

TRƯỜNG ĐẠI HỌC SÀI GÒN
KHOA CÔNG NGHỆ THÔNG TIN



PHÁT TRIỂN PHẦN MỀM MÃ NGUỒN MỞ

Xây dựng ứng dụng

Clone Spotify sử dụng Django

GVHD: Từ Lăng Phiêu
SV: Tăng Hồng Nguyên Đán - 3121410139
Nguyễn Zi Đan - 3121410138
Email, Phone liên hệ: danquatao@gmail.com - 0372668600
heloemtrail@gmail.com - 0393656955

TP. HỒ CHÍ MINH, THÁNG 5/2025

BẢNG PHÂN CÔNG VÀ KHỐI LƯỢNG CÔNG VIỆC

Tên thành viên	Công việc	Khối lượng hoàn thành (%)
Tăng Hồng Nguyên Đán	<ul style="list-style-type: none">- Thiết kế cơ sở dữ liệu- Xây dựng backend với Django- Xây dựng "Lưu trữ và quản lý dữ liệu đa phương tiện trên Cloudinary"- Chức năng phát nhạc- Chức năng phát video- Chức năng thêm playlist- Chức năng thêm chi tiết nghệ sĩ- Chức năng thêm bài hát yêu thích	100%
Nguyễn Zi Đan	<ul style="list-style-type: none">- Thiết kế giao diện- Chức năng đăng nhập- Chức năng đăng ký- Chức năng quên mật khẩu- Chức năng xem cập nhật thông tin cá nhân (Profile)- Chức năng trang Admin- Chức năng quản lý User ở trang Admin- Chức năng quản lý Song ở trang Admin	100%



Mục lục

1	Mở đầu	4
1.1	Giới thiệu đề tài	4
1.2	Lý do chọn đề tài	4
1.3	Mục tiêu	4
1.4	Phạm vi	5
1.5	Tính ứng dụng	5
1.6	Yêu cầu phi chức năng	5
1.7	Yêu cầu kỹ thuật	5
2	Công nghệ áp dụng	6
2.1	Django	6
2.2	Django REST Framework (DRF)	6
2.3	ReactJS	6
2.4	Cloudinary	7
2.5	MySQL	7
3	Phân tích và thiết kế	8
3.1	Cơ sở dữ liệu	8
3.1.1	ERD	8
3.1.2	Thiết kế vật lý cơ sở dữ liệu	9
3.2	Phân chia hệ thống thành các ứng dụng con	9
3.3	Mô tả chi tiết các ứng dụng con	9
3.3.1	authentication	9
3.3.2	users	9
3.3.3	music	10
3.3.4	playlist	10
3.3.5	social	10
3.4	Mô hình hoạt động tổng thể	11
4	Thực nghiệm và phân tích kết quả	12
4.1	Đăng nhập vào hệ thống	12
4.2	Chức năng Quên Mật Khẩu	12
4.3	Đăng ký tài khoản	14
4.4	Giao diện màn hình chính	15
4.5	Chức năng phát nhạc	15
4.6	Chức năng xem chi tiết bài hát và nghệ sĩ	16
4.7	Chức năng quản lý playlist	16
4.8	Chức năng xem thông tin cá nhân	18
4.9	Chức năng cập nhật thông tin cá nhân	18
4.10	Chức năng Thêm bài hát yêu thích	19
4.11	Chức năng Phát Video Âm Nhạc	20
4.12	Chức năng trang Admin (quản trị hệ thống)	21
4.13	Chức năng Quản lý Người dùng (Admin)	22
4.14	Chức năng Quản lý Bài hát (Admin)	23



5	Cách thức cài đặt, môi trường cài ứng dụng	24
5.1	Công cụ sử dụng	24
5.2	Cấu trúc hệ thống	24
5.3	Yêu cầu môi trường	24
5.4	Tải và cài đặt project	24
5.5	Cấu hình Cloudinary (lưu ảnh/video)	26
5.6	Kiểm thử API bằng Postman	26
6	Kết luận và hướng phát triển	27
6.1	Kết luận	27
6.2	Hướng phát triển	27



1 Mở đầu

1.1 Giới thiệu đề tài

Trong thời đại công nghệ số phát triển mạnh mẽ, việc nghe nhạc trực tuyến đã trở thành thói quen phổ biến đối với người dùng trên toàn thế giới. Các nền tảng như Spotify, Apple Music hay Zing MP3 không chỉ cung cấp kho nhạc phong phú mà còn mang đến trải nghiệm cá nhân hóa, tiện lợi và đa nền tảng.

Tuy nhiên, phần lớn các nền tảng này là sản phẩm thương mại với mã nguồn đóng, hạn chế khả năng tùy chỉnh hoặc mở rộng theo nhu cầu học tập và nghiên cứu. Từ thực tế đó, đề tài “Xây dựng website nghe nhạc trực tuyến mô phỏng Spotify sử dụng Django” được thực hiện nhằm mục đích nghiên cứu, thiết kế và phát triển một hệ thống nghe nhạc trực tuyến mã nguồn mở, tích hợp các chức năng cơ bản như: phát nhạc, phát video nhạc, quản lý playlist, yêu thích bài hát, và hệ thống tài khoản người dùng.

Đề tài không chỉ mang tính thực tiễn mà còn là cơ hội để người học áp dụng tổng hợp các kiến thức về lập trình web (frontend và backend), quản lý cơ sở dữ liệu, bảo mật người dùng và triển khai hệ thống đa phương tiện trên môi trường thực tế.

1.2 Lý do chọn đề tài

Âm nhạc là một phần không thể thiếu trong đời sống hiện đại, và việc xây dựng một nền tảng nghe nhạc trực tuyến từ đầu là một thách thức giúp sinh viên củng cố và mở rộng kỹ năng lập trình toàn diện.

Việc mô phỏng nền tảng Spotify sẽ giúp người thực hiện đề tài hiểu sâu hơn về cách vận hành và kiến trúc hệ thống của một ứng dụng lớn, từ việc tổ chức dữ liệu, xử lý người dùng, phát nhạc/video trực tuyến, đến thiết kế giao diện người dùng và tối ưu hóa trải nghiệm.

Đề tài còn mang lại cơ hội rèn luyện tư duy giải quyết vấn đề thực tiễn, triển khai các công nghệ phổ biến hiện nay như Django, ReactJS và cơ sở dữ liệu MySQL.

1.3 Mục tiêu

- Xây dựng một hệ thống nghe nhạc trực tuyến với giao diện hiện đại, thân thiện với người dùng.
- Cung cấp các chức năng chính như:
 - Đăng ký, đăng nhập, quên mật khẩu.
 - Quản lý hồ sơ cá nhân (profile).
 - Phát nhạc trực tuyến, phát video nhạc.
 - Thêm bài hát vào danh sách yêu thích.
 - Tạo và quản lý playlist cá nhân.
 - Hệ thống quản trị (admin) để quản lý dữ liệu bài hát, người dùng, playlist, v.v.
- Sử dụng Django kết hợp Django REST Framework để xây dựng backend.



- Tích hợp ReactJS cho giao diện frontend hiện đại và phản hồi nhanh.
- Lưu trữ và truy xuất dữ liệu hiệu quả với MySQL.

1.4 Phạm vi

Đề tài tập trung vào việc xây dựng và triển khai các tính năng cốt lõi của một ứng dụng nghe nhạc trực tuyến, mô phỏng theo Spotify, bao gồm hệ thống người dùng, phát nhạc/video, quản lý bài hát và playlist.

Các tính năng nâng cao như gợi ý thông minh dựa trên hành vi người dùng, livestream, hoặc nghe offline sẽ không nằm trong phạm vi thực hiện của đề tài này do giới hạn về thời gian và nguồn lực.

1.5 Tính ứng dụng

Website nghe nhạc trực tuyến mô phỏng Spotify có thể được ứng dụng trong các lĩnh vực sau:

- **Giáo dục – đào tạo:** Là sản phẩm minh họa trực quan cho việc xây dựng hệ thống web toàn diện, phục vụ cho việc học tập, giảng dạy và làm đồ án tốt nghiệp.
- **Thực hành phát triển phần mềm:** Giúp người học thực hành từ việc thiết kế hệ thống, xây dựng API, giao diện người dùng đến xử lý dữ liệu và triển khai ứng dụng thực tế.
- **Tiền đề phát triển sản phẩm mở rộng:** Có thể được nâng cấp với các tính năng như gợi ý thông minh, tích hợp AI, hỗ trợ podcast hoặc phát sóng trực tiếp.

