ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN
CHUYÊN NGÀNH: CÔNG NGHỆ THÔNG TIN
ĐẶC THÙ HỢP TÁC DOANH NGHIỆP

ĐỀ TÀI:

ỨNG DỤNG NGHE NHẠC TRỰC TUYẾN VÀ NHẬN DIỆN BÀI HÁT BẰNG ÂM THANH

Người hướng dẫn: **TS. ĐẶNG HOÀI PHƯƠNG**
Sinh viên thực hiện: **NGUYỄN NGỌC HẢI**
Số thẻ sinh viên: 102200086
Lớp**: 20TCLC_DT2**

Đà Nẵng, 06/2024

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

CHUYÊN NGÀNH: CÔNG NGHỆ THÔNG TIN

ĐẶC THÙ HỢP TÁC DOANH NGHIỆP

ĐỀ TÀI:

ỨNG DỤNG NGHE NHẠC TRỰC TUYẾN VÀ NHẬN DIỆN BÀI HÁT BẰNG ÂM THANH

Người hướng dẫn: **TS. ĐẶNG HOÀI PHƯƠNG**
Sinh viên thực hiện: **NGUYỄN NGỌC HẢI**
Số thẻ sinh viên: 102200086
Lớp**: 20TCLC_DT2**

Đà Nẵng, 06/2024

TÓM TẮT

Tên đề tài: **Ứng dụng nghe nhạc trực tuyến và nhận diện bài hát bằng âm thanh**.

Sinh viên thực hiện: **Nguyễn Ngọc Hải**

Số thẻ SV: **102200086** Lớp: **20TCLC_DT2**

Tóm tắt đồ án

Dự án tốt nghiệp này tập trung vào việc xây dựng một ứng dụng nghe nhạc trực tuyến tích hợp chức năng nhận diện bài hát bằng âm thanh. Ứng dụng này nhằm cung cấp cho người dùng trải nghiệm nghe nhạc phong phú, đồng thời giúp họ dễ dàng nhận diện và tìm kiếm các bài hát dựa trên âm thanh nghe được.

Ứng dụng nghe nhạc trực tuyến được phát triển để đáp ứng nhu cầu giải trí và khám phá âm nhạc của người dùng. Nó cung cấp một nền tảng đa chức năng cho phép người dùng nghe nhạc từ nhiều nguồn khác nhau, nghe nhạc từ các danh sách phát được chọn lọc và trải nghiệm những bài hát nổi bật mà mọi người tận hưởng.

Chức năng nhận diện bài hát bằng âm thanh được thiết kế để xác định các bài hát dựa trên mẫu âm thanh đầu vào. Khi người dùng nghe thấy một bài hát mà họ muốn biết tên, họ chỉ cần ghi lại một đoạn ngắn của bài hát đó, và hệ thống sẽ phân tích và đưa ra kết quả chính xác về tên bài hát, nghệ sĩ và album tương ứng.

Ứng dụng được xây dựng với giao diện thân thiện và dễ sử dụng, giúp người dùng dễ dàng tìm kiếm và nghe nhạc. Công nghệ nhận diện âm thanh sử dụng các thuật toán và xử lý tín hiệu để đảm bảo độ chính xác cao trong việc nhận diện bài hát.

Thông qua dự án này, hy vọng rằng người dùng sẽ có một công cụ mạnh mẽ để khám phá và thưởng thức âm nhạc. Đồng thời, ứng dụng cũng mang lại sự tiện lợi trong việc nhận diện và tìm kiếm các bài hát yêu thích, góp phần nâng cao trải nghiệm âm nhạc của người dùng

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA……

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Họ tên sinh viên: Nguyễn Ngọc Hải

Số thẻ sinh viên: 102200086

Lớp: 20TCLC_DT2. Khoa: CNTT. Ngành: CNTT- Đặc thù hợp tác doanh nghiệp

1. *Tên đề tài đồ án:*

Ứng dụng nghe nhạc trực tuyến và nhận diện bài hát bằng âm thanh

2. *Đề tài thuộc diện:* ☐ *Có ký kết thỏa thuận sở hữu trí tuệ đối với kết quả thực hiện*
3. *Các số liệu và dữ liệu ban đầu:*
4. *Nội dung các phần thuyết minh và tính toán:*

1. Các chức năng của người dùng
2. Các chức năng của nghệ sĩ
3. Các chức năng của quản trị viên

5. *Các bản vẽ, đồ thị (ghi rõ các loại và kích thước bản vẽ):*

4. Sơ đồ cây phân rã chức năng
5. Biểu đồ ca sử dụng
6. Biểu đồ hoạt động
7. Biểu đồ tuần tự

6. *Họ tên người hướng dẫn:* TS. Đặng Hoài Phương
7. *Ngày giao nhiệm vụ đồ án:* *15/04/2024*
8. *Ngày hoàn thành đồ án:* *19/06/2024.*

Đà Nẵng, ngày tháng năm 2024

Trưởng Bộ môn …….. **Người hướng dẫn**

LỜI NÓI ĐẦU

Trong thời đại số hóa hiện nay, nhu cầu giải trí và khám phá âm nhạc của người dùng ngày càng gia tăng. Việc tìm kiếm và nhận diện các bài hát một cách nhanh chóng và chính xác trở thành một phần không thể thiếu trong trải nghiệm âm nhạc của người dùng. Đồ án này tập trung vào việc xây dựng một ứng dụng nghe nhạc trực tuyến tích hợp chức năng nhận diện bài hát bằng âm thanh.

Mục tiêu của hệ thống là cung cấp cho người dùng một công cụ tiện lợi và hiệu quả để nghe nhạc trực tuyến và nhận diện các bài hát dựa trên mẫu âm thanh. Ứng dụng nghe nhạc trực tuyến sẽ mang lại trải nghiệm phong phú với nhiều tính năng hữu ích.

Ứng dụng nghe nhạc trực tuyến sẽ bao gồm các tính năng sau:

- Streaming: Cho phép người dùng nghe nhạc từ nhiều nguồn khác nhau với chất lượng cao;
- Nhận diện bài hát bằng âm thanh: Cho phép người dùng ghi lại một đoạn ngắn của bài hát và hệ thống sẽ phân tích để đưa ra kết quả và phát ngay bài hát đó trên ứng dụng;
- Nghe nhạc theo playlist: Cung cấp các playlist có sẵn theo chủ đề để tăng thêm sự phóng phú cho người dùng trải nghiệm;
- Xem bảng xếp hạng các bài hát theo ngày: cho phép người dùng khám phá gu âm nhạc đại chung hiện nay, mở rộng thêm nguồn nhạc của họ;
- Tải nhạc tên hệ thống: cho phép người dùng đăng ký với tư cách là một nghệ sĩ được đăng tải các sản phẩm âm nhạc của mình lên. Qua đó quảng bá hình ảnh của mình.

Mục tiêu cuối cùng của đề tài này là tạo ra một ứng dụng nghe nhạc trực tuyến tích hợp nhận diện bài hát bằng âm thanh, giúp người dùng có trải nghiệm âm nhạc tối ưu và tiện lợi hơn. Hy vọng rằng hệ thống này sẽ đóng vai trò quan trọng trong việc nâng cao trải nghiệm âm nhạc của người dùng, đồng thời mở ra những cơ hội mới cho việc khám phá và thưởng thức âm nhạc.

Đặc biệt em xin chân thành cảm ơn thầy Đặng Hoài Phương đã tận tình giúp đỡ, đề xuất các phương án để em có thể hoàn thành đề tài này một cách chỉnh chu.

CAM ĐOAN

Em xin cam đoan rằng đồ án tốt nghiệp xây dựng ứng dụng nghe nhạc trực tuyến và nhận diên bài hát bằng âm thanh là nghiên cứu độc lập của em. Đồng thời những phần có sử dụng tài liệu tham khảo có trong đồ án đã được liệt kê và nêu rõ tại phần tài liệu tham khảo và những số liệu hay kết quả trình bày trong đồ án đều mang tính chất trung thực, không sao chép, đạo nhái.

Nếu như sai sót em xin chịu hoàn toàn trách nhiệm và chịu tất cả kỷ luật của bộ môn cũng như nhà trường đề ra.

Sinh viên thực hiện

Nguyễn Ngọc Hải

MỤC LỤC

TÓM TẮT 5
NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP 6
LỜI NÓI ĐẦU i
CAM ĐOAN ii
MỤC LỤC iii
DANH MỤC BẢNG BIỂU v
DANH MỤC HÌNH ẢNH 6
CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 7
1.1 Mô hình Client Server 7
1.2 Ngôn ngữ lập trình JavaScript và HTML/CSS/SCSS 7
1.3 ReactJS 9
1.4 MongoDb 12
1.5 NodeJS (ExpressJS) 12
1.6 React Native 13
1.7 Flask 14
1.8 Restful API 14
1.9 Thuật toán nhận dạng bài hát bằng âm thanh 17
1.10 Kết chương 19
CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG 20
2.1 Phân tích yêu cầu 20
2.2 Sơ đồ đồ cây phân rã chức năng 22
2.3 Biểu đồ Use case 24
2.3.1 Biểu đồ use case tổng quát 24
2.3.2 Biểu đồ use case chi tiết 25
2.4 Biểu đồ hoạt động 34
2.4.1 Biểu đồ hoạt động chức năng quản lý một vài chức năng của admin 34
2.4.2 Biểu đồ hoạt động chức năng quản lý playlist 35
2.4.3 Biểu đồ hoạt động chức năng đăng tải bài hát 36
2.5 Biểu đồ tuần tự 37
2.5.1 Biểu đồ tuần tự chức năng đăng ký tài khoản nghệ sĩ 37
2.5.2 Biểu đồ tuần tự chức năng quản lý yêu cầu tạo hồ sơ nghệ sĩ 38

2.5.3 Biểu đồ tuần tự chức năng tải nhạc lên của nghệ sĩ 39
2.5.4 Biểu đồ tuần tự chức năng tìm kiếm bài hát bằng âm thanh 40
2.6 Cơ sở dữ liệu 41
2.7 Kết chương 46
CHƯƠNG 3: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ 47
3.1 Môi trường triển khai 47
3.2 Kết quả triển khai 48
3.2.1 Trang web cho admin 48
3.2.2 Trang web cho nghệ sĩ 55
3.2.3 Ứng dụng mobile cho người dùng 59
3.3 Kết chương 64
KẾT LUẬN 65
TÀI LIỆU THAM KHẢO 66

DANH MỤC BẢNG BIỂU

Bảng 2.1: Bảng các tác nhân của hệ thống 23
Bảng 2.2: Đặc tả usecase quản lý người dùng 25
Bảng 2.3: Đặc tả usecase quản lý nghệ sĩ 26
Bảng 2.4: Đặc tả usecase quản lý bài hát 27
Bảng 2.5: Đặc tả usecase quản lý thể loại nhạc (genre) 29
Bảng 2.6: Đặc tả usecase quản lý bảng xếp hạng 30
Bảng 2.7: Đặc tả usecase quản lý playlist 31
Bảng 2.8: Đặc tả use case tìm kiếm bài hát 33
Bảng 2.9: Thông tin bảng User 42
Bảng 2.10: Thông tin bảng Profile 42
Bảng 2.11: Thông tin bảng ArtistProfile 42
Bảng 2.12: Thông tin bảng ArtistRequest 43
Bảng 2.13: Thông tin bảng Genre 43
Bảng 2.14: Thông tin bảng Album 43
Bảng 2.15: Thông tin bảng Single 43
Bảng 2.16: Thông tin bảng FeaturedPlaylist 44
Bảng 2.17: Thông tin bảng bài hát 44
Bảng 2.18: Thông tin bảng Join của Track và Genre 44
Bảng 2.19: Thông tin bảng UserStream 45
Bảng 2.20: Thông tin bảng Chart 45
Bảng 2.21: Thông tin bảng track của hệ thống nhận dạng 46
Bảng 2.22: Thông tin bảng fingerprint của hệ thống nhận dạng bài hát 46

DANH MỤC HÌNH ẢNH

Hình 1.1: Hình ảnh HTML DOM 10
Hình 1.2: Thống kê lượt sử dụng React JS 11
Hình 1.3: Restful API 15
Hình 1.4: Sơ đồ các bước tách fingerprint từ một file audio 17
Hình 1.5: Hình ảnh trực quan hóa quá trình trích xuất các fingerprint 18
Hình 2.1: Sơ đồ phân rã cây chức năng 22
Hình 2.2: Sơ đồ usecase tổng quát của hệ thống 24
Hình 2.3: Sơ đồ use case quản lý người dùng 25
Hình 2.4: Sơ đồ usecase quản lý nghệ sĩ 26
Hình 2.5: Sơ đồ usecase quản lý bài hát 27
Hình 2.6: Sơ đồ usecase quản lý thể loại nhạc (genre) 28
Hình 2.7: Sơ đồ usecase quản lý bảng xếp hạng của admin 30
Hình 2.8: Sơ đồ usecase quản lý playlist 31
Hình 2.9: Biểu đồ usecase tìm kiếm bài hát 33
Hình 2.10: Sơ đồ hoạt động chức năng quản lý của admin 34
Hình 2.11: Sơ đồ hoạt động chức năng quản lý playlist 35
Hình 2.12: Sơ đồ hoạt động chức năng đăng tải bài hát 36
Hình 2.13: Sơ đồ tuần tự chức năng tạo tài khoản nghệ sĩ 37
Hình 2.14: Biểu đồ tuần tự quản lý yêu cầu tạo hồ sơ nghệ sĩ 38
Hình 2.15: Sơ đồ tuần tự chức năng tải nhạc lên của nghệ sĩ 39
Hình 2.16: Sơ đồ tuần tự chức năng tìm kiếm bài hát bằng âm thanh 40
Hình 2.17: Biểu đồ cơ sở dữ liệu NoSQL của hệ thống chung 41
Hình 2.18: Biểu đồ cơ sở dữ liệu cho hệ thống nhận dạng bài hát bằng Fingerprint 46
Hình 3.1: Giao diện trang đăng nhập 48
Hình 3.2: Giao diện trang đăng ký tài khoản 48
Hình 3.3: Giao diện trang overview của admin 49
Hình 3.4: Giao diện trang quản lý yêu cầu nghệ sĩ 49
Hình 3.5: Giao diện trang quản lý nghệ sĩ 50
Hình 3.6: Giao diện trang hồ sơ của nghệ sĩ 50
Hình 3.7: Giao diện trang quản lý bài hát 51
Hình 3.8: Giao diện trang preview nghe thử nhạc 51
Hình 3.9: Giao diện trang quản lý album 52
Hình 3.10: Giao diện trang chi tiết album 52
Hình 3.11: Giao diện trang quản lý playlist 53
Hình 3.12: Giao diện trang chi tiết playlist 53
Hình 3.13: Giao diện pop-up để thêm bài hát vào playlist 54
Hình 3.14: Giao diện trang quản lý thể loại nhạc (genre) 54
Hình 3.15: Giao diện trang quản lý bảng xếp hạng 55
Hình 3.16: Giao diện trang overview của nghệ sĩ 55
Hình 3.17: Giao diện trang sản phẩm âm nhạc của nghệ sĩ (discography) 56
Hình 3.18: Giao diện tạo mới một đĩa đơn (single) – bước 1 nhập thông tin 56
Hình 3.19: Giao diện tạo mới một đĩa đơn – bước 2 tải ảnh bìa cho bài hát 57
Hình 3.20: Giao diện tạo mới một đĩa đơn – bước 3 tải file audio của bài hát 57

Hình 3.21: Giao diện tạo mới một album 58
Hình 3.22: Giao diện trang bảng xếp hạng theo ngày 58
Hình 3.23: Giao diện trang chủ của mobile 59
Hình 3.24: Giao diện trang bài hát khi được phát 60
Hình 3.25: Giao diện trang tìm kiếm bài hát bằng từ khóa 61
Hình 3.26: Giao diện trang tìm kiếm bằng âm thanh 62
Hình 3.27: Giao diện trang xem album 63
Hình 3.28: Giao diện trang xem hồ sơ của người dùng 64

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

Trong thời đại số hóa hiện nay, các ứng dụng nghe nhạc trực tuyến đã trở thành một phần không thể thiếu trong cuộc sống hàng ngày của nhiều người. Việc truy cập vào kho nhạc khổng lồ chỉ với vài thao tác đơn giản đã mang lại sự tiện lợi và trải nghiệm âm nhạc tuyệt vời cho người dùng. Tuy nhiên, một trong những thách thức mà người dùng thường gặp phải là khó nhận diện được các bài hát không rõ tên hoặc không biết thông tin cụ thể. Đề tài này giới thiệu một ứng dụng nghe nhạc trực tuyến tích hợp công nghệ nhận diện bài hát bằng âm thanh, giúp người dùng dễ dàng tìm kiếm và khám phá âm nhạc theo cách mới mẻ và hiệu quả hơn.

Đề tài này tập trung vào xây dựng một ứng dụng nghe nhạc trực tuyến, với mục tiêu tạo ra một nền tảng hiện đại, mới mẻ giữa bối cảnh các ứng dụng nghe nhạc trực tuyến ngày càng đa dạng nhưng thiếu tính độc đáo.

Ứng dụng nghe nhạc trực tuyến kết hợp nhận diện bài hát bằng âm thanh hứa hẹn mang lại một bước đột phá trong cách người dùng trải nghiệm âm nhạc. Bằng việc tích hợp công nghệ hiện đại và cung cấp các tính năng hữu ích, ứng dụng sẽ đáp ứng được nhu cầu ngày càng cao của người dùng và góp phần làm phong phú thêm thị trường âm nhạc số.

1.1 Mô hình Client Server

Client server là mô hình mạng máy tính gồm có 2 thành phần chính đó là máy khách (client) và máy chủ (server). Server chính là nơi giúp lưu trữ tài nguyên cũng như cài đặt các chương trình dịch vụ theo đúng như yêu cầu của client. Ngược lại, Client bao gồm máy tính cũng như các loại thiết bị điện tử nói chung sẽ tiến hành gửi yêu cầu đến server.

Mô hình mạng Client Server sẽ cho phép mạng tập trung các ứng dụng có cùng chức năng tại một hoặc nhiều dịch vụ file chuyên dụng. Chúng sẽ trở thành trung tâm của hệ thống. Hệ điều hành của mô hình Client server sẽ cho phép người dùng chia sẻ đồng thời cùng một loại tài nguyên mà không giới hạn vị trí địa lý.

1.2 Ngôn ngữ lập trình JavaScript và HTML/CSS/SCSS

JavaScript [8] là ngôn ngữ lập trình phổ biến dùng để tạo ra các trang web tương tác. Được tích hợp và nhúng vào HTML giúp website trở nên sống động hơn.

JavaScript đóng vai trò như một phần của trang web, thực thi cho phép Client-Side Script từ phía người dùng cũng như phía máy chủ (Nodejs) tạo ra các trang web động.

JavaScript là một ngôn ngữ lập trình thông dịch với khả năng hướng đến đối tượng. Là một trong 3 ngôn ngữ chính trong lập trình web và có mối liên hệ lẫn nhau để xây dựng một website sống động, chuyên nghiệp, có thể nhìn tổng quan như sau:

- HTML: Cung cấp cấu trúc cơ bản, hỗ trợ trong việc xây dựng layout, thêm nội dung dễ dàng trên website;
- CSS: Được sử dụng để kiểm soát và hỗ trợ việc định dạng thiết kế, bố cục, style, màu sắc,…;
- Tạo nên những nội dung "động" trên website.

Lịch sử phát triển:

- JavaScript được phát tiển bởi Brendan Eich tại hãng truyền thông Netscape với tên đầu tiên là Mocha. Sau đó, đổi tên thành LiveScript và cuối cùng là JavaSript được sử dụng phổ biến tới thời điểm bây giờ;
- Phiên bản mới nhất của JavaScript là ECMAScript (là phiên bản chuẩn hóa của JavaScript). Với ECMAScript 2 phát hành năm 1998 và ECMAScript 3 được ra mắt năm 1999.

Cách thức hoạt động: JavaScript thường sẽ được nhúng trực tiếp vào một trang web hoặc được tham chiếu qua file .js riêng. JavaScript là ngôn ngữ từ phía client nên script sẽ được tải về máy client khi truy cập và được xử lý tại đó. Thay vì tải về máy server và sau khi xử lý xong mới phản hồi kết quả đến client.

Ưu điểm của JavaScript:

- Chương trình JavaScript rất dễ học;
- Lỗi JavaScript dễ phát hiện và sẽ giúp sửa lỗi nhanh hơn
- Các trình duyệt web có thể dịch bằng HTML mà không cần một compiler;
- JS hoạt động trên rất nhiều nền tảng và trình duyệt khác nhau. Được đánh giá là ngôn ngữ lập trình nhẹ, nhanh so với các ngôn ngữ khác;
- JS có thể được gắn trên một số element hoặc các events của trang web;
- Khi website có sử dụng JS thì sẽ giúp cho trang web đó tương tác và tăng trải nghiệm người dùng khi truy cập;
- Có thể tận dụng JavaScript để kiểm tra các input thay vì kiểm tra thủ công thông qua việc truy xuất database;

- Giao diện phong phú gồm các thành phần Drag and Drop, Slider để cung cấp một Rich Interface (Giao diện giàu tính năng).

Nhược điểm của Javascript:

- Dễ bị khai thác từ những hacker và scammer;
- Có thể được dụng để thực thi mã độc trên máy tính của người dùng.
- JS code snippet lớn. Các thiết bị khác nhau có thể thực hiện JS khác nhau dẫn đến không đồng nhất
- Vì tính bảo mật nên client-side JavaScript không cho phép đọc và ghi các file.
- JS không được hỗ trợ khi sử dụng trong kết nối mạng. JavaScript không có khả năng đa luồng hoặc đa xử lý.

1.3 ReactJS

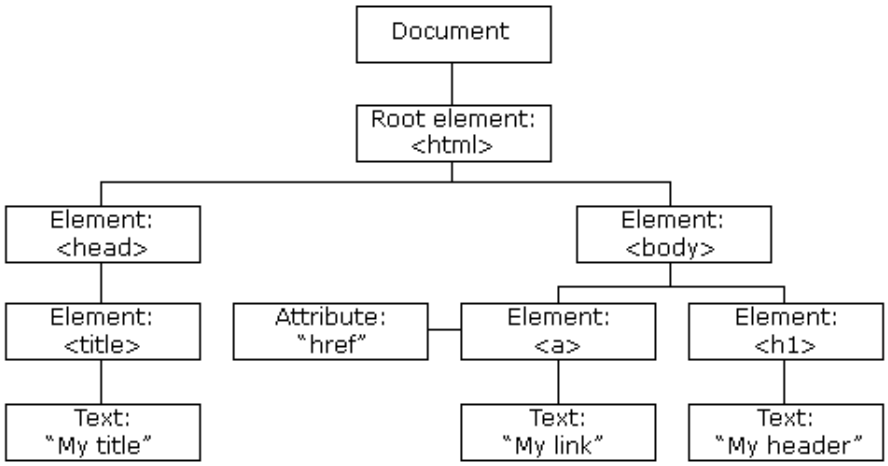
ReactJS [2] là một opensource được phát triển bởi Facebook, ra mắt vào năm 2013, bản thân nó là một thư viện Javascript được dùng để để xây dựng các tương tác với các thành phần trên website. Một trong những điểm nổi bật nhất của ReactJS đó là việc render dữ liệu không chỉ thực hiện được trên tầng Server mà còn ở dưới Client.

ReactJS là một thư viện JavaScript chuyên giúp các nhà phát triển xây dựng giao diện người dùng hay UI. Trong lập trình ứng dụng front-end, lập trình viên thường sẽ phải làm việc chính trên 2 thành phần sau: UI và xử lý tương tác của người dùng. UI là tập hợp những thành phần mà bạn nhìn thấy được trên bất kỳ một ứng dụng nào, ví dụ có thể kể đến bao gồm: menu, thanh tìm kiếm, những nút nhấn, card,… Giả sử đang lập trình một website thương mại điện tử, sau khi người dùng chọn được sản phẩm ưng ý rồi và nhấn vào nút "Thêm vào giỏ hàng", thì việc tiếp theo phải làm đó là thêm sản phẩm được chọn vào giỏ hàng và hiển thị lại sản phẩm đó khi user vào xem. Điều này được được gọi là xử lý tương tác.

Trước khi có ReactJS, lập trình viên thường gặp rất nhiều khó khăn trong việc sử dụng "vanilla JavaScript"(JavaScript thuần) và JQuery để xây dựng UI. Điều đó đồng nghĩa với việc quá trình phát triển ứng dụng sẽ lâu hơn và xuất hiện nhiều bug, rủi ro hơn. Vì vậy vào năm 2011, Jordan Walke – một nhân viên của Facebook đã khởi tạo ReactJS với mục đích chính là cải thiện quá trình phát triển UI.

Hơn nữa, để tăng tốc quá trình phát triển và giảm thiểu những rủi ro có thể xảy ra trong khi coding, React còn cung cấp khả năng Reusable Code (tái sử dụng code) bằng cách đưa ra 2 khái niệm quan trọng bao gồm:

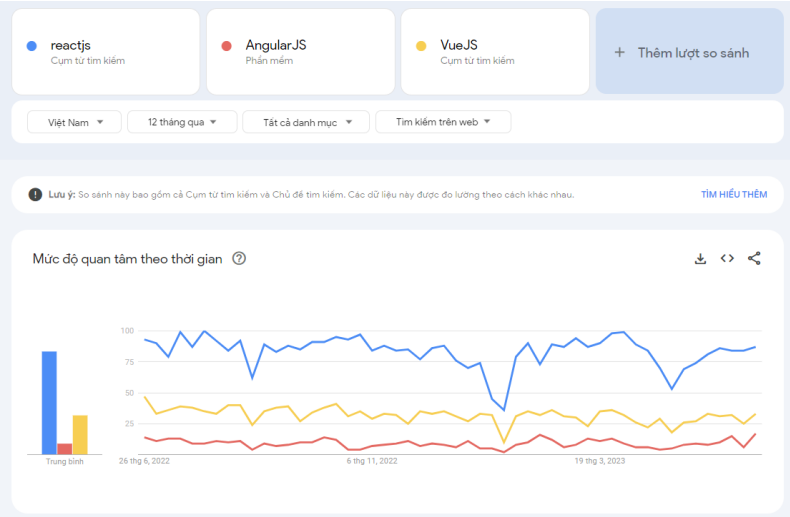
- JSX.
- Virtual DOM.



