

TRƯỜNG ĐẠI HỌC SÀI GÒN
KHOA CÔNG NGHỆ THÔNG TIN



Phát triển phần mềm mã nguồn mở

Phát triển phần mềm Spotify Clone

GVHD: Từ Lãng Phiêu
SV: Bùi Công Thạch - 3121410445
Cao Huy Hưng - 3121410237
Bùi Công Tuấn - 3121410553

Email liên hệ nhóm: buithach.it@gmail.com

TP. HỒ CHÍ MINH, THÁNG 5/2025

NHẬN XÉT CỦA GIẢNG VIÊN



LỜI CẢM ƠN

Trước hết, nhóm chúng em xin gửi lời cảm ơn chân thành đến Trường Đại học Sài Gòn, Khoa Công nghệ Thông tin đã tạo điều kiện để chúng em thực hiện tiểu luận này.

Chúng em xin bày tỏ lòng biết ơn sâu sắc đến giảng viên hướng dẫn, người đã tận tình hỗ trợ, định hướng và góp ý để chúng em có thể hoàn thiện đề tài này.

Bên cạnh đó, chúng em cũng xin gửi lời cảm ơn đến các doanh nghiệp, chuyên gia đã chia sẻ thông tin, kinh nghiệm thực tế, giúp chúng em có cái nhìn rõ hơn về việc phát triển phần mềm mã nguồn mở. Cuối cùng, chúng em xin cảm ơn gia đình, bạn bè đã luôn động viên, hỗ trợ trong suốt quá trình thực hiện đề tài này.

Nhóm chúng em rất mong nhận được ý kiến đóng góp để hoàn thiện hơn nữa nghiên cứu này.



LỜI MỞ ĐẦU

Trong thời đại công nghệ số phát triển không ngừng, các ứng dụng phát nhạc trực tuyến đã trở thành một phần quen thuộc trong đời sống giải trí của con người. Spotify – một trong những nền tảng phát nhạc phổ biến nhất hiện nay – không chỉ thu hút người dùng bởi kho nhạc phong phú mà còn bởi giao diện hiện đại và trải nghiệm người dùng thân thiện. Từ đó, nhóm chúng em đã chọn thực hiện đề tài Spotify Clone như một cách học hỏi và rèn luyện kỹ năng xây dựng ứng dụng thực tế.

Trong quá trình thực hiện, nhóm đã có cơ hội tiếp cận và ứng dụng các công nghệ mã nguồn mở như React, TypeScript, Django và GitHub. Đề tài được triển khai theo mô hình SPA, kết hợp giữa frontend, backend và WebSocket, giúp nhóm hiểu rõ hơn về cách các thành phần trong hệ thống tương tác với nhau. Các tính năng như phát nhạc, tìm kiếm, quản lý danh sách phát và trò chuyện thời gian thực đã được tích hợp nhằm tái hiện một phần trải nghiệm thực tế từ Spotify.

Thông qua đồ án này, nhóm không chỉ cống hiến kiến thức đã học mà còn nâng cao kỹ năng làm việc nhóm, quản lý tiến độ dự án, tìm hiểu công nghệ mới và giải quyết các vấn đề phát sinh trong quá trình phát triển phần mềm.

Chúng em xin chân thành cảm ơn thầy đã tạo điều kiện để nhóm thực hiện đề tài này. Mặc dù đã cố gắng hoàn thiện trong khả năng của mình, đồ án vẫn không tránh khỏi những thiếu sót. Nhóm rất mong nhận được sự góp ý để cải thiện tốt hơn trong tương lai.



Mục lục

1 Giới thiệu đề tài	5
1.1 Giới thiệu Spotify Clone	5
1.2 Lý do chọn đề tài	5
1.3 Mục tiêu	5
1.4 Đối tượng và phạm vi sử dụng	5
2 CÔNG NGHỆ VÀ PHẦN MỀM MÃ NGUỒN MỞ SỬ DỤNG	6
2.1 Tổng quan dự án	6
2.2 Kiến trúc Frontend	6
2.3 Kiến trúc Backend	7
2.4 Tính năng chat	7
3 PHÂN TÍCH VÀ THIẾT KẾ SPOTIFY CLONE	7
3.1 Mô hình ứng dụng	7
3.2 Các tính năng được xây dựng	8
3.3 Xây dựng Database	8
4 CÁCH CÀI ĐẶT ỨNG DỤNG	9
4.1 Yêu cầu hệ thống	9
4.2 Frontend	9
4.3 Backend	9
5 CẤU TRÚC VÀ MÔ HÌNH	10
5.1 Cấu trúc thư mục của Project	10
5.2 Thư mục Backend	11
5.3 Thư mục Frontend	12
5.4 Routing và phân chia các module	13
6 NHIỆM VỤ VÀ VAI TRÒ THÀNH VIÊN	16
7 KẾT QUẢ ĐẠT ĐƯỢC	17
7.1 Màn hình trang Home	17
7.2 Màn hình Đăng nhập/Dăng ký	17
7.3 Artist component	18
7.4 Search component	19
7.5 Modal premium	19
7.6 Quản lý Người dùng	20
7.7 Quản lý bài hát	20
7.8 Quản lý ca sĩ	21
7.9 AI chat	21



