ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

Xây dựng mạng xã hội cho người thích nghe nhạc

Trần Khánh Duy
duy.tk215016@sis.hust.edu.vn

Giảng viên hướng dẫn: ThS Vũ Đức Vượng _____

Chữ kí GVHD

Chương trình đào tạo: Việt - Nhật

Trường: Công nghệ Thông tin và Truyền thông

HÀ NỘI, 06/2025

LỜI CẢM ƠN

Lời đầu tiên, tôi xin gửi lời cảm ơn sâu sắc, chân thành nhất tới các giảng viên tại Đại học Bách Khoa Hà Nội, những người thầy người cô đã dày công truyền đạt kiến thức cho tôi trong 4 năm học qua. Đặc biệt, tôi xin gửi lời cảm ơn sâu sắc, chân thành đến ThS. Vũ Đức Vượng giảng viên Bộ môn Khoa Học Máy tính, Trường Công nghệ thông tin và Truyền thông, Đại học Bách Khoa Hà Nội, là người đã tận tình, chỉ bảo, dẫn tôi đi đúng hướng trong suốt thời gian hoàn thiện đồ án này. Tôi xin gửi lời cảm ơn tới gia đình, người thân đã động hành cùng tôi, tạo điều kiện và truyền động lực cho tôi trong suốt quả trình học ở Đại học Bách Khoa Hà Nội và quá trình hoàn thành đồ án tốt nghiệp một cách tốt nhất có thể. Cuối cùng, tôi muốn cảm ơn chính bản thân vì đã nỗ lực không ngừng, vượt qua những thử thách để đạt được kết quả như mong muốn. Đồ án này không chỉ là thành quả học tập mà còn là kỷ niệm đáng nhớ trong hành trình trưởng thành của tôi.

Do kiến thức, chuyên môn của tôi vẫn còn những hạn chế nên đồ án này không thể tránh khỏi những sai sót, nhầm lẫn. Tôi rất mong nhận được những lời nhận xét, đánh giá và đóng góp tích cực từ thầy cô và bạn bè để tôi có thể hoàn thiện sản phẩm trong tương lai.

TÓM TẮT NỘI DUNG ĐỒ ÁN

Đề tài xây dựng mạng xã hội cho những người yêu thích âm nhạc xuất phát từ nhu cầu kết nối cộng đồng đam mê âm nhạc, nơi người dùng có thể tự do thể hiện cá tính qua việc nghe, chia sẻ và sáng tạo nội dung âm nhạc, và cải thiện giao diện tương tác sử dụng ứng dụng của người dùng.

Hiện nay, các nền tảng nghe nhạc như Spotify, SoundCloud rất phổ biến nhưng không cho phép người dùng tự do upload nhạc cá nhân. Những nền tảng này thường yêu cầu kiểm duyệt nội dung chặt chẽ hoặc không đủ linh hoạt để người dùng chia sẻ nhạc tự sáng tác. Vì vậy, chúng tôi lựa chọn xây dựng một mạng xã hội âm nhạc đề cao tính cá nhân, cho phép người dùng upload nhạc tự do.

Chúng tôi sử dụng mô hình Traditional N-Layer Architecture, hay còn gọi là Layered Architecture, Spring Boot kết hợp công nghệ React và MySQL, tích hợp Jamendo API để lấy dữ liệu. Hướng tiếp cận này được chọn vì Traditional N-Layer Architecture giúp tổ chức backend rõ ràng, tối ưu hóa các chức năng phức tạp như tìm kiếm toàn văn (full-text search) và gợi ý nhạc, trong khi giao diện React mang lại trải nghiệm mượt mà, hỗ trợ trình phát nhạc (music player). Giải pháp của chúng tôi bao gồm các chức năng nghe nhạc với đầy đủ điều khiển, chia sẻ nhạc/danh sách nhạc, tìm kiếm toàn văn (full-text-search), theo tác giả, thể loại, gợi ý nhạc dựa trên sở thích, lịch sử nghe nhạc của bản thân, người dùng đang theo dõi, và tải nhạc cá nhân lên hệ thống mà không cần phân quyền phức tạp.

Đóng góp chính của đồ án là xây dựng một nền tảng âm nhạc đề cao tính cá nhân, nơi người dùng tự do sáng tạo và tương tác, cùng với việc áp dụng mô hình Traditional N-Layer Architecture hiệu quả trong thiết kế backend để dễ dàng bảo trì và mở rộng.

Kết quả đạt được là một hệ thống hoạt động có giao diện thân thiện, đáp ứng tốt các yêu cầu chức năng và mang lại trải nghiệm mới mẻ hơn cho người dùng.

Sinh viên thực hiện

(Ký và ghi rõ họ tên)

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

DANH MỤC THUẬT NGỮ VÀ TỪ VIẾT TẮT

Thuật ngữ	**Ý nghĩa**
API	Giao diện lập trình ứng dụng (Application Programming Interface)
Backend	Phần xử lý phía máy chủ của hệ thống)
Frontend	Phần giao diện của hệ thống)
HTML	Ngôn ngữ đánh dấu siêu văn bản (HyperText Markup Language)
IaaS	Dịch vụ hạ tầng
JWT	JSON Web Token, là một tiêu chuẩn mở được sử dụng để truyền tải thông tin an toàn giữa các bên dưới dạng một đối tượng JSON được mã hóa)
Maintainability	tính dễ bảo trì của hệ thống
Performance	tính hiệu năng của hệ thống
Security	tính bảo mật của hệ thống
Sustainability	tính bền vững của hệ thống

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

1.1 Đặt vấn đề

Trong bối cảnh công nghệ số phát triển mạnh mẽ, âm nhạc không chỉ là phương tiện giải trí mà còn là cầu nối văn hóa, thể hiện cá tính và thúc đẩy sự kết nối cộng đồng. Sự phổ biến của các nền tảng nghe nhạc trực tuyến đã tạo ra một thị trường âm nhạc số sôi động, với hàng triệu người dùng trên toàn cầu tham gia nghe, khám phá và chia sẻ nội dung âm nhạc mỗi ngày. Tuy nhiên, trong bức tranh toàn cảnh đó, vẫn tồn tại một nhu cầu chưa được đáp ứng đầy đủ: một không gian trực tuyến cho phép người dùng, đặc biệt là các nghệ sĩ nghiệp dư và những người yêu thích âm nhạc, tự do sáng tạo, tải lên và chia sẻ các tác phẩm âm nhạc cá nhân mà không bị giới hạn bởi các quy trình kiểm duyệt phức tạp hoặc yêu cầu thương mại hóa.

Xuất phát từ thực tế này, chúng tôi đặt ra bài toán xây dựng một mạng xã hội âm nhạc tập trung vào việc khuyến khích sự sáng tạo cá nhân và kết nối cộng đồng. Hiện nay, nhiều nền tảng âm nhạc lớn tập trung cung cấp nội dung từ các nghệ sĩ chuyên nghiệp hoặc các hãng thu âm, nên những người yêu nhạc như chúng tôi hoặc các bạn trẻ muốn chia sẻ sáng tác của mình thường gặp khó khăn. Một số nền tảng cho phép tải lên nhạc nhưng quy trình kiểm duyệt lại phức tạp, khiến việc chia sẻ trở nên hạn chế. Trong khi đó, các mạng xã hội phổ thông tuy có thể chia sẻ âm nhạc nhưng không được thiết kế chuyên biệt để tạo ra một cộng đồng yêu nhạc thực sự, làm cho trải nghiệm của người dùng không được trọn vẹn.

Nếu bài toán này được giải quyết, chúng tôi tin rằng nó sẽ mang lại nhiều giá trị. Đối với những người yêu nhạc, họ sẽ có một không gian để tự do thể hiện cá tính qua các bài hát tự sáng tác, nhận phản hồi từ cộng đồng và khám phá những tác phẩm mới từ bạn bè hoặc những người có cùng sở thích. Đối với cộng đồng, nền tảng này sẽ tạo cơ hội để mọi người giao lưu, học hỏi và cùng nhau phát triển đam mê âm nhạc. Hơn nữa, chúng tôi nghĩ nền tảng này còn có thể được áp dụng trong các lĩnh vực khác, như hỗ trợ giảng dạy âm nhạc ở trường học, quảng bá các thể loại nhạc truyền thống của Việt Nam, hoặc thậm chí tích hợp vào các ứng dụng giải trí để mang đến nội dung độc đáo hơn.

Động lực để chúng tôi thực hiện đồ án này xuất phát từ mong muốn được tự tay xây dựng một sản phẩm công nghệ của riêng mình về âm nhạc. Trong bối cảnh các doanh nghiệp công nghệ tại Việt Nam ngày càng chú trọng đến việc làm chủ công nghệ và phát triển các nền tảng độc lập, tôi thấy đây là cơ hội để học hỏi và áp dụng những kiến thức đã được học vào thực tế. Một mạng xã hội âm nhạc do chúng tôi thiết kế và xây dựng không chỉ giúp đáp ứng nhu cầu của người dùng mà còn là

nền tảng để tôi khám phá tiềm năng của công nghệ trong việc tạo ra những giá trị mới cho cộng đồng.

Chúng tôi tin rằng bài toán xây dựng mạng xã hội âm nhạc là một thách thức đầy ý nghĩa, không chỉ về mặt công nghệ mà còn về khả năng kết nối con người qua âm nhạc. Với quy mô tiềm năng lớn, khi hàng triệu người dùng trên khắp thế giới đang tìm kiếm một không gian để thể hiện bản thân, chúng tôi hy vọng đồ án này sẽ là bước khởi đầu để tạo ra một sản phẩm có giá trị, góp phần thúc đẩy sự sáng tạo và lan tỏa đam mê âm nhạc.

1.2 Mục tiêu và phạm vi đề tài

1.2.1 Tổng quan về các sản phẩm hiện tại và nhu cầu người dùng

Chúng tôi nhận thấy rằng các nền tảng nghe nhạc trực tuyến hiện nay, như Spotify, Apple Music, SoundCloud, và YouTube, đã đạt được những thành tựu ấn tượng trong việc mang âm nhạc đến với hàng triệu người dùng trên toàn thế giới. Spotify và Apple Music nổi bật với thư viện nhạc bản quyền phong phú, giao diện thân thiện, và các thuật toán gợi ý nhạc thông minh dựa trên sở thích người dùng. SoundCloud cung cấp không gian cho các nghệ sĩ độc lập chia sẻ nhạc, thu hút nhiều người sáng tạo nội dung âm nhạc. YouTube, với tính linh hoạt, cho phép đăng tải video âm nhạc và thu hút lượng lớn người xem. Ngoài ra, các mạng xã hội như TikTok và Instagram đã tạo nên những xu hướng âm nhạc lan tỏa mạnh mẽ thông qua các video ngắn tích hợp nhạc.

Tuy nhiên, nhu cầu của người dùng, đặc biệt là giới trẻ và các nghệ sĩ nghiệp dư, không chỉ dừng lại ở việc nghe nhạc. Tại Việt Nam, chúng tôi quan sát thấy nhiều bạn trẻ đam mê sáng tác nhạc, từ các bài hát tự sáng tác đến bản cover, và mong muốn có một không gian để chia sẻ những tác phẩm này với cộng đồng. Người dùng không chỉ muốn thể hiện cá tính qua âm nhạc mà còn kỳ vọng được giao lưu, nhận phản hồi, và kết nối với những người có chung sở thích. Nhu cầu về một nền tảng chuyên biệt, nơi người yêu nhạc có thể vừa sáng tạo vừa xây dựng cộng đồng, đang ngày càng trở nên rõ ràng.

1.2.2 So sánh và đánh giá tổng quan

Các nền tảng như Spotify và Apple Music được thiết kế chuyên nghiệp, tập trung vào việc cung cấp nội dung âm nhạc chất lượng cao từ các nghệ sĩ nổi tiếng, với trải nghiệm người dùng mượt mà và khả năng cá nhân hóa mạnh mẽ. SoundCloud tạo cơ hội cho các nghệ sĩ độc lập, nhưng quy trình kiểm duyệt nội dung đôi khi đòi hỏi các tiêu chuẩn nhất định để đảm bảo chất lượng. YouTube là một nền tảng đa năng, hỗ trợ cả video và âm thanh, nhưng thiếu các tính năng chuyên biệt như trình phát nhạc tích hợp hay quản lý danh sách phát hiệu quả. TikTok và Instagram

rất thành công trong việc lan tỏa âm nhạc qua các video ngắn, nhưng không được tối ưu cho việc xây dựng một cộng đồng yêu nhạc lâu dài.

Những nền tảng này đều mang lại giá trị lớn và phục vụ tốt các mục tiêu riêng của mình. Tuy nhiên, chúng được thiết kế để đáp ứng các nhóm người dùng khác nhau, nên chưa hoàn toàn phù hợp với những người yêu nhạc muốn tự do sáng tạo và tương tác sâu sắc hơn. Ví dụ, việc tải lên nhạc trên một số nền tảng thường bị giới hạn bởi quy trình kiểm duyệt hoặc yêu cầu kỹ thuật, trong khi các mạng xã hội phổ thông thiếu các tính năng như tìm kiếm nhạc chi tiết, gợi ý dựa trên mối quan hệ xã hội, hoặc không gian để thảo luận về âm nhạc.

1.2.3 Hạn chế hiện tại

Dựa trên phân tích trên, chúng tôi khái quát một số hạn chế của các nền tảng hiện tại:

- Thiếu một nền tảng chuyên biệt cho phép người dùng tự do tải lên và chia sẻ nhạc cá nhân mà không bị ràng buộc bởi các quy trình kiểm duyệt phức tạp.
- Chưa có không gian để xây dựng một cộng đồng yêu nhạc, nơi người dùng có thể giao lưu, theo dõi, và ủng hộ các nghệ sĩ mới, cũng như khám phá nhạc dựa trên sở thích và mối quan hệ xã hội.

Những hạn chế này khiến nhiều người yêu nhạc, đặc biệt là các bạn trẻ tại Việt Nam, chưa tìm được một không gian lý tưởng để vừa thể hiện bản thân qua âm nhạc vừa kết nối với cộng đồng.

1.2.4 Mục tiêu của đề tài

Nhằm khắc phục các hạn chế trên, chúng tôi hướng tới xây dựng một mạng xã hội âm nhạc tập trung vào việc khuyến khích sáng tạo cá nhân và kết nối cộng đồng yêu nhạc. Mục tiêu cụ thể của đề tài bao gồm:

- Phát triển một nền tảng cho phép người dùng tự do tải lên và chia sẻ nhạc cá nhân, tạo cơ hội để các nghệ sĩ nghiệp dư thể hiện tài năng và nhận phản hồi từ cộng đồng.
- Xây dựng một cộng đồng yêu nhạc, nơi người dùng có thể giao lưu, khám phá các tác phẩm mới, và tương tác thông qua việc chia sẻ bài hát, danh sách phát, hoặc bình luận.
- Cung cấp các chức năng chính của phần mềm, bao gồm:

  Nghe nhạc với trình phát tích hợp đầy đủ điều khiển (play, pause, next, volume).

  Tìm kiếm nhạc theo từ khóa, tác giả, hoặc thể loại.

Gợi ý nhạc dựa trên sở thích cá nhân, lịch sử nghe, và người dùng đang theo dõi.

Chia sẻ bài hát hoặc danh sách phát với cộng đồng.

- Mang đến một nền tảng thân thiện, dễ sử dụng, khuyến khích sự sáng tạo và tạo nên trải nghiệm âm nhạc mới mẻ cho người dùng.

Đột phá mà chúng tôi mong muốn đạt được là tạo ra một nền tảng không chỉ là nơi nghe nhạc mà còn là một cộng đồng sống động, nơi mỗi người dùng đều có thể trở thành một nghệ sĩ, tự do thể hiện cá tính và kết nối với những người có chung đam mê âm nhạc.

1.2.5 Phạm vi của đề tài

Trong phạm vi đồ án này, chúng tôi tập trung phát triển một nền tảng mạng xã hội âm nhạc dưới dạng ứng dụng web, hỗ trợ truy cập trên máy tính và laptop. Các chức năng chính đã được liệt kê trong mục tiêu, tập trung vào việc hỗ trợ sáng tạo cá nhân và tương tác cộng đồng. Để làm phong phú nội dung, chúng tôi dự kiến sử dụng dữ liệu âm nhạc bổ sung từ một nguồn bên thứ ba, nhưng trọng tâm vẫn là các tính năng cốt lõi của nền tảng.

Do giới hạn về thời gian và nguồn lực, đồ án chưa bao gồm phát triển ứng dụng di động hoặc các tính năng nâng cao như livestream âm nhạc, phân tích nội dung nhạc, gợi ý nhạc bằng trí tuệ nhân tạo, hoặc tích hợp các công cụ chỉnh sửa âm thanh. Phạm vi đề tài cũng tập trung vào việc đảm bảo nền tảng hoạt động ổn định, dễ sử dụng, và đáp ứng tốt các nhu cầu cơ bản của người dùng yêu nhạc.

1.3 Định hướng giải pháp

Để giải quyết bài toán xây dựng một mạng xã hội âm nhạc khuyến khích sáng tạo cá nhân và kết nối cộng đồng, chúng tôi đề xuất định hướng giải pháp như sau:

(i) **Định hướng công nghệ**: Chúng tôi định hướng sử dụng phương pháp thiết kế theo mô hình Traditional N-Layer Architecture (Layered Architecture) cho backend, kết hợp với các công nghệ Spring Boot cho phía server, ReactJS cho giao diện người dùng, và MySQL làm hệ quản trị cơ sở dữ liệu. Ngoài ra, chúng tôi sẽ tích hợp Jamendo API để bổ sung dữ liệu âm nhạc. Lý do lựa chọn: Traditional N-Layer Architecture giúp tổ chức backend rõ ràng và dễ mở rộng, Spring Boot và ReactJS hỗ trợ phát triển nhanh chóng và tạo trải nghiệm mượt mà, trong khi MySQL và Jamendo API đảm bảo quản lý dữ liệu hiệu quả.

(ii) **Mô tả ngắn gọn giải pháp**: Giải pháp của chúng tôi là phát triển một ứng dụng web cho phép người dùng tự do tải lên nhạc cá nhân, nghe nhạc, tìm

kiếm, nhận gợi ý dựa trên sở thích, lịch sử nghe nhạc, người dùng theo dõi và chia sẻ nội dung với cộng đồng. Nền tảng này sẽ có giao diện thân thiện, tích hợp trình phát nhạc tiện lợi và hỗ trợ tương tác xã hội giữa người dùng.

(iii) **Đóng góp chính và kết quả mong đợi**: Đóng góp chính của đồ án là xây dựng một mạng xã hội âm nhạc đề cao tính sáng tạo cá nhân, tạo ra một cộng đồng yêu nhạc sống động và dễ dàng mở rộng nhờ áp dụng Traditional N-Layer Architecture. Kết quả mong đợi là một nền tảng hoạt động ổn định, thân thiện với người dùng, đáp ứng các chức năng chính và mang lại trải nghiệm mới mẻ cho người yêu nhạc.

1.4 Bố cục đồ án

Phần còn lại của báo cáo đồ án tốt nghiệp này được tổ chức như sau:

Chương 2 trình bày quá trình khảo sát hiện trạng các nền tảng âm nhạc hiện nay và phân tích chi tiết các yêu cầu của hệ thống mạng xã hội âm nhạc mà chúng tôi hướng tới phát triển. Nội dung chương bao gồm việc đánh giá các nhu cầu thực tế của người dùng, đặc biệt là các nghệ sĩ nghiệp dư và cộng đồng yêu nhạc, cùng với việc xác định các yêu cầu chức năng như tải lên nhạc, tìm kiếm, gợi ý, và chia sẻ nội dung, cũng như các yêu cầu phi chức năng như tính dễ sử dụng và hiệu năng hệ thống. Các biểu đồ mô hình hóa như use case và đặc tả yêu cầu được trình bày để làm rõ phạm vi và chức năng của nền tảng.

Chương 3, chúng tôi giới thiệu tổng quan về các công nghệ và phương pháp được sử dụng để phát triển hệ thống mạng xã hội âm nhạc. Chương này mô tả các công nghệ, phương pháp chính như Traditional N-Layer Architecture để tổ chức backend, Spring Boot cho phía server, ReactJS cho giao diện người dùng, MySQL làm hệ quản trị cơ sở dữ liệu, và Jamendo API để bổ sung dữ liệu âm nhạc. Mỗi công nghệ được trình bày ngắn gọn về đặc điểm và lý do lựa chọn, nhằm làm rõ sự phù hợp của chúng với mục tiêu phát triển một nền tảng thân thiện, hiệu quả và dễ mở rộng.

Chương 4 tập trung vào quá trình thiết kế, triển khai và đánh giá hệ thống mạng xã hội âm nhạc. Nội dung chương bao gồm thiết kế kiến trúc hệ thống theo mô hình client-server, thiết kế giao diện người dùng với các màn hình chính như trang chủ, trình phát nhạc, và màn hình tải lên nhạc, cũng như thiết kế cơ sở dữ liệu để quản lý thông tin người dùng, bài hát, và danh sách phát. Quá trình triển khai hệ thống trên môi trường thực tế và các kịch bản kiểm thử để đảm bảo chất lượng cũng được trình bày chi tiết, cùng với các kết quả đánh giá hiệu năng và trải nghiệm người dùng.

