**HỆ THỐNG BACKEND DÀNH CHO WEB LƯU TRỮ VÀ NGHE NHẠC TRỤC TUYẾN**

**1. Các yêu cầu chức năng**

***1.1 Các tác nhân***

Hệ thống bao gồm tác nhân chính Người dùng và Quản trị viên. Khách sau khi đăng kí và đăng nhập thành công vào hệ thống trở thành Người dùng, có quyền sử dụng các chức năng của hệ thống. Quản trị viên có vai trò quản trị các Người dùng.

***1.2 Các chức năng của hệ thống***

Các chức năng của Người dùng:

- Đăng nhập: Chức năng xác thực danh tính của người dùng khi sử dụng hệ thống nhằm cung cấp dữ liệu chính xác cho người sử dụng.

- Đăng kí: Khách cần được đăng kí tài khoản để thành Người dùng và có thể sử dụng hệ thống

- Thêm bài hát vào kho lưu trữ: Người dùng thêm bài hát vào kho lưu trữ.

- Xóa trong kho lưu trữ: Người dùng xóa các bài hát của mình ra khỏi kho lưu trữ.

- Tải bài hát về: Người dùng có thể tải các bài hát trong kho lưu trữ cá nhân về thiết bị đang sử dụng.

- Chỉnh sửa danh sách yêu thích: Người dùng có thể thêm, xóa bài hát ra khỏi danh sách các bài hát đã yêu thích của mình.

- Tạo danh sách bài hát: Người dùng tự tạo ra danh sách các bài hát của mình.

- Sửa danh sách bài hát: Người dùng có thể chỉnh sửa tên, thêm, xóa các bài hát ra khỏi danh sách đã được tạo.

- Phát nhạc trực tuyến: Người dùng có thể nghe nhạc trực tuyến.

-Tìm kiếm: Người dùng đã đăng nhập đầy đủ có thể tìm kiếm các bài hát theo tên, năm phát hành, nghệ sĩ, album các bài hát có trong kho lưu trữ và các danh sách.

Phía quản trị viên:

- Đăng nhập: Chức năng xác thực danh tính cảu quản trị viên để cấp quyền điều khiển hệ thống.

-Xem thông tin của người dùng trong hệ thống: Quản trị viên có khả năng lấy ra danh sách của các người dùng trong hệ thông cùng các thông tin liên quan.

- Tìm kiếm người dùng: Quản trị viên có thể lấy ra thông tin của một người dùng.

- Xóa người dùng: Quản trị viên có thể xóa tài khoản của một người dùng

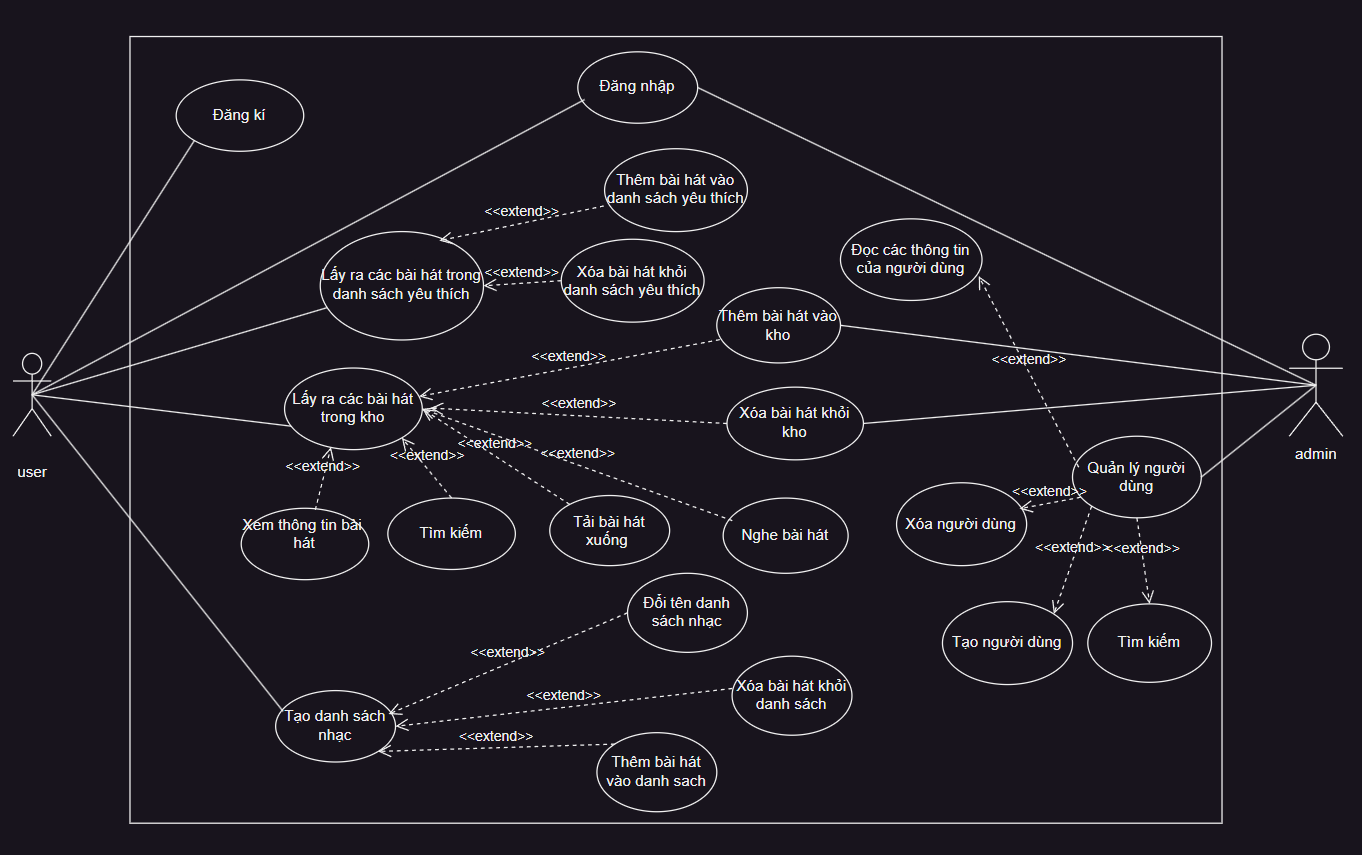
- Tạo người dùng: Quản trị viên có thể tạo người dùng mới.