1 Giới thiệu đề tài

1.1 Giới thiệu Spotify Clone

Spotify Clone là một ứng dụng nghe nhạc trực tuyến được xây dựng với mục tiêu mô phỏng lại giao diện và chức năng cơ bản của Spotify – nền tảng phát nhạc trực tuyến nổi tiếng toàn cầu. Ứng dụng cho phép người dùng nghe nhạc, xem danh sách bài hát, tìm kiếm bài hát và phát bài hát theo thời gian thực với trải nghiệm thân thiện. Dự án này được phát triển theo mô hình SPA (Single Page Application) sử dụng Angular ở phía frontend, Django ở phía backend và WebSocket cho việc xử lý dữ liệu thời gian thực.

1.2 Lý do chọn đề tài

Lý do chọn đề tài Trong thời đại công nghệ số phát triển mạnh mẽ, nhu cầu nghe nhạc trực tuyến ngày càng trở nên phổ biến và thiết yếu trong đời sống hằng ngày. Các nền tảng như Spotify không chỉ đơn thuần là công cụ giải trí mà còn trở thành một phần trong thói quen và phong cách sống của người dùng hiện đại. Tuy nhiên, để xây dựng một ứng dụng nghe nhạc trực tuyến chuyên nghiệp, hiện đại và có tính tương tác cao như Spotify đòi hỏi nhiều kiến thức liên quan đến lập trình giao diện, xử lý âm thanh, quản lý cơ sở dữ liệu và tối ưu hiệu năng.

Vì vậy, em chọn đề tài Spotify Clone với mục tiêu:

Tìm hiểu và thực hành quy trình xây dựng một ứng dụng nghe nhạc từ đầu đến cuối.

Áp dụng các kiến thức về phát triển phần mềm như UI/UX, kết nối backend, quản lý media và phát triển tính năng tương tác người dùng.

Rèn luyện khả năng thiết kế hệ thống phần mềm có cấu trúc rõ ràng, dễ mở rộng và nâng cấp trong tương lai.

Đề tài không chỉ mang tính thực tiễn cao mà còn giúp em phát triển kỹ năng tư duy hệ thống, làm việc với công nghệ hiện đại và chuẩn bị nền tảng tốt cho các dự án phần mềm quy mô lớn hơn sau này.

1.3 Mục tiêu

Thiết kế giao diện web phần mềm Spotify Clone và lập trình bằng ngôn ngữ Python.

Front end: React hoặc Angular

Backend: Django

Database: MySQL

Các tính năng cơ bản cho ứng dụng sử dụng trên mã nguồn mở Linux. Yêu cầu phát triển phần mềm Spotify Clone:

1. Chức năng phát nhạc:
2. Chức năng phát video âm nhạc:
3. Chức năng tải video âm nhạc:
4. Chức năng User tạo album, bài hát yêu thích
5. Trang Admin.
6. Optional: Có thêm tính năng chat tích hợp trong giao diện web spotify clone.

1.4 Đối tượng và phạm vi sử dụng

Đối tượng sử dụng: người dùng cá nhân, sinh viên, lập trình viên học tập và trải nghiệm giao diện nghe nhạc hiện đại.



Phạm vi: ứng dụng chủ yếu tập trung vào frontend với một số chức năng backend mô phỏng như phát nhạc, danh sách bài hát và xử lý trạng thái phát. Chưa tích hợp thanh toán, tải bài hát, đề xuất AI hoặc tính năng bảo mật cao.

2 CÔNG NGHỆ VÀ PHẦN MỀM MÃ NGUỒN MỞ SỬ DỤNG

2.1 Tổng quan dự án

Dự án Spotify Clone là một ứng dụng web mô phỏng các chức năng chính của Spotify, bao gồm phát nhạc, quản lý playlist, tìm kiếm bài hát và phân quyền người dùng. Ứng dụng được xây dựng với:

Frontend: React (TypeScript), Tailwind CSS, Ant Design

Backend: Django REST Framework, MySQL, JWT Authentication

2.2 Kiến trúc Frontend

Công nghệ sử dụng:

- React : Thư viện JavaScript để xây dựng giao diện người dùng.
- TypeScript: Ngôn ngữ lập trình mở rộng của JavaScript, cung cấp kiểu dữ liệu tĩnh.
- Tailwind CSS: Framework CSS tiện ích-first để thiết kế giao diện nhanh chóng.
- Vite: Công cụ build nhanh chóng cho các dự án frontend.
- React Router DOM: Thư viện định tuyến cho React.
- Axios: Thư viện HTTP client để giao tiếp với backend.
- Ant Design: Thư viện UI component cho React.
- React Spinners: Thư viện spinner cho trạng thái loading.
- SimpleBar React: Thư viện thanh cuộn tùy chỉnh cho React.