1.6 Yêu cầu phi chức năng

- **Hiệu suất:** Hệ thống cần đảm bảo tốc độ tải trang nhanh, phát nhạc mượt, hạn chế độ trễ và gián đoạn.
- **Bảo mật:** Áp dụng cơ chế xác thực, phân quyền, mã hóa mật khẩu và bảo vệ dữ liệu người dùng khỏi các cuộc tấn công phổ biến như SQL Injection, CSRF.
- **Khả năng mở rộng:** Thiết kế hệ thống linh hoạt, dễ mở rộng để tích hợp thêm các tính năng nâng cao như gợi ý bài hát bằng AI hoặc podcast trong tương lai.
- **Tính tương thích:** Website có thể hoạt động tốt trên các trình duyệt phổ biến (Chrome, Firefox, Safari, Edge) và các thiết bị có độ phân giải khác nhau (responsive design).

1.7 Yêu cầu kỹ thuật

- Công cụ sử dụng: Visual Studio Code, Xampp, Postman, Git
- Frontend: ReactJs.
- Backend: Django, Django Rest Framework.
- Database: MySQL.

2 Công nghệ áp dụng

2.1 Django

Django là một framework phát triển web mạnh mẽ được xây dựng bằng ngôn ngữ Python. Trong đề án Clone Spotify, Django được sử dụng để xây dựng phần backend – xử lý logic nghiệp vụ, quản lý người dùng, phân quyền truy cập, và cung cấp các API cho frontend.

- **ORM (Object-Relational Mapping):** Django cho phép thao tác dữ liệu MySQL thông qua mô hình đối tượng Python, giúp đơn giản hóa việc truy vấn dữ liệu, đặc biệt là trong việc quản lý người dùng, bài hát, nghệ sĩ và playlist.
- **URL Routing:** Django ánh xạ các URL tới các view logic tương ứng, giúp xây dựng hệ thống API rõ ràng và có cấu trúc cho ReactJS sử dụng.
- **Admin Interface:** Django cung cấp giao diện admin mạnh mẽ, cho phép quản trị viên dễ dàng thêm, sửa, xóa người dùng, bài hát, nghệ sĩ,... mà không cần xây dựng thủ công.
- **Tính năng bảo mật:** Django tích hợp các cơ chế bảo mật như CSRF, XSS, và hệ thống xác thực người dùng mạnh mẽ, đảm bảo an toàn cho hệ thống.
- **Khả năng mở rộng:** Django hỗ trợ phân quyền và xác thực linh hoạt – rất cần thiết để phân biệt giữa user, admin và artist trong hệ thống Spotify Clone.

2.2 Django REST Framework (DRF)

Django REST Framework là một thư viện mở rộng cho Django, giúp xây dựng các API RESTful một cách dễ dàng. Trong dự án, DRF đóng vai trò cốt lõi trong việc tạo ra các endpoint API để frontend ReactJS giao tiếp với hệ thống backend.

- **Serializers:** Sử dụng để chuyển đổi dữ liệu từ các model (bài hát, người dùng, nghệ sĩ...) sang định dạng JSON mà frontend có thể sử dụng, đồng thời hỗ trợ kiểm tra tính hợp lệ khi người dùng gửi yêu cầu (ví dụ: tạo tài khoản, upload ảnh avatar...).
- **Viewsets và Routers:** Tối ưu hóa việc viết các API CRUD cho các đối tượng chính như User, Artist, Track, Album,...
- **Authentication và Permissions:** Được dùng để kiểm soát truy cập, ví dụ như chỉ admin mới có quyền truy cập vào trang quản trị, còn artist có quyền thêm nhạc.
- **Throttling và Pagination:** Giúp giới hạn truy cập API và phân trang danh sách bài hát hoặc người dùng.
- **Tích hợp tốt với Django:** DRF tận dụng toàn bộ hệ thống ORM, middleware, và authentication có sẵn của Django, giúp hệ thống đồng nhất và dễ quản lý.

2.3 ReactJS

ReactJS được sử dụng để xây dựng giao diện người dùng (frontend) cho hệ thống Clone Spotify. Nhờ kiến trúc component-based, giao diện được chia thành các thành phần như navbar, trang người dùng, trang admin, trang quản lý nhạc...

- **Component-Based:** Mỗi phần giao diện như thanh điều hướng, trang chi tiết user, danh sách bài hát đều được xây dựng thành các component tái sử dụng.



- **Virtual DOM:** Giúp tối ưu hiệu năng khi hiển thị danh sách lớn như danh sách người dùng hoặc danh sách nhạc.
- **One-Way Data Binding:** Dữ liệu di chuyển một chiều giúp dễ kiểm soát luồng dữ liệu và tránh lỗi.
- **JSX:** Kết hợp logic xử lý và giao diện trong cùng một file giúp phát triển nhanh và dễ kiểm soát.
- **Tích hợp thư viện ngoài:** Dự án có thể sử dụng các thư viện như React Router (định tuyến giữa các trang), Axios (gửi yêu cầu HTTP đến backend), và các UI framework như TailwindCSS hoặc Material-UI để tăng tính thẩm mỹ.

2.4 Cloudinary

Trong đồ án, Cloudinary được sử dụng để lưu trữ và quản lý ảnh đại diện (avatar) người dùng, ảnh nghệ sĩ, bìa album... Việc này giúp giảm tải cho máy chủ và tối ưu hóa hiệu suất tải trang.

- **Tải lên và Lưu trữ ảnh:** Ảnh từ người dùng (ví dụ avatar khi tạo tài khoản) được tải lên Cloudinary và lưu trữ dưới dạng đường dẫn URL trên cloud.
- **Xử lý ảnh động và tối ưu:** Cloudinary hỗ trợ cắt, resize ảnh trực tuyến bằng cách chỉnh sửa URL – rất tiện lợi để hiển thị ảnh theo đúng kích thước mong muốn.
- **Phân phối nhanh qua CDN:** Tăng tốc độ tải ảnh trên toàn cầu, đặc biệt khi hệ thống phát triển mở rộng.
- **Tích hợp với Django:** Django có thư viện tích hợp Cloudinary (như `CloudinaryField`), giúp dễ dàng lưu và hiển thị ảnh qua DRF.

2.5 MySQL

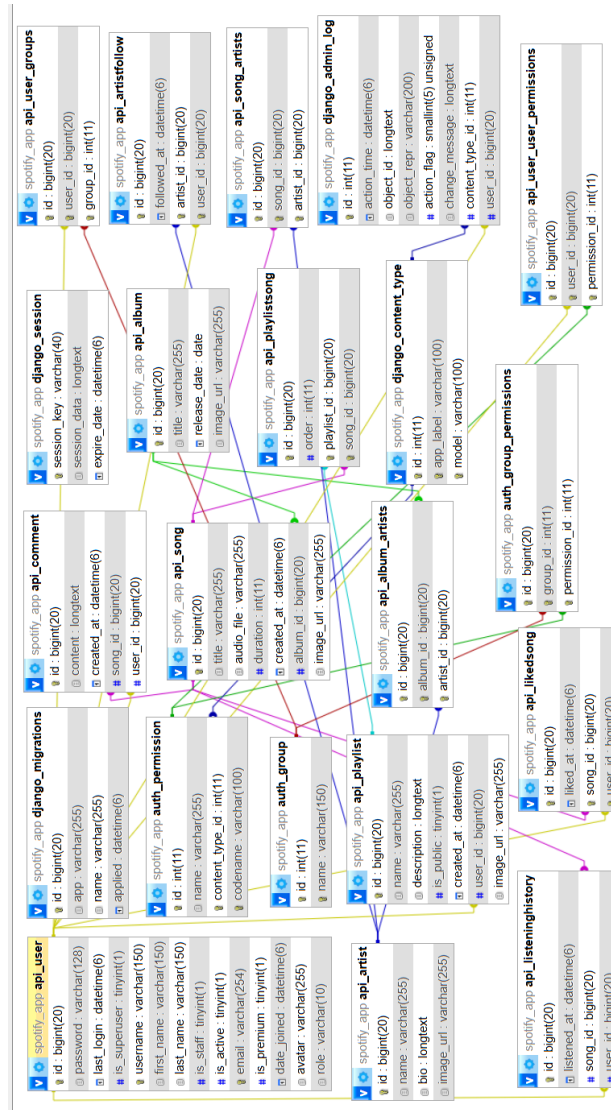
MySQL là hệ quản trị cơ sở dữ liệu được sử dụng trong đồ án để lưu trữ dữ liệu của hệ thống như tài khoản người dùng, thông tin bài hát, nghệ sĩ, album, playlist...

