Stackers 포팅 메뉴얼

1. 프로젝트 개요

💡 **한줄 소개** : 숏폼을 활용한 합주 소셜 네트워크 서비스

서비스 명 : **Stackers**

2. 프로젝트 사용 도구

이슈 관리 : JIRA

형상 관리 : Gitlab

커뮤니케이션 : Notion, Mattermost, Discord, KakaoTalk

디자인 : Figma

UCC : movavi

Deploy : Docker

3. 개발 환경

VSCode : 1.74.3

npm : 8.19.2

React : 18.2.0 (자동으로 최신버전 다운)

Node.js : 16.18.0

IntelliJ : 2022.3.1

JVM : Zulu-17

Spring boot : 3.0.2

DB : MariaDB 10.10.2

서버 : Ubuntu 20.04 LTS

nginx : 1.23.3

ffmpeg : 4.3.5

Swagger : 2.0.2

S3 : 2.2.6jpa

4. 외부 서비스

AWS EC2

S3 : S3Config에 해당 설정 내용 있음

ffmpeg : ffmpeg 프로그램 다운로드

Redis : RedisConfig에 해당 설정 내용 있음

(2) 빌드

1. 환경변수 형태

application.yml

#마리아 DB(배포)

```
datasource:
    driverClassName: com.mysql.cj.jdbc.Driver
    password: root
    username: root
    url: jdbc:mysql://mariadb:3306/stackers?useSSL=false
```

#REDIS

```
redis:
    pool:
      max-active: 10
      max-idle: 10
      min-idle: 2
  data:
    redis:
      port: 6379
      host: redis
      password: "1234"
```

#mail

```
mail:
    host: smtp.gmail.com
    port: 587
    username: www.stackers.site
    password: pmldwhwltgkldcgb
    properties:
      mail:
        smtp:
          socketFactory.class: javax.net.ssl.SSLSocketFactory
          auth: true
```

```
            starttls:
              enable: true
```

#Swagger

```
springdoc:
  api-docs:
    enabled: true
  swagger-ui:
    path: /swagger-ui.html
    disable-swagger-default-url: true
    display-request-duration: true
    tags-sorter: alpha
    operations-sorter: alpha
    doc-expansion: none
    syntax-highlight:
      theme: nord
    urls-primary-name: TEST API
    persist-authorization: true
    query-config-enabled: true
  pre-loading-enabled: true
  packages-to-scan: com.ssafy.stackers
```

#S3

```
cloud:
  aws:
    s3:
      bucket: stackers.bucket
    region:
      static: ap-northeast-2
      auto: false
    stack:
      auto: false
    credentials:
      access-key: ${STORAGE_PUBLIC_KEY}
      secret-key: ${STORAGE_PRIVATE_KEY}
```

