



음악 검색 및 플레이리스트 관리 웹 서비스

React

TypeScript

Flask

MySQL

Spotify API

Listify 2 Team

이성훈 · 정혜원 · 조우성 · 최두석 · 김기열

2025.12.22

목차

01 프로젝트 동기 및 목적

02 기술 스택 (Tech Stack)

03 팀원 및 역할

04 기능 요구사항

05 비기능 요구사항

06 프로젝트 파일 구조

07 데이터베이스 설계 (ERD)

08 API 설계

09 UI/UX 화면 설계

10 시나리오

11 결론 및 프로젝트 요약

12 향후 계획

13 팀원 소감

14 Q&A

1. 프로젝트 동기 및 목적

! 프로젝트 동기

- 최신 웹 기술과 외부 API 연동 경험을 쌓기 위해
- 실무에 가까운 웹 서비스 개발 과정을 직접 경험하기 위해
- 음악에 관심이 있어 관련 서비스를 직접 구현하기 위해

🎯 Listify의 목표

- ✓ Spotify API 연동을 통한 전 세계 실시간 음악 데이터 제공
- ✓ 사용자별 **나만의 플레이리스트** 관리 시스템 구축
- ✓ **관리자 공지사항** 기능을 통한 서비스 소통 및 운영 강화

2. 기술 스택 (Tech Stack)

Frontend



React



TypeScript



Vite



tailwind

Backend



Flask



Python

Database

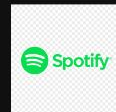


MySQL



DBeaver

External API



Spotify API

DevOps



Docker

3. 팀원 및 역할

5



이성훈

PM / Frontend + backend

- 프로젝트 전체 관리
- 사용자 프로필 조회/수정/삭제
기능 구현
- Docker 배포



정혜원

PL / Frontend + backend

- 공지사항 작성/수정/삭제/전체 리스트
조회/상세 조회 기능 구현
- ppt 제작



조우성

Frontend + backend

- 플레이리스트
생성/수정/삭제/조회 기능 구현
- 음악 재생 기능 구현



최두석

Frontend + backend

- 음악 저장 검색/리스트/카테고리별
조회 기능



김기열

Frontend + backend

- 회원가입/로그인 기능 구현

4. 기능 요구사항



사용자 관리

- ✓ 이메일 기반 회원가입 및 로그인
- ✓ JWT 토큰 기반 보안 인증 시스템
- ✓ 사용자 프로필 조회 및 정보 수정



음악 및 플레이리스트

- ✓ Spotify API 연동 실시간 음악 검색
- ✓ 나만의 플레이리스트 생성 및 관리
- ✓ 플레이리스트 내 곡 추가/삭제 기능



관리자 기능

- ✓ 공지사항 작성, 수정, 삭제 (CRUD)

5. 비기능 요구사항



보안

- ✓ JWT토큰 기반 인증 및 인가
- ✓ 비밀번호 암호화 저장
- ✓ 관리자/사용자 권한 분리



성능

- ✓ 빠른 응답 속도(실시간 음악 검색)
- ✓ 외부 API(Spotify)연동 안정성



사용성

- ✓ 반응형 UI(Tailwind CSS)로 다양한 디바이스 지원
- ✓ 직관적이고 편리한 사용자 경험 (UX)

6. 프로젝트 파일 구조

Backend (Flask)

backend/

└─ app.py	# Flask 메인 앱
└─ config.py	# 환경설정
└─ db.py	# DB 연결 및 설정
└─ requirements.txt	# 의존성 목록
└─ controllers/	# API 엔드포인트
└─ middleware/	# 인증 미들웨어
└─ model/	# DB 모델 및 쿼리
└─ routes/	# 라우트 분리
└─ services/	# 비즈니스 로직
└─ test/	# 백엔드 테스트 코드