Chương 5 trình bày các kết luận của đồ án, là một nền tảng mạng xã hội âm nhạc đề cao tính sáng tạo cá nhân, cho phép người dùng tự do tải lên, chia sẻ và khám phá nhạc trong một cộng đồng yêu nhạc sống động. Nền tảng được phát triển dựa trên phương pháp Traditional N-Layer Architecture, đảm bảo tính rõ ràng trong tổ chức backend và khả năng mở rộng trong tương lai. Chương này cũng tổng kết các kết quả đạt được và đề xuất các hướng phát triển tiếp theo, như tích hợp trí tuệ nhân tạo để gợi ý nhạc hoặc mở rộng sang ứng dụng di động.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

Chương 1 đã trình bày tổng quan về bài toán xây dựng một mạng xã hội âm nhạc, nhấn mạnh nhu cầu kết nối cộng đồng yêu nhạc và khuyến khích sáng tạo cá nhân. Chương này sẽ tập trung vào việc khảo sát hiện trạng các nền tảng âm nhạc hiện nay và phân tích chi tiết các yêu cầu của hệ thống mà chúng tôi hướng tới phát triển. Việc khảo sát hiện trạng giúp chúng tôi hiểu rõ bối cảnh thực tế, từ đó xác định các khoảng trống cần giải quyết. Phân tích yêu cầu sẽ làm rõ các chức năng và đặc điểm phi chức năng của hệ thống, đảm bảo nền tảng đáp ứng nhu cầu của người dùng. Nội dung chương được tổ chức thành các phần chính: khảo sát hiện trạng, phân tích các tác nhân và yêu cầu chức năng, yêu cầu phi chức năng, và mô hình hóa yêu cầu thông qua các biểu đồ use case và đặc tả.

2.1 Khảo sát hiện trạng

Để định hình rõ bối cảnh phát triển mạng xã hội âm nhạc, chúng tôi đã tiến hành khảo sát các nền tảng âm nhạc trực tuyến phổ biến hiện nay, bao gồm Spotify, Apple Music, SoundCloud, và Zing MP3, cùng với nhu cầu thực tế của người dùng, đặc biệt tại Việt Nam. Những nền tảng này đều đã đạt được những thành tựu nổi bật, mang âm nhạc đến với hàng triệu người dùng trên toàn thế giới và tạo ra các trải nghiệm chất lượng cao, phục vụ tốt các mục tiêu riêng của từng hệ thống [1].

Spotify và Apple Music là hai nền tảng nghe nhạc hàng đầu trên thế giới, nổi bật với thư viện nhạc bản quyền phong phú, giao diện thân thiện, và khả năng cá nhân hóa thông qua các thuật toán gợi ý nhạc dựa trên lịch sử nghe của người dùng [2]. Spotify cung cấp trải nghiệm nghe nhạc mượt mà với các danh sách phát được tạo tự động, trong khi Apple Music tích hợp chặt chẽ với hệ sinh thái Apple, hỗ trợ chất lượng âm thanh cao và nội dung độc quyền từ các nghệ sĩ nổi tiếng. Cả hai nền tảng đều tập trung vào việc cung cấp nhạc chuyên nghiệp từ các hãng thu âm lớn, đáp ứng nhu cầu nghe nhạc của người dùng phổ thông.

SoundCloud, một nền tảng quan trọng cho các nghệ sĩ độc lập, cho phép người dùng tải lên và chia sẻ nhạc, thu hút hơn 40 triệu nhà sáng tạo nội dung âm nhạc trên toàn cầu [3], [4]. Với giao diện đơn giản và các tính năng hỗ trợ chia sẻ như bình luận, thích, và repost, SoundCloud tạo cơ hội để các nghệ sĩ nghiệp dư tiếp cận khán giả mà không cần thông qua các hãng thu âm lớn [4], [5]. Tuy nhiên, việc tải lên nhạc trên SoundCloud thường đi kèm với các tiêu chuẩn nhất định, chẳng hạn yêu cầu định dạng âm thanh chất lượng cao như WAV hoặc FLAC và quy định nghiêm ngặt về bản quyền, điều này có thể tạo rào cản cho một số người dùng mới hoặc những người muốn chia sẻ nhạc một cách tự do hơn [6]–[8].

Zing MP3, nền tảng âm nhạc trực tuyến hàng đầu tại Việt Nam, được ra mắt vào năm 2007 bởi VNG Corporation và hiện có hơn 28,7 triệu người dùng hoạt động hàng tháng tính đến quý 2 năm 2023 [9]. Zing MP3 cung cấp một kho nhạc chất lượng cao với hàng triệu bài hát bản quyền, bao gồm các ca khúc từ Sony và Universal, được sắp xếp theo album, nghệ sĩ, thể loại, và bảng xếp hạng như #zingchart [9], [10]. Nền tảng này nổi bật với các tính năng như hỗ trợ nhạc lossless, tìm kiếm thông minh bằng giọng nói tiếng Việt, và tích hợp Chromecast, Android Auto, mang lại trải nghiệm nghe nhạc đa dạng trên nhiều thiết bị [11]. Zing MP3 đặc biệt phổ biến với thế hệ Y (nhóm người sinh ra từ giữa những năm 1981 đến 1996) tại Việt Nam nhờ vào nội dung nhạc Việt phong phú và các nghệ sĩ nổi tiếng như Sơn Tùng M-TP, Jack, và Mỹ Tâm [12]. Tuy nhiên, hiện nay, có một số người dùng phản ánh rằng ứng dụng Zing MP3 đang cực kì "hút máu" người dùng bằng cách yêu cầu tài khoản VIP để tải nhạc chất lượng cao (320kbps hoặc lossless), hay thậm chí phải yêu cầu tài khoản VIP để thực hiện chức năng cơ bản trong hệ thống là nghe nhạc. Đó có thể là rào cản đối với người dùng phổ thông hiện nay [13].

Về nhu cầu người dùng, chúng tôi nhận thấy rằng tại Việt Nam, giới trẻ và các nghệ sĩ nghiệp dư ngày càng tích cực tham gia sáng tác và chia sẻ âm nhạc. Nhiều bạn trẻ sáng tạo các bài hát tự sáng tác, bản cover, hoặc thử nghiệm các thể loại nhạc độc đáo như indie, rap, hoặc nhạc truyền thống kết hợp hiện đại. Họ mong muốn có một không gian để đăng tải những sản phẩm này, nhận phản hồi từ cộng đồng, và kết nối với những người có chung sở thích. Người dùng yêu nhạc không chỉ muốn nghe các bài hát có sẵn mà còn kỳ vọng khám phá các tác phẩm mới từ các nghệ sĩ nghiệp dư, theo dõi những người sáng tạo mà họ yêu thích, và tham gia vào các cuộc thảo luận về âm nhạc. Theo báo cáo thị trường âm nhạc Việt Nam 2023 của Zing MP3, hơn 28 triệu người nghe nhạc trực tuyến hàng tháng tại Việt Nam, với mỗi ngày có hơn 500 triệu phút nghe nhạc, cho thấy sự phát triển mạnh mẽ của thị trường và nhu cầu lớn về một nền tảng chuyên biệt [14].

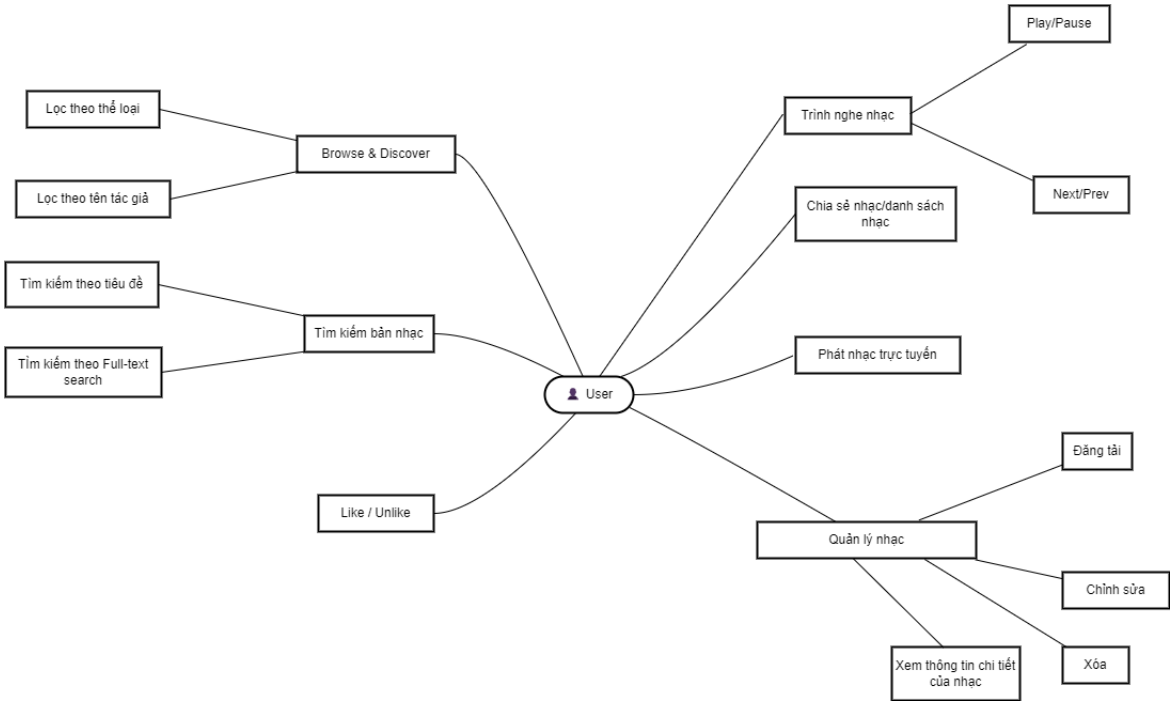
Qua khảo sát, chúng tôi nhận thấy rằng các nền tảng hiện tại, dù mang lại giá trị lớn và phục vụ tốt các mục tiêu riêng, chưa đáp ứng đầy đủ nhu cầu của một bộ phận người dùng muốn tự do sáng tạo và xây dựng cộng đồng yêu nhạc. Spotify và Apple Music tập trung vào nội dung chuyên nghiệp từ các nghệ sĩ nổi tiếng, trong khi SoundCloud, dù hỗ trợ nghệ sĩ độc lập, vẫn có các tiêu chuẩn kiểm duyệt nhất định. Zing MP3 cung cấp trải nghiệm nghe nhạc phong phú và nội dung địa phương hóa tốt, nhưng thiếu các tính năng hỗ trợ xây dựng cộng đồng yêu nhạc hoặc cho phép người dùng tự do tải lên nhạc mà không bị giới hạn bởi tài khoản VIP hoặc các yêu cầu kỹ thuật. Những khoảng trống này là cơ sở để chúng tôi phát triển một mạng xã hội âm nhạc mới, tập trung vào tính sáng tạo cá nhân và tương

tác cộng đồng, đặc biệt phù hợp với nhu cầu của người dùng tại Việt Nam và trên thế giới.

2.2 Tổng quan chức năng

Ở chương 2 sẽ làm rõ về các yêu cầu chức năng và phi chức năng, đảm bảo hệ thống được phát triển theo đúng mong muốn, nhu cầu của đối tượng người dùng. Đồng thời, chương này cũng sẽ phân tích vào usecase tổng quan và phân tích rõ hơn các usecase phân rã để nắm rõ luồng nghiệp vụ. Việc phân tích đóng vai trò quan trọng trong việc thiết kế, phát triển và kiểm thử hệ thống, đảm bảo mọi khía cạnh đều được xem xét và xử lý. 2.3.

2.2.1 Biểu đồ use case tổng quát

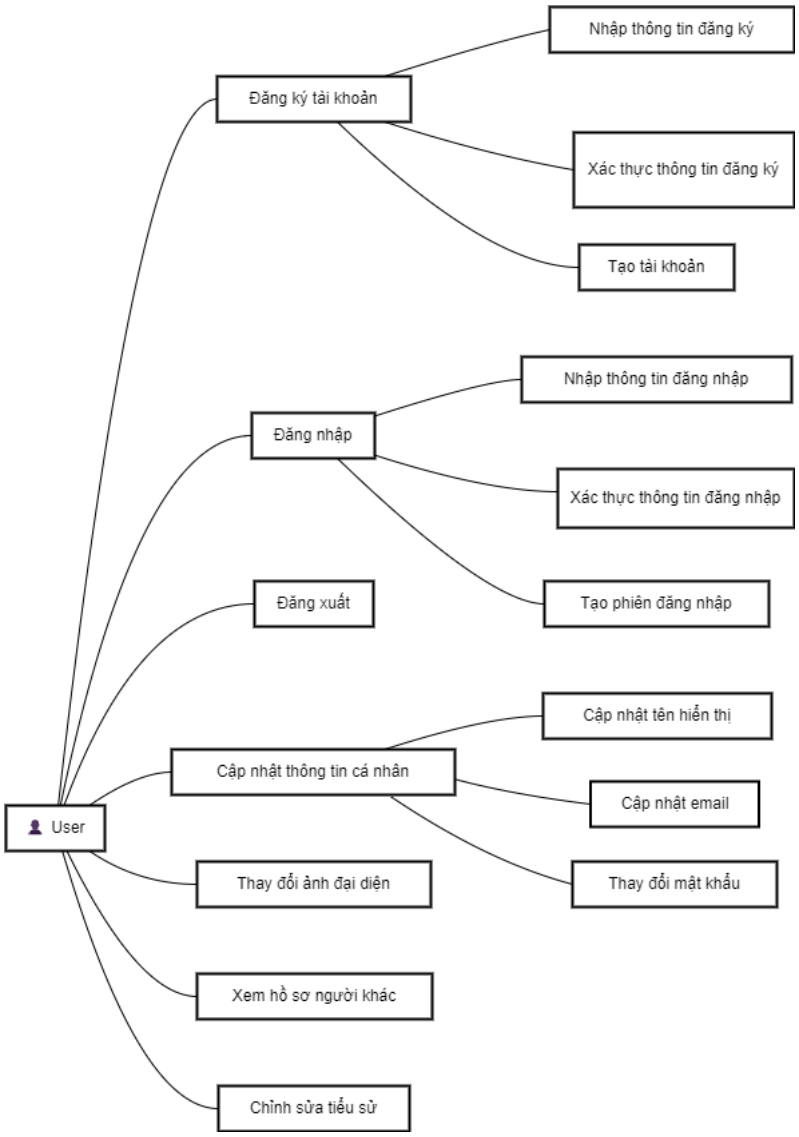


Hình 2.1: Biểu đồ use case tổng quát

Hình 2.1 minh họa sơ đồ usecase tổng quát với các chức năng chính của hệ thống. Hệ thống bao gồm 2 tác nhân chính, mỗi tác nhân có quyền truy cập tới các chức năng khác nhau:

- User: là tác nhân chính của hệ thống, đại diện cho những người dùng cuối đã hoàn tất quá trình đăng ký và đăng nhập thành công vào nền tảng âm nhạc xã hội.
- Admin: là tác nhân có quyền hạn cao nhất trong hệ thống, chịu trách nhiệm quản lý, giám sát và điều hành toàn bộ nền tảng âm nhạc xã hội.

2.2.2 Biểu đồ use case phân rã quản lí người dùng và xác thực



Hình 2.2: Biểu đồ use case phân rã quản lí người dùng và xác thực

Hình vẽ 2.2 là biểu đồ usecase phân rã quản lí người dùng và xác thực có:

- **Tác nhân**

  User: Người dùng muốn truy cập và sử dụng hệ thống

- **Vai trò**

  Quản lý tài khoản cá nhân và thông tin hồ sơ

- **Các use case chính**

  Đăng ký tài khoản: Tạo tài khoản mới với email, username, password

  Đăng nhập: Xác thực và tạo phiên làm việc

  Đăng xuất: Kết thúc phiên làm việc

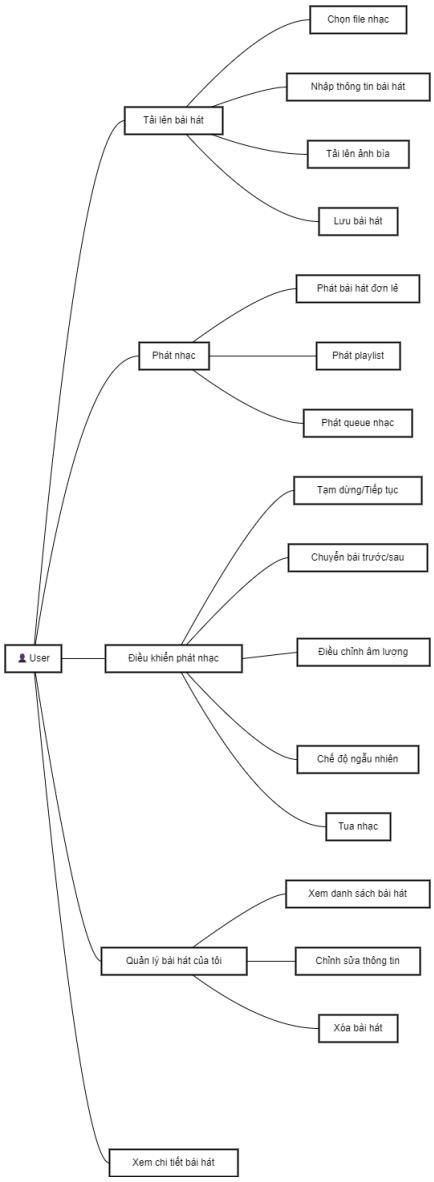
Cập nhật thông tin cá nhân: Thay đổi tên, email, mật khẩu

Thay đổi ảnh đại diện: Upload và cập nhật avatar

Xem hồ sơ người khác: Duyệt profile của user khác

Chỉnh sửa tiểu sử: Cập nhật bio và thông tin mô tả

2.2.3 Biểu đồ use case phân rã quản lý nhạc



Hình 2.3: Biểu đồ use case phân rã quản lí nhạc

Hình vẽ 2.3 là biểu đồ usecase phân rã quản lí nhạc có:

- **Tác nhân**

  User: Người dùng muốn upload, phát và quản lý nhạc

- **Vai trò**

Tạo và quản lý nội dung âm nhạc cá nhân

Sử dụng music player để nghe nhạc

- **Các use case chính**

  Tải lên bài hát: Upload file audio, thêm dữ liệu (title, artist, album, genre), cover image

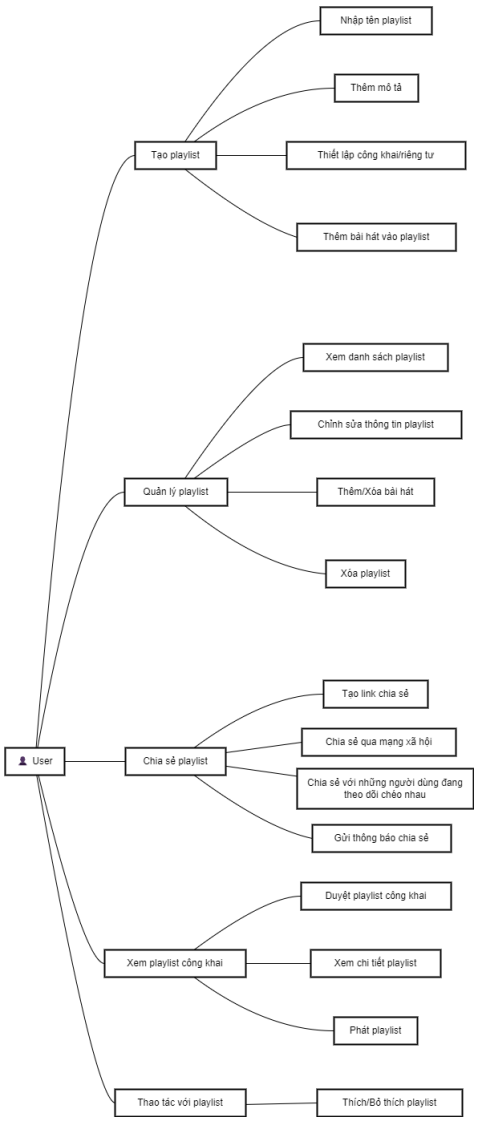
  Phát nhạc: Phát single track, playlist hoặc queue

  Điều khiển phát nhạc: Phát/Dừng, tăng giảm âm lượng trộn bài, tua xuôi, tua ngược

  Quản lý bài hát của tôi: Xem, xóa bài hát đã upload

  Xem chi tiết bài hát: Xem thông tin chi tiết về tác giả, thể loại, thời lượng bài hát

2.2.4 Biểu đồ use case phân rã quản lý danh sách nhạc



Hình 2.4: Biểu đồ use case phân rã quản lí danh sách nhạc

Hình vẽ 2.4 là biểu đồ usecase phân rã quản lí danh sách nhạc có:

- **Tác nhân**

  User: Người dùng muốn tạo và quản lý danh sách nhạc

- **Vai trò**

  Tổ chức nhạc thành các bộ sưu tập

  Chia sẻ playlist với cộng đồng

- **Các use case chính**

  Tạo playlist: Tạo playlist mới với tên, mô tả, cover, thiết lập public/private

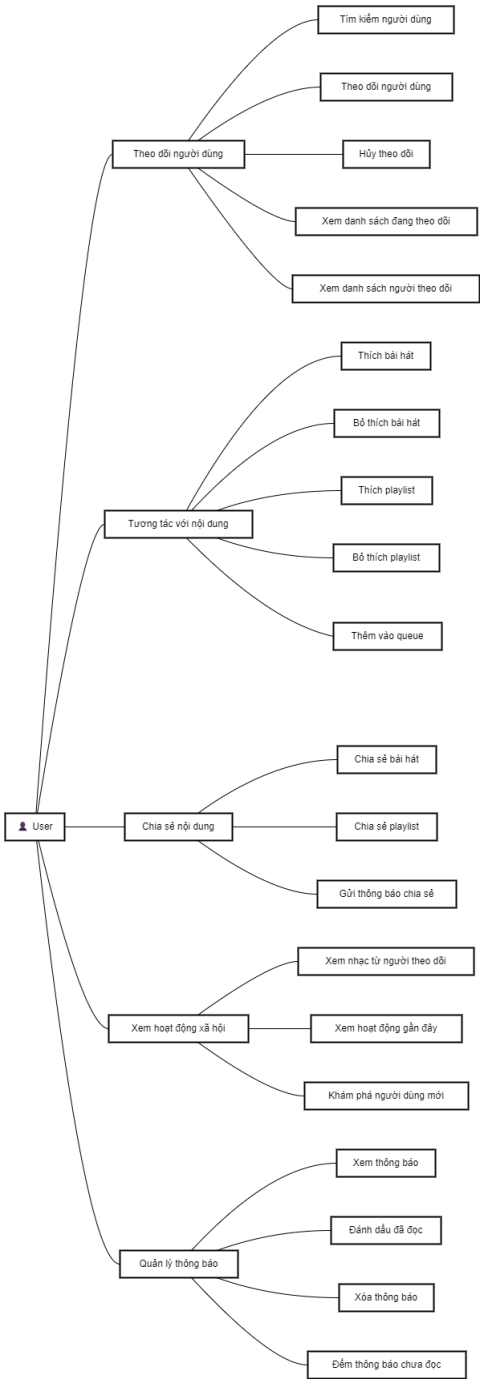
  Quản lý playlist: Chỉnh sửa thông tin, thêm/xóa bài hát trong playlist