Hình 1.1: Hình ảnh HTML DOM

Ưu điểm của React JS:

- Phù hợp với đa dạng thể loại website: ReactJS khiến cho việc khởi tạo website dễ dàng hơn bởi vì bạn không cần phải code nhiều như khi tạo trang web thuần chỉ dùng JavaScript, HTML và nó đã cung cấp đủ loại công cụ để có thể dùng cho nhiều trường hợp;
- Tái sử dụng các Component: Nếu xây dựng các Component đủ tốt, đủ flexible để có thể thõa các yêu cầu của nhiều dự án khác nhau, thì chỉ tốn thời gian xây dựng ban đầu và sử dụng lại hầu như toàn bộ ở các dự án sau. Không chỉ riêng mỗi ReactJS mà các framework hiện nay cũng đều cho phép thực hiện điều đó;
- Có thể sử dụng cho cả Mobile application: Hầu hết được biết rằng ReactJS được sử dụng cho việc lập trình website, nhưng thực chất nó được sinh ra không chỉ làm mỗi đều đó. Nếu bạn cần phát triển thêm ứng dụng Mobile, thì hãy sử dụng thêm React Native – một framework khác được phát triển cũng chính Facebook, có thể dễ dàng "chia sẻ" các Component hoặc sử dung lại các Business Logic trong ứng dụng;
- Thân thiện với SEO: SEO là một phần không thể thiếu để đưa thông tin website được lên top đầu tìm kiếm của Google. Bản chất ReactJS là một thư viện JavaScript, Google Search Engine hiện nay đã crawl và index được code JavaScript, tuy nhiên cũng cần thêm một vài thư viện khác để hỗ trợ điều này;
- Debug dễ dàng: Facebook đã phát hành một Chrome extension dùng trong việc debug trong quá trình phát triển ứng dụng. Điều đó giúp tăng tốc quá trình release sản phẩm cung như quá trình coding của dự án;
- Công cụ phát triển web hot nhất hiện nay: Nhìn vào số liệu thống kê từ Google Trend ở Việt Nam ở hình bên dưới, lướt qua các trang tuyển dụng hàng đầu ở Việt Nam như Topdev, Itviec,... có thể thấy số lượng tuyển dụng cho vị trí React Developer là cực kỳ lớn cùng với mức lương vô cùng hấp dẫn và độ phổ biến hiện tại của ReactJS trên thị trường Việt Nam.



Hình 1.2: Thống kê lượt sử dụng React JS

1.4 MongoDb

MongoDB [4] là một hệ quản trị cơ sở dữ liệu mã nguồn mở, là CSDL thuộc NoSql và được hàng triệu người sử dụng.

MongoDB là một database hướng tài liệu (document), các dữ liệu được lưu trữ trong document kiểu JSON thay vì dạng bảng như CSDL quan hệ nên truy vấn sẽ rất nhanh.

Với CSDL quan hệ chúng ta có khái niệm bảng, các cơ sở dữ liệu quan hệ (như MySQL hay SQL Server...) sử dụng các bảng để lưu dữ liệu thì với MongoDB chúng ta sẽ dùng khái niệm là **collection** thay vì bảng.

So với RDBMS thì trong MongoDB **collection** ứng với **table**, còn **document** sẽ ứng với **row** , MongoDB sẽ dùng các document thay cho row trong RDBMS.

Các collection trong MongoDB được cấu trúc rất linh hoạt, cho phép các dữ liệu lưu trữ không cần tuân theo một cấu trúc nhất định.

Thông tin liên quan được lưu trữ cùng nhau để truy cập truy vấn nhanh thông qua ngôn ngữ truy vấn MongoDB

1.5 NodeJS (ExpressJS)

NodeJS [3] là một mã nguồn được xây dựng dựa trên nền tảng Javascript V8 Engine, nó được sử dụng để xây dựng các ứng dụng web như các trang video clip, các forum và đặc biệt là trang mạng xã hội phạm vi hẹp. NodeJS là một mã nguồn mở được sử dụng rộng bởi hàng ngàn lập trình viên trên toàn thế giới. NodeJS có thể chạy trên nhiều nền tảng hệ điều hành khác nhau từ WIndow cho tới Linux, OS X nên đó cũng là một lợi thế. NodeJS cung cấp các thư viện phong phú ở dạng Javascript Module khác nhau giúp đơn giản hóa việc lập trình và giảm thời gian ở mức thấp nhất

Các đặc tính của NodeJS

- Không đồng bộ: Tất cả các API của NodeJS đều không đồng bộ (non-blocking), nó chủ yếu dựa trên nền của NodeJS Server và chờ đợi Server trả dữ liệu về. Việc di chuyển máy chủ đến các API tiếp theo sau khi gọi và cơ chế thông báo các sự kiện của NodeJS giúp máy chủ để có được một phản ứng từ các cuộc gọi API trước (Realtime);
- Chạy rất nhanh: NodeJS được xây dựng dựa vào nền tảng V8 Javascript Engine nên việc thực thi chương trình rất nhanh;
- Đơn luồng nhưng khả năng mở rộng cao: NodeJS sử dụng một mô hình luồng duy nhất với sự kiện lặp. cơ chế tổ chức sự kiện giúp các máy chủ để đáp ứng một cách không ngăn chặn và làm cho máy chủ cao khả năng mở rộng như trái ngược với các máy chủ truyền thống mà tạo đề hạn chế để xử lý yêu cầu. NodeJS sử dụng một chương trình đơn luồng và các chương

trình tương tự có thể cung cấp dịch vụ cho một số lượng lớn hơn nhiều so với yêu cầu máy chủ truyền thống như Apache HTTP Server;
- NodeJS không đệm bất kì một dữ liệu nào và các ứng dụng này chủ yếu là đầu ra dữ liệu;
- Có giấy phép: NodeJS đã được cấp giấy phép bởi MIT License.

ExpressJS là một framework phổ biến được sử dụng để xây dựng ứng dụng web và API thông qua NodeJS. Nền tảng được xem là một phương thức xử lý các yêu cầu HTTP, quản lý các tuyến đường, xử lý phần mềm trung gian và nhiều tính năng khác để phát triển hiệu quả ứng dụng web.

ExpressJS tập trung vào công việc tối ưu hóa việc xây dựng web ứng dụng bằng cách cung cấp một cấu trúc hoạt động và chỉ định rõ ràng việc xử lý yêu cầu và phản hồi. Nền tảng cũng hỗ trợ tích hợp các phần mềm trung gian bên ngoài để mở rộng chức năng của ứng dụng.

1.6 React Native

React Native [6] là một framework mã nguồn mở được sáng tạo bởi Facebook Nó được sử dụng để phát triển ứng dụng di động Android, iOS, Web và UWP bằng cách cho phép các nhà phát triển sử dụng React cùng với môi trường ứng dụng gốc (native). Trên các phiên bản hệ điều hành Windows đều được trang bị sẵn tính năng Remote Desktop.

- Native App là tên gọi của những ứng dụng được xây dựng và phát triển bằng những công cụ do chính nhà phát triển cung cấp cho lập trình viên. Hai nhà phát triển App hàng đầu là Android và iOS;
- Hybrid App là sự kết hợp giữa ứng dụng Web và ứng dụng mobile. Người dùng có thể cài đặt lên điện thoại của mình giống như những ứng dụng Native bình thường, vừa có thể tìm thấy ở những kho ứng dụng trả phí.

Nguyên lý hoạt động: **React Native** được viết bằng sự kết hợp của JavaScript và JXL, một mã đánh dấu đặc biệt giống với XML. Framework có khả năng thao tác với cả hai thread là main thread và JS thread. Mỗi thread đều có vai trò riêng biệt:

- Main thread: Đảm nhiệm vai trò cập nhật giao diện người dùng và xử lý tương tác người dùng.
- JS thread: Đảm bảo hệ thống hoạt động hiệu quả thông qua việc thực thi và xử lý code JavaScript

Nguyên lý hoạt động của **React Native** gần như tương tự với React. React Native không sử dụng thao tác với DOM và HTML mà chạy một quá trình xử lý nền với nền tảng gốc.

React Native sử dụng Bridge (cầu nối). Mặc dù các thread JavaScript và native được viết bằng các ngôn ngữ hoàn toàn khác nhau, nhưng Bridge là tính năng cầu nối giúp thao tác hai chiều có thể thực hiện dễ dàng hơn.

1.7 Flask

Flask [7] là loại framework web phổ biến được viết bằng trình lập ngôn ngữ Python. Công nghệ thường được sử dụng để xây dựng trang web từ những ứng dụng đơn giản đến những hệ thống phức tạp hơn.

Flask được thiết kế để hoạt động và mở rộng một cách, đồng thời nó cũng cung cấp các công cụ và thư viện cần thiết để phát triển ứng dụng web hiệu quả. Flask cũng có cộng đồng sáng tạo và hỗ trợ mạnh mẽ từ cộng đồng Python.

Dưới đây là một số tính năng chính của một server được viết bằng Flask:

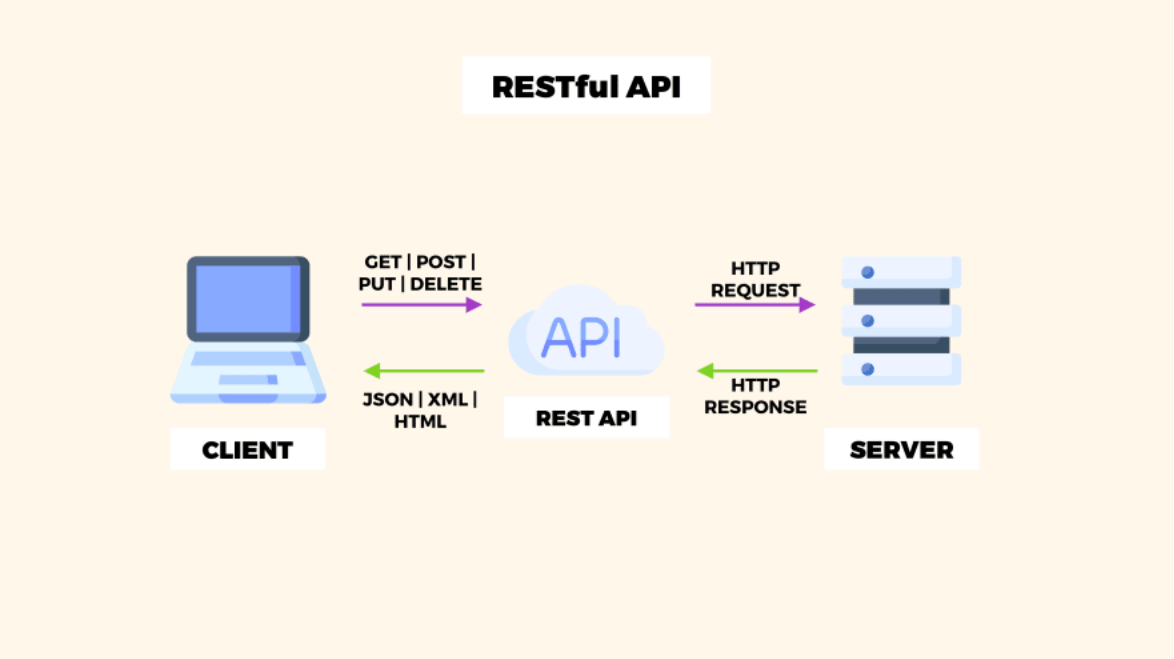
- Nhẹ và dễ sử dụng: Công nghệ có cấu trúc nhẹ nhàng và mã nguồn dễ đọc, giúp người phát triển dễ dàng tiếp cận và tùy chỉnh theo nhu cầu cụ thể;
- Định tuyến linh hoạt: Flask cung cấp cơ chế hoạt động định tuyến, cho phép người phát triển xác định các mẫu URL và phân bổ chúng cho các hàm xử lý tương ứng. Điều này giúp quản lý và xử lý yêu cầu HTTP một cách hiệu quả;
- Được mở rộng rộng rãi: Mặc dù mang đặc điểm rút gọn nhưng Flask vẫn có khả năng mở rộng mạnh mẽ thông qua việc sử dụng các tiện ích và thư viện của cộng đồng. Người dùng có thể phân tích các tính năng như xác thực, đăng nhập, điều hướng, cơ sở dữ liệu tương tác và nhiều tính năng khác;
- Máy chủ phát triển tích hợp: Flask cung cấp máy chủ phát triển hợp đồng, giúp người phát triển dễ dàng kiểm tra và phát triển ứng dụng mà không cần cấu hình bổ sung;
- Gửi yêu cầu RESTful: Flask hỗ trợ xây dựng API và các ứng dụng RESTful theo cách hoạt động và hiệu quả.

1.8 Restful API

Web API là phương thức dùng để cho phép các ứng dụng khác nhau có thể giao tiếp, trao đổi dữ liệu qua lại. Dữ liệu được web api trả về thường có dạng JSON hoặc XML thông qua giao thức HTTP hoặc HTTPS. Một số điểm nổi bật:

- Web API hỗ trợ restful đầy đủ các phương thức: GET, POST, PUT, DELETE dữ liệu, giúp xây dựng các HTTP service một cách rất đơn giản và nhanh chóng. Web API cũng có khả năng hỗ trợ đầy đủ các thành phần HTTP: URI, request/response, header, caching, versioning, content format;
- Tự động hóa sản phẩm: với web api chúng ta sẽ tự động hóa quản lý công việc, cập nhật luồng công việc, giúp tăng năng suất và hiệu quả công việc hơn;

- Khả năng tích hợp linh động: cho phép lấy nội dung từ một website một cách dễ dàng nếu được cho phép, tăng trải nghiệm người dùng, API hoạt động như một chiếc cổng, cho phép các công ty chia sẻ thông tin được chọn nhưng vẫn tránh được những yêu cầu không mong muốn.



Hình 1.3: Restful API

RESTful API là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng web (thiết kế Web services) để tiện cho việc quản lý các resource. Nó chú trọng vào tài nguyên hệ thống (tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu động…), bao gồm các trạng thái tài nguyên được định dạng và được truyền tải qua HTTP.

Chức năng quan trọng nhất của REST là quy định cách sử dụng các HTTP method (như GET, POST, PUT, DELETE…) và cách định dạng các URL cho ứng dụng web để quản các resource. RESTful không quy định logic code ứng dụng và không giới hạn bởi ngôn ngữ lập trình ứng dụng, bất kỳ ngôn ngữ hoặc framework nào cũng có thể sử dụng để thiết kế một RESTful API.

Cách thức hoạt động:

- REST hoạt động chủ yếu dựa vào giao thức HTTP. Các hoạt động cơ bản nêu dưới sẽ sử dụng những phương thức HTTP riêng.
- GET: Trả về một Resource hoặc một danh sách Resource
- POST: Tạo mới một Resource.
- PATCH: Cập nhật thông tin cho một phần Resource.
- PUT: Cập nhật thông tin cho Resource.

- DELETE: Xoá một Resource.

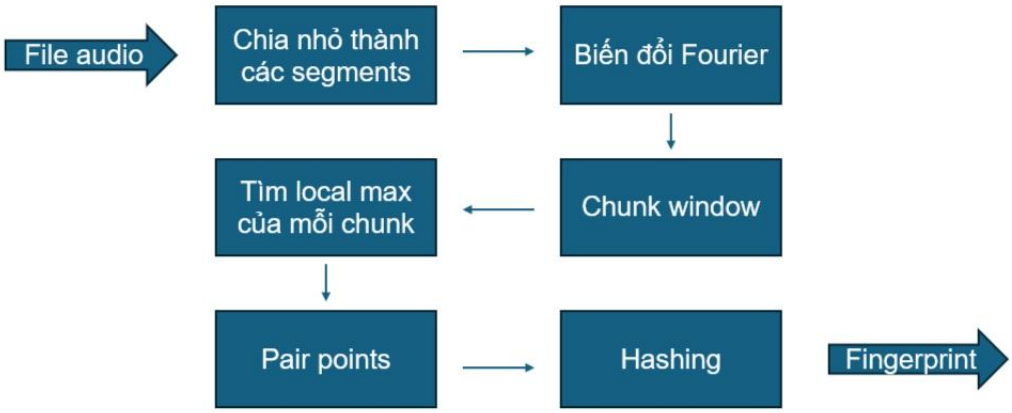
Một số status code khi ta request một API:

- 200 OK – Trả về thành công cho những phương thức GET, PUT, PATCH hoặc DELETE.
- 304 Not Modified – Client có thể sử dụng dữ liệu cache
- 400 Bad Request – Request không hợ
- 401 Unauthorized – Request cần có auth.
- 403 Forbidden – bị từ chối không cho phép.
- 405 Method Not Allowed – Phương thức không cho phép với user hiện tại.
- 410 Gone – Resource không còn tồn tại, Version cũ đã không còn hỗ trợ.
- 429 Too Many Requests – Request bị từ chối do bị giới hạn.

1.9 Thuật toán nhận dạng bài hát bằng âm thanh

Hiện này, có rất nhiều hệ thông nhận diện âm thanh, có thể sử dụng trí tuệ nhân tạo, học máy. Nhưng trong đồ án lần này, chủ yếu sẽ đi sâu vào việc nhận diện bài hát bằng **Ascoustic Fingerprint [1]**.

Thông tin dưới đây sẽ trình bày về các bước để thực hiện tách các fingerprint trong một file audio.



Hình 1.4: Sơ đồ các bước tách fingerprint từ một file audio

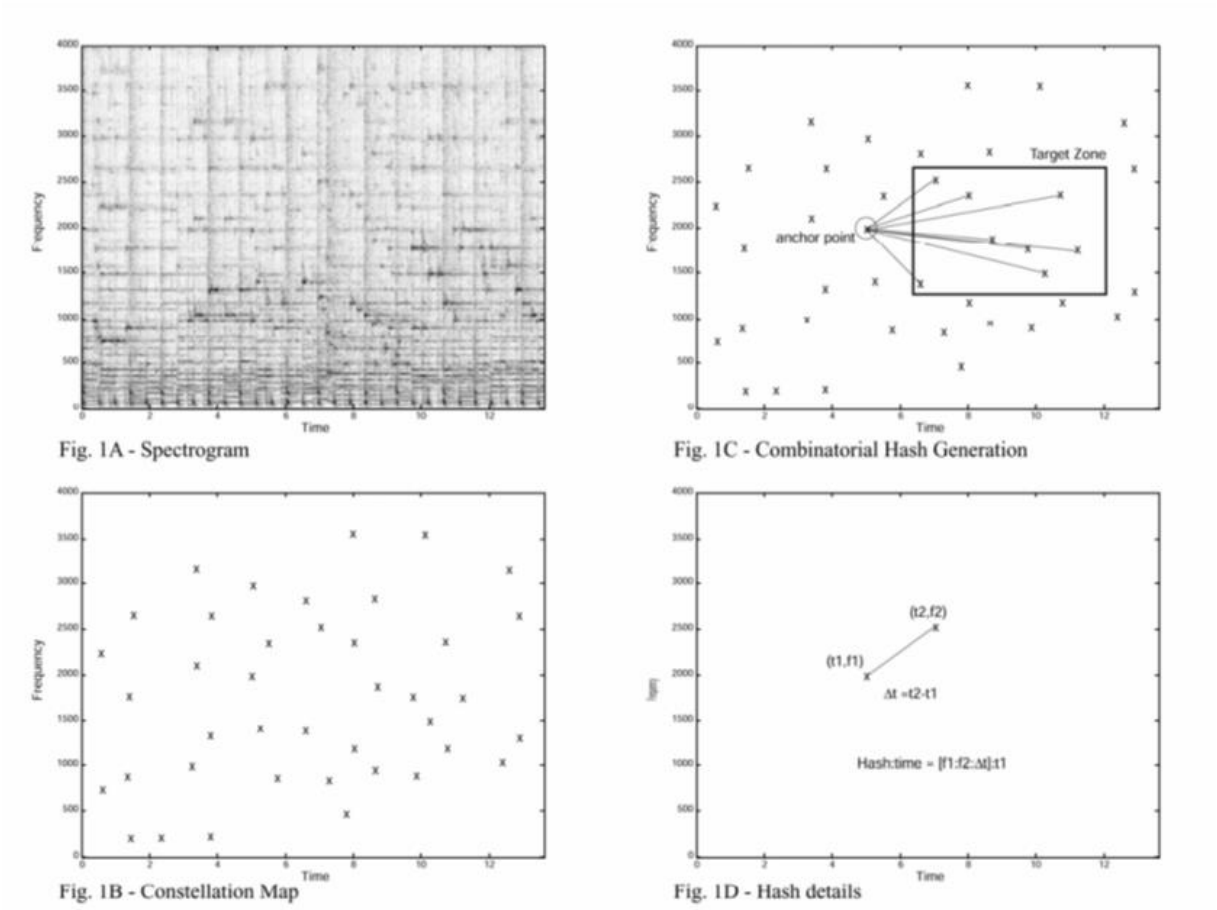
Trước tiên, file audio được chia nhỏ thành các segment và được xử lý bằng biến đổi Fourier. Sau đó tín hiệu tiếp tục được chia nhỏ thành các window frame

Dữ liệu được trực quan hóa lên spectrogram theo như hình 1.5.1A. Từ spectrogram, ta chỉ giữ lại các peak spectrogram (điểm có biên độ cao nhất ứng với từng thời điểm)

Sau bước trên, chiều của biên độ đã bị loại bỏ. Điều này có nghĩa là thuật toán đã giảm chiều dữ liệu xuống chỉ còn time và frequency. Tập hợp các điểm này được gọi là "Constellation Map" (hình 1.5.1B).

Điểm có peak local cao nhất trong từng frame (được gọi là anchor point) sẽ được ghép cặp (pair) với các peak khác trong frame tiếp theo. Mỗi pair này sau khi được hash sẽ sinh ra một fingerprint theo công thức như hình 1.5.1D.

Những fingerprint này không chỉ chứa mỗi frequency theo cách xử lý dữ liệu thông thường, mà còn nhiều thông tin hơn chính là độ chênh lệch thời gian giữa những frequency. Những thông tin này làm cho việc phân biệt các bài hát trở nên dễ dàng hơn.



Hình 1.5: Hình ảnh trực quan hóa quá trình trích xuất các fingerprint

Mỗi khi một bài hát được được thêm mới vào ứng dụng, sẽ trải qua các bước xử lý như trên. Sau quá trình đó ta sẽ thu được các fingerprint được hash vào một cơ sở dữ liệu riêng biệt.

Khi cần tìm kiếm một bài hát bằng một file audio, chỉ cần tách các fingerprint của file audio đó. Rồi so khớp với các hash fingerprint trong cơ sở dữ liệu. Bài hát có số hash trùng khớp nhiều nhất có khả năng cao chính là bài hát đang cần tìm.

Thuật toán sử dụng fingerprint để nhận dạng này có một số ưu điểm sau so với các phương pháp sử dụng trí tuệ nhân tạo hay học máy:

- **Tốc độ xử lý nhanh:** fingerprint thường có tốc độ nhận diện rất nhanh do các dữ liệu được hash và đánh index trong cơ sở dữ liệu;
- **Hiệu suất cao:** phương pháp này không yêu cầu quá trình huấn luyện phức tạp và thường đạt hiệu suất cao ngay cả với các đoạn âm thanh ngắn và chất lượng thấp;
- **Không cần dữ liệu huấn luyện:** khác với các phương pháp AI/machine learning, thuật toán sử dụng fingerprint không cần một bộ dữ liệu huấn luyện lớn để hoạt động hiệu quả. Điều này giúp giảm bớt công sức và thời gian chuẩn bị dữ liệu;
- **Tính mở rộng:** phương pháp này thường dễ dàng mở rộng khi lượng bài các ngày càng tăng, vì chỉ cần trích xuất fingerprint mỗi khi bài hát được thêm vào, không yêu cầu huấn luyện lại mô hình.

Tuy nhiên, fingerprint cũng có một số hạn chế nhất định:

- **Khó khăn khi nhận diện các bài hát cover, remix hay cùng thể loại (genre):** các bài hát có cấu trúc thương tự nhau thường bị nhầm lẫn do có các fingerprint giống nhau;
- **Phụ thuộc và cơ sở dữ liệu:** phương pháp này yêu cầu một cơ sở dữ liệu đủ lớn để lưu trữ toàn bộ fingerprint của tất cả các bài hát;
- **Không nhận diện được nhạc mới:** các bài hát chưa được trích xuất fingerprint sẽ không thể được nhận diện;
- **Dễ bị nhiễu:** tác động âm thanh môi trường bên ngoài nếu đủ lớn sẽ ảnh hưởng tới kết quả nhận diện.

1.10 Kết chương

Chương này cung cấp cho người đọc một cái nhìn tổng quan về các lý thuyết, công nghệ sẽ được sử dụng để xây dựng một website, một ứng dụng mobile.

- Đối với backend: Sử dụng NodeJS (ExpressJS) để xây dựng API tổng cho hệ thống. Áp dụng framework Flask của Python;
- Đối với frontend: sử dụng framework NextJS của ReactJS cùng với HTML/CSS;
- Đối với mobile: sử dụng React Native để triển khai.

Ngoài ra, chương này còn cung cấp thêm thông tin về cách triển khai thuật toán nhận diện bài hát bằng Audio Fingerprint.

CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG

2.1 Phân tích yêu cầu

Website được chia làm 2 phần chính: Quản trị viên của ứng dụng, Nghệ sĩ đã được quản trị viên xác nhận để có thể đăng tải sản phẩm âm nhạc của mình lên ứng dụng.

Ứng dụng mobile app chỉ dành riêng cho người dùng để có thể nghe nhạc trực tuyến.