Tính năng chính:

- Xác thực người dùng: Đăng nhập/Đăng ký với JWT.
- Phát nhạc: Điều khiển play/pause, next/previous.
- Quản lý playlist: Tạo, chỉnh sửa, xóa playlist.
- Tìm kiếm bài hát: Tìm kiếm theo tên bài hát hoặc nghệ sĩ.
- Điều chỉnh âm lượng: Tăng/giảm âm lượng phát nhạc.
- Chế độ Shuffle và Repeat: Phát ngẫu nhiên hoặc lặp lại bài hát.
- Phân quyền người dùng: Quản trị viên (superuser) có quyền quản lý nội dung.



2.3 Kiến trúc Backend

Công nghệ sử dụng:

- Django: Framework web mạnh mẽ cho Python.
- Django REST Framework: Thư viện xây dựng API RESTful.
- MySQL: Hệ quản trị cơ sở dữ liệu quan hệ.
- JWT Authentication: Xác thực người dùng bằng JSON Web Token.
- CORS Headers: Xử lý Cross-Origin Resource Sharing cho API.

Tính năng chính:

- API RESTful: Cung cấp các endpoint cho frontend tương tác.
- Xác thực và phân quyền: Sử dụng JWT để xác thực và phân quyền người dùng.
- Quản lý dữ liệu: CRUD cho người dùng, bài hát, playlist.
- Bảo vệ bản quyền: Đảm bảo chỉ người dùng có quyền mới truy cập được nội dung.

2.4 Tính năng chat

Đối tượng chat: Người dùng sẽ chat với một hệ thống AI (không phải người thật).
Cụ thể, backend sử dụng mô hình AI (Google GenAI (Gemini 2.0 Flash)) để:

1. Nhận câu hỏi tự nhiên từ người dùng (ví dụ: "Các bài hát trong album Vol 1 là gì?").
2. Sinh ra câu lệnh SQL phù hợp với câu hỏi đó.
3. Thực thi SQL trên cơ sở dữ liệu nhạc (tracks, albums, artists).
4. Trả về kết quả và sinh ra câu trả lời tự nhiên, lịch sự, bằng tiếng Việt cho người dùng.

Luồng xử lý:

- Người dùng gửi câu hỏi qua API /chatbox/.
- Backend dùng AI để chuyển đổi câu hỏi thành SQL, truy vấn dữ liệu, rồi lại dùng AI để trả lời tự nhiên dựa trên kết quả truy vấn.
- Người dùng nhận được câu trả lời như đang trò chuyện với một trợ lý ảo am hiểu về nhạc.

3 PHÂN TÍCH VÀ THIẾT KẾ SPOTIFY CLONE

3.1 Mô hình ứng dụng

Kiến trúc tổng quan

Ứng dụng được xây dựng theo mô hình client-server:



- Frontend: React (Vite, TypeScript, Ant Design)
Giao diện người dùng, routing, gọi API, quản lý state, bảo vệ route, hiển thị dữ liệu nhạc, quản trị, xác thực...
- Backend: Python Django
Xử lý logic nghiệp vụ, quản lý dữ liệu (users, tracks, albums, playlists, chatbox...), cung cấp API RESTful, tích hợp AI Gemini cho chat thông minh.
- Database: MySQL
Lưu trữ dữ liệu nhạc, người dùng, playlist, v.v.

Giao tiếp:

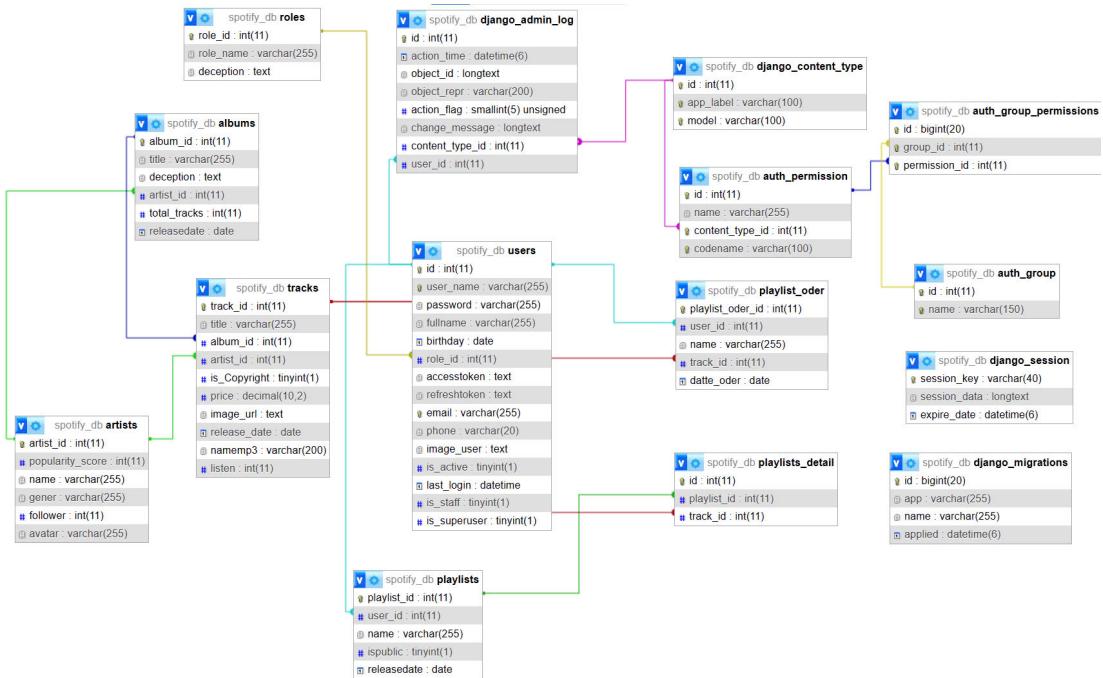
- Frontend gọi API của backend qua HTTP (REST API).
- Backend xử lý, truy vấn database, trả kết quả về frontend.

