CI/CD 포팅 매뉴얼

1. 사용도구

- 이슈 관리 : Jira
- 형상 관리 : GitLab
- 커뮤니케이션 : Notion, MatterMost
- 디자인 : Figma
- CI/CD : Jenkins

2. 개발 도구

- Visual Studio Code : 1.86.00
- Intellij : 2023.3.0(Ultimate Edition)
- Arduino 2.3.0

3. 개발 환경

IoT

- Arduino Mega 2560
- Web Socket

- LiquidCrystal_I2C
- Adafruit_NeoPixel

Frontend

node: v20.10.0

react: v18.2.0

create-react-app

Backend

JAVA: 17

Spring Boot 3.1.8

gradle 8.5

Server

EC2 (RAM 16G)

OS: Ubuntu

Service

Jenkins **2.426.3**

Docker 25.0.3

Docker-Compose 1.29.2

4. 환경변수

Back

- application.yml

```
spring:
 datasource:
   driver-class-name: com.mysql.cj.jdbc.Driver
   url: jdbc:mysql://localhost:3306/soyu
   username: root
```

```
    password: yourpassword
  profiles:
    include: auth
  servlet:
    multipart:
      max-file-size: 50MB
      max-request-size: 50MB

  jpa:
    hibernate:
      ddl-auto: validate
    properties:
      hibernate:
        #         show_sql: true # sysout 단계에서 쿼리문을 날린다
        format_sql: true
        default_batch_fetch_size: 1000 # select 배치 조회 크기

logging:
  level:
    org.hibernate.SQL: debug
    org.hibernate.type: trace # log 단계에서 쿼리문을 나타낸다 !

file:
  path:
    upload-images: '/home/ubuntu/soyu'

server:
  servlet:
    context-path: '/api'
```

Front

- .env

```
REACT_APP_FCM_APIKEY="your_password"
REACT_APP_FCM_AUTHDOMAIN="soyu-fe75a.firebaseapp.com"
REACT_APP_FCM_PROJECTID="soyu-fe75a"
REACT_APP_FCM_STORAGEBUCKET="soyu-fe75a.appspot.com"
REACT_APP_FCM_MESSAGINGSENDERID="592048161601"
REACT_APP_FCM_APPID="1:592048161601:web:d2b8c5829193486ec517c0"
REACT_APP_FCM_MEASUREMENTID="G-your_MEASUREMENTID"
```

```
REACT_APP_FCM_VAPID_KEY="your_key"
REACT_APP_NAVER_CLIENT_ID="your_client_id"
REACT_APP_NAVER_REDIRECT_URI="your_redirect_uri"
REACT_APP_NAVER_STATE="hLiDdL2uhPtsftcU"
REACT_APP_MANIFEST="manifest.json"
REACT_APP_BASE_URL="http://localhost:8080"
```

IoT(WebSocket Client)

```
server.port=9000
```

5. CI/CD 구축

Jenkins `20001:20001`

- EC2 환경에 직접 다운로드
- 20001 번의 포트로 Jenkins 설정, gitLap, WebHook 연결

pipe-line

```
pipeline {
    agent any

    tools {
        gradle "gradle"
        nodejs "nodejs"
    }

    stages {
        stage('Clone') {
            steps {
                git branch: 'develop',
                credentialsId: 'gitlab token',
                url: 'https://lab.ssafy.com/s10-webmobile3-sub2/S10P
            }
        }
         stage('secret download') {
```

```
            steps {
                withCredentials([file(credentialsId: 'secret-auth', 
                                file(credentialsId: 'secret-firebase
                                file(credentialsId: 'front-env', var
                                file(credentialsId: 'docker-compose'
                    script {
                        sh 'sudo cp $authConfigFile back/src/main/re
                        sh 'sudo cp $firebaseConfigFile back/src/mai
                        sh 'sudo cp $frontEnvFile front/.env'
                        sh 'sudo cp $dockerComposeFile ./docker-comp
                    }
                }
            }
        }
         stage('Backend Build') {
            steps {
                dir('back') {
                    sh 'gradle clean build'
                }
            }
        }
        stage('Backend image') {
            steps {
                dir('back') {
                    sh 'docker build -t back .'
                    sh 'docker image prune -f'
                }
            }
        }
        stage('Front Build') {
            steps {
                dir('front') {
                    sh 'npm install'
                    sh 'CI=false npm run build'
                }
            }
        }
        stage('Front image') {
            steps {
                dir('front') {
                    sh 'docker build -t front .'
                    sh 'docker image prune -f'
                }
```

```
            }
        }
        stage('Docker Compose') {
            steps {
                dir('') {
                    sh 'docker-compose -f docker-compose.yml up -d'
                }
            }
        }
    }
}
```

DB `3300:3306`

mysql image 생성 → compose 시 컨테이너 등록

Spring `8080:8080`

```
FROM openjdk:17-ea-jdk-slim
ARG JAR_FILE=./build/libs/*.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java","-jar","app.jar"]
```

React `3000:3000`

```
FROM node:21
RUN npm install -g serve
RUN mkdir ./build
ADD ./build ./build
ENTRYPOINT ["serve", "-s", "build"]
```