- Xóa bài hát của người dùng: Quản trị viên có thể xóa bài hát của người dùng cụ thể.

- Thêm bài hát: Quản trị viên có thể thêm bài hát vào kho nhạc của một người dùng cụ thể.

Để có thể dễ hình dung ra , phần tiếp theo là sơ đồ use case của hệ thống backend.

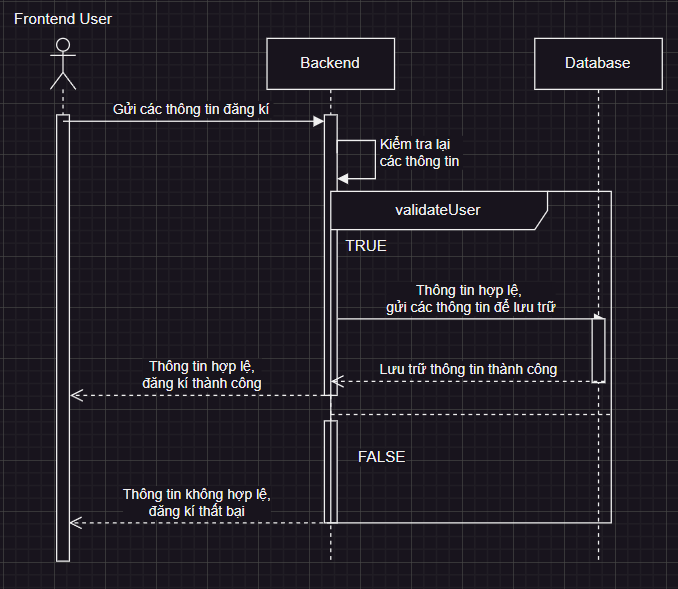
**1.3 Sơ đồ use case hệ thống**



**2. Mô tả chức năng của hệ thống**

***2.1 Các chức năng của frontend user:***

***2.1.1 Chức năng xử lý yêu cầu đăng kí:***



Mô tả:

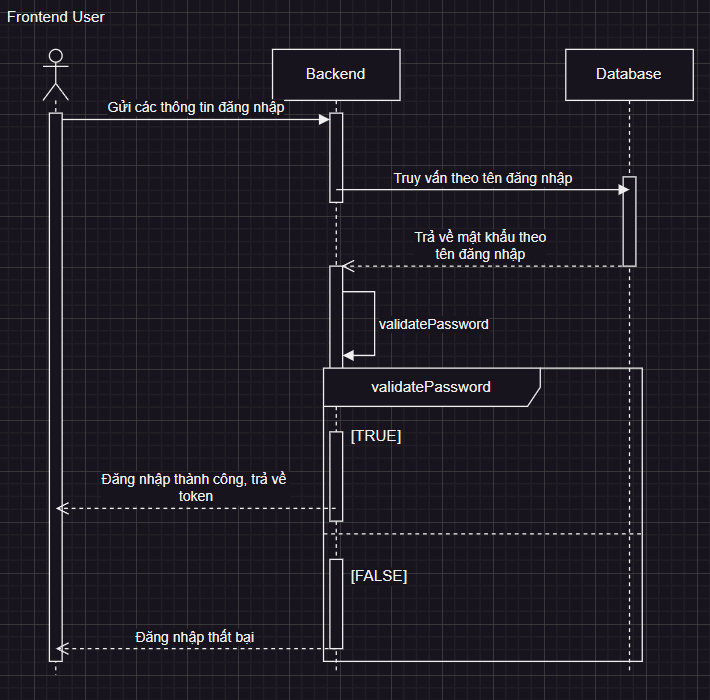
- Gửi các thông tin đăng kí: thông tin đăng kí bao gồm username, email, password.

- Kiểm tra lại các thông tin : username có yêu cầu độ dài tối thiểu 3, độ dài tối đa 20, email kiểm tra tính hợp lệ theo regex, password có độ dài tối thiểu là 5 kí tự.

- Nếu thông tin hợp lệ, tiến hành lưu trữ, sau khi lưu trữ thành công thì trả về frontend tạo đăng kí thành công

- Nếu thông tin không hợp lệ, trả về frontend thông tin không hợp lệ, đăng kí thất bại (thông tin không hợp lệ bao gồm cả username này đã có người sử dụng).

***2.1.2 Chức năng xử lý yêu cầu đăng nhập:***



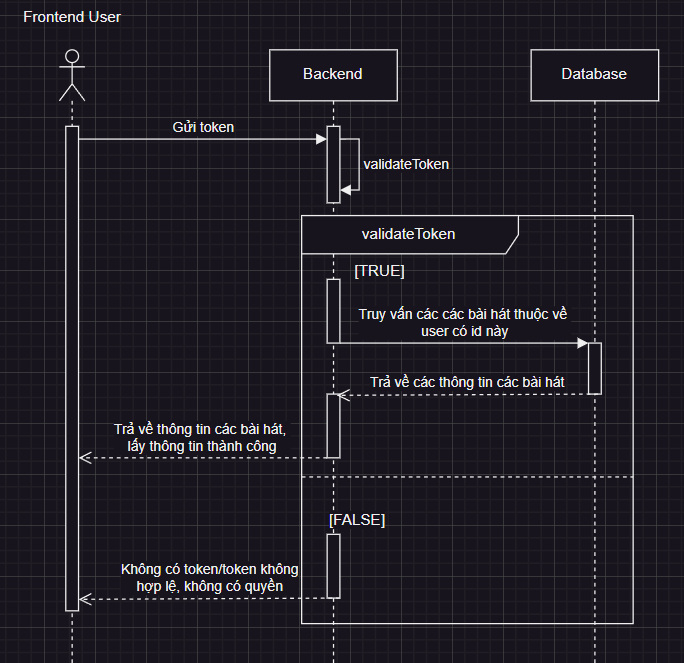
Mô tả:

- Gửi các thông tin đăng nhập: thông tin đăng nhập bao gồm username, password.

