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 PHẦN MỀM

ĐỀ TÀI:
XÂY DỰNG WEBSITE NGHE NHẠC SOUNDWAVE

Người hướng dẫn: **ThS. ĐỖ THỊ TUYẾT HOA**

Sinh viên thực hiện: **PHAN QUỐC HÙNG**

Số thẻ sinh viên: **102210162**

Lớp**: 21TCLC_DT2**

Đà Nẵng, 01/2026

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 PHẦN MỀM

ĐỀ TÀI:

XÂY DỰNG WEBSITE NGHE NHẠC SOUNDWAVE

Người hướng dẫn: **ThS. ĐỖ THỊ TUYẾT THOA**
Sinh viên thực hiện: **PHAN QUỐC HÙNG**
Số thẻ sinh viên: **102210162**
Lớp**: 21TCLC_DT2**

Đà Nẵng, 01/2026

Sinh viên thực hiện: Phan Quốc Hùng
Số thẻ SV: 102210162 Lớp: 21TCLC_DT2

SoundWave là hệ thống web nghe nhạc trực tuyến cho phép người dùng truy cập và thưởng thức kho bài hát đa dạng mọi lúc, mọi nơi thông qua giao diện web hiện đại. Hệ thống được xây dựng nhằm đáp ứng nhu cầu giải trí ngày càng cao của người dùng Internet cũng như khai thác xu hướng phát triển mạnh của các nền tảng streaming âm nhạc.

Dự án được phát triển dựa trên mô hình **Frontend – Backend – Database**, sử dụng các công nghệ **React/Next.js** cho giao diện người dùng, **Node.js/Express.js** cho xử lý phía máy chủ và **MongoDB** làm cơ sở dữ liệu chính. Hệ thống cung cấp các chức năng cốt lõi bao gồm: đăng ký, đăng nhập, nghe nhạc trực tuyến, quản lý danh sách phát (playlist), quản lý bài hát, theo dõi lượt nghe.

Đối với quản trị viên, hệ thống cung cấp trang quản lý nội dung cho phép thêm, sửa, xóa bài hát trong kho nhạc công cộng. Người dùng thông thường có thể tạo playlist, quản lý bài hát cá nhân và trải nghiệm nghe nhạc nhanh chóng với tốc độ tải tối ưu.

Thông qua dự án SoundWave, người thực hiện đã áp dụng kiến thức phân tích và thiết kế hệ thống thông tin, mô hình hóa bằng UML, xây dựng RESTful API, thiết kế cơ sở dữ liệu và triển khai các kỹ thuật xử lý file đa phương tiện. Hệ thống hướng đến khả năng mở rộng, dễ dàng phát triển thêm trong tương lai và phù hợp triển khai thực tế.

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

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

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Họ tên sinh viên: PHAN QUỐC HÙNG

Số thẻ sinh viên: 102210162 Lớp: 21TCLC_DT2

Khoa: Công nghệ thông tin Ngành: Công nghệ phần mềm

1. *Tên đề tài đồ án:* Xây dựng website nghe nhạc SoundWave
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:*

Không có

4. *Nội dung các phần thuyết minh và tính toán:*

Chương I: Cơ sở lý thuyết

Chương II: Phân tích và thiết kế hệ thống

Chương III: Triển khai

Chương IV: Kết luận và hướng phát triển

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

…...……
…
…...……
…
…...……
…
…...……
…

6. *Họ tên người hướng dẫn:* ThS. Đỗ Thị Tuyết Hoa
7. *Ngày giao nhiệm vụ đồ án: ……/……./202…..*
8. *Ngày hoàn thành đồ án: ……/……./202…..*

Đà Nẵng, ngày tháng năm 202

Trưởng Bộ môn công nghệ phần mềm **Người hướng dẫn**

LỜI CẢM ƠN

Em xin gửi lời cảm ơn chân thành và sâu sắc đến các thầy cô trong Khoa Công nghệ thông tin, cũng như tất cả các thầy cô trong trường Đại học Bách khoa – Đại học Đà Nẵng đã dìu dắt, dạy dỗ và truyền đạt kiến thức, kinh nghiệm quý báu của mình trong suốt quá trình em học tập và nghiên cứu tại trường.

Em xin bày tỏ tình cảm và lòng biết ơn chân thành của em tới cô giáo Ths. Đỗ Thị Tuyết Hoa, người đã từng bước hướng dẫn, giúp đỡ em tận tình trong quá trình thực hiện đồ án tốt nghiệp của mình. Nhờ đó em có thể hoàn thành đồ án đúng tiến độ và tích lũy cho mình nhiều kiến thức quý báu.

Tuy nhiên, vì còn bị giới hạn bởi kiến thức của bản thân và thời gian thực hiện đồ án có hạn nên không thể tránh khỏi có những thiếu sót còn tồn đọng trong dự án. Em mong nhận được sự góp ý từ các thầy cô và hội đồng để em có thể hoàn thiện dự án này hơn trong tương lai.

Em xin chân thành cảm ơn!

CAM ĐOAN

Tôi tên là Phan Quốc Hùng, sinh viên lớp 21TCLC_DT2, khoa Công nghệ Thông tin, Trường Đại học Bách Khoa Đà Nẵng.

Tôi xin cam đoan rằng:

1. Mọi kết quả nghiên cứu, phân tích và thiết kế trong đồ án này đều là kết quả của quá trình làm việc và nghiên cứu của riêng tôi, do tôi thực hiện dưới sự hướng dẫn của Cô Đỗ Thị Tuyết Hoa. Tôi không sao chép, mượn hoặc sử dụng kết quả của những người khác mà không có sự cho phép.

2. Mọi tham khảo dùng trong đồ án đều được trích rõ ràng tên tác giả, tên công trình và thời điểm công bố.

3. Tôi cam kết tuân thủ tất cả các quy định của nhà trường. Tôi hiểu rằng việc vi phạm các quy định này sẽ dẫn đến các hình thức kỷ luật của nhà trường, bao gồm cả việc hủy bỏ kết quả học tập và xử lý theo quy định.

Tôi xin chịu toàn bộ trách nhiệm về lời cam đoan này

Sinh viên thực hiện

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

DANH SÁCH CÁC KÝ HIỆU, CHỮ VIẾT TẮT

CHỮ VIẾT TẮT

STT	Từ viết tắt	Từ đầy đủ	Ý nghĩa
1	HTML	HyperText Markup Language	Ngôn ngữ tạo cấu trúc trang web
2	API	Application Programming Interface	Giao diện lập trình ứng dụng
3	CSS	Cascading Style Sheets	Ngôn ngữ định dạng giao diện web
4	UI	User Interface	Giao diện người dùng
5	UX	User Experience	Trải nghiệm người dùng
6	HTTP	HyperText Transfer Protocol	Giao thức truyền tải siêu văn bản
7	RESTful	Representational State Transfer	Một kiểu thiết kế hệ thống web dựa trên nguyên lý của REST
8	JSON	JavaScript Object Notation	Định dạng dữ liệu dạng đối tượng
9	SQL	Structured Query Language	Ngôn ngữ truy vấn cơ sở dữ liệu quan hệ
10	MVC	Model-View-Controller	Mô hình phân tách logic, giao diện, điều khiển
11	ERD	Entity Relationship Diagram	Sơ đồ thực thể - quan hệ

MỞ ĐẦU

1. Lý do chọn đề tài

Âm nhạc là một phần không thể thiếu trong đời sống con người hiện đại, giúp thư giãn, truyền cảm xúc và kết nối mọi người trên toàn thế giới. Tuy nhiên, đa số các nền tảng nghe nhạc hiện nay đều mang tính thương mại cao, giới hạn người dùng miễn phí, hoặc chưa tối ưu cho việc cá nhân hóa trải nghiệm nghe nhạc. Từ thực tế đó, nhóm quyết định thực hiện đề tài "SoundWave – Ứng dụng nghe nhạc trực tuyến", nhằm tạo ra một nền tảng nghe nhạc mở, thân thiện, dễ sử dụng và mang tính cá nhân hóa cao, giúp người dùng nghe – tìm kiếm – lưu trữ – quản lý playlist của riêng mình một cách thuận tiện, đồng thời rèn luyện kỹ năng phát triển ứng dụng web full-stack hiện đại.

2. Mục đích và nhiệm vụ

a) Mục đích

Xây dựng hệ thống nghe nhạc trực tuyến SoundWave cho phép người dùng nghe nhạc, thực hiện chức năng tìm kiếm bài hát, tạo và quản lý playlist cá nhân, lưu bài hát yêu thích. Người quản trị quản lý nội dung và người dùng.

b) Nhiệm vụ

- Thiết kế và xây dựng giao diện người dùng bằng ReactJS.
- Xây dựng backend với NodeJS + ExpressJS để xử lý logic nghiệp vụ.
- Xây dựng API và quản lý dữ liệu bằng MongoDB.
- Tích hợp phát nhạc trực tuyến và tìm kiếm.
- Đảm bảo bảo mật bằng JWT và phân quyền người dùng.

3. Đối tượng và phạm vi đề tài

a) Mục đích

- Người dùng thông thường (User): có thể đăng ký, đăng nhập, tìm kiếm bài hát, nghe nhạc, tạo playlist, lưu bài hát yêu thích.
- Người quản trị (Admin): có quyền quản lý bài hát và tài khoản người dùng.

b) Phạm vi đề tài

- Phục vụ cho người dùng yêu thích âm nhạc.
- Cho các tổ chức muốn quản lý nội dung âm nhạc.

4. Kết quả dự kiến

a) Về mặt thực tiễn

- Giúp người dùng trải nghiệm âm nhạc cá nhân hóa.

- Có thể thêm nhwuxng bài hát mà người dùng nhặt được ở đâu đó trên nền tảng khác.
- Tạo nền tảng cơ bản để mở rộng ứng dụng thành hệ thống nghe nhạc trực tuyến đầy đủ các tính năng.

b) Về mặt khoa học

- Vậy dụng kiến thức lập trình web full-stack, cơ sở dữ liệu NoSQL, bảo mật và xử lí API.
- Là nền tảng nghiên cứu mở rộng cho AI gợi ý nhạc và hệ thống khuyến nghị.
- Tạo cơ sở cho việc nghiên cứu mở rộng về xử lý dữ liệu đa phương tiện (âm thanh, hình ảnh) và phát triển ứng dụng thời gian thực trong tương lai.

c) Về mặt ứng dụng

- Website SoundWave hoạt động ổn định, có giao diện thân thiện.
- Đầy đủ các chức năng người dùng (đăng ký đăng nhập, nghe nhạc,…).
- Admin có thể quản lý người dùng, bài hát.
- API hoạt động đúng chuẩn RESTful, dễ mở rộng cho mobile app.
- Giúp người dùng khám phá những bài hát mới dựa trên lịch sử tương tác, từ đó tăng thời gian sử dụng hệ thống.

CHƯƠNG I: CƠ SỞ LÝ THUYẾT

1.1. Kiến thức cơ bản

1.1.1 HTML

HTML (HyperText Markup Language) là ngôn ngữ đánh dấu chuẩn để xây dựng cấu trúc và nội dung cho các trang web. Được phát triển bởi World Wide Web Consortium (W3C), HTML5, phiên bản mới nhất (ra mắt năm 2014), cung cấp các tính năng như thẻ ngữ nghĩa và hỗ trợ đa phương tiện. HTML sử dụng các thẻ để định dạng nội dung như văn bản, hình ảnh, hoặc liên kết, tạo khung nền tảng cho ứng dụng web. Trong hệ thống nghe nhạc, HTML tổ chức giao diện người dùng với các thành phần như danh sách bài hát và trình phát nhạc. Thẻ HTML5, chẳng hạn như thẻ âm thanh, cho phép phát nhạc trực tiếp trên trình duyệt, cải thiện hiệu suất và khả năng tiếp cận.

1.1.2 CSS

CSS (Cascading Style Sheets) là ngôn ngữ định dạng kiểm soát giao diện và cách hiển thị các thành phần HTML. CSS tách biệt nội dung và thiết kế, giúp mã nguồn dễ bảo trì. CSS3 hỗ trợ bố cục linh hoạt (Flexbox, Grid) và thiết kế responsive cho các thiết bị khác nhau.

Trong hệ thống nghe nhạc, CSS được triển khai qua Tailwind CSS, một framework utility-first, sử dụng các lớp tiện ích để tạo giao diện responsive cho danh sách bài hát và trình phát nhạc. Tailwind CSS đảm bảo tính nhất quán, thẩm mỹ, và tối ưu hiệu suất.

1.1.3 TypeScript

TypeScript là một ngôn ngữ lập trình mã nguồn mở do Microsoft phát triển, được xây dựng dựa trên JavaScript. TypeScript mở rộng JavaScript bằng cách bổ sung hệ thống kiểu dữ liệu tĩnh (static typing), giúp lập trình viên phát hiện lỗi sớm trong quá trình phát triển phần mềm. Sau khi viết mã, TypeScript sẽ được biên dịch sang JavaScript để có thể chạy trên trình duyệt hoặc môi trường Node.js.

TypeScript hiện được sử dụng rộng rãi trong các framework và thư viện hiện đại như React, Angular và Next.js, đặc biệt phù hợp với các dự án có quy mô lớn và yêu cầu cao về khả năng bảo trì.

1.1.4 Cloudinary

Cloudinary là một nền tảng quản lý tài nguyên phương tiện (Media Management Platform) dựa trên đám mây (SaaS). Nó cung cấp giải pháp toàn diện cho toàn bộ "vòng đời" của hình ảnh và video trên web/ứng dụng: từ việc tải lên, lưu trữ, chỉnh sửa, tối ưu hóa cho đến phân phối nội dung qua mạng lưới truyền tải nội dung (CDN).

1.1.5 NPM

NPM là trình quản lý thư viện (package manager) mặc định cho môi trường thực thi Node.js. Đây là kho lưu trữ phần mềm lớn nhất thế giới, giúp các lập trình viên chia sẻ và tái sử dụng mã nguồn một cách hiệu quả.

1.1.6 Tailwind CSS

Tailwind CSS là một Utility-first CSS framework. Khác với các framework truyền thống như Bootstrap (cung cấp các thành phần dựng sẵn như .btn, .card), Tailwind cung cấp các lớp (classes) tiện ích nhỏ, đơn lẻ để bạn xây dựng giao diện trực tiếp trong file HTML/JSX.

1.1.7 GitHub

Git là một hệ thống quản lý phiên bản phân tán (Distributed Version Control System), cho phép người phát triển ghi lại trạng thái của toàn bộ mã nguồn tại từng thời điểm nhất định. Nhờ đó, các phiên bản trước có thể dễ dàng được truy xuất, so sánh hoặc phục hồi khi cần thiết, từ đó giảm thiểu rủi ro mất mát dữ liệu và hỗ trợ việc cộng tác hiệu quả giữa các thành viên trong nhóm.

GitHub là một dịch vụ lưu trữ mã nguồn trực tuyến nổi tiếng và được sử dụng rộng rãi trong cộng đồng lập trình. Đây là một nền tảng thuộc sở hữu của Microsoft, cung cấp môi trường thân thiện cho việc quản lý dự án, hỗ trợ các tính năng như phân quyền truy cập, theo dõi lỗi, quản lý yêu cầu hợp nhất (pull requests. GitHub phù hợp với cả các dự án cá nhân nhỏ lẻ lẫn các hệ thống phần mềm quy mô lớn, với chính sách hỗ trợ tài khoản miễn phí kèm theo các gói nâng cao dành cho doanh nghiệp.

1.2. Tổng quan về ReactJS

ReactJS là một thư viện JavaScript được phát triển bởi Facebook (nay là Meta) nhằm xây dựng giao diện người dùng (UI).

Được sử dụng rộng rãi để phát triển các ứng dụng web hiện đại có khả năng tương tác cao và thay đổi dữ liệu theo thời gian thực mà không cần tải lại toàn bộ trang.

Ra mắt lần đầu năm 2013 và hiện là một trong các công nghệ Frontend chủ đạo trên thế giới, bên cạnh là Angular và Vue.

1.2.1 Các tính năng chính của ReactJS

Component-Based Architecture (Kiến trúc thành phần): Giao diện được xây dựng từ các component độc lập, có thể tái sử dụng. Giúp quản lý code tốt hơn, dễ bảo trì và mở rộng.

Virtual DOM: Sử dụng Virtual DOM để chỉ cập nhật những phần giao diện thay đổi, tăng hiệu suất và giảm thời gian render so với thao tác trực tiếp trên DOM thật.

JSX: Cho phép viết HTML trong JavaScript thông qua JSX, giúp code ngắn gọn, trực quan và dễ phát triển component UI.

React Hooks: Cung cấp các hàm như useState, useEffect để quản lý state, vòng đời component mà không cần class, đơn giản hóa lập trình.

1.2.2 Ứng dụng trong đồ án

Được dùng làm giao diện người dùng cho website nghe nhạc SoundWave, các component như trình phát nhạc, trang bài hát, playlist, upload nhạc được xây dựng dưới dạng thành phần tách rời, dễ tái sử dụng.

