포팅 메뉴얼

서버 버전 설정

백앤드

- JAVA - `17`
- Spring Boot - `3.4.2`
- Spring Data JPA - `3.4.2`
- Spring Security - `6.6.2`
- JJWT - `0.11.2`
- Auth0 JWT - `4.4.0`
- MySQL - `8.3.0`
- Redis - `7.2.4`
- AWS Secrets Manager Config - `2.4.4`
- Log4j2
- Nurigo SDK (문자 메시지) - `4.3.2`
- JCodec (비디오 처리) - `0.2.5`
- Commons IO - `2.11.0`

프론트앤드

- Flutter SDK - `>=3.0.0 <4.0.0`
- Flutter Version - 1.0.0+1
- 상태 관리 - Flutter Riverpod `2.4.9`
- 네트워크 - Dio `5.7.0`
- 라우팅 - Go Router `13.0.1`

- 데이터 저장 - Shared Preferences 2.5.1
- 위치 서비스 - Geolocator 13.0.2
- 로그 관리 - Logger 2.5.0
- 비디오/이미지
  - Camera 0.10.5+9
  - Video Player 2.8.1
  - Image Picker 1.1.2
  - Saver Gallery 3.0.2
  - FC Native Video Thumbnail 0.17.2
  - Video Compress 3.1.3
- 권한 관리 - Permission Handler 11.3.1
- 색상 선택 - Flutter Colorpicker 1.1.0
- 3D 모델링 - Flutter 3D Objects 1.0.2+1
- 로컬 알림 - Flutter Local Notifications 18.0.1
- OAuth/Kakao SDK
  - Kakao Flutter SDK 1.9.6
  - Kakao Flutter SDK Auth 1.9.6
- 기계 학습 (ML Kit)
  - Google MLKit Pose Detection 0.13.0
  - Google MLKit Commons 0.9.0
- TTS (음성 변환) - Flutter TTS 4.2.2
- 캘린더 - Table Calendar 3.1.3
- 환경 변수 - Flutter Dotenv 5.2.1
- 딥링크 - App Links 6.4.0
- 웹 뷰 - WebView Flutter 4.9.0
- 스플래시 화면 - Flutter Native Splash 2.4.4

인프라

- Ubuntu - `22.04.3 LTS`
- Jenkins - `2.448`
- Docker - `25.0.5`
- Nginx - `1.18.0`

포트 설정

- Jenkins : `9005 → 8080`
- Backend : `8080 → 8080`
- AI Server : `8000 → 8000`
- Redis : `6379 → 6379`
- Mysql : `3306 → 3306`
- Nginx : `80 → 80`

Ubuntu

EC2 접속

```
ssh -i I12E206T.pem ubuntu@i12e206.p.ssafy.io
```

서버 기본 세팅

```
sudo timedatectl set-timezone Asia/Seoul
sudo apt-get -y update && sudo apt-get -y upgrade
```

JDK 설정

```
sudo apt list openjdk-17*
sudo apt install openjdk-17-jdk
```

Docker

패키지 설치

```
sudo apt update
sudo apt install ca-certificates curl gnupg lsb-release

sudo mkdir -p /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --d

echo "deb [arch=$(dpkg --print-architecture) \
signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/
$(. /etc/os-release && echo "$VERSION_CODENAME") stable" | sudo tee /etc/

sudo apt update
sudo apt install docker-ce docker-ce-cli containerd.io docker-compose-plugi
```

Jenkins

Jenkins 실행

```
docker run -itd --name jenkins -p 9005:8080 \ -v /var/run/docker.sock:/va
r/run/docker.sock \ jenkins/jenkins:jdk17
```

Docker 소켓 보안 설정 개선

```
sudo usermod -aG docker jenkins
sudo systemctl restart docker
```

Jenkins 초기 패스워드 확인

```
docker exec jenkins cat /var/jenkins_home/secrets/initialAdminPassword
```