- Truy vấn theo tên đăng nhập: truy vấn đến cơ sở dữ liệu để lấy được mật khẩu được lưu trên hệ thống theo tên đăng nhập frontend gửi về.

- validatePassword: so sánh mật khẩu được lưu trên hệ thống với mật khẩu được gửi về, nếu trùng khớp, gửi lại frontend token cùng thông báo đăng nhập thành công, nếu không trùng khớp gửi thông báo đăng nhập thất bại

***2.1.3 Chức năng lấy ra các bài hát trong kho cá nhân:***

******

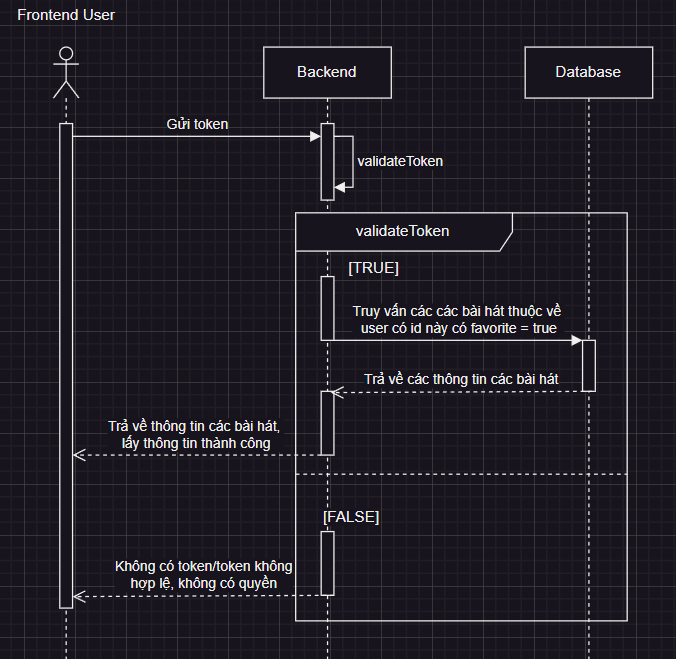
Mô tả

- Gửi token: Frontend sẽ gửi về token xác thực, nếu token xác thực thất bại, trả về không có quyền.

- validateToken: Token cần được xác định để xác định truy cập hợp lệ, token cần được giải mã để lấy được id user cần thực hiện hành động. Nếu token không có token thì trả về token không tồn tại, nếu không thể giải mã token thì báo không có quyền truy cập.

- Sau khi lấy ra được dữ liệu từ database, dữ liệu sẽ được xử lý để trả về dưới dạng Json Array.

***2.1.4 Chức năng lấy ra các bài hát trong danh sách favorite:***

******

Mô tả

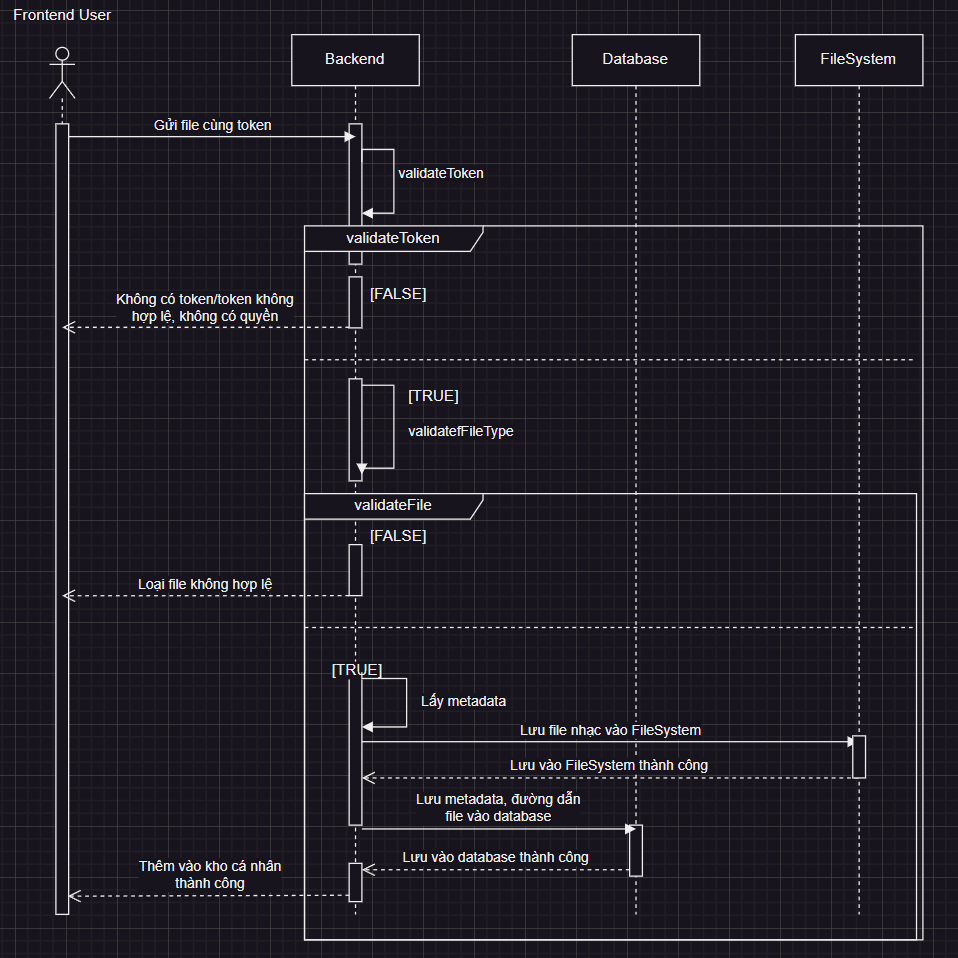
- Gửi token: Frontend sẽ gửi về token xác thực, nếu token xác thực thất bại, trả về không có quyền.