3.2 Các tính năng được xây dựng

- Tìm kiếm, nghe nhạc, xem thông tin nghệ sĩ, album, playlist.
- Đăng ký, đăng nhập, phân quyền user/admin.
- Quản trị hệ thống (admin): Quản lý users, tracks, artists.
- Chat thông minh với AI Gemini: Hỏi đáp về nhạc, album, ca sĩ.
- Giao diện hiện đại, responsive, trải nghiệm người dùng tốt.
- Bảo vệ route, xác thực, phân quyền chặt chẽ.

3.3 Xây dựng Database

Xây dựng Cơ sở dữ liệu theo mô hình:



4 CÁCH CÀI ĐẶT ỨNG DỤNG

4.1 Yêu cầu hệ thống

Node.js (v14 trở lên)
Python (v3.8 trở lên)
MySQL

4.2 Frontend

```

1 % Di chuyển vào thư mục frontend %
2 cd du-an-fontend-react/spotifyFrontEndProject
3
4 % Cài đặt dependencies %
5 npm install
6
7 % Chạy development server %
8 npm run dev

```

4.3 Backend

```

1 % Di chuyển vào thư mục backend %
2 cd du-an-backend-python-django-be

```



```
3
4      % Tao moi truong ao %
5      python -m venv venv
6
7      % Kich hoat moi truong ao %
8      % Windows
9      venv\Scripts\activate
10     % Linux/Mac %
11     source venv/bin/activate
12
13    % Cai dat dependencies %
14    pip install -r requirements.txt
15
16    % Chay migrations %
17    python manage.py migrate
18
19    % Chay development server %
20    python manage.py runserver
```

5 CẤU TRÚC VÀ MÔ HÌNH

5.1 Cấu trúc thư mục của Project



```
✓ LISTENTOMUSICWEB
  ✓ du-an-backend-python-django-be
    > albums
    > artists
    > chatbox
    > music
    > orders
    > playlist_detail
    > playlists
    > public
    > roles
    > spotify_project
    > tracks
    > users
    ≡ haydoctoidi.txt
    ✎ manage.py
    📄 Procfile
    ≡ requirements.txt
  ✓ du-an-frontend-react\spotifyFrontEnd...
    > public
    > src
    $ .env.development
    $ .env.production
    ♦ .gitignore
    JS eslint.config.js
    <> index.html
    {} package-lock.json
    {} package.json
    JS postcss.config.mjs
    ⓘ README.md
    {} tsconfig.app.json
    TS tsconfig.json
    {} tsconfig.node.json
    TS vite.config.ts
```

5.2 Thư mục Backend

Đây là phần phía máy chủ (server), sử dụng Django – một framework Python mạnh mẽ để xây dựng API, xử lý dữ liệu, xác thực người dùng và giao tiếp với cơ sở dữ liệu. Dựa vào cấu trúc, có thể thấy mỗi thư mục con bên trong là một Django App, đại diện cho từng chức năng riêng biệt:



Tên thư mục	Mô tả chức năng
albums/	Quản lý thông tin các album âm nhạc
artists/	Quản lý thông tin nghệ sĩ
tracks/	Quản lý các bài hát (track)
users/	Quản lý tài khoản, thông tin người dùng
roles/	Quản lý vai trò, phân quyền người dùng
playlists/; playlistdetail/	Quản lý danh sách phát nhạc
chatbox/	Xử lý tính năng trò chuyện
manage.py	Tập tin chính để khởi chạy và quản lý dự án Django

Bảng 1: Thư mục Backend

Mỗi app sẽ có các tệp chuẩn như:

- models.py: định nghĩa bảng dữ liệu (model).
- views.py: viết các xử lý logic.
- urls.py: định nghĩa các API endpoint riêng cho app.
- serializers.py: chuyển đổi dữ liệu từ model sang JSON (nếu dùng Django REST Framework).

5.3 Thư mục Frontend

Dây là dự án frontend chính, được xây dựng bằng React và TypeScript, sử dụng Vite làm công cụ build. Dự án này có cấu trúc thư mục rõ ràng, bao gồm các thư mục như src/ (chứa code chính), public/ (tài nguyên tĩnh), và các file cấu hình như package.json, tsconfig.json, vite.config.ts, v.v.

Tên thư mục	Mô tả chức năng
components/	Xây dựng UI, chia nhỏ theo chức năng.
pages/	Các trang chính, routing.
services/	Xử lý logic, gọi API.
styles/	Quản lý style chung.
types/	Định nghĩa kiểu dữ liệu.

