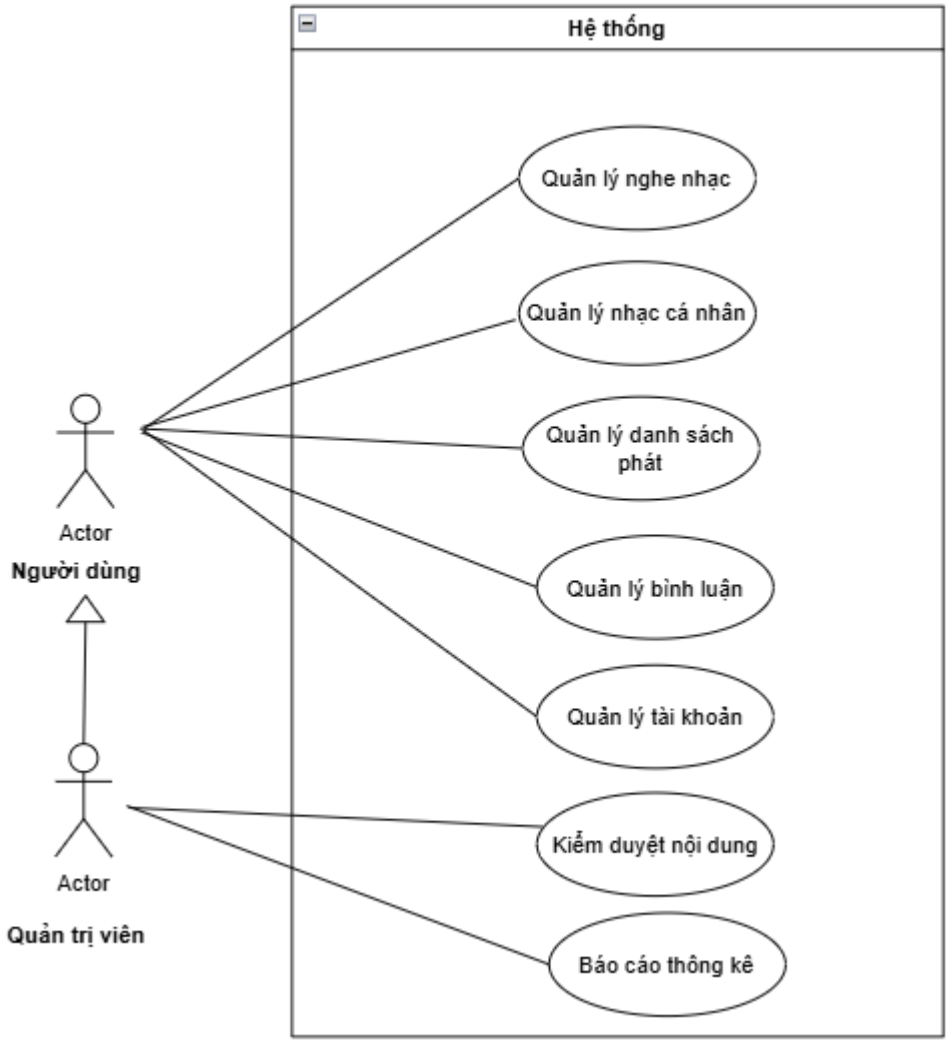
Vũ Huy Hoàng -21A100100149

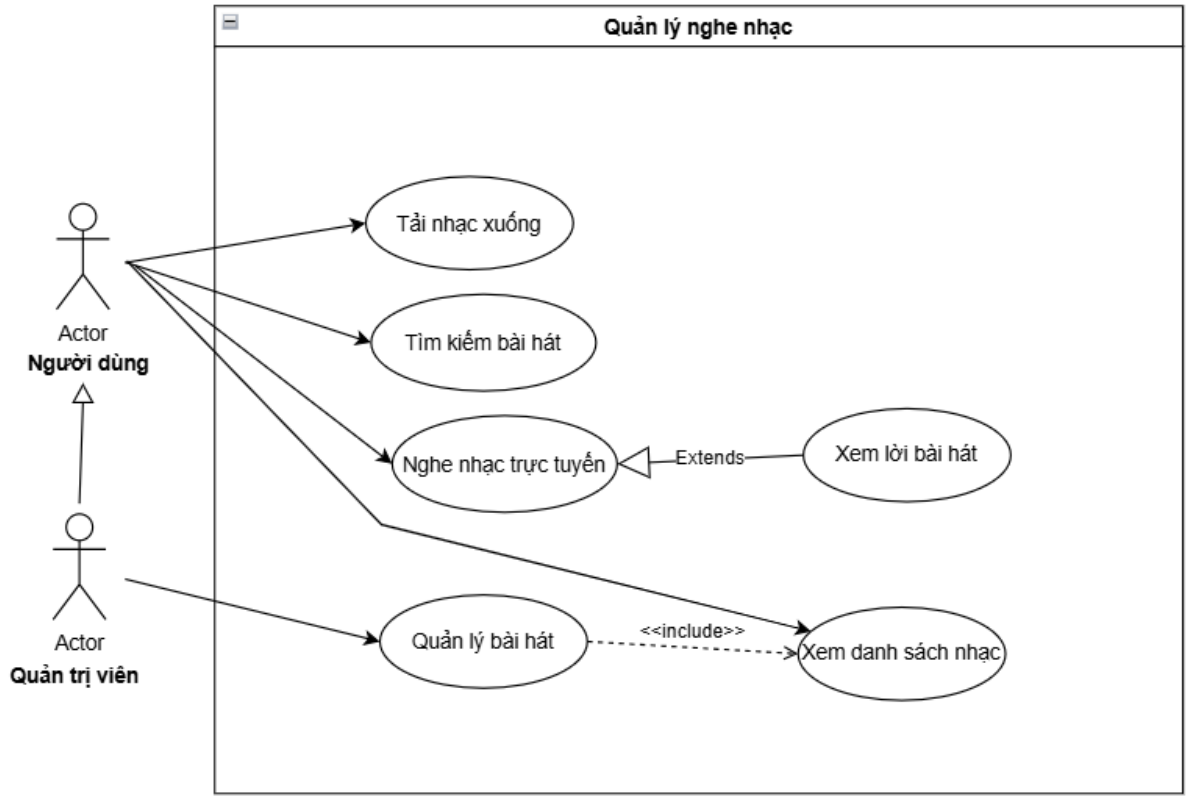
Thiết kế chi tiết cá nhân

Mục lục

1.**Sơ đồ use case tổng quát:**



2.Sơ đồ quản lý nghe nhạc



3. Use case nghe nhạc trực tuyến

a. Mục đích mô tả Use Case "Nghe nhạc trực tuyến"

Chức năng Nghe nhạc trực tuyến và tải nhạc cho phép người dùng phát nhạc ngay trên hệ thống hoặc tải bài hát về máy tính/thiết bị di động để nghe offline. Người dùng có thể điều khiển trình phát nhạc với các chức năng cơ bản như play, pause, chuyển bài, lặp lại, phát ngẫu nhiên. Nếu muốn lưu trữ nhạc trên thiết bị, họ có thể tải xuống bài hát dưới dạng file .mp3.

UC1.01	<Nghe nhạc trực tuyến>
Primary Actor(s)	Quản trị viên, Người dùng
Stakeholders and Interest	*Người dùng muốn nghe nhạc trực tuyến thông qua hệ thống phát nhạc.*
(Điều kiện/hành động thúc đẩy sự bắt đầu) Trigger	*<Condition/action that initiates/starts the use-case>* Người dùng nhấn vào một bài hát để phát nhạc.

<table>
<tr><td>Những điều kiện cần có trước đó

(Pre-conditions)</td><td><Condition assumed to be true before the first step>

-Hệ thống có sẵn danh sách bài hát hợp lệ.

-Người dùng có kết nối internet (nếu phát nhạc trực tuyến).

-Ứng dụng hoặc trình phát nhạc đang hoạt động bình thường.</td></tr>
<tr><td>Những diễn biến sau

(Post-conditions)</td><td><Condition after the use case is successfully executed >

-Bài hát bắt đầu phát, thông tin bài hát hiển thị trên giao diện.

-Người dùng có thể tương tác với trình phát nhạc (tạm dừng, chuyển bài, thay đổi âm lượng, v.v.).

-Nếu có lỗi xảy ra (ví dụ: bài hát không thể phát), hệ thống sẽ hiển thị thông báo lỗi.</td></tr>
<tr><td>Những kịch bản thành công chính (Main Success Scenario)</td><td>
<table>
<tr><td>STT</td><td>Thực hiện bởi</td><td>Hành động</td></tr>
<tr><td>1</td><td>Người dùng</td><td>Nhấn vào một bài hát để phát</td></tr>
<tr><td>2</td><td>Hệ thống</td><td>Hiển thị thông tin bài hát trên thanh phát nhạc

Bắt đầu phát nhạc, thanh tiến trình di chuyển</td></tr>
<tr><td>3</td><td>Người dùng</td><td>Có thể nhấn nút "Pause" để tạm dừng phát nhạc</td></tr>
<tr><td>4</td><td>Người dùng</td><td>Có thể nhấn "→" để chuyển bài hoặc "←" để quay lại bài trước</td></tr>
<tr><td>5</td><td>Người dùng</td><td>Có thể bật chế độ phát ngẫu nhiên hoặc phát lặp lại</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td>
<table>
<tr><td>6</td><td>Người dùng</td><td>Có thể kéo thanh âm lượng để điều chỉnh âm lượng của bài hát</td></tr>
<tr><td>7</td><td>Người dùng</td><td>Có thể nhấn "Pause" để dừng nhạc khi cần</td></tr>
</table>
</td></tr>
<tr><td>Mở rộng (Extensions)</td><td>
If Condition, then

ve Steps

<List any extended steps/ scenarios that occur, other than the main success scenario.>
<table>
<tr><td>STT</td><td>Thực hiện bởi</td><td>Hành động</td></tr>
<tr><td>5</td><td>Hệ thống</td><td>Nếu bài hát không thể phát, hiển thị thông báo lỗi trên màn hình</td></tr>
<tr><td>5a</td><td>Hệ thống</td><td>Nếu không có bài hát nào tiếp theo khi bật chế độ phát ngẫu nhiên, quay lại danh sách phát</td></tr>
<tr><td>5b</td><td>Hệ thống</td><td>Nếu hệ thống không nhận diện được thanh âm lượng, giữ mức âm lượng hiện tại</td></tr>
</table>
</td></tr>
<tr><td>Độ ưu tiên (Priority)</td><td><indicate priority of high, medium or low)

Cao</td></tr>
<tr><td>Những yêu cầu đặc biệt

(Special Requirements</td><td>
<Any system related special requirements needed to fulfill the use case>

Hệ thống hỗ trợ nhiều định dạng âm thanh phổ biến.

Thanh phát nhạc phải phản hồi nhanh và cập nhật thời gian phát chính xác.

Hệ thống cần xử lý tốt lỗi khi bài hát không thể phát.
</td></tr>
<tr><td>Những câu hỏi mở rộng</td><td><Notes and questions></td></tr>
</table>

(Open Questions)	*Có hỗ trợ tải xuống bài hát để nghe offline không?* *Có cần hiển thị lời bài hát trong quá trình phát nhạc không?* *Hệ thống có hỗ trợ gợi ý bài hát tiếp theo dựa trên sở thích người dùng không?*

Kiến trúc phần mềm microservice

4.Hồ sơ kiến trúc

4.1.1 Giới thiệu

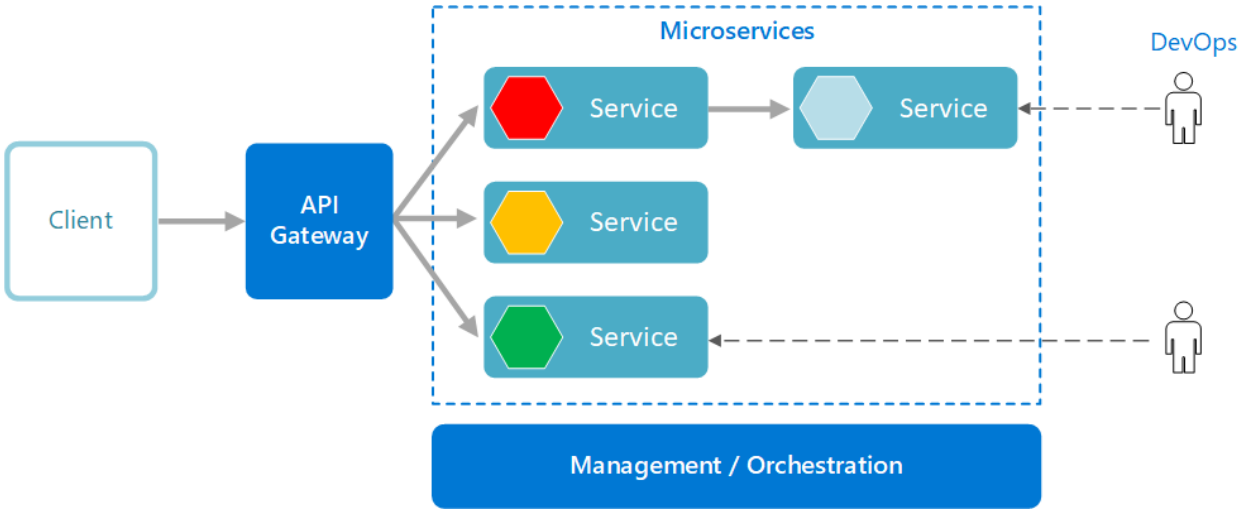
Dự án SoundShare là một nền tảng mạng xã hội chuyên biệt dành cho âm nhạc, cho phép người dùng chia sẻ, khám phá và tương tác với âm nhạc theo cách sáng tạo và linh hoạt. Không giống như các nền tảng nghe nhạc truyền thống, SoundShare tập trung vào việc xây dựng một cộng đồng âm nhạc năng động, nơi người dùng có thể kết nối thông qua sở thích âm nhạc, tạo danh sách phát chung, bình luận và đề xuất bài hát cho nhau.