- **Tương thích tốt với Django ORM:** Django hỗ trợ MySQL rất tốt, cho phép thao tác dễ dàng qua model.
- **Hiệu suất tốt và dễ triển khai:** MySQL phù hợp với các hệ thống có quy mô trung bình và có thể mở rộng về sau.
- **Dễ sao lưu và phục hồi:** Thuận tiện khi phát triển hoặc chuyển hệ thống sang môi trường triển khai thực tế.

3 Phân tích và thiết kế

3.1 Cơ sở dữ liệu

3.1.1 ERD



Hình 1: Sơ đồ ERD hệ thống clone Spotify



STT	Tên bảng	Mô tả
1	user	Thông tin người dùng: username, email, password (mã hoá), avatar (lưu trên Cloudinary), role (user/admin/artist), ngày tạo tài khoản
2	artist	Thông tin nghệ sĩ âm nhạc: tên nghệ sĩ, mô tả, ảnh đại diện, ngày tạo
3	album	Thông tin album nhạc: tiêu đề, nghệ sĩ, ảnh bìa, ngày phát hành
4	playlist	Danh sách phát nhạc do người dùng tạo: tên playlist, mô tả, chế độ (riêng tư/công khai), người tạo
5	track_genre	Bảng trung gian giữa bài hát và thể loại
6	like_album	Danh sách album được người dùng yêu thích
7	follow_artist	Người dùng theo dõi nghệ sĩ để nhận thông báo mới
8	comment	Bình luận của người dùng dưới bài hát: nội dung, người bình luận, bài hát, thời gian

Bảng 1: Danh sách bảng trong hệ thống clone Spotify

3.1.2 Thiết kế vật lý cơ sở dữ liệu

3.2 Phân chia hệ thống thành các ứng dụng con

Để dễ quản lý, phát triển và mở rộng, hệ thống Django được chia thành các ứng dụng con (app) như sau:

- **authentication:** Xử lý các chức năng xác thực và phân quyền (đăng ký, đăng nhập, quên mật khẩu, xác minh OTP, phân vai trò).
- **users:** Quản lý thông tin hồ sơ người dùng, avatar, quyền admin quản lý người dùng.
- **music:** Quản lý nghệ sĩ, bài hát, album, thể loại nhạc.
- **playlist:** Tạo, chỉnh sửa, xoá playlist cá nhân, thêm bài hát vào playlist.
- **social:** Tương tác người dùng như like bài hát/album, bình luận, theo dõi nghệ sĩ.

3.3 Mô tả chi tiết các ứng dụng con

3.3.1 authentication

- Đăng ký tài khoản với vai trò người dùng hoặc admin.
- Đăng nhập bằng username và mật khẩu.
- Gửi OTP về email để khôi phục mật khẩu.
- Xác thực và phân quyền giữa các vai trò: user, admin.

3.3.2 users

- Xem và chỉnh sửa thông tin cá nhân: username, avatar, tiểu sử.
- Quên mật khẩu, cập nhật ảnh đại diện.
- Quản lý danh sách người dùng (dành cho admin).



3.3.3 music

- Tạo và cập nhật bài hát, album, nghệ sĩ (admin).
- Phân loại bài hát theo thể loại nhạc.
- Giao diện tìm kiếm bài hát theo tên, nghệ sĩ, thể loại.

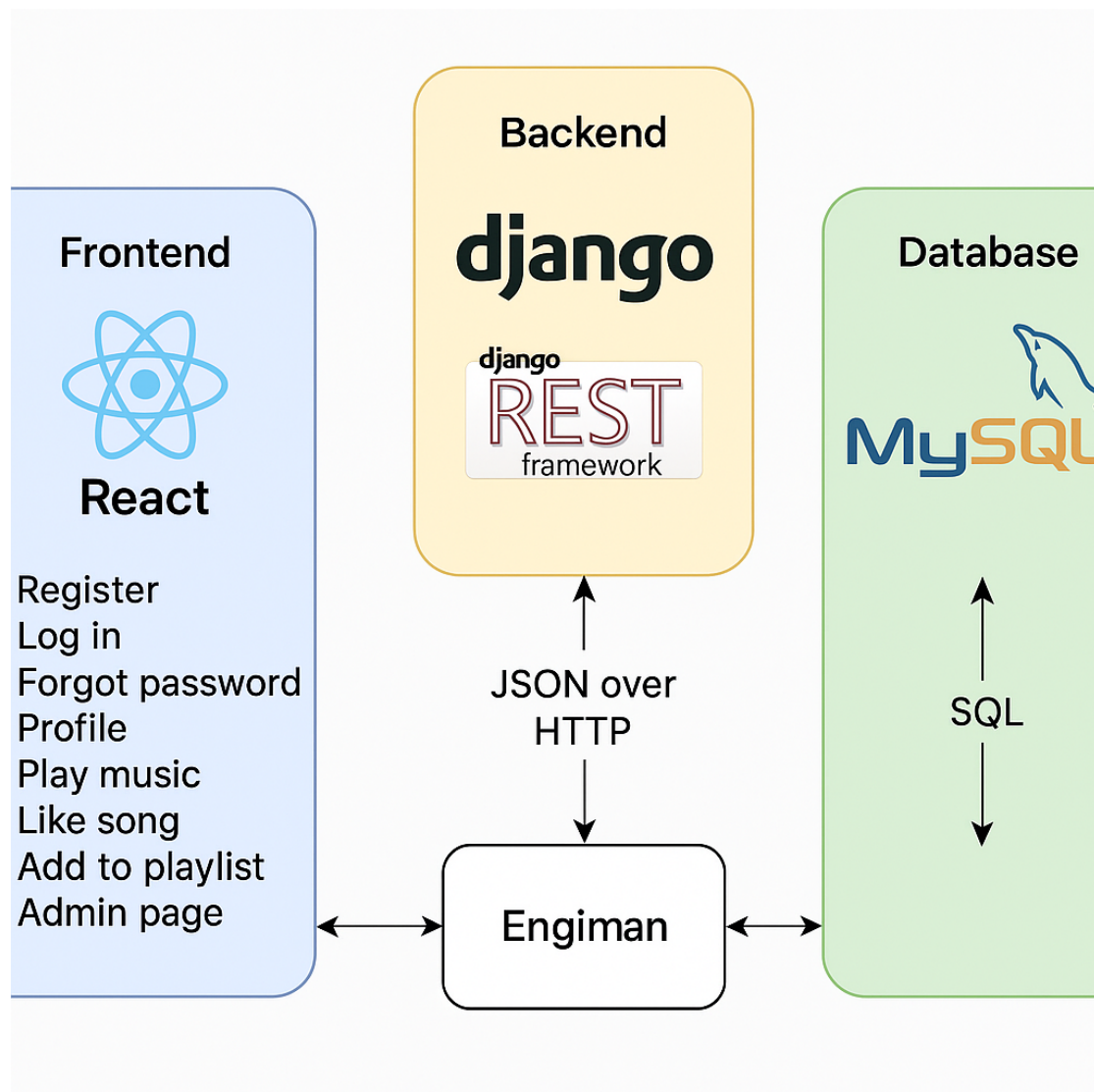
3.3.4 playlist

- Người dùng có thể tạo playlist riêng tư hoặc công khai.
- Thêm hoặc xóa bài hát vào playlist.
- Xem danh sách các playlist đã tạo.

3.3.5 social

- Like bài hát hoặc album để lưu vào danh sách yêu thích.
- Theo dõi nghệ sĩ để nhận thông báo khi có bài hát mới.
- Bình luận bài hát, hiển thị thời gian và người bình luận.