Bảng 2: Thư mục Frontend

Cấu trúc thư mục src/ được tổ chức hợp lý, mỗi phần như pages, components, services, types,... đều đảm nhận một vai trò riêng biệt, giúp mã nguồn rõ ràng, dễ bảo trì và mở rộng.

Thư mục pages/

- Chứa các trang chính của ứng dụng, mỗi trang (ví dụ: home, search, playlist, artist, login, register,...) được tách thành các file/module riêng biệt. Điều này giúp dễ dàng quản lý routing, tối ưu hiệu suất khi chỉ tải những phần cần thiết cho người dùng truy cập từng trang.

Thư mục services/



- Tập trung toàn bộ các logic giao tiếp với API (backend), ví dụ như cấu hình axios, các hàm gọi API. Việc này giúp mã nguồn tách biệt rõ ràng giữa phần giao diện và phần xử lý dữ liệu, dễ kiểm thử và tuân thủ nguyên tắc Separation of Concerns (SoC).

Thư mục components/

- Chứa các component tái sử dụng như layout, header, navbar, footer, các thành phần giao diện nhỏ,... Việc gom nhóm này giúp đảm bảo tính đồng nhất giao diện, tiết kiệm công sức khi phát triển hoặc thay đổi layout sau này.

Thư mục styles/

- Quản lý các file style chung (CSS/SCSS), giúp dễ dàng tùy biến giao diện toàn ứng dụng và tái sử dụng các biến, mixin, style toàn cục.

Thư mục types/

- Định nghĩa các kiểu dữ liệu dùng chung cho toàn dự án, giúp tăng tính an toàn khi lập trình với TypeScript, giảm lỗi khi truyền dữ liệu giữa các thành phần.

5.4 Routing và phân chia các module

Dự án được tổ chức theo kiến trúc module hóa, tức là chia nhỏ hệ thống thành các phần riêng biệt tương ứng với từng chức năng cụ thể. Việc phân chia này được áp dụng ở cả hai phía: Backend (Django) và Frontend (Angular), nhằm tăng tính mở rộng, dễ bảo trì và giúp các thành viên trong nhóm có thể phát triển song song mà không ảnh hưởng đến nhau.

Backend (Django)

Phía backend được xây dựng bằng framework Django, mỗi chức năng được tổ chức dưới dạng một app riêng biệt. Ví dụ: albums, artists, tracks, users, chat,... Mỗi app đóng vai trò như một module độc lập, đảm nhiệm một phần nghiệp vụ của hệ thống như quản lý bài hát, người dùng, hoặc trò chuyện.

Mỗi app sẽ có các file riêng như:

- models.py: định nghĩa bảng dữ liệu
- views.py: xử lý logic nghiệp vụ
- urls.py: định tuyến API cho từng app

Tất cả các tuyến đường (route) của các app sẽ được gom lại trong file backend/urls.py chính bằng cách sử dụng hàm include(), giúp dự án dễ tổ chức và mở rộng.



```
urls.py x
du-an-backend-python-django-be > users > urls.py > ...
thach3112003, 5 days ago | 2 authors (Cao Huy Hưng and one other)
1 # users/urls.py      Cao Huy Hưng, 3 weeks ago • test
2 from django.urls import path, include
3 from .views import *
4 urlpatterns = [
5     path('register/', RegisterUserView.as_view(), name='register'),#push
6     path('login/', LoginView.as_view(), name='user-login'),
7     path('refresh/', RefreshTokenView.as_view(), name='token_refresh'),
8     path('<int:pk>/update/', UpdateUserView.as_view(), name='user-update'),
9     path('<int:pk>/delete/', DeleteUserView.as_view(), name='user-delete'),
10    path('list/', ListUsersView.as_view(), name='list-users'),
11    path('getuserbyid/<int:user_id>', GetUserByIdView, name='usersbyid'),
12    path('token/<str:accesstoken>', GetUserByTokenView, name='usersbyToken'),
13    path('profile/', UserProfileView.as_view(), name='user-profile'),
14    path('logout/', LogoutView.as_view(), name='logout'),
15 ]
16
```

Nhờ đó, khi frontend gọi các API như GET : /users/list/ , Django sẽ biết được phải định tuyến đến app users

Frontend (React)

Ứng dụng sử dụng thư viện react-router-dom để quản lý routing.

Tất cả các route được khai báo tập trung trong file src/main.tsx thông qua createBrowserRouter.

Routing (Định tuyến)

Phân chia route chính:

Người dùng thông thường:

- / → Trang chủ (HomePage)
- /search → Trang tìm kiếm (SearchPage)
- /playlist/:playlistId/:playlistName → Trang chi tiết playlist
- /artist/:artistId/:artistName → Trang nghệ sĩ
- /login, /register → Đăng nhập, đăng ký

Trang đặc biệt:

- /checkout → Trang thanh toán (yêu cầu đăng nhập, dùng ProtectedRoute)

Quản trị (Admin):