  Chia sẻ playlist: Tạo link chia sẻ, gửi thông báo

  Xem playlist công khai: Duyệt và phát playlist của người khác

  Thao tác với playlist: Like/unlike

2.2.5 Biểu đồ use case phân rã Tương Tác Xã Hội



Hình 2.5: Biểu đồ use case phân rã Tương Tác Xã Hội

Hình vẽ 2.5 là biểu đồ usecase phân rã Tương Tác Xã Hội có:

- **Tác nhân**

  User: Người dùng muốn tương tác với cộng đồng

- **Vai trò**

  Xây dựng mạng lưới xã hội trong cộng đồng âm nhạc

Tương tác và chia sẻ nội dung

- **Các use case chính**

  Theo dõi người dùng: Follow/unfollow, xem danh sách following/followers

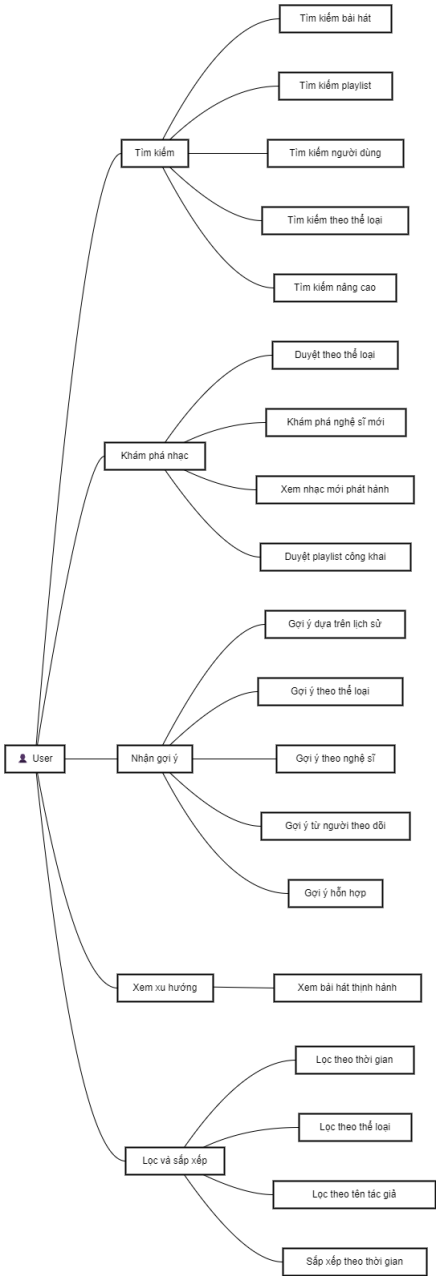
  Tương tác với nội dung: Like/unlike tracks và playlists, thêm vào playlist/queue

  Chia sẻ nội dung: Share tracks/playlists, gửi notification

  Xem hoạt động xã hội: Xem nhạc từ người theo dõi, khám phá user mới

  Quản lý thông báo: Xem, đánh dấu đã đọc, xóa notifications

2.2.6 Biểu đồ use case phân rã Tìm kiếm và Khám phá



Hình 2.6: Biểu đồ use case phân rã Tìm kiếm và Khám phá

Hình vẽ 2.6 là biểu đồ usecase phân rã Tìm kiếm và Khám phá có:

- **Tác nhân**

  User: Người dùng muốn tìm kiếm và khám phá nhạc mới

- **Vai trò**

  Tìm kiếm nội dung trong hệ thống

  Khám phá nhạc và nghệ sĩ mới

- **Các use case chính**

  Tìm kiếm: Search tracks, playlists, album, artist theo từ khóa

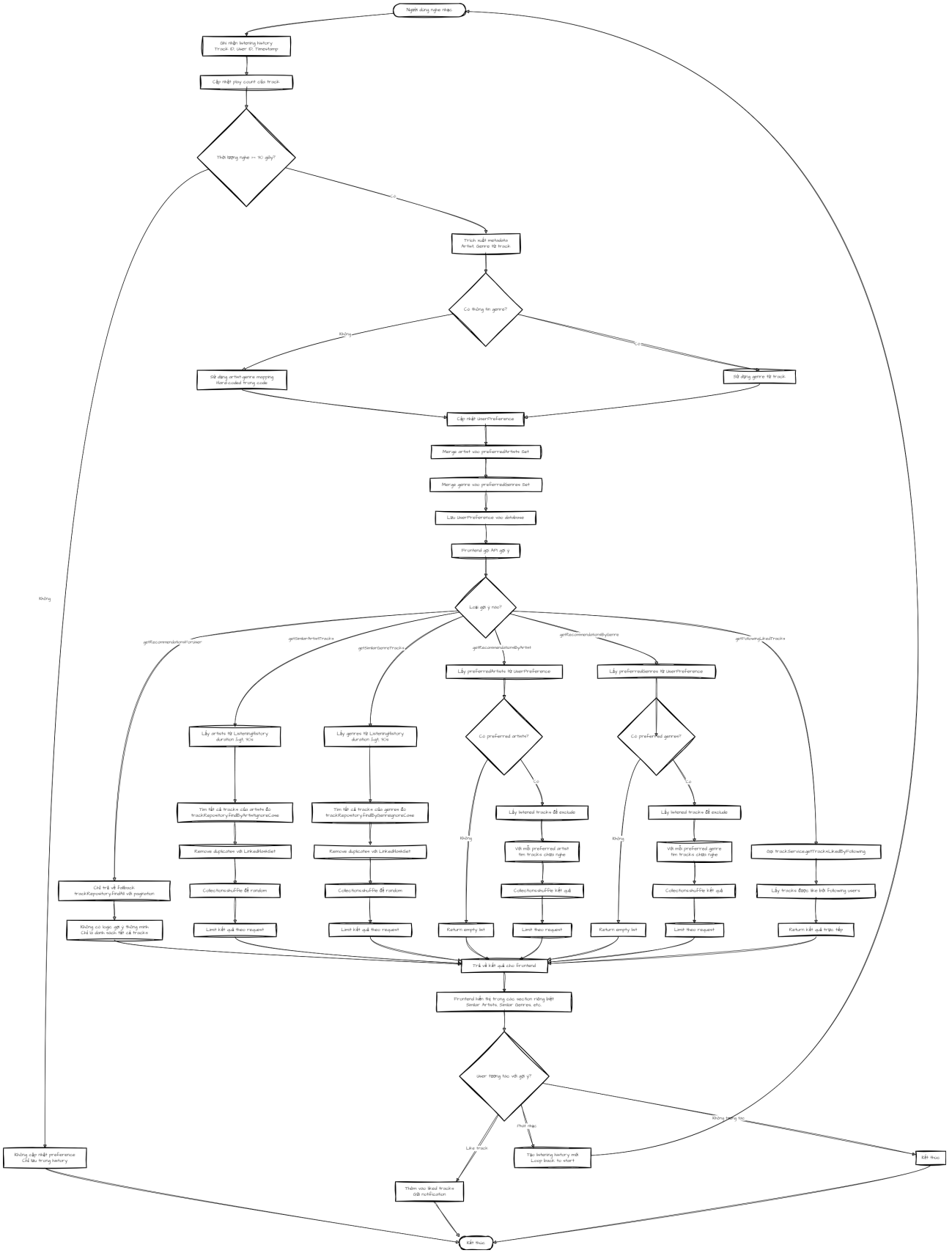
  Khám phá nhạc: Duyệt theo genre, nghệ sĩ, nhạc mới, playlist công khai

  Nhận gợi ý: Recommendations dựa trên lịch sử, thể loại, nghệ sĩ, người đang theo dõi

  Xem xu hướng: Bài hát có nhiều lượt nghe nhất

  Lọc và sắp xếp: Filter theo thời gian/thể loại, tác giả

2.2.7 Quy trình nghiệp vụ



Hình 2.7: Biểu đồ use case phân rã Tìm kiếm và Khám phá

Hình vẽ 2.7 là sơ đồ luồng hoạt động của quy trình xây dựng hệ thống gợi ý cá nhân hóa

Quy trình bắt đầu khi người dùng phát nhạc trên nền tảng. Hệ thống ghi nhận thông tin phiên nghe bao gồm người dùng, bài hát và thời điểm bắt đầu vào cơ sở dữ liệu lịch sử nghe nhạc. Đồng thời, số lượt phát của bài hát được cập nhật để phục vụ tính toán độ phổ biến.

Hệ thống theo dõi thời gian nghe thực tế và áp dụng ngưỡng 30 giây để lọc dữ liệu chất lượng. Chỉ những phiên nghe đạt ngưỡng này mới được sử dụng để xây dựng hồ sơ sở thích. Khi đạt ngưỡng, hệ thống trích xuất thông tin nghệ sĩ và thể loại từ bài hát. Nếu thiếu thông tin thể loại, hệ thống sử dụng bảng ánh xạ nghệ sĩ-thể loại được định nghĩa sẵn. Dữ liệu nghệ sĩ và thể loại được sử dụng để cập nhật hồ sơ sở thích người dùng. Hệ thống lưu trữ danh sách nghệ sĩ yêu thích và thể loại yêu thích dưới dạng tập hợp để tránh trùng lặp. Quá trình cập nhật thực hiện việc hợp nhất - bổ sung thông tin mới vào danh sách hiện có mà không ghi đè dữ liệu cũ.

Hệ thống cung cấp nhiều thuật toán gợi ý hoạt động độc lập khi frontend yêu cầu. Thuật toán gợi ý chính trả về danh sách tất cả bài hát có trong hệ thống với phân trang, không áp dụng logic cá nhân hóa.

Thuật toán gợi ý theo nghệ sĩ tương tự phân tích lịch sử nghe nhạc để tìm các nghệ sĩ mà người dùng đã nghe đủ lâu, sau đó gợi ý tất cả bài hát của những nghệ sĩ đó. Thuật toán gợi ý theo thể loại tương tự hoạt động theo cách tương tự nhưng dựa trên thể loại nhạc.

Thuật toán gợi ý dựa trên sở thích sử dụng hồ sơ sở thích đã được xây dựng. Hệ thống lấy danh sách nghệ sĩ hoặc thể loại yêu thích, loại trừ những bài đã nghe, và gợi ý các bài hát mới từ những nghệ sĩ hoặc thể loại đó.

Thuật toán gợi ý xã hội tìm những bài hát được thích bởi những người mà người dùng đang theo dõi, giúp khám phá nhạc mới thông qua mạng lưới xã hội.

Frontend gọi từng thuật toán gợi ý một cách độc lập và hiển thị kết quả trong các phần riêng biệt trên giao diện. Mỗi phần thể hiện một loại gợi ý cụ thể mà không có sự kết hợp giữa các thuật toán.

Khi người dùng tương tác với các gợi ý, hệ thống ghi nhận phản hồi. Việc phát nhạc tạo ra dữ liệu lịch sử mới và khởi động lại quy trình. Việc thích bài hát được lưu trữ và có thể ảnh hưởng đến gợi ý xã hội cho người khác. Mỗi lần yêu cầu gợi ý đều truy vấn cơ sở dữ liệu trực tiếp và áp dụng xáo trộn ngẫu nhiên để tạo sự đa dạng trong kết quả.

2.3 Đặc tả chức năng

2.3.1 Đặc tả use case Đăng nhập (Login)

Mã usecase	UC01
Tên usecase	Đăng nhập vào hệ thống (Login)
Tác nhân	Người dùng (User)
Mô tả	Người dùng nhập thông tin đăng nhập để xác thực và truy cập vào hệ thống. Nếu thông tin đúng, hệ thống sẽ cho phép truy cập và chuyển đến giao diện chính.
Tiền điều kiện	• Người dùng đã đăng ký tài khoản trên hệ thống. • Hệ thống đang hoạt động và có kết nối internet.
Luồng sự kiện chính	1. Người dùng truy cập vào trang đăng nhập của hệ thống. 2. Người dùng nhập tên đăng nhập và mật khẩu. 3. Hệ thống kiểm tra thông tin đăng nhập với cơ sở dữ liệu. 4. Nếu thông tin hợp lệ, hệ thống cấp quyền truy cập và chuyển hướng người dùng đến trang chính (home).
Luồng sự kiện thay thế	3a. Nếu thông tin đăng nhập không hợp lệ (email hoặc mật khẩu sai), hệ thống hiển thị thông báo lỗi và yêu cầu người dùng nhập lại.
Hậu điều kiện	• Người dùng được cấp quyền truy cập vào hệ thống. • Có thể sử dụng các chức năng dành cho người dùng đã đăng nhập. • Thông tin phiên đăng nhập được lưu trữ (session hoặc token).

Bảng 2.1: Đặc tả use case Đăng nhập vào hệ thống

Bảng 2.1 trình bày đặc tả chi tiết cho use case Đăng nhập trong hệ thống

2.3.2 Đặc tả use case Tải nhạc lên (Upload Track)

Mã usecase	UC02
Tên usecase	Tải nhạc lên (Upload Track)
Tác nhân	Người dùng (User)
Mô tả	Người dùng có thể tải bài hát từ thiết bị của mình lên hệ thống, cung cấp các thông tin liên quan đến bài hát. Nếu hợp lệ, bài hát sẽ được lưu trữ và hiển thị trong thư viện cá nhân.
Tiền điều kiện	• Người dùng đã đăng nhập vào hệ thống. • Người dùng có file nhạc hợp lệ để tải lên.
Luồng sự kiện chính	1. Người dùng đã đăng nhập vào hệ thống. 2. Người dùng truy cập vào trang hoặc phần tải nhạc lên. 3. Người dùng chọn file nhạc và điền các thông tin: tên bài hát, nghệ sĩ, thể loại, mô tả. 4. Hệ thống kiểm tra định dạng file và thông tin nhập vào. 5. Nếu hợp lệ, hệ thống lưu file nhạc vào máy chủ và cập nhật cơ sở dữ liệu. 6. Hệ thống thông báo thành công và bài hát được hiển thị trong thư viện cá nhân.
Luồng sự kiện thay thế	4a. Nếu file không đúng định dạng hoặc vượt quá kích thước, hệ thống hiển thị lỗi và yêu cầu chọn file khác. 4b. Nếu thiếu thông tin như tên bài hát, hệ thống yêu cầu người dùng bổ sung. 5a. Nếu có lỗi trong quá trình tải lên (mất kết nối, lỗi máy chủ), hệ thống thông báo lỗi và cho phép thử lại.
Hậu điều kiện	• Bài hát được lưu trữ trên hệ thống. • Bài hát hiển thị trong thư viện cá nhân của người dùng. • Bài hát có thể được truy cập bởi người dùng khác (tùy theo cài đặt riêng tư).

Bảng 2.2: Đặc tả use case Tải nhạc lên (Upload Track)

Bảng 2.2 trình bày đặc tả chi tiết cho use case *Tải nhạc lên (Upload Track)*. Đây là một chức năng quan trọng trong hệ thống, cho phép người dùng chia sẻ nội dung âm nhạc cá nhân bằng cách tải file nhạc từ thiết bị của họ. Bên cạnh việc kiểm tra định dạng và thông tin hợp lệ, hệ thống còn hỗ trợ xử lý các lỗi phát sinh trong quá trình tải lên, đảm bảo trải nghiệm liền mạch cho người dùng. Chức năng này đóng vai trò nền tảng cho việc xây dựng thư viện nhạc cá nhân cũng như chia sẻ nội dung với cộng đồng người dùng khác.

2.3.3 Đặc tả use case Tạo danh sách nhạc (Create Playlist)

Mã usecase	UC04
Tên usecase	Tạo danh sách phát (Create Playlist)
Tác nhân	Người dùng
Mô tả	Hệ thống cho phép người dùng tạo danh sách phát cá nhân để lưu trữ và quản lý các bài hát yêu thích.
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống.
Luồng sự kiện chính	1. Người dùng đã đăng nhập vào hệ thống. 2. Truy cập phần quản lý danh sách phát hoặc thư viện cá nhân. 3. Chọn tùy chọn "Tạo danh sách phát mới". 4. Nhập tên danh sách phát và mô tả (tùy chọn). 5. Hệ thống lưu danh sách phát vào cơ sở dữ liệu và hiển thị trong thư viện của người dùng. 6. Người dùng thêm bài hát vào danh sách phát từ thư viện hoặc kết quả tìm kiếm.
Luồng sự kiện thay thế	4a. Nếu tên danh sách phát đã tồn tại trong thư viện người dùng, hệ thống yêu cầu đặt tên khác. 6a. Nếu bài hát không có sẵn hoặc người dùng không có quyền truy cập, hệ thống thông báo lỗi khi thêm bài hát.

Bảng 2.3: Còn tiếp ở trang sau

Bảng 2.3: Tiếp tục ở trang trước

Hậu điều kiện	1. Danh sách phát được lưu trữ trong thư viện cá nhân của người dùng. 2. Người dùng có thể chỉnh sửa hoặc chia sẻ danh sách phát.

Bảng 2.3: Đặc tả use case Tạo danh sách phát (Create Playlist)

Bảng 2.3 trình bày đặc tả chi tiết cho use case *Tạo danh sách phát*. Đây là chức năng giúp người dùng cá nhân hóa trải nghiệm âm nhạc của mình bằng cách tạo, quản lý và tổ chức các bài hát yêu thích vào từng danh sách riêng biệt. Việc hỗ trợ thêm bài hát linh hoạt và khả năng xử lý lỗi khi trùng tên hoặc thiếu quyền truy cập giúp đảm bảo tính toàn vẹn và thân thiện của hệ thống.

2.3.4 Đặc tả use case Theo dõi người dùng khác (Follow User)

Bảng 2.4: Đặc tả use case Theo dõi người dùng khác

Mã usecase	UC04
Tên usecase	Theo dõi người dùng khác
Tác nhân	Người dùng đã đăng nhập
Mô tả	Hệ thống cho phép người dùng theo dõi người dùng khác để cập nhật các hoạt động của họ trên bảng tin cá nhân.
Tiền điều kiện	• Người dùng đã đăng nhập vào hệ thống. • Người dùng khác tồn tại trên hệ thống.
Luồng sự kiện chính	1. Người dùng đã đăng nhập vào hệ thống. 2. Người dùng truy cập hồ sơ của người dùng khác. 3. Người dùng nhấn nút "Theo dõi". 4. Hệ thống cập nhật danh sách theo dõi và gửi thông báo đến người được theo dõi (nếu bật). 5. Người dùng thấy hoạt động của người được theo dõi trên bảng tin của mình.

Luồng sự kiện thay thế	3a. Nếu người dùng đã theo dõi từ trước, nút sẽ là "Đang theo dõi", và cho phép hủy theo dõi. 4a. Nếu người được theo dõi cài đặt tài khoản riêng tư, hệ thống yêu cầu xác nhận trước khi theo dõi.
Hậu điều kiện	• Người dùng được thêm vào danh sách theo dõi. • Thông báo được gửi đến người được theo dõi (nếu bật).

Bảng 2.4 trình bày đặc tả chi tiết cho use case *Theo dõi người dùng khác*. Chức năng này giúp kết nối người dùng với nhau, tạo nên một mạng xã hội âm nhạc năng động. Việc hỗ trợ quản lý trạng thái theo dõi và xử lý các tùy chọn quyền riêng tư đảm bảo tính linh hoạt và cá nhân hóa trong trải nghiệm người dùng.

2.3.5 Đặc tả use case gợi ý nhạc (Receive Music Recommendation

Bảng 2.5: Đặc tả use case Nhận gợi ý nhạc

Mã usecase	UC05
Tên usecase	Nhận gợi ý nhạc
Tác nhân	Người dùng đã đăng nhập
Mô tả	Hệ thống gợi ý các bài hát hoặc danh sách phát phù hợp với sở thích và hành vi của người dùng thông qua các thuật toán đề xuất.
Tiền điều kiện	• Người dùng đã đăng nhập vào hệ thống.

Luồng sự kiện chính	1. Người dùng đã đăng nhập vào hệ thống. 2. Người dùng truy cập vào phần gợi ý hoặc khám phá. 3. Hệ thống phân tích lịch sử nghe, sở thích, hành vi người dùng. 4. Hệ thống hiển thị danh sách bài hát hoặc playlist gợi ý. 5. Người dùng có thể nghe hoặc lưu các gợi ý.
Luồng sự kiện thay thế	3a. Nếu người dùng chưa có dữ liệu cá nhân, hệ thống hiển thị gợi ý phổ biến hoặc xu hướng. 5a. Người dùng không thích gợi ý có thể làm mới hoặc điều chỉnh sở thích.
Hậu điều kiện	• Người dùng nhận được gợi ý nhạc phù hợp. • Hành vi tương tác được ghi nhận để cải thiện gợi ý tương lai.

Bảng 2.5 trình bày đặc tả chi tiết cho use case *Nhận gợi ý nhạc*. Đây là chức năng quan trọng trong việc nâng cao trải nghiệm người dùng bằng cách cung cấp các đề xuất cá nhân hóa dựa trên hành vi, sở thích và lịch sử nghe nhạc. Việc hỗ trợ nhiều thuật toán đề xuất và cơ chế phản hồi giúp hệ thống ngày càng trở nên thông minh và phù hợp hơn với từng người dùng.