SoundShare hướng tới việc cải thiện trải nghiệm nghe nhạc cá nhân hóa, đồng thời tạo ra một không gian giao lưu cho những người yêu âm nhạc. Để đáp ứng các yêu cầu mở rộng và tính linh hoạt trong phát triển, hệ thống được thiết kế theo kiến trúc Microservices, giúp tăng khả năng mở rộng, tối ưu hiệu suất và dễ dàng tích hợp các tính năng mới.

Dự án SoundShare được thiết kế theo kiến trúc **Microservices**, cho phép hệ thống mở rộng linh hoạt và tối ưu hóa hiệu suất. Kiến trúc này giúp phân tách hệ thống thành các dịch vụ độc lập, dễ bảo trì và có thể triển khai riêng lẻ. SoundShare không chỉ là một nền tảng nghe nhạc, mà còn tích hợp các tính năng mạng xã hội giúp người dùng kết nối thông qua sở thích âm nhạc.

4.1.1.1 Mô tả kiến trúc microservice

Kiến trúc Microservices (Microservices Architecture) là một phương pháp thiết kế phần mềm trong đó ứng dụng được chia thành nhiều dịch vụ nhỏ (microservices), mỗi dịch vụ đảm nhận một chức năng cụ thể và có thể hoạt động độc lập với nhau. Các dịch vụ này giao tiếp với nhau thông qua API hoặc Message Queue, giúp hệ thống có tính mở rộng cao, linh hoạt và dễ bảo trì.



4.1.2 Nhận dạng các bên liên quan và mối quan tâm

4.1.2.1 Các Bên Liên Quan

Trong quá trình phát triển phần mềm âm nhạc, có nhiều bên liên quan với vai trò và mức độ ảnh hưởng khác nhau. Các bên liên quan chính của sản phẩm này bao gồm:

Bên liên quan (stakeholders) là các cá nhân, nhóm hoặc tổ chức có mối quan tâm đến hệ thống phần mềm **SoundShare**. Các bên liên quan chính trong dự án bao gồm:

+>Người dùng cuối (End Users)

Người dùng cuối là nhóm người trực tiếp sử dụng hệ thống để nghe nhạc, tạo danh sách phát, chia sẻ nhạc hoặc tương tác với cộng đồng. Nhóm này bao gồm:

-Người nghe nhạc:

- Sử dụng hệ thống để phát nhạc trực tuyến, tìm kiếm bài hát và khám phá nội dung mới.
- Tạo và quản lý danh sách phát cá nhân.
- Chia sẻ nhạc và danh sách phát với bạn bè hoặc cộng đồng.
- Thêm bình luận, đánh giá bài hát hoặc nội dung.

-Nghệ sĩ/Người tải nhạc:

- Tải nhạc lên hệ thống để chia sẻ với công chúng hoặc bán nhạc.
- Theo dõi số lượt nghe, số lượt tải xuống và thống kê doanh thu từ nội dung của họ.
- Kiểm soát bản quyền, đăng ký nội dung độc quyền hoặc cấp phép sử dụng nhạc.

-Người quản trị danh sách phát:

- Tạo danh sách phát công khai hoặc riêng tư.
- Quản lý và cập nhật danh sách phát theo xu hướng hoặc sở thích cá nhân.
- Chia sẻ danh sách phát trên các nền tảng khác như mạng xã hội hoặc các ứng dụng nghe nhạc khác.

+>Quản trị viên hệ thống (System Administrators)

- Đảm bảo hệ thống hoạt động ổn định, giám sát hiệu suất và bảo trì phần mềm.
- Quản lý tài khoản người dùng, xử lý các yêu cầu hỗ trợ và giám sát hoạt động của nền tảng.

+>Bộ phận kiểm duyệt nội dung (Content Moderators)

- Đánh giá nội dung nhạc, danh sách phát và bình luận để đảm bảo tuân thủ quy định về bản quyền và tiêu chuẩn cộng đồng.
- Xóa hoặc chặn nội dung vi phạm, xử lý các khiếu nại liên quan đến nội dung.

+>Nhà phát triển phần mềm (Developers & DevOps Team)

- Thiết kế, phát triển và bảo trì hệ thống phần mềm.
- Quản lý cơ sở hạ tầng, triển khai cập nhật và khắc phục sự cố kỹ thuật.

+>Nhà cung cấp dịch vụ lưu trữ (Cloud Storage Providers)

- Cung cấp giải pháp lưu trữ nhạc (ví dụ: AWS S3, Google Cloud Storage) để đảm bảo dữ liệu nhạc được truy cập nhanh và an toàn.
- Đảm bảo tính khả dụng và tối ưu hóa chi phí lưu trữ.

+>Chủ sở hữu nền tảng (Sound Entertainment)

- Đơn vị sở hữu nền tảng, có quyền quyết định về chiến lược kinh doanh, chính sách giá, bản quyền và bảo mật.
- Đảm bảo nền tảng tuân thủ pháp lý và bảo vệ quyền lợi của người dùng.

+>Đối tác kinh doanh (Business Partners & Advertisers)

- Các đơn vị quảng cáo muốn hợp tác để hiển thị quảng cáo trên nền tảng.
- Các công ty âm nhạc, nhà sản xuất nội dung muốn tích hợp dịch vụ của họ với nền tảng.

+>Cơ quan quản lý và pháp luật (Regulatory Authorities)

- Cơ quan giám sát về bản quyền nhạc, luật bảo vệ dữ liệu cá nhân (GDPR, Nghị định 53/2022/NĐ-CP).

- Kiểm tra tính tuân thủ của hệ thống với các quy định về bảo mật, quyền riêng tư và giao dịch điện tử.

4.1.2.2 Các mối quan tâm

Mỗi bên liên quan có các mối quan tâm khác nhau đối với hệ thống SoundShare. Các mối quan tâm chính bao gồm:

+>***Trải nghiệm người dùng (User Experience - UX)***

-Giao diện đơn giản, dễ sử dụng và hỗ trợ đa nền tảng.

-Tốc độ tải trang nhanh, tìm kiếm nhạc mượt mà, thời gian phản hồi tối ưu.

-Tính năng cá nhân hóa đề xuất nhạc dựa trên sở thích người dùng.

+>***Hiệu suất và khả năng mở rộng (Performance & Scalability)***

-Hệ thống có khả năng xử lý hàng triệu yêu cầu phát nhạc đồng thời.

-Tối ưu hóa băng thông, giảm độ trễ khi phát nhạc.

-Khả năng mở rộng linh hoạt để đáp ứng nhu cầu người dùng tăng cao.

+>**Bảo mật và quyền riêng tư (Security & Privacy)**

-Xác thực hai bước (2FA) và phân quyền người dùng để bảo vệ tài khoản.

-Mã hóa dữ liệu cá nhân và bảo vệ quyền riêng tư của người dùng.

-Kiểm soát truy cập dữ liệu nhạc và chống sao chép trái phép.

+>***Bản quyền và quản lý nội dung (Copyright & Content Management)***

- Hệ thống phải đảm bảo nhạc tải lên không vi phạm bản quyền.
- Cơ chế kiểm duyệt nội dung tự động và thủ công để đảm bảo chất lượng.
- Chính sách báo cáo và xử lý khiếu nại liên quan đến nội dung.

+>**Khả năng tích hợp (Interoperability & API Support)**

-Hỗ trợ kết nối với các nền tảng khác như Facebook, TikTok, YouTube để chia sẻ nhạc.

-Tương thích với các thiết bị như loa thông minh, TV, ô tô (Apple CarPlay, Android Auto).

-Cung cấp API mở để bên thứ ba có thể phát triển ứng dụng tích hợp.

+>***Quản lý tài khoản và thanh toán (Account & Monetization)***

-Hỗ trợ mô hình đăng ký Premium, quảng cáo để tạo doanh thu.

-Đảm bảo giao dịch an toàn khi người dùng mua nhạc hoặc nâng cấp tài khoản.

-Hệ thống quản lý thanh toán minh bạch, hỗ trợ nhiều phương thức thanh toán.

+>Tính khả dụng và sao lưu dữ liệu (Availability & Backup)

-Hệ thống cần đảm bảo hoạt động 24/7, không bị downtime.

-Cơ chế sao lưu dữ liệu hàng ngày để tránh mất mát thông tin.

-Kế hoạch dự phòng để khôi phục hệ thống trong trường hợp sự cố.

4.1.2.3 Mối quan tâm - Bên liên quan

Trong đó:

x chỉ ra bên liên quan có mối quan tâm với chủ đề tương ứng.

- nghĩa là không có mối quan tâm trực tiếp.

Bên liên quan	Mối Quan Tâm Chính
Người dùng cuối (End Users)	- Giao diện dễ sử dụng, trải nghiệm mượt mà trên nhiều thiết bị. - Tìm kiếm nhạc nhanh, cá nhân hóa nội dung theo sở thích. - Chất lượng nhạc cao, không bị gián đoạn khi phát. - Khả năng tương tác: chia sẻ nhạc, bình luận, tạo danh sách phát. - Tính bảo mật cao, bảo vệ quyền riêng tư cá nhân.
Nghệ sĩ/Người tải nhạc	- Công cụ tải nhạc lên nhanh chóng, dễ quản lý. - Kiểm soát bản quyền và cấp phép sử dụng nhạc. - Kiếm tiền từ nhạc qua quảng cáo hoặc gói đăng ký. - Thống kê chi tiết về lượt nghe, tải xuống và thu nhập.

Người quản trị danh sách phát	- Quản lý danh sách phát dễ dàng. - Tùy chỉnh danh sách phát công khai hoặc riêng tư. - Chia sẻ danh sách phát đến cộng đồng hoặc qua mạng xã hội.
Quản trị viên hệ thống (System Admins)	- Đảm bảo hệ thống hoạt động ổn định 24/7. - Quản lý tài khoản, phân quyền truy cập hệ thống. - Giám sát hiệu suất, phát hiện và xử lý lỗi nhanh chóng. - Cập nhật, bảo trì hệ thống định kỳ.
Bộ phận kiểm duyệt nội dung	- Xác minh nội dung tải lên có vi phạm bản quyền không. - Quản lý nội dung vi phạm hoặc bị khiếu nại. - Duy trì nền tảng lành mạnh, tránh nội dung không phù hợp.
Nhà phát triển phần mềm & DevOps	- Hệ thống dễ mở rộng và bảo trì. - Đảm bảo hiệu suất cao, tối ưu thời gian phản hồi API. - Quản lý CI/CD để triển khai nhanh và ổn định. - Hỗ trợ tích hợp API với bên thứ ba.
Nhà cung cấp dịch vụ lưu trữ (Cloud Providers)	- Lưu trữ nhạc an toàn, tối ưu hóa chi phí. - Đảm bảo tốc độ tải nhanh, giảm độ trễ khi phát nhạc. - Cung cấp dịch vụ sao lưu và khôi phục dữ liệu khi cần thiết.
Chủ sở hữu nền tảng (Sound Entertainment)	- Tối ưu hóa doanh thu từ đăng ký, quảng cáo và đối tác.

	- Đảm bảo nền tảng tuân thủ quy định pháp luật. - Mở rộng thị trường và phát triển nền tảng. - Duy trì quyền kiểm soát hệ thống và dữ liệu người dùng.
Đối tác kinh doanh và nhà quảng cáo	- Hợp tác quảng cáo để tiếp cận khách hàng. - Đảm bảo quảng cáo không ảnh hưởng đến trải nghiệm người dùng. - Tích hợp quảng cáo thông minh, nhắm đúng đối tượng.
Cơ quan quản lý và pháp luật	- Hệ thống tuân thủ luật bản quyền (DMCA, GDPR). - Đảm bảo chính sách bảo mật và quyền riêng tư của người dùng. - Kiểm soát nội dung nhạc theo quy định pháp luật.

4.1.3 Quan kiểm kiến trúc

4.1.3.1 Quan điểm kiến trúc(Architecture viewpoints)

Quan điểm kiến trúc (Architecture Viewpoint) là một tập hợp các quan điểm được xây dựng để giải quyết các mối quan tâm của các bên liên quan khác nhau trong hệ thống phần mềm. Mỗi quan điểm cung cấp một cái nhìn cụ thể về kiến trúc của hệ thống, giúp nhóm phát triển đảm bảo rằng thiết kế tổng thể phù hợp với các yêu cầu chức năng và phi chức năng đã đề ra.

Ý nghĩa trong dự án:

- Các quan điểm kiến trúc giúp phân tích và thiết kế hệ thống từ nhiều góc độ khác nhau, chẳng hạn như chức năng, dữ liệu, bảo mật và hạ tầng.
- Đối với dự án xây dựng phần mềm mạng xã hội chia sẻ âm nhạc SoundShare cho Sound Entertainment theo Công nghệ phần mềm, mỗi quan điểm sẽ tập trung vào một khía cạnh như giao diện người dùng, xử lý nghiệp vụ, và cơ sở dữ liệu để đảm bảo hệ thống hoạt động ổn định và đáp ứng yêu cầu.

