**[SSAFY 12기 A204팀]**

1. **시스템 환경 설정(사용한 JVM, 웹서버, WAS 제품 등의 종류와 설정 값, 버전(IDE버전 포함) 기재)**
   1. **Backend**
      1. JDK
         1. Oracle OpenJDK 17.0.12
      2. WAS
         1. Apache Tomcat (Spring Boot 내장)
      3. Image
         1. openjdk:17-jdk-alpine
         2. port
            1. 8080
         3. context path : /
         4. encoding : UTF-8
      4. Development Tool
         1. IDE : InteliJ IDEA 2024.3.3 (Ultimate Edtion)
         2. Build Tool : Gradle 8.12.1
      5. Framework
         1. Spring Boot : 3.4.3
      6. Dependencies
         1. Spring-boot-starter-data-jpa
         2. Lombok
         3. mysql-connector-j
         4. firebase-admin 9.4.3
         5. spring-boot-starter-data-redis
         6. queryDSL
         7. minio 8.5.17
   2. **Android App**
      1. SDK
         1. minSDK : 26
         2. targetSDK : 35
      2. Development Tool
         1. IDE : Android Studio Ladybug Feature Drop | 2024.2.2
         2. Build Tool : Gradle 8.11.1
      3. Framework
         1. JetPack Compose
      4. Dependencies
         1. compose
         2. gson 2.10.1
         3. mapbox SDK 11.10.3
         4. altbeacon 2+
         5. com.github.bumptech.glide 4.16.0
         6. retrofit2 2.9.0
         7. hilt
         8. core-splashscreen 1.0.1
   3. **Frontend**
      1. JS Runtime
         1. node.js 22.13.1
      2. Image
         1. nginx:alpine
         2. Port
            1. 관리자 페이지 : 80
      3. Development Tool
         1. Vite
         2. ESLint
         3. Prettier
         4. vite-plugin-svgr
         5. Visual Studio Code
      4. Dependencies
         1. react 19.0.0
         2. react-dom 19.0.0
         3. react-router-dom 7.4.0
         4. axios 1.8.4
         5. mapbox-gl 3.10.0
   4. **Streaming Server**
      1. JS Runtime
         1. Node.js 22.13.0
      2. Image
         1. node:22-alpine
         2. Port
            1. url 요청 : 3000
            2. CCTV 스트리밍 : 10000-10100
      3. Development Tool
         1. Visual Studio Code
      4. Dependencies
         1. axios 1.8.4
         2. cors 2.8.5
         3. dotenv 16.4.7
         4. express 4.21.2
         5. fluent-ffmpeg 2.1.3
         6. form-data 4.0.2
         7. fs
         8. node-rtsp-stream 0.0.9
         9. sqlite3 5.1.7
         10. ws 8.18.1
   5. **AI Server**
      1. Image
         1. Python 3.11.9
         2. Port
            1. 8102 -> 8000
      2. AI Model
         1. YOLO v11
            1. n\_best.pt (경량)
            2. x\_best.pt (고성능)
      3. Dependencies

* + - 1. 
  1. **Infra**
     1. Redis 7.4.2
        1. Port
           1. 6379
     2. MinIO RELEASE.2025-03-12T18-04-18Z
        1. Port
           1. Web UI : 8101 -> 9001
           2. API : 8100 -> 9000
     3. MariaDB ( SSAFY 제공 )
        1. Port
           1. 3306
  2. **CI/CD** 
     1. Docker 28.0.1
     2. Jenkins 2.492.2
        1. Port
           1. 8000 -> 8080
           2. 50000 -> 50000
        2. Plugins
           1. Timestamper
           2. Workspace Cleanup Plugin
           3. Strict Crumb Issuer Plugin
           4. Pipeline
           5. Gradle Plugin
           6. GitLab Plugin
           7. GitLab API Plugin
           8. Generic Webhook Trigger Plugin
           9. Build Timeout
  3. **IP CCTV**
     1. 실제 IP CCTV 송출 대체
     2. Protocol : RTSP
     3. Tool
        1. OBS
        2. OBS-RTSPServer 3.1.0

1. **빌드 시 사용되는 환경 변수 등의 내용 상세 기재**
   1. **Backend** 
      1. env.yml
         1. 경로
            1. 프로젝트 루트 디렉토리 (for test)
            2. Jenkins 마운트 볼륨 (for build)
         2. 항목
            1. MYSQL\_URL
            2. MYSQL\_USERNAME
            3. MYSQL\_PASSWORD
            4. REDIS\_HOST
            5. REDIS\_PASSWORD
            6. REDIS\_USERNAME
            7. MINIO\_URL
            8. MINIO\_ACCESS\_KEY
            9. MINIO\_SECRET\_KEY
      2. Jangan204-firebase-adminsdk.json
         1. Firebase 프로젝트에서 발급  
            
