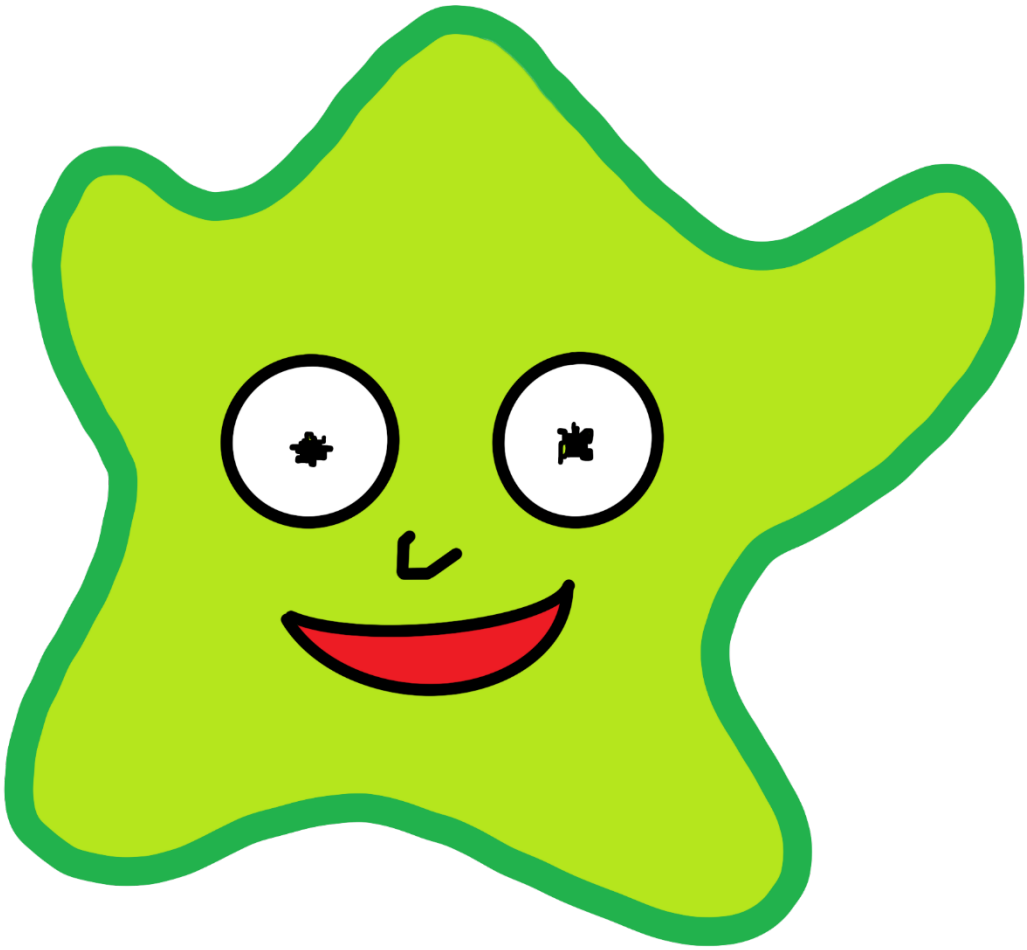

에듀 세포

포팅 매뉴얼

목차

1. 개발환경

A. 프로젝트 사용 도구

와이어프레임) Figma

이슈관리) Jira

형상관리) Gitlab

문서관리) Notion

소통) Mattermost, KakaTalk

B. 개발환경 및 기술 스택

개발 환경	기술 스택
Server	- AWS EC2 - OS: Ubuntu 22.04.1 LTS
IDE	- Backend: IntelliJ IDEA Ultimate 2021.2 - Frontend: Visual Studio Code 1.78.2
DevOps	- GitLab - Docker 23.0.5 - Docker Compose 2.17.3 - Nginx 1.23.4 - Jenkins 2.387.2
SSL	- letsencrypt - certbot 1.7.0
DataBase	- MySQL 8.0.31
Backend	- Java OpenJDK 11.0.17 - SpringBoot 2.7.11 - Gradle 7.6.1
Frontend	- Nodejs 18.14.2 - npm 9.5.0 - vite 4.3.0 - react 18.2.0 - recoil 0.7.7