React Hooks hỗ trợ quản lý trạng thái như bài hát đang phát, danh sách playlist, trong khi JSX và component giúp tổ chức giao diện rõ ràng và mở rộng nhanh chóng.

Giao tiếp với Backend thông qua RESTful API, dùng fetch hoặc axios để truy vấn dữ liệu.

Nhờ Virtual DOM, ứng dụng phản hồi nhanh, hiển thị mượt mà mà cần tải lại toàn trang, từ đó mang lại trải nghiệm người dùng tốt hơn.

1.3. Tổng quan về ExpressJS

ExpressJS là framework chạy trên nền tảng Node.js, được viết bằng TypeScript, giúp các lập trình viên xây dựng ứng dụng web và API phía máy chủ một cách dễ dàng, hiệu quả và có tổ chức hơn.

1.3.1 Các tính năng chính của ExpressJS

Middleware Architecture: Sử dụng mô hình middleware, cho phép xử lý tuần tự các tác vụ.

Routing: Cung cấp hệ thống để định nghĩa các URL (endpoint) và xác định cách ứng dụng phản hồi các loại yêu cầu HTTP khác nhau (GET, POST, PUT, DELETE).

Xây dựng ứng dụng đa dạng: Phù hợp để tạo các ứng dụng web đơn trang (SPA), API RESTful, ứng dụng web full-stack,…

1.3.2 Lợi ích của ExpressJS

Cung cấp cấu trúc rõ ràng, dễ mở rộng.

Tích hợp tốt với MongoDB qua mongoose, đảm bảo hiệu suất và khả năng bảo trì.

1.4. Tổng quan về mô hình MVC

Mô hình MVC (Model-View-Controller) là một kiến trúc phần mềm phân tách logic nghiệp vụ, giao diện người dùng, và điều khiển, giúp tăng tính bảo trì, tái sử dụng, và khả năng mở rộng của ứng dụng. MVC được sử dụng rộng rãi trong phát triển phần mềm.

1.4.1 Các thành phần của MVC

Model: Đại diện cho dữ liệu và logic nghiệp vụ, tương tác trực tiếp với cơ sở dữ liệu. Trong hệ thống nghe nhạc, Model quản lý các collection MongoDB như thông tin bài hát (tên, nghệ sĩ, thể loại), danh sách phát, hoặc dữ liệu người dùng. Ví dụ, Model xử lý truy vấn để lấy danh sách bài hát theo thể loại hoặc cập nhật lượt nghe.

View: Hiển thị dữ liệu từ Model cho người dùng thông qua giao diện. Trong hệ thống, View là giao diện web (ReactJS) hiển thị danh sách bài hát, trình phát nhạc, hoặc thông tin người dùng. View chỉ nhận dữ liệu từ Controller và không trực tiếp truy cập Model.

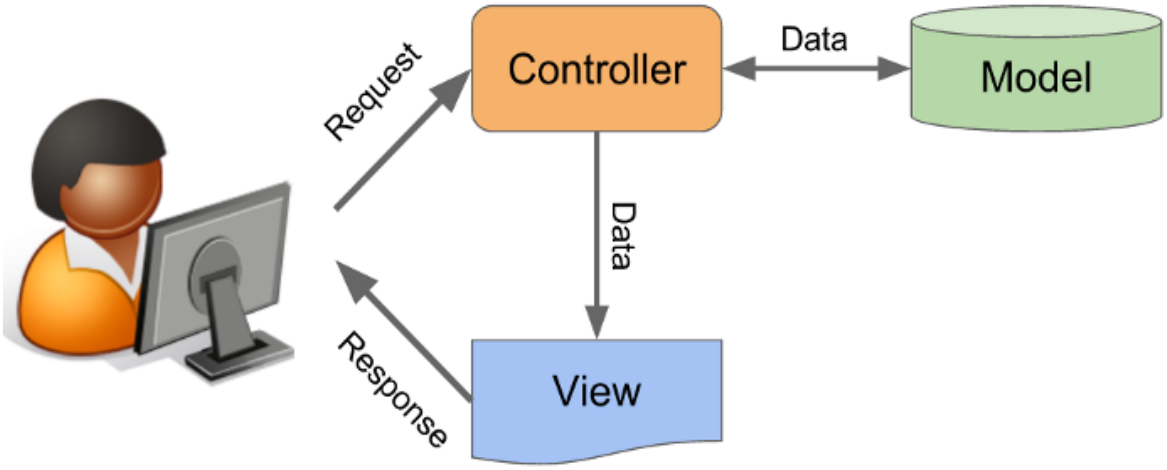
Controller: Điều phối giữa Model và View, xử lý yêu cầu người dùng và trả về dữ liệu phù hợp. Trong hệ thống, Controller (ExpressJS) nhận yêu cầu từ client (như phát bài hát hoặc thêm bình luận), gọi Model để lấy hoặc cập nhật dữ liệu, sau đó gửi dữ liệu đến View để hiển thị.

1.4.2 Lợi ích của MVC

Tính phân tách: Giảm sự phụ thuộc giữa các thành phần, giúp dễ dàng sửa đổi hoặc mở rộng.

Khả năng tái sử dụng: Model và Controller có thể được sử dụng cho cả giao diện web và di động.

Dễ bảo trì: Cấu trúc rõ ràng giúp phát hiện và sửa lỗi nhanh chóng.



Hình 1.4 Mô hình MVC

1.5. Tổng quan về MongoDB

MongoDB là hệ quản trị cơ sở dữ liệu NoSQL, lưu trữ dữ liệu dưới dạng tài liệu JSON (BSON). Mặc dù là NoSQL, trong đồ án, MongoDB được sử dụng với cấu trúc giống hệ cơ sở dữ liệu SQL, áp dụng schema cố định để đảm bảo tính nhất quán và tổ chức dữ liệu.

1.5.1 Đặc điểm của MongoDB

Linh hoạt: Cho phép điều chỉnh cấu trúc dữ liệu, nhưng trong dự án được kiểm soát chặt chẽ qua schema.

Khả năng mở rộng: Sharding và replication đảm bảo xử lý dữ liệu lớn.

1.5.2 Lợi ích của MongoDB

MongoDB phù hợp cho hệ thống nghe nhạc nhờ khả năng lưu trữ dữ liệu phức tạp như thông tin bài hát, danh sách phát, hoặc lịch sử nghe. Việc áp dụng schema cố định thông qua Mongoose giúp dữ liệu được tổ chức chặt chẽ, tương tự cơ sở dữ liệu quan hệ, đồng thời hỗ trợ truy vấn phức tạp như tìm kiếm bài hát theo thể loại hoặc thống kê lượt nghe.

1.6. Tổng quan về RESTful API

REST (Representational State Transfer) là kiến trúc phần mềm thiết kế dịch vụ web, dùng phương thức HTTP như GET, POST, PUT, DELETE để thao tác tài nguyên. RESTful API được sử dụng để giao tiếp giữa client và server.

1.6.1 Nguyên tắc của REST

Stateless: Mỗi yêu cầu HTTP chứa đầy đủ thông tin để xử lý.

Client-Server: Tách biệt client và server, tăng tính độc lập.

Uniform Interface: Sử dụng phương thức HTTP chuẩn và dữ liệu JSON.

1.6.2 Các phương thức HTTP chính

- GET (SELECT): Trả về một Resource hoặc một danh sách Resource.
- POST (CREATE): Tạo mới một Resource.
- PUT (UPDATE): Cập nhật thông tin cho Resource.
- DELETE (DELETE): Xoá một Resource.

1.7. Tổng quan về thuật toán SVD (Singular Value Decomposition)

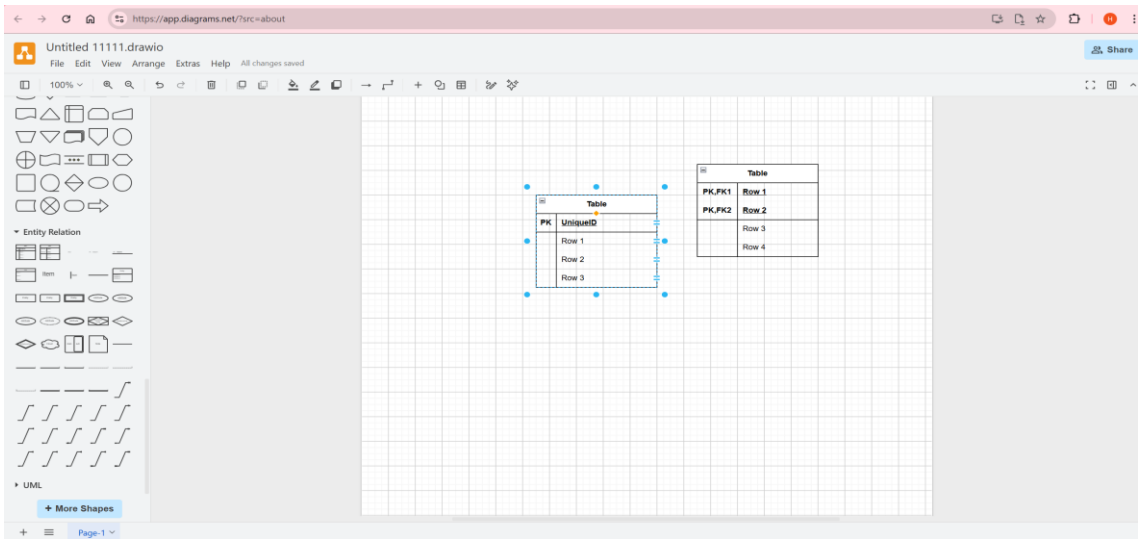
Mô hình Lọc cộng tác (Collaborative Filtering): Hệ thống không dựa vào thuộc tính bài hát (thể loại, nghệ sĩ) mà dựa trên hành vi của cộng đồng người dùng.

Là một kĩ thuật Matrix Factorization (Phân rã ma trận). Nó phân tách ma trận tương tác giữa Người dùng – Bài hát thành các "yếu tố ẩn" (Latent Factors). Một ví dụ cụ thể là: AI học được User A thích nhạc có tính chất "buồn", "acoustic" và bài hát X cũng có những tính chất đó, từ đó đưa ra gợi ý

1.8. Công cụ sử dụng

1.8.1 Diagrams.net

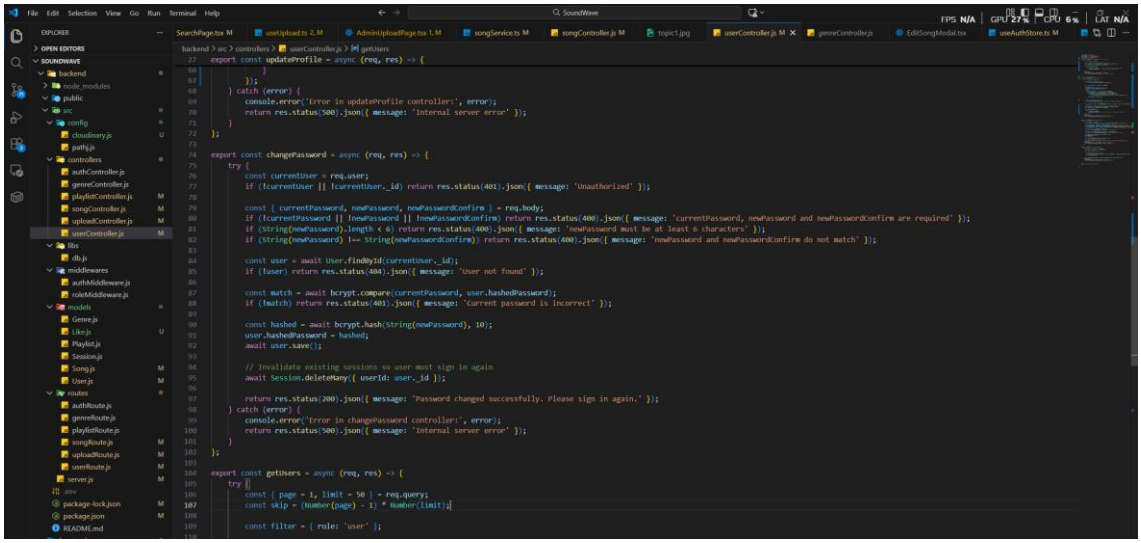
Diagrams.net (draw.io) là công cụ trực tuyến miễn phí để vẽ sơ đồ ca sử dụng, hoạt động, hoặc cơ sở dữ liệu. Trong đồ án, Draw.io thiết kế sơ đồ UML, minh họa rõ ràng phân tích và thiết kế hệ thống.



Hình 1.8.1 Diagrams.net

1.8.2 Visual Studio Code

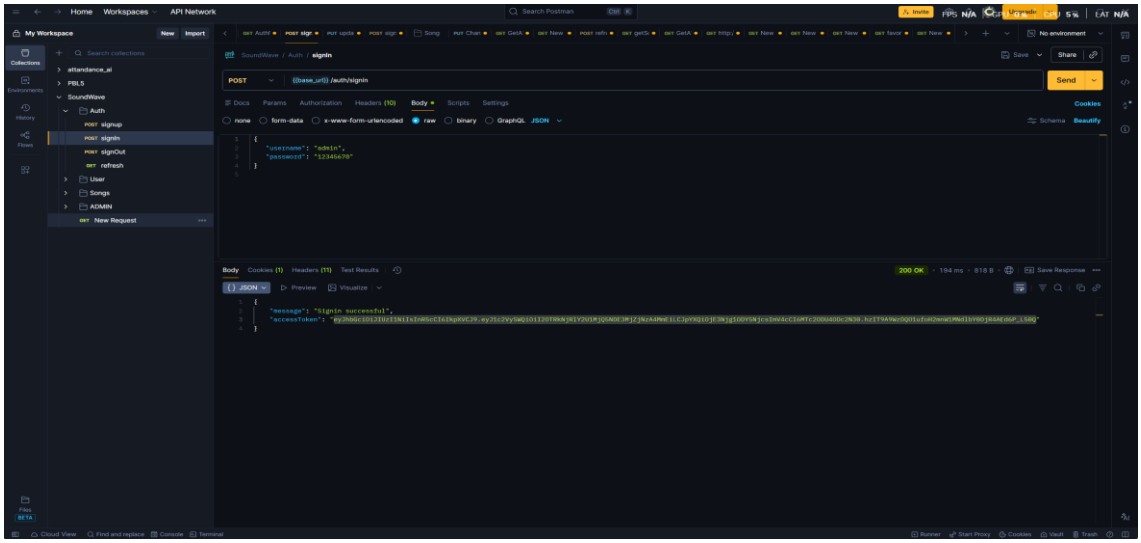
VSCode là trình soạn thảo mã nguồn hỗ trợ JavaScript, TypeScript, HTML. VSCode được sử dụng để viết mã cho frontend và backend.Tiện ích mở rộng như ESLint, Prettier tăng chất lượng mã nguồn.



Hình 1.8.2 Giao diện VSCode

1.8.3 Postman

Postman là công cụ kiểm tra và phát triển API, gửi yêu cầu HTTP và kiểm tra phản hồi. Trong đồ án, Postman thử nghiệm API như đăng ký, lấy bài hát, đảm bảo tính đúng đắn và hiệu suất backend.



Hình 1.8.3 Giao diện Postman

CHƯƠNG II: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

2.1. Khảo sát bài toán thực tế

2.1.1. Tổng quan hiện trạng

Trong bối cảnh hiện nay, nhu cầu nghe nhạc trực tuyến ngày càng tăng nhờ sự phát triển mạnh mẽ của Internet và thiết bị thông minh. Người dùng mong muốn có thể nghe nhạc mọi lúc mọi nơi, dễ dàng tìm kiếm bài hát yêu thích, tạo playlist cá nhân và đặc biệt là sử dụng các nguồn âm thanh đa dạng. Tuy nhiên, khảo sát thực tế cho thấy hầu hết các nền tảng nghe nhạc phổ biến hiện nay vẫn tồn tại nhiều hạn chế như giới hạn kho nhạc cá nhân, không cho phép người dùng tải lên nội dung riêng hoặc phải sử dụng các công cụ bên ngoài để chuyển đổi video thành nhạc. Bên cạnh đó, các dịch vụ miễn phí thường kèm theo quảng cáo gây khó chịu, còn nhiều tính năng nâng cao chỉ dành cho tài khoản trả phí, khiến trải nghiệm người dùng bị giới hạn.

Xuất phát từ thực trạng đó, mong muốn về một hệ thống nghe nhạc trực tuyến đơn giản, miễn phí, hỗ trợ người dùng tự tạo kho nhạc riêng và quản lý nội dung cá nhân trở nên cần thiết. Người dùng có nhu cầu sở hữu không gian âm nhạc của riêng mình, nơi họ có thể upload video, chuyển đổi sang định dạng MP3, lưu trữ trên cloud và tạo playlist linh hoạt mà không phụ thuộc vào nền tảng dịch vụ bên ngoài.

