TRƯỜNG ĐẠI HỌC SÀI GÒN

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO

ĐỀ TÀI: PHÁT TRIỂN PHẦN MỀM SPOTIFY CLONE

Giảng viên hướng dẫn: ThS. Từ Lãng Phiêu

Nhóm sinh viên thực hiện:
3121410309 Lê Trọng Lực
3121410543 Phạm Hoàng Đan Trường
3122560021 Lê Văn Hoàng
3122560092 Trần Kim Yến
3122410298 Trần Tiến Phát
Email liên hệ: letrongluc123456@gmail.com

TP. Hồ Chí Minh, ngày 3 tháng 5 năm 2025

LỜI CẢM ƠN

- Lời đầu tiên, nhóm chúng em xin gửi lời biết ơn sâu sắc đến Thầy Từ Lăng Phiêu, người đã là nguồn động viên và hỗ trợ không ngừng trong quá trình thực hiện đề tài và báo cáo này. Những sự chỉ dạy tận tâm, kinh nghiệm quý báu, cùng sự sẵn lòng giải đáp mọi thắc mắc và góp ý xây dựng đã giúp nhóm vượt qua khó khăn và hoàn thành đề tài đúng tiến độ.

- Nhóm chúng em cũng bày tỏ lòng biết ơn đến Ban Giám Hiệu Trường Đại Học Sài Gòn, cùng quý Thầy Cô trong Khoa Đào Tạo, đặc biệt là ngành Công Nghệ Thông Tin, đã tạo điều kiện thuận lợi và cung cấp nền tảng kiến thức vững chắc để nhóm thực hiện đề tài môn Phát Triển Phần Mềm Mã Nguồn Mở.

- Dù đã nỗ lực hoàn thiện, nhóm nhận thức rằng đề tài và báo cáo vẫn còn nhiều hạn chế do kiến thức và kỹ thuật còn hạn chế. Nhóm mong nhận được sự góp ý quý báu từ Thầy Cô để hoàn thiện hơn trong tương lai. Cuối cùng, nhóm xin chúc Thầy Cô luôn khỏe mạnh, thành công và hạnh phúc trong mọi công việc.

Nhóm chúng em xin chân thành cảm ơn.

Mục lục

1 Giới thiệu

1.1 Lý do chọn đề tài

Ngày nay, âm nhạc đóng vai trò quan trọng trong đời sống tinh thần của con người. Với sự phát triển mạnh mẽ của Internet và các nền tảng số, nhu cầu nghe nhạc trực tuyến ngày càng phổ biến. Spotify là một trong những nền tảng nghe nhạc trực tuyến hàng đầu thế giới, mang đến trải nghiệm nghe nhạc tiện lợi, cá nhân hóa và dễ sử dụng.

Từ đó, nhóm quyết định chọn đề tài "Phát triển phần mềm Spotify Clone" với mục tiêu tìm hiểu cách xây dựng một hệ thống nghe nhạc trực tuyến hoàn chỉnh, mô phỏng các chức năng chính của Spotify như phát nhạc, tạo playlist, quản lý người dùng, và hỗ trợ nghệ sĩ đăng tải nội dung.

1.2 Mục tiêu đề tài

Đề tài hướng đến các mục tiêu cụ thể sau:

- Xây dựng một hệ thống web có thể phát nhạc trực tuyến với giao diện người dùng thân thiện.
- Hỗ trợ các chức năng quản lý playlist, bài hát yêu thích, nghệ sĩ, và quản trị hệ thống.
- Áp dụng các công nghệ hiện đại như ReactJS (frontend), Django (backend) và MySQL (cơ sở dữ liệu).
- Tìm hiểu và triển khai mô hình phân quyền người dùng (người dùng, nghệ sĩ, quản trị viên).

1.3 Phạm vi thực hiện

Đề tài được thực hiện trong khuôn khổ môn học "Phát Triển Phần Mềm Mã Nguồn Mở". Do giới hạn về thời gian và nguồn lực, hệ thống Spotify Clone được phát triển với các chức năng cơ bản sau:

- Giao diện người dùng cho phép đăng ký, đăng nhập, nghe nhạc và tạo playlist.
- Giao diện nghệ sĩ cho phép đăng tải và quản lý bài hát.
- Giao diện quản trị viên quản lý người dùng và nội dung.
- Hệ thống được triển khai thử nghiệm trên môi trường cục bộ.

2 Cơ sở lý thuyết

2.1 Công nghệ sử dụng

Để phát triển phần mềm Spotify Clone, nhóm đã lựa chọn các công nghệ hiện đại, phổ biến và phù hợp với yêu cầu của một hệ thống nghe nhạc trực tuyến, bao gồm:

2.1.1 ReactJS

ReactJS là một thư viện JavaScript do Facebook phát triển, dùng để xây dựng giao diện người dùng. React giúp phát triển các thành phần UI theo hướng component-based, từ đó tái sử dụng code và dễ dàng quản lý trạng thái (state). React còn hỗ trợ khả năng tương tác nhanh và mượt mà với người dùng nhờ cơ chế Virtual DOM.

2.1.2 Django

Django là một framework web mạnh mẽ viết bằng Python, giúp xây dựng backend nhanh chóng và bảo mật. Django hỗ trợ mô hình MVC, tích hợp ORM để làm việc với cơ sở dữ liệu, và có sẵn nhiều công cụ hỗ trợ quản lý người dùng, phân quyền, gửi email, v.v. Điều này rất phù hợp với hệ thống có phân vai người dùng như Spotify Clone.

2.1.3 MySQL

MySQL là hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở, nổi tiếng với khả năng xử lý dữ liệu nhanh chóng, bảo mật cao và dễ tích hợp với nhiều nền tảng. Nhóm sử dụng MySQL để lưu trữ thông tin người dùng, bài hát, video, playlist, lịch sử phát, v.v.