2. EC2 세팅

A. Docker 설정

1. EC2에 Docker 및 Docker Compose 설치

```
sudo apt-get update
sudo apt-get install ca-certificates curl gnupg
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg
echo \
  "deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-
compose-plugin
```

B. Docker Compose 설정

1. /home/ubuntu 경로에 compose.yaml 파일 작성

```yaml
services: # 컨테이너 설정
  mysql:
    image: mysql:8.0.31 # 컨테이너에서 사용하는 base image 지정
    container_name: mysql
    restart: unless-stopped # 컨테이너 다운 시 재시작하라는 명령어
    volumes: # mount container's /var/lib/mysql volume to host's mysqldata vol
      - /home/ubuntu/mysql/databases:/var/lib/mysql # 상대경로로 지정
    environment: # 컨테이너 안의 환경변수 설정
      MYSQL_ROOT_PASSWORD: [redacted]
      MYSQL_DATABASE: edusetpo # dont touch db anymore!!!!!!!!!!!
    ports: # -p 옵션과 동일
      - 3306:3306

  jenkins:
    image: jenkins/jenkins:lts
    container_name: jenkins
    volumes:
      - /usr/bin/docker:/usr/bin/docker
      - /var/run/docker.sock:/var/run/docker.sock
      - /jenkins:/var/jenkins_home
    ports:
      - 9090:8080
    privileged: true
    user: root
    restart: unless-stopped

  nginx:
    image: nginx
    container_name: nginx
    ports:
      - 80:80
      - 443:443
    volumes:
      - /home/ubuntu/nginx/conf.d:/etc/nginx/conf.d
      - /etc/letsencrypt:/etc/letsencrypt
    restart: unless-stopped
```

i. Nginx 설정

1. /home/ubuntu 경로에 nginx 디렉토리 생성

2. /home/ubuntu/nginx 경로에 conf.d 디렉토리 생성

3. /home/ubuntu/nginx/conf.d 경로에 default.conf 파일 작성

```
server {
  listen 80;
  listen [::]:80;

  server_name edusetpo.com www.edusetpo.com k8a103.p.ssafy.io;
  server_tokens off;

  location / {
    return 301 https://$server_name$request_uri;
  }
}
server {
  listen 443 ssl;
  listen [::]:443 ssl;

  server_name edusetpo.com www.edusetpo.com k8a103.p.ssafy.io;
  server_tokens off;
  access_log off;
  ssl_certificate /etc/letsencrypt/live/k8a103.p.ssafy.io/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/k8a103.p.ssafy.io/privkey.pem;
  include /etc/letsencrypt/options-ssl-nginx.conf;
  ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;

  location / {
    proxy_pass http://k8a103.p.ssafy.io:3126/;
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-Host $server_name; proxy_set_header X-Real-IP
$remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme; proxy_set_header Upgrade
$http_upgrade;
    proxy_set_header Connection "upgrade";
    proxy_redirect off;
  }

    location /api/ {
      proxy_pass http://k8a103.p.ssafy.io:8080/;
      proxy_redirect off;
      proxy_set_header Host $host;
      proxy_set_header X-Real-IP $remote_addr;
      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
  }
}
```

ii. MySQL 설정

1. /home/ubuntu 경로에 mysql 디렉토리 생성

2. /home/Ubuntu/mysql 경로에 databases 디렉토리 생성

iii. SSL 인증서 발급

1. 인증서 발급을 위한 임시 컨테이너 run 후 삭제

```
docker run -it --rm --name certbot \
  -p 80:80 \
  -v '/etc/letsencrypt:/etc/letsencrypt' \
  -v '/var/lib/letsencrypt:/var/lib/letsencrypt' \
  certbot/certbot certonly -d 'edusetpo.com' -d 'www.edusetpo.com' -d 'k8a103.p.ssafy.io'
--standalone \
  --server https://acme-v02.api.letsencrypt.org/directory
```

iv. 도커 컨테이너 run

1. SSL 인증서 발급 후 MySQL, Nginx, Jenkins 컨테이너 run

```
sudo docker compose up -d
```

v. Jenkins 설정

1. edusetpo.com:9090 으로 접속하여 젠킨스 로그인 후 플러그인 설치 및 계정 생성
2. Gitlab 플러그인 설치
3. Gitlab 토큰 발급받아 권한 설정

