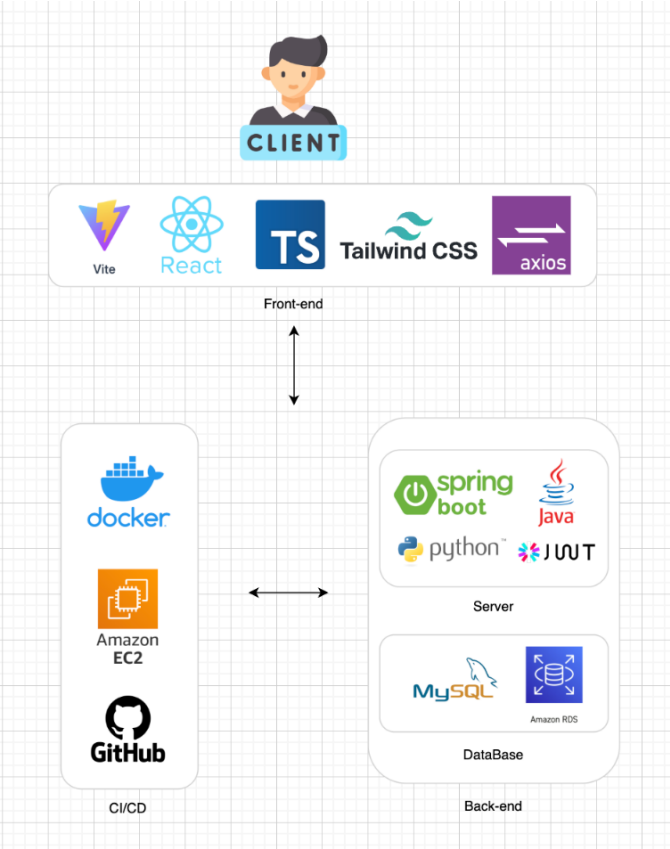
오픈소스프로젝트 제품 구성, 배포 및 운영 자료

제 품 명	감정 분석 기반 맞춤형 음악 생성 및 정서 케어 시스템

프로젝트 제품 구성



- Front-end : Vite + React 기반 SPA, TypeScript, Tailwind CSS, Axios
- Back-end : Spring Boot (Java), AI 모델 연동 (Python), JWT 인증
- DataBase : MySQL (AWS RDS)
- CI/CD: Docker + AWS EC2 + GitHub

본 프로젝트는 Vite와 React 기반의 프론트엔드, Spring Boot (Java) 기반의 백엔드, AI 모델 (Python), MySQL 데이터베이스로 구성되어 있으며, Docker 컨테이너를 통해 배포되었습니다. 모든 구성 요소는 AWS EC2 인스턴스 내에서 실행됩니다.

프로젝트 제품 배포 방법

본 프로젝트는 AWS EC2 인스턴스를 기반으로 프론트엔드와 백엔드를 각각 Docker 컨테이너로 배포하여 실행하였습니다. 전체 배포 절차는 다음과 같습니다.

1) 실행 환경 구성

HTTP 허용: 도메인 구매 및 SSL 인증서 설정이 어려운 상황에서 시연을 위해
chrome://flags/#unsafely-treat-insecure-origin-as-secure
에 서비스 IP 주소를 등록하여 HTTP를 임시로 허용.

2) EC2 접속

```
ssh -i <pem파일 경로> ec2-user@<IP 주소>
```

3) 백엔드(Spring Boot) 빌드 및 배포 절차
로컬 개발 환경에서 백엔드 빌드 및 이미지 생성

```
./gradlew build -x test
docker build --platform=linux/amd64 -t <이미지명> .
docker save <이미지명> > <이미지명>.tar
scp -i <pem파일 경로> <이미지명>.tar ec2-user@<IP 주소>:~
```

EC2에서 이미지 로드 및 컨테이너 생성, 실행

```
docker load < <이미지명>.tar
docker run -d -p 8080:8080 <env 파일 연결> --name <컨테이너명> <이미지명>
```

컨테이너 생성 이후 실행 : `docker start <컨테이너명>`

4) 프론트엔드(Vite + React) 빌드 및 배포 절차
로컬에서 프론트 빌드 및 이미지 생성

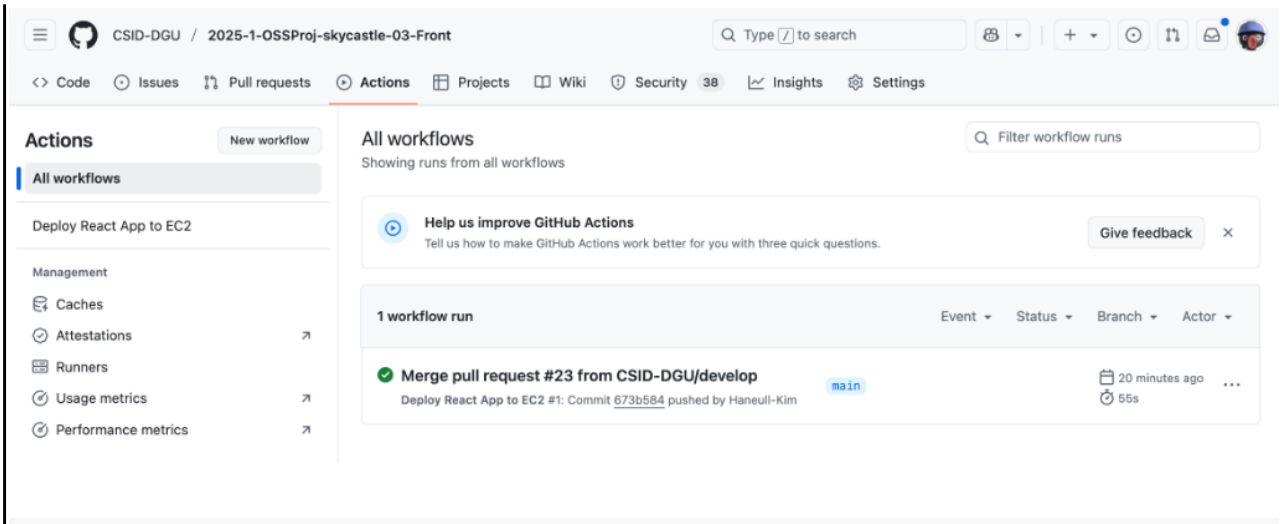
```
npm run build
docker build --platform=linux/amd64 -t <이미지명> .
docker save <이미지명> > <이미지명>.tar
scp -i <pem파일 경로> <이미지명>.tar ec2-user@<IP 주소>:~
```