Người dùng ứng dụng sẽ có một số chức năng sau:

- **Nghe nhạc trực tuyến:** người dùng có thể đăng nhập vào ứng dụng để phát các bài hát có sẵn trên nền tảng, có thể tạm dừng bài hát, nghe tiếp bài hát hát trong cùng một playlist hoặc album, tua tới hoặc tua lùi bài hát;
- **Tìm kiếm bài hát:** người dùng có thể tìm kiếm bài hát, album, nghệ sĩ, playlist có sẵn trên nền tảng bằng cách nhập vào ô tìm kiếm. Ngoài ra, đặc biệt người dùng có thể tìm kiếm bài hát bằng một file audio có sẵn trên máy;
- **Xem lại lịch sử phát:** người dùng đã đăng ký có thể xem lại toàn bộ lịch sử nghe nhạc của mình trên ứng dụng;
- **Xem lại nghệ sĩ đã nghe:** người dùng đã đăng ký có thể xem nghệ sĩ nào mình đã và đang nghe nhiều nhất.

Nghệ sĩ được xác nhận sẽ có một số chức năng sau:

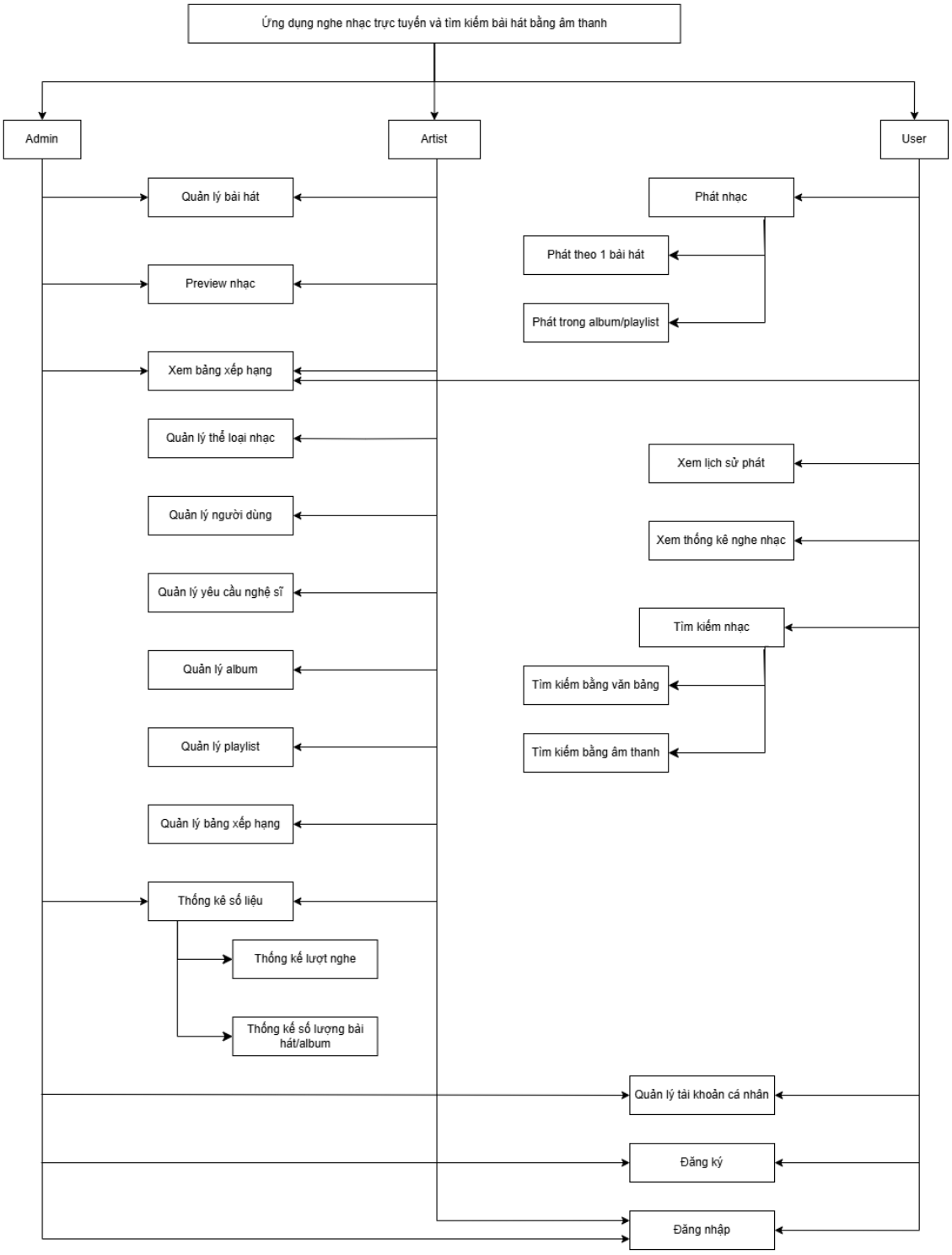
- **Tạo hồ sơ cá nhân:** nghệ sĩ có thể tạo hồ sơ, tải ảnh cá nhân của mình lên hệ thống để dễ dàng nhận diện bản thân trước khi được admin chấp nhận;
- **Đăng tải nhạc:** nghệ sĩ có thể đăng tải các sản phẩm âm nhạc của mình lên nền tảng dưới dạng đĩa đơn(single) hoặc album (gồm nhiều bài hát) để người dùng có thể trải nghiệm;
- **Quản lý hồ sơ cá nhân:** nghệ sĩ có thể xem lại các sản phẩm âm nhạc của mình đã đăng tải lên nền tảng, bao gồm đĩa đơn, album, các bài hát riêng lẻ;
- **Xem bảng xếp hạng:** nghệ sĩ có thể xem bảng xếp hạng các bài hát theo từng ngày trên ứng dụng, qua đó có thể nắm bắt tình hình nhạc số của mình có được khả quang hay không, có được người dùng đón nhận hay không;
- **Xem các bát hát nổi bật:** nghệ sĩ có thể xem được tổng số lượt nghe của mình, tổng số bài hát, album, xem bài hát nào được nghe nhiều nhất trên nền tảng;

Quản trị viên sẽ có một số chức năng sau:

- **Quản lý tài khoản:** quản trị viên có quyền xem danh sách toàn bộ người dùng đã đăng kí trên nền tảng của mình, có thể khóa tài khoản của họ;
- **Quản lý yêu cầu nghệ sĩ:** quản trị viên có quyền xem danh sách toàn bộ yêu cầu tạo tài khoản của nghệ sĩ, có quyền từ chối hoặc chấp nhận yêu cầu của họ. Nếu được chấp nhận, nghệ sĩ có thể đăng nhập vào nền tảng và đăng tải các sản phẩm âm nhạc;

- **Quản lý thể loại nhạc (genre):** quản trị viên có khả năng xem danh danh toàn bộ thể loại nhạc, có thể thêm, sửa các thể loại cho phù hợp với nhu cầu của ứng dụng;
- **Quản lý bài hát:** quản trị viên có khả năng xem toàn bộ bài hát trên nền tảng, có thể xem thông tin chi tiết của bài hát như tên, nghệ sĩ sở hữu, người sáng tác, sản xuất âm nhạc. Ngoài ra có thể preview nghe thử bài hát;
- **Quản lý album:** quản trị viên có khả năng xem toàn hộ album trên nền tảng, có thể xem thông tin chi tiết của album và cũng có thể preview nghe thử tất cả các bài hát có trong album đó;
- **Quản lý playlist:** quản trị viên có quyền xem toàn bộ playlist đã được tạo, có thể thêm mới playlist cho người dùng nghe, có thể thêm, xóa các bài hát trong playlist và cũng có thể preview nghe thử các bài hát đó;
- **Quản lý bảng xếp hạng:** quản trị viên có quyền cập nhạc bảng xếp hạng hoặc bảng xếp hạng có thể tự động cập nhật vào cuối ngày. Ngoài ra, quản trị viên có thể xem chi tiết bảng xếp hạng theo từng ngày được chọn;
- **Xem thông tin tổng quan của nền tảng:** quản trị viên có thể xem một số thông tin tổng quan của ứng dụng như: số người dùng trên hệ thống, số nghệ sĩ, số bài hát và album được đăng tải, số lượng lượt nghe nhạc trên toàn bộ hệ thống.

2.2 Sơ đồ đồ cây phân rã chức năng



Hình 2.1: Sơ đồ phân rã cây chức năng

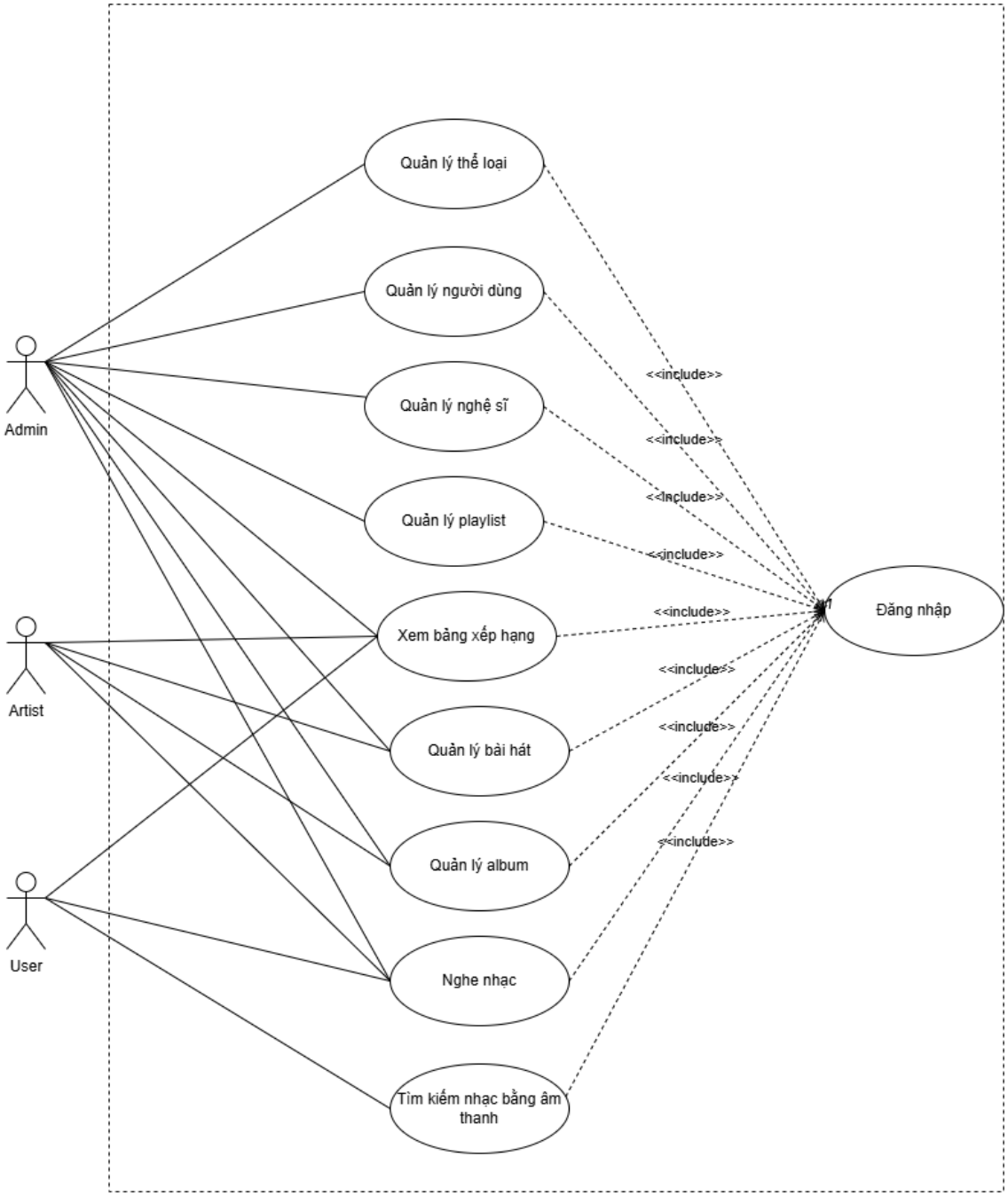
Bảng dưới đây sẽ mô tả tất cả các tác nhân tham gia vào hệ thống, mỗi tác nhân sẽ có một vai trò khác nhau trong hệ thống.

Bảng 2.1: Bảng các tác nhân của hệ thống

STT	Tên tác nhân	Mô tả
1	Admin	- Là người quản lý toàn bộ hệ thống; - Có quyền xem, tìm kiếm, thêm, sửa, xóa các dữ liệu như bài hát, album, các thể loại nhạc, nghệ sĩ, người dùng và tạo plalylist các bài hát cho người dùng; - Có quyền chấp thuận yêu cầu tạo tài khoản của một nghệ sĩ. Có thể xem thống kê toàn bộ hệ thống như tổng số người dùng, tổng số nghệ sĩ, tổng số các bài hát trên nền tảng và tổng số lượt nghe nhạc của toàn bộ người dùng.
2	Artist	- Là người có thể đăng tải các sản phẩm âm nhạc của mình lên nền tảng, qua đó người dùng ứng dụng có thể thỏa thích tận hưởng các bài hát; - Có quyền xem các thông số của bản thân như số album đăng tạo, số bài hát đã tải lên và tổng số lượt nghe của các bài hát của mình; - Có thể chỉnh sửa thông tin cá nhân bản thân.
3	User	- Là người trực tiếp sử dụng ứng dụng, ở đây họ có thể thỏa thích nghe tất cả các bài hát có trên nền tảng; - Có khả năng tìm kiếm bài hát bằng một đoạn âm thanh có sẵn.

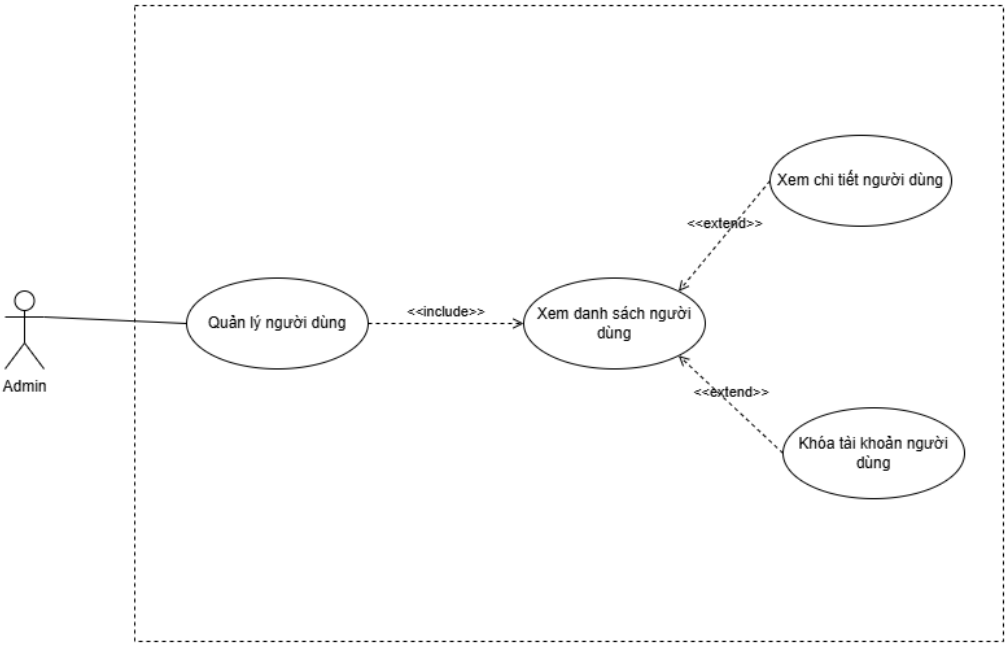
2.3 Biểu đồ Use case

2.3.1 Biểu đồ use case tổng quát



Hình 2.2: Sơ đồ usecase tổng quát của hệ thống

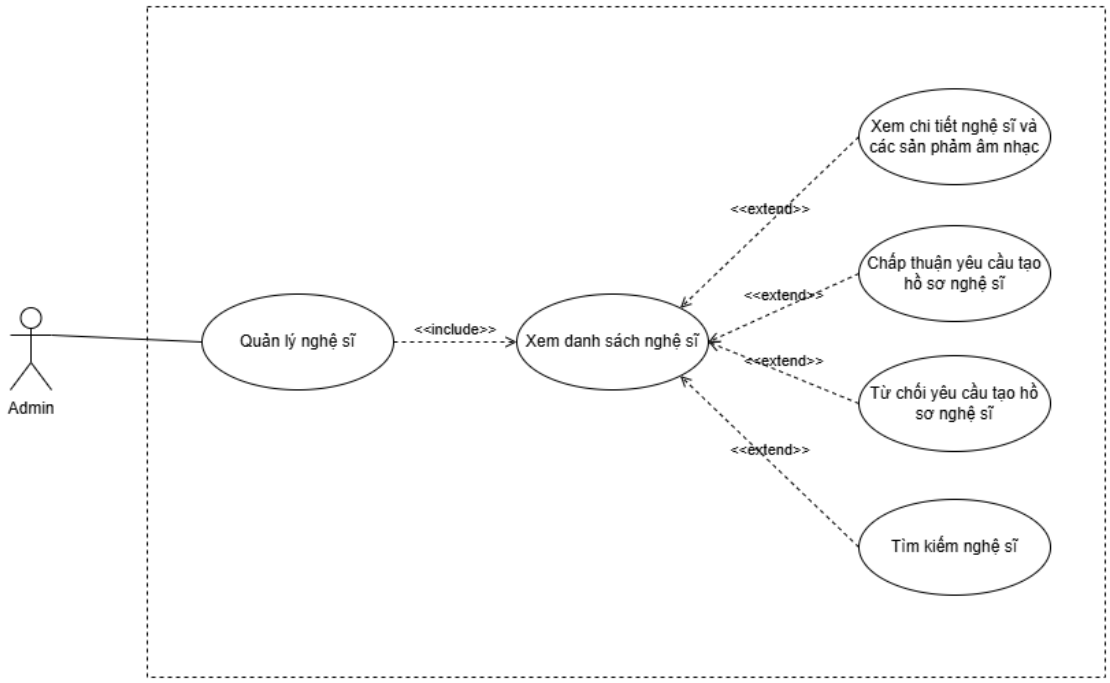
2.3.2 Biểu đồ use case chi tiết



Hình 2.3: Sơ đồ use case quản lý người dùng

Bảng 2.2: Đặc tả usecase quản lý người dùng

Use case quản lý người dùng			
Tác nhân	Admin		
Mô tả	Admin có thể quản lý các người dùng trong hệ thống		
Điều kiện trước	Phải đăng nhập vào hệ thống và chọn mục [User Management] ở thanh menu		
Điều kiện sau			
Kịch bản		**Hành động của tác nhân**	**Hành động của hệ thống**
	1	Đăng nhập vào hệ thống và nhấn vào [User Management] trên thanh menu	Thông tin của tất cả các nhân viên hiện ra
	2	Nhấn vào nút xem chi tiết người dùng	Hiện ra đầy đủ thông tin của người dùng
	3	Nhấn vào nút [Lock]	Thực hiện khóa người dùng và hiển thị thông báo

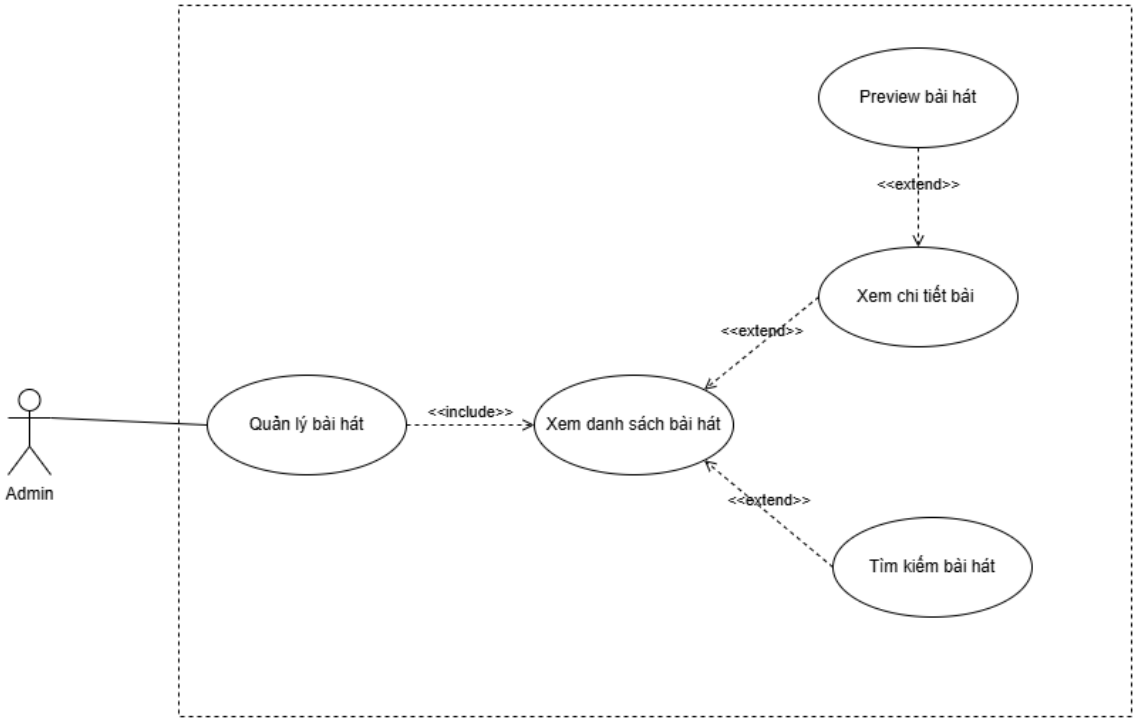


Hình 2.4: Sơ đồ usecase quản lý nghệ sĩ

Bảng 2.3: Đặc tả usecase quản lý nghệ sĩ

Use case quản lý nghệ sĩ			
Tác nhân	Admin		
Mô tả	Admin có thể quản lý tất cả nghệ sĩ, cho phép chấp thuận hoặc từ chối yêu cầu tạo hồ sơ cho nghệ sĩ		
Điều kiện trước	Phải đăng nhập vào hệ thống, chọn mục [Artist] [Artist Request]		
Điều kiện sau			
Kịch bản		**Hành động của tác nhân**	**Hành động của hệ thống**
	1	Đăng nhập và nhấn nút [Artist] và [Artist Request] ở thanh menu	Toàn bộ thông tin của nghệ sĩ hoặc yêu cầu tạo tài khoản của nghệ sĩ được hiển thị
	2	Bấm nút xem chi tiết nghệ sĩ	Chuyển hướng sang trang hồ sơ của nghệ sĩ, hiển thị thông tin của cá nhân và các sản phẩm gần đây nhất của họ
	3	Bấm nút chấp nhận yêu cầu tạo hồ sơ nghệ sĩ	Thực hiện chấp thuận việc tạo hồ sơ nghệ sĩ và hiển thị thông báo

	4	Bấm nút từ chối yêu cầu tạo hồ sơ nghệ sĩ	Thực hiện từ chối việc tạo hồ sơ nghệ sĩ và hiển thị thông báo
	5	Nhập tên nghệ sĩ hoặc email của họ vào thanh tìm kiếm	Hiện thị danh sách các nghệ sĩ tương ứng

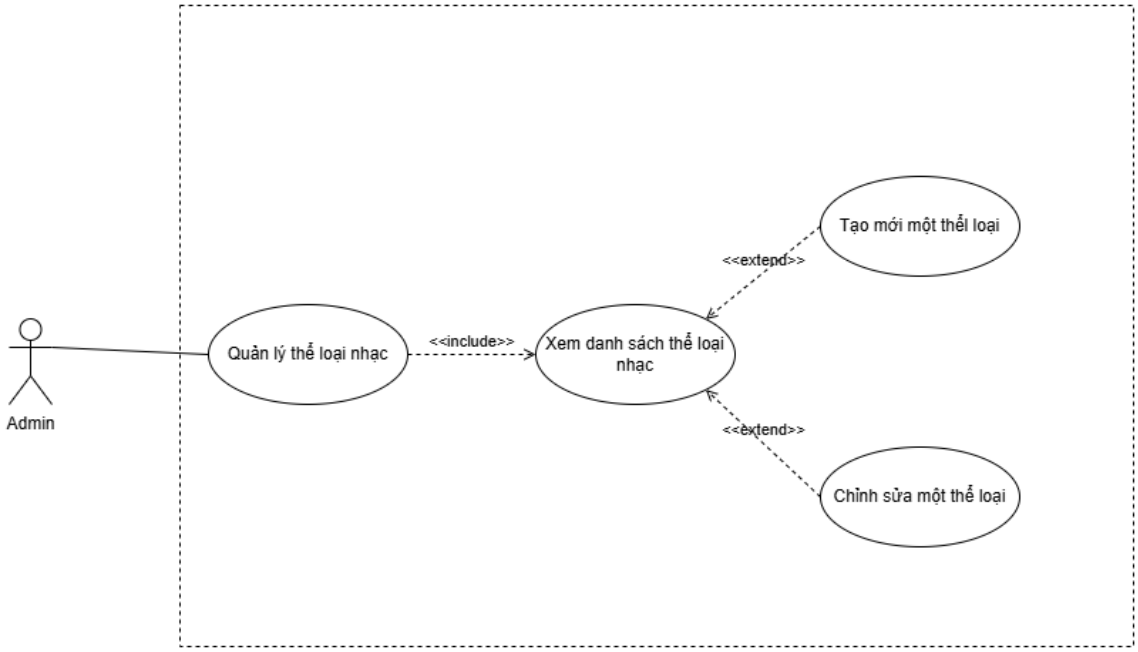


Hình 2.5: Sơ đồ usecase quản lý bài hát

Bảng 2.4: Đặc tả usecase quản lý bài hát

Use case quản lý bài hát			
Tác nhân	Admin		
Mô tả	Admin có thể quản lý tất cả các bài hát được tải lên hệ thông		
Điều kiện trước	Đăng nhập vào hệ thống, chọn mục [Track]		
Điều kiện sau			
Kịch bản		**Hành động của tác nhân**	**Hành động của hệ thống**
	1	Nhấn bài [Track] bên thanh menu	Danh sách các bài hát được hiển thị ra màn hình
	2.1	Bấm vào nút xem chi tiết bài hát của một bài hát bất	Chuyển hướng tới trang xem thông tin chi tiết bài hát

