포팅 매뉴얼

버전

- FrontEnd
  - node: 16.13.0
  - npm: 8.1.0
- BackEnd
  - java 11
  - spring
    - spring boot: 2.7.10
    - gradle: 6.8.3

Server 설정

▼ Docker

1. Docker 설치

```
sudo apt-get update
```

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

▼ MariaDB

1. MaraiDB Docker 설치

```
sudo docker pull maraidb/server:latest
```

2. 컨테이너 생성 및 실행

```
sudo docker run --name mariadb -e MARIADB_ROOT_PASSWPRD={password} -p 3306:3306 -d mariadb/server:latest
```

▼ Nginx & SSL

1. Nginx 설치

```
sudo apt-get install nginx
```

2. Nginx 중지

```
sudo systemctl stop nginx
```

3. Let’s Encrypt 설치

```
sudo apt-get install letsencrypt
```

4. 인증서 적용

```
sudo letsencrypt certonly --standalone -d [도메인]
```

5. Nginx conf 파일

```
cd /etc/nginx/sites-available

sudo vi config.conf
```

```
server {
        location /{
                proxy_pass http://localhost:3000;
        }


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

        location /flask{
                proxy_pass http://localhost:5000/flask;

        }

        listen 443 ssl;
        ssl_certificate /etc/letsencrypt/live/[도메인]/fullchain.pem;
        ssl_certificate_key /etc/letsencrypt/live/[도메인]/privkey.pem;
}
server {
        if ($host = [도메인]) {
           return 301 https://$host$request_uri;
        }
           listen 80;
           server_name [도메인];
        return 404;
}
```

6. 파일 연동

```
sudo ln -s /etc/nginx/sites-available/config.conf /etc/nginx/sites-enabled/config.conf
```

7. Nginx 재시작

```
sudo systemctl restart nginx
```

▼ Jenkins

1. Jenkins 설치

```
sudo docker pull jenkins/jenkins:latest
```

2. jenkins 실행

```
sudo docker run -d -p [포트 번호]:8080 -v /jenkins:/var/jenkins_home -v /var/run/docker.sock:/var/run/docker.sock --name jenkin
```

3. https://{도메인}:{포트 번호} 접속

▼ Backend .yml 파일

```
spring:
  datasource:
    url: jdbc:mariadb:/[도메인]:3306/eyecan
    username: root
    password: [password]
    driver-class-name: org.mariadb.jdbc.Driver
  jpa:
    open-in-view: false
    generate-ddl: true
    hibernate:
      ddl-auto: update
    properties:
      hibernate:
```

```
        show_sql: true
      format_sql: true
      use_sql_comments: true
  servlet:
    multipart:
      max-request-size: 10MB
      max-file-size: 10MB
  logging:
    level:
      root: trace
      org:
        hibernate:
          type:
            descriptor:
              sql: trace
  security:
    oauth2:
      client:
        registration:
          kakao:
            client-id: [client-id]
            client-secret: [client-secret]
            scope:
              - profile_nickname
              - account_email
            redirect-uri: "https:/[도메인]/api/login/oauth2/code/kakao"
            authorization-grant-type: authorization_code
            client-name: Kakao
            client-authentication-method: POST

        provider:
          kakao:
            authorization-uri: https://kauth.kakao.com/oauth/authorize
            token-uri: https://kauth.kakao.com/oauth/token
            user-info-uri: https://kapi.kakao.com/v2/user/me
            user-name-attribute: id
server:
  port: 8080
  servlet:
    contextPath: /api

cloud:
  aws:
    credentials:
      access-key: [access key value]
      secret-key: [secret key value]
    s3:
      bucket: eye-can-speak
      dirName: upload
    region:
      static: ap-northeast-2
    stack:
      auto: false

jwt:
  secretKey: [jwt secretKey]
  refreshKey: [jwt refreshKey]
```