2.4 Yêu cầu phi chức năng

Trong phần này, hệ thống được thiết kế với các yêu cầu phi chức năng nhằm đảm bảo không chỉ đáp ứng các chức năng cốt lõi mà còn duy trì hiệu quả vận hành, bảo mật, khả năng mở rộng và trải nghiệm người dùng lâu dài. Dưới đây là tổng kết về năm yêu cầu phi chức năng chính của hệ thống *Music Social Platform*.

2.4.1 Yêu cầu về hiệu năng (Performance)

Hệ thống cần đảm bảo thời gian phản hồi nhanh chóng, với mục tiêu xử lý các thao tác cơ bản như đăng nhập hoặc tìm kiếm trong vòng dưới 2 giây, và các thao tác phức tạp như tải nhạc lên không vượt quá 5 giây. Đồng thời, hệ thống được yêu cầu phải có khả năng phục vụ tối thiểu 1.000 người dùng đồng thời mà không làm

giảm hiệu suất xử lý. Ngoài ra, quá trình xử lý dữ liệu gợi ý nhạc cần được hoàn thành trong vòng 3 giây, và hệ thống phải đủ khả năng lưu trữ hàng triệu tệp nhạc mà vẫn duy trì tốc độ truy xuất ổn định.

2.4.2 Yêu cầu về tính dễ dùng (Usability)

Giao diện người dùng phải thân thiện, trực quan và dễ sử dụng cho cả những người không am hiểu công nghệ. Hệ thống cần hỗ trợ ít nhất hai ngôn ngữ là tiếng Việt và tiếng Anh để phục vụ người dùng đa dạng. Bên cạnh đó, đối với các chức năng phức tạp, hệ thống cần cung cấp hướng dẫn cụ thể giúp người dùng dễ tiếp cận và sử dụng. Việc tuân thủ các tiêu chuẩn truy cập như WCAG cũng là yếu tố bắt buộc nhằm hỗ trợ người dùng khuyết tật và nâng cao khả năng tiếp cận của hệ thống.

2.4.3 Yêu cầu về tính dễ bảo trì (Maintainability)

Kiến trúc phần mềm được tổ chức theo mô hình phân lớp rõ ràng, giúp cho việc bảo trì và phát triển về sau trở nên thuận tiện. Mã nguồn cần được tài liệu hóa đầy đủ, đặc biệt là đối với các module chính trong hệ thống. Hệ thống cần có cơ chế ghi log chi tiết các sự kiện và lỗi xảy ra, hỗ trợ công tác kiểm tra và xử lý sự cố. Ngoài ra, các hoạt động nâng cấp và cập nhật hệ thống phải có khả năng thực hiện mà không gây gián đoạn dịch vụ đang vận hành.

2.4.4 Yêu cầu về bảo mật (Security)

Về mặt bảo mật, hệ thống phải áp dụng các phương thức xác thực hiện đại và an toàn như JSON Web Token (JWT). Quan trọng hơn, người dùng cần được cấp quyền kiểm soát nội dung cá nhân, bao gồm việc thiết lập mức độ riêng tư cho các bài hát hoặc danh sách phát đã tải lên.

2.4.5 Yêu cầu về tính bền vững (Sustainability)

Cuối cùng, hệ thống cần được xây dựng hướng đến tính bền vững, nghĩa là có thể duy trì, nâng cấp và vận hành lâu dài mà không đòi hỏi thiết kế lại từ đầu. Điều này được đảm bảo thông qua việc tối ưu hiệu năng bằng các kỹ thuật như tải chậm (lazy loading), cũng như lựa chọn các công nghệ hiện đại, phổ biến và có cộng đồng hỗ trợ mạnh như React.js và Spring Boot. Việc sử dụng các nền tảng này giúp giảm nguy cơ lỗi thời trong vòng 5 đến 10 năm tới và góp phần duy trì sự ổn định, khả năng mở rộng của toàn hệ thống.

CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG

Chương 2 đã khảo sát hiện trạng các nền tảng âm nhạc như Spotify, Apple Music, SoundCloud, và Zing MP3, đồng thời phân tích các yêu cầu của hệ thống mạng xã hội âm nhạc, bao gồm các chức năng như tải lên nhạc, nghe nhạc, tìm kiếm, gợi ý, và chia sẻ nội dung, cùng với các yêu cầu phi chức năng như tính dễ sử dụng, hiệu năng, bảo mật, và khả năng mở rộng. Chương này trình bày các công nghệ được chúng tôi sử dụng để phát triển hệ thống, bao gồm React.js, React DOM, React Router DOM, Material-UI cho giao diện người dùng, Spring Framework và Spring Boot cho backend, và MySQL cho cơ sở dữ liệu. Với mỗi công nghệ, chúng tôi phân tích cách chúng giải quyết các yêu cầu cụ thể từ Chương 2, liệt kê các lựa chọn thay thế, và giải thích lý do lựa chọn. Đặc biệt, chúng tôi mở rộng trình bày về Spring Framework và Spring Boot để làm rõ vai trò cốt lõi của chúng trong backend. Nội dung được tóm tắt ngắn gọn, tập trung vào tính ứng dụng, và tham chiếu đến các nguồn tài liệu đáng tin cậy để đảm bảo tính khoa học.

3.1 Thư viện giao diện React.js

3.1.1 Giới thiệu về React.js

React.js (phiên bản 19.1.0) là thư viện JavaScript mã nguồn mở do Meta phát triển, dùng để xây dựng giao diện người dùng động theo mô hình component-based React [15]–[17]. Trong đồ án, chúng tôi sử dụng React.js để phát triển giao diện cho mạng xã hội âm nhạc, bao gồm trình phát nhạc, form tải lên nhạc, danh sách bài hát, và hồ sơ người dùng, nhằm đáp ứng yêu cầu về tính dễ sử dụng và hiệu năng cao.

React.js cho phép tạo các thành phần tái sử dụng, như `<MusicPlayer>` để điều khiển phát nhạc hoặc `<UploadForm>` để tải lên bài hát [15], [18]. Cơ chế Virtual DOM tối ưu hóa hiệu năng bằng cách chỉ cập nhật phần giao diện thay đổi, đảm bảo thời gian phản hồi nhanh (dưới 1 giây) khi người dùng nhấp nút play hoặc gửi form [15], [19]. Các Hooks như `useState` và `useEffect` giúp quản lý trạng thái và các tác vụ bất đồng bộ, ví dụ: gọi API để lấy danh sách gợi ý bài hát [15], [20]. React 19 giới thiệu async transitions, hỗ trợ hiển thị trạng thái chờ (như loading indicator) khi tải dữ liệu, nâng cao trải nghiệm người dùng [16].

3.1.2 Ưu điểm của React.js

- **Component-based**: Các thành phần được tái sử dụng, giảm thời gian phát triển và dễ bảo trì.
- **Hiệu năng cao**: Virtual DOM giảm thao tác với DOM thực, phù hợp với giao

diện động như trình phát nhạc.

- **Cộng đồng lớn**: Tài liệu phong phú, nhiều thư viện hỗ trợ như React Router DOM và Material-UI (MUI).
- **Linh hoạt**: Dễ dàng tích hợp với backend RESTful như Spring Boot qua `fetch` hoặc `axios`.
- **Hỗ trợ hiện đại**: React 19 cung cấp *async transitions*, giúp xử lý bất đồng bộ mượt mà.

3.1.3 So sánh với các công nghệ khác

- **Vue.js**

  Ưu điểm: Nhẹ hơn, dễ học, hiệu năng tương đương do cũng sử dụng Virtual DOM.

  Nhược điểm: Cộng đồng nhỏ hơn, ít thư viện hỗ trợ, khó mở rộng cho dự án lớn.

  So sánh: React.js phù hợp hơn với đồ án do cộng đồng lớn và dễ tích hợp các thư viện như MUI, phù hợp với SPA quy mô vừa.

- **Angular**

  Ưu điểm: Framework toàn diện, hỗ trợ two-way data binding, phù hợp cho ứng dụng doanh nghiệp lớn.

  Nhược điểm: Cấu trúc phức tạp, bắt buộc dùng TypeScript, thời gian học lâu, bundle size lớn ảnh hưởng hiệu năng.

  So sánh: React.js nhẹ hơn và linh hoạt hơn, phù hợp với mục tiêu đồ án cần phát triển nhanh.

3.1.4 Lựa chọn thay thế

- Vue.js: Nhẹ, dễ học, nhưng cộng đồng và thư viện hỗ trợ hạn chế.
- Angular: Toàn diện, mạnh mẽ nhưng phức tạp, không phù hợp với SPA quy mô vừa.

3.1.5 Lý do lựa chọn

React.js cung cấp hiệu năng cao, hỗ trợ phát triển nhanh nhờ kiến trúc component-based, cộng đồng lớn và thư viện phong phú. Các tính năng mới trong React 19 như async transitions giúp cải thiện trải nghiệm người dùng, đặc biệt phù hợp với ứng dụng giao diện SPA của mạng xã hội âm nhạc trong đồ án này.

3.2 Framework Spring Boot

3.2.1 Giới thiệu về Spring Framework

Spring là một framework mã nguồn mở cho Java, nổi bật với khả năng phát triển các ứng dụng web, REST API, hoặc ứng dụng desktop hiệu quả và có khả năng mở rộng [21].

Spring sử dụng nguyên lý *Dependency Injection* để quản lý vòng đời các bean và đảm bảo sự tách biệt giữa các tầng trong kiến trúc ứng dụng [22].

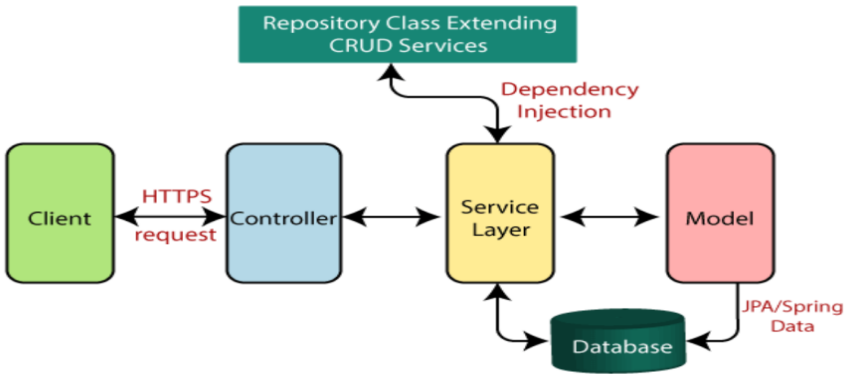
3.2.2 Giới thiệu về Spring Boot

Spring Boot [23] là một phần mở rộng của Spring Framework, giúp đơn giản hóa quá trình cấu hình và khởi tạo ứng dụng Spring. Mục tiêu của Spring Boot là "tự động hóa mọi thứ có thể", giúp lập trình viên tập trung vào logic nghiệp vụ.

Một số tính năng chính:

- **Auto-Configuration**: Tự động cấu hình các thành phần dựa vào thư viện hiện có.
- **Embedded Server**: Cho phép chạy ứng dụng như một chương trình Java đơn lẻ, không cần deploy lên server bên ngoài như Tomcat.
- **Starter Dependencies**: Cung cấp các nhóm thư viện được định nghĩa sẵn để dễ dàng tích hợp như ‘spring-boot-starter-web‘, ‘spring-boot-starter-data-jpa‘

3.2.3 Luồng hoạt động



Hình 3.1: Mô hình luồng hoạt động của Spring Boot

Hình 3.1 mô tả luồng xử lý chính: người dùng gửi request đến Controller, Controller xử lý và giao tiếp với Service. Service gọi đến Repository để truy xuất hoặc cập nhật dữ liệu trong Database, sau đó trả kết quả ngược về cho người dùng.

3.2.4 Ưu điểm của Spring Boot

- **Giảm cấu hình thủ công**: Giúp tiết kiệm thời gian setup ban đầu.

- **Dễ tích hợp và mở rộng**: Dễ dàng thêm các module bảo mật, cơ sở dữ liệu, cloud, caching,...
- **Cộng đồng lớn**: Hỗ trợ từ cộng đồng tốt và cập nhật thường xuyên.

3.2.5 So sánh với các framework backend khác

- **Spring Boot vs Express.js (Node.js)**: Express.js nhẹ hơn và đơn giản hơn cho ứng dụng nhỏ hoặc prototyping nhanh. Spring Boot mạnh hơn cho các ứng dụng lớn, yêu cầu quản lý phức tạp và bảo mật cao.
- **Spring Boot vs Django (Python)**: Django có hệ thống ORM tích hợp, thích hợp với sản phẩm MVP. Spring Boot phù hợp cho hệ thống quy mô lớn, cần hiệu năng và cấu hình chi tiết hơn.
- **Lý do chọn Spring Boot**: Phù hợp với hệ sinh thái Java, hỗ trợ REST tốt, dễ mở rộng, có công cụ phát triển mạnh mẽ và dễ tích hợp với các hệ thống khác như PostgreSQL, Kafka,...

3.3 Hệ quản trị cơ sở dữ liệu MySQL

3.3.1 Giới thiệu về MySQL

MySQL (phiên bản 9.0.1) là hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) mã nguồn mở, lưu trữ dữ liệu trong bảng với các quan hệ như người dùng-bài hát hoặc bài hát-danh sách phát [24]. Trong đồ án, chúng tôi sử dụng MySQL để đáp ứng quản lý dữ liệu cho người dùng, bài hát, danh sách phát, lượt thích, và người theo dõi, đồng thời hỗ trợ gợi ý nhạc dựa trên sở thích.

MySQL hỗ trợ truy vấn SQL như `SELECT`, `INSERT`, và `JOIN` [24]. JSON handling với `JSON_TABLE()` và vector store trong MySQL 9.0.1 cho phép lưu trữ metadata phức tạp và tìm kiếm ngữ nghĩa [25]. Cơ chế xác thực SHA-256 và kết nối bảo mật TLS 1.3 đảm bảo an toàn dữ liệu người dùng [26]. Spring Data JPA tích hợp MySQL với Spring Boot, ánh xạ bảng (như `songs`) thành đối tượng Java (như `SongEntity`) [27].

3.3.2 Ưu điểm của MySQL

- **Hiệu năng cao**: Tối ưu cho truy vấn đọc/ghi nhanh, phù hợp với tìm kiếm bài hát và tải lên nhạc (thời gian phản hồi dưới 1 giây).
- **Tích hợp tốt với Spring Boot**: Spring Data JPA giảm mã boilerplate, cho phép truy vấn dễ dàng qua Repository.
- **Phổ biến và cộng đồng lớn**: Tài liệu phong phú, nhiều công cụ hỗ trợ (như MySQL Workbench), dễ triển khai và bảo trì.
- **Hỗ trợ JSON và vector store**: MySQL 9.0.1 xử lý dữ liệu phi cấu trúc và tìm

kiếm ngữ nghĩa, hỗ trợ gợi ý nhạc.

- **Khả năng mở rộng**: Hỗ trợ replication và sharding, phù hợp khi hệ thống phát triển (ví dụ: tăng số người dùng).
- **Miễn phí và mã nguồn mở**: Giảm chi phí triển khai cho đồ án.

3.3.3 So sánh với các hệ quản trị cơ sở dữ liệu khác

- **PostgreSQL**:

  Ưu điểm: Hỗ trợ JSON tốt hơn (`JSONB`), tính năng nâng cao như full-text search, phù hợp cho ứng dụng phức tạp.

  Nhược điểm: Cấu hình phức tạp hơn, hiệu năng thấp hơn MySQL với khối lượng đọc/ghi lớn.

  So sánh: MySQL nhanh hơn cho truy vấn đơn giản (tìm kiếm bài hát), dễ cấu hình hơn, phù hợp với đồ án quy mô vừa.

- **MongoDB**:

  Ưu điểm: Cơ sở dữ liệu NoSQL, linh hoạt với dữ liệu không cấu trúc (như metadata bài hát), dễ mở rộng ngang.

  Nhược điểm: Kém hiệu quả với quan hệ phức tạp (như người dùng-bài hát-danh sách phát).

  So sánh: MySQL phù hợp hơn cho dữ liệu quan hệ trong đồ án, đảm bảo tính nhất quán và hỗ trợ SQL quen thuộc.

- **SQLite**:

  Ưu điểm: Nhẹ, không cần server, phù hợp cho ứng dụng nhỏ hoặc nhúng.

  Nhược điểm: Không hỗ trợ tải đồng thời lớn, thiếu tính năng như vector store hoặc JSON phức tạp.

  So sánh: MySQL vượt trội về hiệu năng và mở rộng, cần thiết cho mạng xã hội âm nhạc với nhiều người dùng đồng thời.

3.3.4 Lựa chọn thay thế

- **PostgreSQL**: Phù hợp ứng dụng phức tạp, nhưng không cần thiết cho đồ án.
- **MongoDB**: Tốt cho dữ liệu không cấu trúc, nhưng không tối ưu cho quan hệ.
- **SQLite**: Nhẹ, nhưng không đáp ứng tải đồng thời và gợi ý nhạc.

3.3.5 Lý do lựa chọn

MySQL cung cấp hiệu năng cao, tích hợp tốt với Spring Boot qua JPA, hỗ trợ JSON/vector store cho gợi ý nhạc, và dễ triển khai. Cộng đồng lớn và tính phổ biến đảm bảo bảo trì dễ dàng, phù hợp với quy mô đồ án.

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

4.1 Kiến trúc hệ thống

4.1.1 Lựa chọn kiến trúc phần mềm

Trong đồ án phát triển hệ thống mạng xã hội âm nhạc, chúng tôi lựa chọn **Traditional N-Layer Architecture** (hay còn gọi là **Layered Architecture**) làm kiến trúc phần mềm chính. Kiến trúc này tổ chức hệ thống thành các tầng độc lập, mỗi tầng đảm nhận một vai trò cụ thể và tương tác với các tầng khác theo cách có tổ chức, đảm bảo tính dễ bảo trì, mở rộng, và tái sử dụng mã nguồn.Kiến trúc N-Layer thường bao gồm các tầng chính như Presentation Layer (giao diện người dùng), Business Logic Layer (xử lý logic nghiệp vụ), Data Access Layer (truy cập dữ liệu), và Database Layer (lưu trữ dữ liệu) [28]. Các tầng được phân tách rõ ràng, với luồng dữ liệu di chuyển từ tầng trên xuống tầng dưới và ngược lại, giúp giảm sự phụ thuộc và tăng tính module hóa.

Traditional N-Layer Architecture phù hợp với đồ án vì các lý do sau:

- **Phù hợp quy mô vừa**: Hệ thống mạng xã hội âm nhạc có các chức năng như tải lên nhạc, tìm kiếm, gợi ý, và chia sẻ nội dung, không quá phức tạp như các hệ thống phân tán lớn, nên không cần kiến trúc phức tạp như Microservices.
- **Tính dễ phát triển và bảo trì**: Phân tách tầng giúp nhóm phát triển làm việc song song (frontend, backend, cơ sở dữ liệu) và dễ dàng thay thế hoặc nâng cấp từng tầng.
- **Hỗ trợ tích hợp công nghệ**: Các công nghệ đã chọn (React.js, Spring Boot, MySQL) ánh xạ tự nhiên vào các tầng, với React.js cho *Presentation Layer*, Spring Boot cho *Business Logic* và *Data Access Layers*, và MySQL cho *Database Layer*.
- **Đáp ứng yêu cầu phi chức năng**: Tính phân tách đảm bảo hiệu năng (qua tối ưu từng tầng), bảo mật (qua kiểm soát truy cập tại *Business Logic Layer*), và mở rộng.

So với các kiến trúc khác, N-Layer có ưu điểm rõ ràng:

- **So với MVC**: MVC (Model-View-Controller) phù hợp cho ứng dụng web nhỏ, nhưng khó mở rộng với hệ thống có nhiều nghiệp vụ phức tạp như gợi ý nhạc. N-Layer cung cấp phân tách rõ ràng hơn, với *Data Access Layer* riêng biệt.
- **So với Microservices**: Microservices hỗ trợ mở rộng lớn, nhưng phức tạp và yêu cầu quản lý dịch vụ phân tán, không cần thiết cho đồ án.

- **So với SOA**: Service-Oriented Architecture (SOA) tập trung vào dịch vụ độc lập, nhưng đòi hỏi tích hợp phức tạp hơn so với N-Layer cho hệ thống quy mô vừa.

Áp dụng kiến trúc N-Layer vào hệ thống mạng xã hội âm nhạc:

Chúng tôi thiết kế hệ thống với bốn tầng chính, ánh xạ các công nghệ và thành phần cụ thể như sau:

a, Presentation Layer

Vai trò: Tầng này chịu trách nhiệm hiển thị giao diện người dùng, xử lý các tương tác như nhấp nút play hoặc gửi form tải lên nhạc, và gửi yêu cầu đến backend.

Công nghệ: React.js v19.1.0, React DOM v19.1.0, React Router DOM v7.6.2, Material-UI v7.1.1.

Thành phần cụ thể:

- *Components*: `MusicPlayer` (trình phát nhạc), `UploadForm` (form tải lên nhạc), `playlist` (danh sách bài hát), `ProfilePage` (hồ sơ người dùng).
- *Routing*: `BrowserRouter` và các tuyến như `/upload`, `/profile/:id`.
- *UI Elements*: `Button`, `Card` từ Material-UI, với style tùy chỉnh qua `ThemeProvider`.

Chức năng: Đáp ứng *tính dễ sử dụng* với giao diện trực quan, *hiệu năng* nhờ Virtual DOM và lazy loading, và *khám phá nội dung* thông qua SEO với server-side rendering (SSR).

b, Business Logic Layer

Vai trò: Tầng này xử lý logic nghiệp vụ, bao gồm xác thực người dùng, xử lý tệp nhạc, tìm kiếm bài hát, và tạo gợi ý dựa trên sở thích.