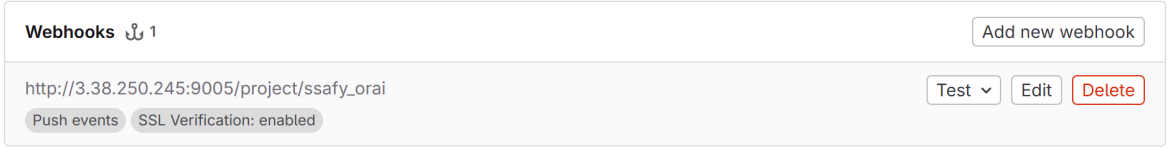
Jenkins 파이프라인 플러그인 설치

- GitLab
- Docker
- GitLab Authentication
- Generic WebHook Trigger
- ssh

환경 변수 및 Credential 설정

T	P	Store ↓	Domain	ID
		System	(global)	gitlab-token
		System	(global)	3c1d6aad-eb79-4568-9aa4-3d2b622caf89
		System	(global)	DOCKER_USER
		System	(global)	DOCKER_REPO
		System	(global)	EC2_SERVER_IP
		System	(global)	my-ssh-credentials
		System	(global)	DOCKER_HUB
		System	(global)	application-yml

GitLab Connection, WebHook 설정



Tools 설정

- JDK - 17
- Gradle - 8.5
- Docker - latest

Nginx

Nginx Docker 이미지

```
docker pull nginx:latest
```

nginx 리버스 프록시 설정

- /etc/nginx/sites-available/default 설정

```
server {
    listen 80;
    server_name i12e206.p.ssafy.io;
    return 301 https://$host$request_uri;
}

server {
    listen 443 ssl http2;
    server_name i12e206.p.ssafy.io;

    ssl_certificate /etc/letsencrypt/live/i12e206.p.ssafy.io/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/i12e206.p.ssafy.io/privkey.pem;

    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers HIGH:!aNULL:!MD5;
    ssl_prefer_server_ciphers on;

    root /var/www/html;
    index index.html index.htm index.nginx-debian.html;

    client_max_body_size 100M;
```

```
location /api/ {
    proxy_pass http://localhost:8080;
    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;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;

    add_header 'Access-Control-Allow-Origin' '*' always;
    add_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS, PUT
    add_header 'Access-Control-Allow-Headers' '*';
    add_header 'Access-Control-Allow-Credentials' 'true';

    if ($request_method = 'OPTIONS') {
        return 204;
    }
}

location /fastapi/ {
    proxy_pass http://localhost:8000;
    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;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;

    add_header 'Access-Control-Allow-Origin' '*' always;
    add_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS, PUT
    add_header 'Access-Control-Allow-Headers' '*';
    add_header 'Access-Control-Allow-Credentials' 'true';
}

location / {
```

```
        try_files $uri $uri/ =404;
    }
}
```

Certbot, SSL 인증서

Let's Encrypt 및 Certbot을 사용한 SSL 인증서 발급

```
sudo apt-get install letsencrypt
sudo apt-get install certbot python3-certbot-nginx

sudo certbot --nginx
sudo certbot --nginx -d i12e206.p.ssafy.io

sudo service nginx restart
sudo systemctl reload nginx
```

Infrastructure Setup

Database Services

- docker-compose.yml

```
services:
  mysql:
    image: mysql:latest
    container_name: mysql
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: "1234"
      MYSQL_DATABASE: "climbing"
    ports:
      - "3306:3306"
    volumes:
      - mysql_data:/var/lib/mysql
    networks:
```

```
      - mynetwork

  redis:
    image: redis:latest
    container_name: redis
    restart: always
    ports:
      - "6379:6379"
    networks:
      - mynetwork

volumes:
  mysql_data:

networks:
  mynetwork:
    driver: bridge
    external: true
```