Backend 배포

▼ Spring Boot Jenkins pipeline

```
pipeline{
    environment{
        registry="{user name}/{repository}:{tag}"
        registryCredential='{docker hub credential}'
        dockerImage=''
    }
    agent any
    stages{
        stage('gitlab clone'){
            steps{
                git branch: 'BE',
                credentialsId: '{gitlab credential}',
                url: 'https://lab.ssafy.com/s08-final/S08P31D204'
            }
        }
        stage('build'){
            steps{
                sh "cp /var/jenkins_home/property/*.yml /var/jenkins_home/workspace/Back/backend/ecs-spring/src/main/resources"
                dir('backend/ecs-spring'){
```

```
                    sh '''
                        chmod +x ./gradlew
                        ./gradlew clean build
                    '''
                }
            }
        }
        stage('docker build'){
            steps{
                dir('backend/ecs-spring'){
                    script{
                        dockerImage=docker.build registry
                    }
                }
            }
        }
        stage('docker image push'){
            steps{
                script{
                    docker.withRegistry('', registryCredential){
                        dockerImage.push()
                    }
                }
            }

        }
        stage('ssh-server'){
            steps{
                sshagent(credentials: ['{ssh credential}']){
                    sh'''
                        ssh -o StrictHostKeyChecking=no ubuntu@{ip} "sudo docker run --name ecs_back -d -p 8080:8080 {docker image
                    '''
                }
            }
        }
    }
}
```

▼ Fastapi Jenkins pipeline

```
pipeline{
    environment{
        registry="{user name}/{repository}:{tag}"
        registryCredential='{docker hub credential}'
        dockerImage=''
    }
    agent any
    stages{
        stage('gitlab clone'){
            steps{
                git branch: 'AI2',
                credentialsId: '{gitlab credential}',
                url: 'https://lab.ssafy.com/s08-final/S08P31D204'
            }
        }
        stage('docker build'){
            steps{
                dir('backend/ecs-fastapi'){
                    script{
                        dockerImage=docker.build registry
                    }
                }
            }
        }
        stage('docker image push'){
            steps{
                script{
                    docker.withRegistry('', registryCredential){
                        dockerImage.push()
                    }
                }
            }
        }
        stage('ssh-server'){
            steps{
                sshagent(credentials: ['{ssh credential}']){
                    sh'''
                        ssh -o StrictHostKeyChecking=no ubuntu@{ip} "sudo docker run --name ecs_fastapi -d -p 5000:5000 {docker im
                    '''
                }
            }
        }
    }
}
```

Frontend 배포

▼ React Jenkins pipeline

```
pipeline{
    environment{
        registry="{user name}/{repository}:{tag}"
        registryCredential='{docker hub credential}'
        dockerImage=''
    }
    agent any
    stages{
        stage('git clone'){
            steps{
                git branch: 'FE',
                credentialsId: '{gitlab credential}',
                url: 'https://lab.ssafy.com/s08-final/S08P31D204.git'
            }
        }
        stage('build'){
            steps{
                dir('frontend/ecs-react'){
                    sh'''
                        npm install
                        CI='' npm run build
                    '''
                }
            }
        }
        stage('docker build'){
            steps{
                dir('frontend/ecs-react'){
                    script{
                        dockerImage=docker.build registry
                    }
                }
            }
        }
        stage('docker image push'){
            steps{
                script{
                    docker.withRegistry('', registryCredential){
                        dockerImage.push()
                    }
                }
            }
        }
        stage('ssh-server'){
            steps{
                sshagent(credentials: ['{ssh credential}']){
                    sh'''
                        ssh -o StrictHostKeyChecking=no ubuntu@{ip} "sudo docker run --name ecs_front -d -p 3000:80 {docker image}
                    '''
                }
            }
        }
    }
}
```

외부 API

- 소셜 로그인
  - KaKao : Oauth 기반 소셜 로그인 API 제공
  - https://developers.kakao.com