프론트 배포

Created	@2022년 10월 31일 오전 10:06
Tags	

젠킨스 & 도커 설치

```
## apt를 이용하여 docker를 설치 할 예정이라 apt를 update합니다.
$ sudo apt update

## docker 설치에 필요한 패키지 들을 설치합니다.
$ sudo curl https://get.docker.com/ > dockerinstall && chmod 777 dockerinstall && ./dockerinstall

## 도커 컨테이너 생성 젠킨스
sudo docker run -d -p 9090:8080 -v /var/jenkins:/var/jenkins_home -v /var/run/docker.sock:/var/run/docker.sock --name jenkins -u root

## 젠킨스 컨테이너 접속
sudo docker exec -it jenkins bash

## 젠킨스 내부에 도커 설치
curl https://get.docker.com/ > dockerinstall && chmod 777 dockerinstall && ./dockerinstall
```

도커파일

```
FROM node as build-stage

WORKDIR /app

COPY package*.json ./


RUN npm install

COPY . .

RUN npm run build

# production stage
FROM nginx:stable-alpine as production-stage
COPY  ./default /etc/nginx//sites-available/default.conf

COPY --from=build-stage /app/dist /usr/share/nginx/html
EXPOSE 8080
CMD ["nginx", "-g", "daemon off;"]
```

젠킨스 파일

```
pipeline {
    agent any
    stages {
        stage("build"){
            steps{
                 script{
                     try {
                         sh 'docker stop frontend'
                         sh 'docker rm frontend'
                         sh 'docker rmi front-unit'

                     } catch (e) {
                         sh 'echo "fail remove test-images"'
                     }
                 }
                 sh 'cp /etc/nginx/sites-available/default ./Front/theme'
                 sh 'docker build -t front-unit -f Front/DockFile.e2e ./Front/theme'
                 sh 'docker run -d -p 8080:80 --name frontend front-unit'
             }
             post {
                 success{
                     sh 'echo unit 테스트 성공'
                 }
                 failure{
                     sh 'echo unit 테스트 실패'
```

```
                }
            }
        }
    }
}
```

Nginx

```
// Nginx 위치 경로 : /var/jenkins_home/workspace/front/Front/theme/default

sudo docker exit -it jenkins bash
// jenkins 도커 내부
nano /var/jenkins_home/workspace/front/Front/theme/default


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


        root /var/www/html;

        index index.html index.htm index.nginx-debian.html;

        server_name _;

        location / {
                alias /usr/share/nginx/html/;
                try_files $uri $uri/ /index.html;

        }

}

server {
  listen 80;
  server_name k7c2031.p.ssafy.io;
  return 301 https://k7c2031.p.ssafy.io$request_uri;

}
server {
  listen 443 ssl http2;
  server_name k7c2031.p.ssafy.io;
  client_max_body_size 5M;
  # ssl 인증서 적용하기
  ssl_certificate /etc/letsencrypt/live/k7c2031.p.ssafy.io/fullchain.pem
  ssl_certificate_key /etc/letsencrypt/live/k7c2031.p.ssafy.io/privkey.pem

  location / {
    proxy_pass http://k7c2031.p.ssafy.io:8080;
    proxy_set_header Host $http_host;
    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 $scheme;
  }
}
server {
    if ($host = k7c2031.p.ssafy.io) {
        return 301 https://$host$request_uri;
    }


  listen 80;
  server_name example.com;
    return 404;


}
```

SSL

```
// 도커 내부
apt-get update
apt-get install software-properties-common
add-apt-repository universe
add-apt-repository ppa:certbot/certbot
```

```
apt-get install python3-certbot-nginx
certbot certonly --nginx -d k7c2031.p.ssafy.io
apt-get update

/etc/letsencrypt/live/k7c2031.p.ssafy.io/fullchain.pem

/etc/letsencrypt/live/k7c2031.p.ssafy.io/privkey.pem
```

```
server {
  listen 80;

  location / {
    root /usr/share/nginx/html;

    index index.html index.htm;

    try_files $uri $uri/ /index.html;

    gzip_static always;
  }
}

server {
  listen 8080;

  location / {
    root /usr/share/nginx/html;

    index index.html index.htm;

    try_files $uri $uri/ /index.html;

    gzip_static always;
  }
}

server {
  listen 443;

  location / {
    root /usr/share/nginx/html;

    index index.html index.htm;

    try_files $uri $uri/ /index.html;


    gzip_static always;
  }
}
```