Công nghệ: Spring Boot v3.2.3, Spring Framework v6.1.3 (Spring MVC, Spring Security, Micrometer)

Thành phần cụ thể:

- *Controllers*: `TrackController` (xử lý API như `POST /api/tracks`, `GET /api/search`).
- *Services*: `TrackService` (xử lý logic tải lên nhạc, tìm kiếm), `RecommendationService` (tạo gợi ý).
- *Security*: Spring Security với JWT để xác thực và phân quyền.

Chức năng: Đáp ứng *bảo mật* (JWT), *hiệu năng* (xử lý nhanh API), *quản lý hệ*

thống (giám sát qua Micrometer), và *khám phá nội dung* (gợi ý nhạc).

c, Data Access Layer

Vai trò: Tầng này truy cập và thao tác dữ liệu, ánh xạ đối tượng Java sang bảng cơ sở dữ liệu, và thực thi truy vấn SQL.

Công nghệ: Spring Boot v3.2.3, Spring Data JPA **spring2025**.

Thành phần cụ thể:

- *Repositories*: `TrackRepository` (truy vấn bảng `tracks`), `UserRepository` (quản lý người dùng).
- *Entities*: `Track` (ánh xạ bảng `tracks`), `User` (bảng `users`).

Chức năng: Đáp ứng *quản lý dữ liệu* và *hiệu năng* với truy vấn tối ưu qua JPA.

d, Database Layer

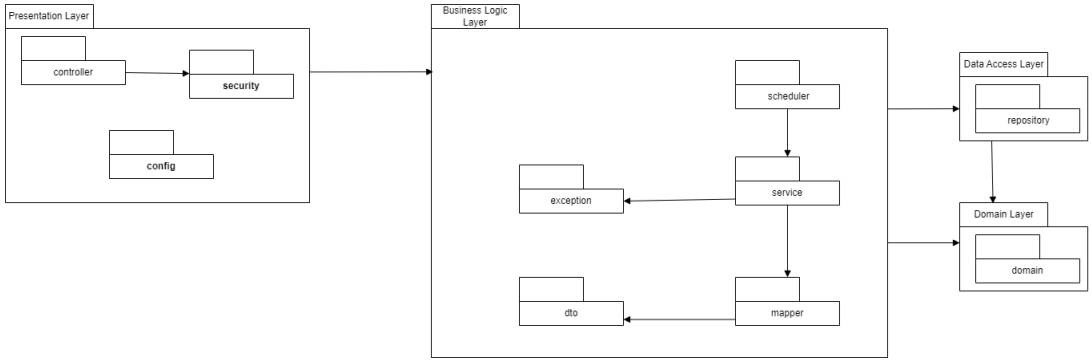
Vai trò: Tầng này lưu trữ dữ liệu quan hệ và hỗ trợ truy vấn phức tạp cho tìm kiếm và gợi ý.

Công nghệ: MySQL v9.0.1

Thành phần cụ thể:

- *Tables*: `tracks` (metadata bài hát), `users` (thông tin người dùng), `playlists` (danh sách nhạc).
- *Features*: JSON handling (`JSON_TABLE`) và vector store.

4.2 Thiết kế tổng quan



Hình 4.1: Biểu đồ gói UML tổng quát

Hình 4.1 minh hoa biểu đồ gói UML tổng quan thể hiện kiến trúc **Traditional N-Layer Architecture** với bốn tầng chính được phân tách rõ ràng, gồm:

- **Presentation Layer** (Tầng trình bày):

  Tầng này chứa ba package chính chịu trách nhiệm xử lý giao tiếp với bên

ngoài:

Package controller: Đóng vai trò là điểm vào chính của hệ thống, chịu trách nhiệm xử lý các HTTP requests từ frontend và trả về responses tương ứng. Controller điều phối luồng hoạt động tổng thể, kết nối giữa giao diện người dùng và logic nghiệp vụ. Trong hệ thống mạng xã hội âm nhạc, controller xử lý các yêu cầu như đăng nhập, tải lên bài hát, tạo playlist, và quản lý thông báo.

Package security: Chịu trách nhiệm xử lý các vấn đề bảo mật như authentication (xác thực) và authorization (phân quyền). Package này chứa các thành phần JWT, security filters, và các cơ chế bảo vệ API endpoints, đảm bảo chỉ những người dùng được ủy quyền mới có thể truy cập các tài nguyên tương ứng.

Package config: Chứa các lớp cấu hình Spring Boot, định nghĩa các beans, database connections, và các thiết lập hệ thống. Đây là nơi tập trung các cấu hình toàn cục của ứng dụng.

- **Business Logic Layer** (Tầng logic nghiệp vụ):
  Tầng này là trung tâm xử lý logic nghiệp vụ với năm package chính:

  Package service: Đóng vai trò cốt lõi trong việc xử lý logic nghiệp vụ phức tạp của hệ thống mạng xã hội âm nhạc. Service điều phối luồng dữ liệu giữa controller và repository, thực hiện các chức năng như gợi ý bài hát, quản lý playlist, xử lý thông báo, và tính toán thống kê nghe nhạc.

  Package mapper: Chịu trách nhiệm chuyển đổi dữ liệu giữa các đối tượng Domain entities và DTOs. Mapper đảm bảo tính tách biệt giữa các tầng bằng cách cung cấp các phương thức ánh xạ, giúp dữ liệu được truyền tải một cách an toàn và nhất quán.

  Package dto: Chứa các Data Transfer Objects được sử dụng để truyền dữ liệu giữa các tầng và với frontend. DTOs giúp kiểm soát thông tin được expose ra bên ngoài và tối ưu hóa việc truyền tải dữ liệu.

  Package exception: Tập trung quản lý các custom exceptions và exception handlers của hệ thống. Package này đảm bảo việc xử lý lỗi được thực hiện một cách nhất quán và cung cấp thông tin lỗi phù hợp cho người dùng.

  Package scheduler: Chứa các tác vụ định kỳ và background jobs như cập nhật thống kê, gửi thông báo theo lịch, hoặc làm sạch dữ liệu tạm thời. Scheduler giúp tự động hóa các quy trình không cần sự can thiệp trực tiếp của người dùng.

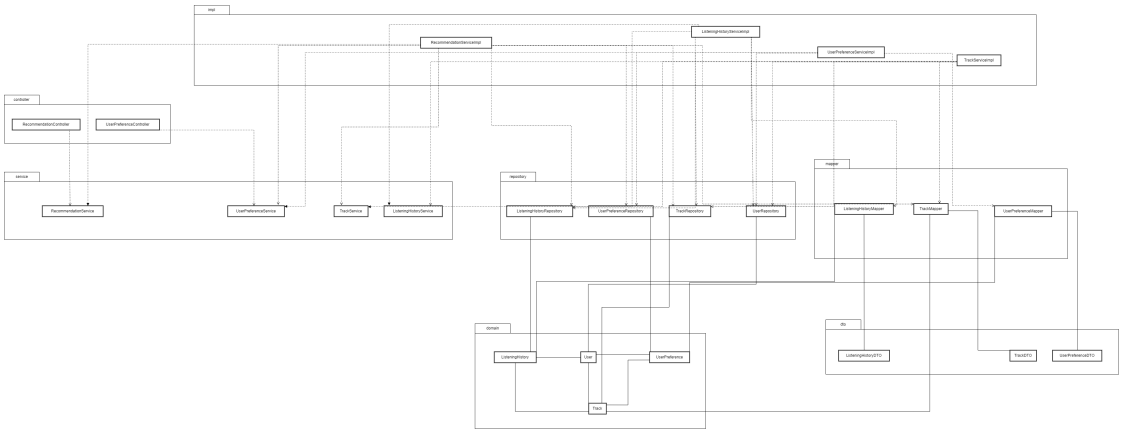
- **Domain Layer** (Tầng miền):
  Package domain: Chứa các thực thể cốt lõi của hệ thống như `User`, `Track`, `Playlist`, `Notification`, `Comment`, và `ListeningHistory`. Đây là trung tâm của toàn bộ kiến trúc, định nghĩa cấu trúc dữ liệu và mối quan hệ giữa các đối tượng nghiệp vụ. Domain entities được ánh xạ trực tiếp với cơ sở dữ liệu thông qua JPA annotations.
- **Data Access Layer** (Tầng truy cập dữ liệu):
  Package repository: Cung cấp lớp trừu tượng để truy cập và thao tác với cơ sở dữ liệu. Repository interfaces định nghĩa các phương thức CRUD và các truy vấn phức tạp, được Spring Data JPA tự động implement. Package này đảm bảo tính độc lập của logic nghiệp vụ khỏi chi tiết cài đặt cơ sở dữ liệu.

Biểu đồ thể hiện rõ ràng việc tuân thủ các nguyên tắc thiết kế quan trọng:

- *Separation of Concerns*: Mỗi tầng có trách nhiệm riêng biệt.
- *Dependency Inversion*: Các tầng trên phụ thuộc vào abstractions của tầng dưới.
- *Single Direction Dependency*: Không có phụ thuộc ngược từ tầng dưới lên tầng trên.
- *Layered Architecture*: Không được phép bỏ qua tầng trong việc truy cập.

Kiến trúc này đảm bảo tính *maintainability*, *scalability* và *testability* cao cho hệ thống mạng xã hội âm nhạc **fowler2002patterns**.

4.2.1 Thiết kế chi tiết gói



Hình 4.2: Thiết kế chi tiết nhóm gói Recommendation System

Hình 4.2 mô tả biểu đồ gói chi tiết cho một nhóm các package thực hiện chức năng gợi ý nhạc

Mục đích thiết kế

Nhóm package này được thiết kế để giải quyết bài toán gợi ý bài hát thông minh dựa trên hành vi nghe nhạc và sở thích của người dùng trong hệ thống mạng xã hội âm nhạc. Mục tiêu là xây dựng một hệ thống có khả năng học hỏi và cải thiện độ chính xác gợi ý theo thời gian thực, đáp ứng nhu cầu cá nhân hóa của người dùng.

Cấu trúc thiết kế

Thiết kế tuân thủ *Traditional N-Layer Architecture* với 6 tầng rõ ràng, được phân tách để đảm bảo tính độc lập và tái sử dụng:

- `Controller`: Xử lý các yêu cầu HTTP cho API gợi ý và quản lý sở thích.
- `Service`: Định nghĩa các giao diện logic nghiệp vụ.
- `Service.impl`: Triển khai logic gợi ý và phân tích sở thích.
- `Repository`: Truy cập dữ liệu từ cơ sở dữ liệu.
- `Domain`: Các thực thể cốt lõi bao gồm `User`, `Track`, `ListeningHistory`, và `UserPreference`.
- `DTO & Mapper`: Chuyển đổi và truyền tải dữ liệu một cách an toàn.

Các mối quan hệ chính

Các mối quan hệ giữa các tầng được thiết kế dựa trên các nguyên tắc thiết kế hướng đối tượng:

- *Realization (Thực thi)*: Các triển khai `Service.impl` thực thi các giao diện `Service`, đảm bảo tuân thủ *Dependency Inversion Principle*.
- *Dependency (Phụ thuộc)*:

  `Controllers` phụ thuộc vào các giao diện `Service` (không phụ thuộc vào các triển khai).

  `Services` phụ thuộc vào `Repositories` và `Mappers` để thực hiện logic nghiệp vụ.

  Các `Service` tương tác với nhau để xử lý các logic phức tạp.

- *Association (Kết hợp)*: `Repositories` quản lý các thực thể `Domain`, trong khi `Mappers` chuyển đổi giữa `Domain` và `DTO`.

Ưu điểm thiết kế

Thiết kế này mang lại nhiều lợi ích quan trọng:

- *Tách biệt rõ ràng*: Mỗi tầng có trách nhiệm cụ thể, giúp dễ dàng bảo trì và phát triển.

- *Linh hoạt*: Có thể thay đổi thuật toán gợi ý mà không ảnh hưởng đến các tầng khác.
- *Tái sử dụng*: Logic phân tích sở thích có thể được sử dụng bởi nhiều `Services`.
- *Kiểm thử*: Dễ dàng mock các dependencies để thực hiện unit testing.

4.3 Thiết kế chi tiết

4.3.1 Thiết kế giao diện

a, Đặc tả thông tin màn hình

Hệ thống mạng xã hội âm nhạc được thiết kế tối ưu cho việc sử dụng trên các thiết bị máy tính để bàn và laptop thông qua trình duyệt web. Ứng dụng hướng đến việc cung cấp trải nghiệm người dùng tốt nhất trên các màn hình có kích thước và độ phân giải phù hợp với việc nghe nhạc, duyệt nội dung và tương tác xã hội.

Độ phân giải màn hình

Ứng dụng được tối ưu hóa cho các độ phân giải màn hình phổ biến:

- **Độ phân giải khuyến nghị:** 1920×1080 pixels (Full HD) - Đây là độ phân giải chuẩn cho việc hiển thị giao diện đầy đủ với tất cả các thành phần được bố trí hợp lý
- **Độ phân giải tối thiểu:** 1366×768 pixels - Đảm bảo ứng dụng vẫn hoạt động tốt trên các laptop cơ bản và màn hình nhỏ hơn
- **Độ phân giải tối đa hỗ trợ:** 2560×1440 pixels (2K) - Tận dụng tối đa không gian hiển thị trên các màn hình cao cấp
- **Tỷ lệ khung hình:** 16:9 và 16:10 - Các tỷ lệ màn hình chuẩn cho desktop và laptop

Kích thước màn hình vật lý

Ứng dụng được thiết kế để hoạt động hiệu quả trên các kích thước màn hình sau:

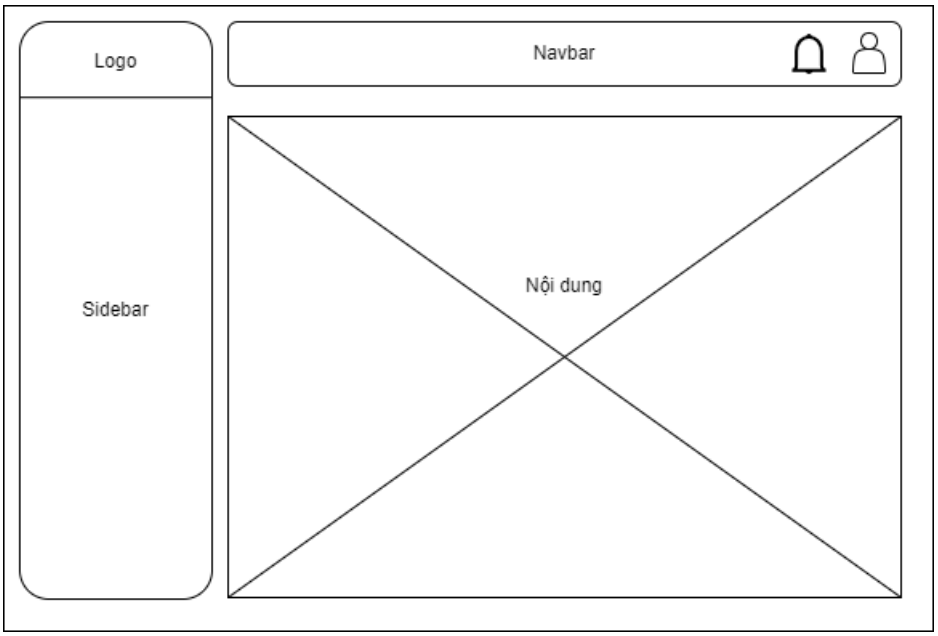
- **Màn hình nhỏ:** 13-15 inch (laptop cá nhân) - Giao diện tự động điều chỉnh để tối ưu không gian hiển thị
- **Màn hình trung bình:** 21-24 inch (desktop monitor phổ biến) - Kích thước lý tưởng cho trải nghiệm đầy đủ
- **Màn hình lớn:** 27-32 inch (widescreen professional) - Tận dụng không gian rộng để hiển thị nhiều nội dung

b, Chuẩn hóa thiết kế giao diện

Hệ thống màu sắc và thành phần Giao diện áp dụng phong cách Dark Theme với bảng màu chuẩn hóa, sử dụng tông màu chính và phụ để tạo sự hài hòa. Các nút bấm được thiết kế đồng nhất với nền và viền màu nổi bật, kết hợp với hiệu ứng hover để tăng tính tương tác. Thông báo trong hệ thống hiển thị dưới dạng snackbar, với màu sắc phân biệt theo trạng thái:

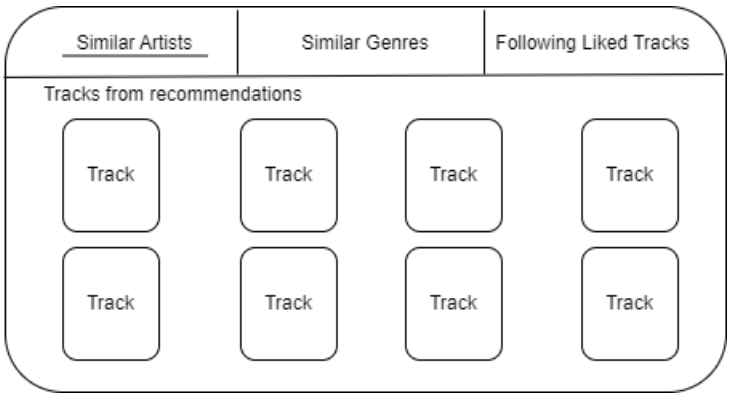
- **Thành công:** Màu xanh lá cây (#4caf50).
- **Cảnh báo:** Màu vàng (#ff9800).
- **Lỗi:** Màu đỏ (#f44336).
- **Thông tin:** Màu xanh dương (#2196f3).

Tất cả thông báo tự động ẩn sau 3 giây để tránh gây cản trở trải nghiệm người dùng.



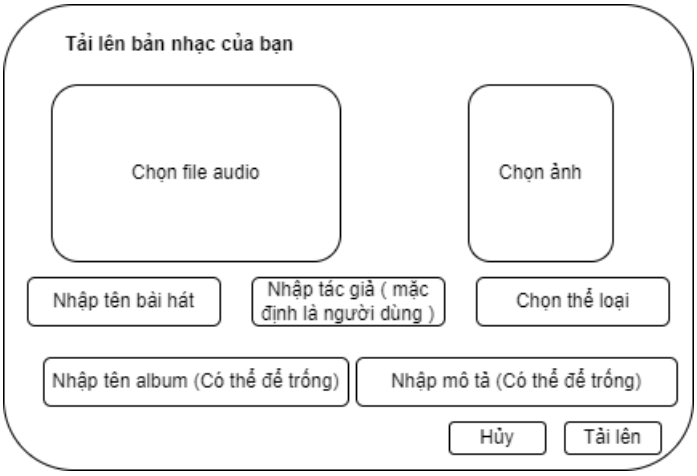
Hình 4.3: Thiết kế giao diện hiển thị nội dung

Hình 4.3 mô tả thiết kế giao diện nội dung. Màn hình giao diện bao gồm thanh Navbar bên trên, chứa nút thông báo của hệ thống và cài đặt người dùng ở bên phải, thanh Navbar ở bên trái màn hình, có thể mở rộng hoặc thu hẹp lại, trong thanh Navbar chứa Logo của trang web ở phía trên cùng của thanh, và dưới là các lựa chọn điều hướng trang web, ở giữa là phần chứa nội dung chính của trang web



Hình 4.4: Thiết kế giao diện hiển thị các bài hát gợi ý của hệ thống

Hình 4.4 mô tả thiết kế giao diện hiển thị các bài hát gợi ý của hệ thống. Trong giao diện này, người dùng có thể chọn 3 lựa chọn gợi ý: Similar Artists - gợi ý theo tác giả, Similar Genres - gợi ý theo thể loại, Following Liked Tracks - gợi ý theo người dùng đang theo dõi, và có thể chọn 1 trong 3 để hệ thống đưa ra gợi ý phù hợp với nhu cầu người dùng.



Hình 4.5: Thiết kế giao diện hiển thị các bài hát gợi ý của hệ thống

Hình 4.5 mô tả thiết kế giao diện tải bản nhạc của cá nhân lên hệ thống. Trong giao diện này, người dùng có thể chọn file audio và tải nhạc lên từ máy cá nhân của người dùng, chọn ảnh - chọn ảnh bìa cho bài hát của mình, và người dùng có thể nhập tên bài hát mình muốn đặt, đặt thể loại, nhập album cho bài hát, nhập mô tả về bát hát, bên cạnh đó, khi nhập tên tác giả, người dùng có 2 lựa chọn, 1 là để trống, thì hệ thống sẽ ghi nhận tác giả là username đăng nhập của người dùng, 2 là người không để trống, và nhập tên khác khi muốn sử dụng tên khác để đăng bài hát của mình lên hệ thống.

4.3.2 Thiết kế lớp

4.4 Thiết kế lớp

4.4.1 Thiết kế chi tiết các lớp chủ đạo

Phần này trình bày thiết kế chi tiết các thuộc tính và phương thức cho 3 lớp chủ đạo quan trọng nhất của hệ thống mạng xã hội âm nhạc.

a, Lớp User

Hình 4.6: Thiết kế lớp User

Hình 4.6 mô tả thiết kế lớp User - lớp đại diện cho thực thể người dùng trong hệ thống, là lớp cốt lõi quản lý toàn bộ thông tin và hành vi của người dùng trong mạng xã hội âm nhạc. Lớp User cung cấp các chức năng để quản lý thông tin cá nhân: Thông tin cơ bản: username, email, password, fullName, bio Hình ảnh đại diện: profileImageUrl Phân quyền: role (USER, ADMIN) Thời gian: createdAt, updatedAt để theo dõi hoạt động, quản lý mối quan hệ xã hội: Following/Followers:

Theo dõi và được theo dõi bởi người dùng khác Mạng lưới xã hội: Tạo kết nối giữa các người dùng trong cộng đồng âm nhạc, quản lý nội dung âm nhạc: Sở hữu tracks: Các bài hát mà người dùng đã upload Sở hữu playlists: Các danh sách phát do người dùng tạo Liked content: Tracks và playlists mà người dùng đã thích, theo dõi hoạt động: Listening history: Lịch sử nghe nhạc chi tiết User preferences: Sở thích âm nhạc cá nhân.