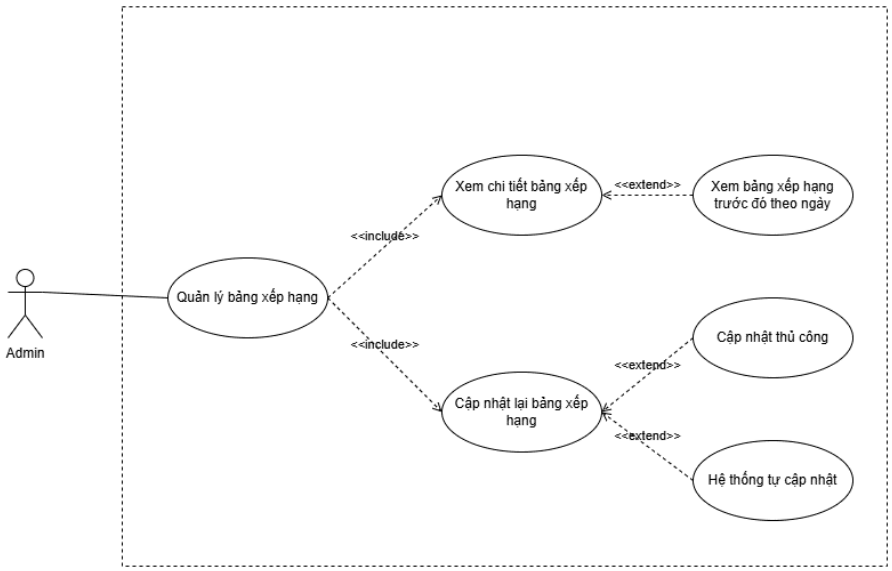
		kì	
	2.2	Chọn nút phát nhạc để preview	Hiển thị một modal lên màn hình, có các control để điều khiển nghe, dừng, tua bài hát.
	2.3	Click nút [play]	Phát bài hát
	2.4	Click nút [pause]	Tạm dừng bài hát
	2.5	Click vào nút [forward]	Tua nhanh bài hát thêm 10s
	2.6	Click vào nút [rewind]	Tua lui bài hát đi 10s
	3	Nhập nội dùng cần tìm kiếm vào thanh search	Lọc các bài hát có tiêu đề phù hợp với kí tự search và hiển thị ra ngoài màn hình



Hình 2.6: Sơ đồ usecase quản lý thể loại nhạc (genre)

Bảng 2.5: Đặc tả usecase quản lý thể loại nhạc (genre)

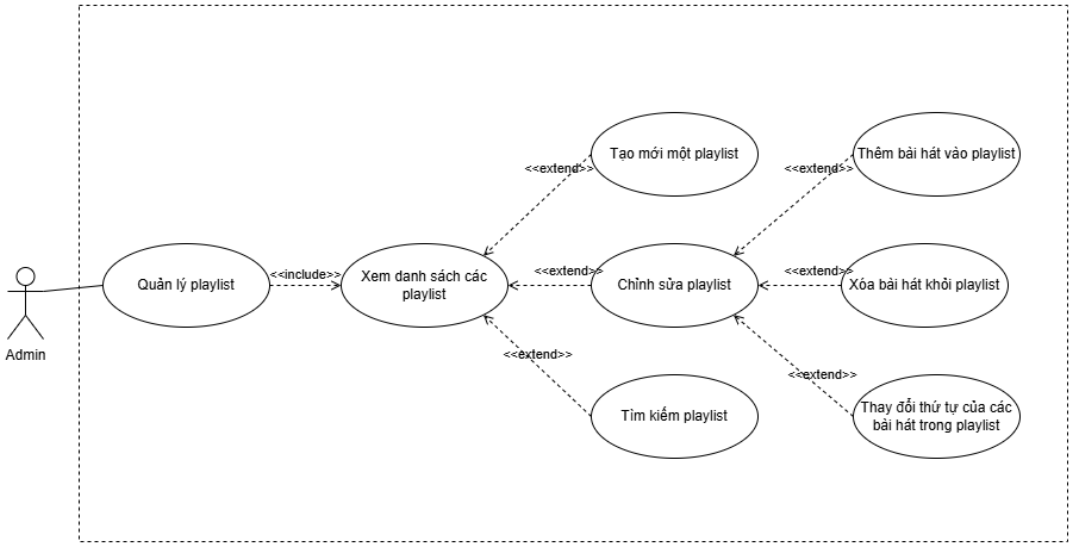
Use case quản lý thể loại nhạc (genre)			
Tác nhân	Admin		
Mô tả	Admin có thể quản lý thể loại nhạc, có thể tạo mới và chỉnh sửa		
Điều kiện trước	Đăng nhập vào hệ thống với role là admin, chọn mục [Genre] bên thanh menu		
Điều kiện sau			
Kịch bản		**Hành động của tác nhân**	**Hành động của hệ thống**
	1	Đăng nhập và nhấn nút [Genre] bên thanh menu	Danh sách tất cả các thể loại nhạc sẽ được hiển thị
	2.1	Bấm nút [Create Genre]	Form thêm genre được hiện ra
	2.2	Nhập tên genre cần tạo và bấm nút [save]	Tạo mới một genre, ấn form và hiện thị thông báo
	2.3	Chọn nút [cancel]	Ẩn form tạo mới đi
	3.1	Bấm nút chỉnh sửa bên cạnh hàng tương ứng của mỗi genre	Form chỉnh sửa genre được hiển thị ra, thông tin của genre được fill vào form.
	3.2	Chỉnh sửa thông tin của genre và bấm nút [save]	Cập nhật lại thông tin của genre, ấn form và hiện thị thông báo
	3.3	Chọn nút [cancel]	Ẩn form cập nhật đi



Hình 2.7: Sơ đồ usecase quản lý bảng xếp hạng của admin

Bảng 2.6: Đặc tả usecase quản lý bảng xếp hạng

Use case quản lý bảng xếp hạng của admin			
Tác nhân	Admin		
Mô tả	Admin có thể quản lý bảng xếp hạng các bài hát. Có thể xem chi tiết từng ngày và cập nhật chúng một cách thủ công hoạt tự động		
Điều kiện trước	Đăng nhập vào hệ thống với role là admin, chọn mục [Chart] bên thanh menu		
Điều kiện sau			
Kịch bản		**Hành động của tác nhân**	**Hành động của hệ thống**
	1	Đăng nhập và nhấn nút [Chart] bên thanh menu	Chuyển hướng tới trang quản lý bảng xếp hạng. Tại đây hiển thị bảng xếp hạng của ngày gần nhất
	2	Bấm nút [Update Chart]	Gọi API cập nhật lại bảng xếp hạng và tải lại trang
	3.1	Bấm nút [View Chart]	Chuyển hướng tới trang bảng xếp hạng riêng.
	3.2	Chọn một ngày bất kì ở DateTimePicker	Hiển thị bảng xếp hạng của đúng ngày được chọn, nếu ngày đó không tồn tại bảng xếp hạng, hiển thị trang PageNotFound

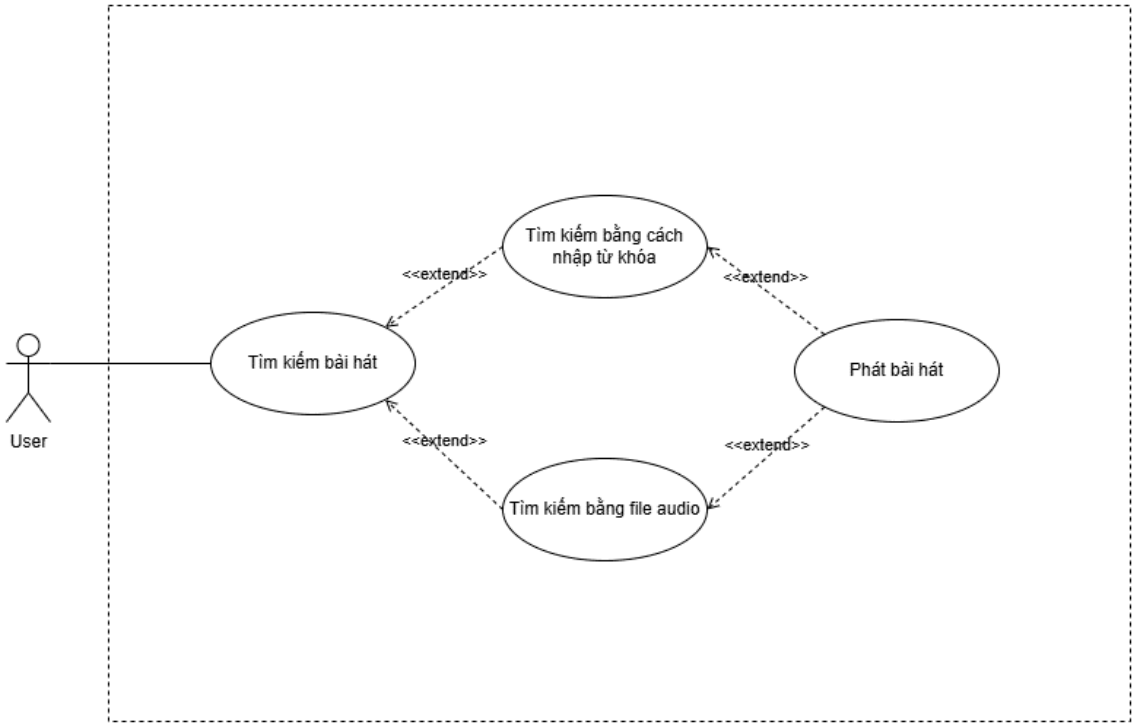


Hình 2.8: Sơ đồ usecase quản lý playlist

Bảng 2.7: Đặc tả usecase quản lý playlist

Use case quản lý playlist			
Tác nhân	Admin		
Mô tả	Admin có thể quản lý playlist, có thể tạo mới và chỉnh sửa		
Điều kiện trước	Đăng nhập vào hệ thống với role là admin, chọn mục [Featured Playlist] bên thanh menu		
Điều kiện sau			
Kịch bản		**Hành động của tác nhân**	**Hành động của hệ thống**
	1	Đăng nhập và nhấn nút [Featured] bên thanh menu	Danh sách tất cả các playlist nhạc sẽ được hiển thị
	2.1	Bấm nút [Create Nẻw Playlist]	Form thêm playlist được hiện ra
	2.2	Nhập tên playlist cần tạo, upload ảnh của playlist và bấm nút [create]	Tạo mới một playlist trống, hiện thị thông báo
	2.3	Chọn nút [cancel]	Ẩn form tạo mới đi
	3.1	Bấm nút chỉnh sửa bên cạnh hàng tương ứng của mỗi playlist	Form chỉnh sửa playlist được hiển thị ra, thông tin của playlist được fill vào form.
	3.2	Chỉnh sửa thông tin của	Cập nhật lại thông tin của

		playlist và bấm nút [save]	playlist, ấn form và hiện thị thông báo
	3.3	Chọn nút [cancel]	Ẩn form cập nhật đi
	4	Nhập tên playlist cần tìm kiếm vào thanh search	Các playlist phù hợp sẽ hiện thị lên màn hình
	5	Bấm vào nút xem chi tiết bên cạnh hàng tương ứng của mỗi playlist	Điều hướng tới trang xem chi tiết playlist
	5.1.1	Chọn nút [Add track]	Một modal được hiển thị để tìm kiếm bài hát
	5.1.2	Nhập tên bài hát cần tìm kiếm vào thanh search và chọn nút [Search]	Thông tin các bài hát phù hợp sẽ được hiển thị ra
	5.1.3	Chọn các bài hát cần thêm vào chọn nút [Apply]	Bài hát sẽ tạm thời được thêm vào playlist, ẩn model đi
	5.1.4	Chọn nút lưu	Tất cả các bài hát sẽ được lưu
	5.2.1	Chọn các bài hát cần xóa khỏi playlist bằng cách tick vào checkbox ở đầu hàng	Các row của các bài hát được chọn sẽ được highlight
	5.2.2	Click vào nút xóa	Bài hát được chọn tạm thời sẽ bị xóa khỏi playlist
	5.2.3	Chọn nút lưu	Bài hát được chọn sẽ vĩnh viễn xóa khỏi playlist
	5.3.1	Chọn các bài hát thay đổi vị trí playlist bằng cách tick vào checkbox ở đầu hàng	Các row của các bài hát được chọn sẽ được highlight
	5.3.2	Nhấn vào các nút [up] hoặc [down] để di chuyển bài hát	Thứ tự của các bài hát sẽ tạm thời thay đổi
	5.3.3	Chọn nút lưu	Thứ tự mới của các bài hát sẽ được lưu
	6	Nhập tên của các playlist vào thanh Search	Các playlist phù hợp sẽ được hiện thị trên màn hình



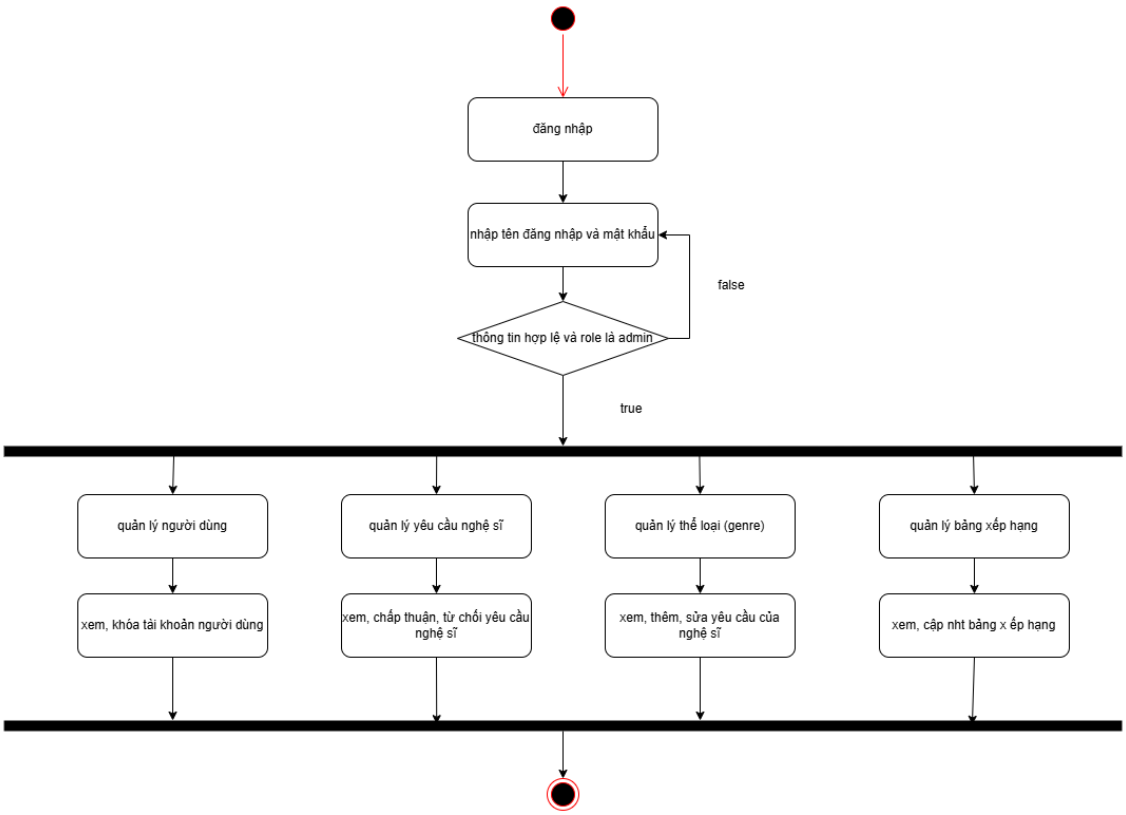
Hình 2.9: Biểu đồ usecase tìm kiếm bài hát

Bảng 2.8: Đặc tả use case tìm kiếm bài hát

Use case tìm kiếm bài hát			
Tác nhân	User		
Mô tả	User có khả năng tìm kiếm bài hát		
Điều kiện trước	Đăng nhập vào ứng dụng, vào mục tìm kiếm bài hát		
Điều kiện sau	Hiển thị, phát bài hát nếu tìm được		
Kịch bản		**Hành động của tác nhân**	**Hành động của hệ thống**
	1	Đăng nhập vào mobile và vào trang tìm kiếm	Hiển thị màn hình tìm kiếm, cho phép
	2	Nhập từ khóa muốn tìm kiếm	Hiển thị kết quả tìm kiếm ra màn hình
	3.1	Nhấn vào biểu tượng [audio]	Chuyển hướng tới trang tìm kiếm bằng âm thanh
	3.2	Chọn upload file audio hoặc nhấn nút thu âm	Thực hiện tìm kiếm bằng âm thanh và phát bài hát nếu có.

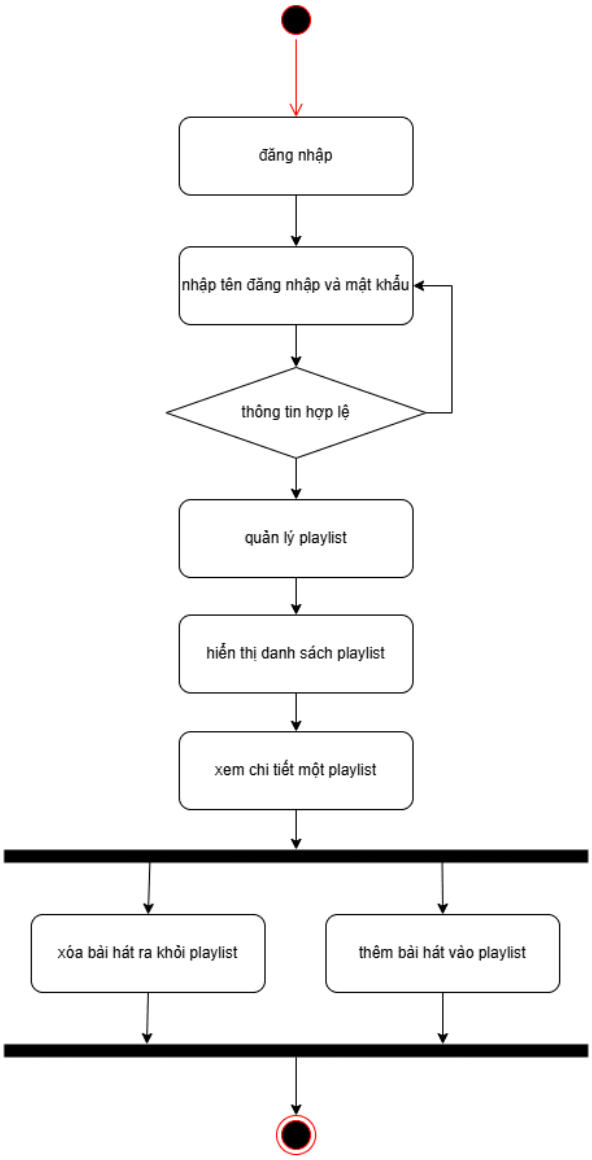
2.4 Biểu đồ hoạt động

2.4.1 Biểu đồ hoạt động chức năng quản lý một vài chức năng của admin



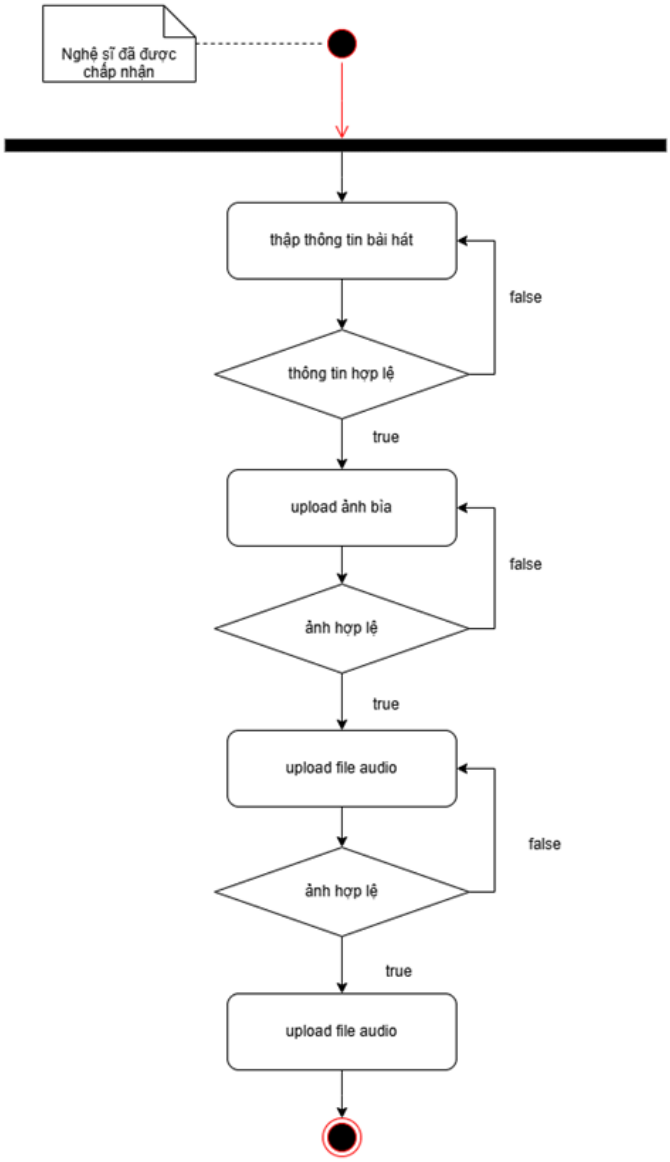
Hình 2.10: Sơ đồ hoạt động chức năng quản lý của admin

2.4.2 Biểu đồ hoạt động chức năng quản lý playlist



Hình 2.11: Sơ đồ hoạt động chức năng quản lý playlist

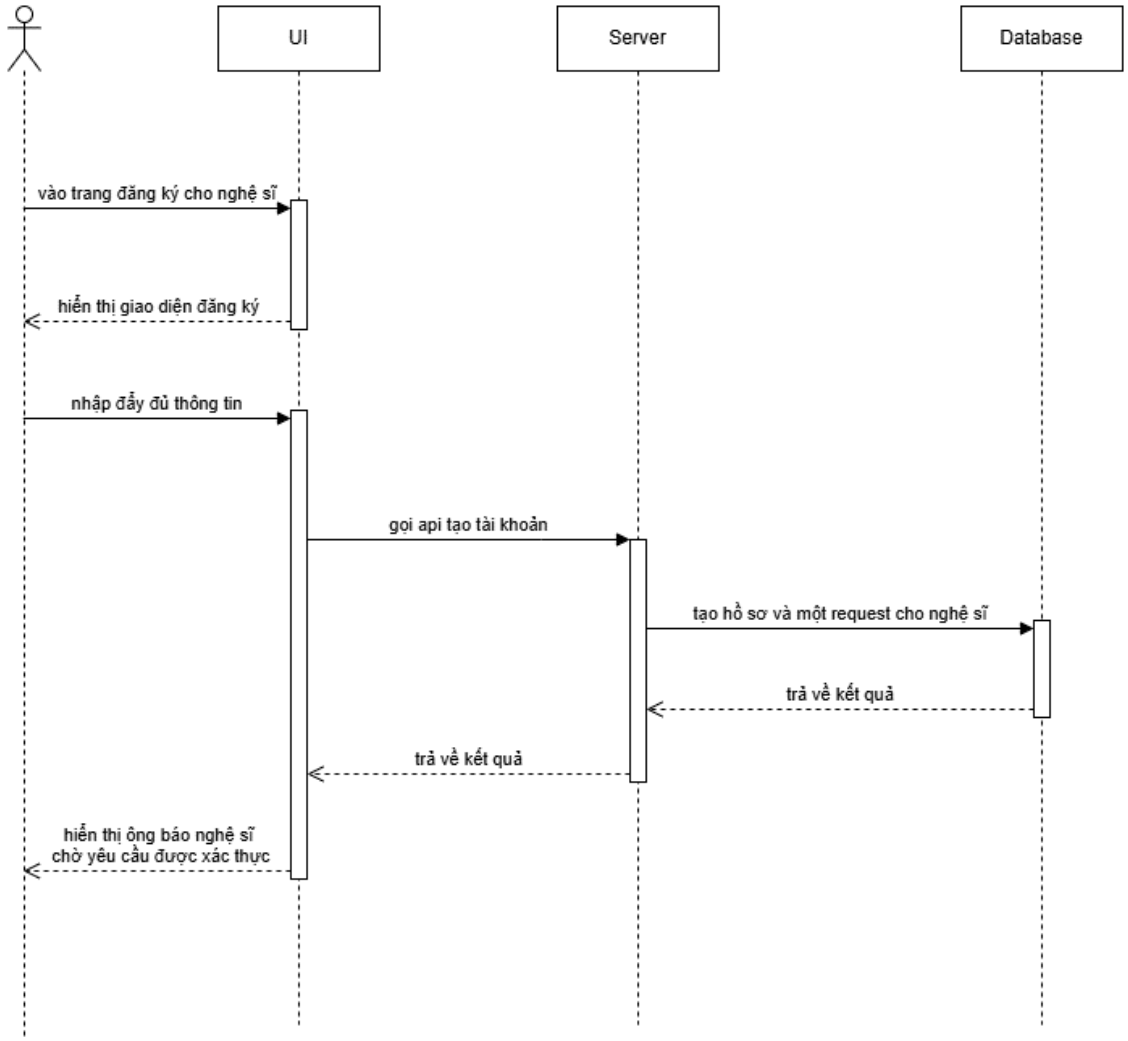
2.4.3 Biểu đồ hoạt động chức năng đăng tải bài hát