Hệ thống SoundWave được xây dựng nhằm đáp ứng những nhu cầu cấp thiết này. SoundWave hướng tới mục tiêu mang lại trải nghiệm nghe nhạc liền mạch, thân thiện, cho phép quản lý bài hát cá nhân, nghe nhạc trực tuyến và chia sẻ nội dung dễ dàng. Đồng thời, hệ thống cung cấp công cụ cho quản trị viên kiểm soát kho nhạc chung, đảm bảo tính ổn định và chất lượng dữ liệu trong hệ thống.

2.1.2 Phân tích chức năng

Website SoundWave được xây dựng nhằm cung cấp nền tảng nghe nhạc trực tuyến, quản lý bài hát cá nhân và tạo playlist linh hoạt cho người dùng. Các chức năng chính được phân tích theo hai nhóm: người dùng thông thường (User) và quản trị viên (Admin).

❖ Người dùng

Đăng ký tài khoản: Người dùng nhập thông tin cá nhân như email, tên đăng nhập, mật khẩu. Hệ thống lưu thông tin tài khoản vào cơ sở dữ liệu và xác nhận tài khoản mới.

Đăng nhập hệ thống: Người dùng nhập username và mật khẩu, hệ thống kiểm tra thông tin và tạo phiên làm việc cho người dùng.

Quản lý hồ sơ: Cập nhật thông tin cá nhân, tải ảnh đại diện

Nghe nhạc trực tuyến: Người dùng lựa chọn một bài hát từ danh sách có sẵn, hệ thống cung cấp trình phát nhạc và ghi nhận lượt nghe.

Tìm kiếm: Tìm kiếm bài hát theo nhiều tiêu chí (tên bài, nghệ sĩ, thể loại), hỗ trợ bộ lọc, sắp xếp để tối ưu trải nghiệm người dùng.

Quản lý Playlist: Người dùng có thể tạo nhiều playlist khác nhau, có thể thêm hoặc xóa bài hát khỏi playlist, chỉ hiển thị với người tạo.

❖ Người quản trị

Quản lý người dùng: Xem danh sách người dùng

Quản lý kho nhạc chung: Thêm bài hát vào hệ thống để tất cả người dùng cùng nghe, chỉnh sửa metadata bài hát, có thể xóa bài hát.

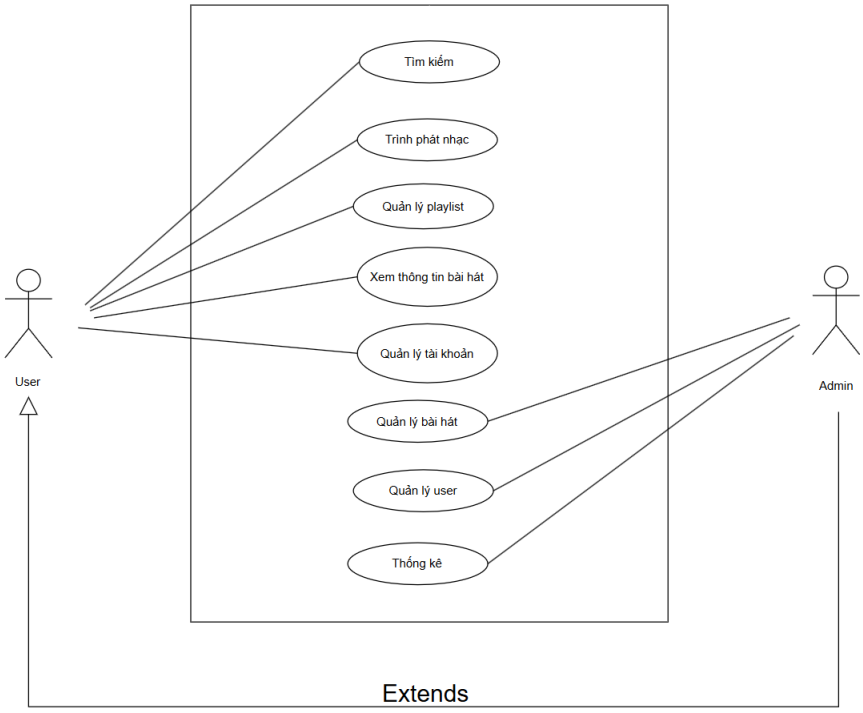
2.2. Biểu đồ Use Case

2.2.1 Các tác nhân trong hệ thống

Hệ thống có 2 tác nhân là: Người dùng và Người quản trị

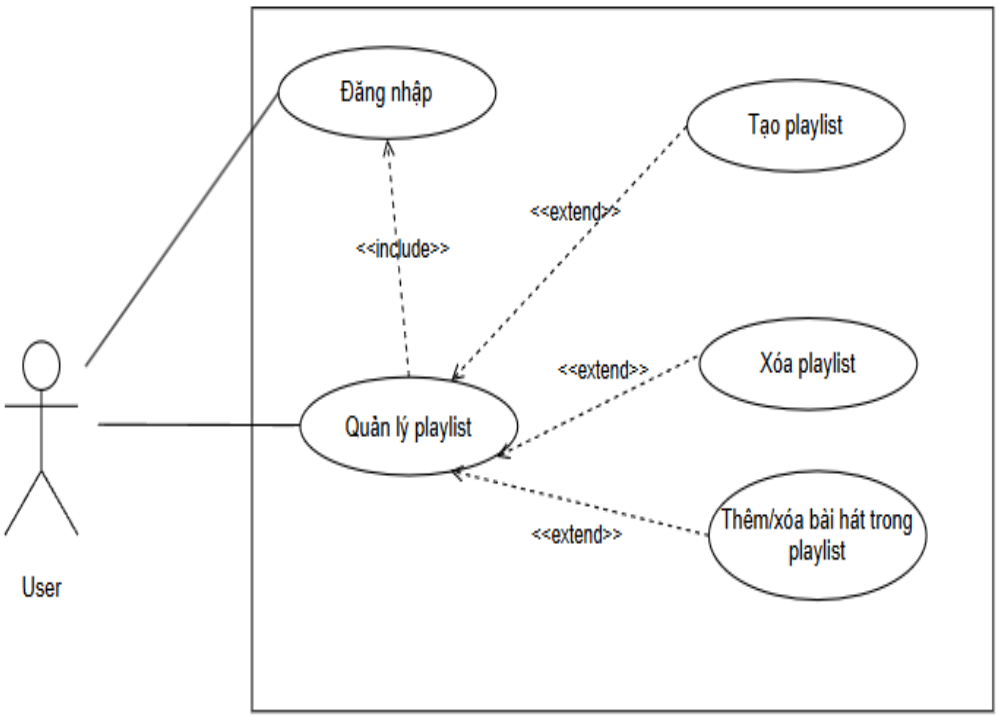
- Người dùng: Là người có thể nghe nhạc, thực hiện các tính năng của trình phát nhạc, quản lý playlist.
- Người quản trị: Là người có mọi quyền trong hệ thống quản lý bài hát và người dùng.

2.2.2 Biểu đồ Use Case tổng quát

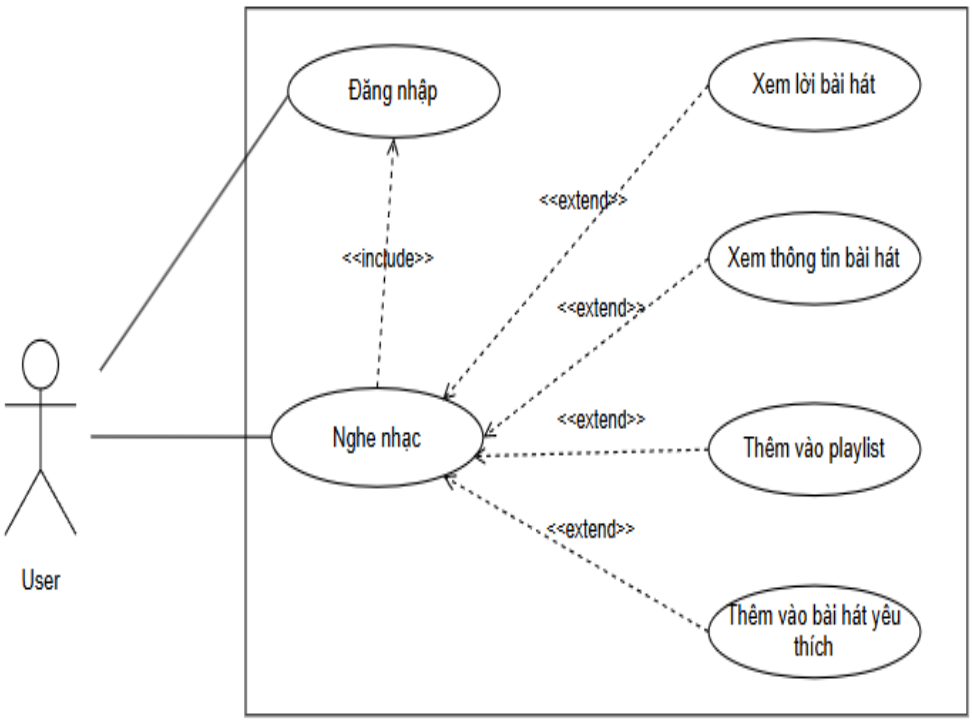


Hình 2.2.2 Biểu đồ Use Case tổng quát

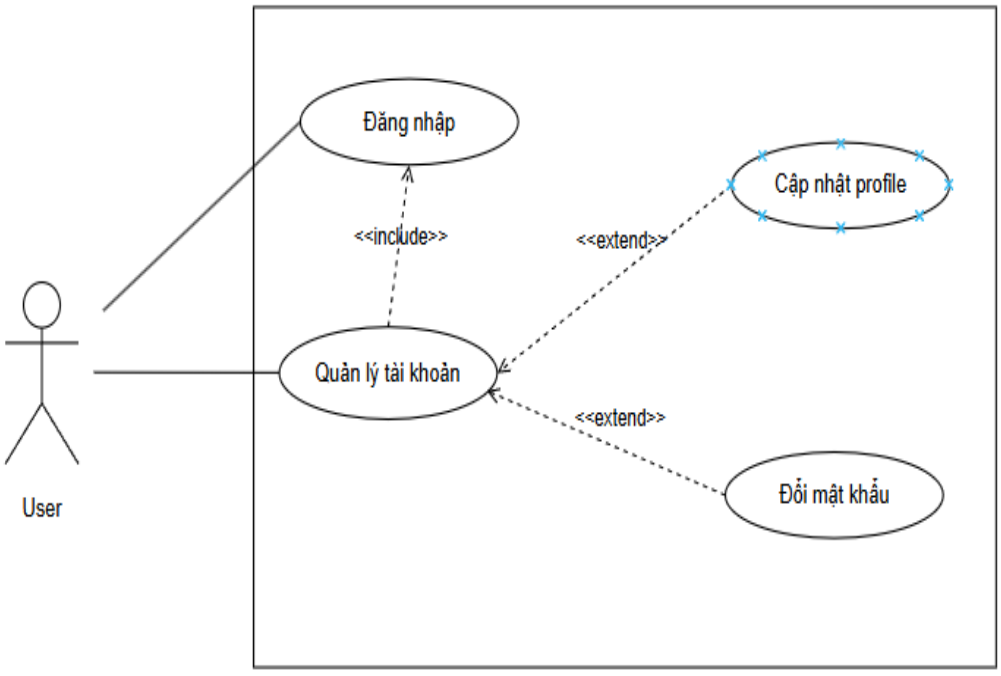
2.2.3 Biểu đồ Use Case sử dụng của Người dùng



Hình 2.2.3.1 Biểu đồ Use Case sử dụng Quản lý Playlist

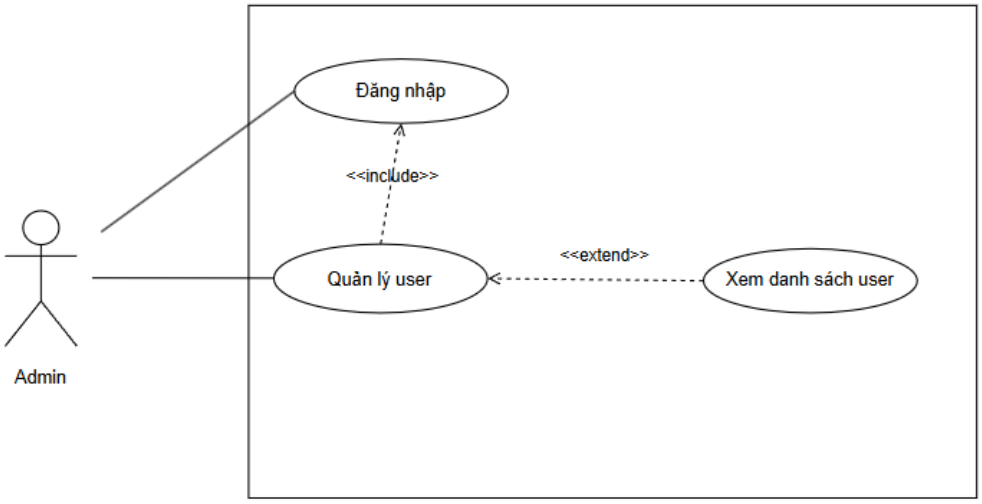


Hình 2.2.3.2 Biểu đồ Use Case sử dụng Nghe nhạc

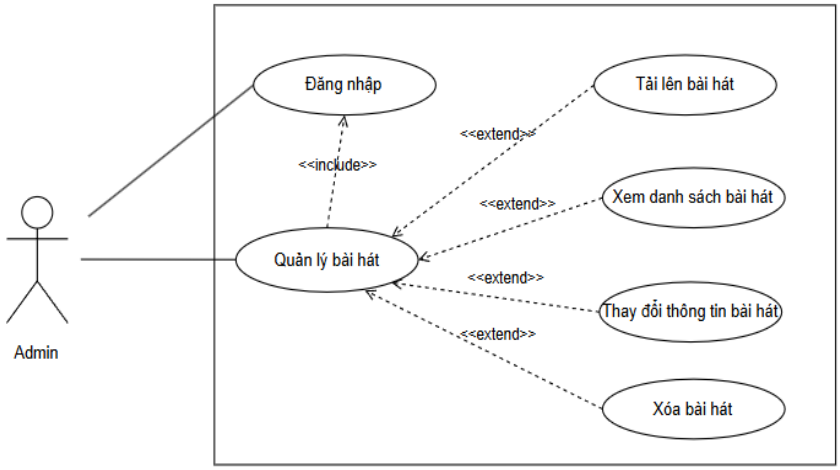


Hình 2.2.3.3 Biểu đồ Use Case sử dụng Quản lý tài khoản

2.2.4 Biểu đồ Use Case sử dụng của Người quản trị



Hình 2.2.4.1 Biểu đồ Use Case sử dụng Quản lý user



Hình 2.2.4.2 Biểu đồ Use Case sử dụng Quản lý bài hát

2.3. Đặc tả Use Case sử dụng

2.3.1 Đặc tả Use Case Đăng ký

Use Case	**Đăng ký**
Tác nhân	**User**
Mô tả	Cho phép người dùng tạo tài khoản mới để trở thành thành viên của SoundWave.
Tiền điều kiện	Người dùng chưa đăng nhập vào hệ thống.
Luồng sự kiện chính	1. Người dùng chọn chức năng "Đăng ký" trên giao diện. 2. Hệ thống hiển thị biểu mẫu đăng ký.3. Người dùng nhập: Họ tên, Email, Tên đăng nhập, Mật khẩu. 4. Người dùng nhấn nút "Đăng ký" 5. Hệ thống kiểm tra định dạng dữ liệu và kiểm tra trùng lặp Email/Username. 6. Hệ thống lưu thông tin người dùng mới vào CSDL. 7. Hệ thống thông báo thành công và chuyển sang trang Đăng nhập.
Luồng thay thế	**4a. Dữ liệu không hợp lệ**: Hệ thống báo lỗi tại trường dữ liệu sai và yêu cầu nhập lại. **5a. Tài khoản tồn tại**: Hệ thống báo "Email hoặc Tên đăng nhập đã được sử dụng".
Hậu điều kiện	Tài khoản mới được tạo trong CSDL với trạng thái Active.

Hình 2.3.1 Đặc tả use case Đăng ký

2.3.2 Đặc tả Use Case Đăng nhập

Use Case	**Đăng nhập**
Tác nhân	**User, Admin**
Mô tả	Xác thực người dùng để cấp quyền truy cập vào các chức năng hệ thống.
Tiền điều kiện	Người dùng đã có tài khoản.
Luồng sự kiện chính	1. Người dùng chọn chức năng "Đăng nhập". 2. Hệ thống hiển thị form đăng nhập. 3. Người dùng nhập Email/Username và Mật khẩu, nhấn "Đăng nhập". 4. Hệ thống mã hóa mật khẩu và đối chiếu với CSDL. 5. Hệ thống xác thực thành công, tạo phiên làm việc (Session). 6. Hệ thống điều hướng người dùng về trang chủ (hoặc trang Admin).
Luồng thay thế	**Sai thông tin**: Hệ thống báo "Tên đăng nhập hoặc mật khẩu không đúng".
Hậu điều kiện	Người dùng đăng nhập thành công.