- /admin → Layout quản trị (yêu cầu đăng nhập, dùng ProtectedRoute)
- /admin/user → Quản lý người dùng
- /admin/track → Quản lý bài hát
- /admin/singer → Quản lý ca sĩ



```
29 const router = createBrowserRouter([
30   {
31     path: "/",
32     element: <Layout />,
33     children: [
34       {
35         index: true,
36         element: <HomePage />
37       },
38       {
39         path: "search",
40         element: <SearchPage />
41       },
42       {
43         path: "/playlist/:playlistId/:playlistName",
44         element: <PlaylistPage />
45       },
46       {
47         path: "/artist/:artistId/:artistName",
48         element: <ArtistPage />
49       }
50     ],
51   },
52   {
53     path: "/login",
54     element: <LoginPage />,
55   },
56   {
57     path: "/register",
58     element: <RegisterPage />
59   },
60   {
61     path: "/checkout",
62     element: (
63       <ProtectedRoute>
64         <div>Checkout Page</div>
65       </ProtectedRoute>
66     )
67   }
68 ]);
69 
```

Phân chia module

- pages/:
Chứa các module trang chính, mỗi trang là một module riêng biệt, ví dụ:
- client/ (trang cho người dùng): home, search, playlist, artist, login, register...
- admin/ (trang quản trị): user, track, singer...
- components/: Chứa các thành phần giao diện tái sử dụng, chia nhỏ theo chức năng:
 - layout/: Các layout tổng thể (header, sidebar, playbar, ...).
 - auth/: Các component liên quan xác thực, bảo vệ route.
 - admin/: Các component phục vụ riêng cho giao diện quản trị.
- services/: Tập trung toàn bộ logic giao tiếp với API, cấu hình axios, giúp tách biệt phần giao diện và xử lý dữ liệu.
- context/: Quản lý state toàn cục (người dùng, player, album...) thông qua React Context.

Một số lợi ích:

Routing được tổ chức rõ ràng, tách biệt giữa các nhóm chức năng (user, admin, auth).
Module được chia nhỏ theo từng chức năng, giúp dễ bảo trì, mở rộng, kiểm thử và tối ưu hiệu suất.
Bảo vệ truy cập được thực hiện chặt chẽ qua ProtectedRoute.



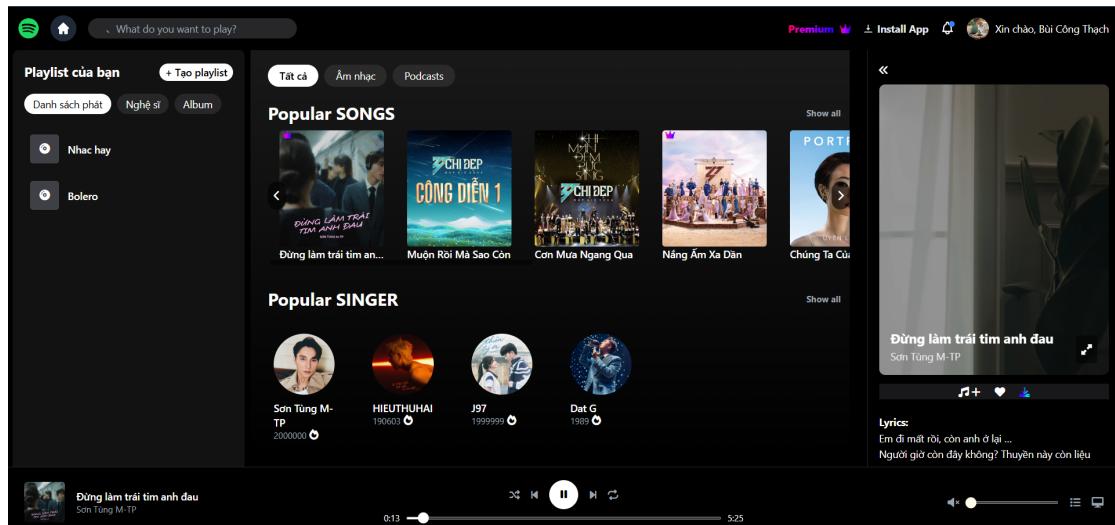
6 NHIỆM VỤ VÀ VAI TRÒ THÀNH VIÊN

MSSV	HỌ VÀ TÊN	NHIỆM VỤ
3121410237	CAO HUY HƯNG	Backend
3121410445	BÙI CÔNG THẠCH	Frontend
3121410553	BÙI CÔNG TUẤN	Latex, Database, Quản lý Role