2.2 Kiến trúc phần mềm Web

Spotify Clone được xây dựng theo mô hình kiến trúc phân lớp:

- **Frontend (Client-side):** Giao diện người dùng được xây dựng bằng ReactJS, đảm nhiệm việc hiển thị và thu thập dữ liệu từ người dùng.
- **Backend (Server-side):** Django xử lý các yêu cầu từ frontend, thực hiện các nghiệp vụ như xác thực, quản lý dữ liệu, xử lý API.
- **Cơ sở dữ liệu:** MySQL lưu trữ dữ liệu một cách bền vững và có cấu trúc rõ ràng.

Các thành phần này giao tiếp thông qua các API được thiết kế theo chuẩn REST, đảm bảo tính mở rộng và dễ tích hợp.

3 Phân tích hệ thống

Trong phần này, nhóm tiến hành phân tích các yêu cầu chức năng của hệ thống, từ đó xây dựng các mô hình để thể hiện luồng hoạt động của các thành phần chính.

3.1 Chức năng người dùng

Người dùng thông thường có thể thực hiện các chức năng sau:

- Đăng ký, đăng nhập và xác thực qua email.
- Nghe nhạc, xem video âm nhạc.
- Tìm kiếm bài hát, nghệ sĩ, video.
- Thêm bài hát/video vào danh sách yêu thích hoặc playlist cá nhân.
- Kết bạn và nhắn tin real-time với người dùng khác.

3.2 Chức năng nghệ sĩ

Nghệ sĩ có thể thực hiện các chức năng:

- Đăng nhập vào hệ thống với vai trò nghệ sĩ.
- Thêm, sửa, xóa bài hát và video âm nhạc của mình.
- Quản lý danh sách bài hát, video cá nhân.
- Xem phản hồi từ người dùng và thống kê lượt phát.

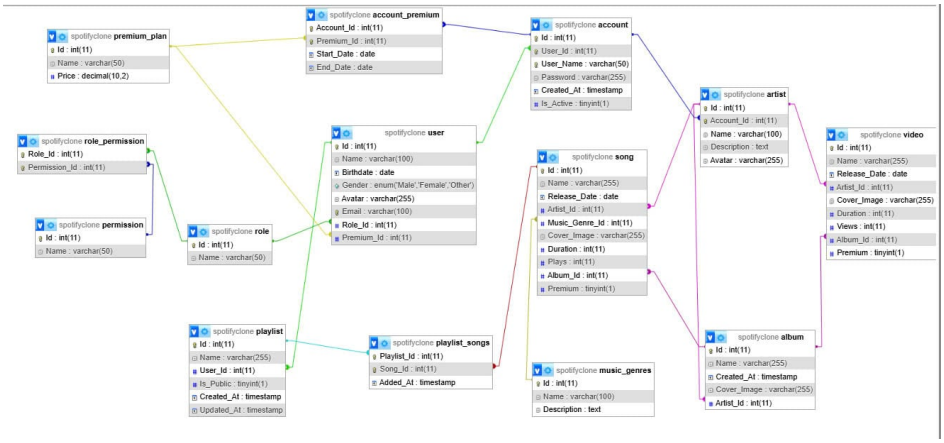
3.3 Chức năng quản trị viên

Quản trị viên có vai trò quản lý toàn bộ hệ thống với các chức năng chính:

- Quản lý tài khoản người dùng và nghệ sĩ.
- Quản lý nội dung: bài hát, video, thể loại, album.
- Phân quyền truy cập.
- Theo dõi hoạt động hệ thống, xử lý báo cáo vi phạm.

3.4 Mô hình ERD (Entity Relationship Diagram)

Hệ thống được phân tích và thiết kế với các thực thể chính như: User, Artist, Song, Video, Playlist, Role, Permission,... Mỗi thực thể có các thuộc tính và mối quan hệ rõ ràng, đảm bảo tính toàn vẹn và dễ mở rộng.



Hình 1: Mô hình ERD

4 Thiết kế hệ thống

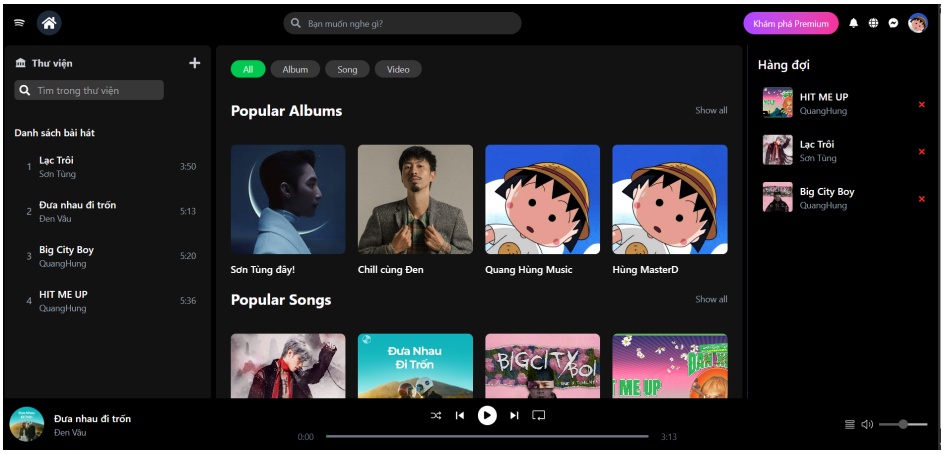
Sau khi phân tích yêu cầu và xác định các chức năng cần thiết, nhóm tiến hành thiết kế tổng thể hệ thống bao gồm giao diện người dùng, cơ sở dữ liệu và API giao tiếp giữa frontend và backend.

