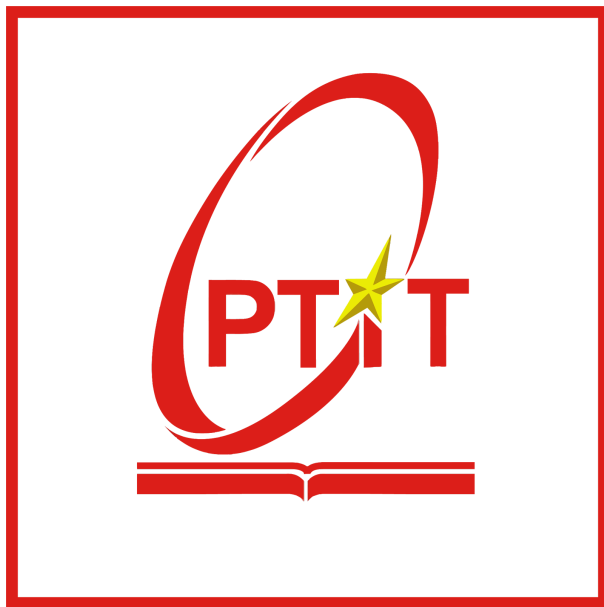


**BỘ THÔNG TIN VÀ TRUYỀN THÔNG
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**



BÁO CÁO THỰC TẬP CƠ SỞ TUẦN 8

**TRIỂN KHAI
UX/UI VÀ FRONT END**

Giảng viên hướng dẫn: TS. Kim Ngọc Bách

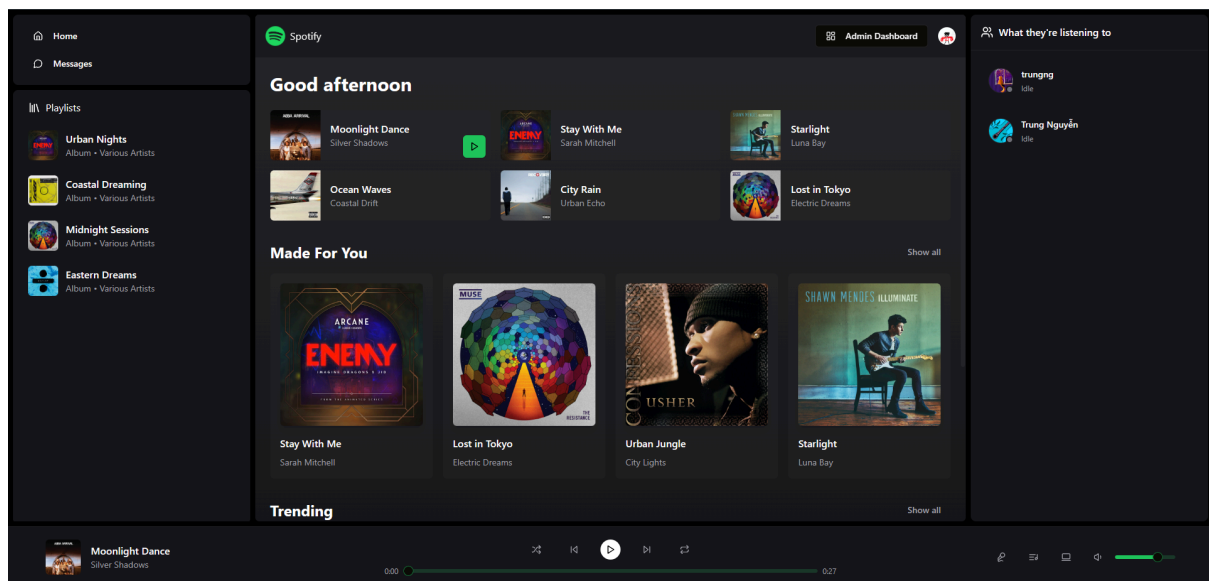
Sinh viên thực hiện:

- Nguyễn Quang Trung - B22DCDT321

MỤC LỤC

| | |
|-----------------------------|----|
| - Trang chủ..... | 3 |
| - Quản lý người dùng..... | 5 |
| - Trang album..... | 6 |
| - Trang nhắn tin..... | 7 |
| - Trang quản trị..... | 8 |
| - Tiến trình thực hiện..... | 10 |

- Trang chủ



1. Tổng quan giao diện:

Trang chủ được thiết kế theo phong cách tối giản, sử dụng giao diện nền tối (dark mode), phù hợp cho việc sử dụng trong môi trường ánh sáng yếu và giảm mỏi mắt khi dùng lâu dài.

2. Cấu trúc giao diện:

Giao diện chia thành ba khu vực chính: thanh điều hướng bên trái, khu vực nội dung trung tâm, và bảng hoạt động bên phải.

3. Thanh điều hướng bên trái (Sidebar):

- Chứa các mục điều hướng chính như "Home" và "Messages".
- Phía dưới là danh sách các playlist cá nhân.
- Mỗi playlist bao gồm hình ảnh đại diện, tiêu đề và ghi chú nguồn phát hành (Album - Various Artists).
- Cấu trúc gọn gàng, thuận tiện cho việc truy cập nhanh.

4. Khu vực nội dung trung tâm (Main Content):

- Hiển thị các gợi ý nghe nhạc tùy theo thời điểm trong ngày với lời chào như "Good afternoon".
- Bao gồm nhiều thẻ album và bài hát được đề xuất.
- Mỗi đề xuất gồm ảnh bìa, tiêu đề và tên nghệ sĩ.
- Phần "Made For You" cung cấp các gợi ý cá nhân hóa theo sở thích người dùng.

- Mục “Trending” xuất hiện ở cuối, nhưng không có nội dung cụ thể trong phiên bản hiện tại.

5. Bảng hoạt động bên phải (Right Panel):

- Hiển thị hoạt động nghe nhạc của bạn bè hoặc người dùng khác.
- Mỗi mục bao gồm tên người dùng, ảnh đại diện, và thông tin bài hát đang nghe.
- Tăng tính tương tác xã hội trong hệ thống.

6. Thanh trên cùng (Header):

- Hiển thị tên thương hiệu hoặc ứng dụng, đóng vai trò như logo định danh.
- Góc phải có nút truy cập vào bảng điều khiển dành cho quản trị viên (Admin Dashboard).
- Tách biệt rõ ràng với các khu vực khác nhằm hỗ trợ truy cập nhanh các chức năng quan trọng.

7. Thiết kế thị giác:

- Toàn bộ nền là màu tối, văn bản sử dụng tông màu sáng để tạo độ tương phản cao.
- Các thẻ nội dung có bo góc và bố trí đồng đều tạo cảm giác gọn gàng, khoa học.
- Kích thước ảnh và chữ được điều chỉnh hợp lý, đảm bảo khả năng đọc và quan sát tốt.