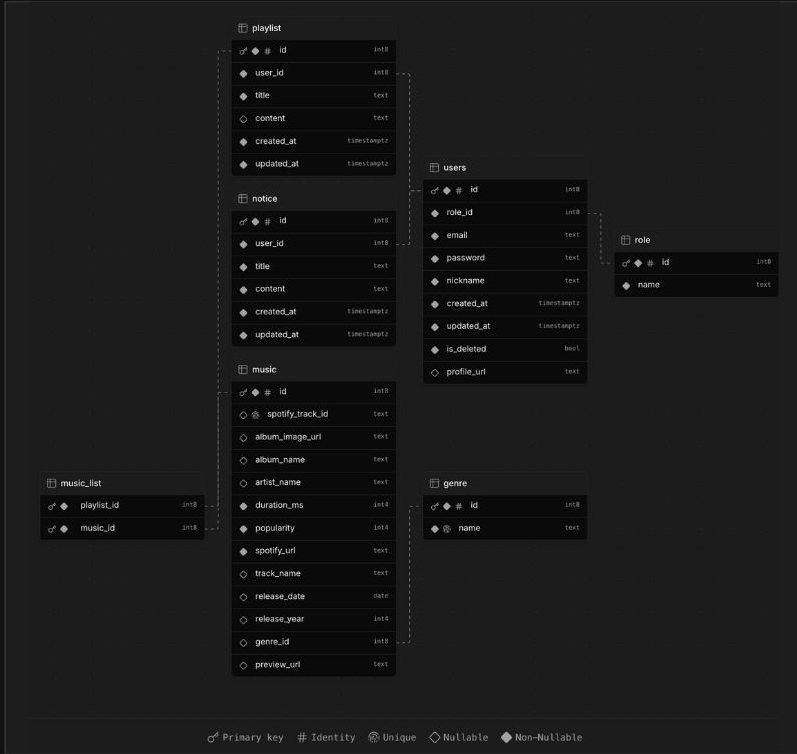
Frontend (React)

frontend/

└─ App.tsx	# 메인 앱 컴포넌트
└─ index.tsx	# 진입점
└─ constants.ts	# 상수/설정
└─ types.ts	# 타입 정의
└─ vite.config.ts	# Vite 설정
└─ package.json	# 의존성 목록
└─ components/	# UI 컴포넌트
└─ hooks/	# 커스텀 훅
└─ pages/	# 주요 페이지
└─ services/	# API 통신
└─ README.md	# 프론트엔드 설명서

7. 데이터베이스 설계 (ERD)

9



테이블명	주요 필드 및 설명
role	role_no, name 사용자 권한(관리자/일반 사용자) 관리
user	user_no, role_no, email, password, nickname, created_at, updated_at, profile_url, is_deleted 회원 정보 및 권한 관리
genre	genre_no, name 음악 장르 정보
music	music_no, spotify_track_id, album_image_url, album_name, artist_name, duration_ms, popularity, tsspotify_url, track_name, release_date, release_year, genre_no, preview_url 음악(곡) 정보, Spotify 연동
playlist	playlist_no, user_no, title, content, created_at, updated_at 사용자별 플레이리스트
music_list	playlist_no, music_no 플레이리스트와 음악의 다대다 관계 매핑
notice	notice_no, user_no, title, content, created_at, updated_at 공지사항(관리자 작성)

8. API 설계

10

Auth & User

메서드	엔드포인트	설명
POST	/api/auth/login	로그인
POST	/api/auth/register	회원가입
GET	/api/user/profile	프로필 조회
PUT	/api/user/update	정보 수정

Music & Playlist

메서드	엔드포인트	설명
GET	/api/music/search	음악 검색
POST	/api/playlist/create	플레이리스트 생성
GET	/api/playlist/:id	플레이리스트 상세 조회
DELETE	/api/playlist/:id	플레이리스트 삭제