3.4 Mô hình hoạt động tổng thể

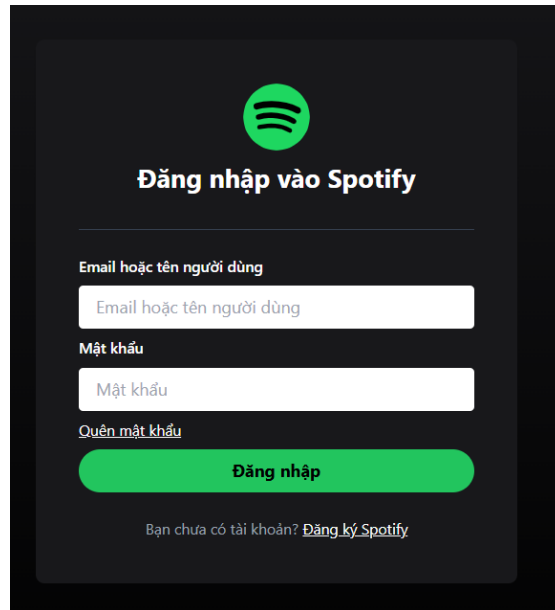


Hình 2: Mô hình hoạt động tổng thể của hệ thống clone Spotify

4 Thực nghiệm và phân tích kết quả

4.1 Đăng nhập vào hệ thống

Trước khi sử dụng hệ thống, người dùng cần thực hiện đăng nhập. Giao diện đăng nhập hiển thị như sau:



Hình 3: Giao diện đăng nhập người dùng

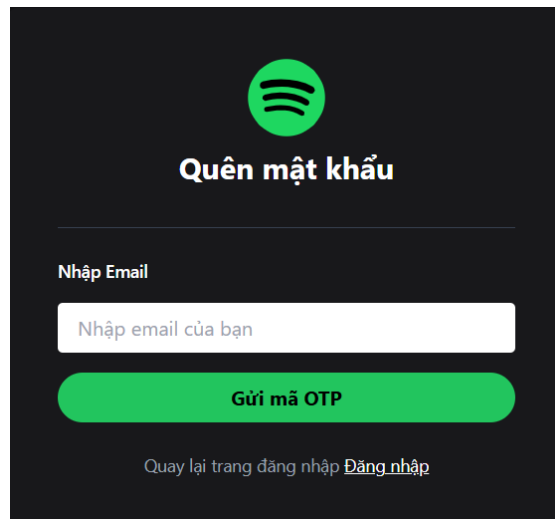
Người dùng có thể đăng nhập theo cách:

- Đăng nhập bằng username và mật khẩu đã đăng ký với hệ thống.

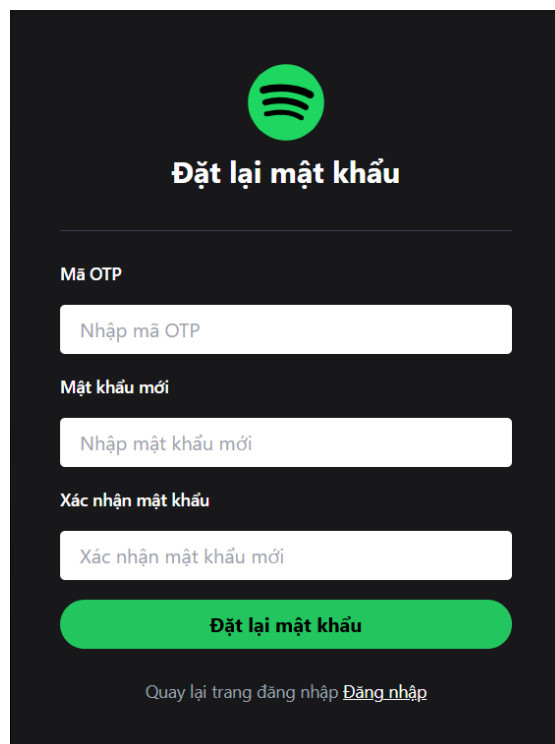
Sau khi đăng nhập thành công, người dùng sẽ được chuyển đến giao diện trang chủ.

4.2 Chức năng Quên Mật Khẩu

Khi người dùng quên mật khẩu, có thể sử dụng tính năng khôi phục mật khẩu trên trang đăng nhập bằng cách nhấn vào liên kết "Quên mật khẩu?". Giao diện minh họa như sau:



Hình 4: Giao diện nhập Email để gửi OTP

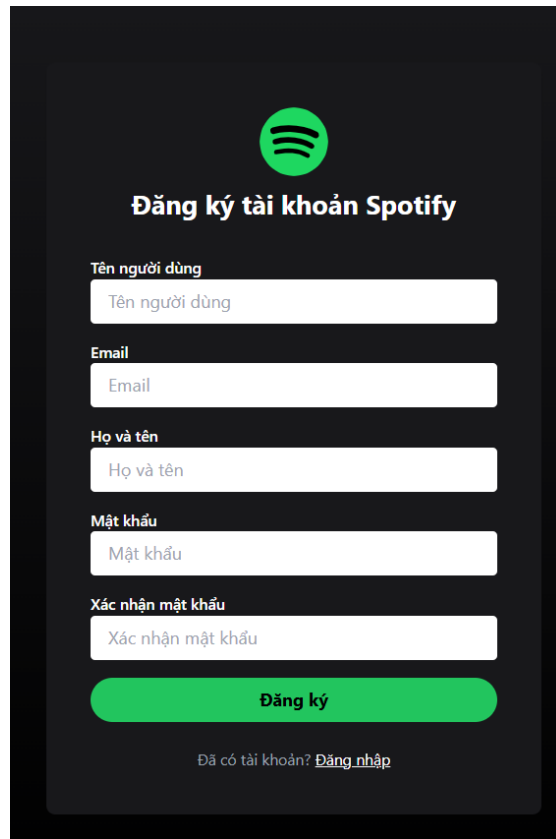


Hình 5: Giao diện khôi phục mật khẩu

Người dùng chỉ cần nhập địa chỉ email đã đăng ký, hệ thống sẽ gửi liên kết khôi phục mật khẩu qua email. Sau đó, người dùng có thể tạo mật khẩu mới và đăng nhập lại vào hệ thống.

4.3 Đăng ký tài khoản

Nếu chưa có tài khoản, người dùng có thể đăng ký tại trang đăng ký như sau:

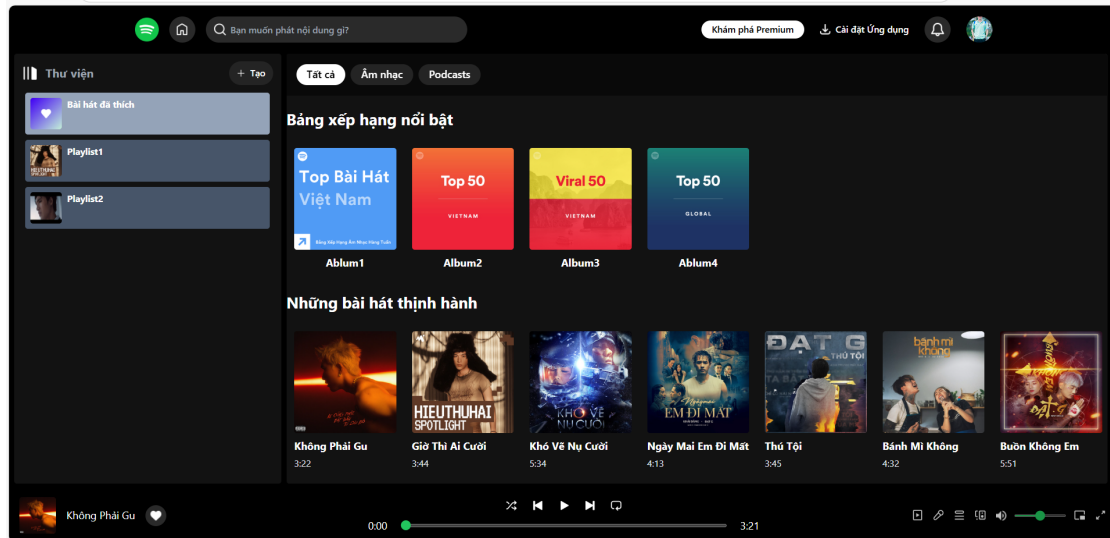


The image shows a Spotify registration form on a dark background. At the top is the Spotify logo (a green circle with three curved lines). Below it is the title "Đăng ký tài khoản Spotify". The form consists of several input fields: "Tên người dùng" (Username) with a placeholder "Tên người dùng", "Email" with a placeholder "Email", "Họ và tên" (Full name) with a placeholder "Họ và tên", "Mật khẩu" (Password) with a placeholder "Mật khẩu", and "Xác nhận mật khẩu" (Confirm password) with a placeholder "Xác nhận mật khẩu". Below these fields is a large green button labeled "Đăng ký". At the bottom, there is a link: "Đã có tài khoản? [Đăng nhập](#)".