Các loại quan điểm phổ biến:

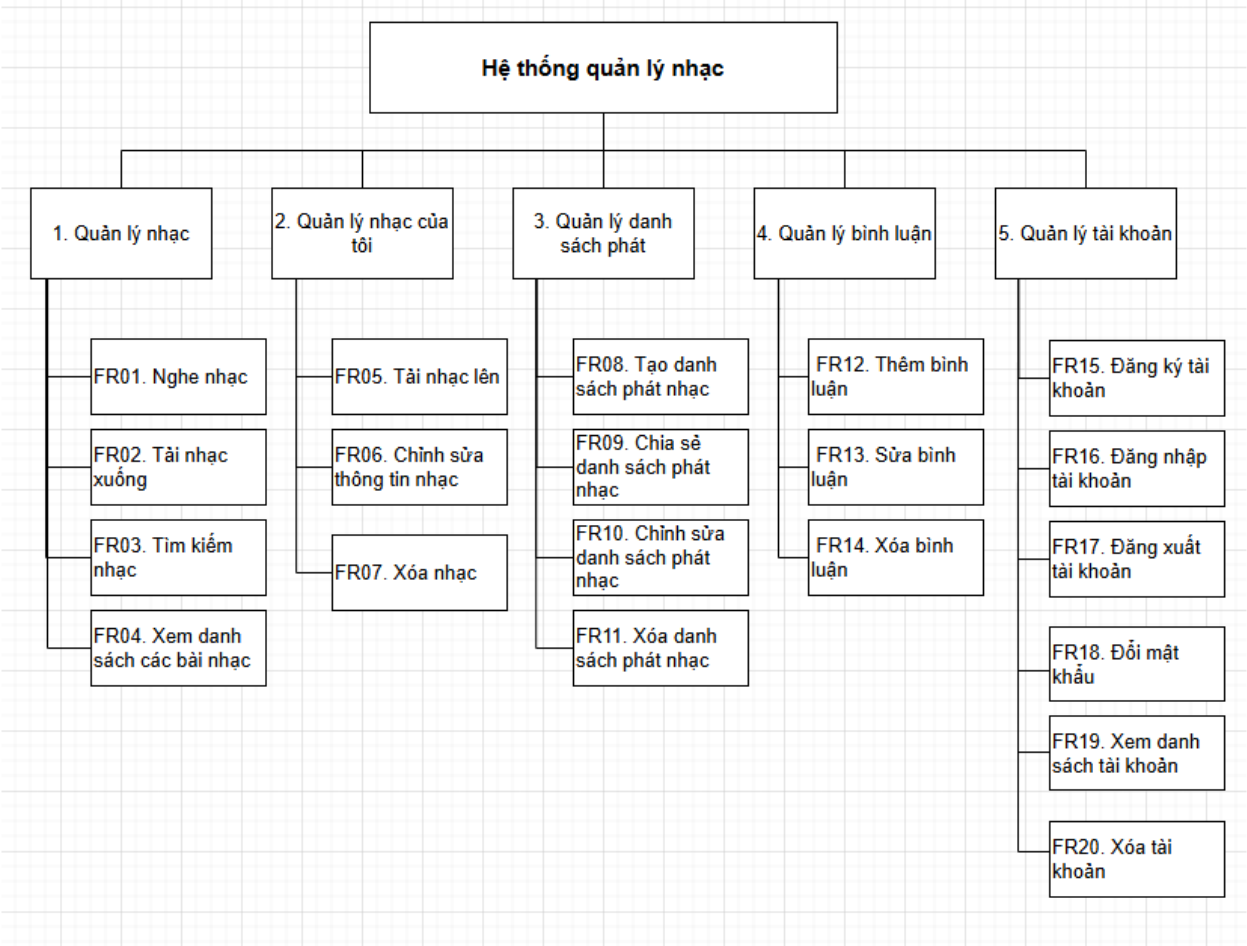
- Module View (Quan điểm chức năng): Tập trung vào các chức năng chính của hệ thống.

- Component and connector view (Quan điểm bảo mật): Xác định các biện pháp bảo vệ thông tin và kiểm soát truy cập.
- Deployment View (Quan điểm triển khai): Mô tả cách thức hệ thống được triển khai trong môi trường thực tế.

4.1.4 Góc nhìn kiến trúc(Views)

3.1.4.1 Module views

3.1.4.1.1 Decomposition Diagram(MH)



Mô tả sơ đồ Decomposition Diagram

- Hệ thống quản lý nhạc được chia thành 5 nhóm chức năng chính:

1. Quản lý nhạc: Bao gồm các chức năng cơ bản như nghe nhạc, tải nhạc, tìm kiếm nhạc và xem danh sách bài hát.

2. Quản lý nhạc của tôi: Cho phép người dùng tải nhạc lên, chỉnh sửa thông tin nhạc và xóa nhạc cá nhân.
3. Quản lý danh sách phát: Hỗ trợ người dùng tạo, chia sẻ, chỉnh sửa và xóa danh sách phát nhạc.
4. Quản lý bình luận: Gồm các chức năng thêm, sửa và xóa bình luận trên hệ thống.
5. Quản lý tài khoản: Gồm các chức năng đăng ký, đăng nhập, đăng xuất, đổi mật khẩu, xem và xóa tài khoản.

- Mục đích của sơ đồ
- Giúp hiểu rõ hệ thống bằng cách chia nhỏ chức năng.
- Hỗ trợ thiết kế hệ thống phần mềm.
- Dễ dàng quản lý các yêu cầu chức năng trong quá trình phát triển.
- Quan hệ giữa các nhóm chức năng

Nhóm chức năng	**Liên kết với**	**Mối quan hệ**
Quản lý nhạc	Quản lý nhạc của tôi, Quản lý danh sách phát	Người dùng có thể nghe, tìm kiếm nhạc từ cả nhạc cá nhân và danh sách phát.
Quản lý nhạc của tôi	Quản lý nhạc	Nhạc do người dùng tải lên sẽ xuất hiện trong danh sách nhạc để tìm kiếm, nghe.
Quản lý danh sách phát	Quản lý nhạc	Người dùng có thể tạo danh sách phát từ các bài hát có sẵn.
Quản lý bình luận	Quản lý nhạc, Quản lý danh sách phát	Người dùng có thể bình luận về bài hát hoặc danh sách phát.
Quản lý tài khoản	Tất cả các chức năng	Người dùng cần có tài khoản để tải nhạc, tạo danh sách phát, bình luận,...

- **Quan hệ giữa các chức năng bên trong từng nhóm**

Nhóm 1: Quản lý nhạc

- FR01 (Nghe nhạc) phụ thuộc vào FR03 (Tìm kiếm nhạc) để tìm bài hát trước khi nghe.
- FR04 (Xem danh sách nhạc) giúp hiển thị các bài hát có thể tìm kiếm, nghe hoặc tải về.

Nhóm 2: Quản lý nhạc của tôi

- FR05 (Tải nhạc lên) là bước đầu tiên để có nhạc cá nhân.
- FR06 (Chỉnh sửa thông tin nhạc) giúp thay đổi thông tin của nhạc đã tải lên.
- FR07 (Xóa nhạc) có thể được sử dụng nếu không còn cần bài hát.

Nhóm 3: Quản lý danh sách phát

- FR08 (Tạo danh sách phát) cần FR04 (Xem danh sách nhạc) để chọn bài hát đưa vào danh sách.
- FR09 (Chia sẻ danh sách phát) chỉ khả dụng khi đã có danh sách phát.
- FR10 (Chỉnh sửa danh sách phát) liên quan đến FR08 (Tạo danh sách phát) để thay đổi nội dung danh sách.
- FR11 (Xóa danh sách phát) là hành động cuối cùng khi không muốn giữ danh sách nữa.

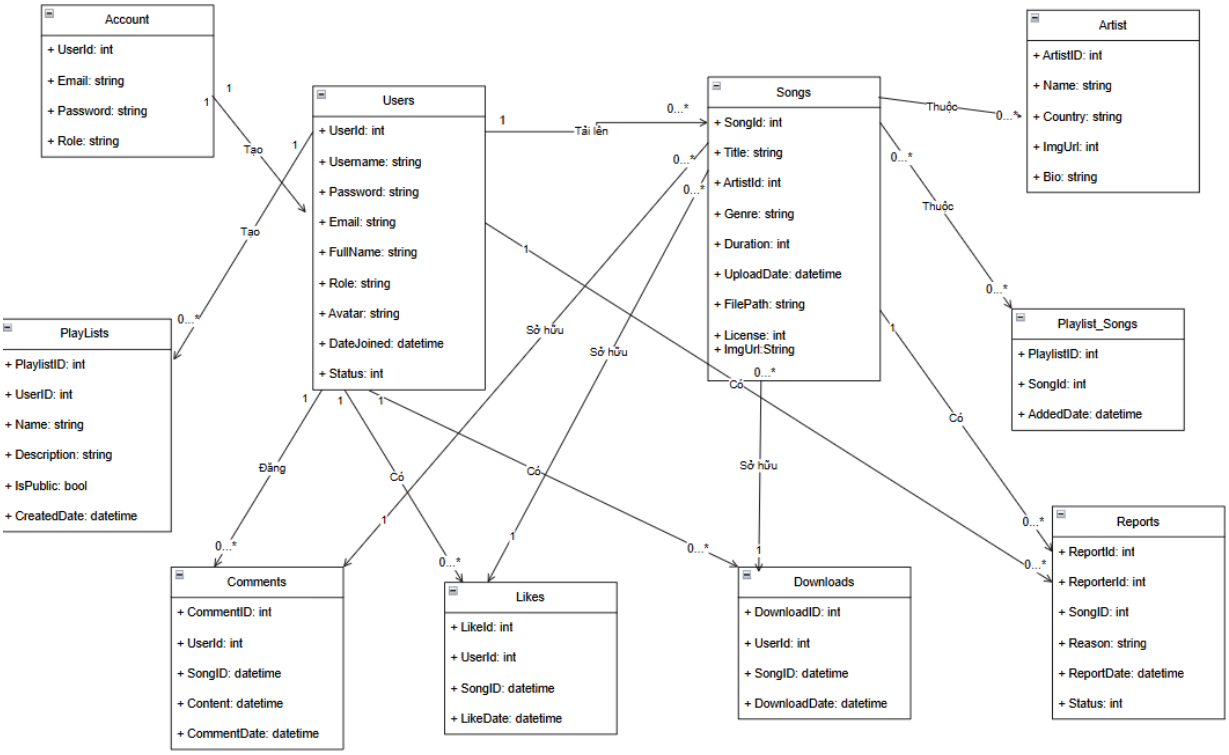
Nhóm 4: Quản lý bình luận

- FR12 (Thêm bình luận) là bước đầu tiên để người dùng tương tác.
- FR13 (Sửa bình luận) cho phép chỉnh sửa bình luận đã đăng.
- FR14 (Xóa bình luận) giúp người dùng quản lý bình luận của mình.

Nhóm 5: Quản lý tài khoản

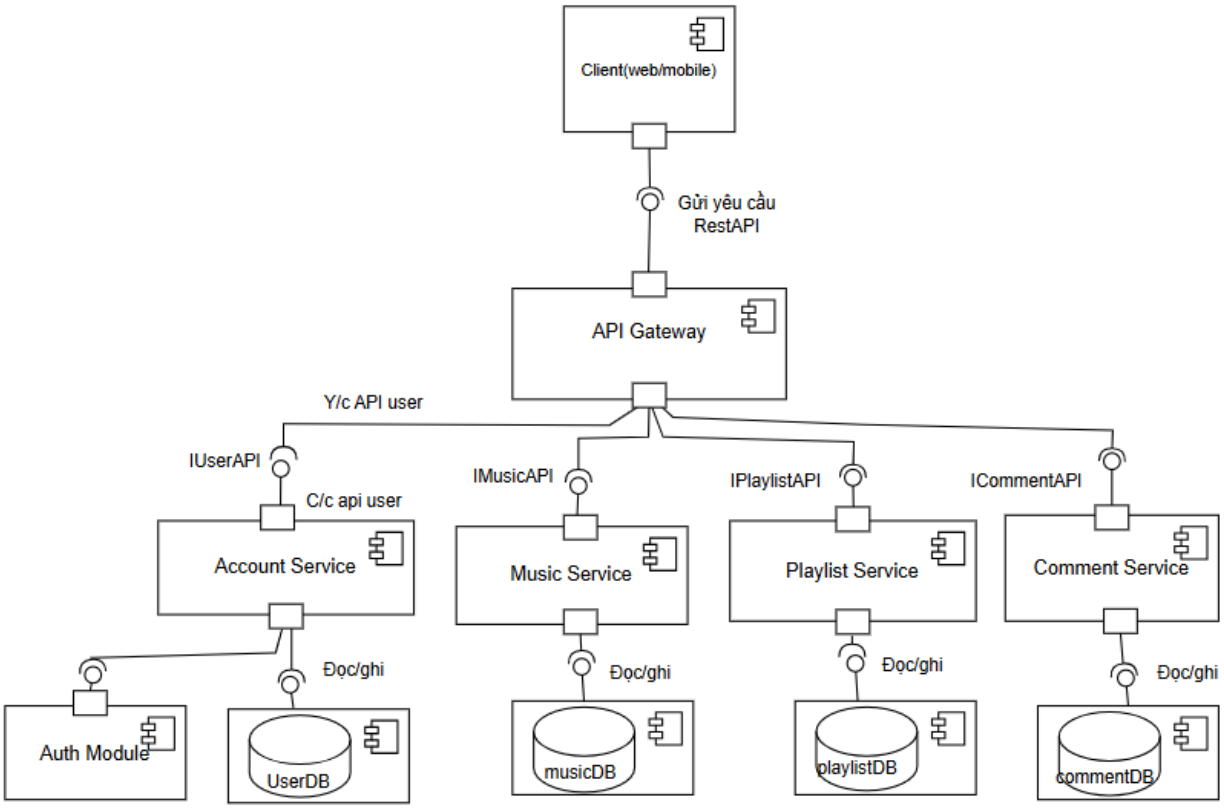
- FR15 (Đăng ký tài khoản) là bước khởi đầu cho tất cả các hoạt động.
- FR16 (Đăng nhập tài khoản) cần thiết để truy cập các chức năng khác.
- FR17 (Đăng xuất tài khoản) giúp người dùng thoát hệ thống.
- FR18 (Đổi mật khẩu) giúp duy trì bảo mật tài khoản.
- FR19 (Xem danh sách tài khoản) có thể dành cho quản trị viên hoặc chính người dùng.
- FR20 (Xóa tài khoản) là hành động cuối cùng để rời khỏi hệ thống.