Hình 2.12: Sơ đồ hoạt động chức năng đăng tải bài hát

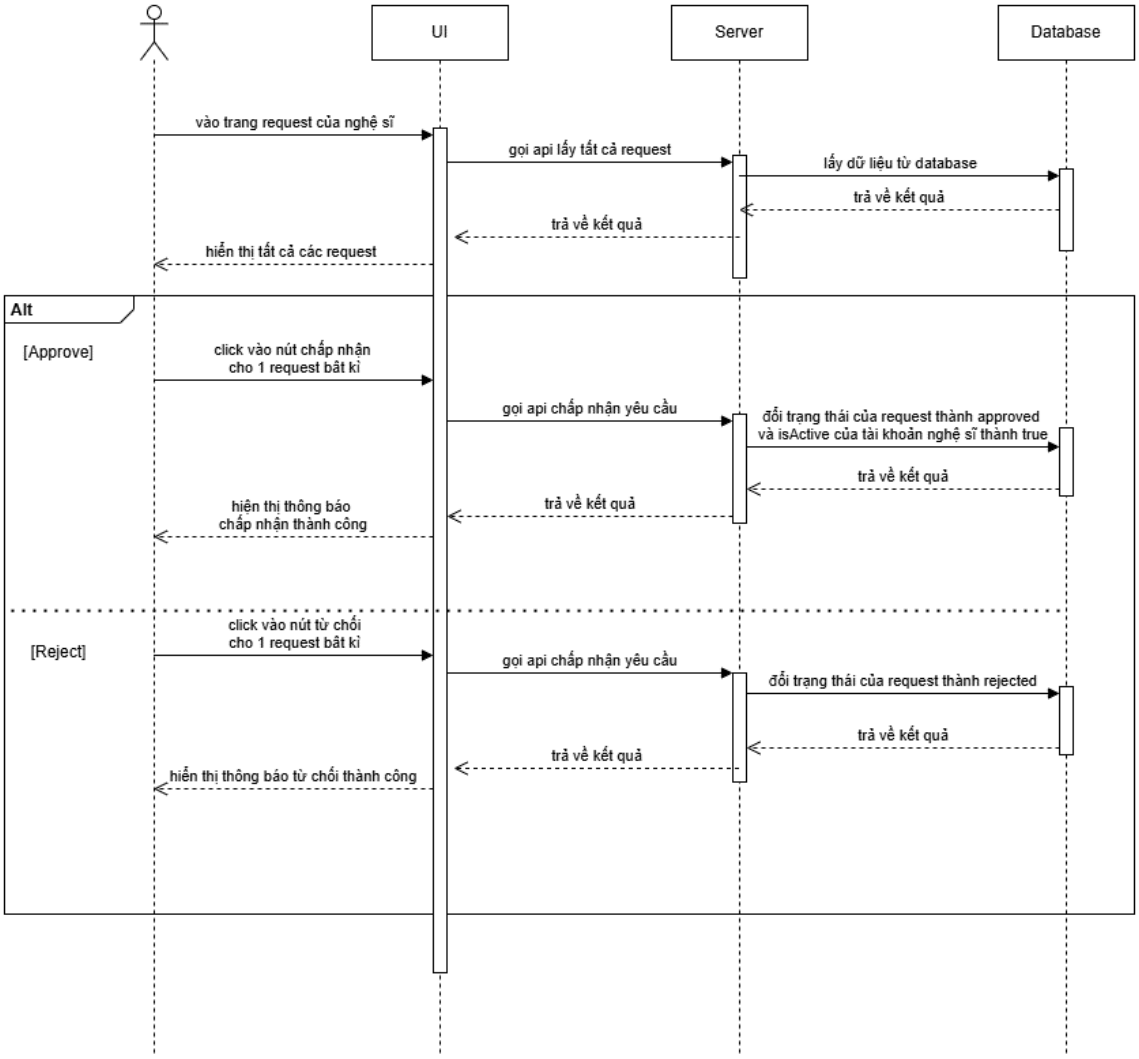
2.5 Biểu đồ tuần tự

2.5.1 Biểu đồ tuần tự chức năng đăng ký tài khoản nghệ sĩ



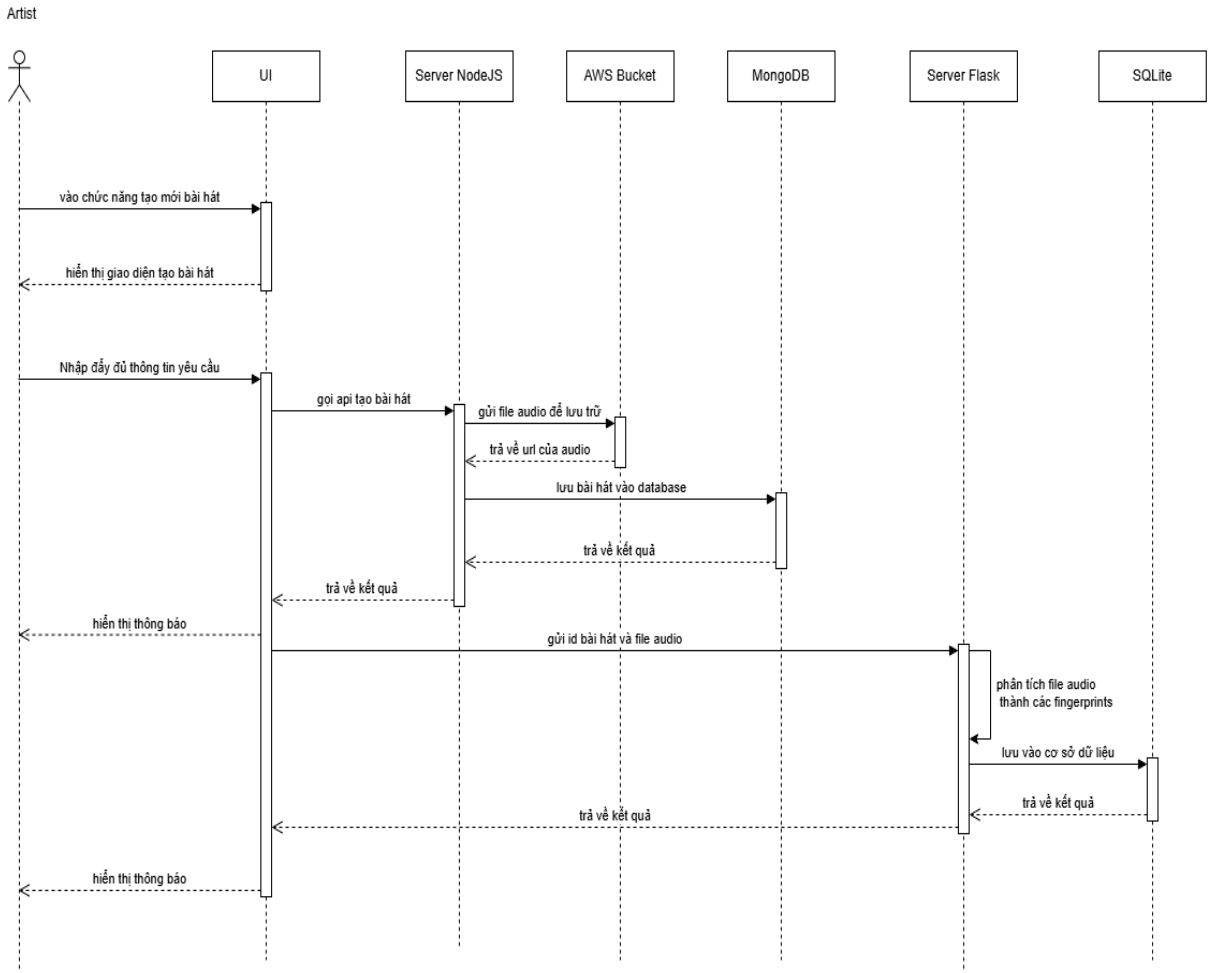
Hình 2.13: Sơ đồ tuần tự chức năng tạo tài khoản nghệ sĩ

2.5.2 Biểu đồ tuần tự chức năng quản lý yêu cầu tạo hồ sơ nghệ sĩ



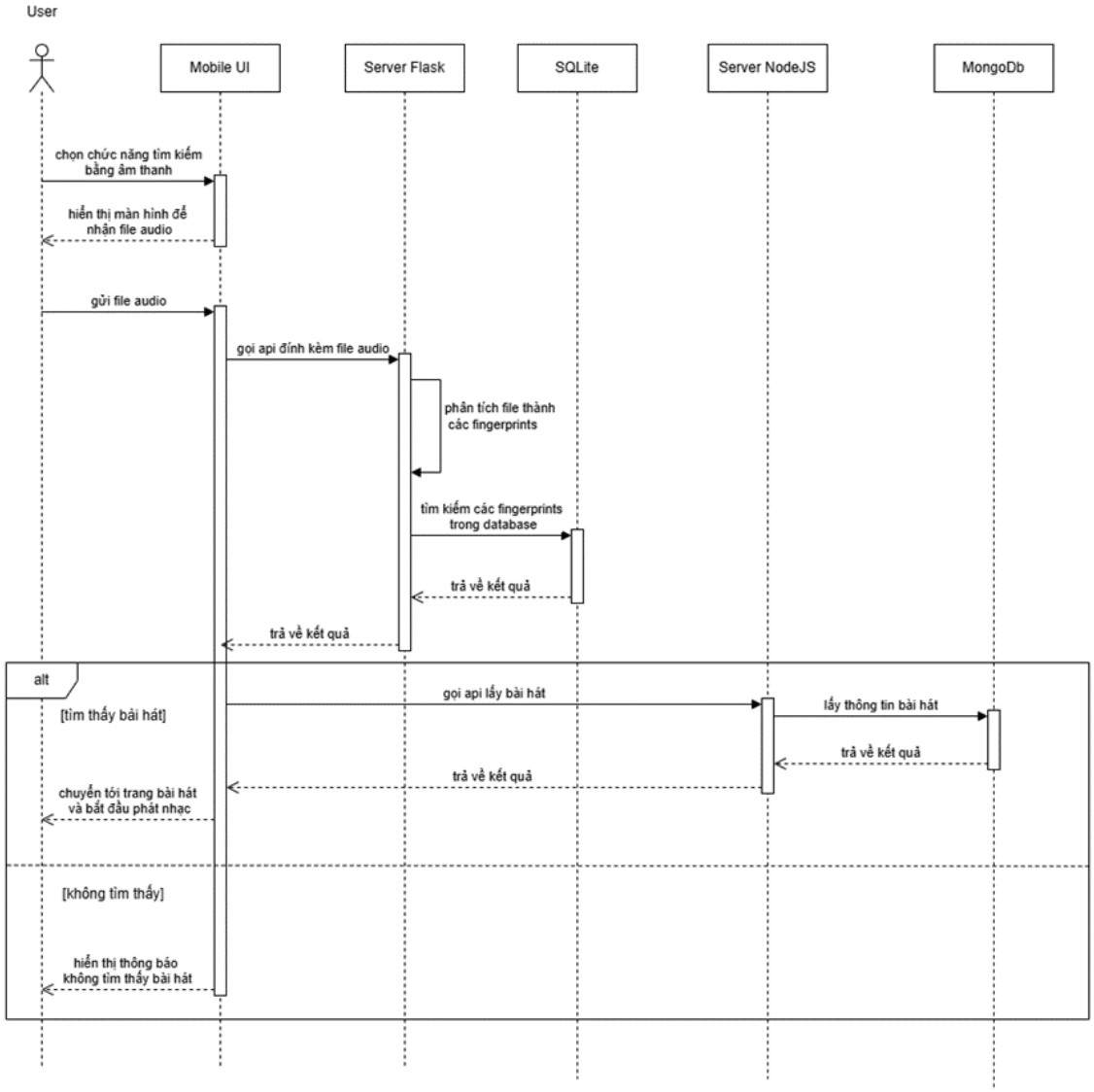
Hình 2.14: Biểu đồ tuần tự quản lý yêu cầu tạo hồ sơ nghệ sĩ

2.5.3 Biểu đồ tuần tự chức năng tải nhạc lên của nghệ sĩ



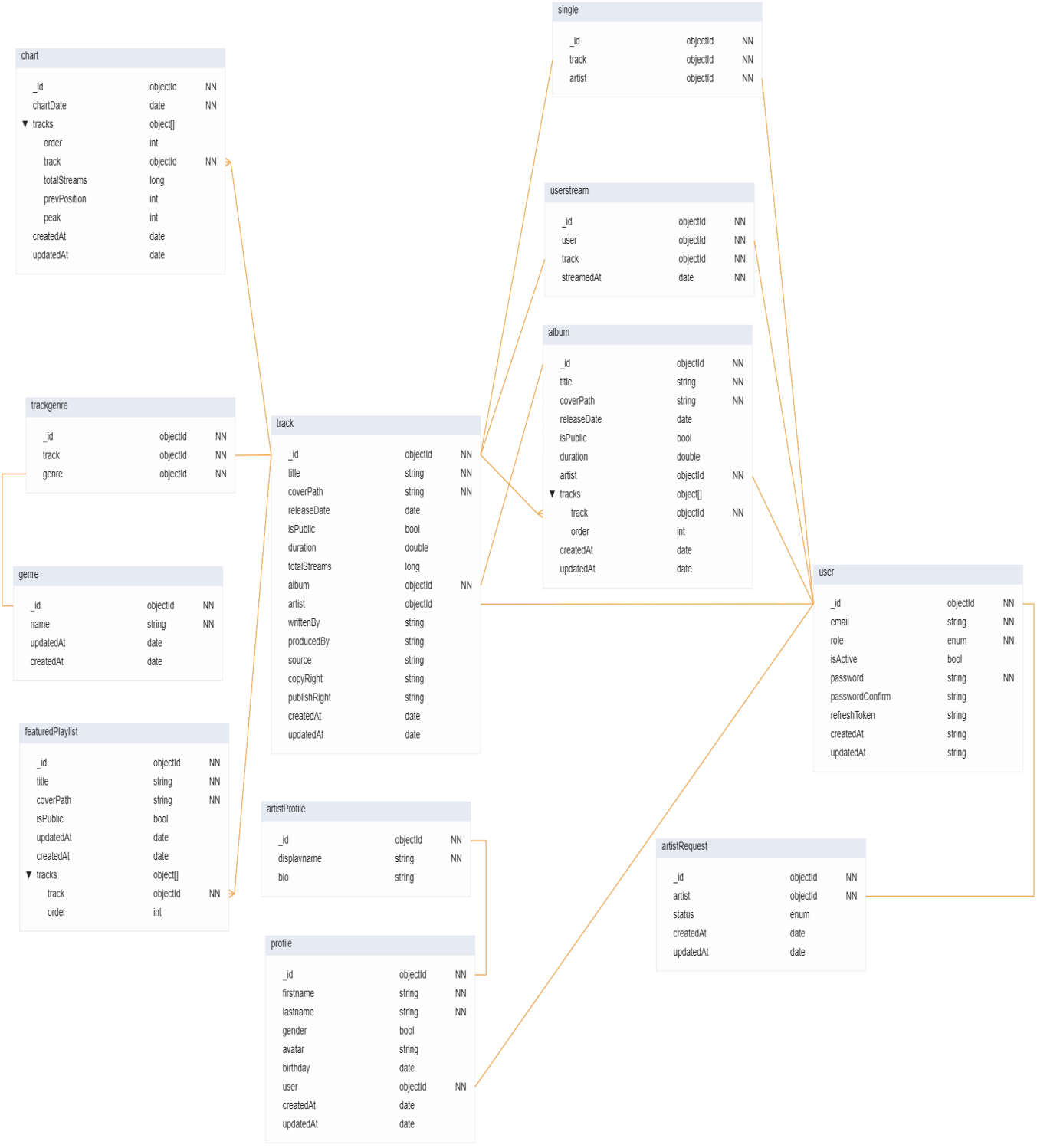
Hình 2.15: Sơ đồ tuần tự chức năng tải nhạc lên của nghệ sĩ

2.5.4 Biểu đồ tuần tự chức năng tìm kiếm bài hát bằng âm thanh



Hình 2.16: Sơ đồ tuần tự chức năng tìm kiếm bài hát bằng âm thanh

2.6 Cơ sở dữ liệu



Hình 2.17: Biểu đồ cơ sở dữ liệu NoSQL của hệ thống chung

Các bảng trong cơ sở dữ liệu:

Bảng 2.9: Thông tin bảng User

Tên trường	Ý nghĩa	Chú thích
email	email	
password	mật khẩu	
isActive	người dùng được phép vào hệ thống hay không	
refreshToken	token	
role	role để phân quyền	Có 3 role: admin, artist và user
createdAt	thời điểm tạo tài khoản	
updatedAt	thời điểm gần nhất tài khoản thay đổi	

Bảng 2.10: Thông tin bảng Profile

Tên trường	Ý nghĩa	Chú thích
firstname	firstname	
lastname	lastname	
gender	giới tính	
avatar	đường dẫn tới avatar	ảnh được lưu trữ trên aws
birthday	ngày sinh của người dùng	
user	chủ sở hữu của hồ sơ	khóa ngoại
createdAt	thời điểm tạo hồ sơ	
updatedAt	thời điểm gần nhất hồ sơ thay đổi	

Bảng 2.11: Thông tin bảng ArtistProfile

Tên trường	Ý nghĩa	Chú thích
displayname	tên hiển thị của nghệ sĩ	
bio	tiểu sử của nghệ sĩ	
createdAt	thời điểm tạo tài khoản	
updatedAt	thời điểm gần nhất tài khoản thay đổi	

Bảng 2.12: Thông tin bảng ArtistRequest

Tên trường	Ý nghĩa	Chú thích
artist	id của nghệ sĩ	
status	trạng thái của request	có 3 status: pending, approved, rejected
createdAt	thời điểm tạo request	
updatedAt	thời điểm gần nhất request thay đổi	

Bảng 2.13: Thông tin bảng Genre

Tên trường	Ý nghĩa	Chú thích
name	tên thể loại	
createdAt	thời điểm tạo genre	
updatedAt	thời điểm gần nhất genre thay đổi	

Bảng 2.14: Thông tin bảng Album

Tên trường	Ý nghĩa	Chú thích
title	tiêu đề album	
coverPath	ảnh cover của album	
artist	nghệ sĩ phát hành album	
tracks	mảng chứa thứ tự của bài hát và id của các bài hát trong album	[track: {order; track}]
createdAt	thời điểm tạo album	
updatedAt	thời điểm gần nhất album thay đổi	

Bảng 2.15: Thông tin bảng Single

Tên trường	Ý nghĩa	Chú thích
track	id của bài hát	khóa ngoại
artist	nghệ sĩ phát hành đĩa đơn	khóa ngoại
createdAt	thời điểm tạo single	
updatedAt	thời điểm gần nhất single thay đổi	

Bảng 2.16: Thông tin bảng FeaturedPlaylist

Tên trường	Ý nghĩa	Chú thích
title	tiêu đề playlist	
coverPath	ảnh cover của playlist	
tracks	mảng chứa thứ tự của bài hát và id của các bài hát trong playlist	[track: {order; track}]
createdAt	thời điểm tạo genre	
updatedAt	thời điểm gần nhất genre thay đổi	

Bảng 2.17: Thông tin bảng bài hát

Tên trường	Ý nghĩa	Chú thích
title	tiêu đề bài hát	
url	đường dẫn tới file nhạc	file audio được lưu ở aws bucket
coverPath	ảnh cover của bài hát	
releaseDate	ngày phát hành của bài hát	
duration	thời lượng của bài hát	
totalStream	tổng số lượt nghe của bài hát	
writtenBy	người sáng tác bài hát	
producedBy	người sản xuất bài hát	
source	nguồn của bài hát	
copyRight	bản quyền sỡ hữu bài hát	
publishRight	quyền phát hành bài hát5	
album	album và hát hát thuộc về	khóa ngoại
artist	nghệ sĩ sở hữu bài hát	khóa ngoại

Bảng 2.18: Thông tin bảng Join của Track và Genre

Tên trường	Ý nghĩa	Chú thích
track	id của bài hát	khóa ngoại
genre	id của genre	khóa ngoại

Bảng 2.19: Thông tin bảng UserStream

Tên trường	Ý nghĩa	Chú thích
user	id của user nghe nhạc	khóa ngoại
track	id của bài hát	khóa ngoại
streamedAt	thời điểm người dùng nghe bài hát	

Bảng 2.20: Thông tin bảng Chart

Tên trường	Ý nghĩa	Chú thích
chartDate	ngày của bảng xếp hạng	
tracks	mảng chứa thứ tự, id, tổng số stream trong ngày, vị trị trước đó trong bảng xếp hạng và vị trí cao nhất bài hát đạt được của bài hát	[track: {order; track, totalStreams, prevPosition, peak}]
createdAt	thời điểm tạo album	
updatedAt	thời điểm gần nhất album thay đổi	

Hình 2.18: Biểu đồ cơ sở dữ liệu cho hệ thống nhận dạng bài hát bằng Fingerprint

Bảng 2.21: Thông tin bảng track của hệ thống nhận dạng

Tên trường	Ý nghĩa	Chú thích
id		khóa chính
name	tiêu đề của bài hát	khóa ngoại
url	đường dẫn tới file nhạc	file audio được lưu ở aws bucket

Bảng 2.22: Thông tin bảng fingerprint của hệ thống nhận dạng bài hát

Tên trường	Ý nghĩa	Chú thích
id		khóa chính
track_fk	id của bài hát	khóa ngoại
hash	giá trị của fingerprint sau khi đã được hash từ thuật toán	
offset		

2.7 Kết chương

Chương này cung cấp cho người đọc cái nhìn thống quan về chuyên môn, nghiệp vụ mà website/mobile cần phải đáp ứng, mô tả thiết kế, cách triển khai và trình bày một số sơ đồ, biểu đồ, bảng biểu cơ bản để trực quan hóa hệ thống.

CHƯƠNG 3: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ

3.1 Môi trường triển khai

Backend (Server NodeJS):

- Sử dụng VScode để triển khai source code;
- Deploy server NodeJS trên onrender;
- Database Mongo được lưu trữ trên cloud của Atlas;
- Route của server: music-tx16.onrender.com/healthcheck

Backend (Server Flask):

- Sử dụng VScode để triển khai source code;
- Database SQLite được lưu trữ local;
- Server Flask được chạy localhost, dùng ngrok để có thể gọi api từ internet.

Frontend:

- Sử dụng VScode để triển khai source code;
- Deploy ứng dụng trên vercel;
- URL của website: https://soundee.vercel.app/

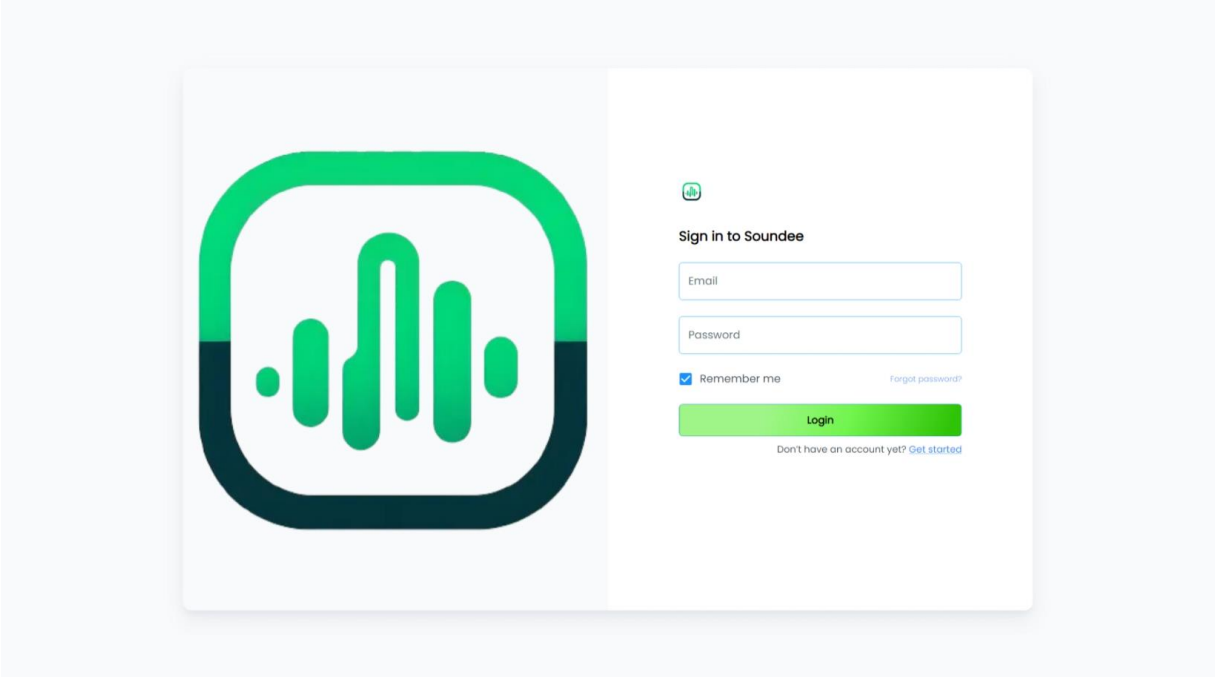
Mobile:

- Sử dụng VScode để triển khai source code;
- Sử dụng Android Studio để khởi chạy emulator.
- Build ứng dụng thành một file apk để chạy trên android.

Toàn bộ source code sẽ được quản lý bởi Git và GitHub.