Hình 2.3.2 Đặc tả use case Đăng nhập

2.3.3 Đặc tả Use Case Quản lý tài khoản

Use Case	**Quản lý tài khoản**
Tác nhân	**User, Admin**
Mô tả	Cho phép người dùng hoặc người quản trị đã đăng nhập vào hệ thống có thể quản lý tài khoản cá nhân của mình bao gồm chỉnh sửa thông tin cá nhân và mật khẩu
Tiền điều kiện	Người dùng đã có tài khoản
Luồng sự kiện chính	1. Người dùng hoặc người quản trị nhập các thông tin cá nhân cần chỉnh sửa. 2. Bấm nút Cập nhật. 3. Hệ thống kiểm tra dữ liệu nhập là đúng. 4. Thực hiện cập nhật thông tin.
Luồng thay thế	2a. **Dữ liệu không hợp lệ**: Hệ thống báo lỗi tại trường dữ liệu sai và yêu cầu nhập lại.
Hậu điều kiện	Cập nhật thành công

Hình 2.3.3 Đặc tả use case Quản lý tài khoản

2.3.4 Đặc tả Use Case Nghe nhạc

Use Case	**Phát nhạc**
Tác nhân	**User, Admin**
Mô tả	Người dùng phát bài hát trên trình phát nhạc (Player) của website.
Tiền điều kiện	Hệ thống hoạt động bình thường, bài hát khả dụng.
Luồng sự kiện chính	1. Người dùng click nút "Play" tại một bài hát bất kỳ. 2. Hệ thống tải thông tin bài hát và file audio từ server. 3. Hệ thống hiển thị thanh Player điều khiển (Play/Pause/Next). 4. Hệ thống bắt đầu phát nhạc. 5. Hệ thống cập nhật lượt nghe (Views) sau một khoảng thời gian phát nhất định.
Luồng thay thế	
Hậu điều kiện	Bài hát được phát, lượt xem được ghi nhận.

Hình 2.3.4 Đặc tả use case Nghe nhạc

2.3.5 Đặc tả Use Case Quản lý playlist

Use Case	Quản lý Playlist
Tác nhân	**User**
Mô tả	Tạo danh sách phát nhạc riêng và thêm bài hát vào danh sách.
Tiền điều kiện	Đã đăng nhập.
Luồng sự kiện chính	1. Người dùng chọn "Tạo Playlist mới". 2. Nhập tên Playlist và xác nhận. 3. Hệ thống lưu Playlist mới. 4. Người dùng duyệt bài hát, chọn "Thêm vào Playlist". 5. Người dùng chọn Playlist đích. 6. Hệ thống liên kết bài hát vào Playlist.
Luồng thay thế	**Xóa Playlist:** Người dùng chọn xóa -> Hệ thống xóa Playlist khỏi CSDL.
Hậu điều kiện	Playlist được cập nhật.

Hình 2.3.5 Đặc tả use case Quản lý playlist

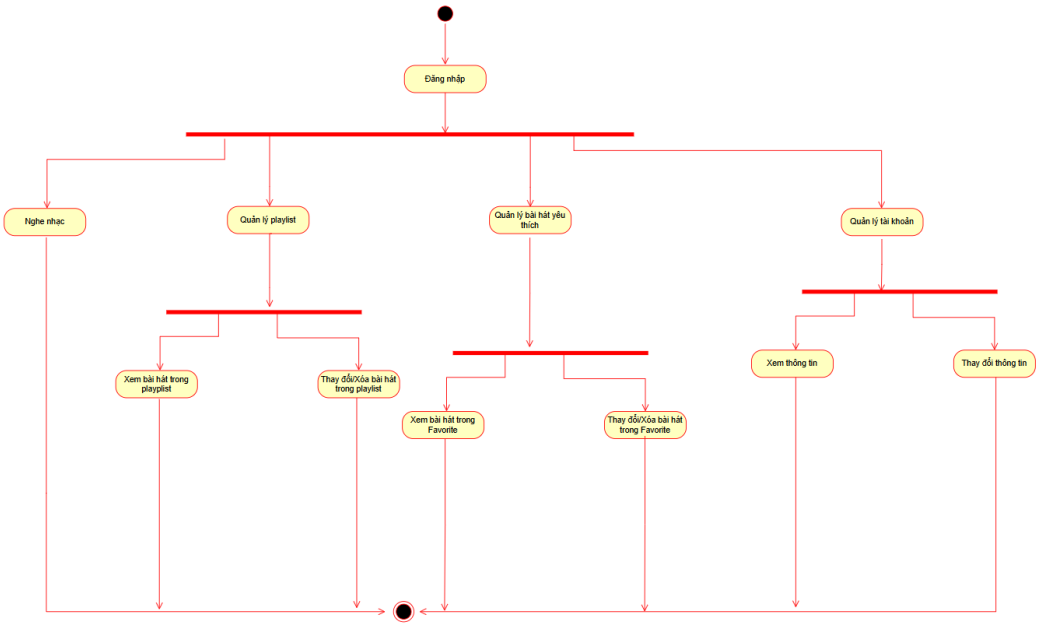
2.3.6 Đặc tả Use Case Quản lý bài hát

Use Case	**Quản lý bài hát**
Tác nhân	**Admin**
Mô tả	Quản lý kho nhạc hệ thống.
Tiền điều kiện	Đăng nhập quyền Admin.
Luồng sự kiện chính	1. Admin vào trang quản trị bài hát. 2. Chọn chức năng "Thêm mới". 3. Nhập thông tin (Tên, Ca sĩ, Thể loại) và upload file nhạc + ảnh bìa. 4. Nhấn "Lưu". 5. Hệ thống upload file và lưu dữ liệu vào CSDL. 6. Hiển thị thông báo thành công.
Luồng thay thế	**Thiếu thông tin:** Hệ thống yêu cầu nhập đầy đủ các trường bắt buộc. **Xóa bài hát:** Admin chọn xóa -> Hệ thống ẩn bài hát khỏi trang chủ.
Hậu điều kiện	Bài hát mới khả dụng cho toàn bộ người dùng.

Hình 2.3.6 Đặc tả use case Quản lý bài hát

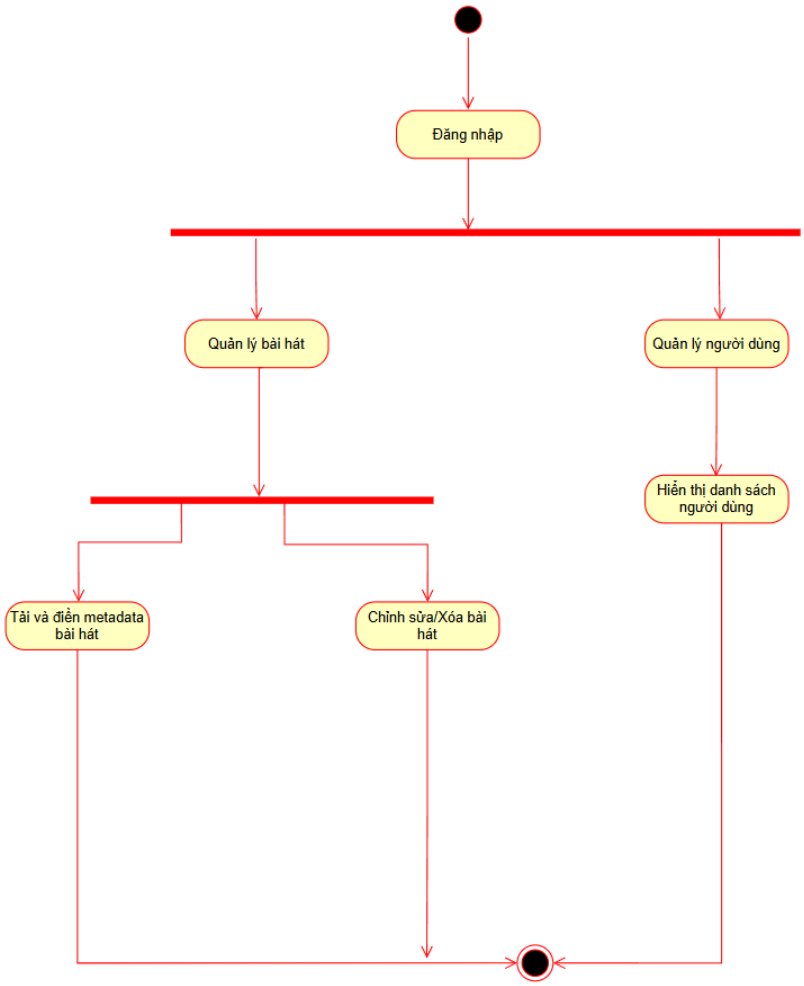
2.4. Biểu đồ hoạt động

2.4.1 Biểu đồ hoạt động của Người dùng



Hình 2.4.1 Biểu đồ hoạt động của Người dùng

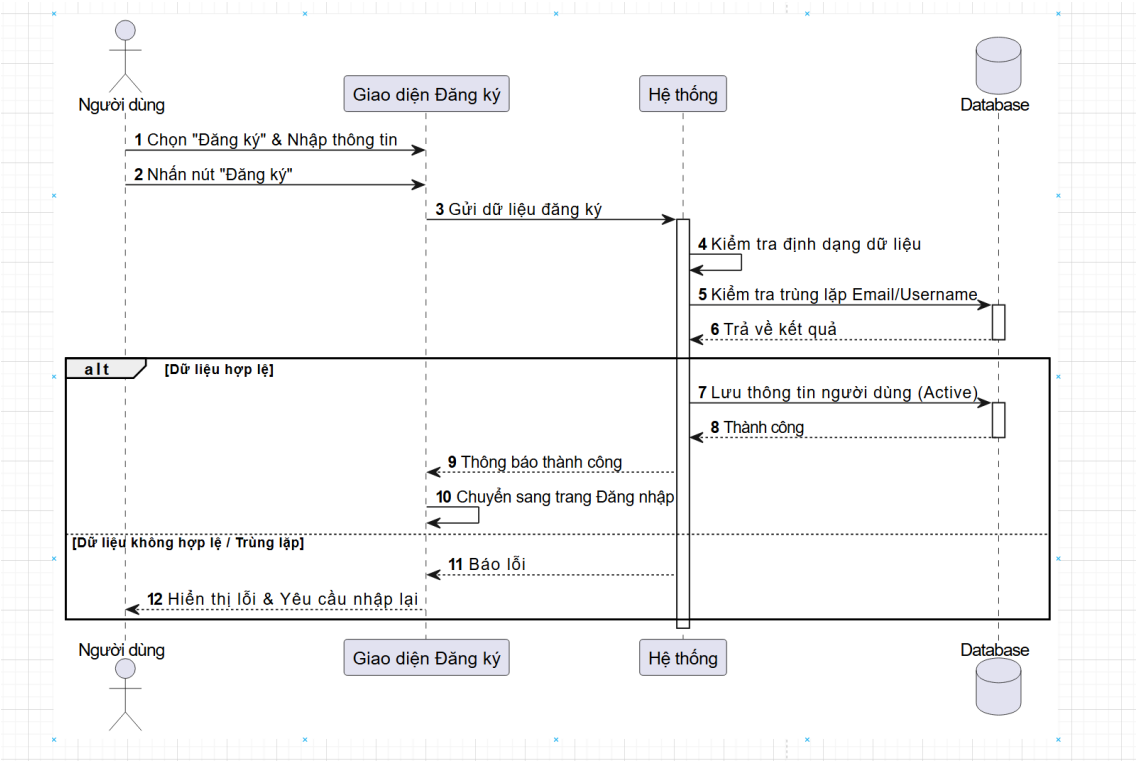
2.4.2 Biểu đồ hoạt động của người quản trị