4.1.4.1.2 Class diagram(Hoàn)



4.1.4.2 Component and connector views

Component diagram (Vu HuyHoang)



Mô tả:

Sơ đồ Component Diagram UML trên thể hiện kiến trúc hệ thống nghe nhạc trực tuyến được xây dựng theo mô hình Microservices Architecture. Hệ thống bao gồm nhiều thành phần riêng biệt, mỗi thành phần chịu trách nhiệm thực hiện một nhóm chức năng cụ thể và giao tiếp với nhau thông qua API Gateway.

Các thành phần chính của hệ thống bao gồm Client (Web/Mobile), API Gateway, Music Service, Playlist Service, Comment Service, User Service, Auth Module, File Storage (S3), và các cơ sở dữ liệu chuyên biệt.

+> Client và API Gateway

- Client (Web/Mobile) là giao diện người dùng, nơi người dùng có thể thực hiện các thao tác như nghe nhạc, tải nhạc, quản lý danh sách phát, bình luận và xác thực tài khoản. Mọi yêu cầu từ Client đều được gửi tới API Gateway thông qua REST API. API Gateway đóng vai trò trung gian, giúp điều phối các yêu cầu, bảo vệ hệ thống, quản lý quyền truy cập và tối ưu hóa hiệu suất.

+>API Gateway và các Microservices

API Gateway là điểm truy cập duy nhất giữa Client và các Microservices trong hệ thống. Nó tiếp nhận request từ người dùng và phân phối chúng đến các Microservices tương ứng để xử lý. API Gateway sử dụng các Required Interfaces để gửi yêu cầu đến các dịch vụ sau:

- **IMusicAPI** để gọi Music Service thực hiện các chức năng liên quan đến phát nhạc.
- IPlaylistAPI để gọi Playlist Service xử lý danh sách phát.
- ICommentAPI để gọi Comment Service quản lý bình luận.
- IUserAPI để gọi User Service thực hiện các chức năng liên quan đến tài khoản người dùng.

Các Microservices này cung cấp Provided Interfaces để API Gateway có thể gọi đến khi cần.

+> Music Service (Quản lý nhạc)

- Cung cấp IMusicAPI để API Gateway gọi đến.
- Chịu trách nhiệm xử lý phát nhạc, tải nhạc lên/xuống, tìm kiếm và chỉnh sửa thông tin bài hát.
- Kết nối với Music DB để lưu trữ thông tin bài hát.
- Kết nối với File Storage (S3) để lưu trữ các file nhạc thực tế.

+>Playlist Service (Quản lý danh sách phát)

- Cung cấp IPlaylistAPI để API Gateway gọi đến.
- Quản lý danh sách phát, bao gồm tạo, chỉnh sửa, chia sẻ và xóa danh sách phát.
- Sử dụng Playlist DB để lưu trữ danh sách phát của người dùng.

+> Comment Service (Quản lý bình luận)

- Cung cấp ICommentAPI để API Gateway gọi đến.
- Cho phép người dùng thêm, chỉnh sửa và xóa bình luận trên bài hát và danh sách phát.
- Kết nối với Comment DB để lưu trữ và truy xuất dữ liệu bình luận.

+>User Service (Quản lý tài khoản)

- Cung cấp IUserAPI để API Gateway gọi đến.
- Đảm nhiệm các chức năng đăng ký, đăng nhập, đổi mật khẩu, quản lý danh sách tài khoản và xóa tài khoản.
- Kết nối với User DB để lưu thông tin người dùng.
- Tích hợp với Auth Module để xác thực danh tính và kiểm soát quyền truy cập.

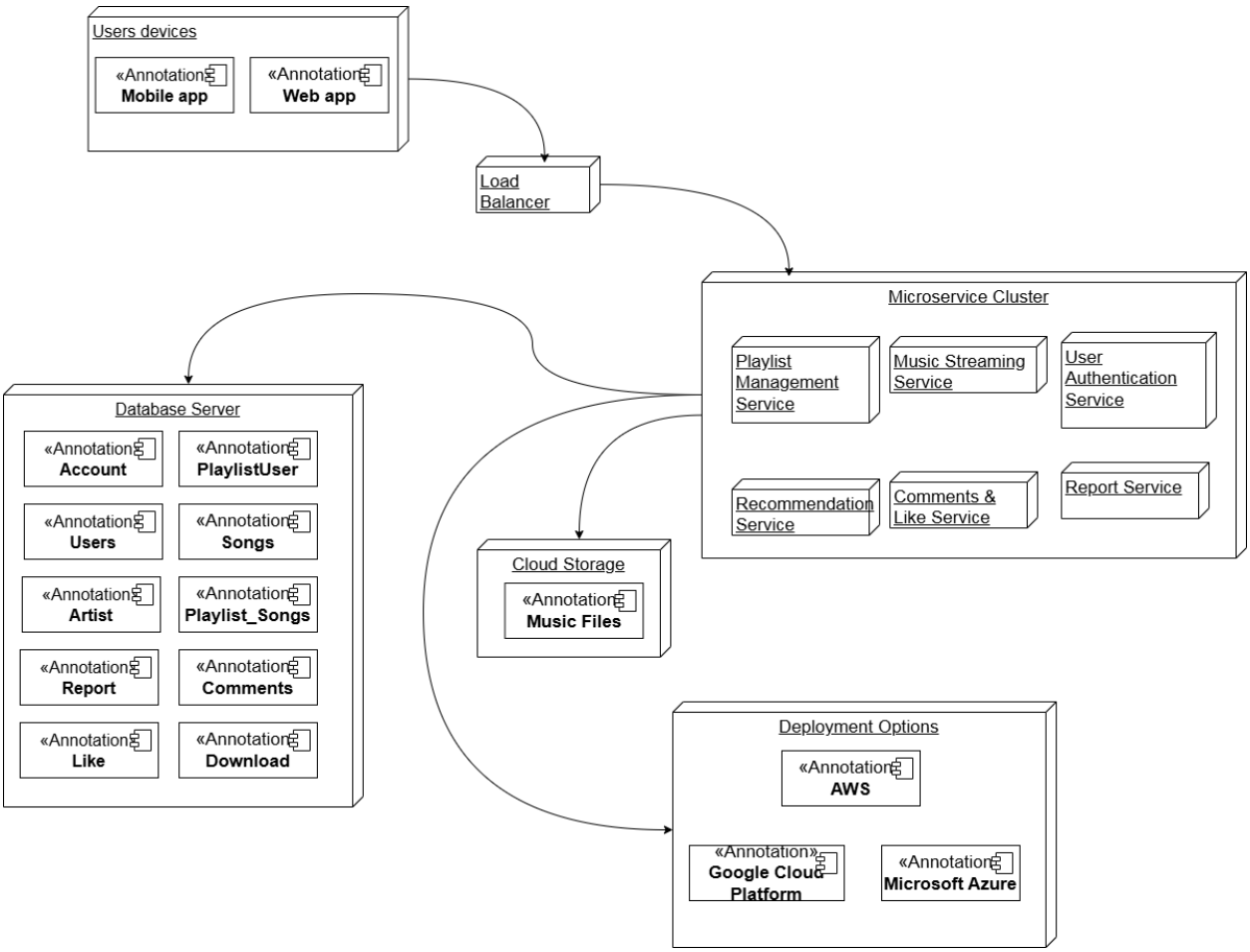
+>Hệ thống lưu trữ và bảo mật

Dữ liệu của hệ thống được lưu trữ trong các cơ sở dữ liệu chuyên biệt cho từng Microservice, gồm Music DB, Playlist DB, Comment DB, và User DB. Điều này giúp đảm bảo tính độc lập của từng dịch vụ, dễ mở rộng và bảo trì.

Bên cạnh đó, Auth Module đóng vai trò xác thực người dùng, giúp đảm bảo chỉ những người dùng hợp lệ mới có thể truy cập vào các chức năng nhất định của hệ thống.

4.1.4.2 Mô hình Allocation views

Deployment view (QA)



- **Mục tiêu:** Mô tả việc phân bổ các thành phần phần mềm lên phần cứng và tài nguyên hệ thống.
- **Phạm vi áp dụng:**

    - Mô tả cách các máy chủ (server), cơ sở dữ liệu, và các dịch vụ được triển khai trong hạ tầng vật lý hoặc đám mây.
- **Biểu diễn:** Deployment Diagram (Sơ đồ triển khai).
- **Chi tiết cho hệ thống quản lý hiệu thu;ốc:**
    - **Client Devices (Web & Mobile):** Người dùng truy cập hệ thống qua trình duyệt hoặc ứng dụng di động.
    - **Web Server:** Chạy giao diện người dùng và API. Có thể triển khai khai trên: IIS, Nginx, hoặc Docker.
    - **Application Server:** Chứa logic nghiệp vụ và xử lý các yêu cầu từ giao diện người dùng. Có thể triển khai trên: Windows Server, Linux Server, hoặc Kubernetes.
    - **Database Server:** Lưu trữ dữ liệu các bài nhạc. Có thể sử dụng máy chủ vật lý, host vps hoặc dịch vụ cloud như Azure SQL, Amazon RDS.

Mô hình allocation view tập trung vào việc ánh xạ các component của phần mềm vào phần cứng hoặc hạ tầng cơ bản. Điều này bao gồm việc các thành phần của phần mềm được phân phối trên các máy vật lý, máy chủ hoặc dịch vụ đám mây

4.1.5 Tính nhất quán và Quy tắc liên kết (Consistency and Correspondences)

4.1.5.1 Lợi thế từ tính nhất quán trong kiến trúc

+>Đảm bảo hệ thống dễ bảo trì và mở rộng

Khi hệ thống tuân theo một kiến trúc nhất quán, việc bảo trì, cập nhật hoặc mở rộng các thành phần trở nên dễ dàng hơn.

Trong kiến trúc Microservices của SoundShare, mỗi dịch vụ được thiết kế theo cùng một chuẩn REST API, giúp các nhóm phát triển có thể tích hợp hoặc mở rộng mà không ảnh hưởng đến toàn bộ hệ thống.

+>Giảm thiểu lỗi và đảm bảo tính ổn định

Nếu hệ thống không có sự nhất quán, các dịch vụ có thể gặp xung đột khi giao tiếp với nhau, dẫn đến lỗi hoặc hành vi không mong muốn.

Việc sử dụng các giao thức và chuẩn thiết kế nhất quán như giao diện API đồng nhất (OpenAPI), chuẩn metadata nhạc (ID3v2, JSON-LD) giúp giảm thiểu rủi ro này.

+>Cải thiện hiệu suất và khả năng mở rộng

Khi hệ thống có sự nhất quán trong cách thiết kế cấu trúc dữ liệu, API, cơ chế giao tiếp, nó giúp cải thiện hiệu suất xử lý và khả năng mở rộng linh hoạt.

Ví dụ: Tất cả các Microservices trong SoundShare đều sử dụng cơ chế caching đồng nhất, giúp giảm tải cho cơ sở dữ liệu và tăng tốc độ phản hồi.

+> Đảm bảo tuân thủ bảo mật và quy định pháp lý

Một hệ thống nhất quán giúp quản lý bảo mật tốt hơn, đảm bảo tuân thủ các tiêu chuẩn như GDPR, Nghị định 53/2022/NĐ-CP về bảo vệ dữ liệu.

Trong SoundShare, tất cả các dịch vụ xử lý dữ liệu người dùng đều tuân theo cơ chế mã hóa AES-256 và giao thức truyền dữ liệu an toàn TLS 1.3.

4.1.5.2 Liên kết trong hồ sơ kiến trúc (Correspondences in the AD).

Hồ sơ kiến trúc (**Architecture Description - AD**) là tài liệu quan trọng giúp đảm bảo rằng tất cả các thành phần của hệ thống được thiết kế và triển khai một cách thống nhất, không có xung đột giữa các khía cạnh kiến trúc khác nhau. **Liên kết trong hồ sơ kiến trúc (Correspondences)** giúp duy trì tính nhất quán, đồng bộ và dễ dàng mở rộng hệ thống trong tương lai.