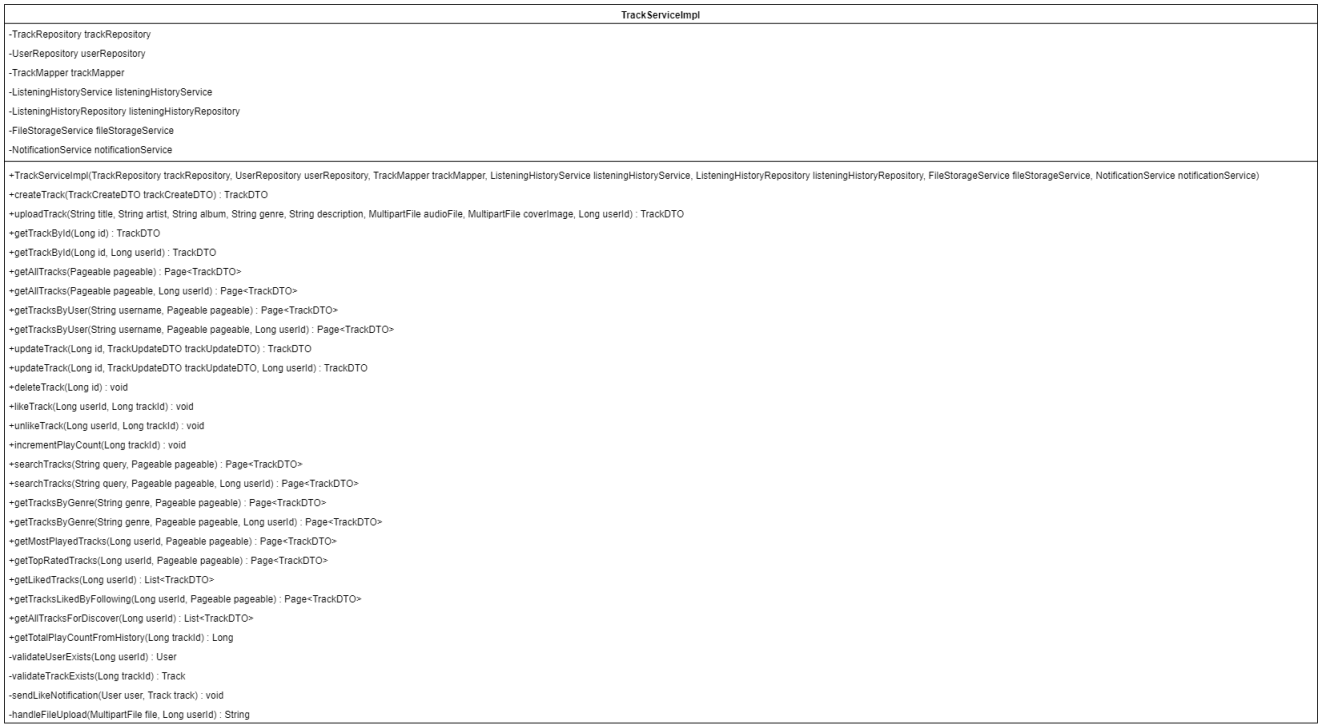
b, Lớp Track

Hình 4.7: Thiết kế lớp Track

Hình 4.7 mô tả thiết kế lớp Track - lớp đại diện cho thực thể bài hát/track âm nhạc, là đơn vị nội dung cốt lõi của toàn bộ hệ thống. Lớp User cung cấp các chức năng để quản lý metadata âm nhạc - Thông tin cơ bản: title, artist, album, genre Thông tin kỹ thuật: duration (thời lượng tính bằng giây) Nguồn gốc: jamendoId (nếu lấy từ API Jamendo), quản lý tài nguyên đa phương tiện: Audio file: audioUrl - đường dẫn đến file âm thanh Cover image: coverImageUrl - ảnh bìa của bài hát, quản lý quan hệ và tương tác: Chủ sở hữu: Liên kết với User đã upload Thuộc

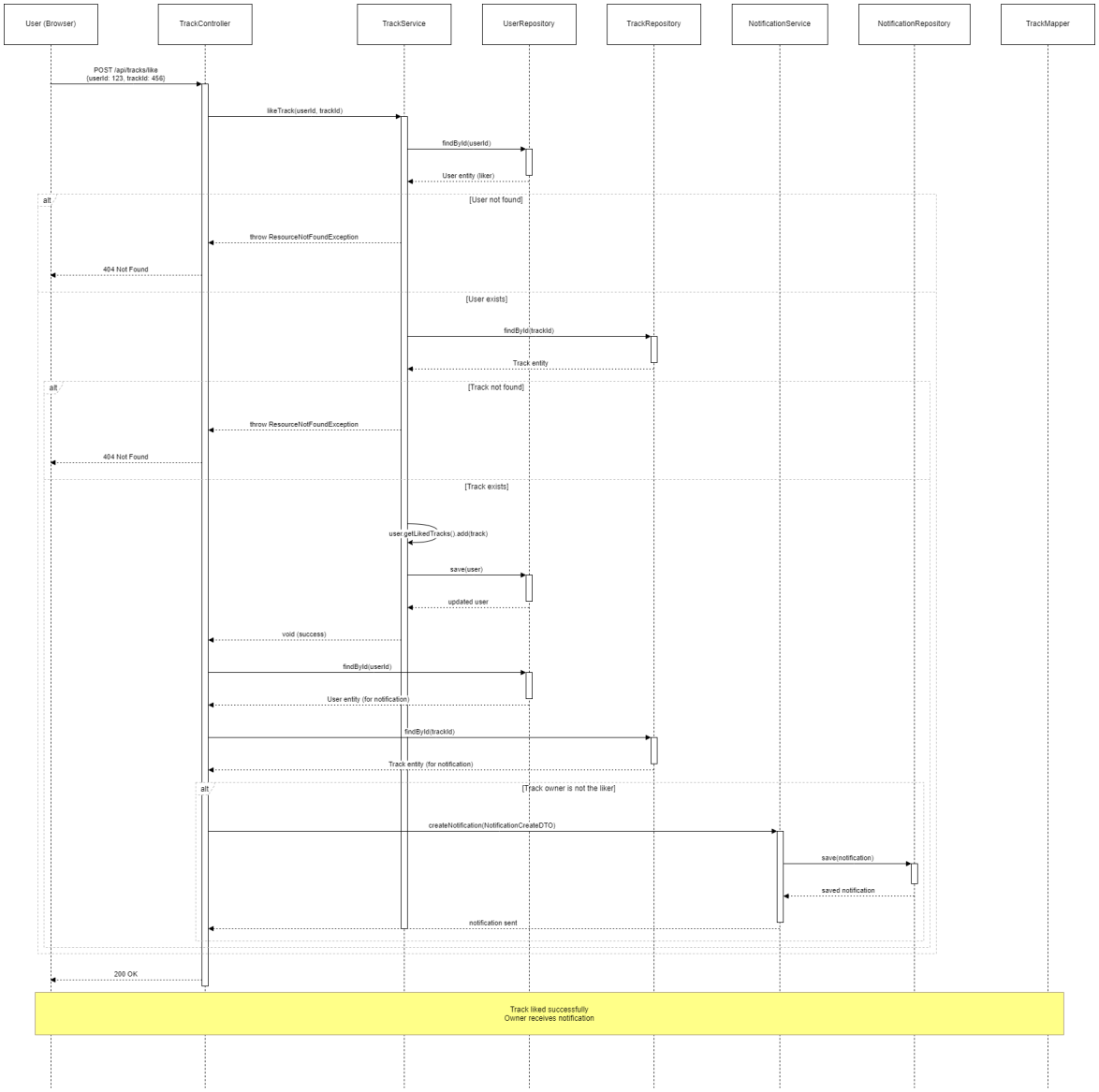
playlists: Có thể nằm trong nhiều playlist khác nhau Được yêu thích: Theo dõi những user đã like track này, thống kê và phân tích: Play count: Số lần phát để xếp hạng độ phổ biến Thời gian: createdAt, updatedAt để theo dõi thời gian tạo và cập nhật bài hát

c, Lớp TrackServiceImpl



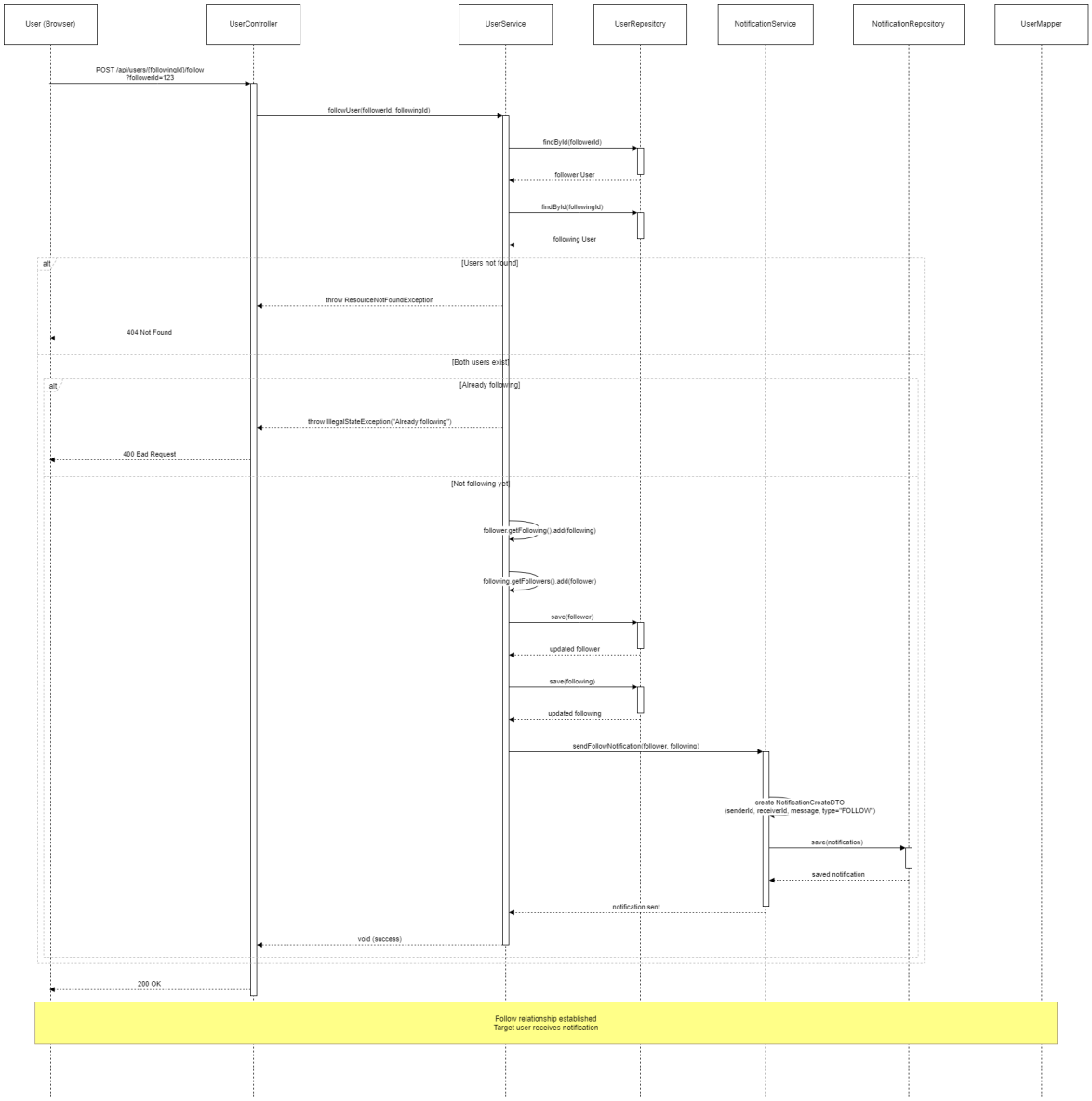
Hình 4.8: Thiết kế lớp TrackServiceImpl

Hình 4.8 mô tả thiết kế lớp TrackServiceImpl - lớp dịch vụ chính xử lý toàn bộ logic nghiệp vụ liên quan đến Track, đóng vai trò cầu nối giữa Controller và Repository. Lớp TrackServiceImpl cung cấp các chức năng quản lí các bài hát cơ bản: tạo mới, thêm mới, sửa thông tin bài hát, xem thông tin bài hát, quản lí tải lên file nhạc: xử lý tải file audio và image Lưu trữ file và tạo URL Tích hợp với FileStorageService, tính năng tìm kiếm và lọc: Tìm kiếm track theo từ khóa, thể loại, người dùng khác, tính năng xã hội: Like/Unlike bài hát, Lấy tracks được like, lấy tracks được like bởi những người đang follow, bên cạnh đó lớp còn hỗ trợ thống kê và phân tích và gợi ý bài hát cho người dùng dựa theo nhu cầu, hành vi của người dùng



Hình 4.9: Biểu đồ trình tự cho nghiệp vụ thích bài hát

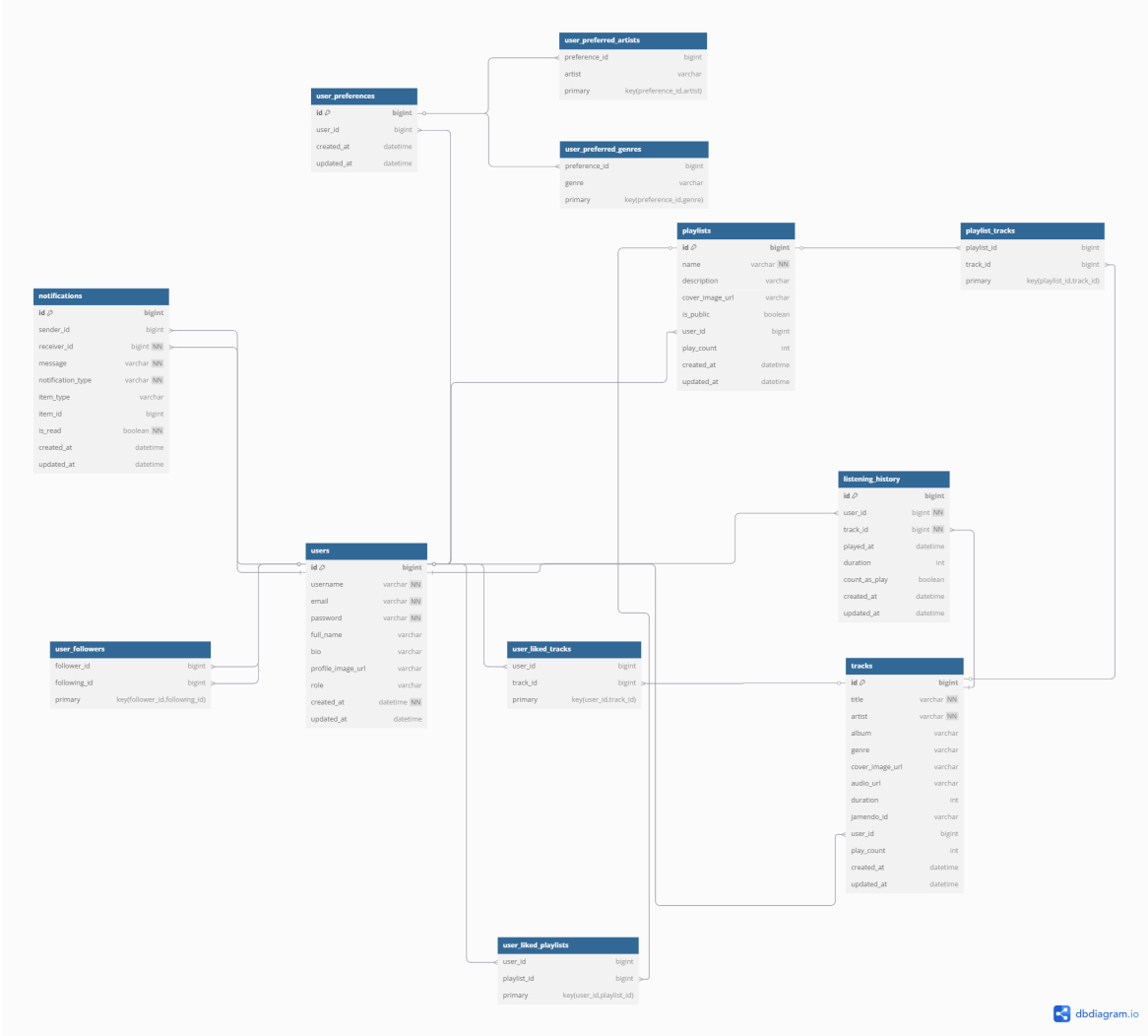
Hình 4.9 mô tả quy trình thích 1 bài hát trong hệ thống. Khi người dùng "like" một bài hát, browser gửi POST request chứa userId và trackId đến TrackController. Controller chuyển tiếp đến TrackService, nơi thực hiện validation user và track thông qua UserRepository và TrackRepository. Nếu validation thành công, service cập nhật mối quan hệ bằng user.getLikedTracks().add(track) và lưu user vào database. Sau đó, TrackController kiểm tra điều kiện không tự like track của mình, rồi gọi NotificationService để tạo thông báo cho chủ sở hữu track. NotificationService lưu notification vào database thông qua NotificationRepository. Cuối cùng, hệ thống trả về 200 OK, xác nhận hành động like thành công và thông báo đã được gửi.



Hình 4.10: Biểu đồ trình tự cho nghiệp vụ người dùng theo dõi người dùng khác

Hình 4.10 mô tả quy trình người dùng theo dõi người dùng khác trong hệ thống. Quá trình follow bắt đầu với POST request đến /api/users/followingId/follow tại UserController. Controller chuyển đến UserService, nơi validation cả hai user thông qua UserRepository. Service kiểm tra mối quan hệ đã tồn tại chưa, nếu chưa thì cập nhật bidirectional relationship: thêm following vào collection của follower và ngược lại. Sau khi lưu cả hai entity vào database, UserService gọi NotificationService để gửi thông báo follow. NotificationService tạo notification với type="FOLLOW" và lưu vào NotificationRepository. Hệ thống trả về 200 OK, xác nhận mối quan hệ follow đã thiết lập và thông báo đã được gửi đến người được follow.

4.4.2 Thiết kế cơ sở dữ liệu



Hình 4.11: Biểu đồ thực thể liên kết ERD

Hình 4.11 là biểu đồ thực thể liên kết của hệ thống. Sơ đồ thực thể liên kết của hệ thống mạng xã hội âm nhạc bao gồm các bảng chính sau:

- **users**: Lưu thông tin người dùng.
- **tracks**: Lưu thông tin bài hát và metadata âm nhạc.
- **playlists**: Lưu danh sách phát do người dùng tạo.
- **listening_history**: Theo dõi lịch sử nghe nhạc của người dùng với thông tin thời gian và thời lượng nghe.

Các bảng quan hệ nhiều-nhiều (many-to-many) bao gồm:

- **user_followers**: Lưu mối quan hệ theo dõi giữa người dùng.
- **user_liked_tracks**: Lưu thông tin bài hát được người dùng yêu thích.

- **user_liked_playlists**: Lưu thông tin playlist được người dùng yêu thích.
- **playlist_tracks**: Lưu thông tin bài hát thuộc về playlist nào.

Ngoài ra, hệ thống còn có các bảng phụ trợ khác nhằm hỗ trợ cá nhân hóa và tương tác xã hội:

- **notifications**: Quản lý thông báo giữa người dùng khi có các hành vi tương tác.
- **user_preferences**: Lưu cài đặt sở thích cá nhân của người dùng.
- **user_preferred_artists**: Chi tiết hóa sở thích về nghệ sĩ của từng người dùng.
- **user_preferred_genres**: Chi tiết hóa sở thích về thể loại âm nhạc của từng người dùng.

Hệ thống cơ sở dữ liệu này tạo nên một nền tảng hoàn chỉnh, hỗ trợ chức năng phát nhạc trực tuyến (streaming), tương tác xã hội và cá nhân hóa trải nghiệm người dùng.