4.1 Thiết kế giao diện

Giao diện người dùng được thiết kế theo hướng đơn giản, trực quan và thân thiện với người sử dụng. Nhóm sử dụng thư viện CSS Tailwind kết hợp với ReactJS để tạo ra trải nghiệm người dùng mượt mà và hiện đại.

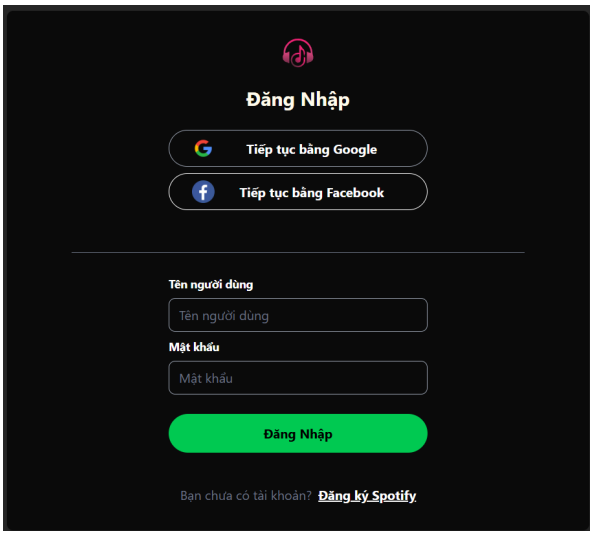
Một số giao diện chính:

- Giao diện trang chủ hiển thị danh sách bài hát, nghệ sĩ nổi bật.