Hình 6: Giao diện đăng ký người dùng

4.4 Giao diện màn hình chính

Sau khi đăng nhập thành công, người dùng được chuyển tới giao diện chính của ứng dụng nghe nhạc:



Hình 7: Giao diện màn hình chính (Trang chủ Spotify clone)

Màn hình chính bao gồm:

- Thanh điều hướng bên trái: bao gồm các mục như Header, Sidebar, Display, Playlist cá nhân.
- Khu vực trung tâm: hiển thị các album, bài hát nổi bật, playlist gợi ý và nội dung được cá nhân hóa.
- Trình phát nhạc (Music Player): nằm phía dưới màn hình, gồm các nút điều khiển nhạc, tiến độ phát nhạc, nút yêu thích và âm lượng.

4.5 Chức năng phát nhạc

Khi người dùng chọn một bài hát hoặc một album, giao diện phát nhạc sẽ hiện ra:



Hình 8: Giao diện phát nhạc

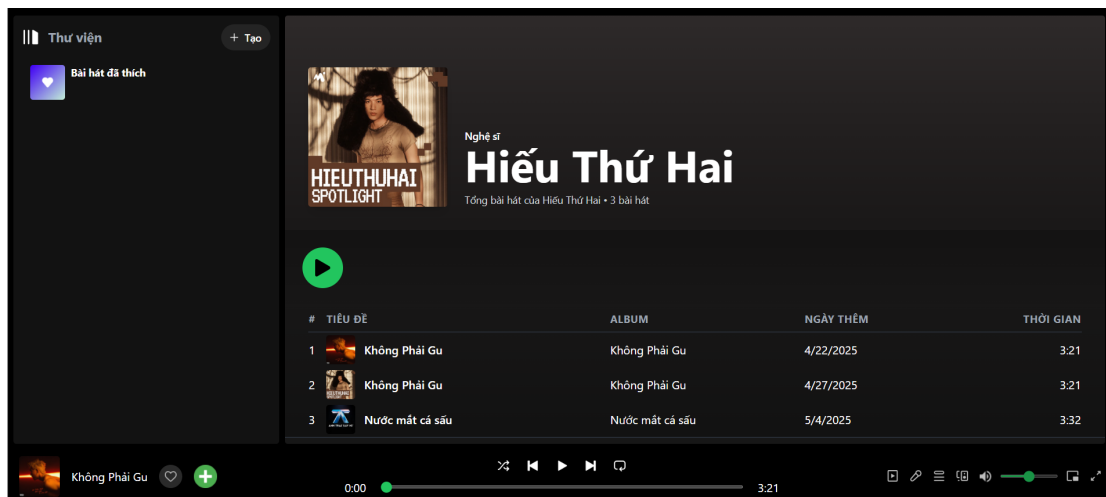
Các chức năng chính:

- Play/Pause, Next, Previous.
- Hiển thị ảnh bìa, tên bài hát.

- Điều chỉnh âm lượng, bật/tắt.
- Yêu thích bài hát và thêm vào playlist cá nhân.

4.6 Chức năng xem chi tiết bài hát và nghệ sĩ

Khi người dùng nhấn vào một bài hát hoặc nghệ sĩ bất kỳ trong ứng dụng, hệ thống sẽ chuyển hướng sang trang chi tiết:



Hình 9: Giao diện chi tiết bài hát và nghệ sĩ

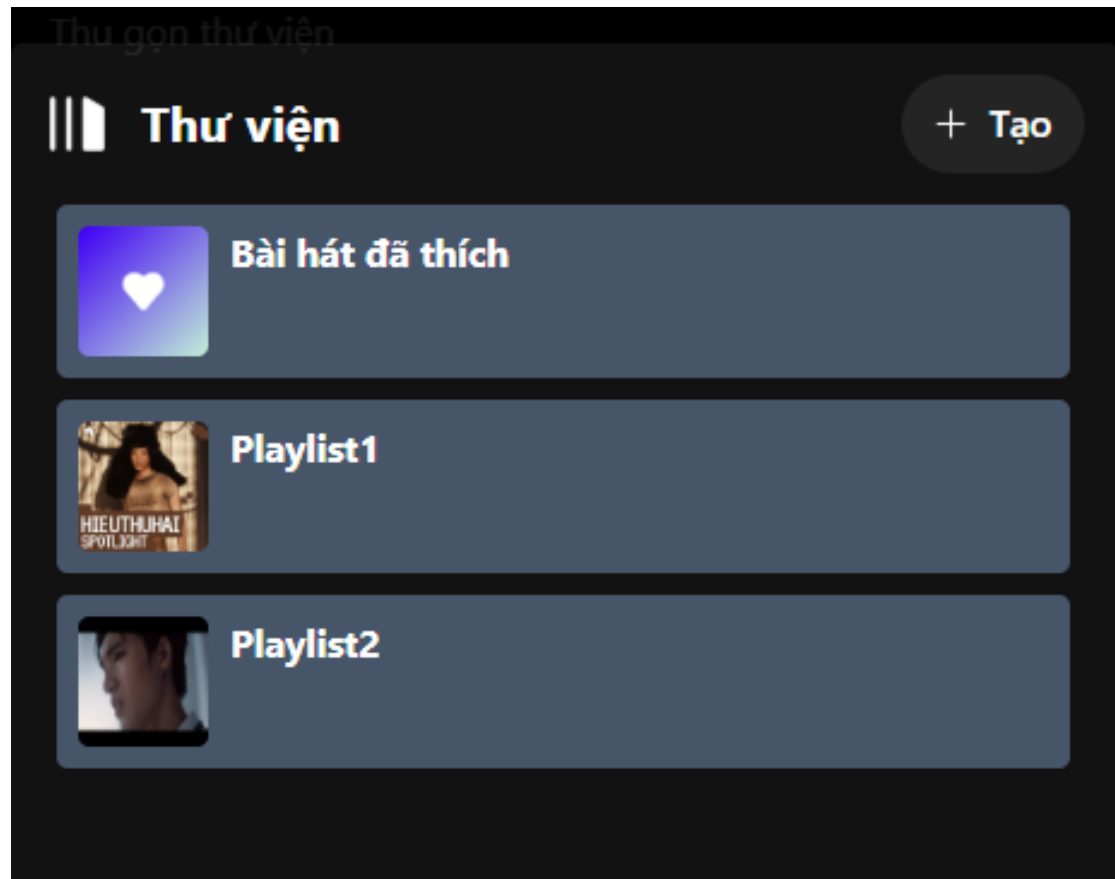
Các thông tin hiển thị bao gồm:

- Đối với bài hát: ảnh bìa, tên bài hát, nghệ sĩ trình bày, thời lượng, ngày phát hành.
- Đối với nghệ sĩ: ảnh đại diện, tên nghệ sĩ, tiểu sử, các bài hát.

Người dùng cũng có thể nhấn nút “Phát tất cả” hoặc thêm bài hát vào playlist yêu thích từ giao diện này.