Đặc tả chi tiết cho các bảng như sau:

Thuộc tính	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
id	BIGINT	PRIMARY KEY, AUTO_INCREMENT	Khóa chính, định danh duy nhất
username	VARCHAR(50)	NOT NULL, UNIQUE	Tên đăng nhập
email	VARCHAR(100)	NOT NULL, UNIQUE	Email người dùng
password	VARCHAR(255)	NOT NULL	Mật khẩu đã mã hóa
full_name	VARCHAR(100)	NULL	Họ và tên đầy đủ
bio	TEXT	NULL	Tiểu sử cá nhân
profile_image_url	VARCHAR(500)	NULL	URL ảnh đại diện
role	VARCHAR(20)	DEFAULT ’USER’	Vai trò (USER, ADMIN)
created_at	DATETIME	NOT NULL	Thời gian tạo tài khoản
updated_at	DATETIME	NOT NULL	Thời gian cập nhật cuối

Bảng 4.1: Bảng mô tả thuộc tính của bảng người dùng

Thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
id	BIGINT	PRIMARY KEY, AUTO_INCREMENT	Khóa chính
title	VARCHAR(200)	NOT NULL	Tên bài hát
artist	VARCHAR(100)	NOT NULL	Tên nghệ sĩ
album	VARCHAR(100)	NULL	Tên album
genre	VARCHAR(50)	NULL	Thể loại nhạc
cover_image_url	VARCHAR(500)	NULL	URL ảnh bìa
audio_url	VARCHAR(500)	NULL	URL file âm thanh
jamendo_id	VARCHAR(50)	NULL	ID từ Jamendo API
user_id	BIGINT	FOREIGN KEY REFERENCES users(id)	Người upload
play_count	INT	DEFAULT 0	Số lượt phát
created_at	DATETIME	NOT NULL	Thời gian tạo
updated_at	DATETIME	NOT NULL	Thời gian cập nhật

Bảng 4.2: Bảng mô tả thuộc tính của bảng bài hát

Thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
id	BIGINT	PRIMARY KEY, AUTO_INCREMENT	Khóa chính
name	VARCHAR(100)	NOT NULL	Tên playlist
description	TEXT	NULL	Mô tả playlist
cover_image_url	VARCHAR(500)	NULL	URL ảnh bìa
is_public	BOOLEAN	DEFAULT TRUE	Công khai hay riêng tư
user_id	BIGINT	FOREIGN KEY REFERENCES users(id)	Người tạo playlist
play_count	INT	DEFAULT 0	Số lượt phát
created_at	DATETIME	NOT NULL	Thời gian tạo
updated_at	DATETIME	NOT NULL	Thời gian cập nhật

Bảng 4.3: Bảng mô tả thuộc tính của bảng danh sách phát

Thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
id	BIGINT	PRIMARY KEY, AUTO_INCREMENT	Khóa chính
user_id	BIGINT	FOREIGN KEY REFERENCES users(id)	Người nghe
track_id	BIGINT	FOREIGN KEY REFERENCES tracks(id)	Bài hát được nghe
played_at	DATETIME	NOT NULL	Thời điểm phát
duration	INT	NULL	Thời lượng nghe (giây)
count_as_play	BOOLEAN	DEFAULT TRUE	Có tính vào thống kê không
created_at	DATETIME	NOT NULL	Thời gian tạo record
updated_at	DATETIME	NOT NULL	Thời gian cập nhật

Bảng 4.4: Bảng mô tả thuộc tính của bảng lịch sử nghe nhạc

Thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả

Thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
id	BIGINT	PRIMARY KEY, AUTO_INCREMENT	Khóa chính
sender_id	BIGINT	FOREIGN KEY REFERENCES users(id)	Người gửi
receiver_id	BIGINT	FOREIGN KEY REFERENCES users(id)	Người nhận
message	VARCHAR(500)	NOT NULL	Nội dung thông báo
notification_type	VARCHAR(50)	NOT NULL	Loại thông báo
item_type	VARCHAR(50)	NULL	Loại đối tượng liên quan
item_id	BIGINT	NULL	ID đối tượng liên quan
is_read	BOOLEAN	DEFAULT FALSE	Đã đọc chưa
created_at	DATETIME	NOT NULL	Thời gian tạo
updated_at	DATETIME	NOT NULL	Thời gian cập nhật

Bảng 4.5: Bảng mô tả thuộc tính của bảng thông báo

Thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
follower_id	BIGINT	FOREIGN KEY REFERENCES users(id)	Người theo dõi
following_id	BIGINT	FOREIGN KEY REFERENCES users(id)	Người được theo dõi
primary	COMPOSITE KEY	(follower_id, following_id)	Khóa chính kép

Bảng 4.6: Bảng mô tả thuộc tính của bảng quan hệ theo dõi

Thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
user_id	BIGINT	FOREIGN KEY REFERENCES users(id)	Người thích
track_id	BIGINT	FOREIGN KEY REFERENCES tracks(id)	Bài hát được thích
primary	COMPOSITE KEY	(user_id, track_id)	Khóa chính kép

Bảng 4.7: Bảng mô tả thuộc tính của bảng bài hát được yêu thích

Thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
user_id	BIGINT	FOREIGN KEY REFERENCES users(id)	Người thích
playlist_id	BIGINT	FOREIGN KEY REFERENCES playlists(id)	Playlist được thích
primary	COMPOSITE KEY	(user_id, playlist_id)	Khóa chính kép

Bảng 4.8: Bảng mô tả thuộc tính của bảng playlist được yêu thích

Thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
playlist_id	BIGINT	FOREIGN KEY REFERENCES playlists(id)	Playlist
track_id	BIGINT	FOREIGN KEY REFERENCES tracks(id)	Bài hát
primary	COMPOSITE KEY	(playlist_id, track_id)	Khóa chính kép

Bảng 4.9: Bảng mô tả thuộc tính của bảng bài hát trong playlist

Thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
id	BIGINT	PRIMARY KEY, AUTO_INCREMENT	Khóa chính
user_id	BIGINT	FOREIGN KEY REFERENCES users(id)	Người dùng
created_at	DATETIME	NOT NULL	Thời gian tạo
updated_at	DATETIME	NOT NULL	Thời gian cập nhật

Bảng 4.10: Bảng mô tả thuộc tính của bảng sở thích người dùng

Thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
preference_id	BIGINT	FOREIGN KEY REFERENCES user_preferences(id)	Sở thích
artist	VARCHAR(100)	NOT NULL	Tên nghệ sĩ
primary	COMPOSITE KEY	(preference_id, artist)	Khóa chính kép

Bảng 4.11: Bảng mô tả thuộc tính của bảng nghệ sĩ ưa thích

Thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
preference_id	BIGINT	FOREIGN KEY REFERENCES user_preferences(id)	Sở thích
genre	VARCHAR(50)	NOT NULL	Thể loại nhạc
primary	COMPOSITE KEY	(preference_id, genre)	Khóa chính kép

Bảng 4.12: Bảng mô tả thuộc tính của bảng thể loại ưa thích

Các ràng buộc và chỉ mục bổ sung:

Indexes

- **users**: INDEX(username), INDEX(email)
- **tracks**: INDEX(user_id), INDEX(artist), INDEX(genre)
- **playlists**: INDEX(user_id), INDEX(is_public)
- **listening_history**: INDEX(user_id), INDEX(track_id), INDEX(played_at)
- **notifications**: INDEX(receiver_id), INDEX(is_read)

Constraints

- Tất cả bảng junction có constraint UNIQUE trên composite key
- **user_followers**: CHECK(follower_id != following_id) để tránh tự follow
- **listening_history.duration**: CHECK(duration >= 0)
- **tracks.play_count**, **playlists.play_count**: CHECK(play_count >= 0)

4.5 Xây dựng ứng dụng

4.5.1 Thư viện và công cụ sử dụng

Mục đích	Công cụ/Công nghệ	Phiên bản	URL

Mục đích	Công cụ/Công nghệ	Phiên bản	URL
Backend Development			
Java Runtime	Java JDK	17	https://www.oracle.com/java/
Backend Framework	Spring Boot	3.2.3	https://spring.io/projects/spring-boot
Web Framework	Spring Web MVC	3.2.3	https://spring.io/guides/gs/serving-web-content/
Data Access	Spring Data JPA	3.2.3	https://spring.io/projects/spring-data-jpa
Security Framework	Spring Security	3.2.3	https://spring.io/projects/spring-security
Validation	Spring Boot Validation	3.2.3	https://spring.io/guides/gs/validating-form-input/
Database	MySQL	8.x	https://www.mysql.com/
Database Driver	MySQL Connector/J	Latest	https://dev.mysql.com/downloads/connector/j/
Build Tool	Apache Maven	3.x	https://maven.apache.org/
JWT Authentication	JJWT	0.11.5	https://github.com/jwtk/jjwt
Scheduling	Spring Scheduling	3.2.3	https://spring.io/guides/gs/scheduling-tasks/
External APIs			
Music API	Jamendo API	v3.0	https://api.jamendo.com/v3.0

Bảng 4.13: Các công nghệ backend và API được sử dụng

Mục đích	**Công cụ/Công nghệ**	**Phiên bản**	**URL**
Frontend Development			
JavaScript Library	React	19.1.0	https://reactjs.org/
React DOM	React DOM	19.1.0	https://reactjs.org/docs/react-dom.html
Build Tool	React Scripts	5.0.1	https://create-react-app.dev/
Routing	React Router DOM	7.6.2	https://reactrouter.com/
HTTP Client	Axios	1.9.0	https://axios-http.com/
UI Framework	Material-UI (MUI)	7.1.1	https://mui.com/
Icons	MUI Icons Material	7.1.1	https://mui.com/material-ui/material-icons/
CSS-in-JS	Emotion React	11.14.0	https://emotion.sh/docs/@emotion/react
CSS-in-JS Styled	Emotion Styled	11.14.0	https://emotion.sh/docs/@emotion/styled
Icons Library	Lucide React	0.513.0	https://lucide.dev/
Notifications	React Hot Toast	2.5.2	https://react-hot-toast.com/
Date Utilities	date-fns	4.1.0	https://date-fns.org/

Bảng 4.14: Các công nghệ frontend được sử dụng

4.5.2 Kết quả đạt được

Thông số	Giá trị	Mô tả
Backend Statistics (Spring Boot)		
Tổng số file Java	84 files	Toàn bộ mã nguồn backend
Tổng số dòng code Java	5,442 lines	Không bao gồm comment và dòng trống
Số lớp Controller	11 classes	REST API endpoints
Số interface Service	11 interfaces	Business logic contracts
Số lớp Service Implementation	12 classes	Business logic implementations
Số interface Repository	6 interfaces	Data access layer
Số lớp Entity (Domain)	6 classes	Core domain models
Số lớp Mapper	6 classes	DTO $\leftrightarrow$ Entity conversion
Số lớp DTO	23 classes	Data transfer objects
Số package/thư mục	26 packages	Tổ chức mã nguồn theo module
Frontend Statistics (React)		
Tổng số file JavaScript	51 files	Toàn bộ mã nguồn frontend
Tổng số dòng code JavaScript	15,963 lines	Bao gồm JSX và React components
Số thư mục/module	11 directories	Tổ chức theo features
Database Statistics		
Số bảng chính	12 tables	Core entities và junction tables
Số quan hệ Many-to-Many	4 relationships	user_followers, user_liked_tracks, user_liked_playlists, playlist_tracks
Số quan hệ One-to-Many	8 relationships	users $\rightarrow$ tracks, playlists, notifications, etc.
Số quan hệ One-to-One	1 relationship	users $\rightarrow$ user_preferences

Bảng 4.15: Thống kê chi tiết về hệ thống mạng xã hội cho người thích nghe nhạc

Chúng tôi đã thành công xây dựng một hệ thống mạng xã hội âm nhạc hoàn chỉnh với kiến trúc **N-Layer Architecture**, bao gồm:

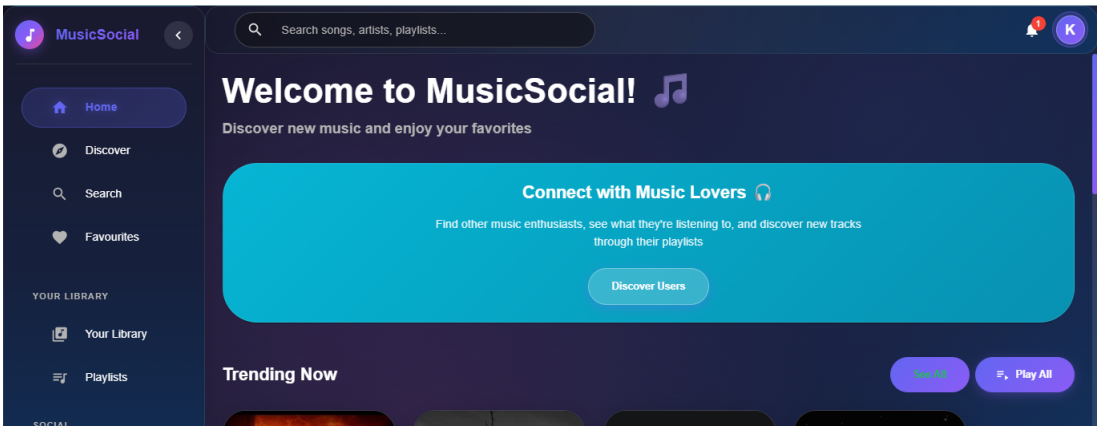
Music Social Backend API (Spring Boot)

- RESTful API server cung cấp toàn bộ dịch vụ backend
- Xác thực và phân quyền người dùng với JWT
- Quản lý người dùng, phát nhạc (streaming), và tính năng mạng xã hội
- Hệ thống gợi ý âm nhạc thông minh

Music Social Frontend Web App (React)

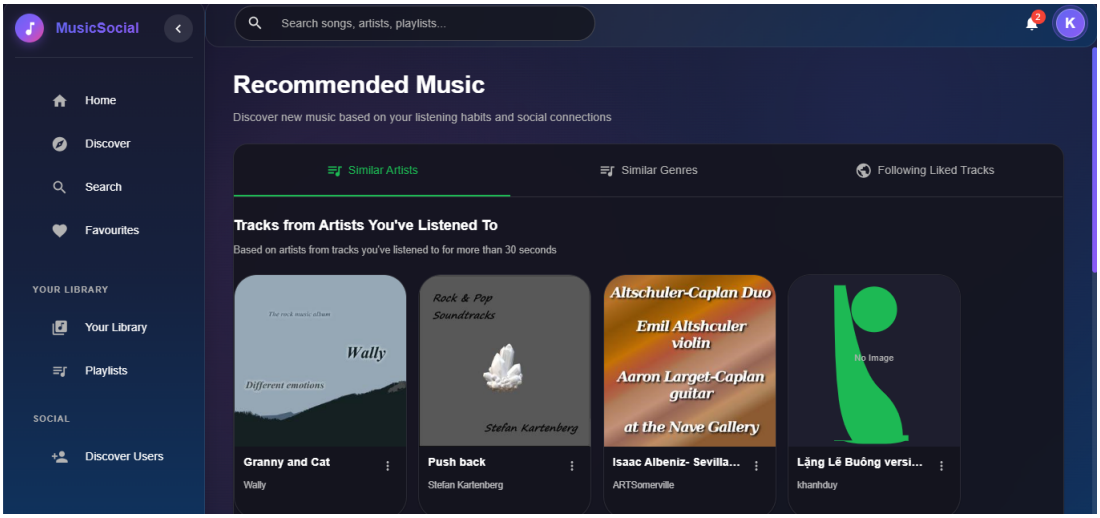
- Giao diện người dùng hiện đại và responsive
- Trình phát nhạc với đầy đủ tính năng điều khiển
- Tính năng mạng xã hội (theo dõi, thích, chia sẻ)
- Quản lý playlist và khám phá âm nhạc

4.5.3 Minh họa các chức năng chính



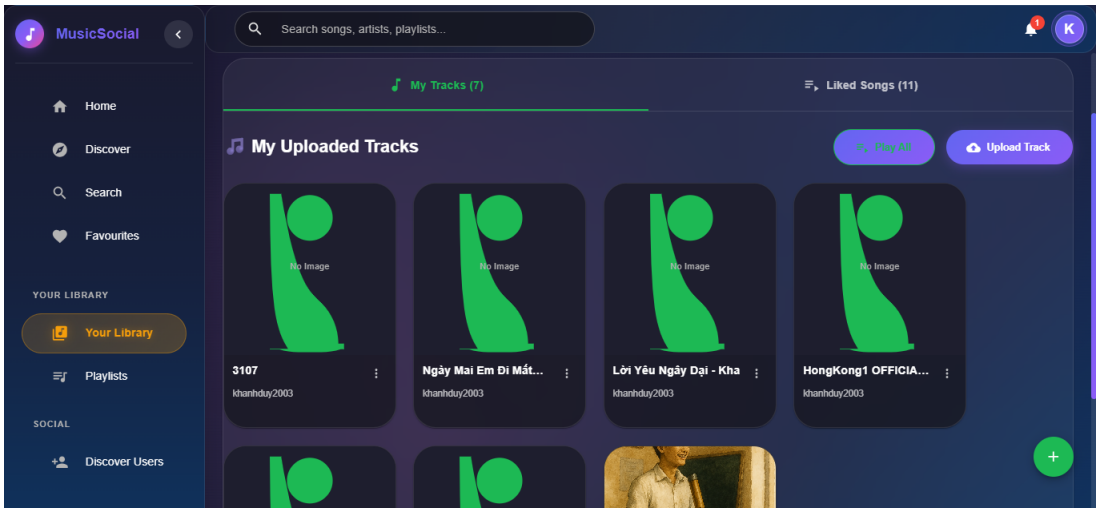
Hình 4.12: Màn hình trang chủ của hệ thống

Hình 4.12 là màn hình trang chủ của hệ thống sau khi người dùng đăng nhập thành công. Trong màn hình trang chủ này, hệ thống sẽ hiển thị cho người dùng những bài hát đang nổi trội, nhiều lượt nghe nhất trong hệ thống. Người dùng có thể chọn See All để được đưa đến màn hình tất cả bài hát trong hệ thống, và nút Play All để người dùng bắt đầu nghe danh sách gồm 20 bài hát đang nhiều lượt nghe nhất hiện tại. Bên cạnh đó, hệ thống gợi ý cho người dùng kết nối với người dùng khác bằng 1 box có nút "Discover User", sau khi người dùng bấm vào sẽ được chuyển sang giao diện danh sách những người dùng tồn tại trong hệ thống.



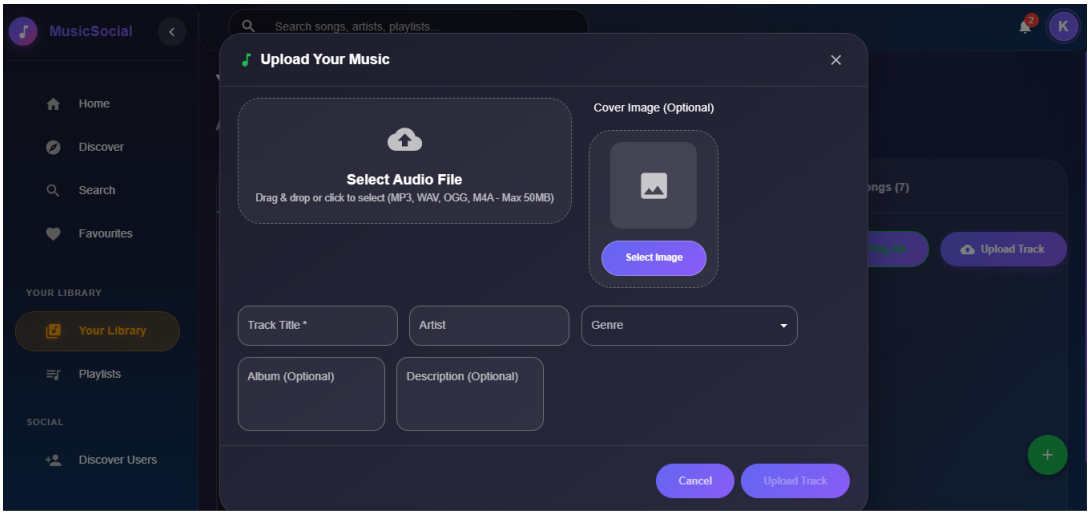
Hình 4.13: Màn hình gợi ý nhạc của hệ thống

Hình 4.13 là màn hình gợi ý nhạc. Trong màn hình này, hệ thống hiểm thị 3 lựa chọn cho người dùng. Người dùng có thể chọn hiển thị gợi ý theo 1 trong 3 lựa chọn sau: Similar Artists - hiển thị những bài hát có cùng tác giả với những bài mà người dùng đã từng hứng thú nghe, Similar Genres - hiển thị những bài hát có cùng thể loại với những bài mà người dùng đã từng hứng thú nghe, Following Liked Track - hiển thị những bài hát được thích bởi những người dùng khác đang được bản thân theo dõi



Hình 4.14: Màn hình hiển thị thư viện của người dùng

Hình 4.14 là màn hình hiển thị thư viện của người dùng. Màn hình này hiển thị 2 danh sách, 1 là những bài hát mà người dùng đã tải lên, 2 là những bài hát mà người dùng đã thích. Sau khi người dùng chọn Upload Track, hệ thống sẽ hiển thị cho người dùng hộp thoại để tải bài hát như ở hình 4.15



Hình 4.15: Màn hình tải nhạc lên hệ thống

Hình 4.15 là màn hình tải nhạc lên hệ thống. Tại đây, người dùng có thể chọn nhạc từ máy tính cá nhân, chọn ảnh cho bài hát đó, người dùng có thể điền tên đặt cho bài hát, với ô Artist, có thể bỏ tống hoặc điền, khi bỏ trống thì hệ thống sẽ mặc định tác giả là username của người dùng, người dùng điền khi muốn bản thân dùng tên khác để tải bài hát lên hệ thống, với Genre, người dùng có thể chọn nhiều thể loại khác nhau, Album-album của bài hát và Description-mô tả bài hát, người dùng có thể điền hoặc để trống. Sau khi đã cập nhật thông tin cho bài hát, người dùng chọn Upload Track để đưa bài hát đó vào hệ thống, hoặc chọn Cancel/nút X ở góc trên bên phải để đóng hộp thoại.