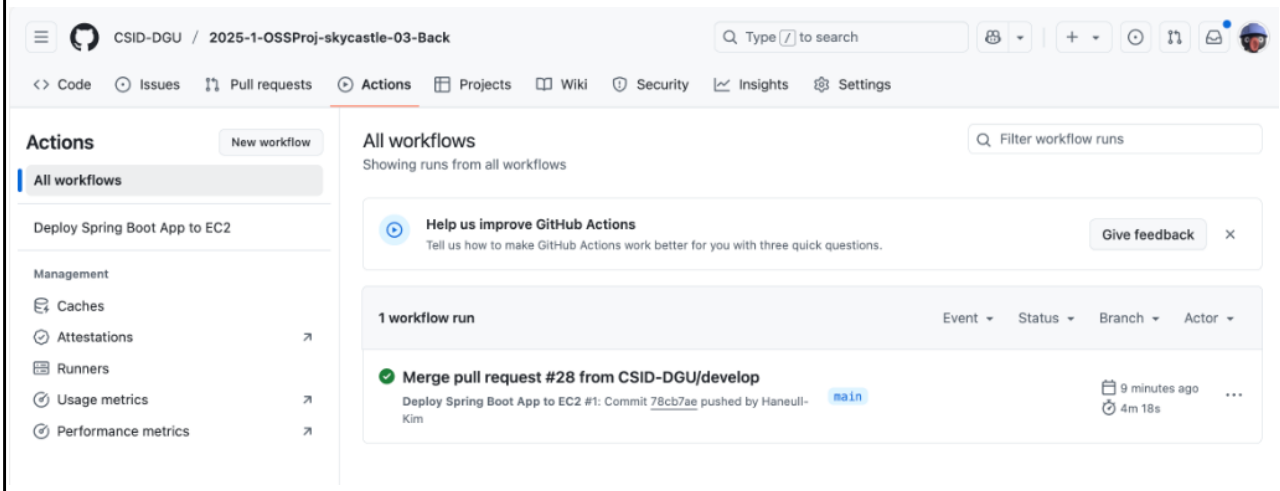
EC2에서 이미지 로드 및 컨테이너 생성 및 실행

```
docker load < <이미지명>.tar
docker run -d -p 80:80 --name <컨테이너명> <이미지명>
```

컨테이너 생성 이후 실행 : `docker start <컨테이너명>`

5) Git Hub Actions로 배포 자동화
위의 내용을 .github/workflows/deploy-react.yml & deploy-spring.yml에 작성





변경 사항 발생 (main에 push) 할 경우 자동으로 Actions에서 빌드 및 배포

6) 배포 결과

접속 경로 : http://<IP주소>

프론트엔드는 80 포트를, 백엔드는 8080 포트를 사용하여 서비스 됩니다.

React → Spring Boot → (AI 모델) → MySQL 흐름으로 통신이 이루어집니다.

프로젝트 제품 운영 방법

본 프로젝트는 AWS EC2 인스턴스에서 프론트엔드와 백엔드 컨테이너가 실행되며, 브라우저를 통해 즉시 접속 가능한 상태로 운영됩니다.

1) 환경 구성

- 접속 URL:

  프론트엔드: http://<IP주소>

  백엔드: http://<IP주소>:8080 (option)

- 접속 환경:

  chrome://flags/#unsafely-treat-insecure-origin-as-secure

설정에서 http://<IP주소>, http://<IP주소>:8080 등록 필요
HTTPS 미적용 상태이므로 위 설정 없이는 일부 기능(마이크 입력 등) 제한

- 서버 조건:
EC2 인스턴스는 수동으로 실행되어야 하며, 컨테이너는 항상 실행 대기 상태
AI 기능을 위한 인증키는 EC2에 비공개 파일로 존재

2) 운영 절차
과금 방지를 위해 컨테이너는 EC2 인스턴스 켜질때 자동 실행되지 않음.
다음 명령으로 컨테이너를 재실행 함 : docker start <컨테이너명>

3) 전체 시스템 흐름
- 사용자가 브라우저에 접속
- JWT 기반 로그인 후 상담 및 감정 기반 음악 생성 기능 사용
- 백엔드는 사용자 요청을 수신하고, AI 모델을 실행하여 요청 처리
- 데이터는 DB에 저장되고, 클라이언트에 반영됨