4.7 Chức năng quản lý playlist

Người dùng có thể tạo playlist cá nhân để lưu trữ các bài hát yêu thích:



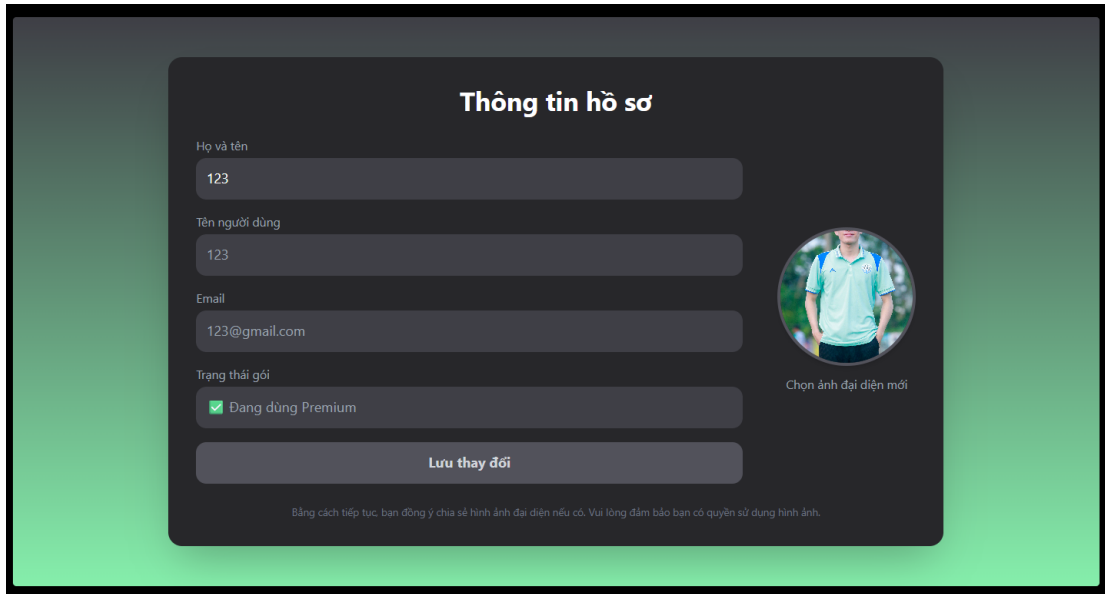
Hình 10: Giao diện playlist cá nhân

Tại đây người dùng có thể:

- Tạo mới playlist và đặt tên.
- Thêm hoặc xóa bài hát khỏi playlist.

4.8 Chức năng xem thông tin cá nhân

Người dùng có thể truy cập trang thông tin cá nhân từ Avatar ở góc trên phải màn hình:



Hình 11: Giao diện hồ sơ người dùng

Tại đây, người dùng có thể:

- Cập nhật ảnh đại diện (avatar), tên hiển thị.
- Cập nhật họ và tên.

4.9 Chức năng cập nhật thông tin cá nhân

Khi muốn thay đổi thông tin cá nhân, người dùng nhấn vào nút Update Profile tại giao diện Thông tin cá nhân, giao diện cập nhật thông tin cá nhân sẽ hiện ra như sau:

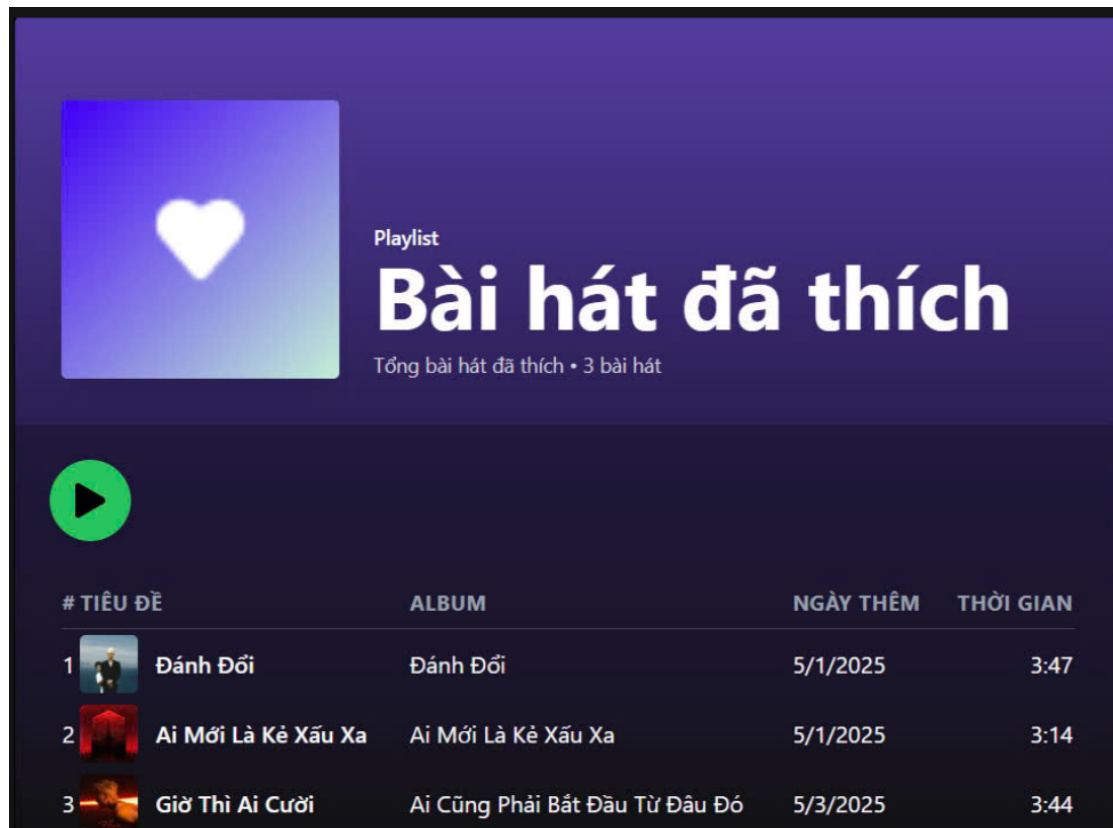


Hình 12: Giao diện cập nhật thông tin cá nhân

Người dùng tiến hành cập nhật các thông tin cần sửa sau đó nhấn nút Lưu, thông tin cá nhân sẽ được cập nhật.

4.10 Chức năng Thêm bài hát yêu thích

Khi người dùng muốn lưu lại những bài hát mình yêu thích để tiện nghe lại sau, có thể nhấn vào biểu tượng trái tim tại giao diện phát nhạc. Khi đó, bài hát sẽ được thêm vào danh sách yêu thích của người dùng. Giao diện minh họa như sau:

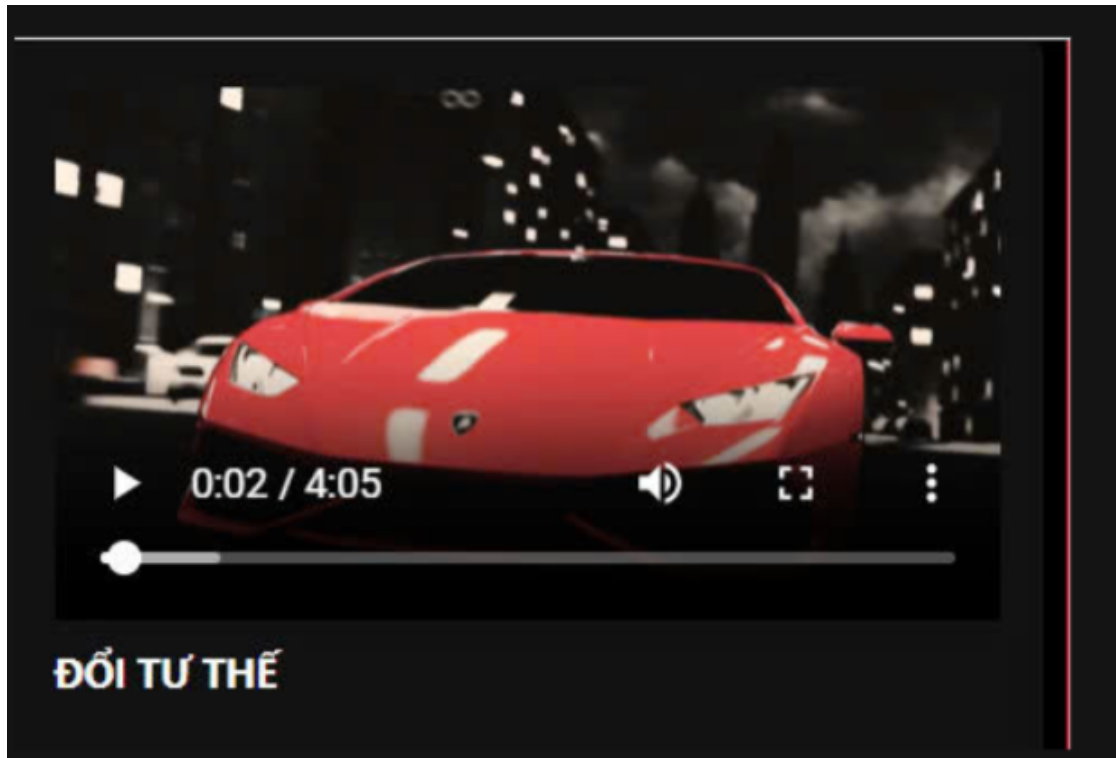


Hình 13: Giao diện thêm bài hát vào danh sách yêu thích

Tại đây, người dùng có thể dễ dàng quản lý các bài hát yêu thích của mình, nghe lại các bài hát đã lưu hoặc xóa khỏi danh sách nếu không còn yêu thích nữa.