4.6 Kiểm thử

Chúng tôi đã tiến hành kiểm thử 9 trường hợp cho 3 chức năng quan trọng của hệ thống mạng xã hội âm nhạc: chức năng Đăng nhập, chức năng Tải nhạc lên hệ thống và chức năng Theo dõi người dùng với các kỹ thuật kiểm thử phù hợp. Kết quả được thể hiện qua bảng 4.16, bảng 4.17 và bảng 4.18

STT	Trường hợp kiểm thử	Kịch bản kiểm thử	Trạng thái mong muốn	Trạng thái thực tế	Kết quả

STT	Trường hợp kiểm thử	Kịch bản kiểm thử	Trạng thái mong muốn	Trạng thái thực tế	Kết quả
Chức năng 1: User Authentication (Đăng nhập người dùng)					
1	Đăng nhập hợp lệ	Nhập username: "khanhduy", password: "khanhduy2003"	HTTP 200, trả về JWT token và user info	HTTP 200, trả về JWT token và user info	Đạt
2	Đăng nhập sai mật khẩu	Nhập username: "khanhduy", password: "wrongpass"	HTTP 401, thông báo "Invalid username or password"	HTTP 401, thông báo "Invalid username or password"	Đạt
3	Đăng nhập user không tồn tại	Nhập username: "nonexistuser", password: "pass123"	HTTP 401, thông báo "Invalid username or password"	HTTP 401, thông báo "Invalid username or password"	Đạt
4	Đăng nhập với dữ liệu rỗng	Nhập username: "", password: ""	Giao diện hiển thị "Vui lòng điền vào trường này"	Giao diện hiển thị "Vui lòng điền vào trường này"	Đạt

Bảng 4.16: Bảng kiểm thử chức năng Đăng nhập

Ở bảng 4.16, chúng tôi đã dùng các kỹ thuật kiểm thử: Equivalence Partitioning (Phân vùng tương đương), Negative Testing (Kiểm thử tiêu cực), Error Handling Testing

STT	Trường hợp kiểm thử	Kịch bản kiểm thử	Trạng thái mong muốn	Trạng thái thực tế	Kết quả

STT	Trường hợp kiểm thử	Kịch bản kiểm thử	Trạng thái mong muốn	Trạng thái thực tế	Kết quả
Chức năng 2: Track Upload (Tải nhạc lên hệ thống)					
5	Upload track thành công	Upload file MP3 hợp lệ với metadata đầy đủ	HTTP 200, giao diện thông báo Track uploaded successfully và track được lưu vào database	HTTP 200, giao diện thông báo Track uploaded successfully và track được lưu vào database	Đạt
6	Upload file quá dung lượng	Upload file audio > 50MB	Giao diện thông báo Audio file size must be less than 50MB	Giao diện thông báo Audio file size must be less than 50MB	Đạt
7	Upload track thiếu metadata	Upload audio nhưng thiếu title hoặc artist	Màn hình không cho click vào nút Upload Track	Màn hình không cho click vào nút Upload Track	Đạt

Bảng 4.17: Bảng kiểm thử chức năng Tải nhạc lên hệ thống

Ở bảng 4.18, chúng tôi đã dùng các kỹ thuật kiểm thử: Boundary Value Analysis (Phân tích giá trị biên), Positive Testing (Kiểm thử tích cực), User Interface Testing, Data Validation Testing

STT	Trường hợp kiểm thử	Kịch bản kiểm thử	Trạng thái mong muốn	Trạng thái thực tế	Kết quả

STT	Trường hợp kiểm thử	Kịch bản kiểm thử	Trạng thái mong muốn	Trạng thái thực tế	Kết quả
Chức năng 3: Social Interaction - Follow User					
8	Follow user thành công	User A follow User B (chưa follow trước đó)	HTTP 200, tạo mối quan hệ follow, chuyển trạng thái đang follow	HTTP 200, tạo mối quan hệ follow, chuyển trạng thái đang follow	Đạt
9	Unfollow user thành công	User A unfollow User B (đang follow)	HTTP 200, xóa mối quan hệ follow	HTTP 200, xóa mối quan hệ follow	Đạt

Bảng 4.18: Bảng kiểm thử chức năng Theo dõi người dùng

Ở bảng 4.18, chúng tôi đã dùng các kỹ thuật kiểm thử: State Transition Testing (Kiểm thử chuyển đổi trạng thái), Business Logic Testing, Positive Testing, Integration Testing

4.7 Triển khai

4.7.1 Cấu hình triển khai

a, Môi trường phát triển

Hệ thống được triển khai trên môi trường phát triển local với các cấu hình sau:

- **Operating System**: Windows 10 (Build 26100)
- **Development Machine**: Personal Computer
- **RAM**: 8GB+ available
- **Storage**: SSD với đủ dung lượng cho development

b, Backend Configuration

- **Platform**: Spring Boot 3.2.3
- **Java Version**: JDK 17
- **Port**: 8080
- **Database Connection**: MySQL local instance
- **Build Tool**: Maven 3.8+

c, Frontend Configuration

- **Framework**: React 18
- **Development Server**: Webpack Dev Server
- **Port**: 3000
- **Node Version**: 16+
- **Package Manager**: npm

d, Database Configuration

- **DBMS**: MySQL 8.0
- **Port**: 3306 (default)
- **Schema**: music_social_db
- **Connection Pool**: HikariCP (default Spring Boot)

4.7.2 Quy trình triển khai

a, Database Setup

Thiết lập cơ sở dữ liệu MySQL với schema chính:

```
CREATE DATABASE music_social_db;
-- Tables auto-created by JPA/Hibernate
```

b, Backend Deployment

Triển khai ứng dụng Spring Boot:

```
cd backend
mvn clean install
mvn spring-boot:run
```

c, Frontend Deployment

Khởi động React development server:

```
cd frontend
npm install
npm start
```

4.7.3 Đánh giá triển khai

a, Điểm mạnh

- **Stability**: Hệ thống ổn định trong môi trường local
- **Performance**: Đáp ứng tốt yêu cầu testing
- **User Experience**: Interface thân thiện

- **Functionality**: Các tính năng core hoạt động đầy đủ

b, Hạn chế

- **Network**: Local deployment, chưa test qua internet

4.7.4 Kết luận

Hệ thống đã được triển khai thành công trong môi trường local development. Performance và stability đáp ứng tốt yêu cầu demonstration và initial testing. Hệ thống sẵn sàng cho giai đoạn beta testing với nhóm người dùng mở rộng và có tiềm năng triển khai production sau khi optimize infrastructure.

CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 So sánh với các sản phẩm tương tự

Sau khi tìm hiểu chúng tôi nhận thấy rằng việc so sánh với các sản phẩm nổi tiếng như ZingMP3, Spotify, và Apple Music là cần thiết để đánh giá vị thế và tiềm năng của dự án. Quá trình này giúp chúng tôi hiểu rõ điểm mạnh, điểm yếu, và các cơ hội cải tiến dựa trên cách tổ chức code và các tính năng cốt lõi. Mục tiêu là so sánh về mặt kiến trúc, chức năng xã hội, và khả năng xử lý file, dựa trên những gì chúng tôi quan sát được các đặc điểm chung của các nền tảng này.

Qua phân tích, chúng tôi nhận thấy hệ thống của mình sở hữu một số đặc điểm độc đáo so với các nền tảng tương tự:

- **Kiến trúc hệ thống:** Hệ thống của chúng tôi sử dụng Traditional N-Layer Architecture với 4 tầng (Presentation, Business Logic, Data Access, Database), mang lại sự tách biệt rõ ràng giữa các thành phần. Trong khi đó, ZingMP3 dường như dựa trên một kiến trúc đơn giản hơn, tập trung vào streaming và giao diện người dùng, trong khi Spotify và Apple Music áp dụng các mô hình lai giữa microservices và MVC để hỗ trợ quy mô lớn [29], [30]. Điều này cho thấy hệ thống của chúng tôi phù hợp với các dự án vừa và nhỏ, nhưng có thể cần điều chỉnh để mở rộng hơn nữa.
- **Chức năng xã hội:** Chúng tôi đã implement các tính năng như follow/unfollow và real-time notifications, tương tự như Spotify với Group Session và Apple Music với khả năng theo dõi bạn bè. Tuy nhiên, code của chúng tôi cho thấy sự tập trung vào tính cá nhân hóa trong thông báo, điều mà ZingMP3 chưa khai thác mạnh mẽ, trong khi Spotify và Apple Music đã tích hợp sâu vào trải nghiệm cộng đồng.
- **Hệ thống quản lý file:** Với hệ thống file upload có validation toàn diện (kích thước, định dạng, metadata), chúng tôi vượt trội hơn ZingMP3 về kiểm soát chất lượng file, vốn dường như tập trung vào streaming hơn là upload từ người dùng. So với Spotify và Apple Music, chúng tôi cung cấp một giải pháp lý thuyết mạnh mẽ với CDN và cloud storage, nhưng hai nền tảng này có lợi thế về cơ sở hạ tầng lưu trữ quy mô toàn cầu và tích hợp AI để phân tích nội dung.

Kết luận Hệ thống mạng xã hội âm nhạc của chúng tôi mang lại một cách tiếp cận cân bằng giữa tính mô-đun và khả năng tùy chỉnh, phù hợp với các dự án học thuật. So với ZingMP3, chúng tôi có lợi thế về cấu trúc tổ chức và tính năng xã hội, nhưng kém về quy mô và trải nghiệm streaming. So với Spotify và Apple Music,

chúng tôi cần cải tiến về cơ sở hạ tầng và tích hợp công nghệ tiên tiến để cạnh tranh. Tuy nhiên, tính linh hoạt của kiến trúc N-Layer và sự tập trung vào validation file là những điểm sáng, mở ra tiềm năng phát triển trong tương lai nếu được đầu tư thêm.

5.2 Giải pháp và đóng góp nổi bật

Trong quá trình phát triển hệ thống mạng xã hội âm nhạc, chúng tôi đã đối mặt với nhiều thách thức kỹ thuật và đưa ra những giải pháp thiết thực. Chương này trình bày các đóng góp nổi bật nhất mà chúng tôi đã thực hiện, bao gồm việc thiết kế kiến trúc Traditional N-Layer Architecture với 4 tầng và phát triển hệ thống quản lý file upload với validation toàn diện.

5.2.1 Thiết kế kiến trúc Traditional N-Layer Architecture tối ưu

a, Giới thiệu vấn đề

Khi bắt đầu dự án, chúng tôi đứng trước thách thức lựa chọn kiến trúc phù hợp cho một hệ thống mạng xã hội âm nhạc với nhiều chức năng phức tạp. Yêu cầu bao gồm user management, music streaming, playlist management, social interactions (follow/unfollow), và real-time notifications. Vấn đề chính là làm thế nào để tổ chức code sao cho:

- Dễ maintain và extend trong tương lai
- Các components có thể test độc lập
- Business logic tách biệt rõ ràng với data access
- Frontend và backend communicate hiệu quả
- Đảm bảo separation of concerns

Giải pháp kiến trúc N-Layer Chúng tôi đã thiết kế và implement Traditional N-Layer Architecture với 4 tầng được tối ưu cho music social network:

- **Presentation Layer (Tầng Giao diện)**

  Backend: 6 REST Controllers (AuthController, UserController, TrackController, PlaylistController, NotificationController, AdminController)

  Frontend: 32 React components với routing và state management

  Communication: RESTful APIs với JSON data format

- **Business Logic Layer (Tầng Logic Nghiệp vụ)**

  Service Interfaces: 11 service interfaces định nghĩa business contracts

  Service Implementations: UserServiceImpl, TrackServiceImpl, Playlist-

ServiceImpl với complete business logic

DTOs và Mappers: 15+ DTOs cho data transfer và MapStruct mappers cho object conversion

- **Data Access Layer (Tầng Truy cập Dữ liệu)**

  Repository Interfaces: 8 repositories extending JpaRepository

  Custom Queries: Native queries cho complex operations như recommendation algorithms

  Database Operations: CRUD operations với proper transaction management

- **Database Layer (Tầng Cơ sở Dữ liệu)**

  Domain Entities: 8 main entities (User, Track, Playlist, Notification, ListeningHistory, UserPreference)

  MySQL Schema: Properly designed relationships với foreign keys và constraints

  Indexing: Strategic indexes cho performance optimization

Kiến trúc này đảm bảo dependency flow từ trên xuống, loose coupling giữa các layers, và high cohesion trong mỗi layer.

5.2.2 Hệ thống quản lý File Upload với validation toàn diện

a, Dẫn dắt và giới thiệu vấn đề

Music streaming application yêu cầu robust file upload system cho audio files và cover images. Challenges bao gồm:

- File size validation (giới hạn 50MB cho audio files)
- File type validation (chỉ accept audio formats)
- Metadata validation (title, artist bắt buộc)
- Upload progress và error handling
- Storage management và file serving

b, Giải pháp file upload comprehensive

Theo lý thuyết, hệ thống quản lý file upload với validation toàn diện cần được xây dựng dựa trên các nguyên tắc thiết kế phần mềm và quản lý tài nguyên hiệu quả. Giải pháp lý thuyết cho hệ thống này bao gồm các thành phần sau:

- **Backend File Upload Service**: Một dịch vụ trung tâm được thiết kế để xử lý

tất cả các yêu cầu upload, bao gồm:

Xây dựng một API endpoint (ví dụ: '/api/upload') nhận file từ client và kiểm tra kích thước dựa trên giới hạn được định nghĩa trước (như 50MB).

Sử dụng middleware để kiểm tra định dạng file, chỉ chấp nhận các loại như MP3, WAV thông qua kiểm tra MIME type.

Áp dụng cơ chế queue (hàng đợi) để quản lý tiến trình upload, đảm bảo không làm tắc nghẽn server khi xử lý nhiều file đồng thời.

- **Validation Logic**: Một lớp logic kiểm tra toàn diện được tích hợp vào service, bao gồm:

  Kiểm tra metadata (title, artist) là bắt buộc, sử dụng các quy tắc validation như không rỗng và độ dài tối đa.

  Xử lý lỗi bằng cách trả về phản hồi HTTP (ví dụ: 400 Bad Request) với thông điệp lỗi chi tiết khi validation thất bại.

  Triển khai cơ chế rollback nếu quá trình upload bị gián đoạn, đảm bảo tính toàn vẹn dữ liệu.

Giải pháp này dựa trên các nguyên tắc thiết kế phần mềm như Single Responsibility Principle (mỗi thành phần chỉ xử lý một trách nhiệm) và Dependency Inversion Principle (tách biệt phụ thuộc giữa các lớp). Nó đảm bảo tính mở rộng, dễ bảo trì, và khả năng xử lý lỗi một cách hiệu quả trong môi trường lý thuyết.

c, Kết luận

Hệ thống quản lý file upload với validation toàn diện mang lại nhiều lợi ích tiềm năng cho ứng dụng mạng xã hội âm nhạc. Thiết kế này không chỉ giải quyết các thách thức về kích thước file, định dạng, và metadata mà còn cung cấp cơ chế quản lý lưu trữ và theo dõi tiến trình hiệu quả. Sự tách biệt giữa validation logic, storage management, và error handling tạo ra một cấu trúc module hóa, dễ dàng mở rộng cho các yêu cầu tương lai như hỗ trợ video upload hoặc tích hợp AI để phân tích nội dung file. Trong thực tế, việc triển khai giải pháp này đòi hỏi sự cân nhắc kỹ lưỡng về tài nguyên server và tích hợp với các thành phần khác của hệ thống, nhưng về mặt lý thuyết, nó đại diện cho một đóng góp quan trọng trong việc nâng cao chất lượng trải nghiệm người dùng.

5.3 Hướng phát triển

5.3.1 Công việc cần thiết để hoàn thiện các chức năng đã làm

Để hoàn thiện các chức năng hiện tại của hệ thống mạng xã hội âm nhạc, chúng tôi dự kiến tập trung vào việc củng cố các thành phần đã xây dựng. Trước tiên,

chúng tôi cần tối ưu hóa kiến trúc Traditional N-Layer Architecture bằng cách cải thiện hiệu suất giao tiếp giữa các tầng, đảm bảo tính ổn định khi xử lý lưu lượng lớn. Tiếp theo, hệ thống quản lý file upload cần được hoàn thiện với việc triển khai các cơ chế theo dõi tiến trình và xử lý lỗi một cách nhất quán, nhằm nâng cao trải nghiệm người dùng. Cuối cùng, các tính năng xã hội như follow/unfollow và thông báo thời gian thực cần được mở rộng để hỗ trợ nhiều người dùng đồng thời, đảm bảo tính mở rộng và khả năng tương tác.

5.3.2 Hướng đi mới để cải thiện và nâng cấp

Nhìn về tương lai, chúng tôi định hướng sử dụng trí tuệ nhân tạo (AI) để nâng cấp hệ thống, đặc biệt trong việc đưa ra gợi ý nhạc cá nhân hóa cho người dùng. Lý thuyết cho thấy AI, thông qua các mô hình học máy như lọc cộng tác hoặc hệ thống đề xuất dựa trên nội dung, có thể phân tích lịch sử nghe nhạc, sở thích, và hành vi người dùng để đưa ra các đề xuất chính xác hơn. Một hướng đi tiềm năng là tích hợp thuật toán học sâu (deep learning) để dự đoán sở thích dựa trên dữ liệu thưa thớt, kết hợp với phân tích cảm xúc từ metadata âm nhạc. Ngoài ra, việc áp dụng AI để tự động phân loại thể loại hoặc phát hiện nội dung bản quyền trong file upload có thể nâng cao chất lượng và tính pháp lý của hệ thống. Những cải tiến này không chỉ tăng cường giá trị cho người dùng mà còn tạo cơ hội cạnh tranh với các nền tảng lớn như Spotify và Apple Music, đồng thời mở ra hướng nghiên cứu mới trong lĩnh vực ứng dụng AI vào âm nhạc.

TÀI LIỆU THAM KHẢO

[1] Spotify Technology S.A., *Form f-1 registration statement*. [Online]. Available: https://www.sec.gov/Archives/edgar/data/1639920/000119312518063434/d494294df1.htm (visited on 06/16/2025).

[2] Apple Inc., *Apple music*. [Online]. Available: https://www.apple.com/apple-music/ (visited on 06/16/2025).

[3] SoundCloud Limited, *About soundcloud*, 2025. [Online]. Available: https://soundcloud.com/pages/about (visited on 06/16/2025).

[4] SoundCloud Limited, *What is soundcloud?* 2025. [Online]. Available: https://help.soundcloud.com/hc/en-us/articles/115003445087-What-is-SoundCloud- (visited on 06/16/2025).

[5] MIDiA Research, *Independent artist sector 2022*, 2022. [Online]. Available: https://www.midiaresearch.com (visited on 06/16/2025).

[6] SoundGuys, *Soundcloud review: The platform for all*, 2021. [Online]. Available: https://www.soundguys.com/soundcloud-review-48586/ (visited on 06/16/2025).

[7] SoundCloud Limited, *Soundcloud copyright information*, 2025. [Online]. Available: https://soundcloud.com/pages/copyright (visited on 06/16/2025).

[8] SoundCloud Limited, *Fan-powered royalties: A new way for independent artists to get paid*, 2022. [Online]. Available: https://soundcloud.com/press (visited on 06/16/2025).

[9] VNG Corporation, *Zing mp3's parent company files for ipo in the us*, 2023. [Online]. Available: https://www.musicbusinessworldwide.com/vietnams-vng-corp-owner-of-music-streaming-service-zing-mp3-files-for-ipo-in-the-us/ (visited on 06/13/2025).

[10] Zing MP3, *Nghe nhạc mới, hot nhất và tải nhạc miễn phí*, 2025. [Online]. Available: https://zingmp3.vn (visited on 06/13/2025).

[11] Zing MP3, *Zing mp3 - apps on google play*, 2023. [Online]. Available: https://play.google.com/store/apps/details?id=com.zing.mp3 (visited on 06/13/2025).

[12] Vietnam Briefing, *Music streaming services in vietnam: Opportunities and challenges*, 2023. [Online]. Available: https://www.vietnam-briefing.com (visited on 06/13/2025).

[13] Google Play Store, *User reviews: Zing mp3 - apps on google play*, 2023. [Online]. Available: https://play.google.com/store/apps/details?id=com.zing.mp3 (visited on 06/13/2025).

[14] Zing MP3, *Vietnam music market report 2023*, 2024. [Online]. Available: https://pencil.vn/zing-mp3-ra-mat-bao-cao-thi-truong-am-nhac-viet-nam-2023-va-cong-bo-ket-qua-best-of-2023/ (visited on 06/13/2025).

[15] React Team, *React: The library for web and native user interfaces*, 2025. [Online]. Available: https://react.dev (visited on 06/16/2025).

[16] React Team, *React v19*, 2024. [Online]. Available: https://react.dev/blog/2024/12/05/react-19 (visited on 06/16/2025).

[17] Meta Open Source, *React: The library for web and native user interfaces*, 2025. [Online]. Available: https://github.com/facebook/react (visited on 06/16/2025).

[18] SaM Solutions, *Why choose react for mobile app development*, 2024. [Online]. Available: https://sam-solutions.com/blog/why-choose-react-for-mobile-app-development/ (visited on 06/16/2025).

[19] Kinsta, *What is react.js? a look at the popular javascript library*, 2023. [Online]. Available: https://kinsta.com/knowledgebase/what-is-react-js/ (visited on 06/16/2025).

[20] PubNub, *A developers guide to the react library*, 2023. [Online]. Available: https://www.pubnub.com/learn/developers-guide-to-react-library/ (visited on 06/16/2025).

[21] Spring Team, *Spring framework overview*, 2025. [Online]. Available: https://docs.spring.io/spring-framework/reference/overview.html (visited on 06/16/2025).

[22] Baeldung, *Inversion of control and dependency injection with spring*, 2024. [Online]. Available: https://www.baeldung.com/inversion-control-and-dependency-injection-in-spring (visited on 06/16/2025).

[23] Spring Team, *Spring boot overview*, 2025. [Online]. Available: https://docs.spring.io/spring-boot/reference/getting-started/introduction.html (visited on 06/17/2025).

[24] Oracle Corporation, *Mysql 9.0 reference manual: What is mysql?* 2025. [Online]. Available: https://dev.mysql.com/doc/refman/9.0/en/what-is-mysql.html (visited on 06/17/2025).

[25] Oracle Corporation, *Mysql 9.0 reference manual: Json data type*, 2025. [Online]. Available: https://dev.mysql.com/doc/refman/9.0/en/json.html (visited on 06/17/2025).

[26] Oracle Corporation, *Mysql 9.0 reference manual: Security*, 2025. [Online]. Available: https://dev.mysql.com/doc/refman/9.0/en/security.html (visited on 06/17/2025).

[27] Spring Team, *Spring data jpa reference documentation*, 2025. [Online]. Available: https://docs.spring.io/spring-data/jpa/reference/jpa.html (visited on 06/17/2025).

[28] Microsoft, *N-tier architecture*, 2025. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/common-web-application-architectures#n-tier-architecture (visited on 06/17/2025).

[29] Spotify, *Spotify system architecture*, 2025. [Online]. Available: https://engineering.atspotify.com (visited on 06/17/2025).

[30] Apple Inc., *Apple music api*, 2025. [Online]. Available: https://developer.apple.com/documentation/applemusicapi (visited on 06/17/2025).