docker-compose.yml

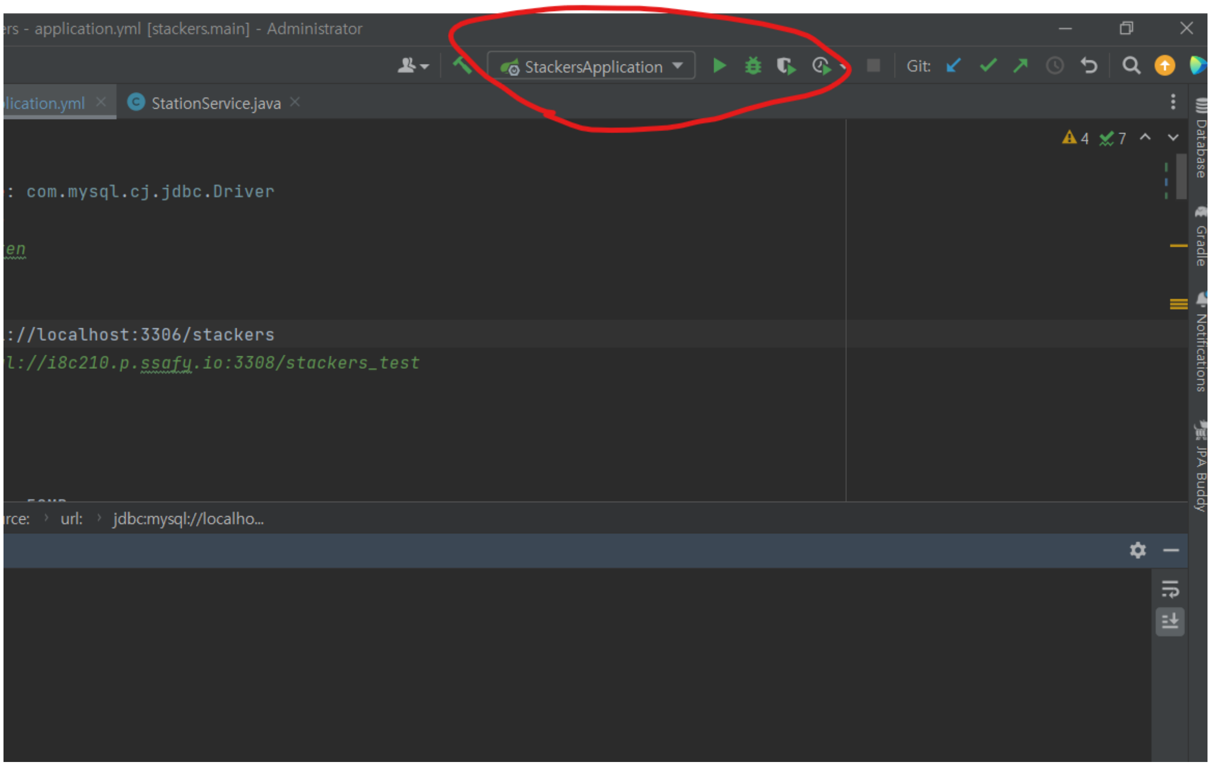
```
version: "3.7"

services:
  redis:
    image: redis
    container_name: redis
    ports:
      - 6379:6379
    networks:
      - stackers
    restart: always
  mariadb:
    container_name: mariadb
    build:
      context: ./mariadb
      dockerfile: Dockerfile
    ports:
      - 3306:3306
    environment:
      TZ: Asia/Seoul
    networks:
      - stackers
    restart: always
  backend:
    container_name: backend
    build:
      context: ./backend/
      dockerfile: Dockerfile
    ports:
      - 5000:5000
    depends_on:
      - redis
      - mariadb
    networks:
      - stackers
  frontend:
    container_name: frontend
    build:
      context: ./frontend
      dockerfile: Dockerfile
    ports:
      - 3000:3000
    networks:
      - stackers
  nginx:
    container_name: nginx
    restart: always
    build:
      context: ./nginx
      dockerfile: Dockerfile
    ports:
      - 80:80
      - 443:443
    depends_on:
      - frontend
      - backend
    networks:
      - stackers
    volumes:
      - ../certbot/conf:/etc/letsencrypt
      - ../certbot/www:/var/www/certbot
    command: '/bin/sh -c ''while :; do sleep 6h & wait $${!}; nginx -s reload; done & nginx -g "daemon off;"'''
  certbot:
    image: certbot/certbot
    restart: always
    volumes:
      - ../certbot/conf:/etc/letsencrypt
      - ../certbot/www:/var/www/certbot
    entrypoint: "/bin/sh -c 'trap exit TERM; while :; do certbot renew; sleep 12h & wait $${!}; done;'"

networks:
  stackers:
    driver: bridge
```

2. 빌드하기: 로컬 빌드

1) 프론트엔드 : 터미널 명령어 실행

```
npm install
npm start
```

2) 백엔드 - spring boot



```
: com.mysql.cj.jdbc.Driver

en

://localhost:3306/stackers
l://i8c210.p.ssafy.io:3308/stackers_test
```

rce: › url: › jdbc:mysql://localho...

Gradle StackersApplication 실행

3. 배포하기

1) 프론트엔드 배포

```
FROM node:16.18.0-alpine # 노드 버전

RUN mkdir /app
WORKDIR /app

ENV PATH /app/node_modules/.bin:$PATH

COPY . /app
RUN npm install

CMD ["npm", "start"]
```

2) 데이터베이스 배포

```
FROM mariadb # 이미지

ENV MYSQL_ROOT_PASSWORD root # root 계정 비밀번호
```

```
COPY ./config/setDB.sql /docker-entrypoint-initdb.d # 도커 시작할 때 계정, 디비 생성

# MariaDB Config Setting (table 소문자, 한국 시간, 한글 깨짐 수정 등)
RUN echo lower_case_table_names=1 >> /etc/mysql/conf.d/docker.cnf
RUN echo default-time-zone='+9:00' >> /etc/mysql/conf.d/docker.cnf
RUN echo collation-server = utf8mb4_unicode_ci >> /etc/mysql/conf.d/docker.cnf
RUN echo collation-server = utf8mb4_0900_ai_ci >> /etc/mysql/conf.d/docker.cnf
RUN echo character-set-server = utf8mb4 >> /etc/mysql/conf.d/docker.cnf
RUN echo skip-character-set-client-handshake >> /etc/mysql/conf.d/docker.cnf
```

```
CREATE DATABASE stackers;

create user 'ssafy'@'%' identified by 'ssafy';
grant all privileges on DB_CREATEDB.* to 'ssafy'@'%' identified by 'ssafy';
flush privileges;
```