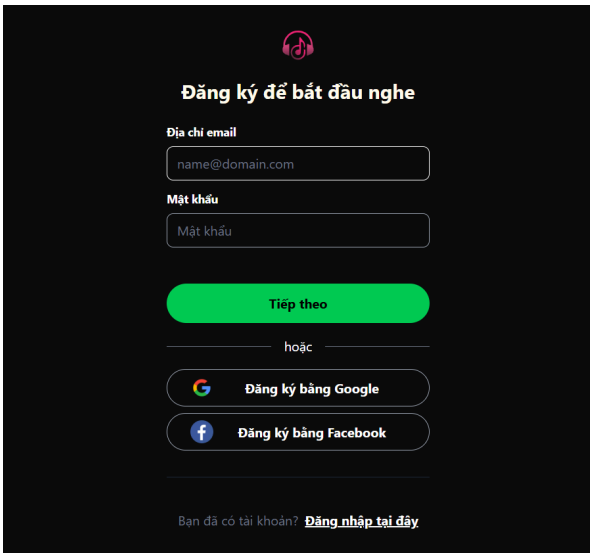
Hình 2: Giao diện trang chủ

- Giao diện đăng nhập



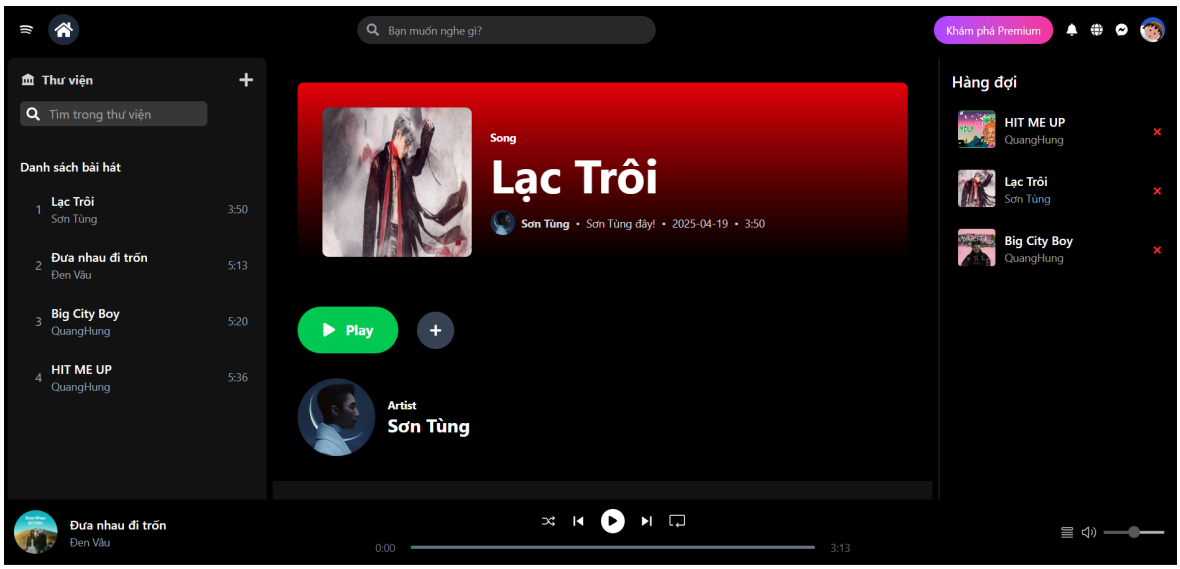
Hình 3: Giao diện đăng nhập

- Giao diện đăng ký



Hình 4: Giao diện đăng ký

- Giao diện nghe nhạc

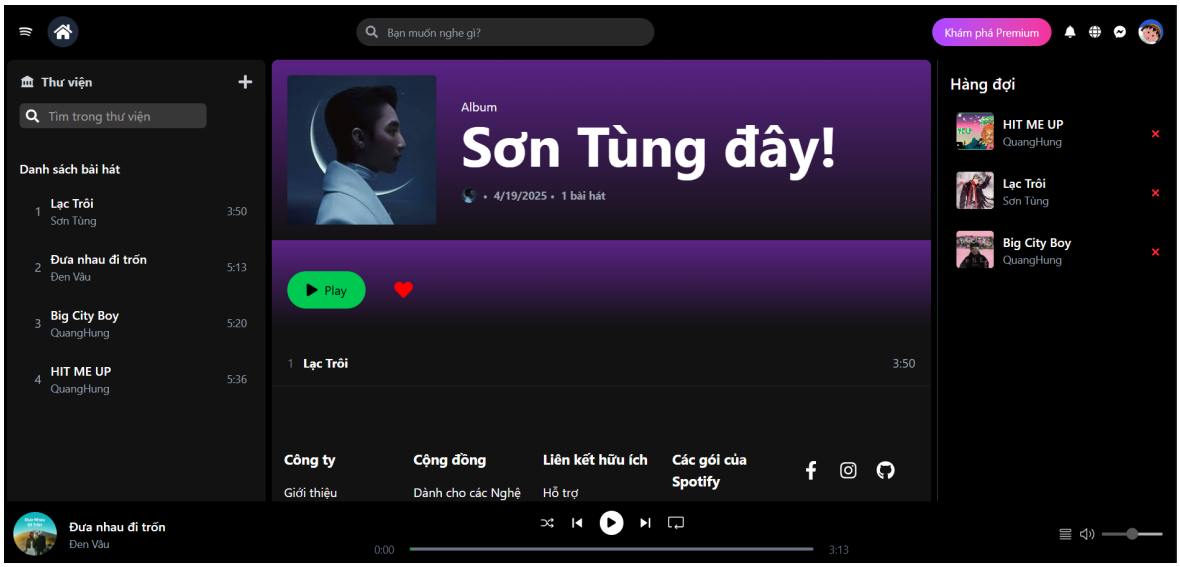


Hình 5: Giao diện nghe bài hát

- Giao diện xem video

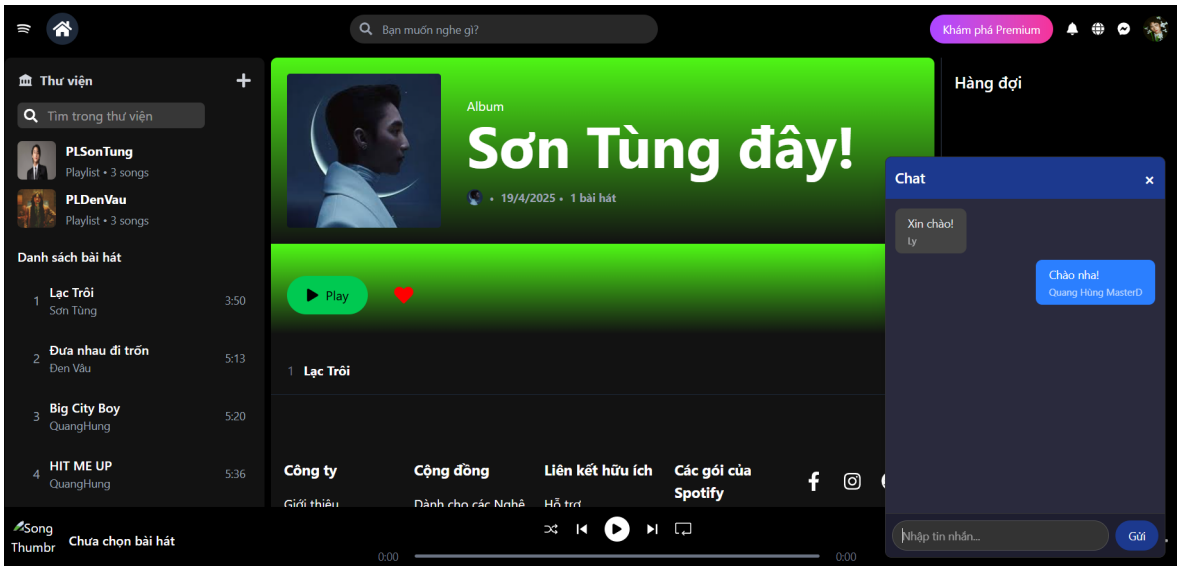
Hình 6: Giao diện xem video