Notice

메서드	엔드포인트	설명
GET	/api/notice	공지사항 목록 조회
GET	/api/notice/:id	공지사항 상세 조회
POST	/api/notice	공지사항 작성 (관리자)
PUT	/api/notice/:id	공지사항 수정 (관리자)
DELETE	/api/notice/:id	공지사항 삭제 (관리자)

9. UI/UX 화면 설계 개요

01

로그인 / 회원가입

JWT 기반 보안 인증 및
사용자 계정 관리

02

메인 (음악 리스트)

Spotify API 연동 실시간
음악 검색 및 탐색

03

플레이리스트 관리

플레이리스트 CRUD

04

공지사항 (관리자)

서비스 소식 전달 및
관리자 전용 게시판

05

프로필 / 설정

개인 정보 관리



9-1. UI/UX 화면 설계

회원가입

이메일

example@email.com

닉네임

닉네임 (최대 30자)

비밀번호

비밀번호 (6자 이상)

영문, 숫자, 특수문자 중 2가지 이상 포함

비밀번호 확인

비밀번호 확인

회원가입

이미 계정이 있으신가요? [로그인](#)

로그인

이메일

example@email.com

비밀번호

비밀번호

로그인

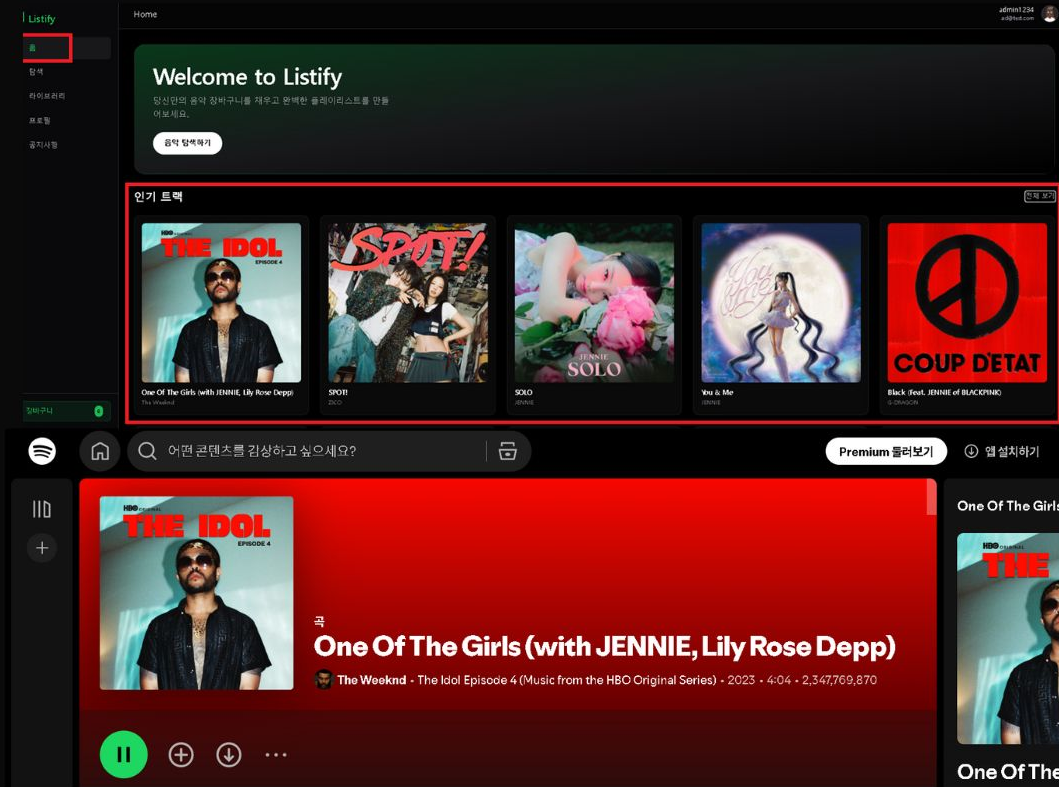
계정이 없으신가요? [회원가입](#)

