/*'  // 기존 워크스페이스 삭제
            }
        }

        stage('git clone repository') {
            steps {
                sh 'echo $JAVA_HOME'
                sh 'java -version'
                git branch: "${BRANCH}", url: "${GIT_REPO}", credentialsId: 'gitlab-t
            }
        }
        stage('build') {
            steps {
                dir("${WORKSPACE}/opt-back") {
                    sh "chmod +x gradlew"
                    sh "./gradlew clean build -x test"
                }
            }
        }

        stage('deploy') {
            steps {
                withCredentials([sshUserPrivateKey(credentialsId: 'aws-key', keyFile
                    sh '''
```

```
                    echo "Removing existing project files from remote server..."
                    ssh -i $SSH_KEY -o StrictHostKeyChecking=no ubuntu@172.26
                    ssh -i $SSH_KEY -o StrictHostKeyChecking=no ubuntu@172.26
                    ssh -i $SSH_KEY -o StrictHostKeyChecking=no ubuntu@172.26


                    scp -i $SSH_KEY -r ${WORKSPACE}/opt-back/* ubuntu@172.2(
                    scp -i $SSH_KEY -r ${WORKSPACE}/opt-fast ubuntu@172.26.13

                    echo "Executing deploy script on remote server..."
                    ssh -i $SSH_KEY -o StrictHostKeyChecking=no ubuntu@172.26
                    mkdir -p /home/ubuntu/docker/backend-src/src/main/resource
                    touch /home/ubuntu/docker/opt-fast/.env
                    touch /home/ubuntu/docker/opt-fast/opt-ocr-938590945c20-w
                    cp /home/ubuntu/S12P11A309/opt-ocr-938590945c20-wk51sb
                    cp /home/ubuntu/.env /home/ubuntu/docker/opt-fast/.env
                    cp /home/ubuntu/S12P11A309/application.properties /home/ubu
                    /home/ubuntu/docker/deploy.sh
                '''
            }
        }
    }
}
```

백엔드 (Spring Boot)와 Python 서버 (FastAPI)

EC2 서버에서 docker-compose.yml이 있는 디렉토리로 이동 후, 아래의 명령어를 실행

```
docker compose -d up --build
```

fastapi 서버 jdk설치 및 환경 변수 설정(EC2접속 후 실행)

```
docker exec -it fastapi /bin/sh
apt update && apt install -y openjdk-17-jdk
readlink -f $(which java)
export JAVA_HOME=/usr/lib/jvm/java-17-openjdk-amd64
export PATH=$JAVA_HOME/bin:$PATH
```

```
echo $JAVA_HOME
export LD_LIBRARY_PATH=$JAVA_HOME/lib/server:$LD_LIBRARY_PATH
echo $LD_LIBRARY_PATH
```

모바일 앱 (React Native & Expo)

```
eas build -p android --profile preview
```

---

6. 프로젝트 실행 방법

전제 조건 : EC2 서버에 docker의 이미지가 실행 된 상태

VSCode에서 /project_root/opt-front 디렉토리로 이동
아래 명령어들을 통해 패키지 설치

```
npx expo install
npm install @react-navigation/native
npx expo install react-native-screens react-native-safe-area-context react-native-gesture-handler react-native-reanimated react-native-vector-icons
npm install @tanstack/react-query
npm install jotai
npm install recoil
npm install axios
npx expo install @react-native-async-storage/async-storage
npx expo install firebase
npm install react-hook-form
npm install yup @hookform/resolvers
npm install nativewind
npx expo install react-native-svg
npm install expo-router
npm install @gorhom/bottom-sheet
npm install react-native-dotenv
```

/project_root/opt-front에 .env 파일 추가하기

```
EXPO_PUBLIC_BASE_URL=https://i12a309.p.ssafy.io
EXPO_PUBLIC_API_KEY=b74e72417a62d1b6d0fc3b1e25087c3d
EXPO_PUBLIC_KAKAO_API_KEY=66d3dcfbe76e5757180675235bdde36f
EXPO_PUBLIC_KAKAO_REST_API_KEY=2298aef09ede9d8a34308bbb415ee6fb
```

디바이스와 연결 후 모바일 앱 실행 (로컬 테스트)

```
npx expo start -c
```

---

7. .gitignore 파일에 포함된 내용

```
HELP.md
.gradle
build/
!gradle/wrapper/gradle-wrapper.jar
!**/src/main/**/build/
!**/src/test/**/build/
.env.infastapi
.env.outfastapi
.env

### STS ###
.apt_generated
.classpath
.factorypath
.project
.settings
.springBeans
.sts4-cache
bin/
!**/src/main/**/bin/
!**/src/test/**/bin/

### IntelliJ IDEA ###
```

```
.idea
*.iws
*.iml
*.ipr
out/
!**/src/main/**/out/
!**/src/test/**/out/

### NetBeans ###
/nbproject/private/
/nbbuild/
/dist/
/nbdist/
/.nb-gradle/

### VS Code ###
.vscode/
/src/main/resources/application.properties
/src/main/resources/docker-compose.yml
```