- validateToken: Token cần được xác định để xác định truy cập hợp lệ, token cần được giải mã để lấy được id user cần thực hiện hành động. Nếu token không có token thì trả về token không tồn tại, nếu không thể giải mã token thì báo không có quyền truy cập.

- Tiến hành truy vấn các bài hát thuộc về người dùng có trường favorite = true.

- Sau khi lấy ra được dữ liệu từ database, dữ liệu sẽ được xử lý để trả về dưới dạng Json Array.

***2.1.5 Chức năng thêm bài hát vào kho cá nhân:***



Mô tả:

- Gửi file cùng token: frontend gửi cho hệ thống file cần thêm vào kho cùng với token được lưu.

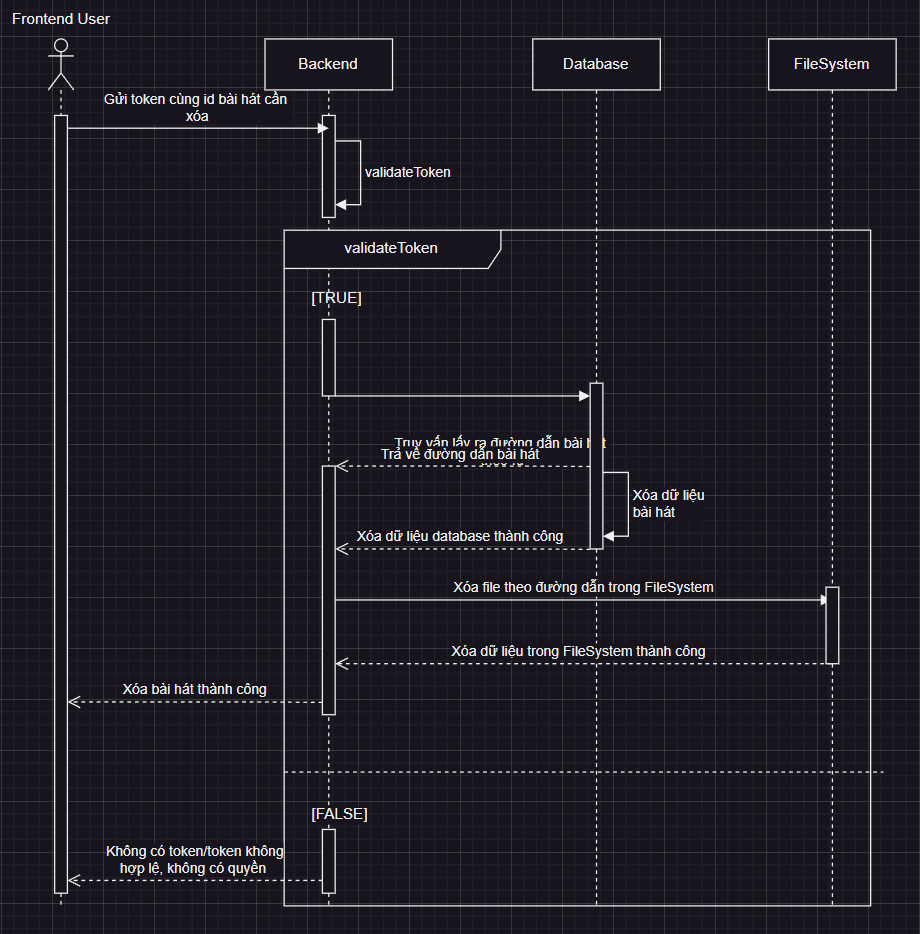
- validateToken: Token cần được xác định để xác định truy cập hợp lệ, token cần được giải mã để lấy được id user cần thực hiện hành động. Nếu token không có token thì trả về token không tồn tại, nếu không thể giải mã token thì báo không có quyền truy cập.

- validateFileType: Xác định xem file upload có phải chuẩn file nhạc với định dạng .mpeg .mp3 .flac không, nếu không đúng định dạng thì trả về thông báo là loại file không hợp lệ.

- Lấy metadata: Từ file nhận được, lấy ra metadata của file, lưu vào database.

- Lưu nhạc vào FileSystem: Lưu file nhạc vào thư mục được quy định trong hệ thống.

***2.1.6 Chức năng xóa bài hát ra khỏi kho cá nhân:***

******

Mô tả:

- Frontend gửi về token cùng id của bài hát cần xóa.

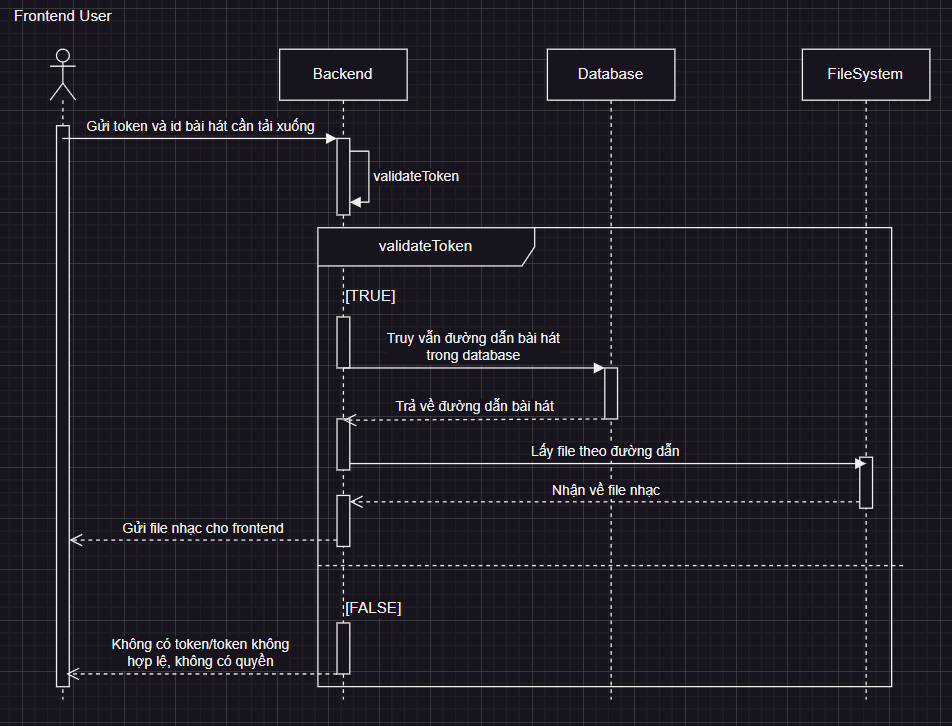
- validateToken: Token cần được xác định để xác định truy cập hợp lệ, token cần được giải mã để lấy được id user cần thực hiện hành động. Nếu token không có token thì trả về token không tồn tại, nếu không thể giải mã token thì báo không có quyền truy cập.