Docker-Compose

```
version: '3'

services:
```

```
  database:
    container_name: soyu-DB
    image: mysql:latest
    restart: always
    environment:
      MYSQL_DATABASE: soyu
      MYSQL_ROOT_HOST: '%'
      MYSQL_ROOT_PASSWORD: yourpaassword
#     db 서버 문자셋 utf-8 설정 and 서버에서 데이터 요청시 utf-8 인코딩 사용
      MYSQL_CHARSET: utf8mb4
      MYSQL_COLLATION: utf8mb4_unicode_ci
#     db 시간 동기화
      TZ: Asia/Seoul
    volumes:
      - ./db:/var/lib/mysql
    ports:
      - "3300:3306"
    networks:
      - soyu-network

  back:
    container_name: soyu-back
    restart: always
    image: back:latest
    ports:
      - "8080:8080"
    environment:
      SPRING_DATASOURCE_URL: jdbc:mysql://database:3306/soyu?allowPu
      SPRING_DATASOURCE_USERNAME: root
      SPRING_DATASOURCE_PASSWORD: yourpaassword
#     시간 동기화
      TZ: Asia/Seoul
    volumes:
      - ./upload-images:/home/ubuntu/soyu
    networks:
      - soyu-network

  front:
    container_name: soyu-front
    restart: always
    image: front:latest
    ports:
```

```
      - "3000:3000"
    environment:
      TZ: Asia/Seoul
    networks:
      - soyu-network

networks:
  soyu-network:
    driver: bridge
```

NGINX

```
FROM nginx:stable-alpine
COPY --from=build /app/build /usr/share/nginx/html
RUN rm /etc/nginx/conf.d/default.conf
COPY nginx/nginx.conf /etc/nginx/conf.d
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

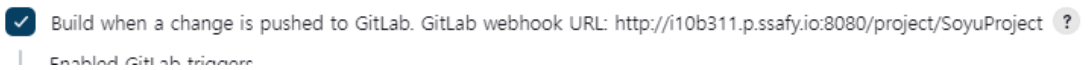
6. 빌드 및 실행

SoyuProject

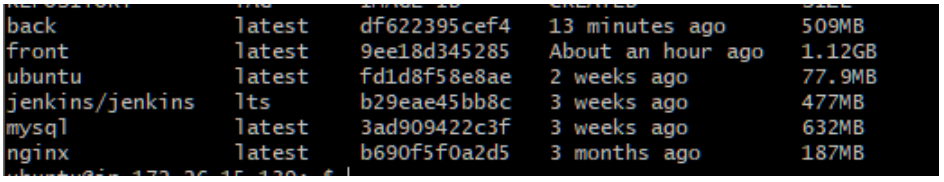
Stage View

	Declarative: Tool Install	Clone	secret download	Backend Build	Backend image	Front Build	Front image	Docker Compose
Average stage times: (Average full run time: ~1min 2s)	116ms	492ms	1s	25s	5s	22s	2s	4s
#263 2월 14일 21:48 No Changes	115ms	542ms	1s	25s	4s	21s	2s	4s
#262 2월 14일 21:07 No Changes	118ms	442ms	1s	25s	5s	22s	2s	4s

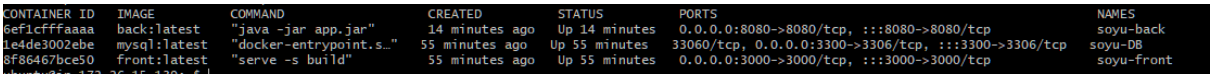
- 지정해 놓은 webhook commit 이 발생하면 자동 빌드가 실행됨



- back build, front build, image build 진행



- compose 통한 컨테이너 생성



- EC2 의 로컬의 NGINX 리버시 프록시를 통해서 어플리케이션 사용이 가능하다

```
# /etc/nginx/nginx.conf
user www-data;
worker_processes auto;
pid /run/nginx.pid;

events {
        worker_connections 768;
}

http {
        sendfile on;
        tcp_nopush on;
        tcp_nodelay on;
        keepalive_timeout 65;
        types_hash_max_size 2048;

        include /etc/nginx/mime.types;
        default_type application/octet-stream;

        ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3; # Dropping SSLv
        ssl_prefer_server_ciphers on;
```

```
        access_log /var/log/nginx/access.log;
        error_log /var/log/nginx/error.log;

        gzip on;

        client_max_body_size 0;
                include /etc/nginx/conf.d/*.conf;

}

# /etc/nginx/conf.d/domain-name.conf
server {
    root /usr/share/nginx/html;
    server_name soyubox.shop www.soyubox.shop;

    listen [::]:443 ssl ipv6only=on; # managed by Certbot
    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/soyubox.shop/fullchain.pem
    ssl_certificate_key /etc/letsencrypt/live/soyubox.shop/privkey.p
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Ce
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Cert

    location /api {
        proxy_pass http://localhost:8080;
    }

    location /api/ws/chat {
        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;
    }

    location /naver/callback {
        try_files $uri $uri/ /index.html;
    }
}
server {
    if ($host = www.soyubox.shop) {
        return 301 https://$host$request_uri;
    } # managed by Certbot
```

```
    if ($host = soyubox.shop) {
        return 301 https://$host$request_uri;
    } # managed by Certbot


    listen 80;
    listen [::]:80;
    server_name soyubox.shop www.soyubox.shop;
    return 404; # managed by Certbot

}
```

- www.soyubox.shop DNS RECODE