로그인 / 회원가입

-사용자는 이메일과 비밀번호로 간편하게 회원가입 및 로그인을 진행

-JWT(JSON Web Token) 인증 방식으로 세션 관리의 효율성과 보안을 강화

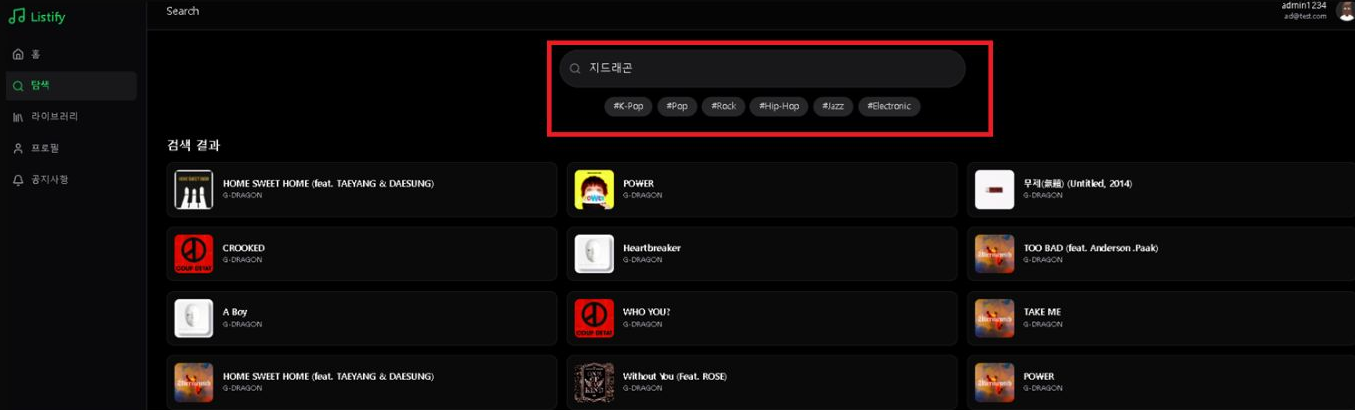
9-2. UI/UX 화면 설계



메인 (음악 리스트)

- Spotify API를 통해 전 세계 실시간 음악 데이터를 받아와 리스트로 제공
- 인기 Top 50 음악 리스트 제공
- 음악의 spotify_url(Spotify 곡 링크)이 새 브라우저 탭으로 이동해서 웹 플레이어로 열림

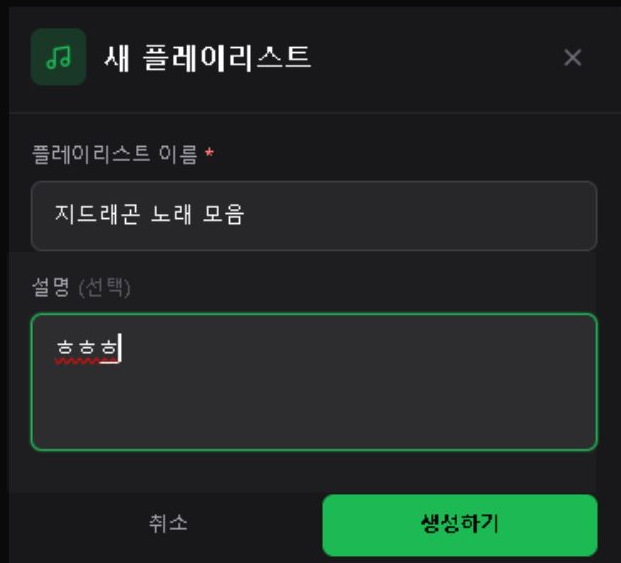
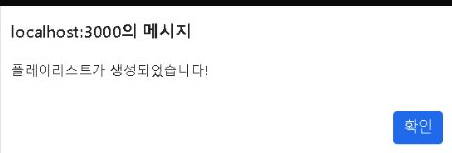
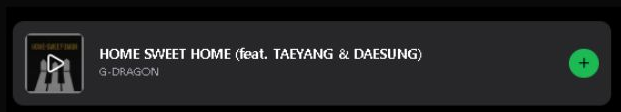
9-3. UI/UX 화면 설계