- Truy vấn đến database theo id bài hát để lấy được đường dẫn của bài hát trong file hệ thống, đồng thời xóa bài hát trong database

- Từ đường dẫn lấy được, xóa file có đường dẫn trên trong filesystem.

- Sau khi thực hiện yêu cầu thành công, trả về đã xóa bài hát.

***2.1.7 Chức năng tải xuống bài hát trong kho cá nhân:***

******

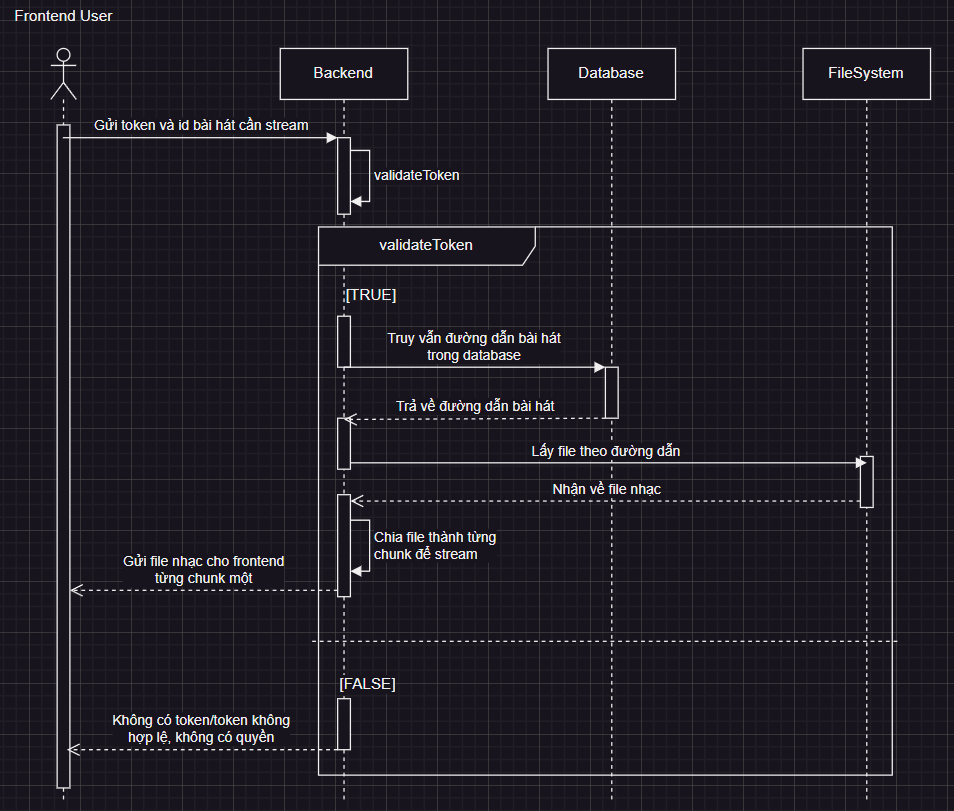
Mô tả:

- Frontend sẽ gửi về token xác thực và id cho bài hát cần tải xuống.

- validateToken: Token cần được xác định để xác định truy cập hợp lệ, token cần được giải mã để lấy được id user cần thực hiện hành động. Nếu token không có token thì trả về token không tồn tại, nếu không thể giải mã token thì báo không có quyền truy cập.

- Bằng id nhận được, tiến hành truy vấn đến database đường dẫn của file bài hát, sau khi lấy được file bài hát, backend gửi file bài hát cho file frontend.

***2.1.8 Chức năng nghe nhạc trực tuyến:***

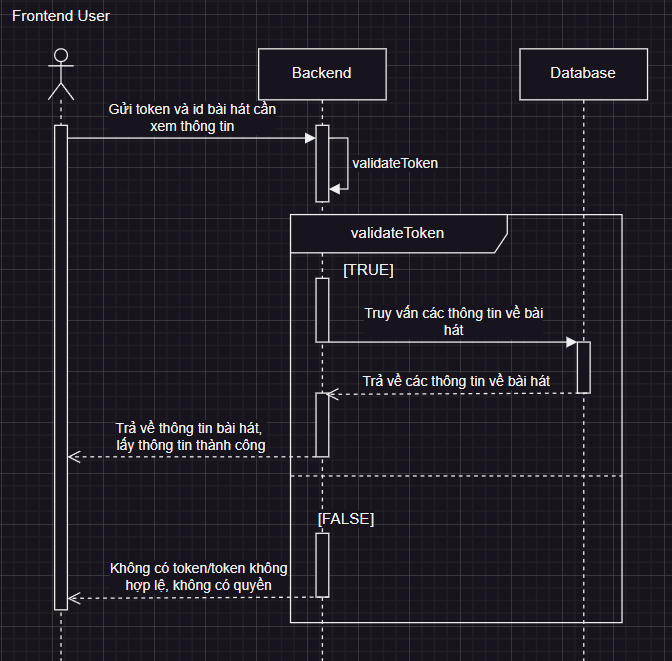
******

Mô tả:

- Frontend sẽ gửi về token xác thực và id cho bài hát cần stream, nếu token xác thực thất bại, trả về không có quyền.