Một hệ thống phần mềm phức tạp thường bao gồm nhiều quan điểm kiến trúc để mô tả các khía cạnh khác nhau của hệ thống. Các quan điểm chính trong hồ sơ kiến trúc SoundShare bao gồm:

- **Module View (Quan điểm mô-đun):** Xác định các thành phần logic chính của hệ thống và cách chúng được tổ chức.
- **Component & Connector View (Quan điểm thành phần & kết nối):** Mô tả cách các thành phần tương tác với nhau thông qua API.
- **Deployment View (Quan điểm triển khai):** Mô tả cách hệ thống được triển khai trên cơ sở hạ tầng thực tế.
- **Runtime View (Quan điểm hoạt động):** Mô tả cách hệ thống xử lý yêu cầu trong thời gian thực.
- **Security View (Quan điểm bảo mật):** Xác định các biện pháp bảo vệ hệ thống khỏi các mối đe dọa an ninh.

4.1.5.3 Quy tắc liên kết (Correspondence rules)

Các **quy tắc liên kết (Correspondence Rules)** giúp duy trì sự nhất quán trong kiến trúc hệ thống bằng cách đảm bảo rằng tất cả các phần tử trong hệ thống có sự liên kết hợp lý và tuân thủ các quy chuẩn chung.

+> Quy Tắc Đồng Nhất API

Tất cả các Microservices phải tuân theo chuẩn API RESTful.

Mọi API phải có tài liệu OpenAPI/Swagger.

Ví dụ: API của Music Service, Playlist Service, Comment Service, User Service đều sử dụng OpenAPI.

+>Quy Tắc Kiểm Soát Truy Cập

Người dùng chỉ có thể chỉnh sửa nội dung do chính họ tải lên.

Quản trị viên có quyền kiểm duyệt nội dung nhưng không thể chỉnh sửa dữ liệu cá nhân của người dùng khác.+

+>Quy Tắc Quản Lý Dữ Liệu

Dữ liệu nhạc và metadata phải tuân theo chuẩn ID3v2, JSON-LD.

Tất cả dữ liệu nhạy cảm phải mã hóa trước khi lưu trữ.

+> Quy Tắc Kiểm Tra Và Giám Sát

Hệ thống phải có cơ chế logging để ghi nhận mọi thao tác quan trọng.

Mọi thay đổi quan trọng (cập nhật dữ liệu, thay đổi quyền truy cập) đều phải ghi lại trong nhật ký hệ thống.

Kiểm tra bảo mật phải được thực hiện định kỳ.

4.2 Lý do lựa chọn kiến trúc phần mềm

Phân tích các lý do dẫn đến kiến trúc phần mềm (trong mục 3.1) được chọn (và kiến trúc khác không được chọn)

Kiến trúc microservices được chọn thay vì kiến trúc monolithic (nguyên khối) hoặc SOA (Service-Oriented Architecture) vì những lợi ích sau:

+>Khả năng mở rộng (Scalability)

Hệ thống cần xử lý hàng chục nghìn yêu cầu đồng thời (NFR04, NFR05). Microservices cho phép mở rộng từng dịch vụ riêng biệt, giúp tối ưu tài nguyên mà không làm quá tải toàn bộ hệ thống.

+>Hiệu suất và độ trễ thấp (Performance & Low Latency)

Yêu cầu phản hồi nhanh (NFR01, NFR02, NFR03). Microservices có thể triển khai caching và tối ưu hoá từng dịch vụ để đảm bảo hiệu suất.

+>Khả năng bảo trì và cập nhật dễ dàng (Maintainability)

Microservices có kiến trúc module hoá (NFR23), giúp cập nhật từng phần mềm mà không ảnh hưởng toàn bộ hệ thống. Hệ thống cũng có thể triển khai các bản cập nhật không cần downtime (NFR24).

+>Độ tin cậy cao (Reliability)

Microservices giúp cách ly lỗi: nếu một dịch vụ gặp sự cố, các dịch vụ khác vẫn tiếp tục hoạt động (NFR06, NFR07, NFR08). Điều này giúp hệ thống ổn định hơn so với kiến trúc monolithic.

+>Bảo mật (Security)

Hệ thống yêu cầu kiểm soát truy cập theo vai trò (RBAC - NFR16), mã hóa dữ liệu (NFR17, NFR18) và giám sát bảo mật thường xuyên (NFR19). Microservices giúp giới hạn phạm vi ảnh hưởng khi xảy ra lỗ hổng bảo mật.

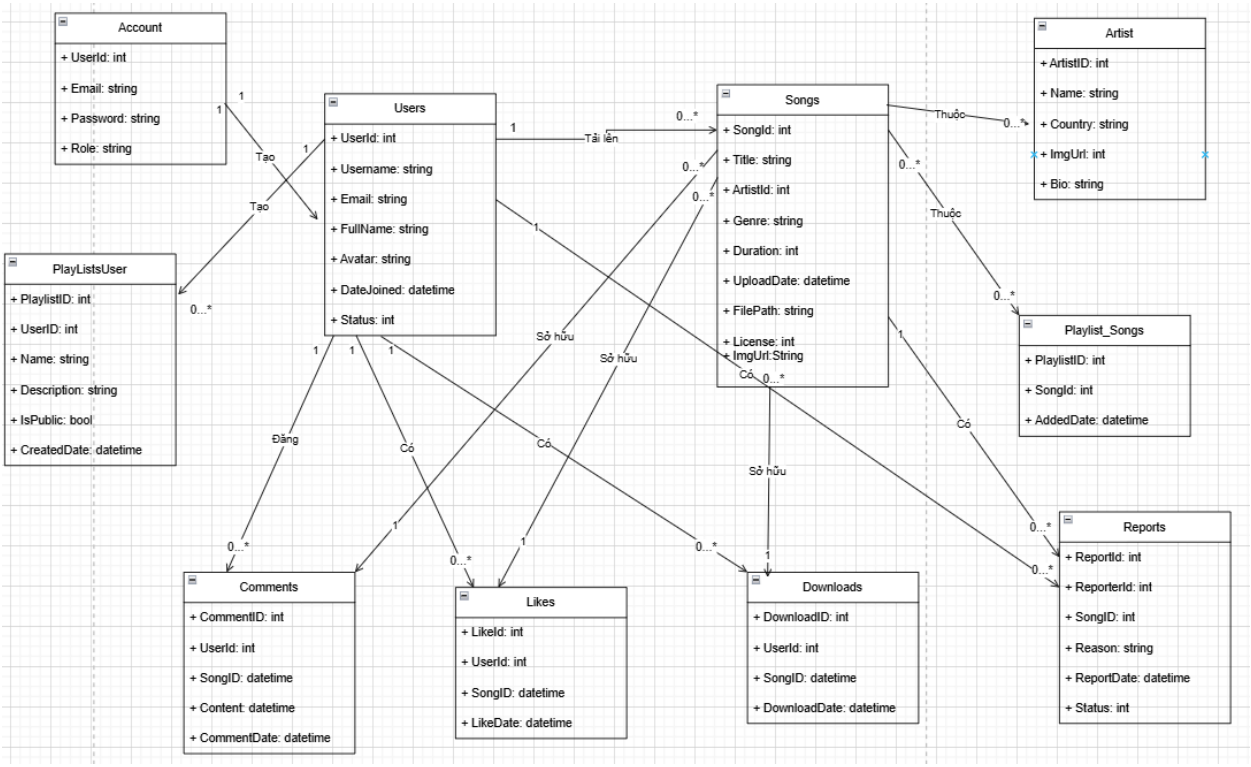
+>Khả năng tích hợp với các nền tảng khác (Interoperability)

Hệ thống cần hỗ trợ loa thông minh, TV, CarPlay (NFR20), chia sẻ mạng xã hội (NFR22) và chuẩn metadata (NFR21). Microservices giúp triển khai API dễ dàng cho từng dịch vụ mà không ảnh hưởng đến toàn bộ hệ thống.

+>Tuân thủ pháp luật và dễ dàng mở rộng

Hệ thống cần ghi log theo dõi (NFR36), kiểm soát bản quyền nhạc (NFR37) và tuân thủ quy định bảo vệ dữ liệu (NFR35). Microservices giúp mỗi dịch vụ có thể tuân thủ từng quy định cụ thể mà không ảnh hưởng hệ thống chung.

5. CSDL đã thiết kế:



5. Đặc tả bảng dữ liệu

5.1. Bảng: Account

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
UserId	int	PRIMARY KEY, AUTO_INCREMENT	Định danh duy nhất của người dùng
Email	string	NOT NULL, UNIQUE	Địa chỉ email dùng để đăng nhập
Password	string	NOT NULL	Mật khẩu được mã hóa của người dùng
Role	string	NOT NULL, DEFAULT 'user'	Vai trò của người dùng (admin, user, v.v.)

5.2 Bảng: PlayListsUser

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
PlaylistID	int	PRIMARY KEY, AUTO_INCREMENT	Định danh duy nhất của danh sách phát
UserID	int	FOREIGN KEY REFERENCES Account(UserId)	Người dùng sở hữu danh sách phát
Name	string	NOT NULL	Tên của danh sách phát
Description	string	NULL	Mô tả danh sách phát
IsPublic	bool	NOT NULL, DEFAULT 1	Trạng thái công khai (true = công khai, false = riêng tư)
CreatedDate	datetime	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Ngày tạo danh sách phát

5.3 Bảng: Users

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
UserId	int	PRIMARY KEY, AUTO_INCREMENT	Định danh duy nhất của người dùng
Username	string	NOT NULL, UNIQUE	Tên đăng nhập của người dùng
Email	string	NOT NULL, UNIQUE	Địa chỉ email của người dùng

FullName	string	NULL	Họ và tên đầy đủ
Avatar	string	NULL	Đường dẫn đến ảnh đại diện
DateJoined	datetime	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Ngày tham gia hệ thống
Status	int	NOT NULL, DEFAULT 1	Trạng thái tài khoản (1 = hoạt động, 0 = bị khóa)

5.4 Bảng: Songs

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
SongId	int	PRIMARY KEY, AUTO_INCREMENT	Định danh duy nhất của bài hát
Title	string	NOT NULL	Tên bài hát
ArtistId	int	FOREIGN KEY REFERENCES Users(UserId)	ID nghệ sĩ hoặc người tải lên
Genre	string	NULL	Thể loại nhạc
Duration	int	NOT NULL	Thời lượng bài hát (tính bằng giây)
UploadDate	datetime	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Ngày tải lên
FilePath	string	NOT NULL	Đường dẫn đến file nhạc

License	int	NOT NULL, DEFAULT 1	Loại bản quyền (1 = miễn phí, 2 = có phí, v.v.)
ImgUrl	string	NULL	Ảnh bìa bài hát

5.5 Bảng: Artist

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
ArtistID	int	PRIMARY KEY, AUTO_INCREMENT	Định danh duy nhất của nghệ sĩ
Name	string	NOT NULL	Tên nghệ sĩ
Country	string	NULL	Quốc gia của nghệ sĩ
ImgUrl	string	NULL	Đường dẫn đến ảnh nghệ sĩ
Bio	string	NULL	Tiểu sử hoặc mô tả nghệ sĩ

Bảng: Playlist_Songs

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
PlaylistID	int	FOREIGN KEY REFERENCES Playlists(PlaylistID)	Định danh danh sách phát
SongID	int	FOREIGN KEY REFERENCES Songs(SongID)	Định danh bài hát
AddedDate	datetime	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Ngày bài hát được thêm vào danh sách phát

5.6 Bảng: Reports

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
ReportId	int	PRIMARY KEY, AUTO_INCREMENT	Định danh duy nhất của báo cáo
ReporterId	int	FOREIGN KEY REFERENCES Users(UserId)	ID của người báo cáo
SongID	int	FOREIGN KEY REFERENCES Songs(SongID)	ID bài hát bị báo cáo
Reason	string	NOT NULL	Lý do báo cáo
ReportDate	datetime	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Ngày báo cáo được gửi
Status	int	NOT NULL, DEFAULT 0	Trạng thái báo cáo (0 = Chờ xử lý, 1 = Đã giải quyết)

5.7 Comments

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
CommentID	int	PRIMARY KEY, AUTO_INCREMENT	Định danh duy nhất của bình luận
UserId	int	FOREIGN KEY REFERENCES Users(UserId)	Người dùng thực hiện bình luận
SongID	int	FOREIGN KEY REFERENCES Songs(SongID)	Bài hát được bình luận
Content	string	NOT NULL	Nội dung bình luận

CommentDate	datetime	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Ngày bình luận được đăng