- Giao diện album



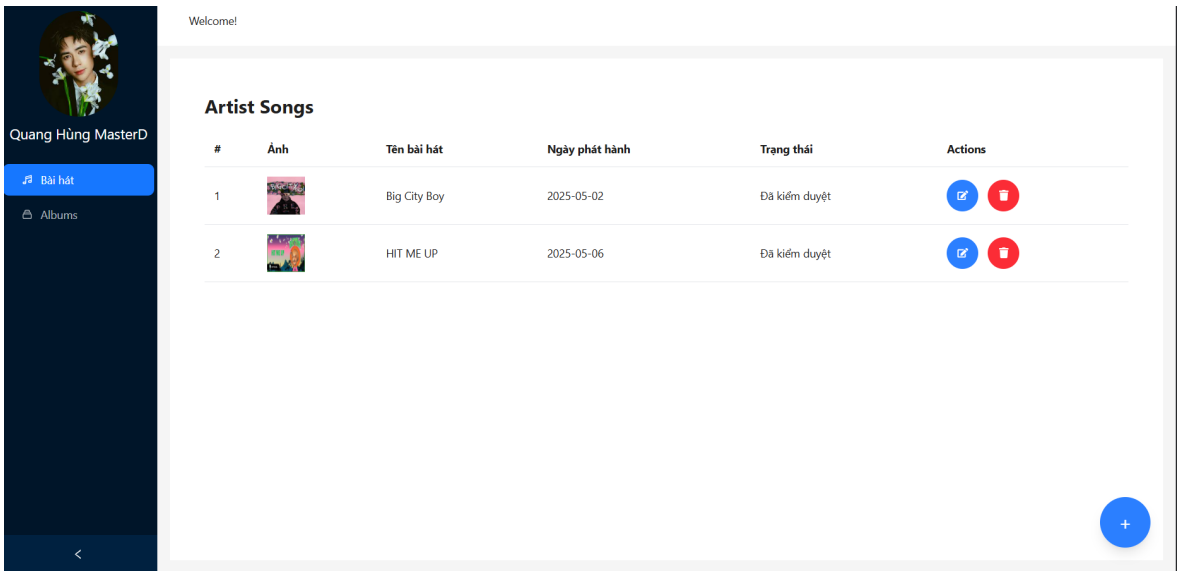
Hình 7: Giao diện album

- Giao diện tích hợp chức năng chat



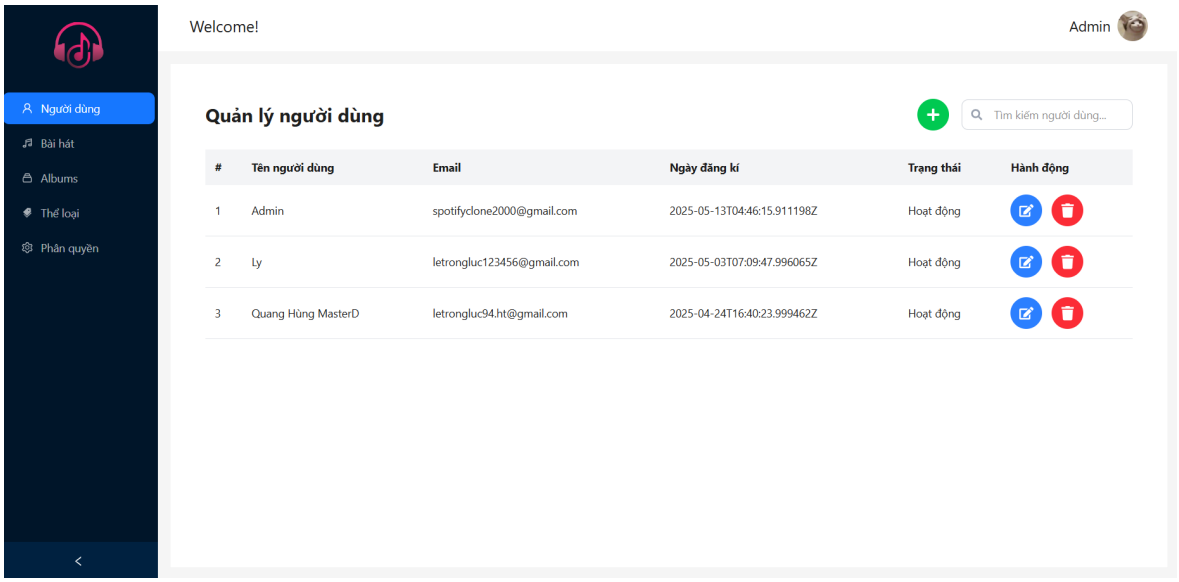
Hình 8: Giao diện album

- Giao diện quản lý bài hát cho nghệ sĩ.



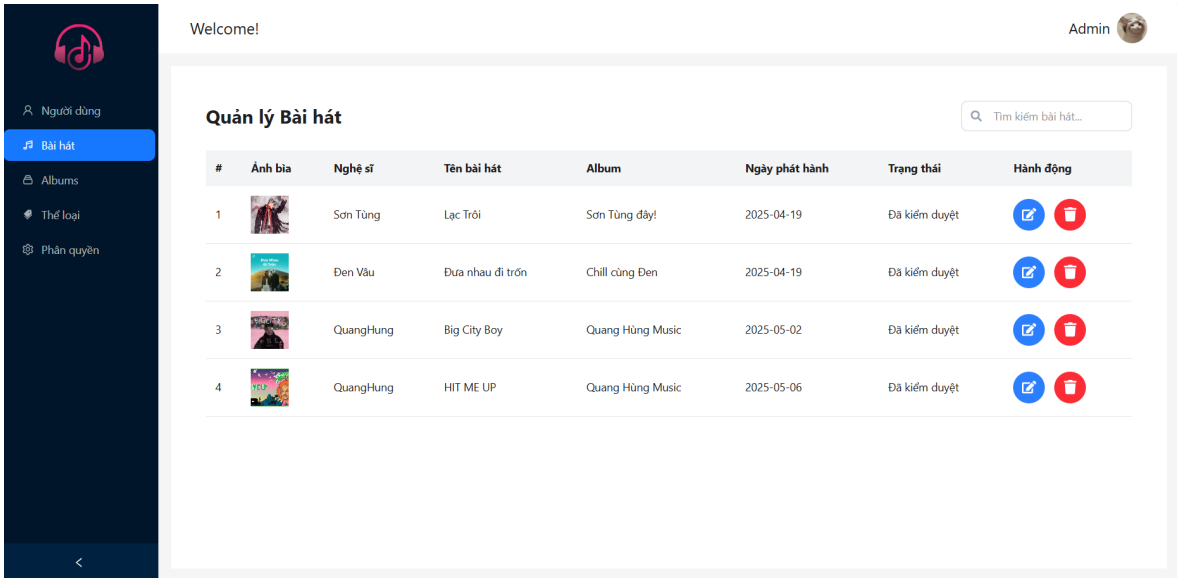
Hình 9: Giao diện quản lý bài hát của nghệ sĩ

- Giao diện quản lý hệ thống dành cho quản trị viên.
  - Giao diện quản lý người dùng