- Bằng id nhận được, tiến hành truy vấn đến database đường dẫn của file bài hát, sau khi lấy được file bài hát, backend tiến hành chia nhỏ bài hát thành các chunk data, rồi gửi từng chunk cho frontend cho đến khi hoàn thành gửi toàn bộ bài hát.

***2.1.9 Chức năng xem thông tin bài hát***

******

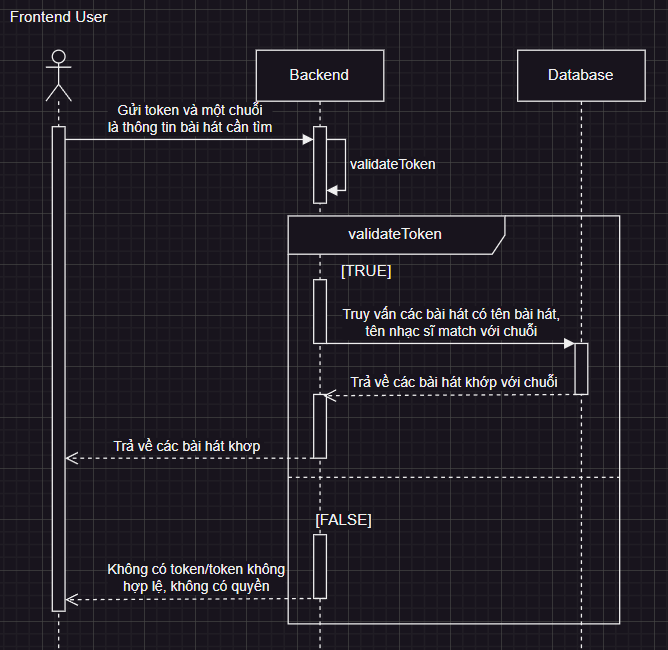
Mô tả:

- Frontend sẽ gửi về token xác thực và id cho bài hát cần tải xuống, nếu token xác thực thất bại, trả về không có quyền.

- Bằng id nhận được, tiến hành truy vấn đến database lấy ra toàn bộ thông tin về bài hát như name, artist, album, duration, năm phát hành, ngày upload file, kiểu file.

- Dữ liệu truy vấn được sẽ được xử lý để trả về cho frontend dưới dạng Json.

***2.1.10 Chức năng tìm kiếm bài hát***

******

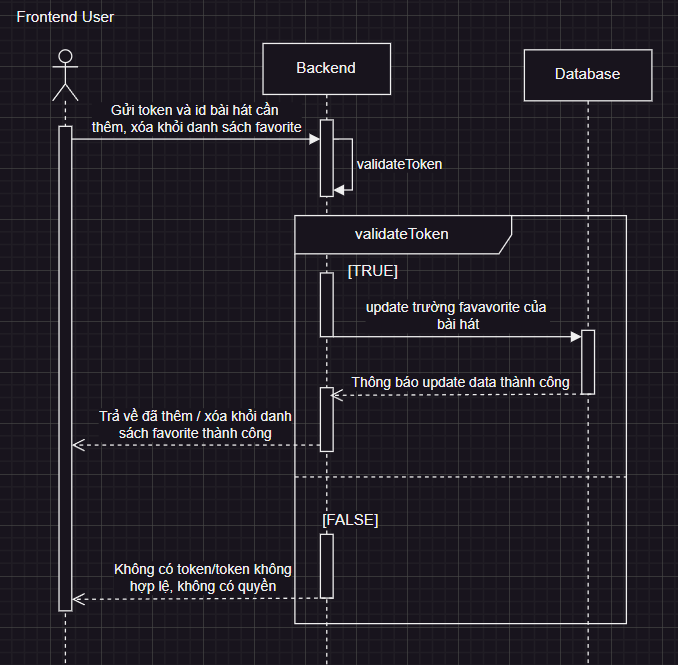
Mô tả:

- Frontend sẽ gửi về token xác thực và một chuỗi là giá trị tìm kiếm, nếu token xác thực thất bại, trả về không có quyền.

- Từ chuỗi được gửi về, truy vấn đến database tìm những bài hát có tên hoặc có tác giả khớp với chuỗi đã được truyền vào.

- Dữ liệu truy vấn được sẽ được xử lý để trả về cho frontend dưới dạng Json Array.

***2.1.11 Chức năng thêm / xóa khỏi danh sách bài hát yêu thích***

******

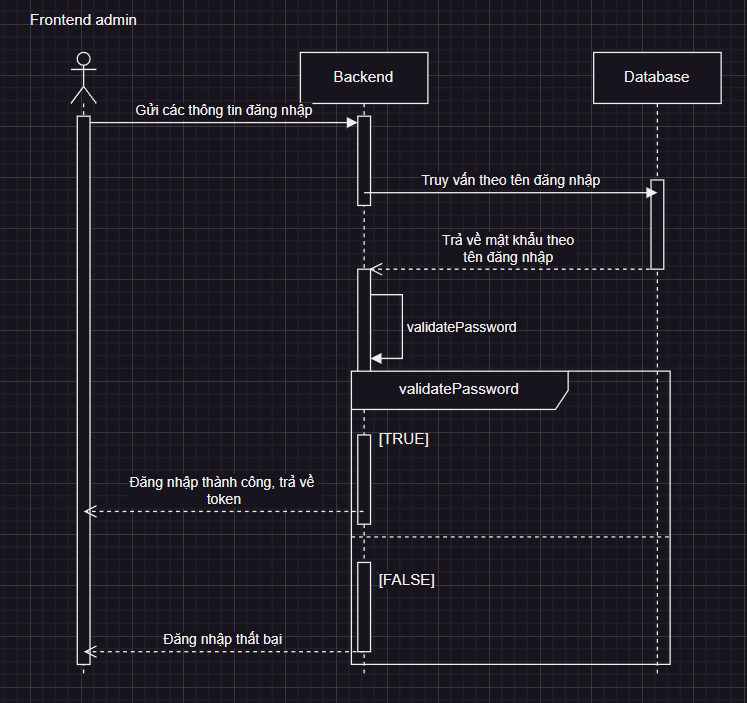
Mô tả:

- Frontend sẽ gửi về token xác thực, giá trị cần set ở trường favorite và id bài hát cần set, nếu token xác thực thất bại, trả về không có quyền.