3) 백엔드 배포

```
FROM gradle:7.4-jdk17-alpine as builder
WORKDIR /build

# 그래들 파일이 변경되었을 때만 새롭게 의존패키지 다운로드 받게함.
COPY build.gradle settings.gradle /build/
RUN gradle build -x test --parallel --continue  > /dev/null 2>&1 || true

# 빌더 이미지에서 애플리케이션 빌드
COPY . /build
RUN gradle clean build -x test --parallel

# APP
FROM openjdk:17.0-slim
WORKDIR /build

# 빌더 이미지에서 jar 파일만 복사
COPY --from=builder /build/build/libs/stackers-0.0.1-SNAPSHOT.jar .

EXPOSE 5000

# ffmpeg
RUN apt update
RUN apt install -y ffmpeg

# root 대신 nobody 권한으로 실행
ENTRYPOINT [                                                \
    "java",                                                 \
    "-jar",                                                 \
    "-Djava.security.egd=file:/dev/./urandom",              \
    "-Dsun.net.inetaddr.ttl=0",                             \
    "stackers-0.0.1-SNAPSHOT.jar"              \
]
```

4) nginx 배포

```
server {
    listen 80;
    server_name i8c210.p.ssafy.io;

    location / {
        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_pass http://client;
    }

    location /api {
        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_pass http://back/api;
    }
}
```

```
#!bin/bash
mkdir /etc/nginx/sites-available
mkdir /etc/nginx/sites-enabled

mv /etc/nginx/test.conf /etc/nginx/sites-available/test.conf
ln -s /etc/nginx/sites-available/test.conf /etc/nginx/sites-enabled/test.conf

mv /etc/nginx/default.conf /etc/nginx/conf.d/default.conf
```

설정파일

```
upstream client {
    server frontend:3000; # front: reactjs container name
}

upstream back {
    server backend:5000; # back: springboot container name
}

server {
    listen 80; # http
    server_name stackers.site www.stackers.site;
    server_tokens off;

    location /.well-known/acme-challenge/ {
        root /var/www/certbot;
    }

    location / {
        return 301 https://$host$request_uri; # https 리다이렉트
    }

}
server {
    listen 443 ssl; # https
    server_name www.stackers.site;
    server_tokens off;

    location / {
        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_pass http://client;
    }

    location /api {
        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_pass http://back/api;
    }

    # 인증서
    ssl_certificate /etc/letsencrypt/live/www.stackers.site/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/www.stackers.site/privkey.pem;

    include /etc/letsencrypt/options-ssl-nginx.conf;
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;
}
```

```
user  nginx;
worker_processes  auto;

error_log  /var/log/nginx/error.log notice;
pid        /var/run/nginx.pid;


events {
    worker_connections  1024;
}


http {
    client_max_body_size 50M;
    include       /etc/nginx/mime.types;
    default_type  application/octet-stream;

    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    access_log  /var/log/nginx/access.log  main;

    sendfile        on;
    #tcp_nopush     on;

    keepalive_timeout  65;

    #gzip  on;

    include /etc/nginx/conf.d/*.conf;
    include /etc/nginx/sites-enabled/*.conf; # 도메인 설정 파일 위치
    server_names_hash_bucket_size 64;
}
```

배포 명령어 정리

`docker-compose`

docker-compose 옵션 기본 명령어 (tistory.com)

정리하기,,

EC2 세팅 : Docker, ffmpeg, mariadb

AWS S3 Bucket

```
cloud:
  aws:
    s3:
      bucket: stackers.bucket
    region:
      static: ap-northeast-2
```

```
        auto: false
      stack:
        auto: false
      credentials:
        access-key: ${STORAGE_PUBLIC_KEY}
        secret-key: ${STORAGE_PRIVATE_KEY}
```