4.11 Chức năng Phát Video Âm Nhạc

Với những bài hát có hỗ trợ video, người dùng có thể trải nghiệm trực quan hơn thông qua tính năng phát video nhạc. Giao diện phát video nhạc được hiển thị như sau:

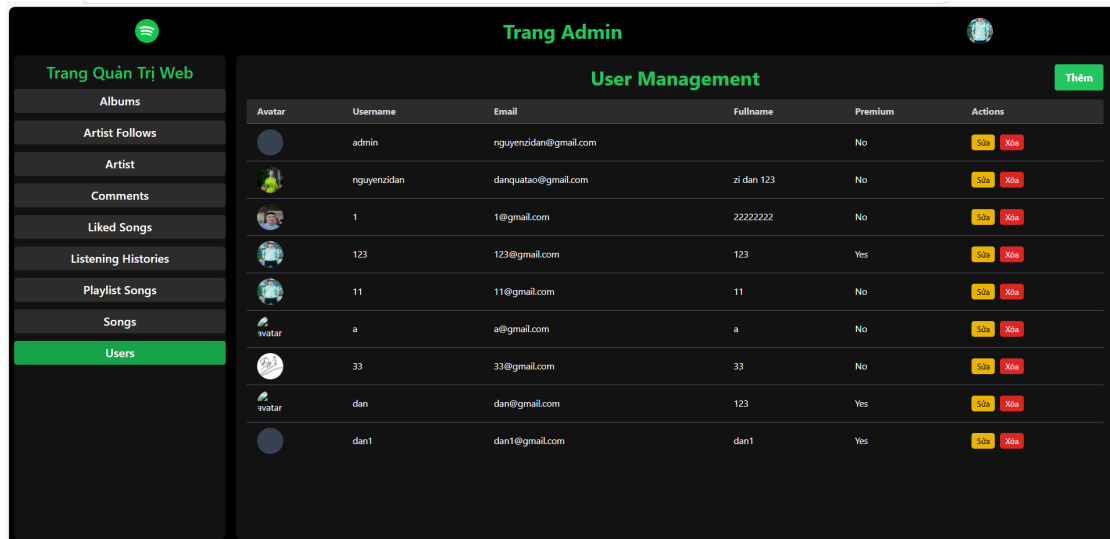


Hình 14: Giao diện phát video nhạc

Người dùng chỉ cần nhấn vào biểu tượng video hoặc chọn bài hát có hỗ trợ video, hệ thống sẽ hiển thị khung phát video tích hợp. Tại đây, người dùng có thể xem video toàn màn hình, điều chỉnh âm lượng, dừng/phát hoặc tua video tương tự như các nền tảng phát nhạc phổ biến.

4.12 Chức năng trang Admin (quản trị hệ thống)

Nếu người dùng có vai trò admin, có thể truy cập vào trang quản trị người dùng và nội dung:



Hình 15: Giao diện trang quản trị hệ thống

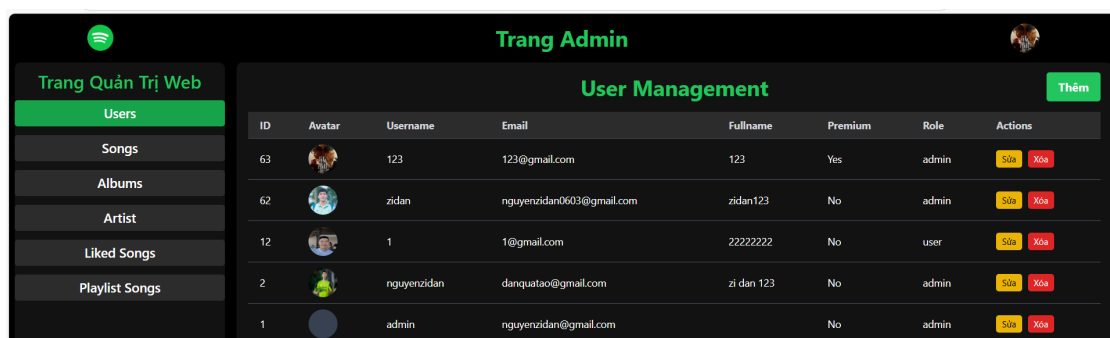
Chức năng chính:

- Quản lý người dùng: xem danh sách, chỉnh sửa hoặc xóa tài khoản.
- Quản lý nội dung: thêm mới nghệ sĩ, album, bài hát hoặc xóa các nội dung vi phạm.
- Phân quyền vai trò người dùng (User, Admin).

Giao diện sẽ hiển thị thông tin cá nhân của người dùng như: hình ảnh (avatar), tên người dùng, tiểu sử, email, ngày sinh, số điện thoại và cuối cùng là nút Update Profile cho phép người dùng cập nhật thông tin cá nhân.

4.13 Chức năng Quản lý Người dùng (Admin)

Từ trang quản trị, admin có thể truy cập giao diện quản lý người dùng như sau:



Hình 16: Giao diện quản lý người dùng (Admin)

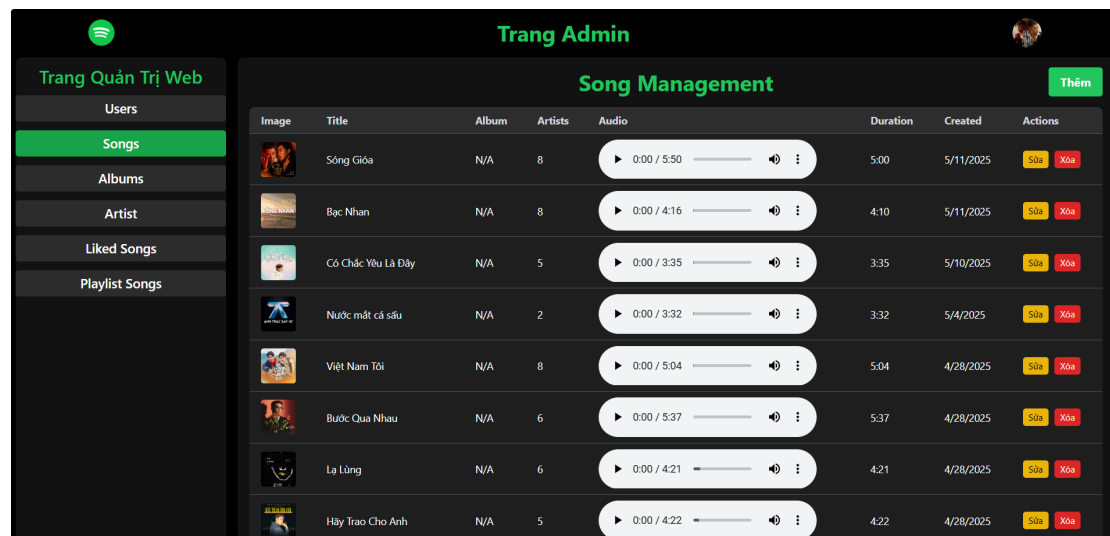
Chức năng bao gồm:



- Xem danh sách người dùng đã đăng ký.
- Chỉnh sửa thông tin người dùng: vai trò, tên, email, avatar,...
- Xóa tài khoản người dùng nếu cần.