   2. **Android**
      1. google-services.json
         1. 경로
            1. {루트 디렉토리}/app
         2. Firebase 프로젝트에서 발급
      2. local.properties
         1. 경로
            1. 루트 디렉토리
         2. 항목
            1. MAPBOX\_DOWNLOADS\_TOKEN
   3. **Frontend** 
      1. .env
         1. 경로
            1. 프로젝트 루트 디렉토리 (for test)
            2. Jenkins 마운트 볼륨 (for build)
         2. 항목
            1. VITE\_MAPBOX\_ACCESS\_TOKEN

Mapbox access 토큰

* + - * 1. VITE\_API\_BASE\_URL

NGINX 서버 주소

* + - * 1. VITE\_NODE\_API\_BASE\_URL

Streaming 서버 주소

* 1. **Streaming Server**
     1. .env
        1. 경로
           1. 프로젝트 루트 디렉토리
        2. 항목
           1. SPRING\_API\_URL
           2. FAST\_API\_URL
  2. **AI Server**
     1. fastapi.env
        1. 경로
           1. 프로젝트 루트 디렉토리 (for test)
           2. Jenkins 마운트 볼륨 (for build)
        2. 항목
           1. REDIS\_HOST
           2. REDIS\_PORT
           3. REDIS\_PASSWORD
  3. Infra
     1. **MinIO**
        1. minio.env
           1. 경로

Jenkins 마운트 볼륨

* + - * 1. 항목

MINIO\_ROOT\_USER

MINIO\_ROOT\_PASSWORD

MINIO\_SERVER\_URL

MINIO\_BROWSER\_REDIRECT\_URL

* + 1. **Redis**
       1. redis.conf
          1. 경로

Jenkins 마운트 볼륨

* + - * 1. 수정사항

requirepass 수정

bind 주소 수정

1. **배포 시 특이사항 기재**
   1. EC2
      1. Jenkins 공유 파일 디렉토리 생성
         1. ~/env\_files
         2. ~/Jenkins\_home
      2. Docker 설치
         1. Docker Volume 사전 생성
            1. $ docker volume create minio-data
            2. $ docker volume create redis-data
         2. Docker network 사전 생성
            1. $ docker network create –driver=bridge –subnet=172.18.0.0/16 jangan-net
      3. Compose 파일
         1. docker-compose-backend.yml



* + - 1. docker-compose-frontend.yml



* + - 1. docker-compose-fastapi.yml
    1. Docker Container 설치
       1. $ docker run -d --name janganjenkins -p 8000:8080 -p 50000:50000 -e DOCKER\_GID=999 -v /home/ubuntu/jenkins\_home:/var/jenkins\_home -v /home/ubuntu/env\_files:/env\_files -v /var/run/docker.sock:/var/run/docker.sock jenkins/jenkins:lts
       2. $ docker run -d --name jangan-redis --network jangan-net -p 6379:6379 -v redis-data:/data -v ./redis.conf:/usr/local/etc/redis/redis.conf redis redis-server /usr/local/etc/redis/redis.conf
       3. $ docker run -d -p 8100:9000 -p 8101:9001 --name jangan-minio -v minio-data:/data --network jangan-net --env-file minio.env quay.io/minio/minio:RELEASE.2025-03-12T18-04-18Z server /data --console-address ":9001"
    2. Jenkins
       1. Credentials 설정
          1. GitLab account
       2. Pipeline 생성
          1. jangan-ai, jangan-backend, jangan-frontend
          2. GitLab Webhook 설정, pipeline과 연동 (Jenkins Token 생성 필요)
          3. Pipeline script 작성

jangan-ai



jangan-backend



jangan-frontend



* 1. MinIO
     1. 컨테이너 실행 후 Web UI 접속
     2. Access Key 발급
     3. Spring Boot env.yml 파일에 토큰 추가
     4. ufw 방화벽 포트 설정
        1. 80 (Web)
        2. 8000 (Jenkin)
        3. 8101 (MinIO)
        4. 8102 (AI Server)
  2. Streaming Server
     1. IP CCTV(OBS)가 RTSP 영상 스트리밍을 송출 중인 네트워크 내에 설치
     2. 절차
        1. Docker image build
           1. $ docker build –t cctv-server .
        2. Docker 컨테이너 생성 / 실행
           1. $ docker run --name cctv-stream -p 3000:3000 -p 10000-10010:10000-10010 cctv-server
  3. IP CCTV
     1. OBS 설치
     2. OBS-RTSPServer 3.1.0 플러그인 설치
     3. 송출 해상도 : 1280 \* 720
     4. 송출 프레임 : 60 FPS
     5. OBS RTSP 서버 송출 주소 : rtsp://localhost:554/live
  4. Mapbox
     1. Mapbox 회원가입 후 Key 관리
     2. Secret Key 생성 -> read, download 권한 필요
     3. Frontend, mobile에 추가

1. **DB 접속 정보 등 프로젝트(ERD)에 활용되는 주요 계정 및 프로퍼티가 정의된 파일 목록**
   1. .env
   2. env.yml
   3. redis.conf
   4. fastapi.env
   5. jangan204-firebase-adminsdk.json