5.8 Likes

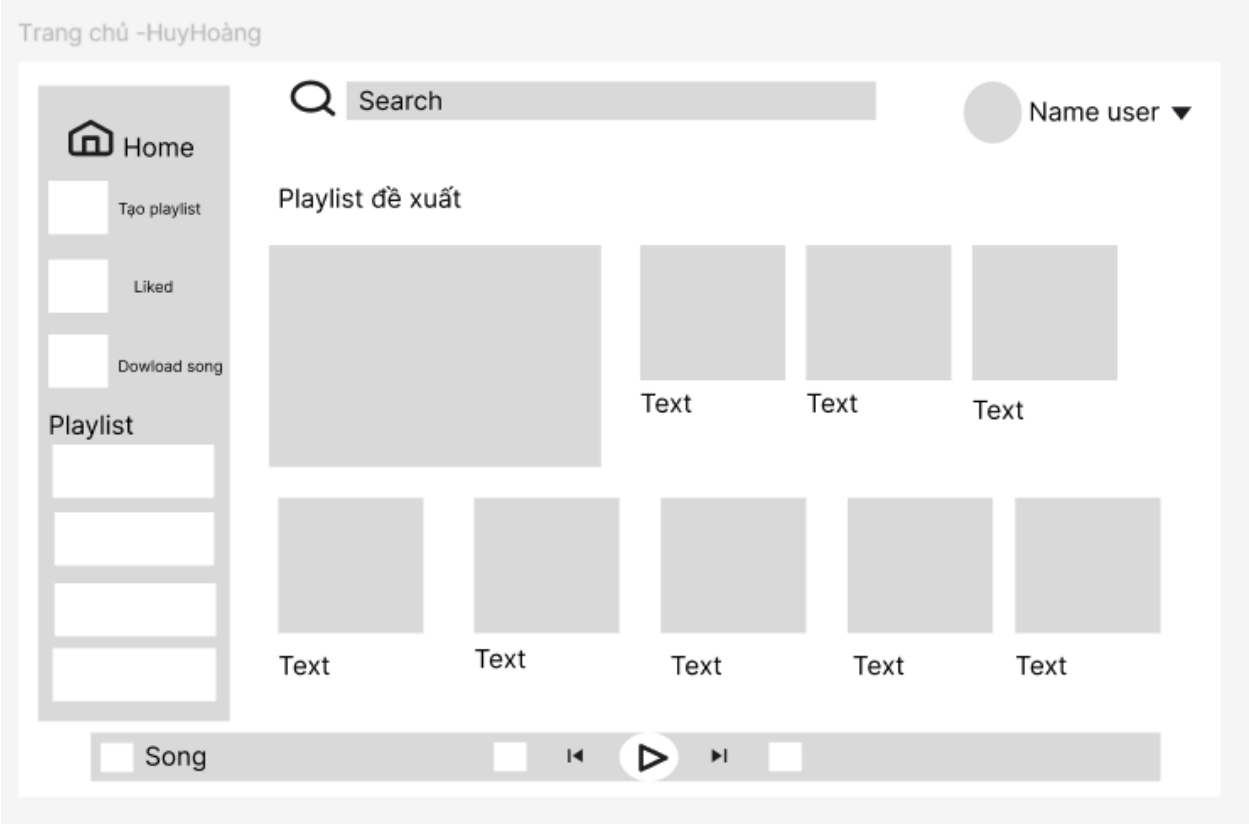
Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
LikeId	int	PRIMARY KEY, AUTO_INCREMENT	Định danh duy nhất của lượt thích
UserId	int	FOREIGN KEY REFERENCES Users(UserId)	Người dùng thực hiện lượt thích
SongID	int	FOREIGN KEY REFERENCES Songs(SongID)	Bài hát được thích
LikeDate	datetime	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Ngày bài hát được thích

5.9Downloads

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
DownloadID	int	PRIMARY KEY, AUTO_INCREMENT	Định danh duy nhất của lượt tải xuống
UserId	int	FOREIGN KEY REFERENCES Users(UserId)	Người dùng thực hiện tải xuống
SongID	int	FOREIGN KEY REFERENCES Songs(SongID)	Bài hát được tải xuống
DownloadDate	datetime	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Ngày bài hát được tải xuống

6.Giao diện:

6.1 Đặc tả giao diện trang chủ



+>Thanh điều hướng bên trái (Sidebar)

- Biểu tượng Home: Dẫn về trang chủ.
- Các mục điều hướng:

-Tạo playlist: Cho phép người dùng tạo danh sách phát mới.

-Liked: Danh sách các bài hát đã thích.

-Download song: Truy cập các bài hát đã tải xuống.

- Mục Playlist: Hiển thị danh sách các playlist của người dùng.

+>Thanh tìm kiếm & Hồ sơ người dùng

- Thanh tìm kiếm: Một ô tìm kiếm để nhập từ khóa tìm bài hát, playlist hoặc nghệ sĩ.
- Ảnh đại diện và tên người dùng: Hiển thị ảnh đại diện và tên tài khoản người dùng (menu dropdown để cài đặt tài khoản).

+> Mục "Playlist đề xuất"

- Gồm nhiều khung hình chữ nhật đại diện cho các playlist đề xuất.
- Mỗi playlist có ảnh đại diện và tiêu đề bên dưới.

+>Thanh điều khiển nhạc (Music Player)

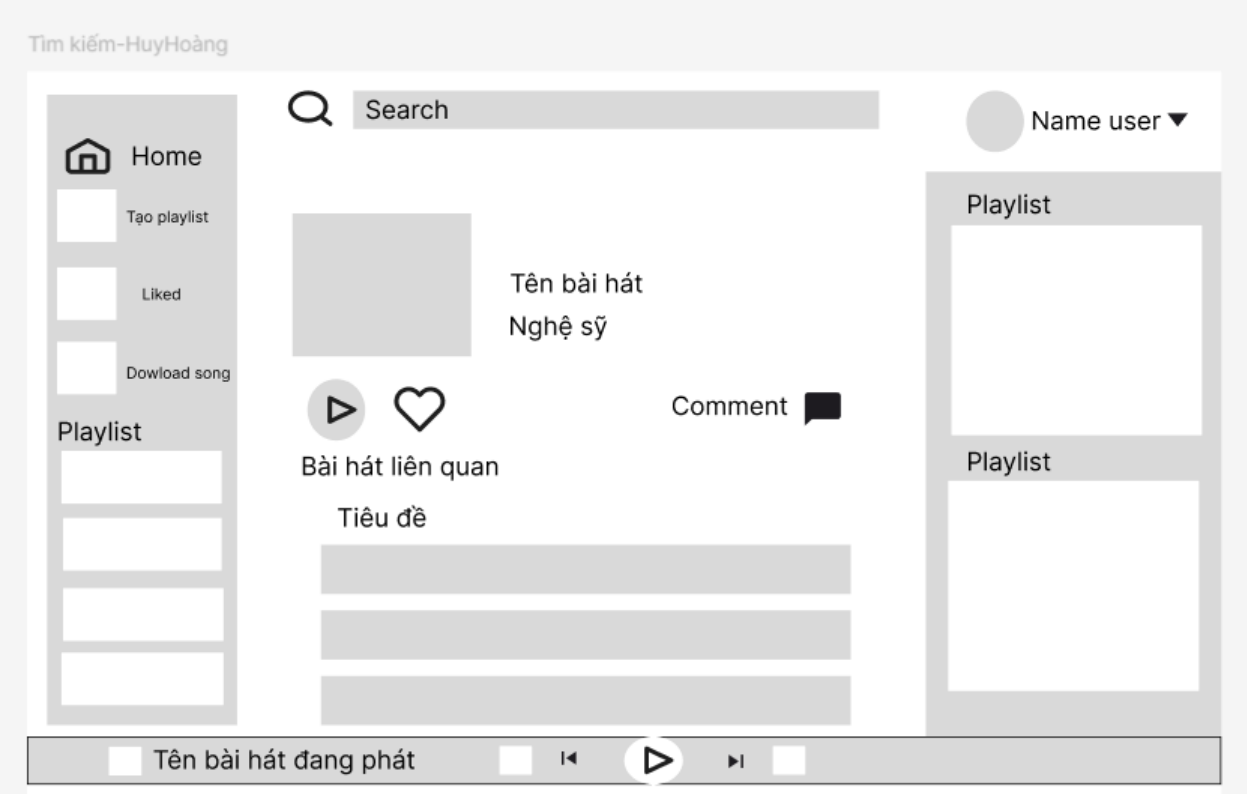
- Thanh phát nhạc dưới cùng của màn hình gồm:

-Nút Play/Pause ở giữa.

-Nút Next/Previous để chuyển bài.

-Thanh tiến trình phát nhạc để theo dõi và điều chỉnh thời gian bài hát.

6.2 Đặc tả Giao diện chi tiết nghe nhạc



Thanh tìm kiếm (Search)

- Ở đầu trang có một thanh tìm kiếm để người dùng nhập từ khóa tìm kiếm bài hát.
- Có biểu tượng kính lúp bên trái thể hiện chức năng tìm kiếm.

Sidebar bên trái (Home & Playlist)

Home

- Chứa các tùy chọn điều hướng chính:

- **Tạo playlist**
- **Liked** (Danh sách bài hát yêu thích)
- **Download song** (Tải nhạc xuống)

Playlist

- Hiển thị danh sách các playlist đã tạo hoặc lưu.

Khu vực hiển thị bài hát

- Ảnh bìa của bài hát.
- **Tên bài hát** và **tên nghệ sĩ** được hiển thị bên phải ảnh bìa.
- Các nút chức năng chính:
  - **Nút Play** (Phát nhạc)
  - **Nút yêu thích** (Biểu tượng trái tim)
  - **Nút bình luận** (Comment - có biểu tượng khung chat)

Danh sách bài hát liên quan

- Bên dưới bài hát chính có một danh sách các bài hát liên quan.
- Mỗi bài hát liên quan có tiêu đề và sắp xếp dưới dạng danh sách.

Sidebar bên phải (Playlist)

- Chứa danh sách các **playlist khác** của người dùng.

Thanh điều khiển nhạc (Music Player)

Ở phía dưới cùng là thanh điều khiển nhạc:

Hiển thị **tên bài hát đang phát**.

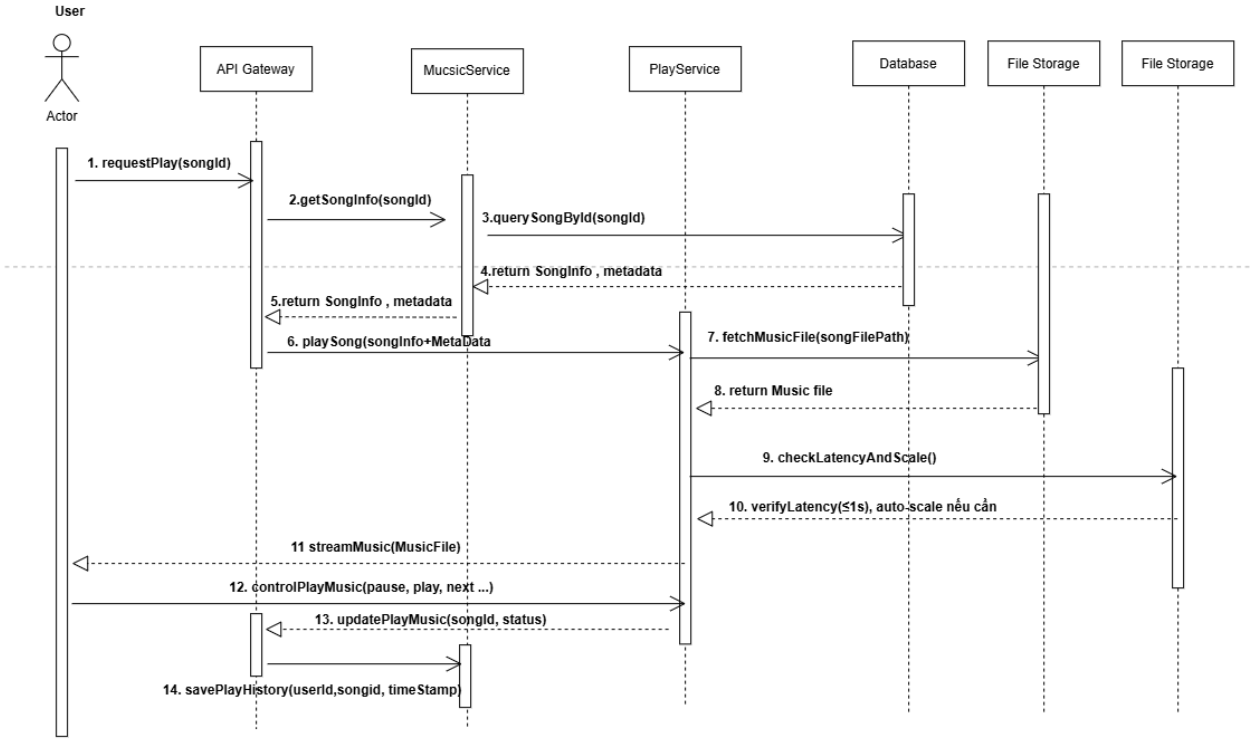
Các nút điều khiển:

- **Lùi bài hát**
- **Phát/Tạm dừng**
- **Tiến bài hát**

.

7.Sơ đồ tuần tự chức năng phát nhạc trực tuyến

Sơ đồ tuần tự



7.1 Mô tả:

-Sơ đồ **sequence diagram** này mô tả **quy trình phát nhạc trực tuyến**, bao gồm các thành phần chính và cách chúng tương tác với nhau để phục vụ yêu cầu phát nhạc và điều khiển nhạc của người dùng.

Hệ thống được thiết kế theo mô hình **microservices**, trong đó mỗi dịch vụ đảm nhiệm một vai trò cụ thể, giúp hệ thống **dễ mở rộng, tối ưu hiệu suất, và đảm bảo trải nghiệm người dùng mượt mà**.

