

빌드 및 배포

빌드

1. back- end

App

Spring Build to Dockerfile -> 도커파일을 이용해 jar 파일 이미지 생성

```
FROM openjdk:8-jdk-alpine
RUN apk add --no-cache tzdata
ENV TZ Asia/Seoul
COPY build/libs/bloom-0.0.1-SNAPSHOT.jar app.jar
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

Web

Spring Build to Dockerfile -> 도커파일을 이용해 jar 파일 이미지 생성

```
FROM openjdk:8-jdk-alpine
RUN apk add --no-cache tzdata
ENV TZ Asia/Seoul
COPY build/libs/bloom-0.0.1-SNAPSHOT.jar app.jar
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

2. front-end

App

`npm install`을 통해 node-modules 설치

`npm run android`를 통해 실행

Web

`npm install`을 통해 node-modules 설치

`npm start`를 통해 실행

📄 배포

🔗 설치

java 8 설치

android studio(2021.2.1) 설치

App SDK : Android SDK 30 설치

mysql(8.0.0) 설치

node(16.13.12)와 npm(8.4.0) 설치

docker(20.10.12) 설치

nginx(1.18.0) 설치

🔗 DataBase

[Spring]

application.properties -> mysql 로그인 정보 有

🔗 back-end

생성해둔 Dockerfile을 이용

- `docker build ./ -t back:1.0` 를 통해 docker 이미지 파일 생성
- `docker run -i -t -d -p 3000:8080 back:1.0` 를 통해 도커 컨테이너 실행
- localhost:3000에 접속해 실행을 확인한다.

🔗 front-end

App [Android]

- `outputs/apk/piona.apk` 설치 및 실행

Web

- `npm install` 을 이용해 node-modules와 package-lock.json 생성
- `npm run build`를 실행 빌드 파일 생성

- nginx conf 파일의 server-name 수정
- `systemctl start nginx`를 이용 nginx 실행
- https 적용을 위해 certbot을 이용해 ssl 인증서 발급
- 다시 nginx conf 파일을 수정

```
server {
    root [front 빌드파일 경로];

    index index.html;

    server_name [도메인 주소];

    location / {
        try_files $uri $uri/ /index.html =404;
    }

    listen [::]:443 ssl ipv6only=on; # managed by Certbot
    listen 443 ssl; # managed by Certbot

    ssl_certificate /etc/letsencrypt/live/[도메인 주소]/fullchain.pem; #
managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/[도메인 주소]/privkey.pem; #
managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by
Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by
Certbot

    location /api {
        proxy_pass http://[도메인주소]:3000;
    }
}
server {
    if ($host = [도메인주소]) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    listen 80;
    listen [::]:80;

    server_name [도메인주소];
    return 404; # managed by Certbot
}
```

위와 같이 nginx conf 수정

`systemctl reload nginx` 명령어 실행 nginx를 reload하여 https 적용과 실행을 확인한다.



AWS S3

https://docs.aws.amazon.com/s3/index.html?nc2=h_qi_doc_s3

네이버 맵

<https://www.ncloud.com/product/applicationService/maps>

네이버 검색

<https://developers.naver.com/docs/serviceapi/search/local/local.md#%EC%A7%80%EC%97%AD>

부트페이

<https://docs.bootpay.co.kr/>

FCM

<https://firebase.google.com/products/cloud-messaging?authuser=0&hl=ko>

도로명 주소

<https://www.juso.go.kr/addrlink/devAddrLinkRequestGuide.do?menu=roadApi>

사업자 번호 조회

<https://www.data.go.kr/tcs/dss/selectApiDataDetailView.do?publicDataPk=15081808#/%EC%82%AC%EC%97%85%EC%9E%90%EB%93%B1%EB%A1%9D%20%EC%83%81%ED%83%9C%EC%A1%B0%ED%9A%8C%20API/status>

Coolsms

https://developer.coolsms.co.kr/SDK_Java_Getting_Started_ko