탐색 탭

- 가수명을 입력하면 해당 가수의 음악 리스트 추출
- 장르명으로 카테고리 검색 가능 (#장르)

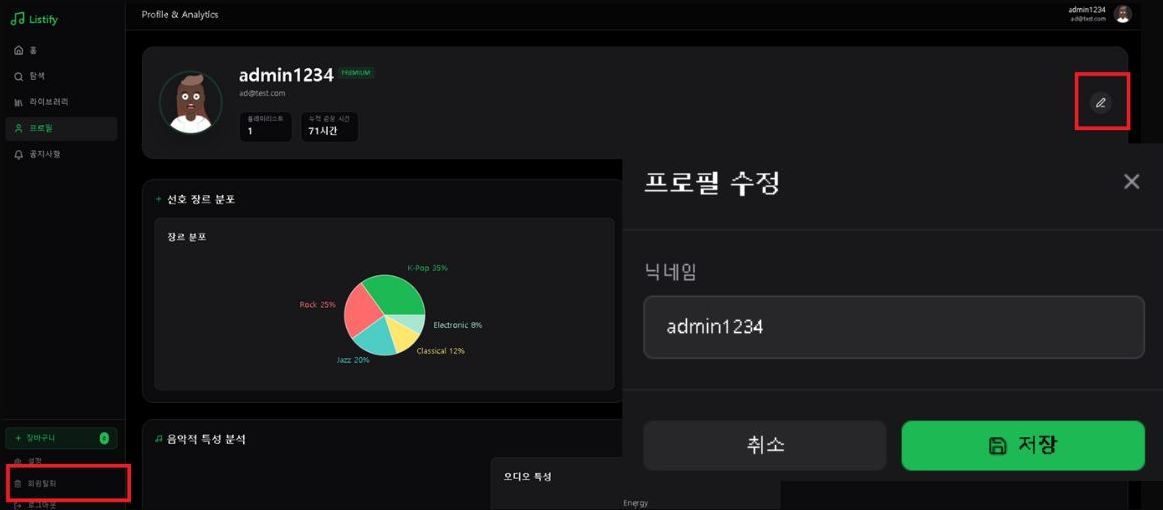
9-4. UI/UX 화면 설계



플레이리스트 관리

- 사용자는 자신만의 플레이리스트를 생성하고, 원하는 곡을 자유롭게 추가하거나 삭제(+버튼 클릭 시 새로운 플레이리스트 생성 모달창이 뜬)
- 직관적인 UI를 통해 플레이리스트 목록을 한눈에 확인하고 관리

9-5. UI/UX 화면 설계



사용자 관리

-프로필 수정 가능

(선호 장르 분포 등 데이터 시각화는 추후 구현 예정)

-회원 탈퇴 가능 (데이터는 존재)