+>User (Người dùng)

Người dùng tương tác với hệ thống thông qua giao diện (web/app) để **yêu cầu phát nhạc**, **điều khiển nhạc** (pause, play, next, prev), và **lưu lịch sử nghe nhạc**.

+>API Gateway

-Trung gian nhận request từ **User** và điều phối đến các dịch vụ phù hợp.

-Đảm bảo hệ thống có thể mở rộng và bảo mật bằng cách kiểm soát request.

+>Music Service

-Xử lý thông tin bài hát.

-Lấy **metadata** từ **Database** để cung cấp thông tin chi tiết cho trình phát nhạc.

-Ghi nhận lịch sử nghe nhạc để hỗ trợ gợi ý bài hát.

+>Player Service

-Xử lý logic phát nhạc.

-Truy xuất file nhạc từ **File Storage** và stream đến **User**.

-Theo dõi hiệu suất hệ thống (latency, auto-scaling).

+> Database

-Lưu **thông tin bài hát** (song ID, tên bài hát, nghệ sĩ, thể loại, thời lượng, file path...).

-Lưu **lịch sử nghe nhạc** của người dùng.

+>File Storage

-Lưu trữ **file nhạc thực tế (.mp3, .wav...)**.

-Cho phép **truy xuất nhanh các file nhạc** khi cần phát.

+>Monitor & Scaling

-Kiểm tra độ trễ (**latency**) của hệ thống.

-Kích hoạt **auto-scale** khi lượng người nghe tăng cao để đảm bảo tốc độ phản hồi nhanh chóng.

7.2 Luồng xử lý phát nhạc

Bước 1: Người dùng yêu cầu phát nhạc

-User gửi yêu cầu requestPlay(songId) đến API Gateway.

-API Gateway gọi getSongInfo(songId) từ **Music Service** để lấy thông tin bài hát.

Bước 2: Lấy thông tin bài hát từ Database

-Music Service truy vấn Database bằng querySongById(songId).

-Database trả về SongInfo + Metadata (bao gồm: tên bài hát, nghệ sĩ, thời lượng, thể loại, file path...).

-Music Service gửi lại SongInfo + Metadata cho **API Gateway**.

Bước 3: Player Service xử lý phát nhạc

-API Gateway gửi playSong(SongInfo + Metadata) đến Player Service.

-Player Service gọi fetchMusicFile(songFilePath) từ **File Storage**.

-File Storage trả về file nhạc.

Bước 4: Kiểm tra hiệu suất hệ thống

-Player Service gửi yêu cầu checkLatencyAndScale() đến Monitor để kiểm tra độ trễ.

-Monitor xác nhận verifyLatency(≤1s), auto-scale nếu cần.

Bước 5: Stream nhạc đến người dùng

-Player Service gửi streamMusic(MusicFile) đến **User**, hoàn tất quá trình phát nhạc

7.3 Các unit trong sơ đồ tuần tự

+> **Unit: requestPlay(songId)**

- Mô tả: Người dùng yêu cầu phát bài hát khi bấm nút play của bài hát
- Input:id của bài hát đang được lựa chọn phát songId: String (VD: "12345")
- Output:

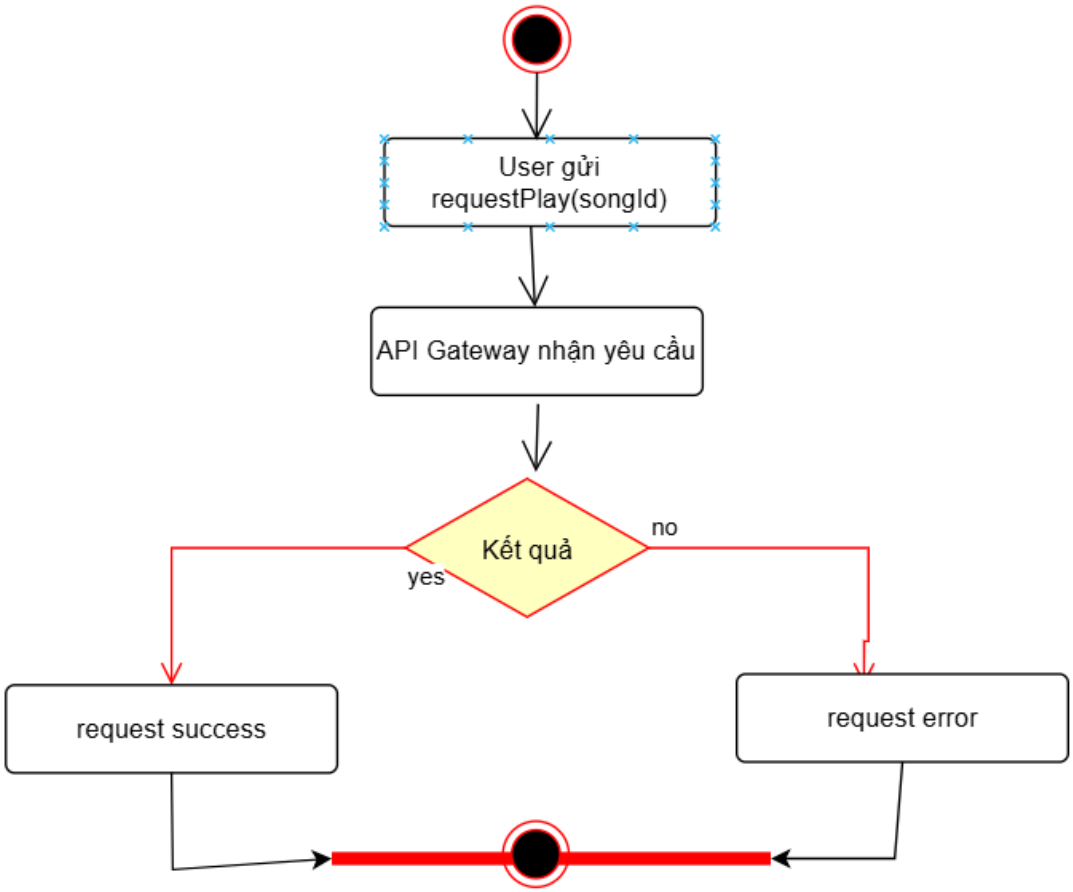
Trả về xác nhận yêu cầu phát nhạc được gửi thành công.

Process:

-Nhận request từ người dùng.

-Gửi request getSongInfo(songId) đến Music Service.

Sơ đồ activity:



+>Unit getSongInfo(songId)

Mô tả: Lấy thông tin bài hát từ Database.

Input: id của bài hát phát -songId: String

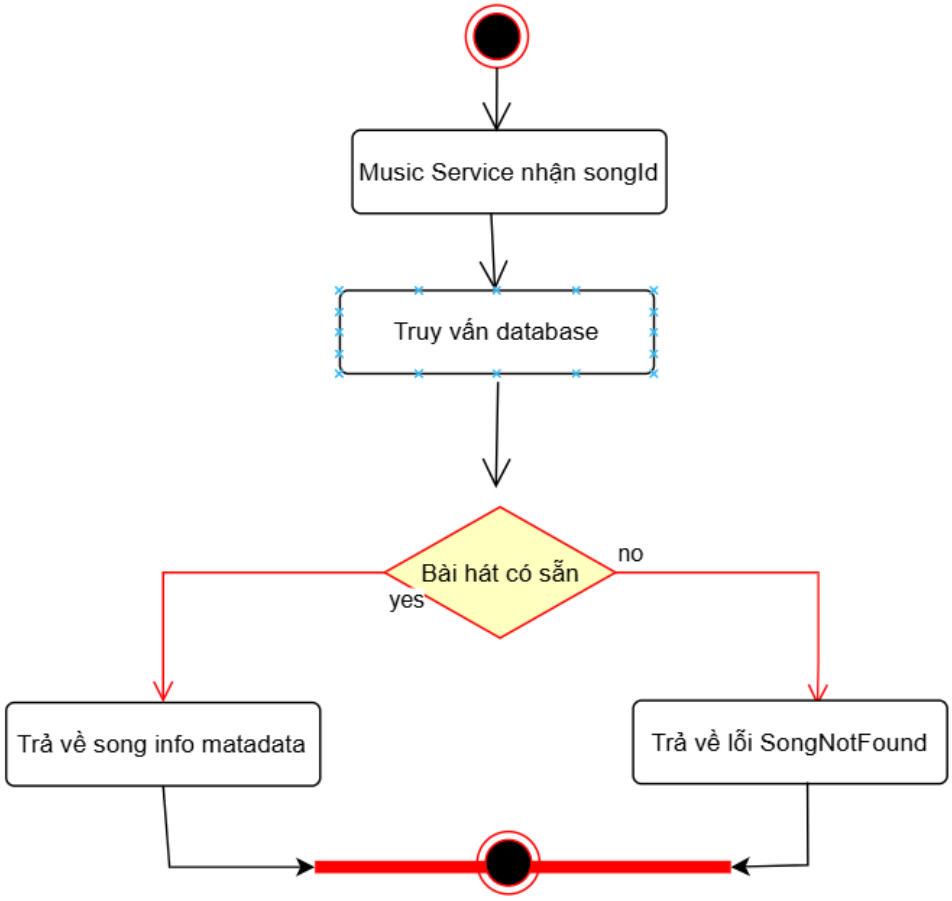
Output: trả về thông tin bài hát

{
"songId": "12345",
"title": "Shape of You",
"artist": "Ed Sheeran",
"duration": 234,
"genre": "Pop",
"filePath": "/music/ed_sheeran/shape_of_you.mp3"
}

Process

- Nhận songId từ API Gateway.
- Truy vấn Database querySongById(songId).
- Nếu bài hát tồn tại, trả về SongInfo + Metadata.
- Nếu không, trả về lỗi SongNotFoundException.

Sơ đồ activity:



+> **Unit: fetchMusicFile(songFilePath):**

Mô tả: Truy xuất file nhạc từ File Storage.

Input: Đường dẫn file của bài hát "filePath":"/music/ed_sheeran/shape_of_you.mp3"

Output:

- File nhạc .mp3 hoặc .wav
- Nếu file không tồn tại:thông báo không tìm thấy đường dẫn bài hát

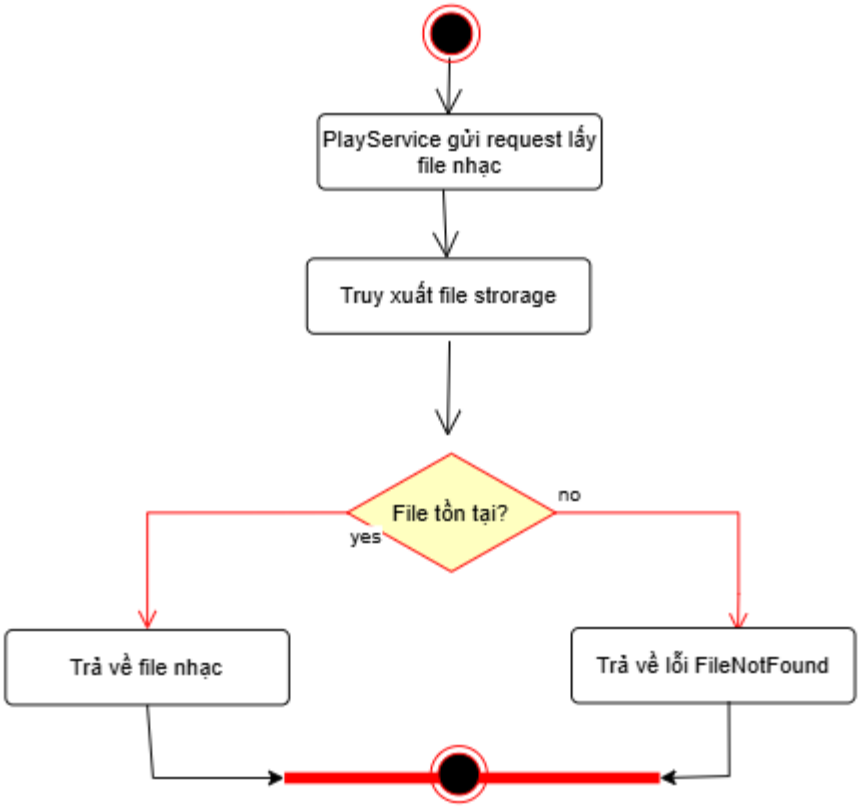
Process:

Tìm kiếm file trong File Storage.

Kiểm tra file có tồn tại không.

- Nếu có, gửi file về PlayerService.
- Nếu không, trả lỗi "File not found"

Sơ đồ activity:



+>**Unit: checkLatencyAndScale()**

Mô tả: Kiểm tra độ trễ hệ thống, đảm bảo phát nhạc nhanh độ trễ phát nhạc không quá 1s

Input: thời gian bắt đầu phát nhạc - startTime: Timestamp

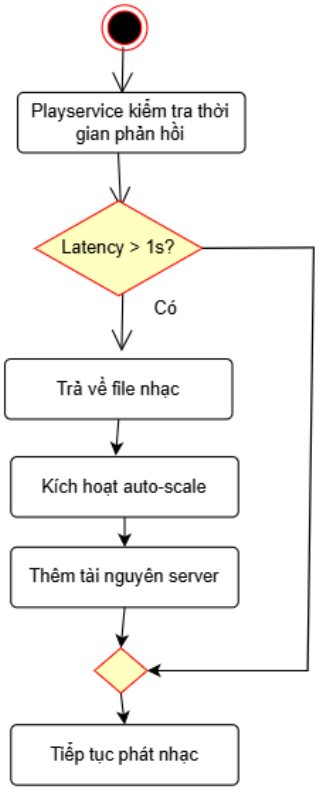
Output:

- Nếu latency ≤ 1s, không thay đổi.
- Nếu latency > 1s, kích hoạt AutoScaling.