3. Frontend 빌드 및 배포

1. 프로젝트의 /frontend 위치에 Dockerfile 작성

```
FROM nginx:stable-alpine
WORKDIR /app
RUN mkdir ./build
ADD ./dist ./build
RUN rm /etc/nginx/conf.d/default.conf
COPY ./front.conf /etc/nginx/conf.d
EXPOSE 3126
CMD ["nginx", "-g", "daemon off;"]
```

2. 프로젝트의 /frontend 위치에 Jenkinsfile 작성

```
pipeline {
  agent any

  tools {
    nodejs "NodeJS 18.14.2"
  }

  stages {
    stage('React Build') {
      steps {
        dir('frontend') {
          sh 'npm install'
          sh 'npm run build'
        }
      }
    }

    stage('Docker Build') {
      steps {
        dir('frontend') {
          sh 'docker build -t edusepo-front:latest .'
        }
      }
    }

    stage('Deploy') {
      steps{
        sh 'docker rm -f front'
        sh 'docker run -d --name front -p 3126:3126 -u root edusepo-front:latest'
      }
    }

    stage('Finish') {
      steps{
        sh 'docker images -qf dangling=true | xargs -I{} docker rmi {}'
      }
    }
  }
}
```

3. /jenkins/workspace/front/frontend 위치에 front.conf 작성

```
server {
  listen 3126;
  location / {
      root /app/build;
      index index.html;
      try_files $uri $uri/ /index.html;
  }
}
```

4. Jenkins Pipeline item 생성

5. Jenkins Webhook 설정. Push Event만 체크

6. Add webhook 에서 포트포워딩한 URL 입력 edusetpo.com:3126

7. Pipeline - Definition을 "Pipeline script form SCM"으로 설정

4. Backend 빌드 및 배포

1. 프로젝트의 /backend 위치에 Dockerfile 작성

```
FROM openjdk:11-jdk-slim
ARG JAR_FILE=build/libs/*.jar
COPY ${JAR_FILE} app.jar
EXPOSE 8080
ENTRYPOINT ["java", "-jar", "app.jar"]
```

2. 프로젝트의 /backend 위치에 Jenkinsfile 작성

```
pipeline {
  agent any

  stages {
    stage('Gradle Build') {
      steps {
        dir('backend/edusetpo') {
          sh 'chmod +x ./gradlew'
          sh './gradlew clean build -x test'
        }
      }
    }

    stage('Docker Build') {
      steps {
        dir('backend/edusetpo') {
          sh 'docker build -t edusepo-back:latest .'
        }
      }
    }

    stage('Deploy') {
      steps{
        sh 'docker rm -f back'
        sh 'docker run -d --name back -p 8080:8080 -u root -e TZ=Asia/Seoul edusepo-back:latest'
      }
    }

    stage('Finish') {
      steps{
        sh 'docker images -qf dangling=true | xargs -I{} docker rmi {}'
      }
    }
  }
}
```

3. /jenkins/workspace/back/backend/edusetpo/src/main 위치에 resources 디렉토리 생성

4. /jenkins/workspace/back/backend/edusetpo/src/main/resources 위치에 application.properties 작성

```
# Server
server.port=8080
server.servlet.encoding.charset=UTF-8
server.servlet.encoding.enabled=true
server.servler.encoding.force=true

# Spring
#datasource
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://k8a103.p.ssafy.io:3306/edusetpo?useSSL=false&
allowPublicKeyRetrieval=true&useUnicode=true&serverTimezone=Asia/Seoul
spring.datasource.username=root
spring.datasource.password=benditlike

# Spring
#jpa
spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.format_sql=true
spring.jpa.database=mysql

jwt.secret=VlwEyVBsYt9V7zq57TejMnVUyzblYcfPQye08f7MGVA9XkHN
```

5. Jenkins Pipeline item 생성

6. Jenkins Webhook 설정. Push Event만 체크

7. Add webhook 에서 포트포워딩한 URL 입력 edusetpo.com:8080

8. Pipeline - Definition을 "Pipeline script form SCM"으로 설정