8. Tính năng mô phỏng:

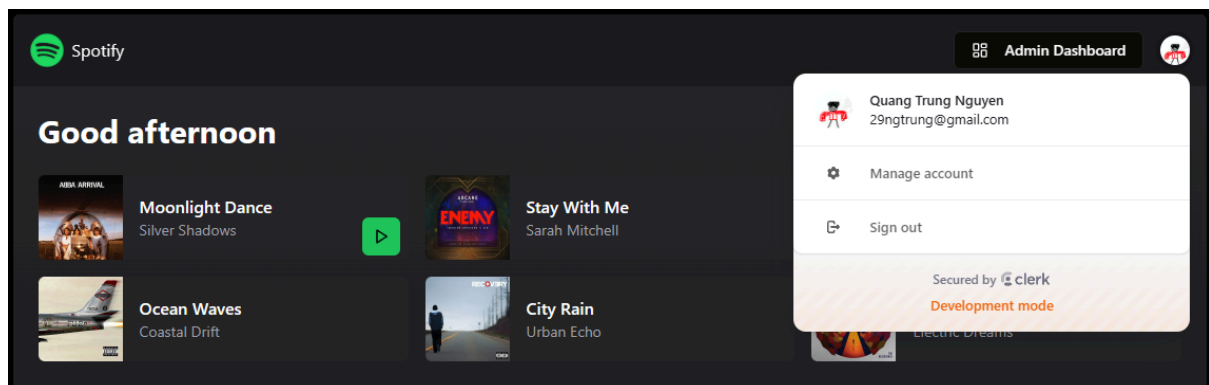
- Giao diện thể hiện các thành phần cơ bản thường có trong một nền tảng âm nhạc trực tuyến như: danh sách phát, gợi ý bài hát, hoạt động bạn bè và điều hướng cơ bản.
- Phù hợp làm mẫu giao diện (prototype) cho mục đích học tập hoặc phát triển ban đầu.

9. Tính mở rộng:

- Giao diện đủ linh hoạt để bổ sung các tính năng khác như: tìm kiếm, lọc thể loại, hiển thị danh sách phát nhạc chi tiết hoặc trình phát nhạc trực tiếp.
- Có thể tích hợp tốt với backend để tạo thành nền tảng hoàn chỉnh.

10. Ứng dụng thực tiễn:

- Giao diện thích hợp cho sinh viên ngành công nghệ thông tin hoặc lập trình viên front-end dùng làm bài tập, demo dự án hoặc sản phẩm thử nghiệm.
 - Mang tính trình diễn khả năng tổ chức giao diện, bố cục và điều hướng nội dung trên nền web.
- **Quản lý người dùng**