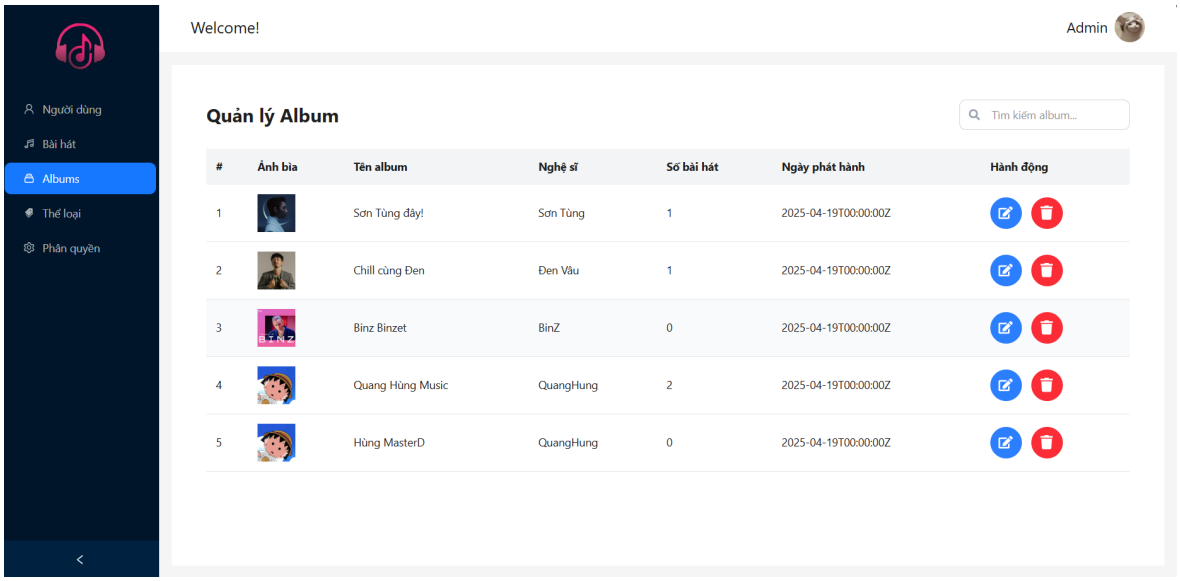
Hình 10: Giao diện quản lý người dùng

  - Giao diện quản lý bài hát



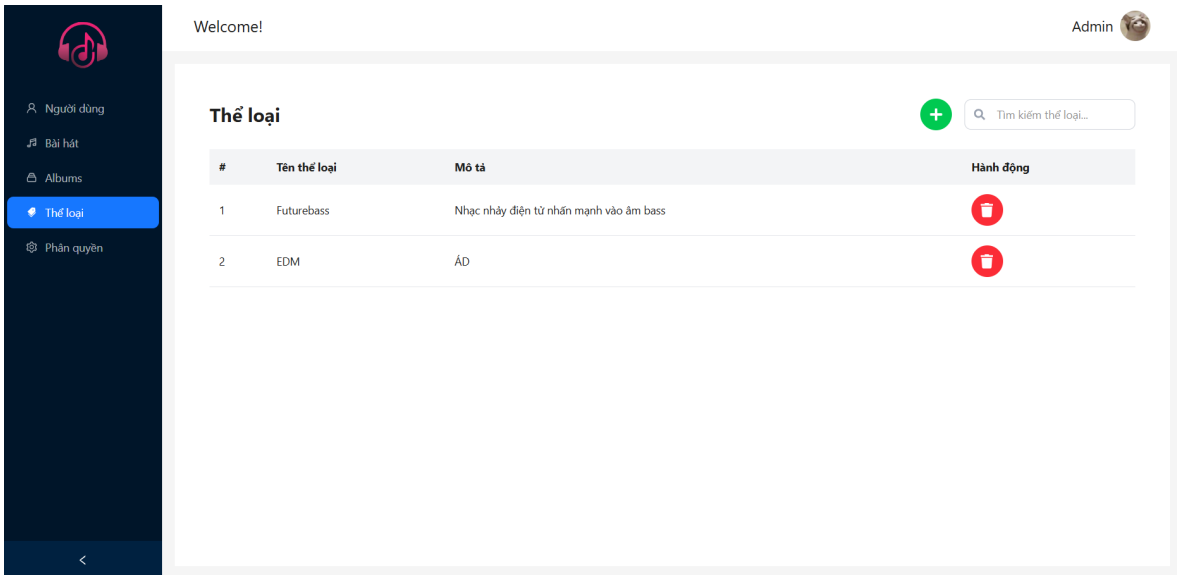
Hình 11: Giao diện quản lý bài hát

– Giao diện quản lý Album



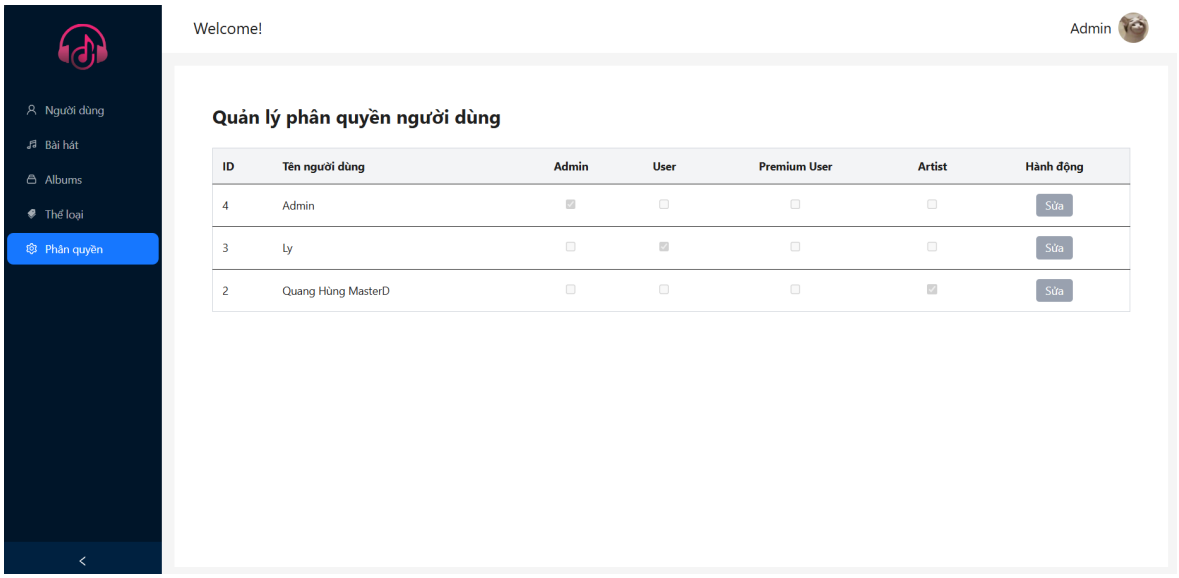
Hình 12: Giao diện quản lý Album

– Giao diện quản lý thể loại



Hình 13: Giao diện quản lý Thể loại

– Giao diện quản lý phân quyền



Hình 14: Giao diện quản lý Phân quyền

4.2 Thiết kế cơ sở dữ liệu

Hệ thống sử dụng cơ sở dữ liệu **MySQL** với các bảng và trường dữ liệu như sau:

- **account**
  - id : int(11)
  - user_id : int(11)
  - user_name : varchar(50)
  - password : varchar(255)
  - created_at : timestamp
  - is_active : tinyint(1)
- **user**
  - id : int(11)
  - name : varchar(100)
  - birthdate : date
  - gender : enum('Male', 'Female', 'Other')
  - avatar : varchar(255)
  - email : varchar(100)
  - role_id : int(11)
  - premium_id : int(11)
- **artist**
  - id : int(11)

  - account_id : int(11)
  - name : varchar(100)
  - description : text
  - avatar : varchar(255)

- **song**
  - id : int(11)
  - name : varchar(255)
  - release_date : date
  - artist_id : int(11)
  - music_genre_id : int(11)
  - cover_image : varchar(255)
  - duration : int(11)
  - plays : int(11)
  - album_id : int(11)
  - premium : tinyint(1)

- **video**
  - id : int(11)
  - name : varchar(255)
  - release_date : date
  - artist_id : int(11)
  - cover_image : varchar(255)
  - duration : int(11)
  - views : int(11)
  - album_id : int(11)
  - premium : tinyint(1)

- **album**
  - id : int(11)
  - name : varchar(255)
  - created_at : timestamp
  - cover_image : varchar(255)
  - artist_id : int(11)

- **music_genres**
  - id : int(11)
  - name : varchar(100)

  - description : text

- **playlist**
  - id : int(11)
  - name : varchar(255)
  - user_id : int(11)
  - is_public : tinyint(1)
  - created_at : timestamp
  - updated_at : timestamp
- **playlist_songs**
  - playlist_id : int(11)
  - song_id : int(11)
  - added_at : timestamp
- **role**
  - id : int(11)
  - name : varchar(50)
- **permission**
  - id : int(11)
  - name : varchar(50)
- **role_permission**