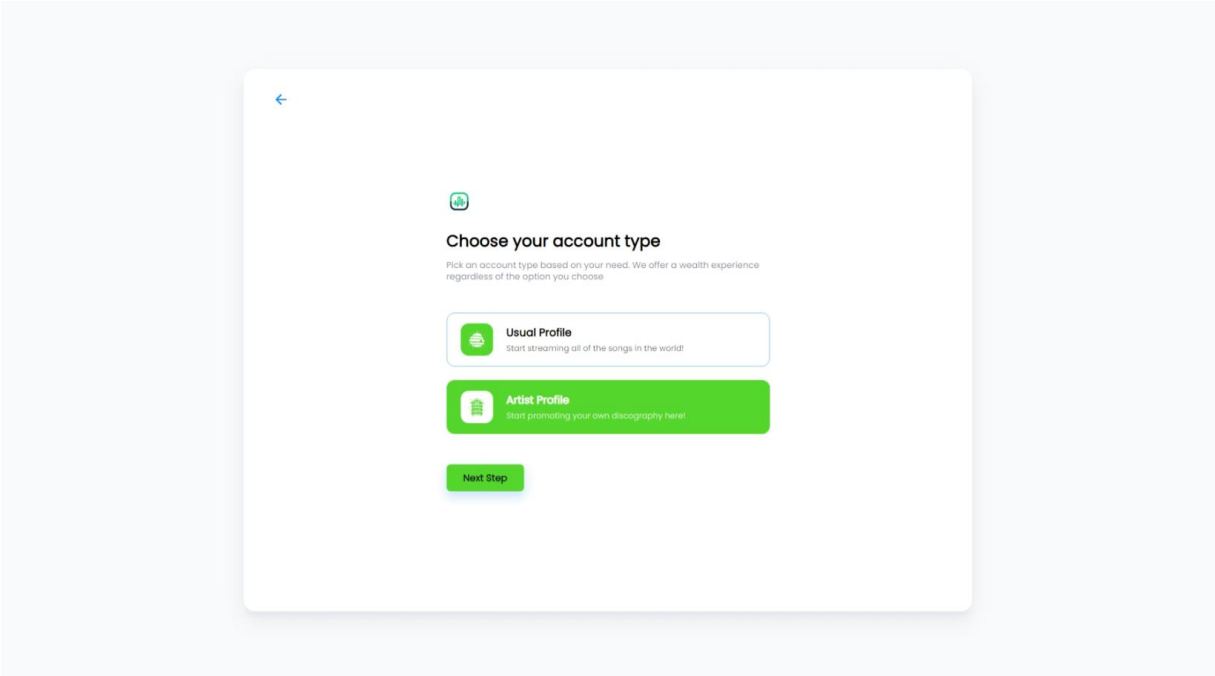
3.2 Kết quả triển khai

3.2.1 Trang web cho admin



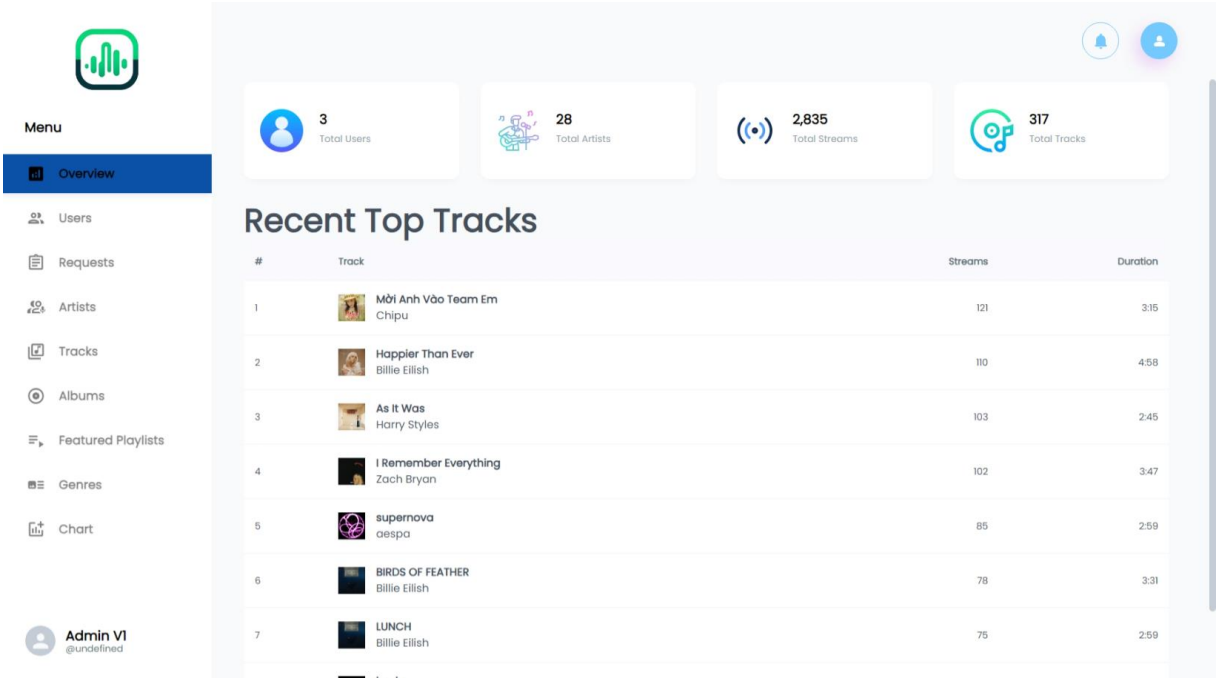
Hình 3.1: Giao diện trang đăng nhập

Đây là giao diện trang đăng nhập cho cả admin và artist. Sau khi login thành công thì sẽ chuyển hướng tới trang overview cho từng role. Nếu đăng nhập sai thông tin hoặc tài khoản chưa được active thì sẽ hiện thông báo.



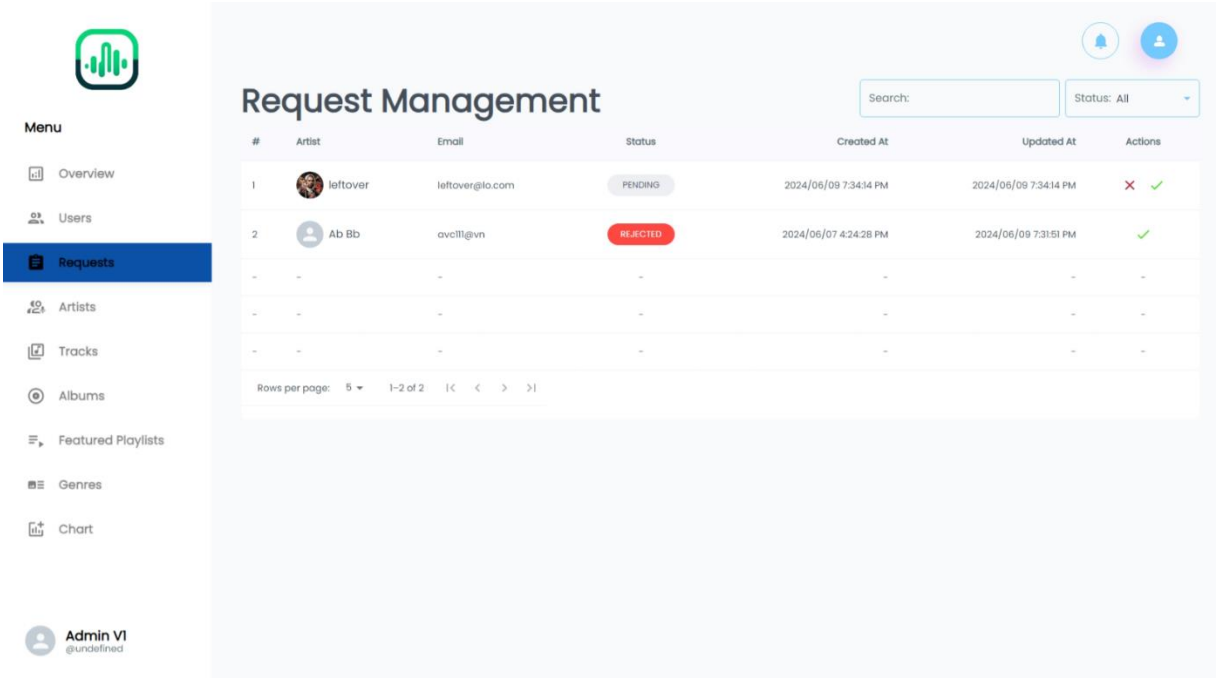
Hình 3.2: Giao diện trang đăng ký tài khoản

Tại giao diện này, người dùng có thể tạo tài khoản với tư cách là nghệ sĩ hoặc người dùng thông thường.



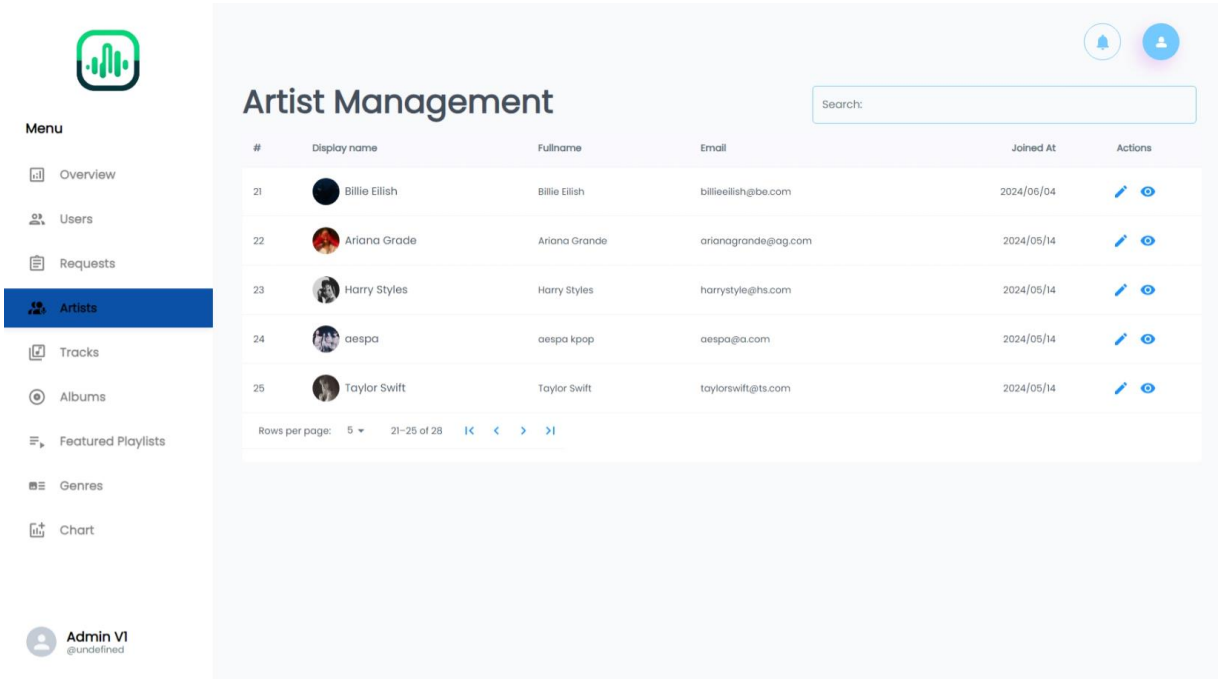
Hình 3.3: Giao diện trang overview của admin

Màn hình trên hiện thị các thông tin tổng quan của hệ thống như là: số người dùng, số nghệ sĩ, tổng số lượt nghe và tổng số bài hát có mặt trên ứng dụng. Ngoài ra bên dưới còn hiển thị lên các bài hát nổi bật trong thời gian gần đây.



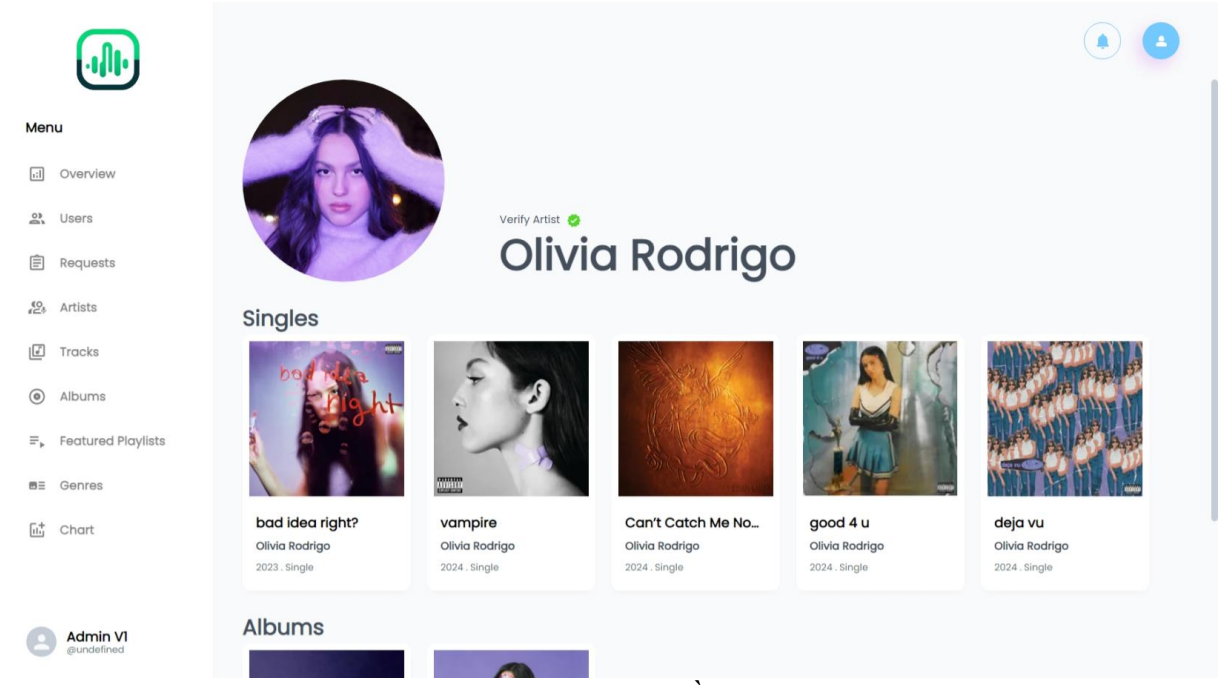
Hình 3.4: Giao diện trang quản lý yêu cầu nghệ sĩ

Màn hình trên hiển thị toàn bộ yêu cầu tạo tài khoản nghệ sĩ. Tại đây admin có quyền chấp nhận hoặc từ chối yêu cầu đó.



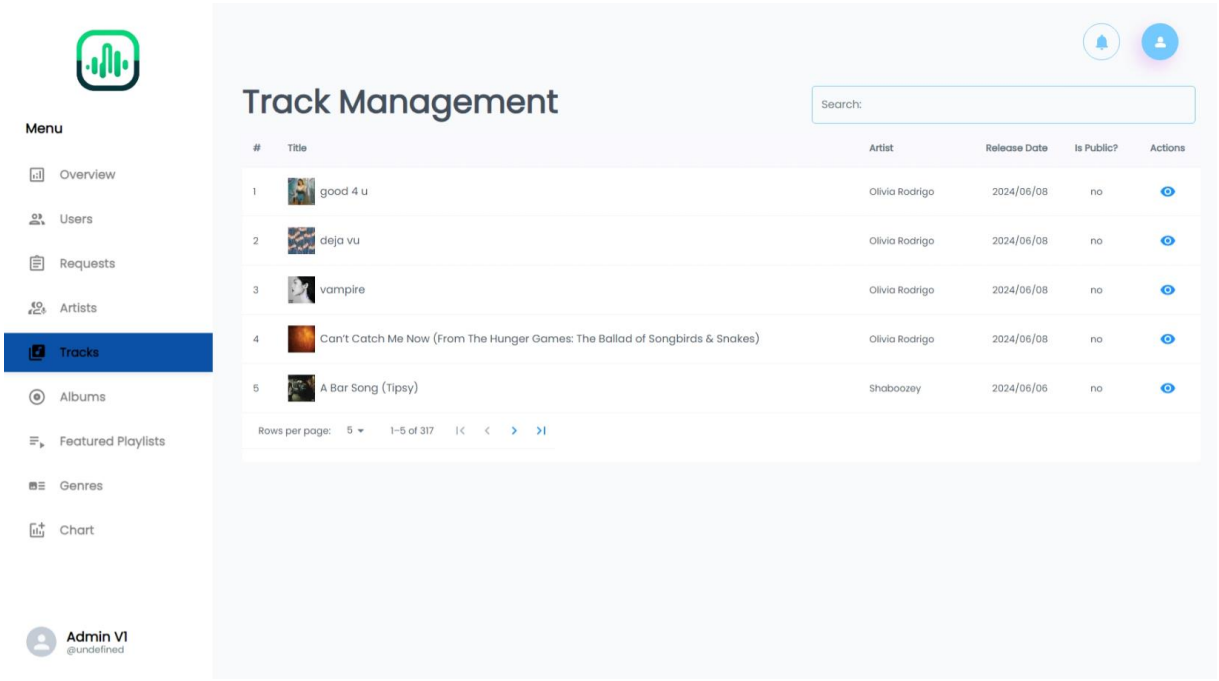
Hình 3.5: Giao diện trang quản lý nghệ sĩ

Màn hình trên hiển thị toàn bộ danh sách nghệ sĩ có mặt trên nền tảng. Tại đây admin có thể tìm kiếm và xem chi tiết hồ sơ của một nghệ sĩ bất kì.



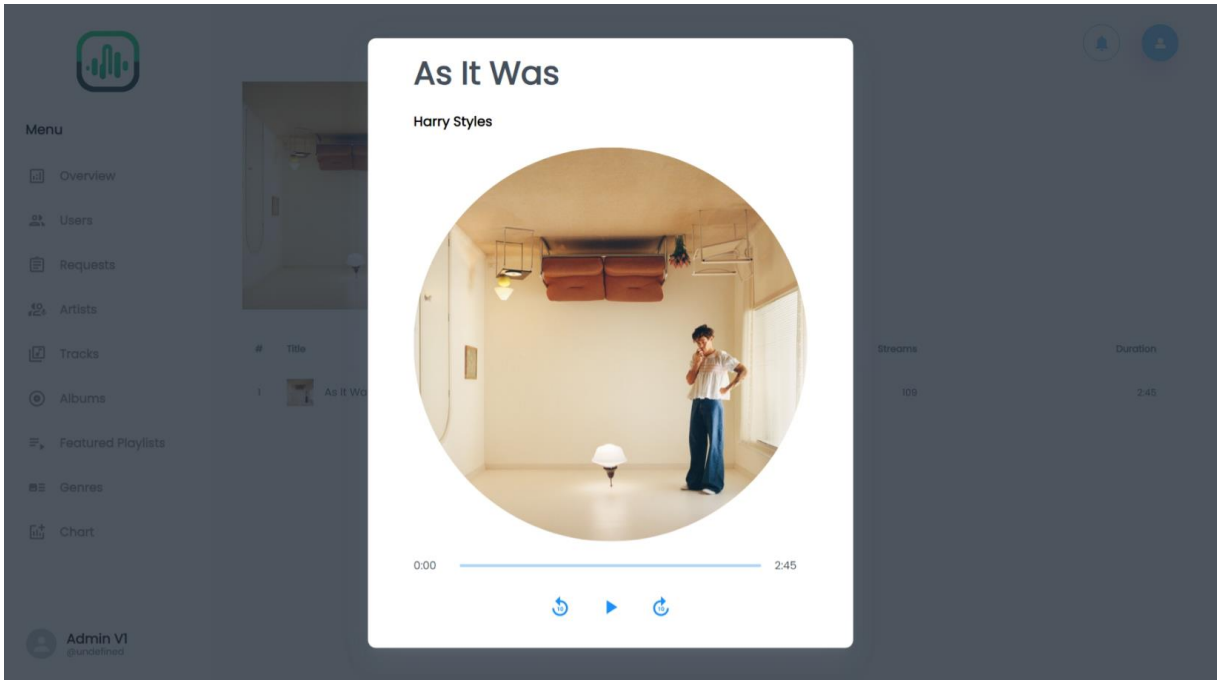
Hình 3.6: Giao diện trang hồ sơ của nghệ sĩ

Màn hình trên hiển thị thông tin cơ bản của nghệ sĩ và toàn bộ sản phẩm âm nhạc của nghệ sĩ đó, có thể là đĩa đơn (single) và album.



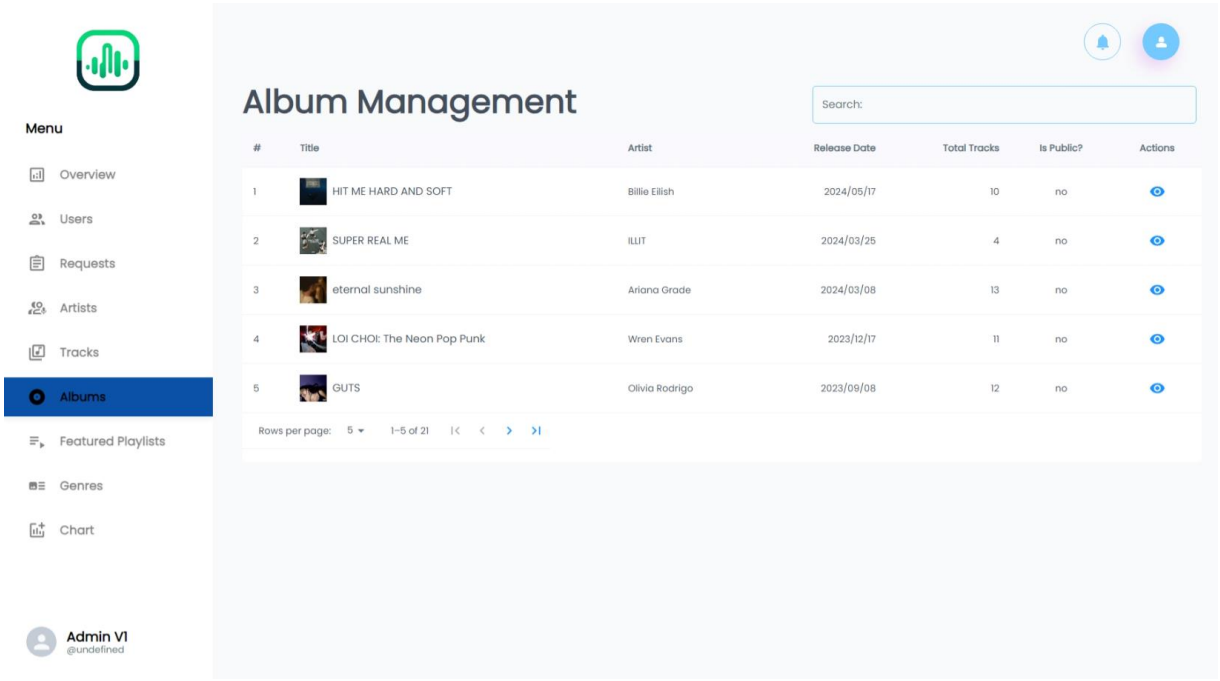
Hình 3.7: Giao diện trang quản lý bài hát

Màn hình trên hiển thị toàn bộ danh sách các bài hát được đăng tải lên nền tảng. Tại đây admin có thể tìm kiếm và xem chi tiết thông của một bài hát.



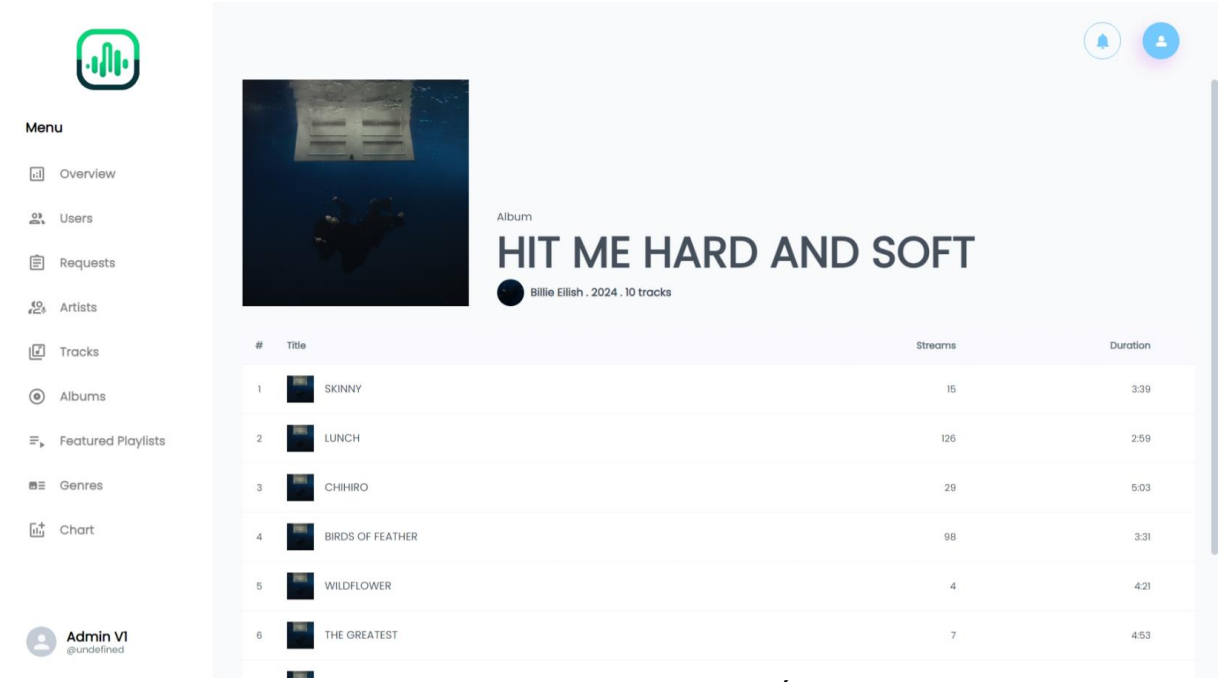
Hình 3.8: Giao diện trang preview nghe thử nhạc

Màn hình trên hiển thị thông tin chi tiết của bài hát. Tại đây cũng có thể nghe thử bài hát, tạm dừng, tua nhanh và tua lùi.