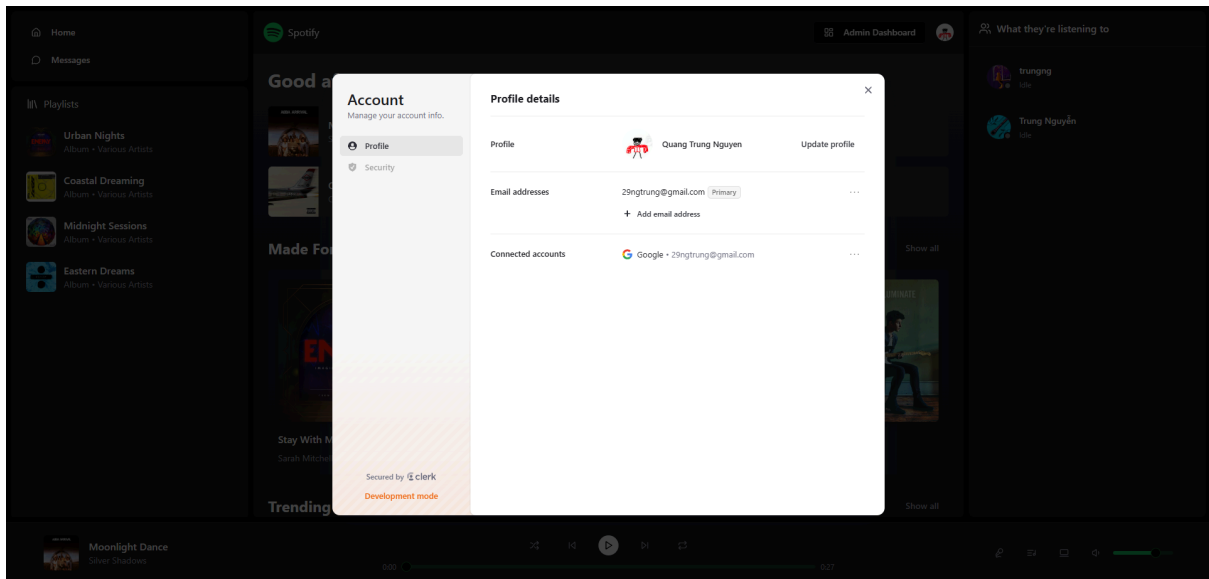


Clerk là một dịch vụ xác thực và quản lý người dùng được tích hợp vào trang web. Nó cung cấp các chức năng như đăng nhập, đăng ký, đăng xuất, và quản lý tài khoản người dùng. Clerk hỗ trợ bảo vệ các trang riêng tư bằng hệ thống phiên đăng nhập và kiểm tra quyền truy cập. Trong chế độ phát triển, Clerk hiển thị nhãn “Development mode” để phân biệt với môi trường thật.

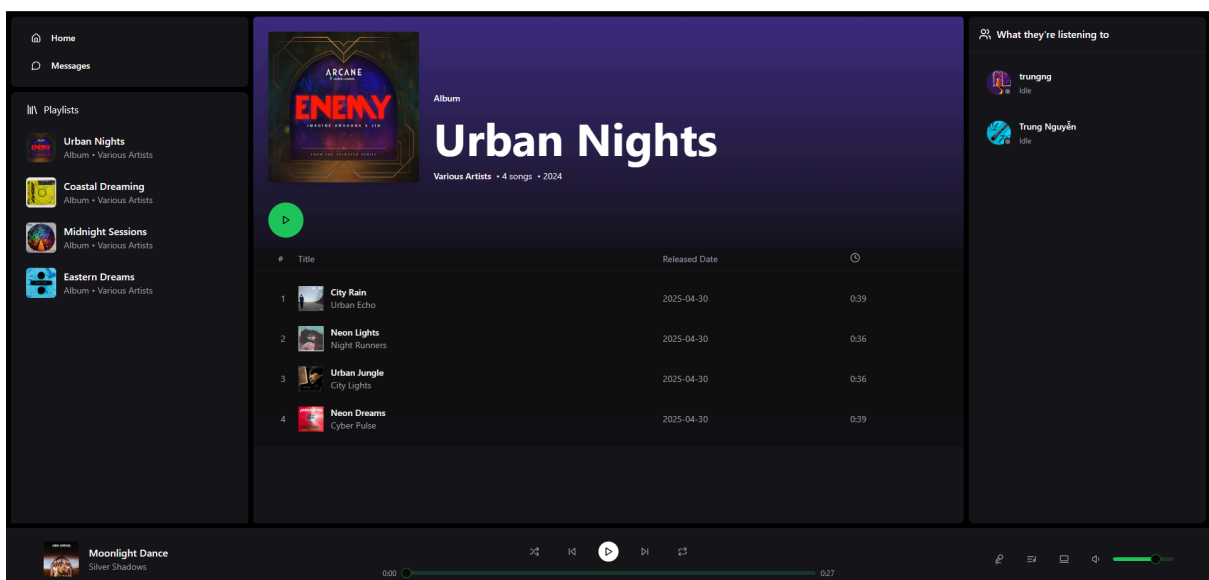
Clerk hỗ trợ xác thực thông qua nhiều phương thức như email, mật khẩu, đăng nhập bằng liên kết (magic link), OAuth (Google, GitHub, v.v.), giúp việc đăng nhập linh hoạt và phù hợp với nhiều loại người dùng. Tất cả các phần tử giao diện như form đăng nhập, đăng ký, và trang quản lý tài khoản đều được cung cấp sẵn, có thể nhúng trực tiếp vào ứng dụng mà không cần tự xây dựng thủ công.

So với việc tự triển khai hệ thống auth, Clerk tiết kiệm đáng kể thời gian phát triển và giảm rủi ro bảo mật. Ngoài ra, Clerk còn đi kèm API rõ ràng và dễ dùng để kiểm tra trạng thái người dùng, bảo vệ route riêng tư (middleware), và xử lý phân quyền trong giao diện người dùng.