Hình 2.4.2 Biểu đồ hoạt động của Người quản trị

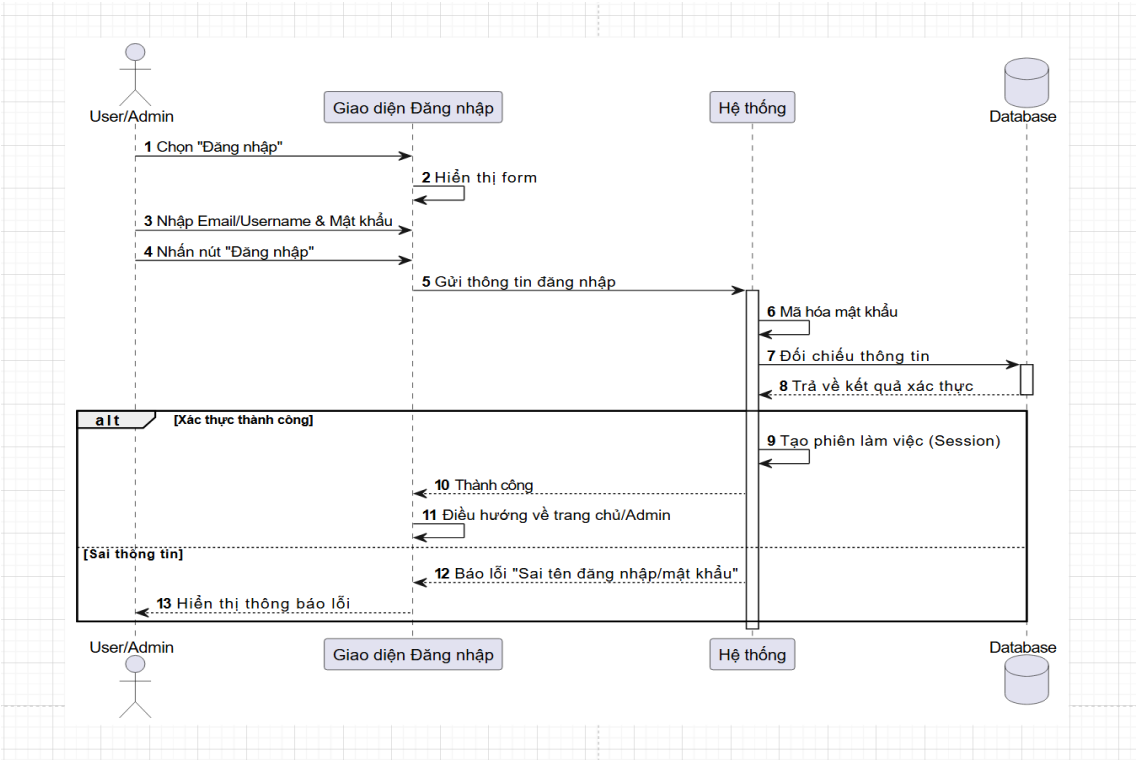
2.5. Biểu đồ tuần tự

2.5.1 Đăng ký



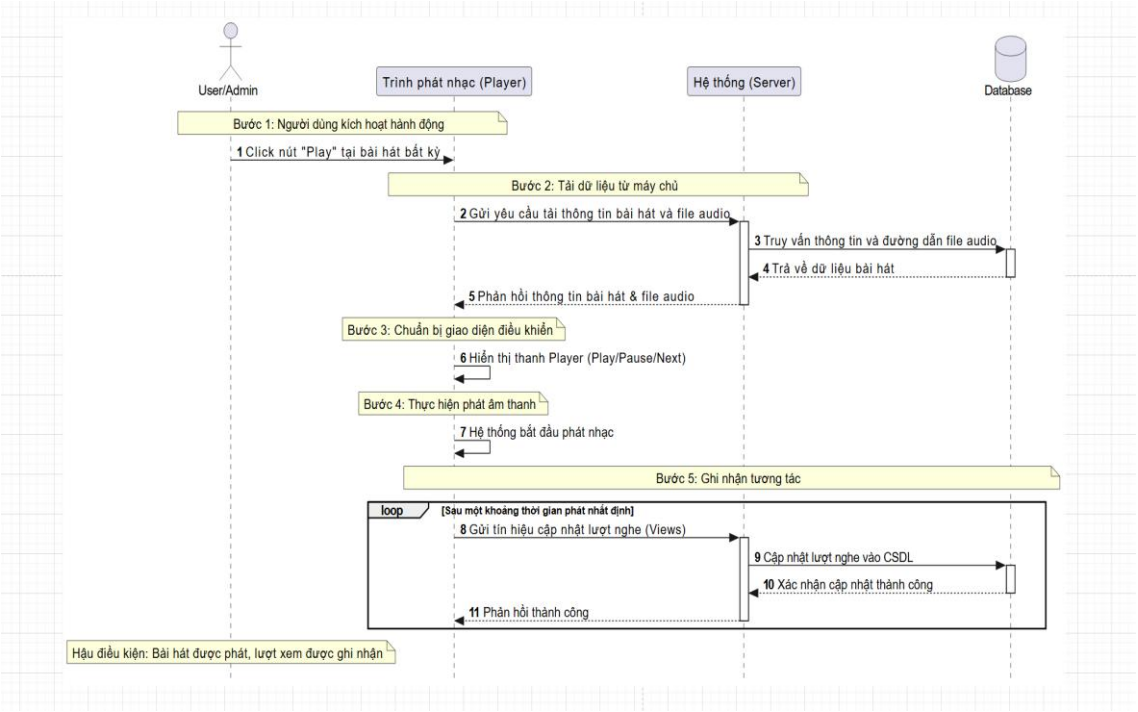
Hình 2.5.1 Biểu đồ tuần tự cho Đăng ký

2.5.2 *Đăng nhập*



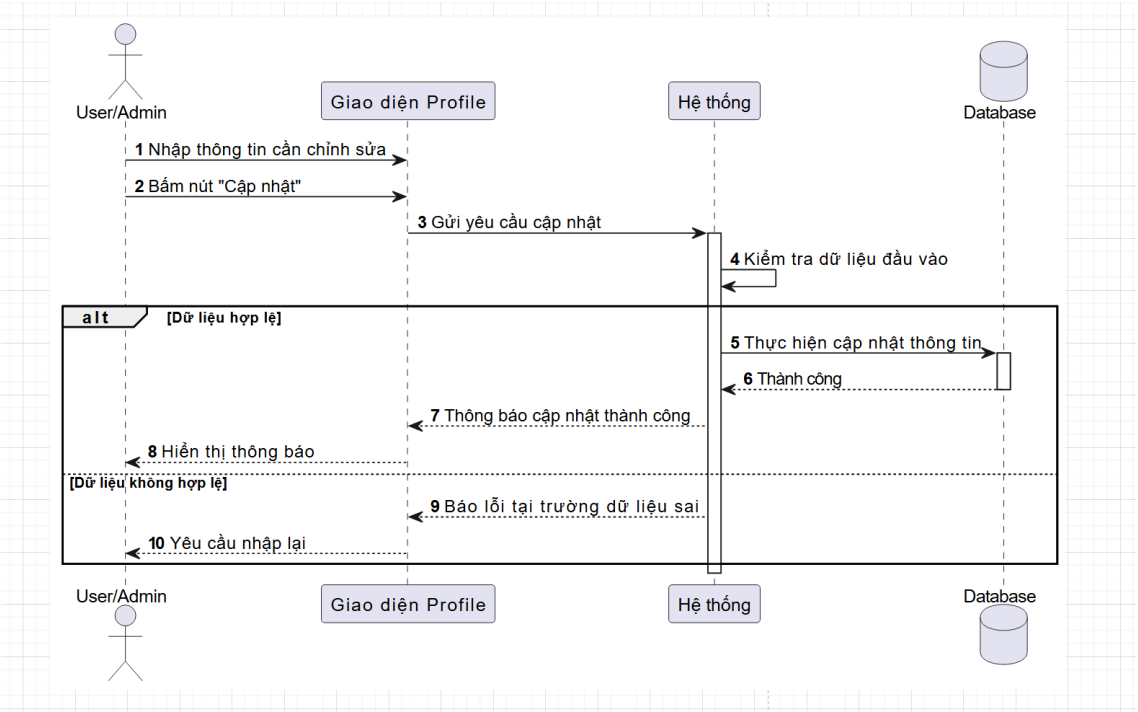
Hình 2.5.2 Biểu đồ tuần tự cho Đăng nhập

2.5.3 *Nghe nhạc*



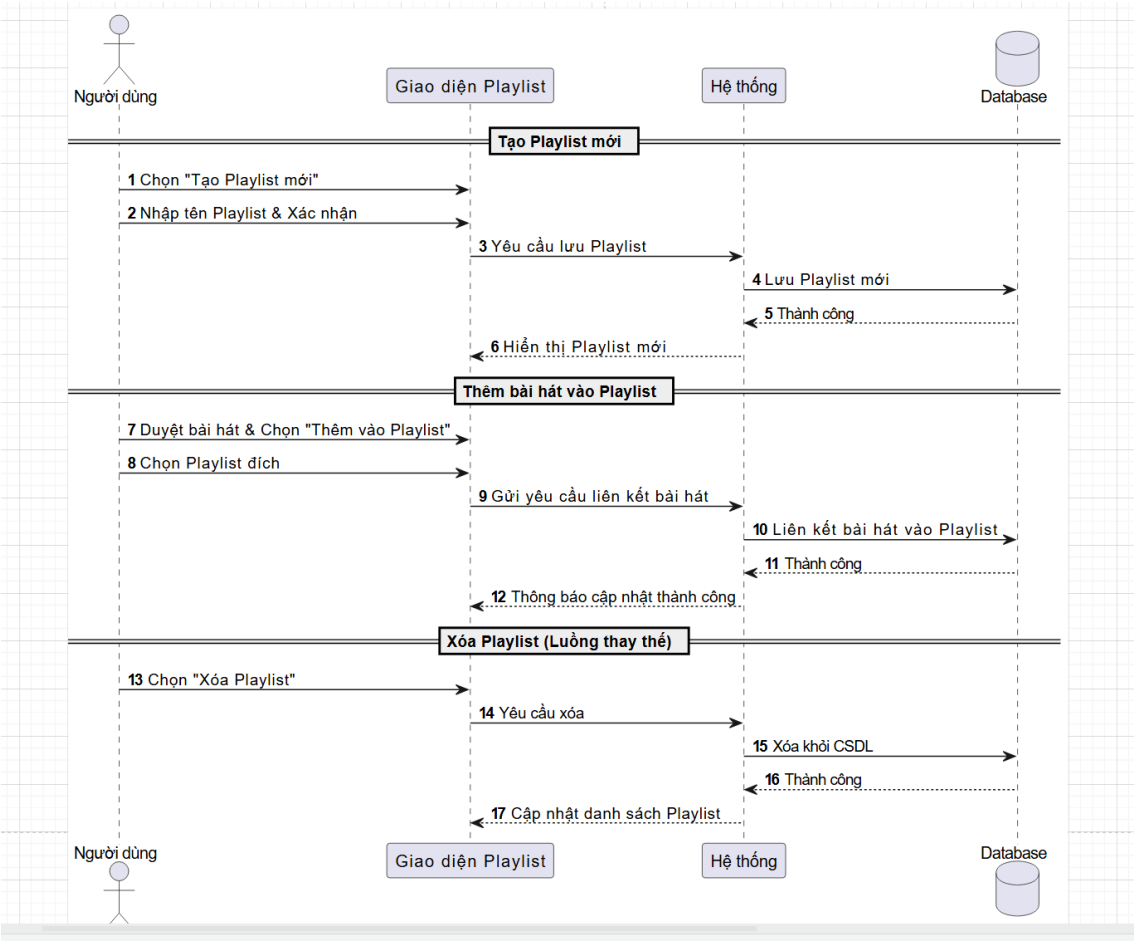
Hình 2.5.3 Biểu đồ tuần tự cho Nghe nhạc

2.5.4 *Quản lý tài khoản*



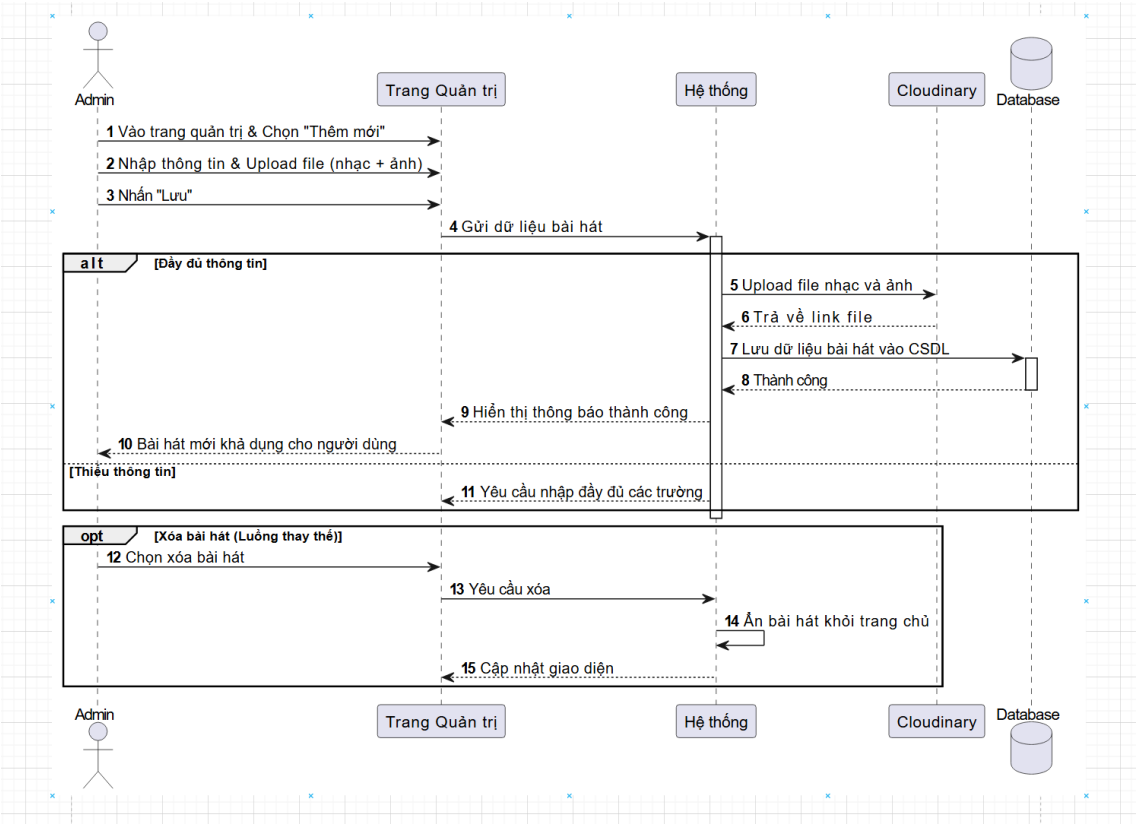
Hình 2.5.4 Biểu đồ tuần tự cho Quản lý tài khoản

2.5.5 *Quản lý playlist*



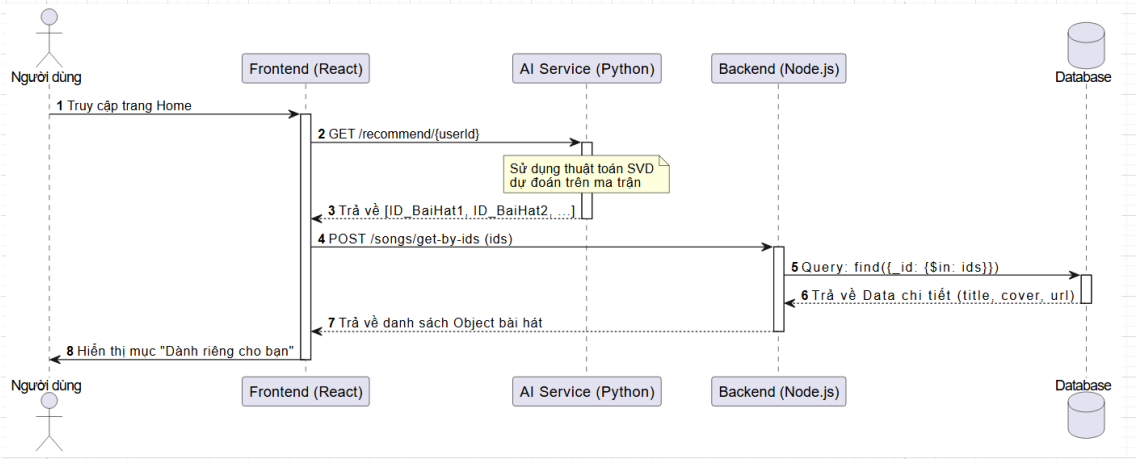
Hình 2.5.5 Biểu đồ tuần tự cho Quản lý playlist

2.5.6 *Quản lý bài hát*



Hình 2.5.6 Biểu đồ tuần tự cho Quản lý bài hát

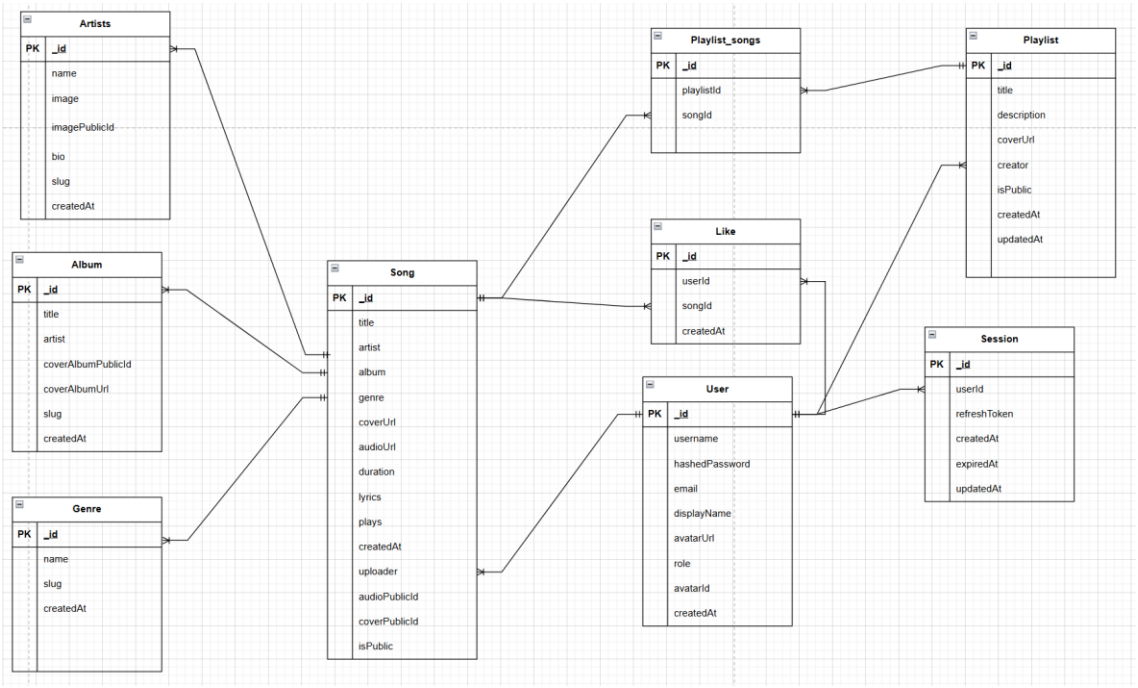
2.5.7 *AI Gợi ý bài hát*



Hình 2.5.7 Biểu đồ tuần tự cho AI gợi ý bài hát

2.6. Thiết kế cơ sở dữ liệu

2.6.1 Mô hình thiết kế cơ sở dữ liệu



Hình 2.6.1 Thiết kế cơ sở dữ liệu

2.6.2 Chi tiết các bảng dữ liệu

users			
Tên cột	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
_id	ObjectID / UUID	PK, Not Null	Mã định danh duy nhất của người dùng.
username	String	Unique, Not Null	Tên đăng nhập (duy nhất).
hashedPassword	String	Not Null	Mật khẩu đã được mã hóa bảo mật.
email	String	Unique, Not Null	Địa chỉ email dùng để liên lạc/khôi phục.
displayName	String	Not Null	Tên hiển thị công khai.
avatarUrl	String	Optional	Đường dẫn đến hình ảnh đại diện.
role	String	Default: 'user'	Vai trò hệ thống (admin, user, artist).
createdAt	DateTime	Not Null	Thời điểm tạo tài khoản.
avatarId	String	Optional	ID quản lý file ảnh trên hệ thống lưu trữ

Hình 2.6.2.1 Bảng dữ liệu users

songs			
Tên cột	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
_id	ObjectID	PK, Not Null	Mã định danh duy nhất của bài hát.
title	String	Not Null	Tiêu đề bài hát.
artist	String	Not Null	Tên nghệ sĩ trình bày.
album	String	Optional	Tên album chứa bài hát.
genre	String	Optional	Thể loại âm nhạc.
coverUrl	String	Optional	Đường dẫn ảnh bìa bài hát.
audioUrl	String	Not Null	Đường dẫn tệp tin âm thanh.
duration	Number	Not Null	Thời lượng (giây).
lyrics	Text	Optional	Lời bài hát.
plays	Integer	Default: 0	Số lượt nghe.
createdAt	DateTime	Not Null	Ngày tải lên hệ thống.
uploader	ObjectID	FK (User), Not Null	ID người dùng đã tải lên.
audioPublicId	String		ID tệp âm thanh trên Cloudinary
coverPublicId	String		ID ảnh bìa trên Cloudinary.
isPublic	Boolean	Default: true	Trạng thái hiển thị

Hình 2.6.2.2 Bảng dữ liệu songs

playlist			
Tên cột	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
_id	ObjectID	PK, Not Null	Mã định danh danh sách phát.
title	String	Not Null	Tên danh sách phát.
description	String	Optional	Mô tả về danh sách phát.
coverUrl	String	Optional	Đường dẫn ảnh bìa playlist.
creator	ObjectID	FK (User), Not Null	ID người tạo playlist.
ispublic	Boolean	Default: True	Chế độ công khai hoặc riêng tư.
createdAt	DateTime	Not Null	Ngày tạo playlist.
updatedAt	DateTime	Not Null	Ngày cập nhật cuối cùng.