Process:

1. Play service lấy thời gian bắt đầu phát nhạc.
2. So sánh với thời gian phản hồi thực tế.
3. Nếu quá 1 giây, tăng tài nguyên (AutoScale).

Sơ đồ activity:



+>Unit: streamMusic(MusicFile)

Mô tả: PlayerService gửi file nhạc về cho người dùng để phát nhạc.

Input: file nhạc MusicFile: Binary (.mp3, .wav)

Output:

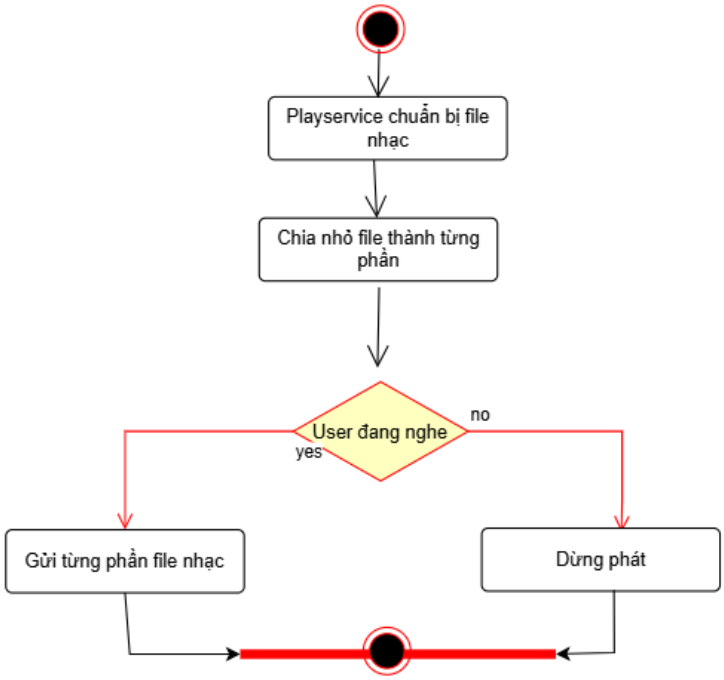
-Người dùng nhận và phát nhạc trực tuyến.

Process:

1. PlayerService gửi file nhạc về cho người dùng để phát nhạc.
2. Kiểm tra kết nối mạng của user.

3. Nếu kết nối ổn định, gửi từng chunk dữ liệu nhạc.
4. Phát nhạc

Sơ đồ activity:



+>**Unit: controlPlayMusic(action)**

Mô tả: Người dùng điều khiển nhạc (Pause, Play, Next, Prev).

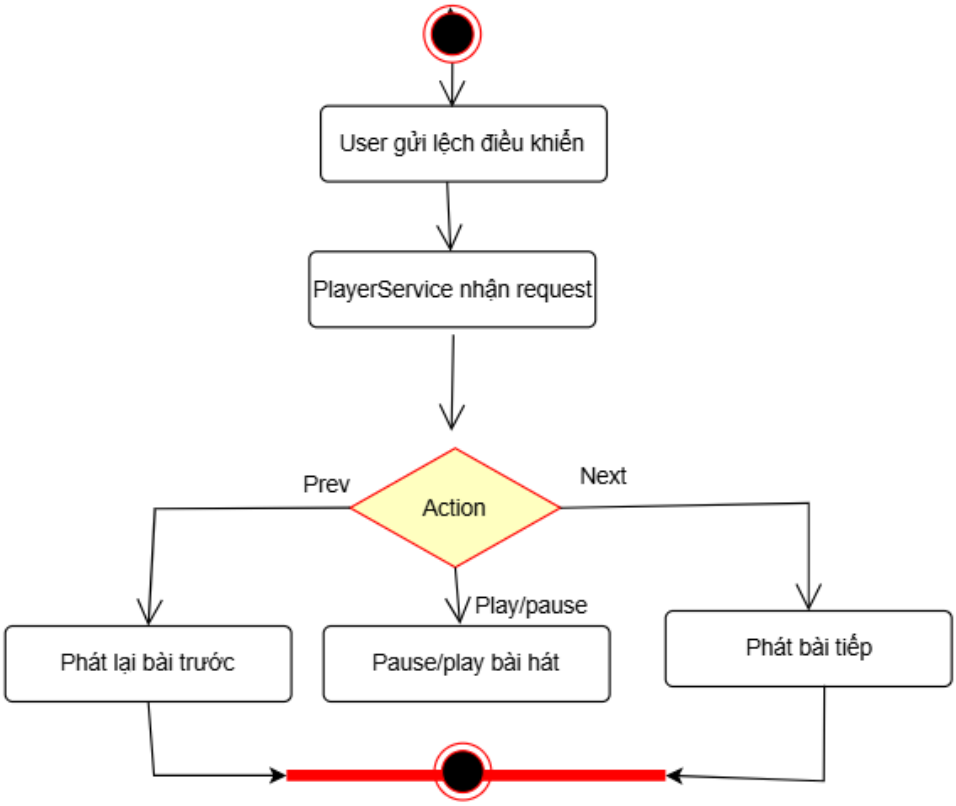
Input: trạng thái hành động -action: "pause" "play" "next" "prev"

Output:

- Cập nhật trạng thái nhạc.

Proces:

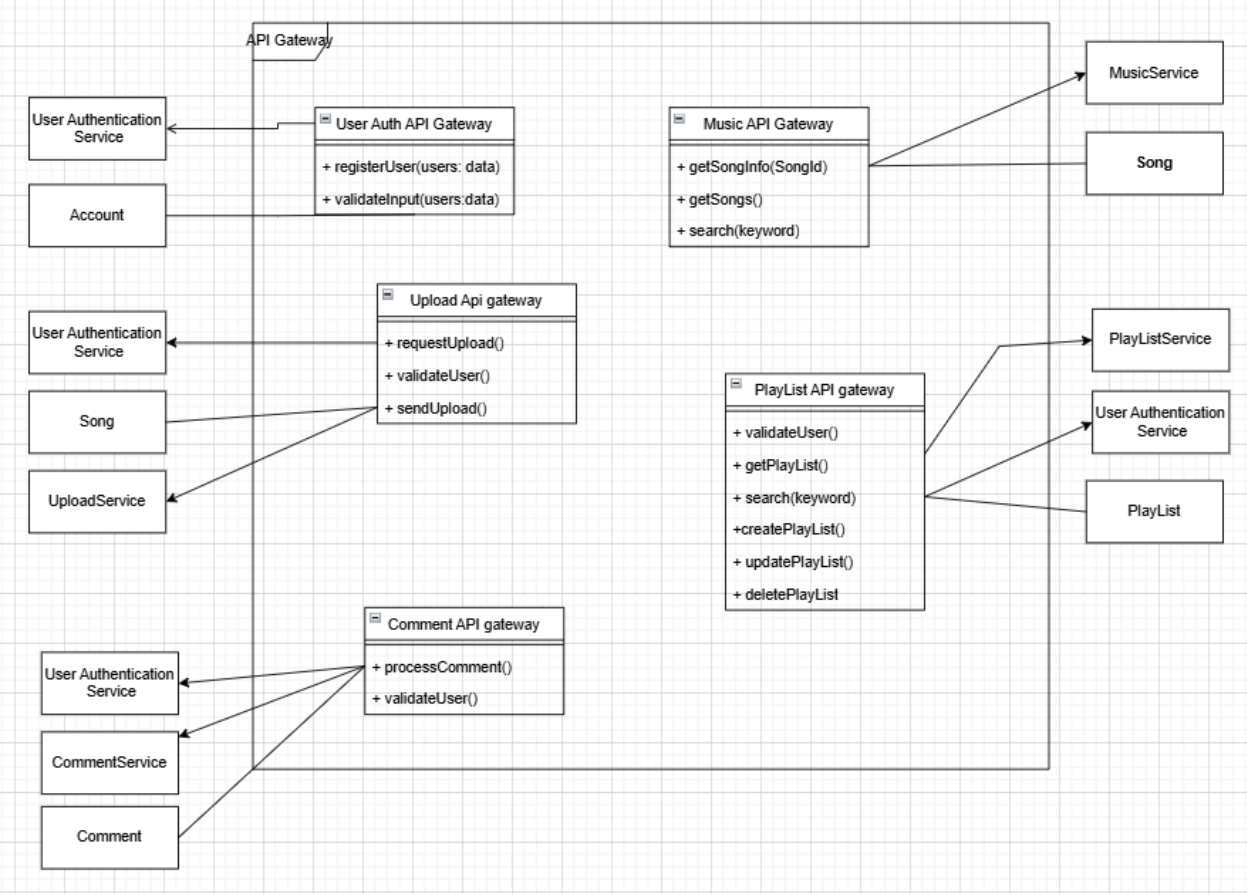
1. Nhận lệnh điều khiển từ người dùng.
2. Cập nhật trạng thái phát nhạc trong **PlayerService**.



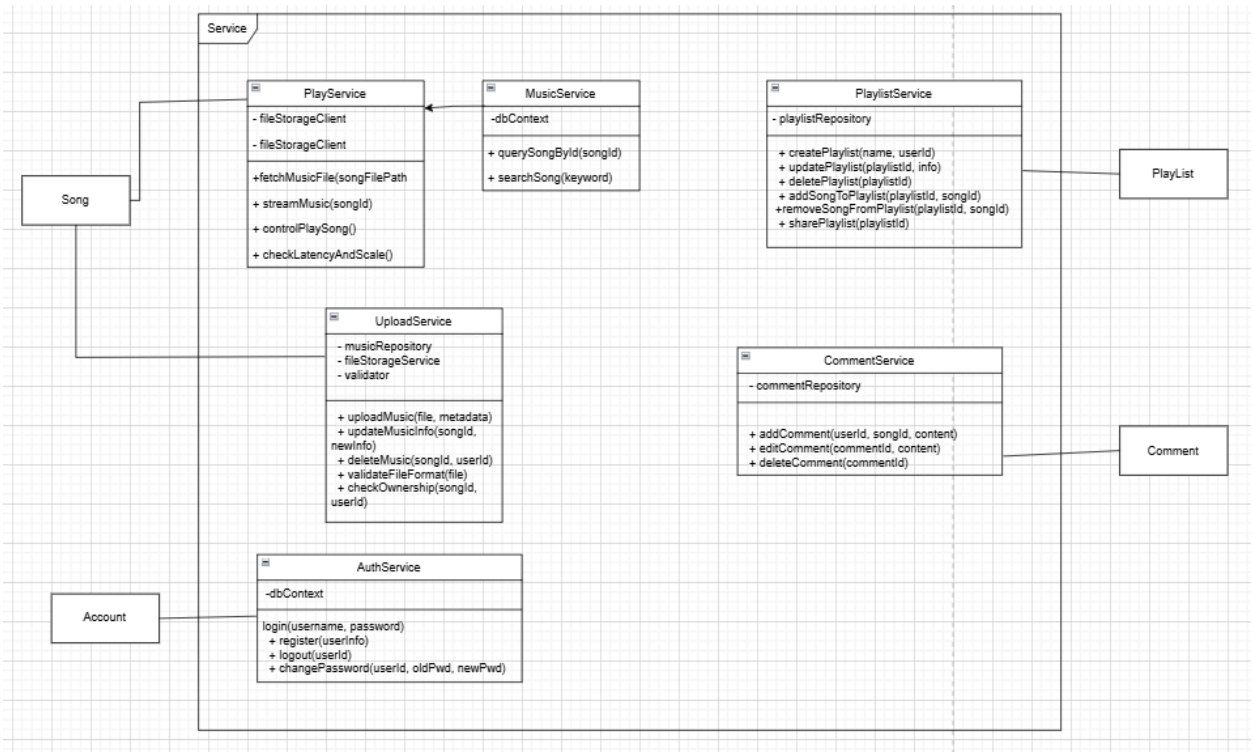
Sơ đồ activity:

8.Các sơ đồ class diagram:

-Api gateway



Service sơ đồ class:


Service
PlayService
- fileStorageClient
- fileStorageClient
+fetchMusicFile(songFilePath
+ streamMusic(songId)
+ controlPlaySong()
+ checkLatencyAndScale()
MusicService
-dbContext
+ querySongById(songId)
+ searchSong(keyword)
PlaylistService
- playlistRepository
+ createPlaylist(name, userId)
+ updatePlaylist(playlistId, info)
+ deletePlaylist(playlistId)
+ addSongToPlaylist(playlistId, songId)
+removeSongFromPlaylist(playlistId, songId)
+ sharePlaylist(playlistId)
PlayList
Song
UploadService
- musicRepository
- fileStorageService
- validator
+ uploadMusic(file, metadata)
+ updateMusicInfo(songId, newInfo)
+ deleteMusic(songId, userId)
+ validateFileFormat(file)
+ checkOwnership(songId, userId)
CommentService
- commentRepository
+ addComment(userId, songId, content)
+ editComment(commentId, content)
+ deleteComment(commentId)
Comment
AuthService
-dbContext
login(username, password)
+ register(userInfo)
+ logout(userId)
+ changePassword(userId, oldPwd, newPwd)
Account