Việc tích hợp Clerk trong ứng dụng giúp nhanh chóng triển khai hệ thống quản lý người dùng đầy đủ chức năng, đặc biệt hữu ích trong các dự án MVP hoặc sản phẩm nhỏ cần triển khai nhanh.



- Trang album



Trang album là nơi hiển thị thông tin chi tiết về một album nhạc cụ thể, bao gồm ảnh bìa, tiêu đề, nghệ sĩ trình bày, năm phát hành và danh sách các bài hát trong album. Người dùng có thể phát toàn bộ album bằng nút phát lớn hoặc chọn từng bài riêng lẻ để nghe.

Các thành phần chính:

- **Ảnh bìa album:** Hiển thị nổi bật ở phía trên cùng giúp người dùng dễ nhận diện album.
- **Thông tin album:** Bao gồm tên album, nghệ sĩ, số lượng bài hát và năm phát hành.

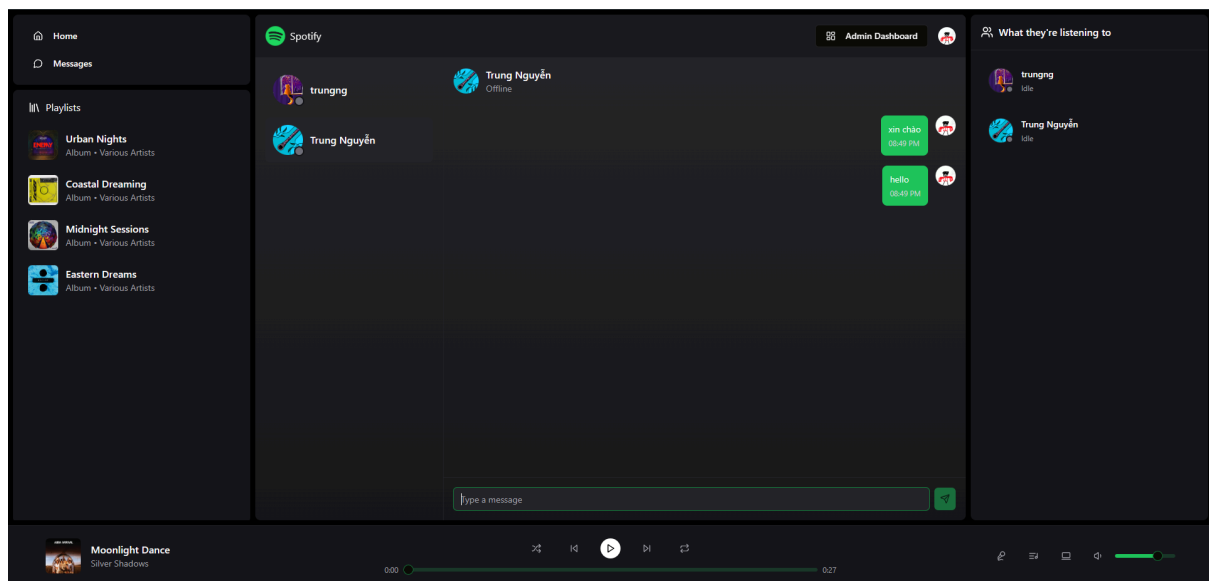
- **Danh sách bài hát:** Liệt kê tất cả các bài trong album cùng với tên nghệ sĩ, ngày phát hành và thời lượng.
- **Thanh điều khiển phát nhạc:** Nằm ở cuối trang, cho phép người dùng điều khiển nhạc như phát/tạm dừng, tua, lặp lại hoặc điều chỉnh âm lượng.

Tiện ích bổ sung:

- **Playlists sidebar:** Giúp chuyển đổi nhanh giữa các album khác.
- **Trạng thái bạn bè:** Hiển thị những người dùng khác và trạng thái nghe nhạc của họ nếu đang hoạt động.

Trang album mang lại trải nghiệm trực quan và nhất quán với các nền tảng nghe nhạc hiện đại, giúp người dùng dễ dàng khám phá và thưởng thức âm nhạc.