Hình 2.6.2.3 Bảng dữ liệu playlist

Playlist_songs			
Tên cột	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
_id	ObjectID	PK, Not Null	Mã định danh bản ghi.
playlistId	ObjectID	FK (Playlist), Not Null	ID của playlist tương ứng.
songId	ObjectID	FK (Song), Not Null	ID của bài hát tương ứng.

Hình 2.6.2.4 Bảng dữ liệu playlist_songs

Like			
Tên cột	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
_id	ObjectID	PK, Not Null	Mã định danh lượt thích.
userId	ObjectID	FK (User), Not Null	ID người dùng nhấn thích.
songId	ObjectID	FK (Song), Not Null	ID bài hát được thích.
createdAt	DateTime	Not Null	Thời điểm nhấn thích.

Hình 2.6.2.5 Bảng dữ liệu Like

Session			
Tên cột	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
_id	ObjectID	PK, Not Null	Mã định danh phiên.
userId	ObjectID	FK (User), Not Null	ID người dùng đăng nhập.
refreshToken	String	Not Null	Token để duy trì đăng nhập.
createdAt	DateTime	Not Null	Thời điểm bắt đầu phiên.
expiredAt	DateTime	Not Null	Thời điểm hết hạn phiên.
updatedAt	DateTime	Not Null	Thời điểm cập nhật phiên.

Hình 2.6.2.6 Bảng dữ liệu Session

Artist			
Tên cột	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
_id	ObjectID	PK, Not Null	Mã định danh nghệ sĩ
name	String	Not Null	Tên nghệ sĩ
iamge	String	Optional	Đường dẫn ảnh của nghệ sĩ
imagePublicId	String		ID ảnh trên Cloudinary
bio	String	Not Null	Tiểu sử/Thông tin giới thiệu của nghệ sĩ
slug	String	Not Null	Chuỗi định danh không dấu cho URL (VD: son-tung-mtp)

Hình 2.6.2.7 Bảng dữ liệu Artist

Album			
Tên cột	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
_id	ObjectID	PK, Not Null	Mã định danh album.
title	String	Not Null	Tên album.
artist	ObjectID	Not Null	Liên kết tới nghệ sĩ sở hữu album
coverUrl	String	Optional	Đường dẫn ảnh bìa album
coverPublicId	String	Not Null	ID ảnh bìa trên Cloudinary
slug	String	Not Null	Chuỗi định danh URL (VD: bat-no-len)
createdAt	DateTime	Not Null	Ngày tạo album.

Hình 2.6.2.8 Bảng dữ liệu Album

Genre			
Tên cột	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
_id	ObjectID	PK, Not Null	Mã định danh thể loại
name	String	Not Null	Tên thể loại nhạc
slug	String	Not Null	Chuỗi định danh không dấu cho URL

Hình 2.6.2.9 Bảng dữ liệu Genre

CHƯƠNG III: XÂY DỰNG VÀ TRIỂN KHAI HỆ THỐNG

3.1. Môi trường phát triển và công cụ sử dụng

3.1.1. Môi trường phần cứng và phần mềm

- Hệ điều hành: Sử dụng Microsoft Windows 11 làm môi trường phát triển chính.
- Ngôn ngữ lập trình: TypeScript cho phần Frontend, JavaScript ES6+ cho Backend và Python 3.10 cho dịch vụ AI.
- Công cụ lập trình (IDE): Visual Studio Code tích hợp các tiện ích như ESLint, Prettier và GitLens nhằm hỗ trợ soạn thảo và đảm bảo chất lượng mã nguồn.
- Hệ quản trị cơ sở dữ liệu: MongoDB Atlas được sử dụng làm cơ sở dữ liệu cloud.
- Lưu trữ tệp đa phương tiện: Cloudinary được dùng để quản lý, tối ưu và phân phối tệp âm thanh và hình ảnh.

3.1.2. Công nghệ và Thư viện

- Frontend: React.js (Vite), Zustand (Quản lý trạng thái), TailwindCSS (Giao diện), Lucide React (Icon).
- Backend: Node.js, Express.js, Mongoose ODM.
- AI Service: FastAPI (Python), Scikit-Surprise (Machine Learning), Pandas (Xử lý dữ liệu).
- Xử lý Media: FFmpeg (Chuyển đổi video/audio).

3.2. Xây dựng hệ thống

3.2.1. Backend

Hệ thống áp dụng mô hình MVC (Model-View-Controller). Được tổ chức theo các module chức năng:

- Models: Sử dụng mongoose để định nghĩa Schema cho MongoDB, đảm bảo tính chặt chẽ của dữ liệu thông qua các ràng buộc
- Controllers: Chứa logic xử lí nghiệp vụ, tiếp nhận yêu cầu từ API và điều phối dữ liệu.
- Middlewares: Xử lý các tác vụ trung gian như kiểm tra Token JWT, phân quyền Admin và bắt lỗi hệ thống.

Hệ thống có cơ chế xác thực và phân quyền: Sử dụng thư viện bcrypt để mã hóa mật khẩu 10 lớp trước khi lưu. Triển khai JWT kép với AccessToken ngắn hạn và RefressToken dài hạn.

Tích hợp Multer-storage-cloudinary để truyền luồng dữ liệu tệp tin trực tiếp lên đám mây. Hệ thống sử dụng cấu trúc try-catch-finally với chức năng là nếu lưu thông tin và MongoDB thất bại thì một tín hiệu xóa (cloudinary.uploader.destroy) sẽ được gửi đi ngay lập tức dựa trên public_id để dọn dẹp bộ nhớ Cloud, tránh dư thừa dữ liệu.

Triển khai một tiến trình con (Child Process) để chạy FFmpeg. Hệ thống thực hiện bóc tách luồng Audio từ file Video của người dùng và chuyển mã về định dạng MP3 trước khi lưu trữ.

3.2.2. Frontend

Quản lý trạng thái toàn cục với hai store chính là PlayerStore và AuthStore. PlayerStore thì quản lý toàn bộ logic trình phát bài hát đang phát, hàng chờ (queue), chỉ mục (index), âm lượng, trạng thái xáo bài (shuffle), lặp lại (repeat). Còn AuthStore thì quản lý thông tin định danh và Token.

Module Trình phát nhạc có các tác vụ phát/tạm dừng, chuyển bài, tua, xáo bài và điều chỉnh âm lượng, có ba chế độ lặp là Tắt, Lặp lại danh sách và Lặp lại một bài. Để tránh lượt nghe ảo, hệ thống triển khai bộ đếm thời gian thực actualSecondsRef để khi bài hát được nghe liên tục từ 15 giây trở lên mới tính là 1 lượt nghe (plays).

Hàng chờ danh sách phát là một Sidetab hiển thị danh sách 10 bài hát sắp được phát và danh sách các bài hát đã nghe.

Sử dụng giao diện bảng hỗ trợ phân trang, chỉnh sửa bài hát hoặc xóa, dễ dàng quản lý thư viện nhạc.

Sử dụng React Portal để tạo Modal chỉnh sửa thông tin nằm trên cùng mọi lớp giao diện.

3.2.3. Cấu hình AI

Hệ thống gợi ý bài hát (Recommendation System) được xây dựng dựa trên phương pháp Lọc cộng tác (Collaborative Filtering) với thuật toán SVD (Singular Value Decomposition).

Dữ liệu được trích xuất từ MongoDB dưới dạng ma trận tương tác (Utility Matrix) và chuyển đổi sang định dạng CSV với 3 cột chính là: user_id, song_id và rating.

Sử dụng thư viện Scikit-Surprise của Python để dựng một ma trận lớn, trong đó hàng là Người dùng, cột là Bài hát. Các ô giao nhau chứa điểm số (5 hoặc để trống).

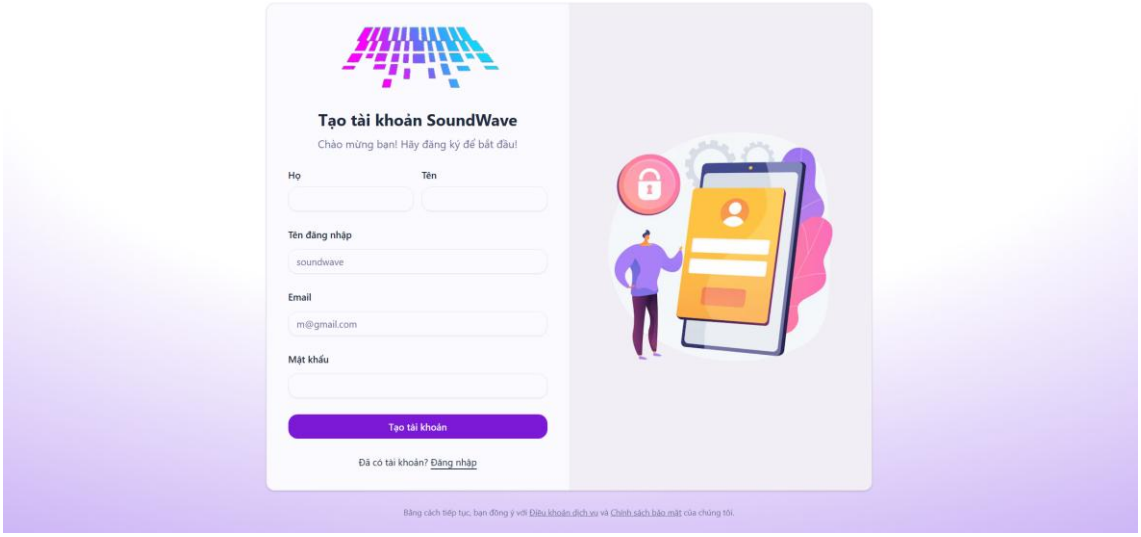
Thuật toán SVD phân tách ma trận thưa thớt (nhiều ô trống) thành 3 ma trận nhỏ hơn để tìm ra các "Yếu tố ẩn" (Latent Factors). Nén toàn bộ kết quả huấn luyện vào tệp music_model.pkl.

Khi người dùng truy cập Trang chủ, quy trình dự đoán được kích hoạt qua FastAPI, hệ thống lấy danh sách tất cả các bài hát hiện có và loại bỏ các bài hát người dùng đã thích (Filtering) giúp người dùng không nghe lại những bài hát đã biết.

Với mỗi bài hát chưa biết, thực hiện phép nhân Vector Người dùng với Vector bài hát đó. Kết quả là một điểm số dự báo (Predicted Rating) từ 1.0 đến 5.0. Và hệ thống lọc lấy 20 bài có điểm cao nhất, sau đó chọn ngẫu nhiên khoảng 10 bài để hiển thị trên giao diện.