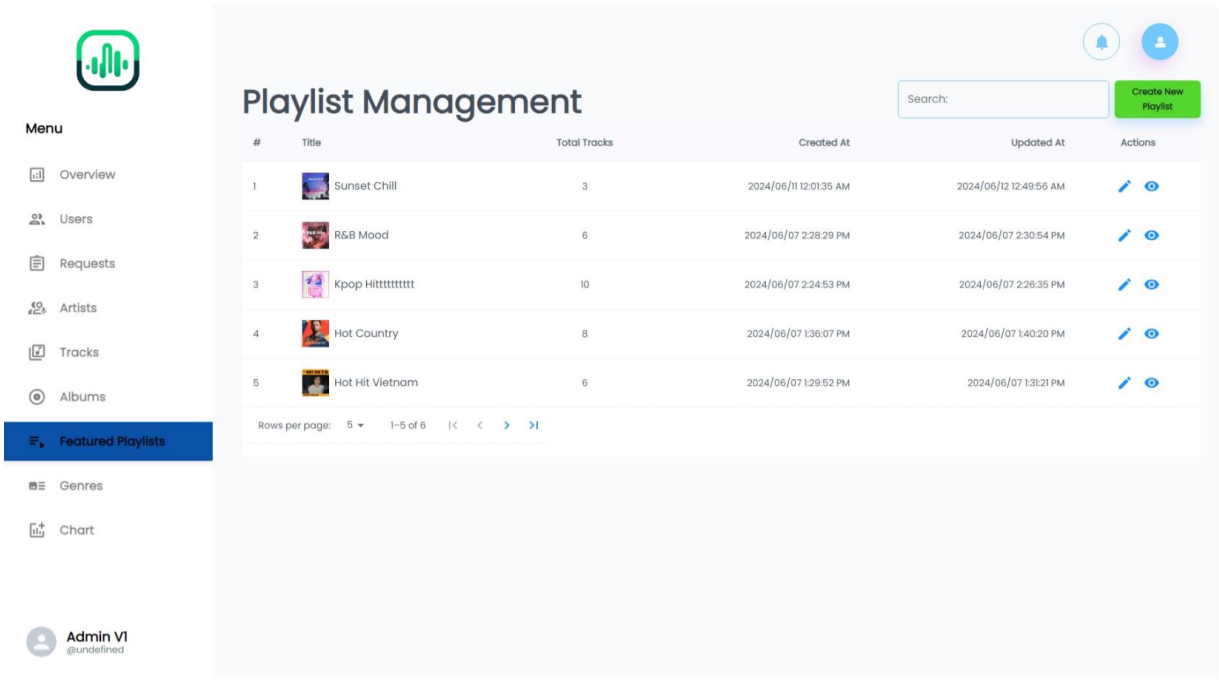
Hình 3.9: Giao diện trang quản lý album

Màn hình trên hiển thị toàn bộ danh sách album do nghệ sĩ tạo. Tại đây admin có thể tìm kiếm và xem chi tiết một album bất kì.



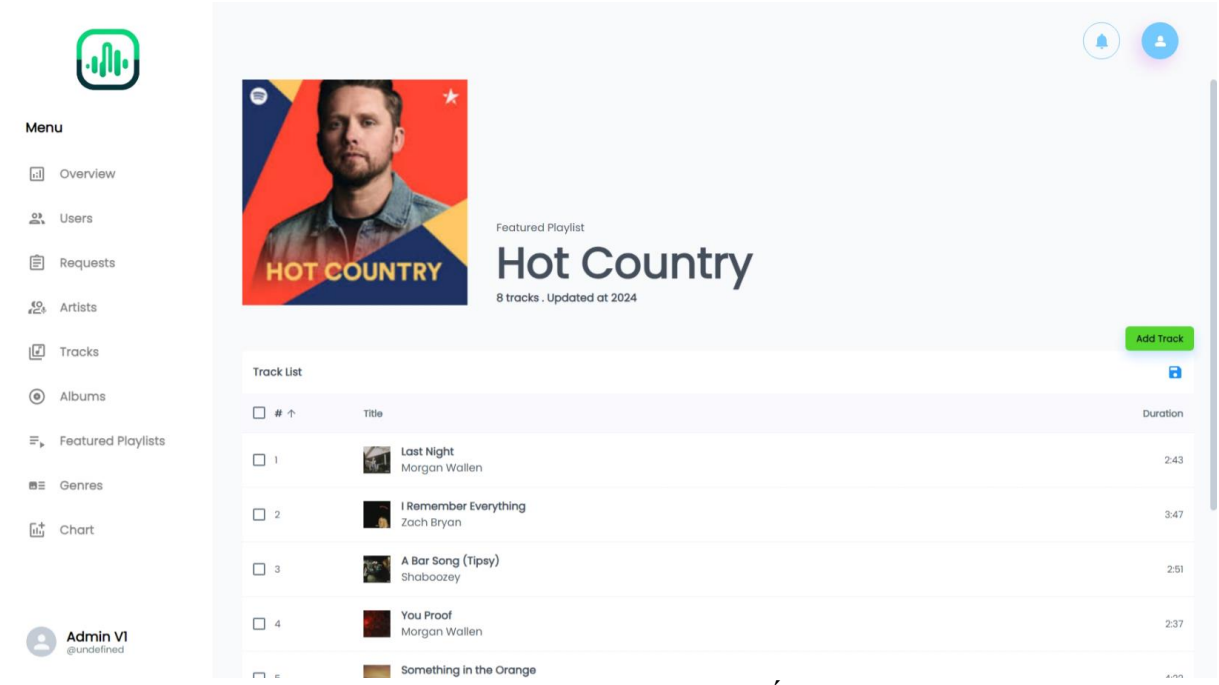
Hình 3.10: Giao diện trang chi tiết album

Màn hình trên hiển thị thông tin chi tiết của một album, bao gồm toàn bộ các bài hát đơn có trong album. Tại đây cũng có thể nghe thử một bài hát bất kì.



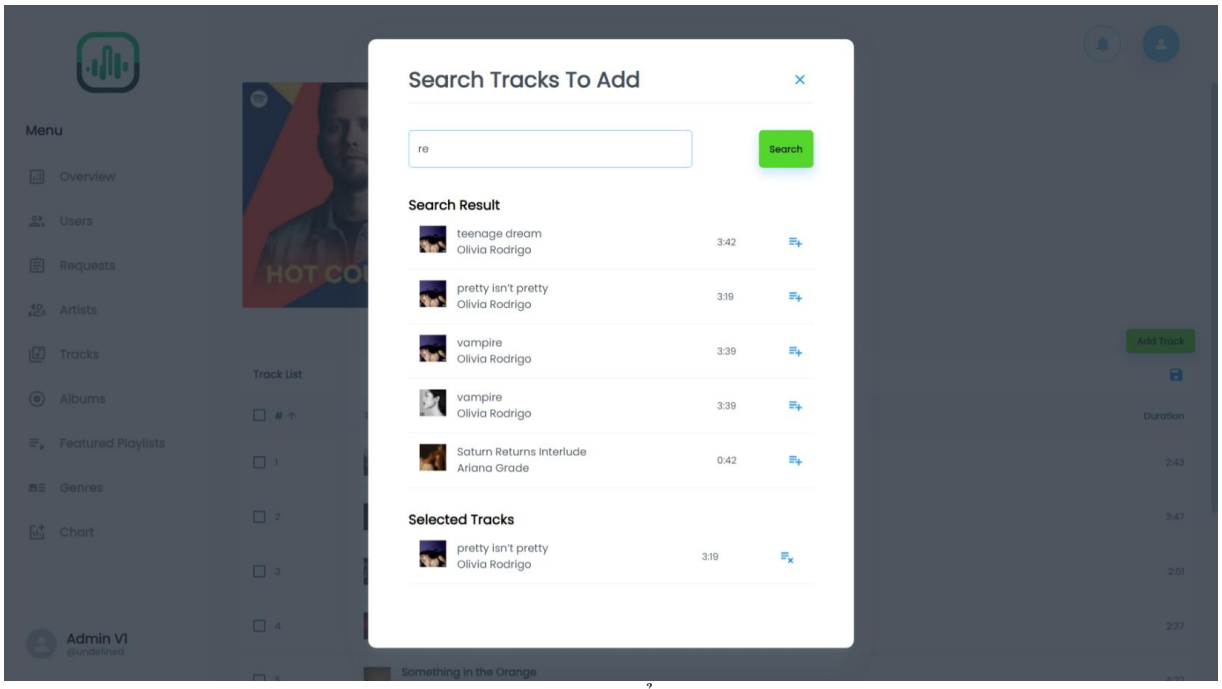
Hình 3.11: Giao diện trang quản lý playlist

Màn hình trên hiển thị toàn bộ danh sách các playlist do admin tạo. Tại đây admin có thể tìm kiếm và xem chi tiết và chỉnh sửa một playlist bất kì.



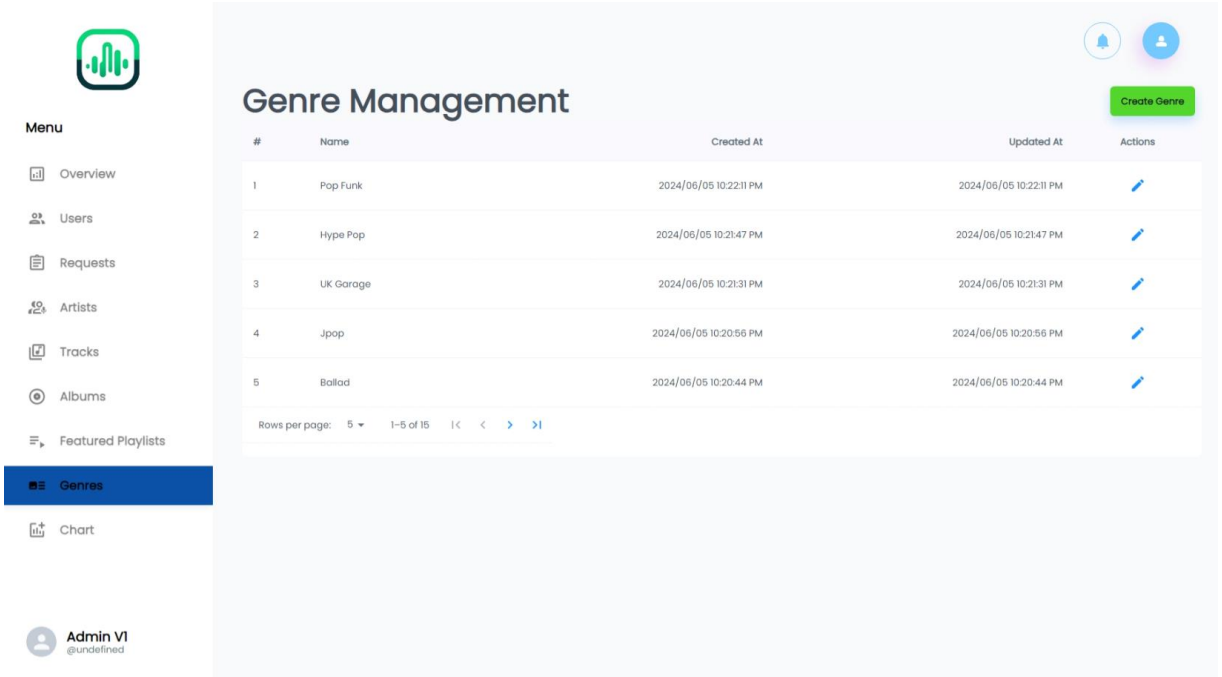
Hình 3.12: Giao diện trang chi tiết playlist

Màn hình trên hiển thị chi tiết playlist, bao gồm các bài hát có trong playlist. Tại đây admin có thể thay đổi thứ tự của các bài hát cũng như thêm hoặc xóa các hát ra khỏi playlist.



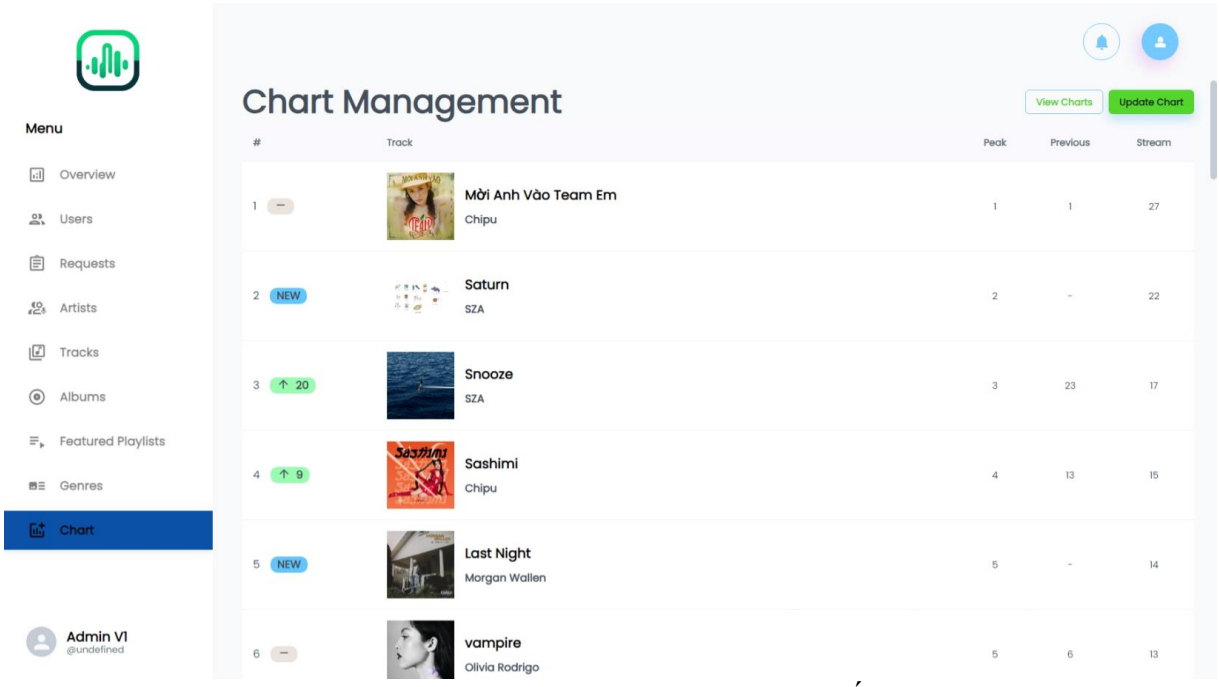
Hình 3.13: Giao diện pop-up để thêm bài hát vào playlist

Pop-up dùng để tìm kiếm và thêm bài hát vào playlist. Sẽ có thông báo lỗi khi thêm các bài hát đã có sẵn trong playlist.



Hình 3.14: Giao diện trang quản lý thể loại nhạc (genre)

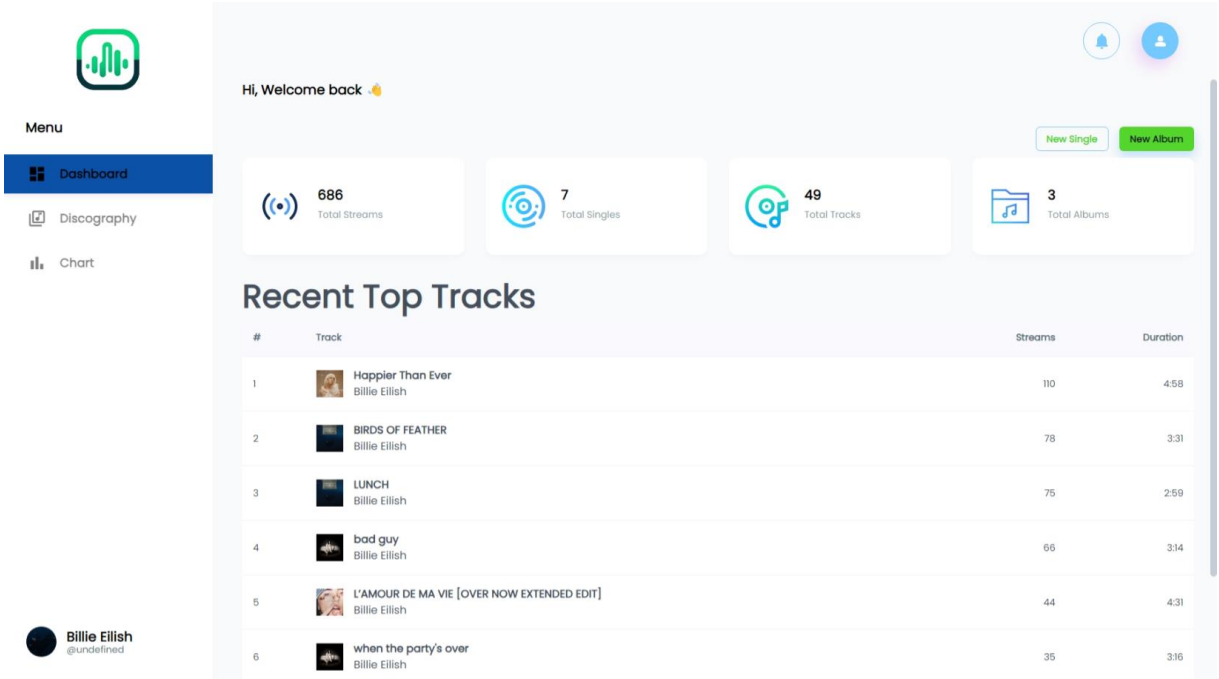
Màn hình trên hiển thị toàn bộ danh sách thể loại nhạc do admin đã tạo ra. Tại đây admin tạo mới, chỉnh sửa một thể loại bất kì.



Hình 3.15: Giao diện trang quản lý bảng xếp hạng

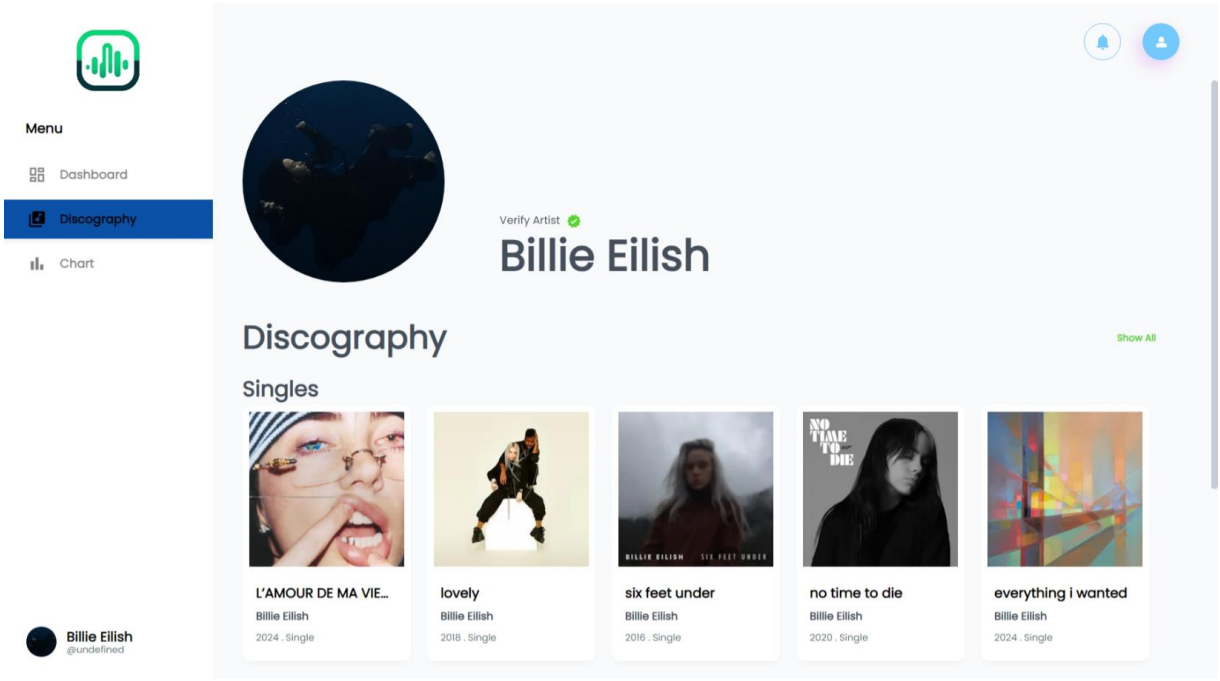
Màn hình trên hiển thị một bảng xếp hạng gần đây nhất. Tại đây admin cập nhật lại bảng xếp hạng hoặc để hệ thống tự động cập nhật hàng ngày.

3.2.2 Trang web cho nghệ sĩ



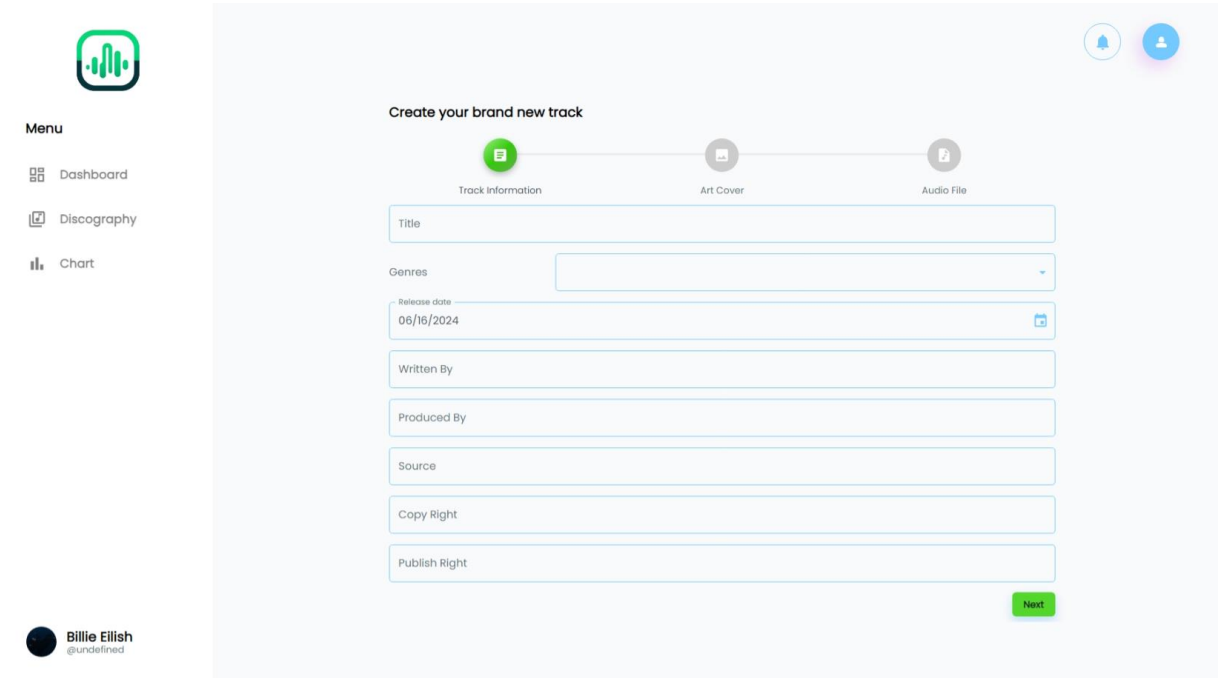
Hình 3.16: Giao diện trang overview của nghệ sĩ

Màn hình trên hiển thị các thông tin tổng quan của nghệ sĩ như là: số lượt nghe nhạc, tổng số đĩa đơn, tổng số bài hát và tổng số album có mặt trên ứng dụng. Ngoài ra bên dưới còn hiển thị lên các bài hát nổi bật của nghệ sĩ trong thời gian gần đây.

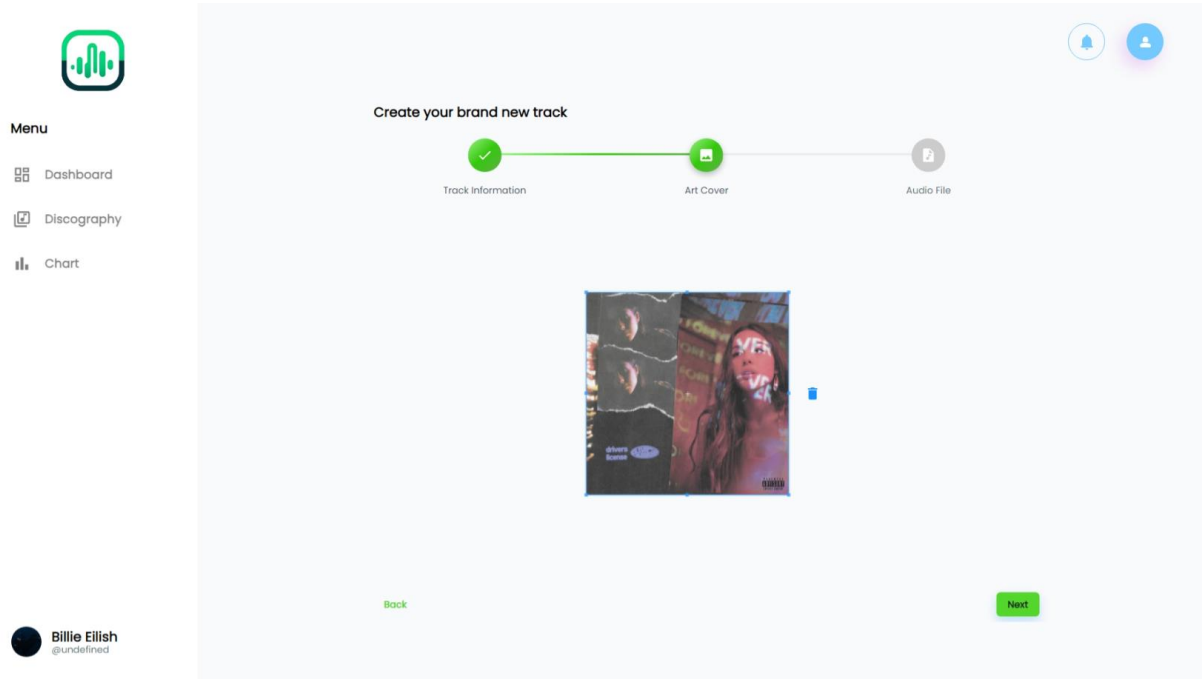


Hình 3.17: Giao diện trang sản phẩm âm nhạc của nghệ sĩ (discography)

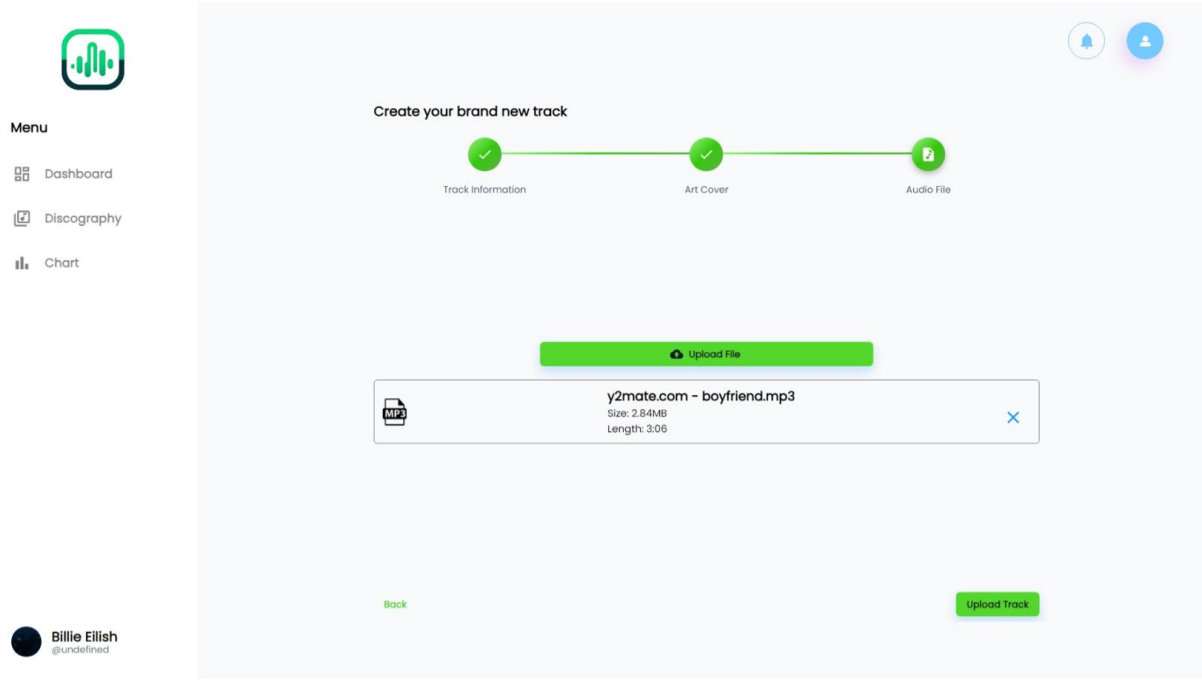
Màn hình trên hiển thị toàn bộ các sản phẩm âm nhạc của nghệ sĩ đó như là đĩa đơn và album. Nghệ sĩ có thể chọn nút show để hiển thị đầy đủ hơn.