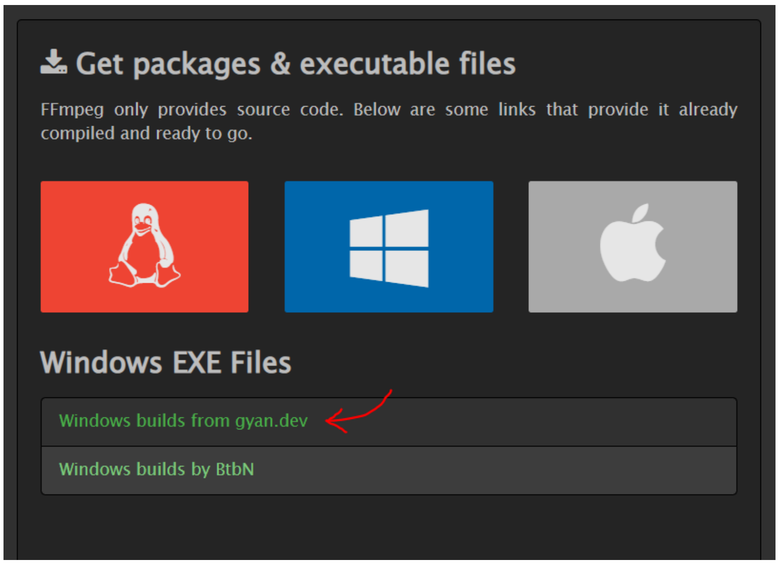
권한 설정

```
{
    "Version": "2012-10-17",
    "Id": "Policy1675837105268",
    "Statement": [
        {
            "Sid": "Stmt1675837104165",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::stackers.bucket"
        }
    ]
}
```

4. ffmpeg

로컬(윈도우)

ffmpeg 공식 홈페이지(https://ffmpeg.org/download.html)에서 프로그램 다운로드

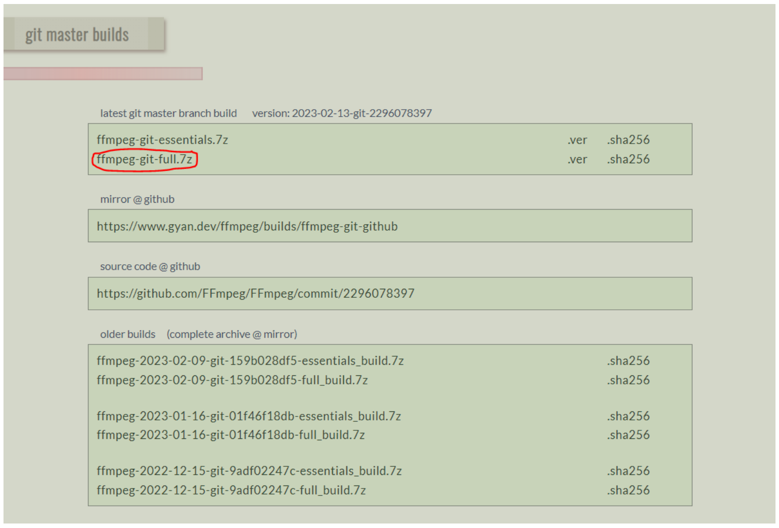

Get packages & executable files

FFmpeg only provides source code. Below are some links that provide it already compiled and ready to go.