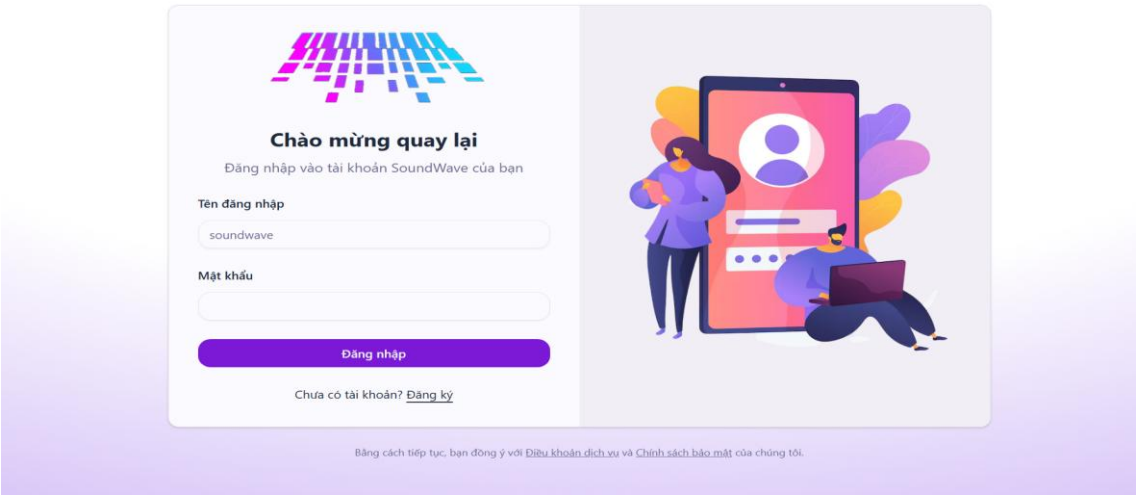
3.3. Kết quả DEMO

3.3.1 Đăng ký



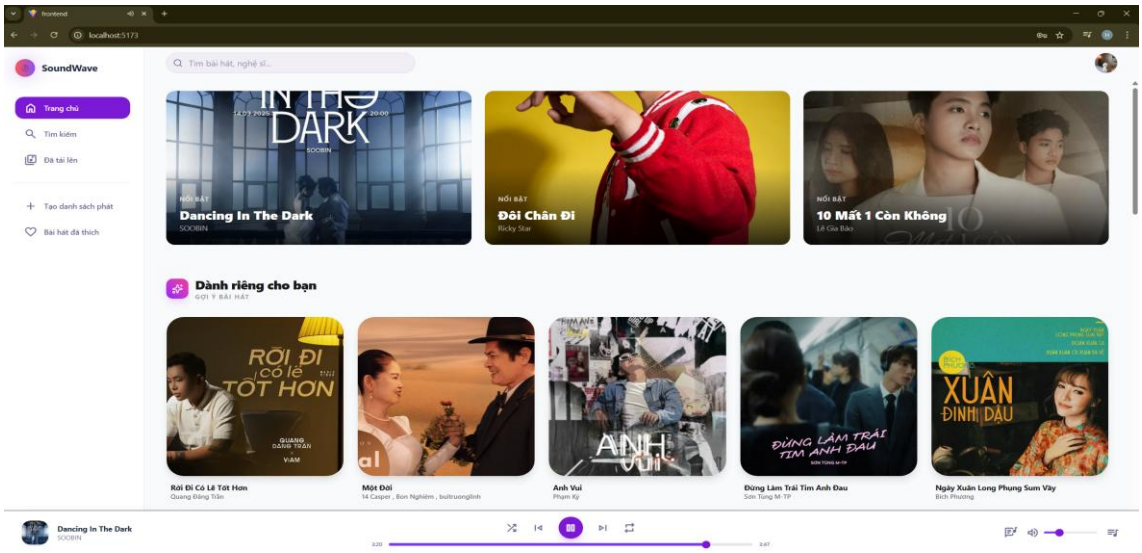
Hình 3.3.1 Màn hình đăng ký

3.3.2 Đăng nhập



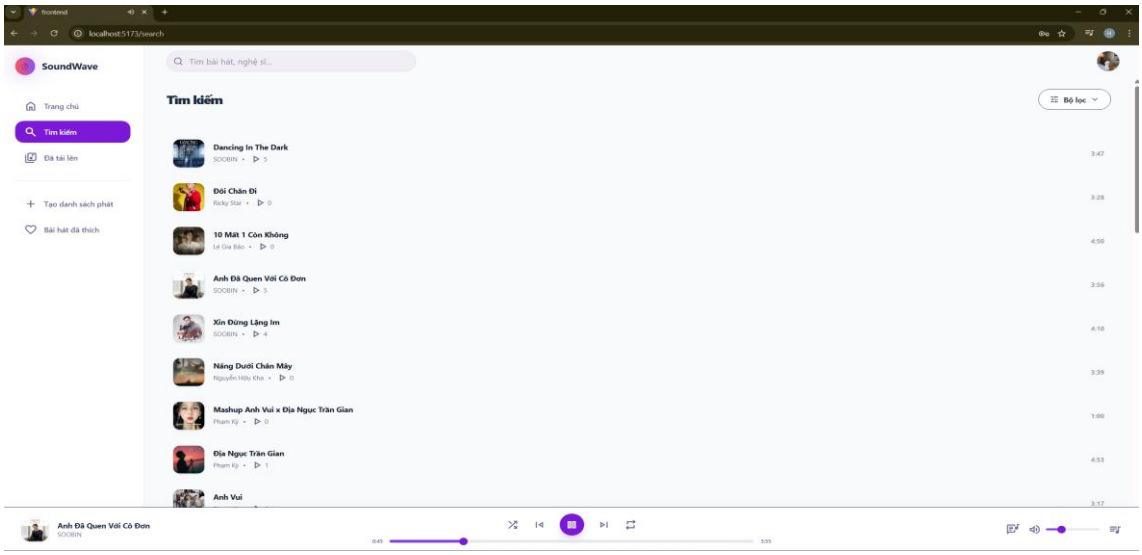
Hình 3.3.2 Màn hình đăng nhập

3.3.3 Trang chủ



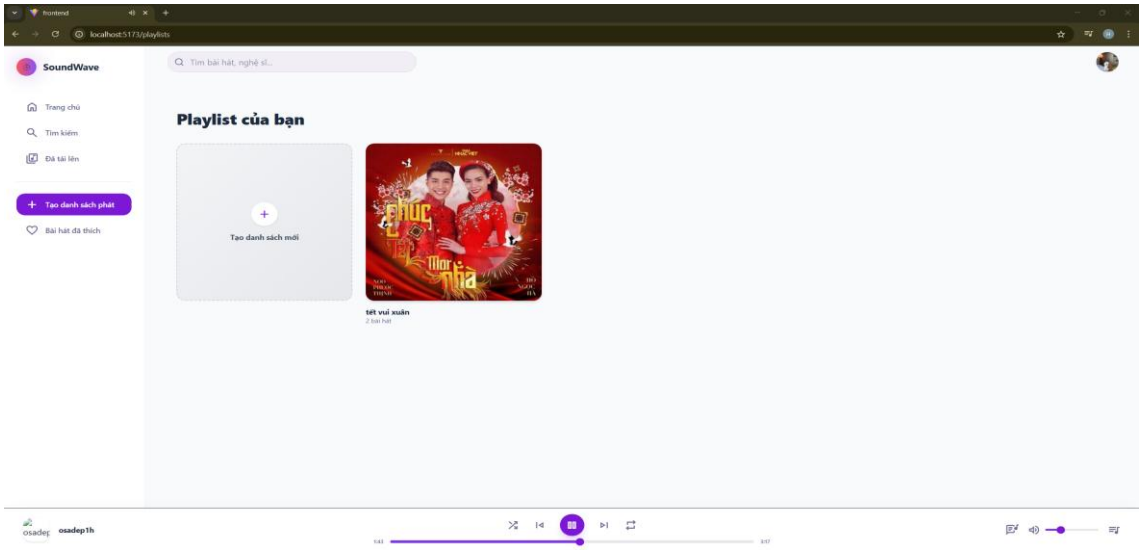
Hình 3.3.3 Màn hình trang chủ

3.3.4 Tìm kiếm



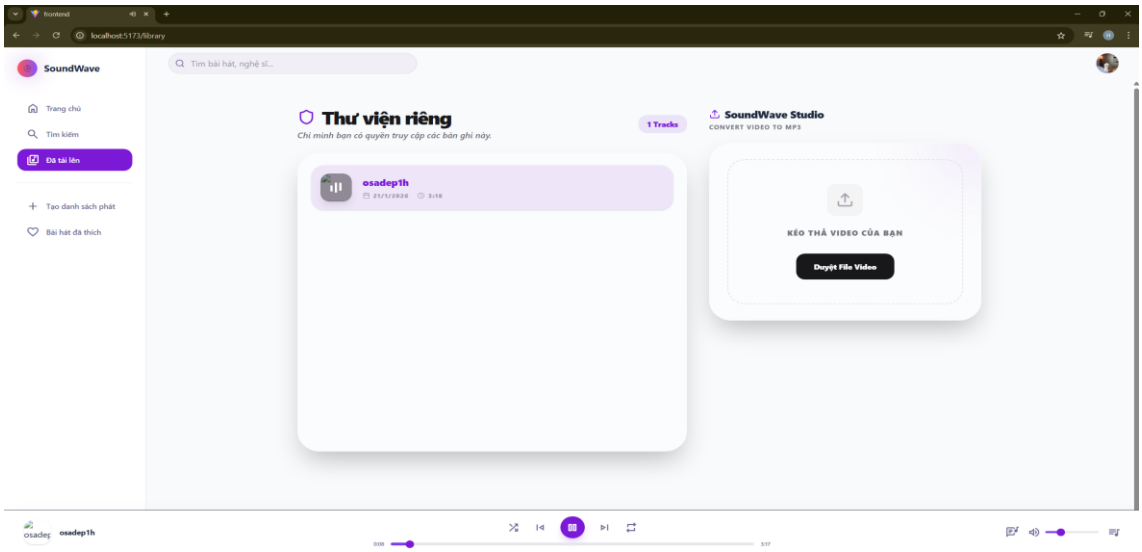
Hình 3.3.4 Màn hình tìm kiếm

3.3.5 Playlist



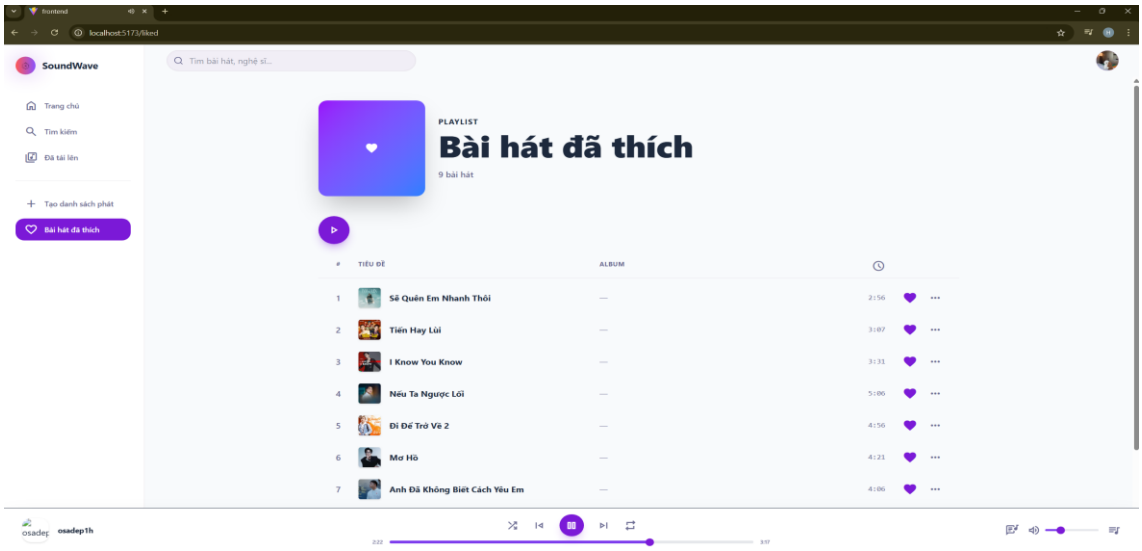
Hình 3.3.5 Màn hình Playlist

3.3.6 Đã tải lên



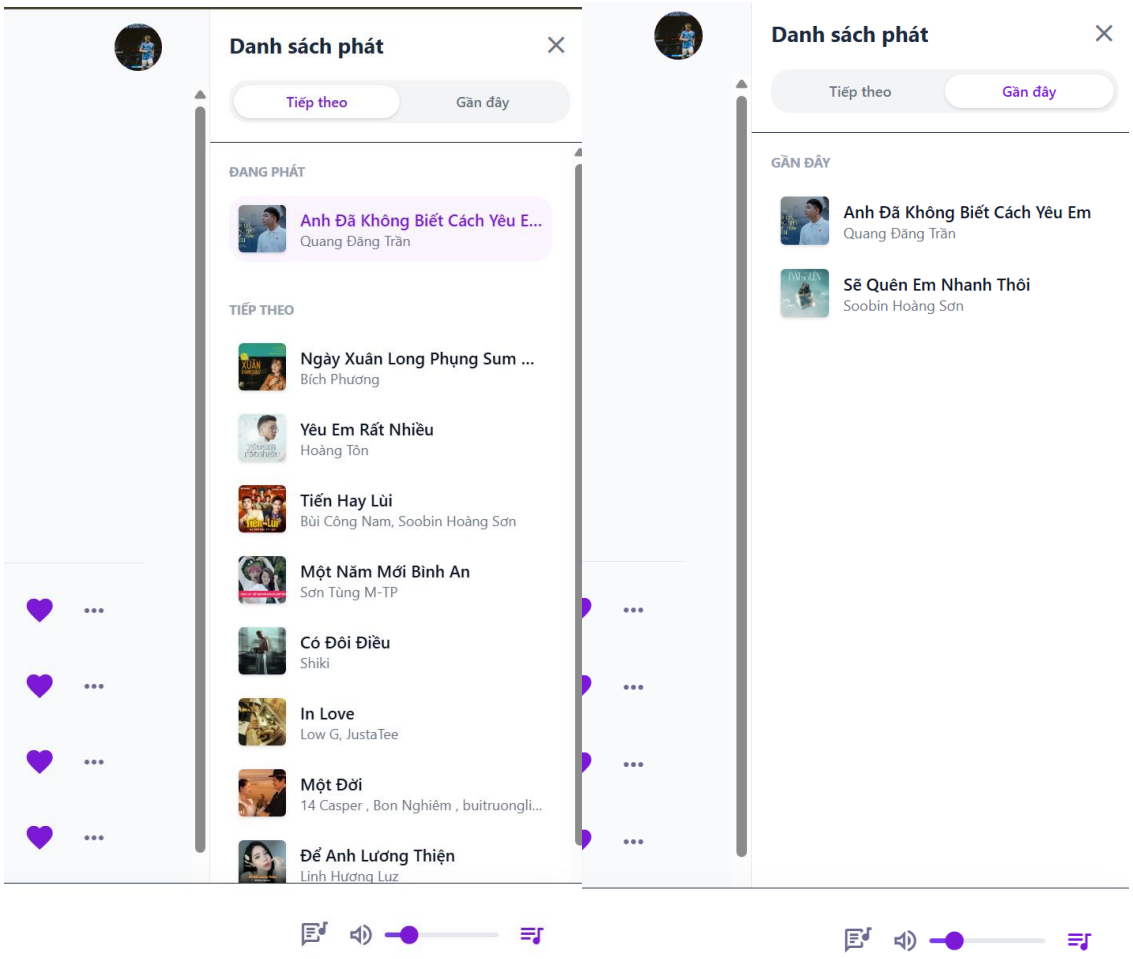
Hình 3.3.6 Màn hình Đã tải lên

3.3.7 Bài hát đã thích



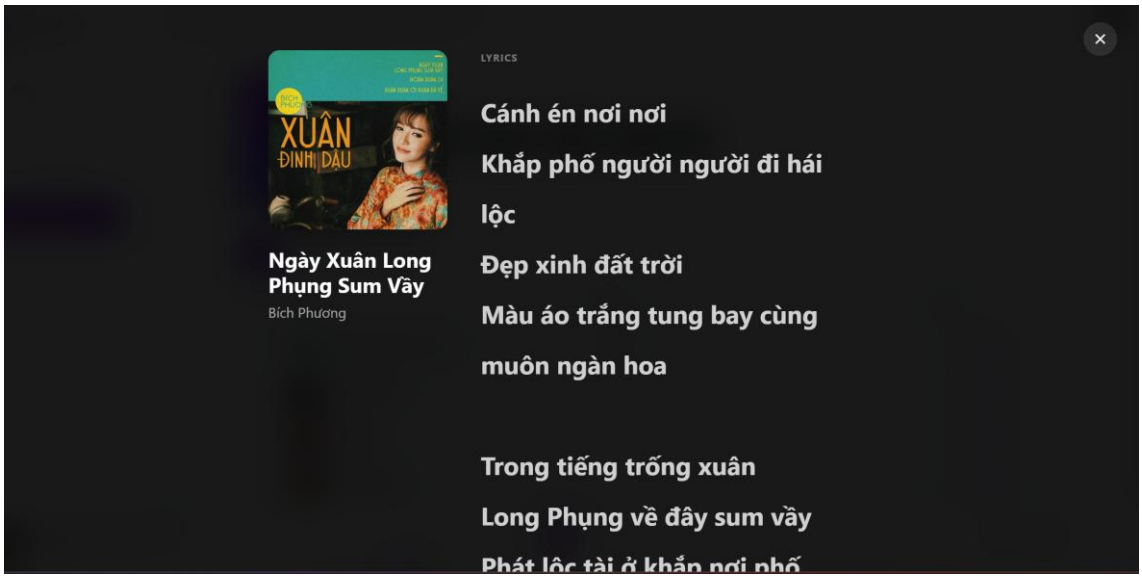
Hình 3.3.7 Màn hình Bài hát đã thích

3.3.8 Danh sách phát và lịch sử nghe



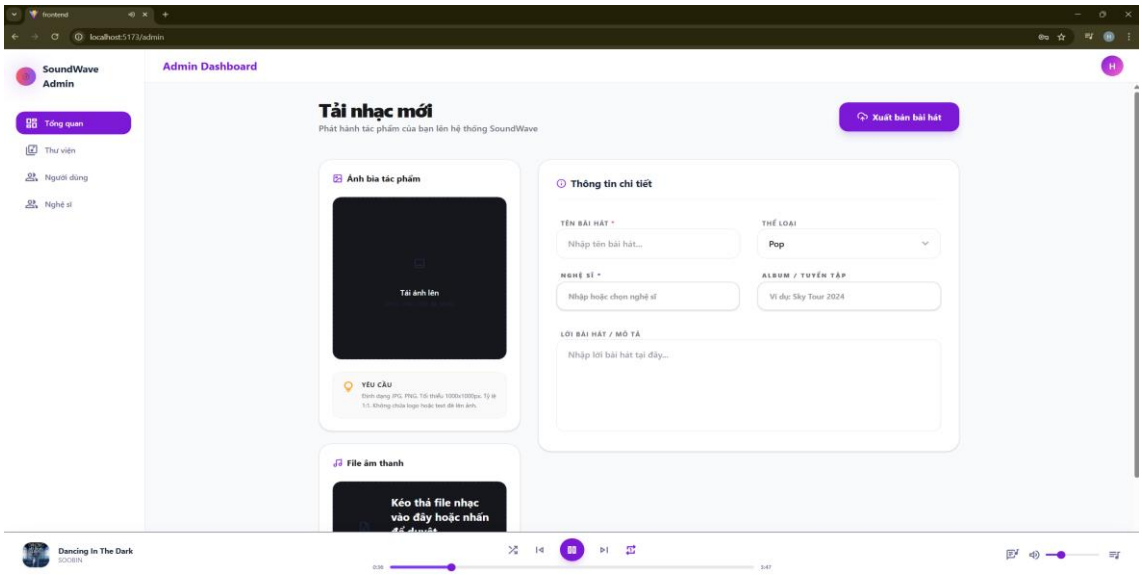
Hình 3.3.8 Mục danh sách phát và lịch sử nghe

3.3.9 Lời bài hát



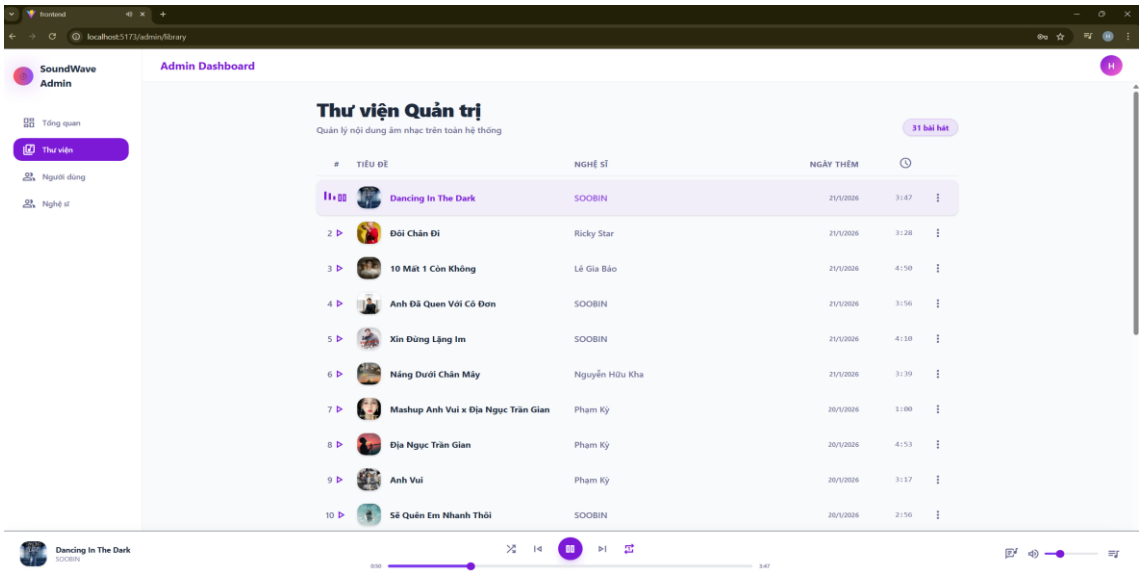
Hình 3.3.9 Màn hình hiển thị Lời bài hát