4.14 Chức năng Quản lý Bài hát (Admin)

Từ trang quản trị, admin có thể truy cập giao diện quản lý bài hát như sau:



Hình 17: Giao diện quản lý bài hát (Admin)

Các chức năng chính:

- Xem danh sách tất cả các bài hát đã được đăng lên hệ thống.
- Thêm mới bài hát: upload ảnh bìa, audio, chọn nghệ sĩ và album liên quan.
- Chỉnh sửa thông tin bài hát: tên, nghệ sĩ, album, thể loại, thời lượng,...
- Xóa bài hát nếu vi phạm hoặc không còn phù hợp.



5 Cách thức cài đặt, môi trường cài ứng dụng

5.1 Công cụ sử dụng

Trong quá trình phát triển hệ thống clone Spotify, nhóm đã sử dụng các công cụ và công nghệ sau:

- **Visual Studio Code (VSCode)**: Công cụ chính để lập trình và phát triển mã nguồn.
- **Git**: Quản lý phiên bản mã nguồn, đồng thời hỗ trợ làm việc nhóm hiệu quả.
- **Postman**: Kiểm thử các API được xây dựng bằng Django REST Framework.
- **XAMPP**: Quản lý và vận hành hệ quản trị cơ sở dữ liệu MySQL.
- **Trình duyệt Chrome/Firefox**: Kiểm tra giao diện frontend ReactJS.

5.2 Cấu trúc hệ thống

- **Frontend**: Sử dụng ReactJS để xây dựng giao diện người dùng.
- **Backend**: Xây dựng bằng Django kết hợp Django REST Framework để cung cấp API cho frontend.
- **Cơ sở dữ liệu**: Sử dụng MySQL, được chạy trên XAMPP.
- **Lưu trữ media**: Sử dụng Cloudinary để lưu trữ ảnh người dùng (avatar), ảnh nghệ sĩ, ảnh bài hát, và video nhạc.

5.3 Yêu cầu môi trường

- **Node.js**: Phiên bản khuyến nghị: v20.11.0
- **Python**: Phiên bản từ 3.10 trở lên
- **MySQL**: Cài đặt qua XAMPP và tạo database tên là `spotify_app`

5.4 Tải và cài đặt project

Bước 1: Clone project từ GitHub

```
1 https://github.com/NguyenDan0606/spotify-app.git
```

Bước 2: Cài đặt frontend (ReactJS)

1. Di chuyển vào thư mục frontend:

```
1 cd frontend
```

2. Cài đặt các gói phụ thuộc:

```
1 npm install
```



3. Chạy ứng dụng React:

```
1 npm run dev
```

Bước 3: Cài đặt backend (Django)

1. Di chuyển vào thư mục backend:

```
1 cd backend
```

2. Tạo và kích hoạt môi trường ảo:

```
1 python -m venv venv
2 source venv/bin/activate # Trn Linux/macOS
3 .\venv\Scripts\activate # Trn Windows
```

3. Cài đặt các thư viện cần thiết:

```
1 pip install -r requirements.txt
```

4. Kết nối MySQL: Cập nhật thông tin kết nối trong settings.py

```
1 DATABASES = {
2     'default': {
3         'ENGINE': 'django.db.backends.mysql',
4         'NAME': 'spotify_app',
5         'USER': 'root',
6         'PASSWORD': '',
7         'HOST': '127.0.0.1',
8         'PORT': '3306',
9     }
10 }
```

5. Tạo database (nếu chưa có) thông qua phpMyAdmin hoặc câu lệnh MySQL:

```
1 CREATE DATABASE spotify_app;
```

6. Tạo các bảng trong cơ sở dữ liệu:

```
1 python manage.py makemigrations
2 python manage.py migrate
```

7. Tạo superuser để truy cập trang admin:

```
1 python manage.py createsuperuser
```

8. Chạy server Django:



```
1 python manage.py runserver
```

5.5 Cấu hình Cloudinary (lưu ảnh/video)

1. Đăng ký tài khoản tại: <https://cloudinary.com>
2. Lấy thông tin: cloud_name, api_key, api_secret
3. Cấu hình trong settings.py:

```
1 CLOUDINARY_STORAGE = {  
2     'CLOUD_NAME': 'your_cloud_name',  
3     'API_KEY': 'your_api_key',  
4     'API_SECRET': 'your_api_secret'  
5 }  
6  
7 DEFAULT_FILE_STORAGE = 'cloudinary_storage.storage.MediaCloudinaryStorage'
```

5.6 Kiểm thử API bằng Postman

- Kiểm thử các API như: Đăng nhập, Đăng ký, Quên mật khẩu, Cập nhật hồ sơ, Playlist, Nghệ sĩ, Phát nhạc,...
- Import file postman_collection.json để kiểm thử nhanh các chức năng.

6 Kết luận và hướng phát triển

6.1 Kết luận

Sau quá trình nghiên cứu và phát triển, nhóm đã hoàn thành đồ án xây dựng hệ thống Spotify Clone với các chức năng cơ bản như: đăng ký, đăng nhập, khôi phục mật khẩu, quản lý hồ sơ người dùng, phát nhạc, phát video nhạc, thêm vào danh sách phát, yêu thích bài hát và hệ thống trang quản trị dành cho admin.

Hệ thống được xây dựng với kiến trúc frontend-backend rõ ràng: sử dụng ReactJS cho giao diện người dùng, Django REST Framework cho việc xây dựng API backend và MySQL làm hệ quản trị cơ sở dữ liệu. Dữ liệu đa phương tiện được lưu trữ trên Cloudinary, đảm bảo hiệu quả và tốc độ truy xuất cao.

Dù hệ thống đã hoạt động ổn định ở mức cơ bản, vẫn còn một số tính năng nâng cao chưa được triển khai và cần được hoàn thiện trong tương lai.

6.2 Hướng phát triển

Để hoàn thiện hệ thống và đáp ứng tốt hơn nhu cầu người dùng, nhóm đề xuất một số hướng phát triển trong tương lai như sau:

- **Hoàn thiện các chức năng nâng cao:**
 - Tìm kiếm nâng cao theo tên bài hát, nghệ sĩ, thể loại, album.
 - Gợi ý bài hát dựa theo hành vi người dùng (recommendation system).
 - Hiển thị lời bài hát khi phát nhạc.
- **Hỗ trợ nhiều nền tảng:**
 - Xây dựng ứng dụng di động (Android/iOS) sử dụng React Native.
 - Đồng bộ hóa playlist và trạng thái phát nhạc giữa các thiết bị.
- **Tăng cường hiệu năng và bảo mật:**
 - Caching dữ liệu để tăng tốc độ truy xuất.
 - Bảo vệ API bằng JWT và phân quyền chi tiết hơn.
 - Mã hóa dữ liệu nhạy cảm và tăng cường cơ chế bảo vệ tài khoản người dùng.
- **Phát triển thêm tính năng cộng đồng:**
 - Cho phép người dùng bình luận, chia sẻ playlist.
 - Hệ thống theo dõi (follow) giữa người dùng và nghệ sĩ.
 - Hỗ trợ chat hoặc thảo luận trong cộng đồng âm nhạc.
- **Tích hợp dịch vụ bên thứ ba:**
 - Thanh toán trực tuyến để mở rộng lên tài khoản Premium.
 - Kết nối với mạng xã hội (Facebook, Google) để đăng nhập nhanh.

Những định hướng trên sẽ giúp hệ thống ngày càng hoàn thiện, mang lại trải nghiệm tốt hơn cho người dùng và tiệm cận với các nền tảng âm nhạc chuyên nghiệp như Spotify, Apple Music, Zing MP3.



Tài liệu

- [1] <https://docs.djangoproject.com/en/5.0/>, lần truy cập cuối: 04/05/2025.
- [2] <https://www.django-rest-framework.org/topics/documenting-your-api/>, lần truy cập cuối: 04/05/2025.
- [3] <https://django-rest-framework-simplejwt.readthedocs.io/en/latest/>, lần truy cập cuối: 04/05/2025.
- [4] <https://gemini.google.com/app>, lần truy cập cuối: 04/05/2025.
- [5] <https://chatgpt.com/>, lần truy cập cuối: 04/05/2025.