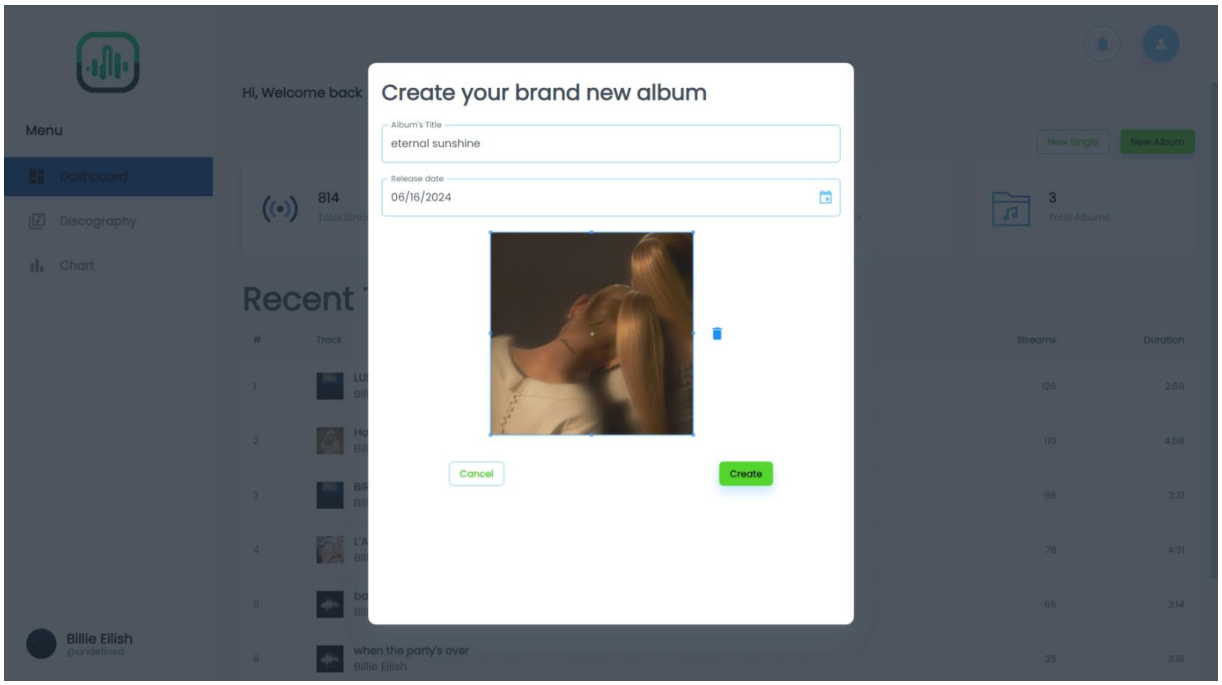
Hình 3.18: Giao diện tạo mới một đĩa đơn (single) – bước 1 nhập thông tin



Hình 3.19: Giao diện tạo mới một đĩa đơn – bước 2 tải ảnh bìa cho bài hát

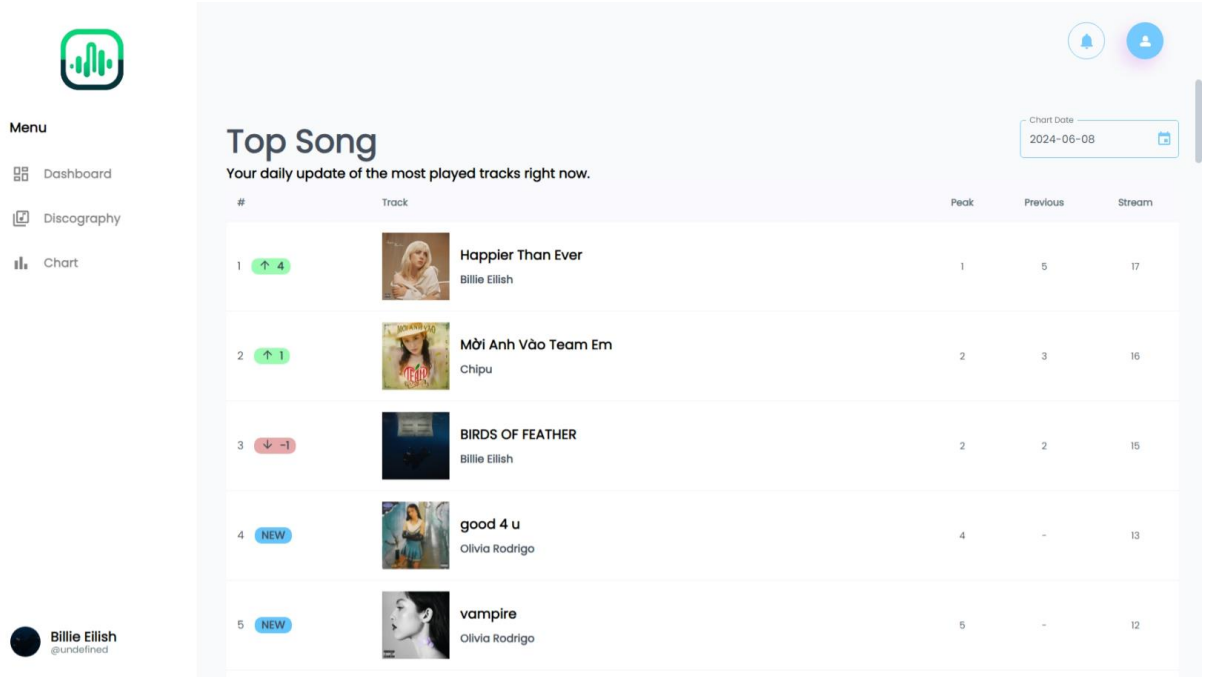


Hình 3.20: Giao diện tạo mới một đĩa đơn – bước 3 tải file audio của bài hát



Hình 3.21: Giao diện tạo mới một album

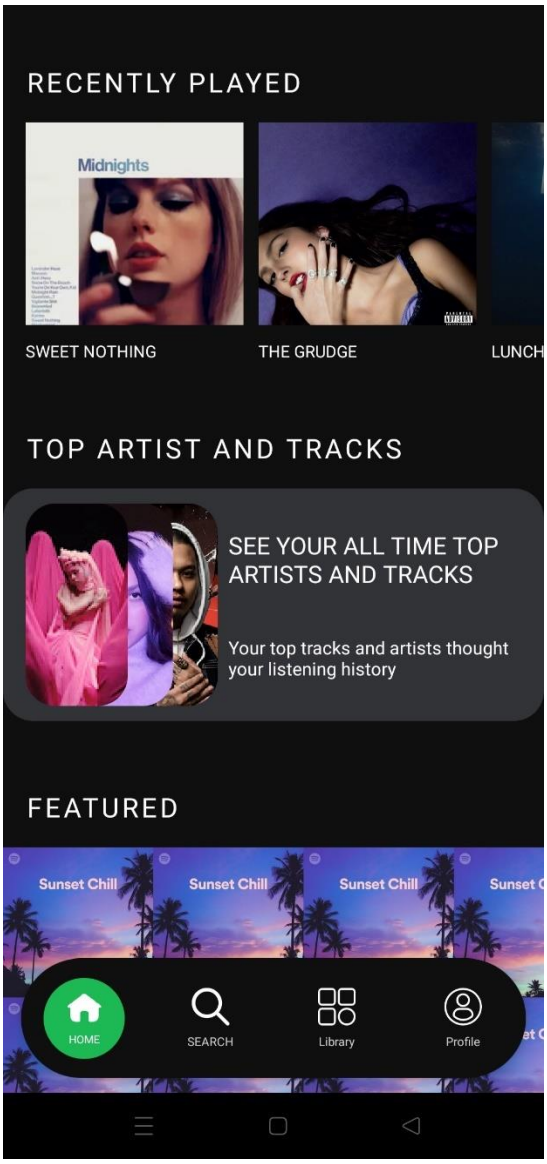
Màn hình trên cho phép nghệ sĩ tạo mới một album bằng cách điền tên album, ngày phát hành và đăng tải ảnh bìa của album.



Hình 3.22: Giao diện trang bảng xếp hạng theo ngày

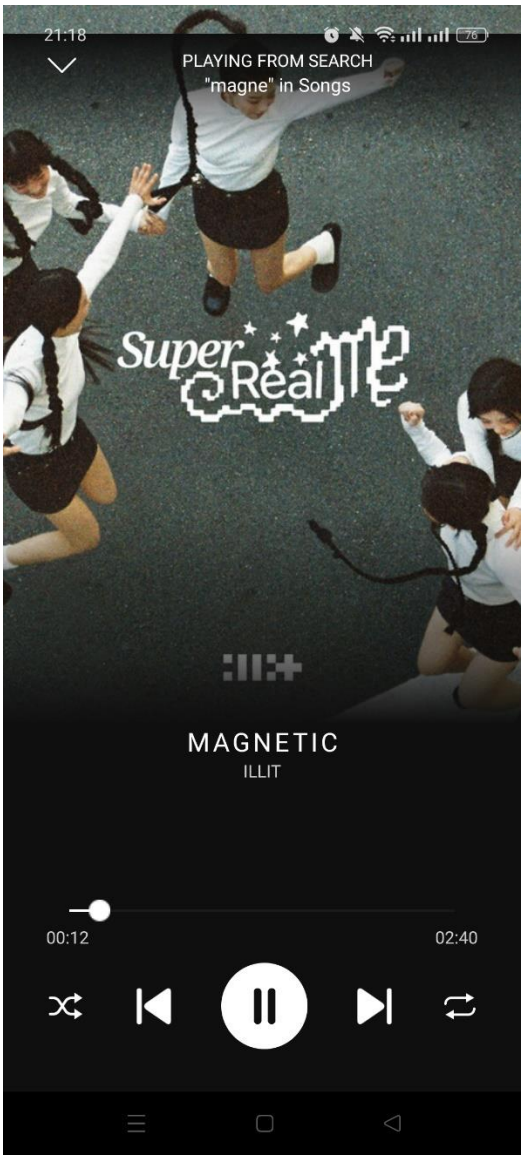
Màn hình trên hiển thị bảng xếp hạng các bài hát theo ngày gần nhất. Ngoài ra người dùng có thể chọn một ngày bất kì để xem lại các bảng xếp hạng cũ.

3.2.3 Ứng dụng mobile cho người dùng



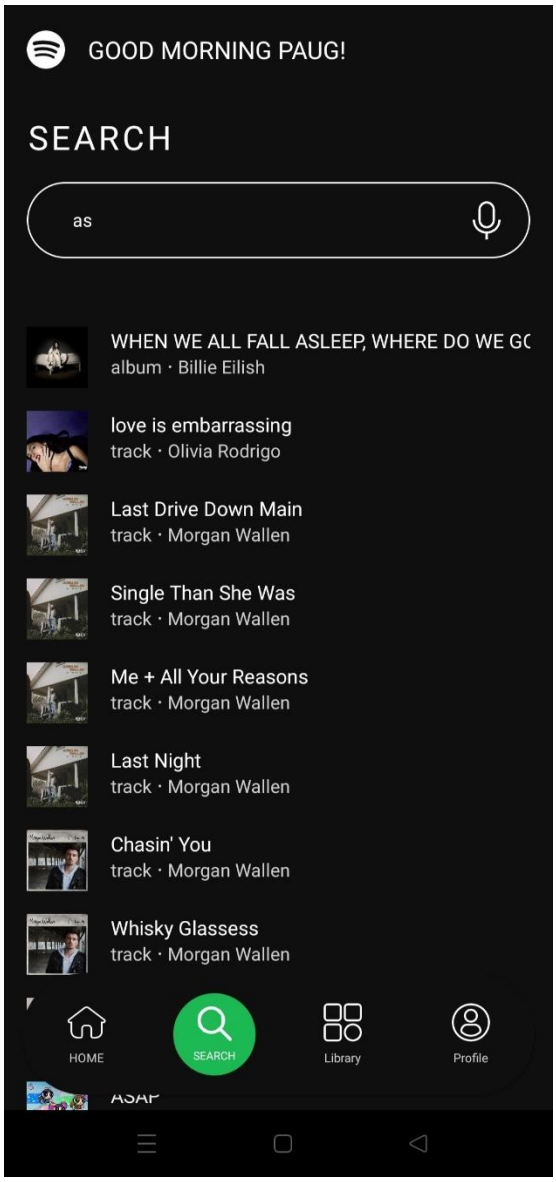
Hình 3.23: Giao diện trang chủ của mobile

Màn hình di động ở trang chủ cho phép người dùng có thể xem một số bài hát gần đây, một số nghệ sĩ, bài hát và playlist nổi bật.



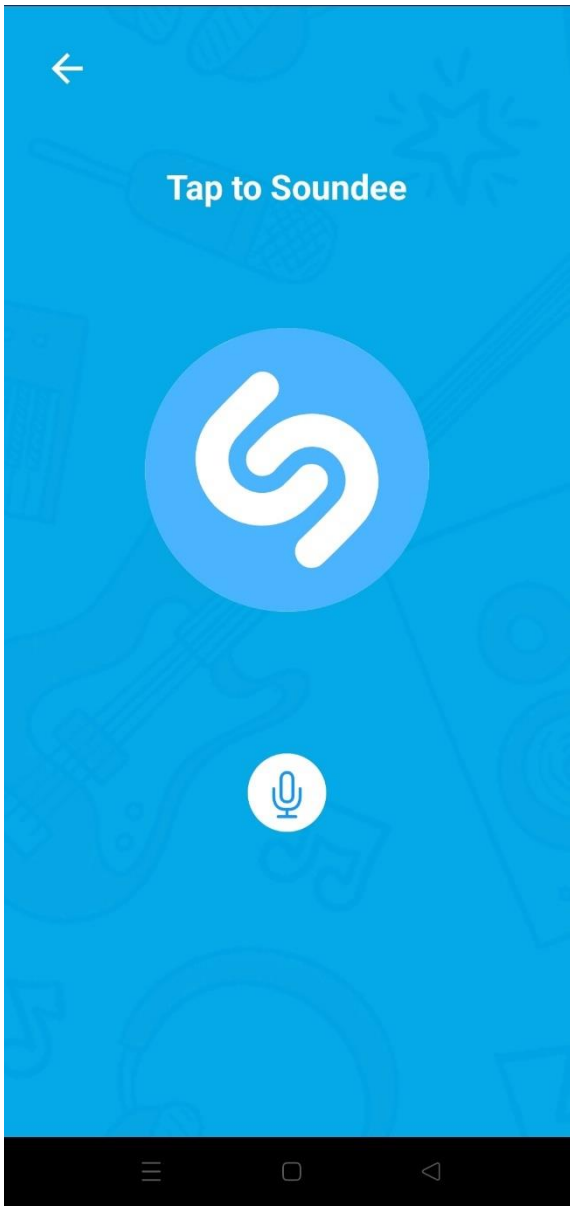
Hình 3.24: Giao diện trang bài hát khi được phát

Đây là màn hình khi phát một bài hát bất kì. Tại đây người dùng có thể tạm dừng nhạc, tua nhanh, tua lùi hoặc là phát bài hát tiếp theo hoặc trước đó trong một album hay playlist.



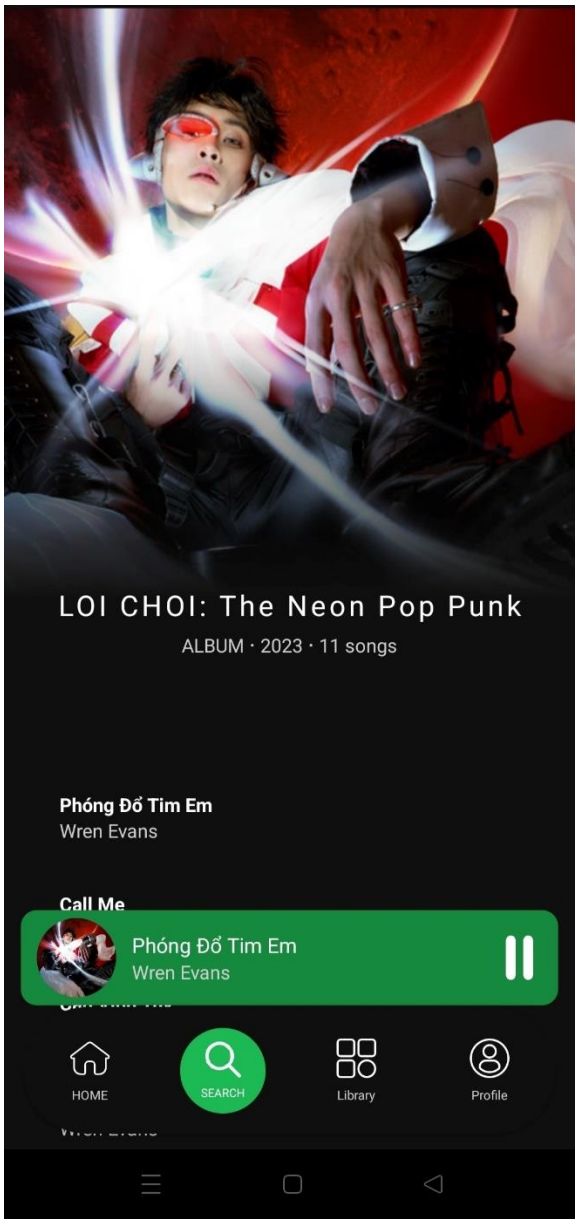
Hình 3.25: Giao diện trang tìm kiếm bài hát bằng từ khóa

Màn hình hiển thị toàn bộ kết quả tìm kiếm bài hát khi người dùng nhập từ khóa. Ngoài ra có thể sử dụng tính năng tìm kiếm bằng âm thanh khi chọn nút có biểu tượng audio trên góc phải màn hình.



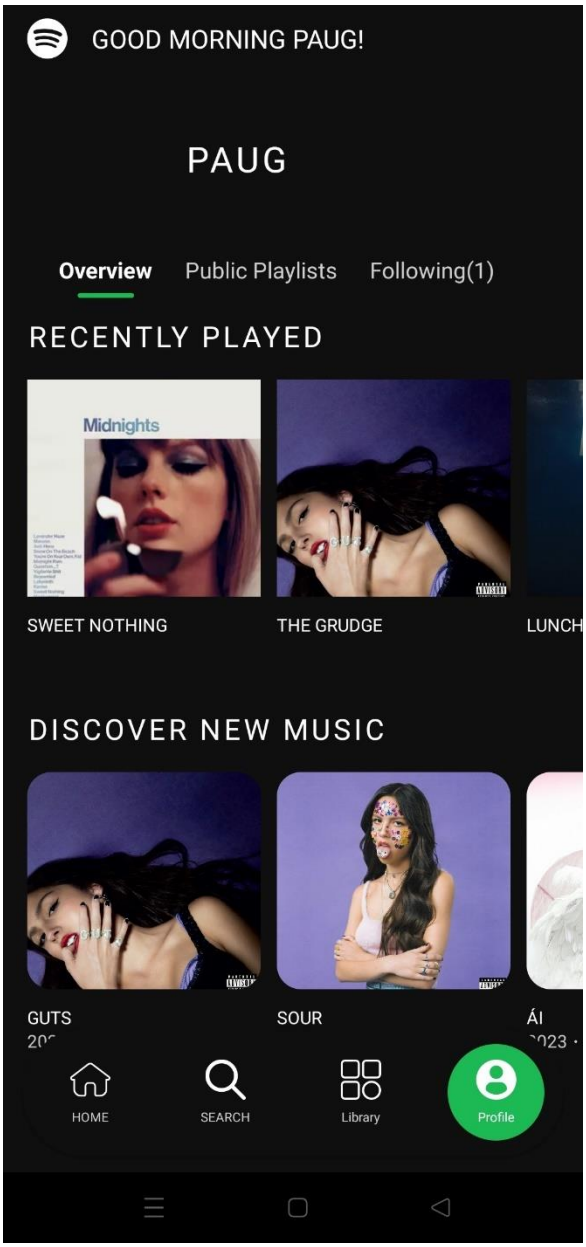
Hình 3.26: Giao diện trang tìm kiếm bằng âm thanh

Màn hình được hiển thị sau khi người dùng chọn tính năng tìm kiếm bằng âm thanh. Người dùng có thể nhấn giữ để thu âm thanh trực tiếp hoặc tải file audio lên để tiến hành tìm kiếm. Khi tìm được bài hát trong hệ thống thì sẽ chuyển hướng và phát bài hát đó lên.



Hình 3.27: Giao diện trang xem album

Màn hình trên hiển thị album và toàn bộ bài hát của album đó. Người dùng có thể nghe tất cả các bài hát trong album ở đây.



Hình 3.28: Giao diện trang xem hồ sơ của người dùng

Màn hình sẽ hiển thị các bài hát được nghe gần đây nhất của người dùng, ngoài ra còn hiển thị thêm các album, bài hát mới phát hành gần đây nhất.

3.3 Kết chương

Chương này trình bày cách triển khai và cấu hình ứng dụng và môi trường triển khai ứng dụng để có thể sử dụng trên dịch vụ internet. Ngoài ra, chương này còn cung cấp một số hình ảnh của ứng dụng, giúp người đọc có thêm một cái nhìn tổng quan về hệ thống.

KẾT LUẬN

Kết quả đạt được

Trong thời gian tìm hiểu, nghiên cứu cơ sở lý thuyết và triển khai phát triển ứng dụng, đồ án đã đạt được những kết quả sau:

Về mặt lý thuyết, đồ án đã ứng hình mô hình client-server, RESTful API để giao tiếp giữa frontend và backend, ứng dụng việc sử dụng AWS S3 Bucket để lưu trữ file audio, ứng dụng thuật toán phân tích fingerprints để nhận diện bài hát bằng âm thanh.

Về mặt thực tiễn, đã xây dụng hiện một website để quản trị viên quản lý toàn bộ hệ thống và cho nghệ sĩ để đăng các sản phẩm âm nhạc của mình. Ngoài ra còn triển khai thêm một ứng dụng mobile để người dùng có thể dễ dàng trải nghiệm âm nhạc.

Hạn chế

- Chưa tích hợp thanh toán online để mua các gói dịch vụ đăng ký để nghe nhạc;
- Chưa tích hợp khả năng đăng ký bằng các dịch vụ OAuth2 bên ngoài như Google, Facebook, X,...;
- Thiếu hệ thống gợi ý nhạc dựa theo nhu cầu, lịch sử nghe của người dùng, do đó giảm khả năng tiếp cận các bài hát cũ tới người dùng mới;
- Đối với hệ thống tìm kiếm bài hát bằng âm thanh, tuy kết quả tìm kiếm bằng file tách từ audio ra có độ chính xác cao, nhưng khi tìm kiếm bằng cách thu âm thành từ mobile thì cho kết quả chính xác thấp.

Hướng phát triển

- Tích hợp thêm dịch vụ thanh toán online, qua đó xây dựng thêm chức năng quản lý nguồn tiền cho nền tảng;
- Tích hợp thêm các dịch vụ Oauth2 nhằm tăng cường trải nghiệm người dùng, không yêu cầu họ nhập quá nhiều thông tin;
- Xây dụng thêm nhiều chức năng hơn cho ứng dụng mobile ví dụ như là hệ thống gợi ý nhạc cho người dùng;
- Đối với chức năng tìm kiếm bằng âm thanh, xây dựng thêm một hệ thống lọc nhiễu tốt hơn, hoặc sử dụng machine learning để gia tăng độ chính xác của chức năng tìm kiếm.

TÀI LIỆU THAM KHẢO

[1] Avery Li-Chun Wang, An Industrial-Strength Audio Search Algorithm, 2023
[2] NextJS, Documentation, 2023
[3] ExpressJS, Documentation, 2019
[4] MongoDb, Documentation, 2024
[5] Arda Yalçıner, Over-the-air Audio Identification, 2016
[6] React Native, Documentation, 2022
[7] Flask, Documentation, 2021
[8] Javascript, Documentation, 2021