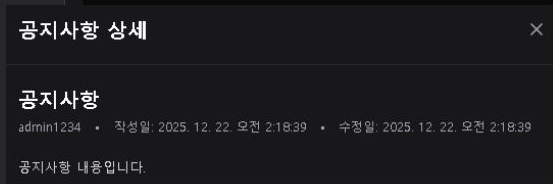
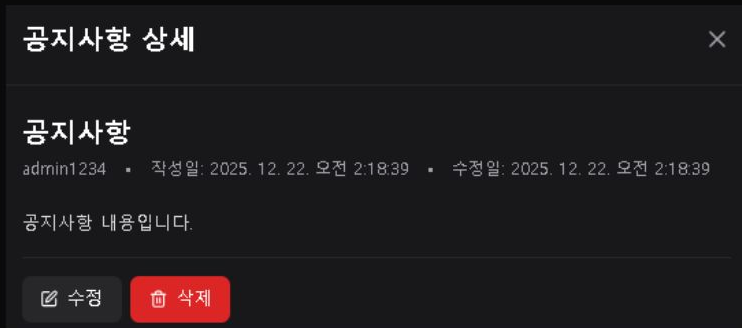
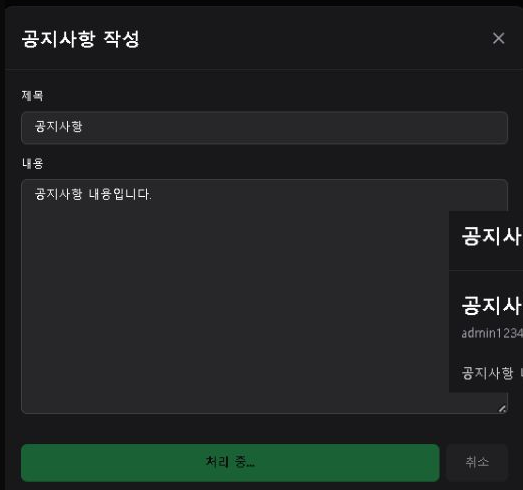
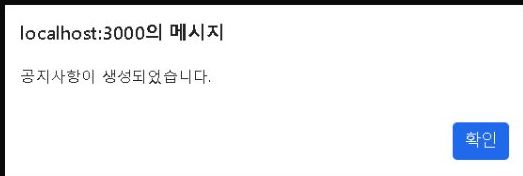
localhost:3000의 메시지

정말로 탈퇴하시겠습니까?
모든 데이터가 삭제되며 복구할 수 없습니다.

확인

취소

9-6. UI/UX 화면 설계



공지사항 관리

- admin 권한으로 공지사항 작성, 수정, 삭제 가능
- user 권한으로 공지사항 상세보기, 전체 조회 가능

01

인증 및 진입

회원가입 및 로그인을 통한
사용자 권한 획득

02

음악 탐색

Spotify API 연동을 통한
실시간 글로벌 음악 검색

03

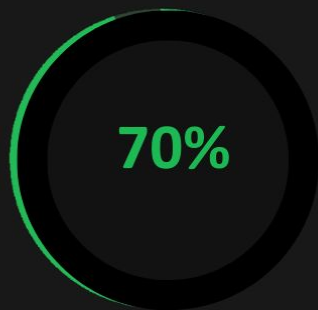
플레이리스트 관리

나만의 리스트 생성 및
검색된 곡 추가/관리

04

커뮤니티 소통

관리자 공지사항 확인 및
서비스 운영 정보 공유



프로젝트 완료율

완료된 기능 체크리스트

- ✓ Spotify API 실시간 연동
- ✓ JWT 기반 회원 인증 시스템
- ✓ 플레이리스트 CRUD 구현
- ✓ 관리자 공지사항 시스템
- ✓ 반응형 웹 디자인 적용

12. 향후 계획

20



AI 기반 음악 추천 기능

사용자의 청취 패턴과 선호 장르를 분석하여 개인별 맞춤형 음악을 자동으로 추천하는 알고리즘 도입



플레이리스트 상세 조회 및 공유

플레이리스트 상세 페이지 구현과 다른 사용자와 공유하는 기능 추가 구현



모바일/다크모드 지원

반응형 UI 고도화 및 다크모드 테마 추가



관리지 기능 고도화

통계, 사용자 관리, 신고/차단 등 운영자 도구 추가



소셜 로그인 및 OAuth 연동

Google, Kakao 등 소셜 계정으로 간편 로그인 기능



다국어 지원 기능 추가

영어, 일본어 등 다양한 언어로 서비스 이용 가능

13. 팀원 소감

21



이성훈



정혜원



조우성



최두석



김기열



Q & A

감사합니다.



<https://github.com/YeAn475/Listify>