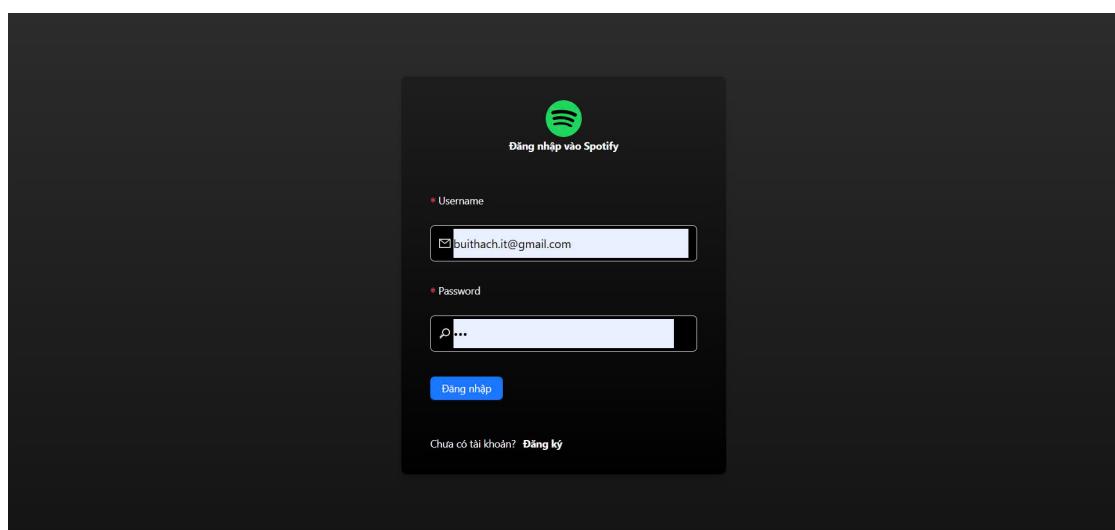
7 KẾT QUẢ ĐẠT ĐƯỢC

7.1 Màn hình trang Home



Hình 1: HomePage

7.2 Màn hình Đăng nhập/Đăng ký



Hình 2: Đăng nhập



Đăng ký để
bắt đầu nghe

* Địa chỉ email

Vui lòng nhập email!

* Họ tên

* Số điện thoại

* Mật khẩu

Vui lòng nhập mật khẩu!

Đăng ký

hoặc

G Đăng ký bằng Google

F Đăng ký bằng Facebook

A Đăng ký bằng Apple

Bạn đã có tài khoản? [Đăng nhập tại đây](#)

Hình 3: Đăng ký

7.3 Artist component

The screenshot shows an artist profile for Sơn Tùng M-TP on a dark-themed interface. At the top, there's a search bar with the placeholder "What do you want to play?", a "Premium" badge, an "Install App" button, and a user icon with the name "Xin chào, Bùi Công Thạch". Below the search bar, there are navigation links for "Playlist của bạn", "+ Tạo playlist", "Danh sách phát", "Nghệ sĩ", and "Album". The main area features a large circular profile picture of Sơn Tùng M-TP and his name "Sơn Tùng M-TP" in bold capital letters, with "7 songs" listed below. To the right, there's a sidebar for "MP ENTERTAINMENT" featuring the logo and a preview of the song "Đừng làm trái tim anh đau". The main content area lists five songs with their covers, titles, artists, and play counts:

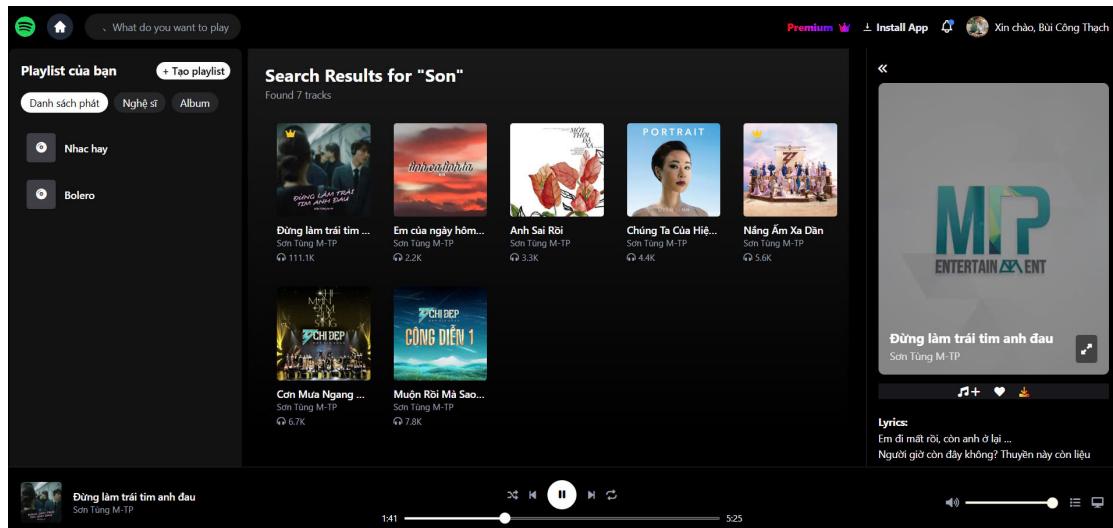
Tên bài hát	Nhạc sĩ	Play count
Đừng làm trái tim anh đau	Sơn Tùng M-TP	111,112
Em của ngày hôm qua	Sơn Tùng M-TP	2,222
Anh Sai Rồi	Sơn Tùng M-TP	3,333
Chúng Ta Của Hiện Tại	Sơn Tùng M-TP	4,444
Nắng Ấm Xa Đàn	Sơn Tùng M-TP	5,555

At the bottom, there's a playback control bar showing the song "Đừng làm trái tim anh đau" by Sơn Tùng M-TP, with a progress bar from 0:44 to 5:25.