Application Services

- docker-compose.yml

```
services:
  api:
    build:
      context: .
      dockerfile: Dockerfile
    container_name: ssafy_orai_api_1
    restart: always
    depends_on:
      - mysql
      - redis
    ports:
      - "8080:8080"
    networks:
      - mynetwork
```

```
  ai:
    build:
      context: ./ai
      dockerfile: Dockerfile
    container_name: ai
    restart: unless-stopped
    depends_on:
      - backend_api
    ports:
      - "8000:8000"

  jenkins:
    image: jenkins/jenkins:jdk17
    container_name: jenkins
    restart: unless-stopped
    ports:
      - "9005:8080"
    volumes:
      - jenkins_home:/var/jenkins_home
    environment:
      - JENKINS_OPTS=--prefix=/jenkins

  nginx:
    image: nginx:latest
    container_name: nginx
    restart: unless-stopped
    ports:
      - "80:80"
    volumes:
      - ./nginx/nginx.conf:/etc/nginx/nginx.conf
    depends_on:
      - backend_api
      - ai

networks:
  mynetwork:
    external: true
```

Jenkins Pipline Item - Backend

Pipline Script

```
pipeline {
    agent any

    tools {
        jdk("jdk17")
    }

    stages {
        stage('Git Clone') {
            steps {
                git branch: 'develop/BE',
                    credentialsId: '3c1d6aad-eb79-4568-9aa4-3d2b622caf89',
                    url: 'https://lab.ssafy.com/s12-webmobile4-sub1/S12P11E206.git'
            }
        }

        stage('Setup Network') {
            steps {
                script {
                    sh '''
                    if ! docker network ls | grep -q mynetwork; then
                        echo "Creating mynetwork..."
                        docker network create mynetwork
                    else
                        echo "mynetwork already exists"
                    fi
                    '''
                }
            }
        }
```

```
        stage('Build') {
            steps {
                sh 'chmod +x backend/gradlew'
                sh 'cd backend && ./gradlew clean build -x test'
            }
        }

        stage('Setup Configuration') {
    steps {
        script {
            withCredentials([file(credentialsId: 'application-yml', variable: 'APPLICA
                sh '''
                    chmod 777 backend
                    cd backend
                    pwd

                    mkdir -p config
                    chmod 777 config
                    cp $APPLICATION_YML config/

                '''
                    }
                }
            }
        }


        stage('Deploy Spring Application') {
            steps {
                script {
                    withCredentials([usernamePassword(credentialsId: 'DOCKER_REP
                    passwordVariable: 'DOCKER_PROJECT',
                    usernameVariable: 'DOCKER_REPO')]) {
                        sh """
                        pwd
                        docker stop ssafy_orai_api_1 || true
                        docker rm -f ssafy_orai_api_1 || true
```

```
                cd /var/jenkins_home/workspace/ssafy_orai/backend
                docker-compose -f docker-compose.yml up --build -d api
                """
            }
        }
    }
  }
}
```

application.yml

```
spring:
  data:
    redis:
      host: redis
      port: 6379
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://mysql:3306/climbing?useSSL=false&serverTimezone=Asia
    username: root
    password: 1234
    hikari:
      pool-name: jpa-hikari-pool
      maximum-pool-size: 5
      jdbc-url: ${spring.datasource.url}
      username: ${spring.datasource.username}
      password: ${spring.datasource.password}
      driver-class-name: ${spring.datasource.driver-class-name}
      data-source-properties:
        rewriteBatchedStatements: true

  jpa:
    database-platform: org.hibernate.dialect.MySQLDialect
    generate-ddl: true
    hibernate:
```

```
      ddl-auto: none
    show-sql: true
    properties:
      hibernate:
        format_sql: true
        default_batch_fetch_size: 100
        jdbc.batch_size: 20
        order_inserts: true
        order_updates: true
        jdbc.time_zone: Asia/Seoul
  jackson:
    time-zone: Asia/Seoul


kakao:
  client-id:
  redirect-uri: https://i12e206.p.ssafy.io/api/user/social/kakao/callback
  client-secret:

app:
  auth:
    tokenSecret:
    tokenExpiry: 1800000
    refreshTokenExpiry: 604800000
  oauth2:
    authorizedRedirectUris:
      - http://localhost:3000/oauth/redirect

logging:
  level:
    org:
      springframework:
        data:
          redis: DEBUG
    io:
      lettuce:
        core: DEBUG
```

```
cloud:
  aws:
    s3:
      bucket: ssafy-ori-bucket
    credentials:
      access-key:
      secret-key:
    region:
      static: ap-northeast-2
      auto: false
    stack:
      auto: false
```