3.3.10 Tải nhạc lên



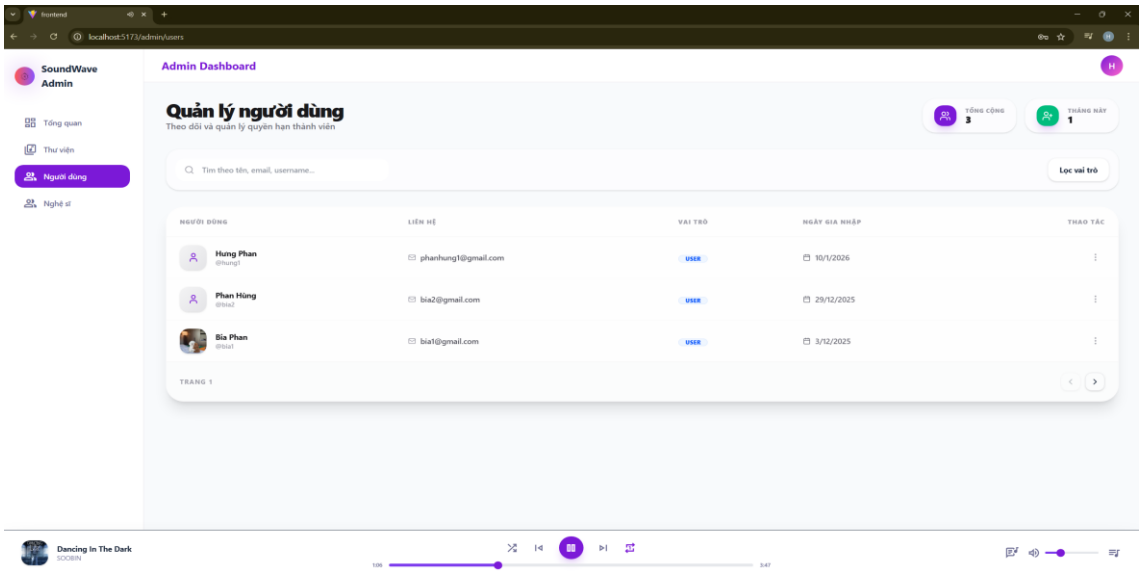
Hình 3.3.10 Màn hình tải nhạc lên cho Admin

3.3.11 Thư viện quản trị



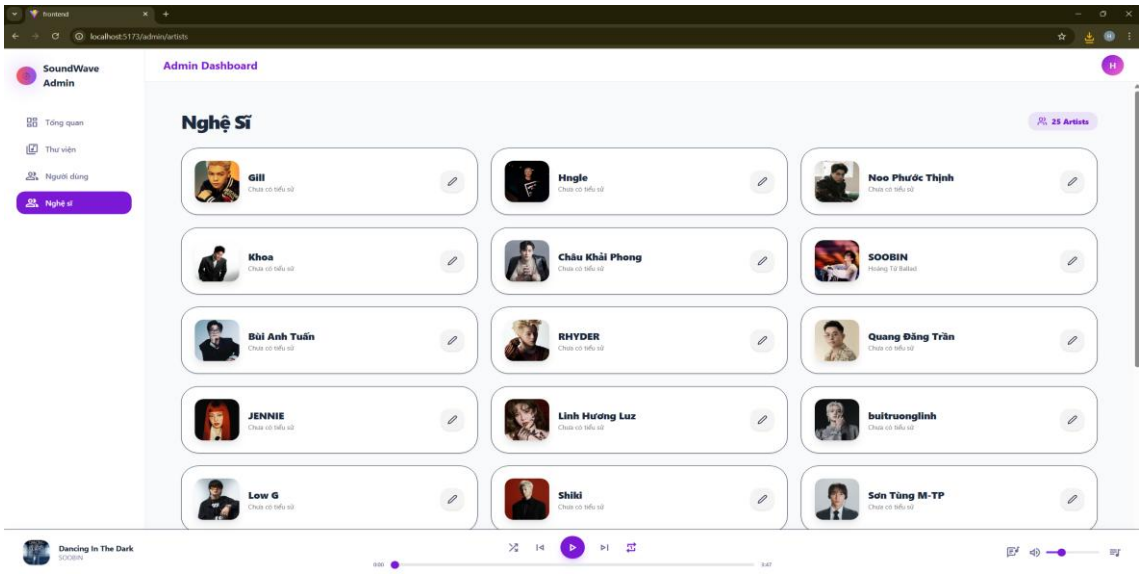
Hình 3.3.11 Màn hình thư viện quản trị bài hát

3.3.12 Quản lý người dùng



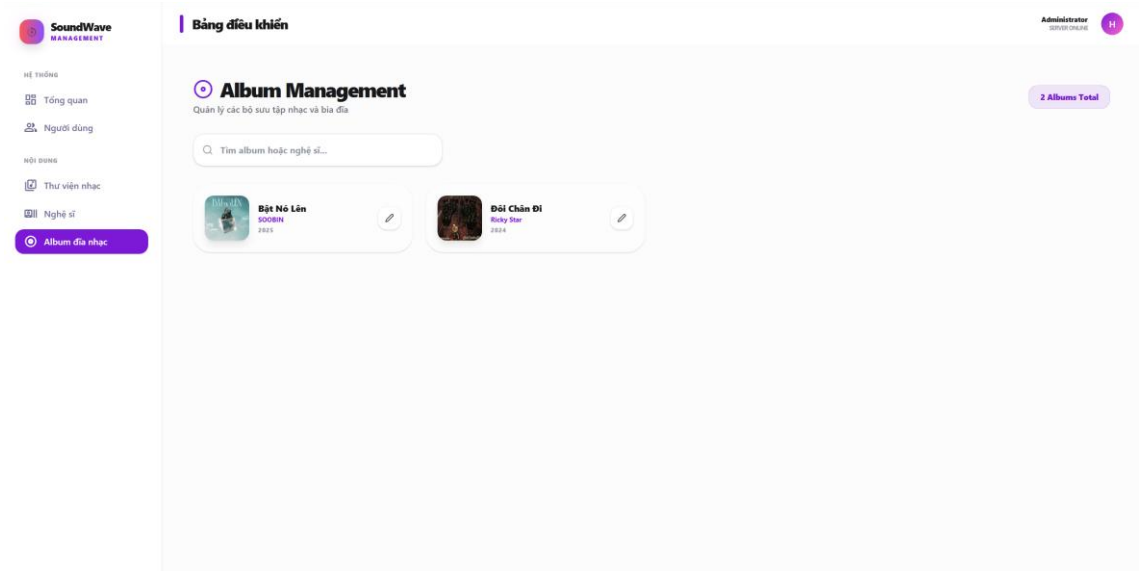
Hình 3.3.12 Màn hình quản lý người dùng

3.3.13 Quản lý Nghệ sĩ



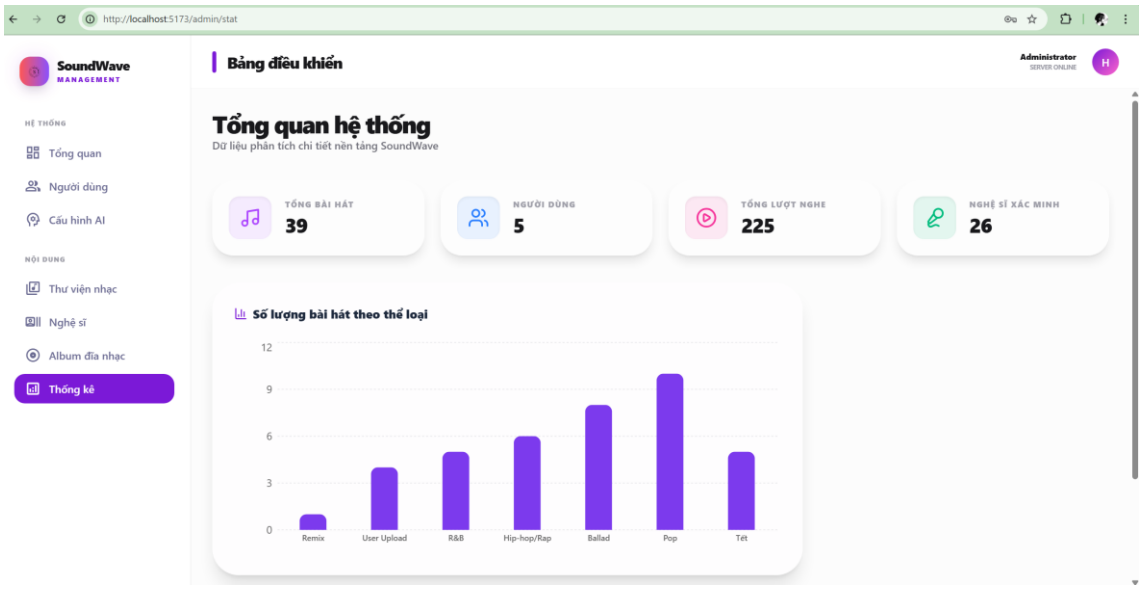
Hình 3.3.13 Màn hình quản lý nghệ sĩ

3.3.14 Quản lý Album



Hình 3.3.14 Màn hình quản lý album

3.3.15 Quản lý Thống kê



Hình 3.3.15 Màn hình thống kê

3.4. Đánh giá kết quả

Hệ thống đã hoàn thiện các chức năng cốt lõi theo đúng mục tiêu đặt ra ban đầu, bao gồm quản lý người dùng, phát nhạc trực tuyến, tương tác với playlist và hiển thị nội dung theo danh mục. Việc áp dụng mô hình Client – Server giúp tách biệt rõ ràng giữa giao diện người dùng và xử lý dữ liệu, tạo điều kiện thuận lợi cho việc mở rộng về sau. Công nghệ được lựa chọn phù hợp với xu hướng hiện tại, đảm bảo tốc độ phát triển nhanh, dễ bảo trì và khả năng tích hợp với các dịch vụ lưu trữ đám mây. Dưới đây là những đánh giá chi tiết:

- Hiệu quả sử dụng

Hệ thống vận hành ổn định trong quá trình thử nghiệm với số lượng người dùng mô phỏng nhỏ và trung bình. Các thao tác như đăng nhập, tải danh sách nhạc, phát/chuyển bài diễn ra nhanh chóng và ít xảy ra lỗi. Việc sử dụng Cloudinary/một dịch vụ lưu trữ đám mây giúp giảm tải cho máy chủ backend, tối ưu băng thông và nâng cao hiệu năng truy cập nội dung đa phương tiện. Kiến trúc API RESTful cho phép frontend và backend giao tiếp hiệu quả, đồng thời hỗ trợ dễ dàng mở rộng thêm tính năng mới trong tương lai như bình luận, chia sẻ hoặc livestream audio.

- Trải nghiệm người dùng

Giao diện người dùng được thiết kế đơn giản, dễ tiếp cận với các nhóm đối tượng phổ thông. Bố cục các trang rõ ràng, người dùng có thể nhanh chóng tìm kiếm bài hát, tạo playlist hoặc quản lý tài khoản mà không mất nhiều thao tác. Tính năng phát nhạc hoạt động trơn tru, không bị gián đoạn khi chuyển trang, mang đến trải nghiệm nghe nhạc liền mạch. Hệ thống hỗ trợ tương tác thân thiện, phản hồi trực quan khi người dùng click, chọn bài hát hoặc thao tác với danh sách phát. Nhìn chung, mức độ hài lòng của người dùng thử nghiệm cao với nhận xét tích cực về tốc độ xử lý và khả năng sử dụng dễ dàng.

CHƯƠNG IV: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Dự án **SoundWave** đã hoàn thành mục tiêu xây dựng một nền tảng nghe nhạc trực tuyến hiện đại, mượt mà và tối ưu trải nghiệm người dùng. Việc kết hợp sức mạnh của hệ sinh thái JavaScript đã mang lại những kết quả cụ thể, không chỉ đáp ứng các yêu cầu ban đầu mà còn mở ra nhiều cơ hội phát triển ứng dụng trong tương lai.

4.1. Những kết quả đạt được

4.1.1. Về mặt lý thuyết

Trong quá trình thực hiện đề tài, em đã tích lũy được nhiều kiến thức nền tảng và chuyên sâu liên quan đến lĩnh vực phát triển ứng dụng web. Cụ thể, người thực hiện đã nắm vững kiến thức về mô hình hoạt động của hệ thống web, bao gồm cấu trúc Client–Server, cơ chế giao tiếp giữa frontend và backend thông qua API, cũng như vai trò của cơ sở dữ liệu trong lưu trữ và quản lý thông tin. Ngoài ra, đề tài giúp củng cố hiểu biết về các khái niệm quan trọng trong lập trình như bất đồng bộ, xử lý yêu cầu bất tuyến tính, quản lý trạng thái ứng dụng và bảo mật tài nguyên trên web.

Bên cạnh đó, việc nghiên cứu các công nghệ như ReactJS, NodeJS, Express và MongoDB giúp mở rộng kiến thức về lập trình hướng thành phần, thiết kế RESTful API, cấu trúc dữ liệu NoSQL và các phương pháp tổ chức dự án phần mềm hiện đại. Người thực hiện cũng tiếp cận được nguyên lý tối ưu hóa trải nghiệm người dùng, phân tích yêu cầu, thiết kế chức năng và đánh giá hiệu năng hệ thống.

Nhìn chung, về mặt lý thuyết, đề tài đã mang lại nền tảng kiến thức vững chắc về phát triển ứng dụng web full-stack, đồng thời tạo cơ sở để tiếp tục nghiên cứu, nâng cấp hệ thống và ứng dụng các công nghệ mới trong tương lai.

4.1.2. Về mặt ứng dụng

Về mặt ứng dụng, đề tài đã xây dựng được hệ thống web nghe nhạc hoạt động theo đúng yêu cầu đặt ra ban đầu. Ứng dụng cho phép người dùng đăng ký và đăng nhập tài khoản, tìm kiếm và nghe nhạc trực tuyến, quản lý playlist cá nhân, cũng như xem thông tin bài hát và nghệ sĩ. Các chức năng chính đã được tích hợp đầy đủ, từ giao diện frontend đến xử lý dữ liệu phía backend, giúp hệ thống vận hành ổn định và dễ sử dụng.

Hệ thống cũng áp dụng thành công các công nghệ hiện đại như ReactJS ở phía giao diện, ExpressJS và NodeJS cho xử lý nghiệp vụ, cùng với MongoDB để lưu trữ dữ liệu. Việc triển khai chức năng tải và quản lý tệp nhạc thông qua dịch vụ lưu trữ cloud như Cloudinary giúp tối ưu tài nguyên máy chủ và cải thiện khả năng mở rộng hệ thống. Ngoài ra, ứng dụng còn triển khai RESTful API, giúp frontend và backend giao tiếp rõ ràng, thuận tiện cho việc phát triển, bảo trì và nâng cấp sau này.

Thể hiện tính cá nhân hóa khi mỗi tài khoản người dùng sẽ nhìn thấy một danh sách gợi ý khác nhau hoàn toàn dựa trên gu âm nhạc riêng.

Như vậy, đề tài không chỉ dừng lại ở mức lý thuyết mà đã được ứng dụng vào thực tế thông qua một hệ thống hoàn chỉnh, có khả năng triển khai và sử dụng trong môi trường thật.

4.2. Hạn chế

Mặc dù hệ thống đã đáp ứng được các yêu cầu cơ bản, nhưng vẫn còn tồn tại một số hạn chế:

Xử lý streaming còn đơn giản: Hệ thống phát nhạc dựa trên URL trực tiếp, chưa hỗ trợ tối ưu như adaptive streaming hay kiểm soát bitrate theo tốc độ mạng.

Bảo mật còn ở mức cơ bản: Các cơ chế như JWT và mã hóa mật khẩu đã có nhưng hệ thống chưa áp dụng các biện pháp nâng cao như rate limiting, xác thực hai bước (2FA) hay phát hiện xâm nhập.

Khả năng mở rộng còn hạn chế: Khi hệ thống có nhiều người dùng truy cập cùng lúc, tốc độ tải dữ liệu có thể giảm do chưa tối ưu hiệu năng và cache.

4.3. Hướng phát triển

Để dự án tiếp tục phát triển và đáp ứng nhu cầu ngày càng cao của người dùng, có một số hướng phát triển cần xem xét:

Tối ưu hiệu năng: Cải thiện tốc độ truy xuất dữ liệu âm thanh và tải trang bằng cách tối ưu cơ sở dữ liệu, caching và cấu hình server.

Mở rộng chức năng: Bổ sung các tính năng như gợi ý bài hát theo sở thích và chia sẻ nội dung.

Mô hình thương mại hóa: Sau khi thử nghiệm thành công, hệ thống có thể triển khai mô hình tài khoản cao cấp (premium), cho phép người dùng trả phí để thêm các chức năng khác

Phát triển cộng đồng: Tích hợp các tính năng mạng xã hội như tạo nhóm nghe nhạc, livestream...

TÀI LIỆU THAM KHẢO

TÀI LIỆU TIẾNG VIỆT

[1] Thiết kế RESTful APIs https://fptcloud.com/restful-api-la-gi/
[2] https://machinelearningcoban.com/2017/05/31/matrixfactorization/

TÀI LIỆU TIẾNG ANH

[1] ReactJS Documentation: https://react.dev

[2] TailwindCSS Documentation: https://tailwindcss.com/docs

[3] JWT Authentication Docs: https://jwt.io

[4] Multer Upload multer - npm

[5] ExpressJS & NodeJS Docs: https://expressjs.com & https://nodejs.org/docs

[6] MongoDB Docs: https://www.mongodb.com/docs/

[7] Cloudinary API Docs: https://cloudinary.com/documentation/

[8] Postman. (n.d.). Postman Learning Center. From https://learning.postman.com