  - role_id : int(11)
  - permission_id : int(11)
- **premium_plan**
  - id : int(11)
  - name : varchar(50)
  - price : decimal(10,2)
- **account_premium**
  - account_id : int(11)
  - premium_id : int(11)
  - start_date : date
  - end_date : date

4.3 Thiết kế API

API là cầu nối giữa frontend (ReactJS) và backend (Django REST Framework). Các nhóm API chính bao gồm:

- **Auth API:** Đăng ký, đăng nhập, xác thực email, đăng xuất.
- **User API:** Lấy thông tin người dùng, cập nhật thông tin cá nhân.
- **Artist API:** Quản lý bài hát và video của nghệ sĩ.
- **Song & Video API:** Lấy danh sách, chi tiết bài hát/video, tìm kiếm,...
- **Playlist API:** Tạo, cập nhật, xóa playlist cá nhân.
- **Friend & Chat API:** Kết bạn, gửi/nhận tin nhắn real-time.
- **Admin API:** Quản lý tài khoản, nội dung hệ thống.

Việc thiết kế API đảm bảo tuân theo mô hình RESTful, dễ tích hợp và mở rộng.

5 Cài đặt và triển khai

Trong phần này, nhóm trình bày quá trình cài đặt và triển khai hệ thống bao gồm các bước thiết lập backend bằng Django, frontend bằng ReactJS, cũng như kết nối giữa hai phần và triển khai thử nghiệm hệ thống.

5.1 Cài đặt backend Django

- Cài đặt môi trường ảo:

```
python -m venv venv
source venv/bin/activate (Linux/Mac)
venv\Scripts\activate (Windows)
```

- Cài đặt các gói cần thiết:

```
pip install django djangorestframework mysqlclient
pip install djoser django-cors-headers channels
```

- Tạo project và ứng dụng:

```
django-admin startproject spotifyclone
python manage.py startapp accounts
```

- Cấu hình kết nối MySQL trong ‘settings.py‘:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'spotifyclone',
        'USER': 'root',
        'PASSWORD': 'yourpassword',
        'HOST': 'localhost',
        'PORT': '3306',
    }
}
```

- Tạo các model, view và route cần thiết.
- Chạy migrate và khởi động server:

```
python manage.py makemigrations
python manage.py migrate
python manage.py runserver
```

5.2 Cài đặt frontend React

- Tạo project bằng Vite:

```
npm create vite@latest spotify-frontend --template react
cd spotify-frontend
npm install
```

- Cài đặt các thư viện:

```
npm install axios react-router-dom tailwindcss
npm install @heroicons/react
npx tailwindcss init -p
```

- Cấu hình tailwind trong ‘tailwind.config.js‘ và ‘index.css‘.
- Tạo cấu trúc thư mục: components, pages, services, layouts, routes,...
- Thiết lập kết nối API bằng ‘axios‘.
- Khởi động ứng dụng:

```
npm run dev
```

5.3 Kết nối frontend-backend

- Backend bật CORS để frontend truy cập:

```
INSTALLED_APPS = [..., 'corsheaders', ...]
MIDDLEWARE = ['corsheaders.middleware.CorsMiddleware', ...]
CORS_ALLOWED_ORIGINS = ['http://localhost:5173']
```

- Frontend gửi request đến backend qua axios với base URL cấu hình sẵn.
- Sử dụng JWT để xác thực người dùng giữa frontend và backend.

5.4 Triển khai thử nghiệm

- Chạy thử hệ thống trên máy cá nhân với React frontend và Django backend.
- Kiểm tra các chức năng cơ bản: đăng ký, đăng nhập, phát nhạc, quản lý bài hát,...
- Có thể triển khai online bằng:
  - **Frontend**: Vercel, Netlify
  - **Backend**: Render, Railway, hoặc server ảo (VPS)
  - **Database**: PlanetScale hoặc dịch vụ MySQL online

6 Kiểm thử

Kiểm thử là bước quan trọng nhằm đảm bảo hệ thống hoạt động đúng theo yêu cầu và phát hiện lỗi trong quá trình phát triển. Nhóm đã thực hiện kiểm thử chức năng và giao diện để đảm bảo chất lượng hệ thống.

6.1 Kiểm thử chức năng

Nhóm đã thực hiện kiểm thử chức năng đối với các vai trò khác nhau trong hệ thống:

- **Người dùng (User):**
  - Đăng ký tài khoản, đăng nhập và đăng xuất.
  - Nghe nhạc, xem video nhạc.
  - Tìm kiếm bài hát, nghệ sĩ.
  - Thêm bài hát vào danh sách yêu thích.
  - Quản lý playlist cá nhân.
- **Nghệ sĩ (Artist):**
  - Đăng nhập vào hệ thống.
  - Thêm/sửa/xoá bài hát, album của mình.
  - Xem danh sách bài hát đã đăng tải.
- **Quản trị viên (Admin):**
  - Quản lý người dùng và nghệ sĩ.
  - Quản lý toàn bộ danh sách bài hát và video.
  - Phê duyệt nội dung bài hát (nếu có chức năng này).

6.2 Kiểm thử giao diện

- Giao diện được kiểm thử trên trình duyệt Chrome, Firefox và Microsoft Edge.
- Đảm bảo hiển thị đúng layout, font chữ, màu sắc và hình ảnh.
- Tương thích với độ phân giải 1366x768 trở lên.
- Các nút bấm, form, thanh điều hướng hoạt động đúng theo mong đợi.
- Trải nghiệm người dùng được đánh giá tốt, thân thiện và dễ sử dụng.

6.3 Kết quả kiểm thử

- Các chức năng cơ bản hoạt động ổn định.
- Giao diện phản hồi tốt, ít lỗi xảy ra trong quá trình thử nghiệm.
- Một số lỗi nhỏ được phát hiện và xử lý kịp thời như lỗi hiển thị tên bài hát dài, lỗi upload khi tệp quá lớn,...

7 Kết luận và hướng phát triển

7.1 Kết quả đạt được

Qua quá trình thực hiện đề tài, nhóm đã xây dựng thành công một ứng dụng web mang tính mô phỏng nền tảng Spotify, với các chức năng chính bao gồm:

- Đăng ký, đăng nhập và quản lý tài khoản người dùng.
- Tìm kiếm và nghe nhạc trực tuyến, xem video âm nhạc.
- Quản lý danh sách bài hát yêu thích và playlist cá nhân.
- Cho phép nghệ sĩ đăng tải và quản lý các bài hát của mình.
- Trang quản trị cho phép kiểm soát hệ thống và người dùng.

Đề tài không chỉ giúp nhóm củng cố kiến thức về lập trình frontend (ReactJS), backend (Django) và cơ sở dữ liệu (MySQL), mà còn nâng cao kỹ năng làm việc nhóm, phân chia công việc và giải quyết vấn đề thực tế trong quá trình xây dựng hệ thống phần mềm.

7.2 Hạn chế

Mặc dù đã cố gắng hoàn thiện đề tài, hệ thống vẫn còn một số hạn chế:

- Giao diện chưa tối ưu cho các thiết bị di động.
- Tính năng upload video chưa xử lý tốt với tệp có dung lượng lớn.
- Thiếu chức năng thống kê và báo cáo cho quản trị viên.
- Chưa triển khai được chat thời gian thực giữa người dùng.

7.3 Hướng phát triển trong tương lai

Trong tương lai, nhóm mong muốn phát triển thêm các tính năng sau để hoàn thiện hệ thống:

- Tối ưu trải nghiệm người dùng trên thiết bị di động (responsive design).
- Triển khai chức năng phát nhạc theo thuật toán đề xuất cá nhân hóa.
- Thêm tính năng bình luận, chia sẻ và đánh giá bài hát.
- Hỗ trợ upload video/audio trực tiếp lên cloud (AWS, Cloudinary,...).
- Xây dựng hệ thống chat real-time giữa người dùng và nghệ sĩ.
- Bảo mật nâng cao với xác thực hai yếu tố và mã hóa dữ liệu.

Nhóm hy vọng đề tài sẽ tiếp tục được phát triển và ứng dụng vào thực tế trong tương lai gần.