- Từ giá trị favorite và id bài hát, trường favorite của bài hát trong database sẽ được update. Sau khi update, thông báo thêm / xóa khỏi danh sách favorite thành công.

***2.2 Các chức năng của frontend admin***

***2.2.1 Chức năng xử lý yêu cầu đăng nhập:***

******

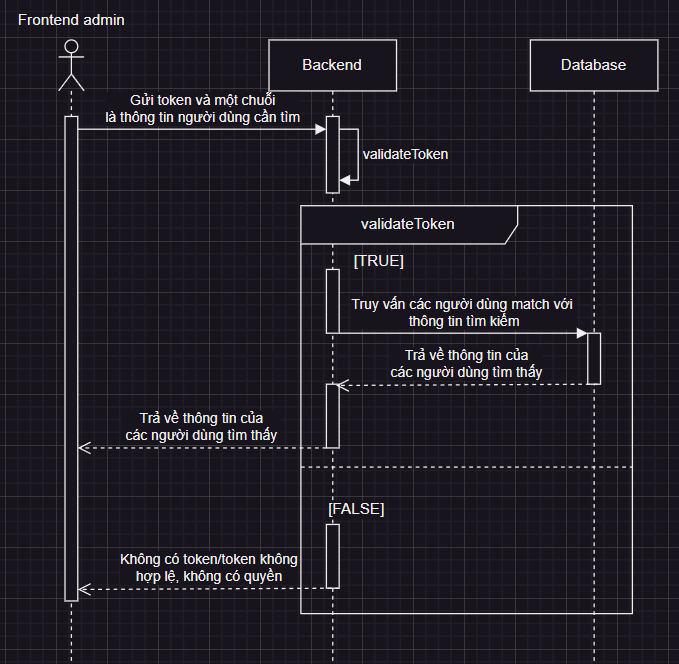
Mô tả:

- Gửi các thông tin đăng nhập: thông tin đăng nhập bao gồm username, password.

- Truy vấn theo tên đăng nhập: truy vấn đến cơ sở dữ liệu để lấy được mật khẩu được lưu trên hệ thống theo tên đăng nhập frontend gửi về.

- validatePassword: so sánh mật khẩu được lưu trên hệ thống với mật khẩu được gửi về, nếu trùng khớp, gửi lại frontend token cùng thông báo đăng nhập thành công, nếu không trùng khớp gửi thông báo đăng nhập thất bại

***2.2.1 Chức năng xử lý yêu cầu tìm kiếm người dùng:***

******

Mô tả:

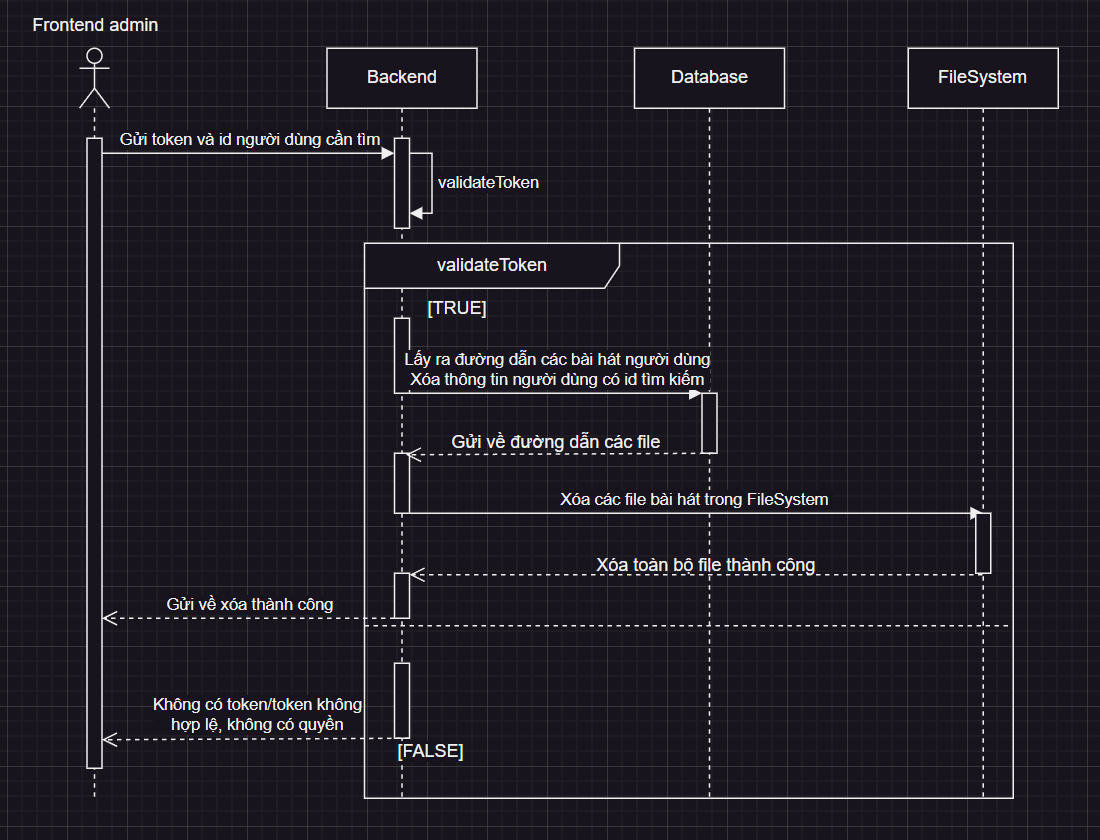
- Frontend sẽ gửi về token xác thực và một chuỗi là giá trị tìm kiếm, nếu token xác thực thất bại, trả về không có quyền.

- Từ chuỗi được gửi về, truy vấn đến database tìm những người dùng có tên hoặc id khớp với yêu cầu tìm kiếm.

- Sau khi truy vấn thành công lấy ra toàn bộ thông tin của các người dùng tìm được.