Windows EXE Files

Windows builds from gyan.dev

Windows builds by BtbN


git master builds

latest git master branch build   version: 2023-02-13-git-2296078397

ffmpeg-git-essentials.7z	.ver	.sha256
ffmpeg-git-full.7z	.ver	.sha256

mirror @ github

https://www.gyan.dev/ffmpeg/builds/ffmpeg-git-github

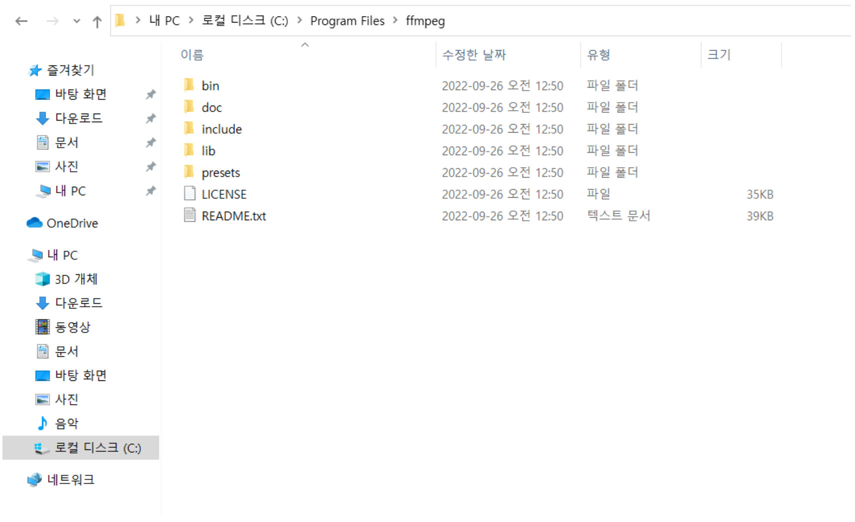
source code @ github

https://github.com/FFmpeg/FFmpeg/commit/2296078397

older builds   (complete archive @ mirror)

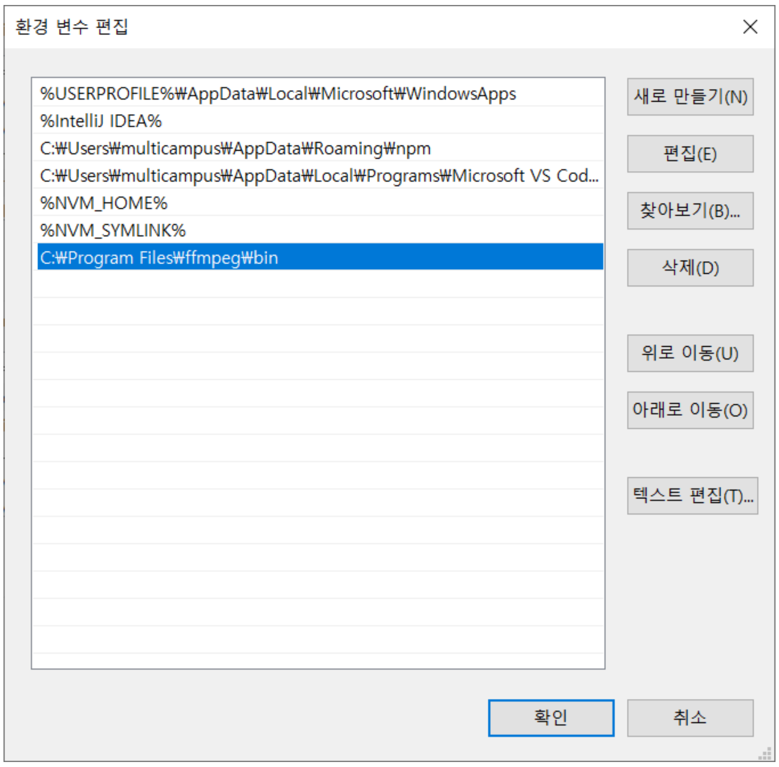
ffmpeg-2023-02-09-git-159b028df5-essentials_build.7z	.sha256
ffmpeg-2023-02-09-git-159b028df5-full_build.7z	.sha256
ffmpeg-2023-01-16-git-01f46f18db-essentials_build.7z	.sha256
ffmpeg-2023-01-16-git-01f46f18db-full_build.7z	.sha256
ffmpeg-2022-12-15-git-9adf02247c-essentials_build.7z	.sha256
ffmpeg-2022-12-15-git-9adf02247c-full_build.7z	.sha256

다운로드 및 압축 해제



이름	수정한 날짜	유형	크기
bin	2022-09-26 오전 12:50	파일 폴더	
doc	2022-09-26 오전 12:50	파일 폴더	
include	2022-09-26 오전 12:50	파일 폴더	
lib	2022-09-26 오전 12:50	파일 폴더	
presets	2022-09-26 오전 12:50	파일 폴더	
LICENSE	2022-09-26 오전 12:50	파일	35KB
README.txt	2022-09-26 오전 12:50	텍스트 문서	39KB

C:/Program Files/ffmpeg 경로로 이동



```
%USERPROFILE%₩AppData₩Local₩Microsoft₩WindowsApps
%IntelliJ IDEA%
C:₩Users₩multicampus₩AppData₩Roaming₩npm
C:₩Users₩multicampus₩AppData₩Local₩Programs₩Microsoft VS Cod...
%NVM_HOME%
%NVM_SYMLINK%
C:₩Program Files₩ffmpeg₩bin
```

bin 폴더 경로 복사해서 환경 변수로 추가