Hình 4: Artist component

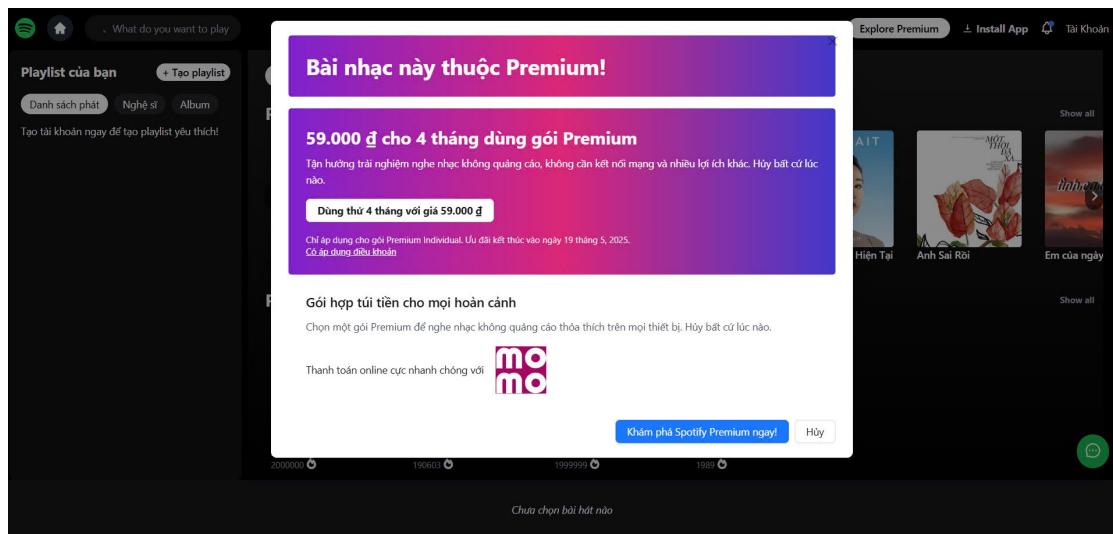


7.4 Search component



Hình 5: Search component

7.5 Modal premium



Hình 6: Mua Premium



7.6 Quản lý Người dùng

ID	Username	Full Name	Email	Phone	Role	Action
8	huyhung	cao huy hung	huyhungdp@gmail.com	111232123	2	
9	buiithach	bui cong thach	buiithach@gmail.com	12312	2	
14	buiithach.it@gmail.com	Bùi Công Thạch		0776711376	1	
24	test1@gmail.com	Hung cao		0776711376	2	
25	123	123	admin@gmail.com	123123	2	

Hình 7: Manage user component

7.7 Quản lý bài hát

ID	Title	Artist	Image	Price	Listens	Action
11	Đừng làm trái tim anh đau	Sơn Tùng M-TP		\$0.00	111112	
20	Muộn Rồi Má Sao Còn	Sơn Tùng M-TP		\$0.00	7777	
19	Cơn Mưa Ngang Qua	Sơn Tùng M-TP		\$0.00	6666	
18	Nắng Ấm Xa Distant	Sơn Tùng M-TP		\$0.00	5555	
17	Chúng Ta Của Hiện Tại	Sơn Tùng M-TP		\$0.00	4444	

Hình 8: Manage track component



7.8 Quản lý ca sĩ

The screenshot shows a web-based application interface for managing singers. On the left, there is a sidebar with navigation links: Admin, Dashboard, Manage Users (with CRUD sub-options), Manage tracks, and Manage singers (which is highlighted). The main area has a header with 'Xin chào, Bùi Công Thạch'. Below the header is a button '+ Add Artist'. A table lists six artists:

ID	Name	Genre	Avatar	Popularity Score	Followers	Action
1	Sơn Tùng M-TP	Pop		90	2000000	
2	HIEUTHUHAI	Bolero		70	190603	
5	J97	Người ngoại lai		190561	1999999	
6	Dat G	Hát thi tham dam thi đấu		1000	1989	

At the bottom right of the table, there are navigation arrows for pagination. The footer of the page shows 'Bùi Thach' and a small profile icon.

Hình 9: Manage singer component

7.9 AI chat

The screenshot shows a mobile application interface with a dark theme. At the top, there are links for 'Premium' with a crown icon, 'Install App', and a user profile with the name 'Xin chào, Bùi Công Thạch'. Below this is a grid of album covers:

- Nắng Ấm Xa Distant (by Nắng Ấm Xa Distant)
- Chúng Ta Của Hiện (by UYÊN LINH)
- Dat G (by Dat G - 1989)

A floating AI Assistant box is open on the right side, containing the text: 'Xin chào! Tôi là trợ lý AI của Spotify. Tôi có thể giúp bạn tìm kiếm bài hát, nghệ sĩ hoặc album. Bạn muốn tìm gì?'. Below the box is a text input field with placeholder text 'Nhập tin nhắn của bạn...' and a green send button with a speech bubble icon.

Hình 10: AI chatbox



Tài liệu

- [1] What is a large language model (LLM)? link: <https://aws.amazon.com/vi/what-is/large-language-model/>
- [2] Gemini 2.0 Flash link: <https://apidog.com/vi/blog/google-gemini-2-0-api-vi/>