- Dữ liệu truy vấn được sẽ được xử lý để trả về cho frontend dưới dạng Json Array.

***2.2.1 Chức năng xử lý yêu cầu xóa người dùng***

******

Mô tả:

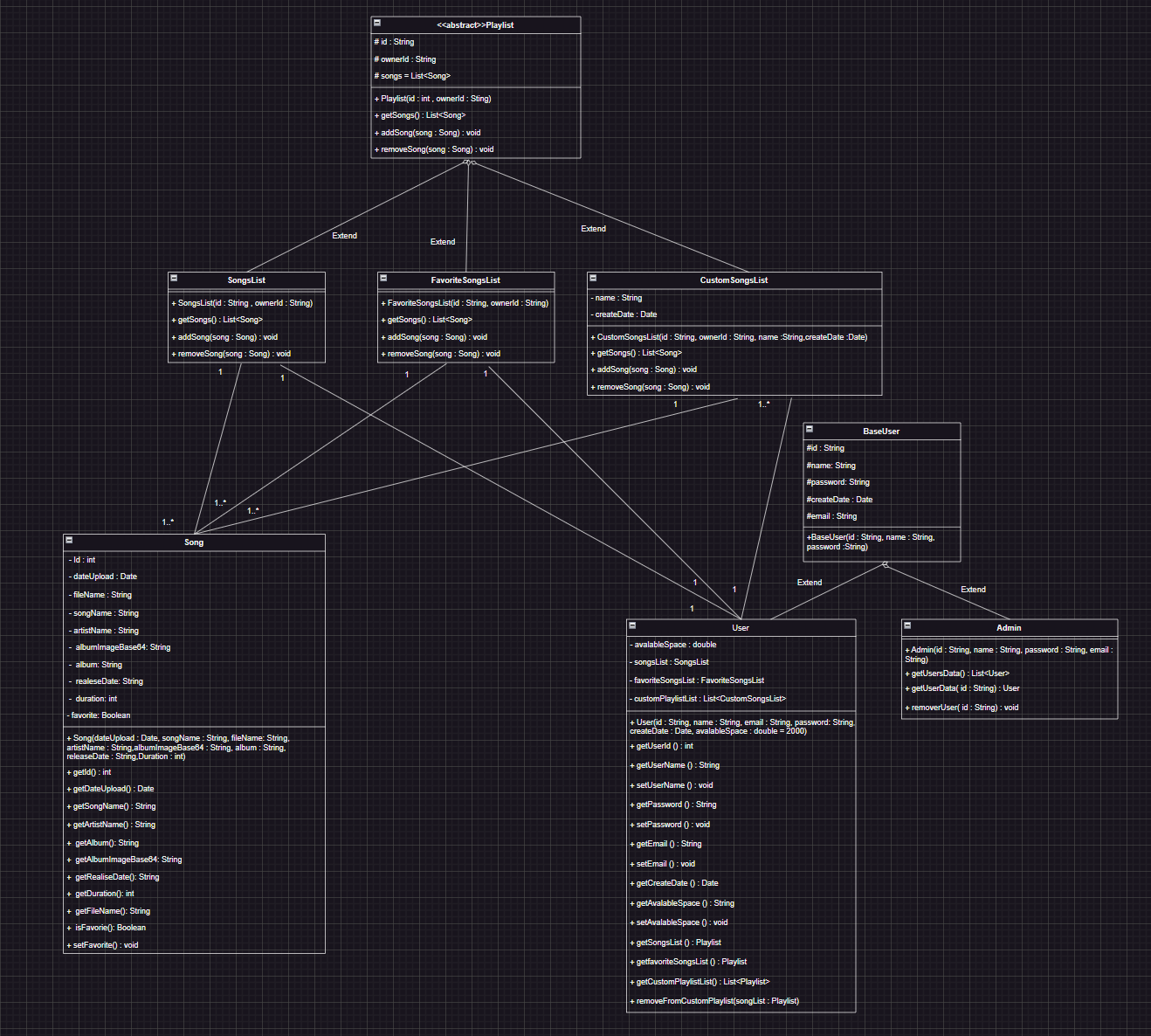
- Frontend sẽ gửi về token xác thực và một chuỗi là id người dùng, nếu token xác thực thất bại, trả về không có quyền.

- Truy vấn đến database lấy về đường dẫn các bài hát thuộc về người dùng, xóa toàn bộ dữ liệu liên quan đến người dùng trên database.

- Xóa các file bài hát có đường dẫn đã truy vấn được trong database trong FileSystem.

- Thông báo đã xóa người dùng thành công

***3. Sơ đồ lớp các đối tượng***

******

Endpoint design (All output send as JSON format)

* api/v1/auth/register POST (DONE)

Input : {username, email, password}

Output: {“msg” : “Account created successfully”}

* api/v1/auth/login POST

Input: {username, password}

Output: { token , username}

* api/v1/users GET

Input: req.headers.authorization (Bearer token)

Output: { username , email, createDate, numberOfSongs, availableStorage}

* api/v1/user DELETE

Input: req.headers.authorization (Bearer token), { password}

Output: {msg: Account delete successfully} / { msg: error.message}

* api/v1/users/songs GET

Input: req.header.authorization (Bearer token)

Output: {“msg” : “Success , “data” : [songs…]}

* api/v1/users/songs POST

Input: req.header.authorization (Bearer token), file

Output: { msg: "File uploaded successfully!" }

* api/v1/users/songs/:id GET

Input: req.header.authorization (Bearer token)

Output: file for streaming

* api/v1/users/songs/:id DELETE

Input: req.header.authorization (Bearer token)

Output: {msg: Song delete successfully}

* api/v1/users/songs/favorites GET

Input: req.header.authorization (Bearer token)

Output: {“msg” : “Success , “data” : [songs…]}

* api/v1/users/songs/favorites/:id POST

Input: req.header.authorization (Bearer token), {favorite}

Output: {“msg” : “Success”}

* api/v1/users/songs/dowload/:id GET

Input: req.header.authorization (Bearer token)

Output: file for dowloading