- Trang nhắn tin



Trang nhắn tin là tính năng cho phép người dùng giao tiếp trực tiếp với nhau thông qua hệ thống tin nhắn thời gian thực, hoạt động tương tự như hộp thoại trên Facebook Messenger. Giao diện đơn giản, gọn gàng và tích hợp liền mạch trong ứng dụng nghe nhạc.

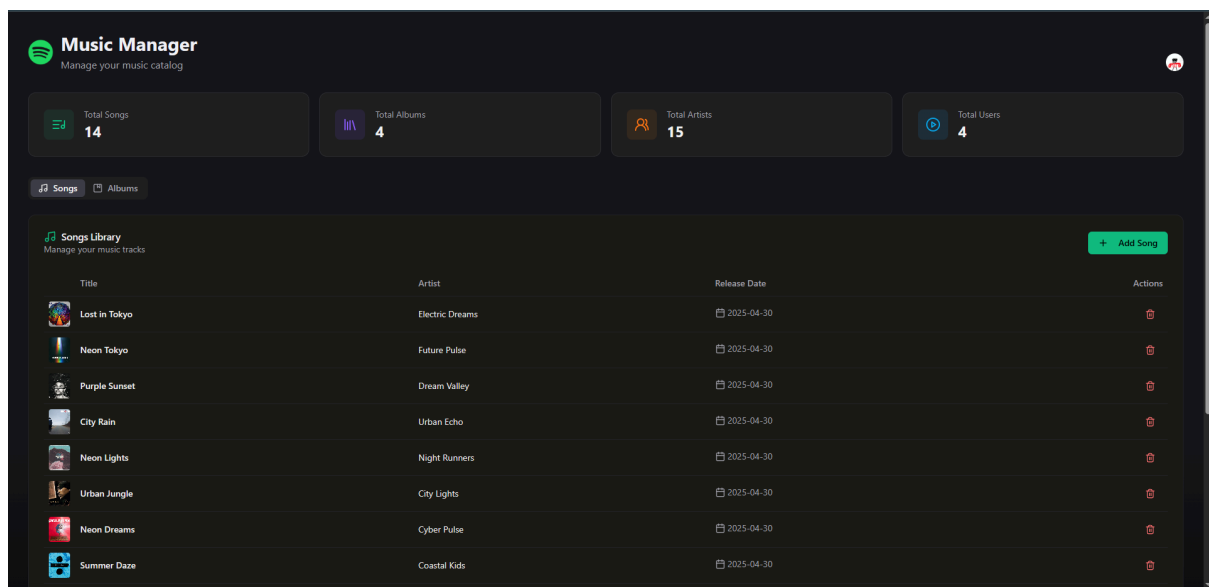
Tính năng chính:

- **Danh sách bạn bè:** Hiển thị danh sách người dùng kèm trạng thái online/offline.

- **Hộp thoại chat:** Cho phép nhắn tin một-một, hiển thị nội dung tin nhắn theo thời gian thực.
- **Socket.io:** Giao tiếp tin nhắn sử dụng WebSocket thông qua Socket.io giúp tin nhắn được cập nhật ngay lập tức mà không cần tải lại trang.
- **Giao diện nhắn tin:** Tin nhắn của người dùng hiển thị bên phải, kèm theo thời gian gửi; tin nhắn từ người khác (sắp triển khai) sẽ hiển thị bên trái.
- **Thanh nhập nội dung:** Gửi tin nhắn nhanh chóng bằng phím "Enter" hoặc nút gửi.

Tiện ích bổ sung:

- **Trạng thái hoạt động:** Ghi nhận người dùng đang trực tuyến hay không để nâng cao trải nghiệm tương tác.
 - **Tích hợp cùng giao diện nghe nhạc:** Người dùng có thể vừa trò chuyện vừa nghe nhạc mà không gián đoạn.
- Trang quản trị



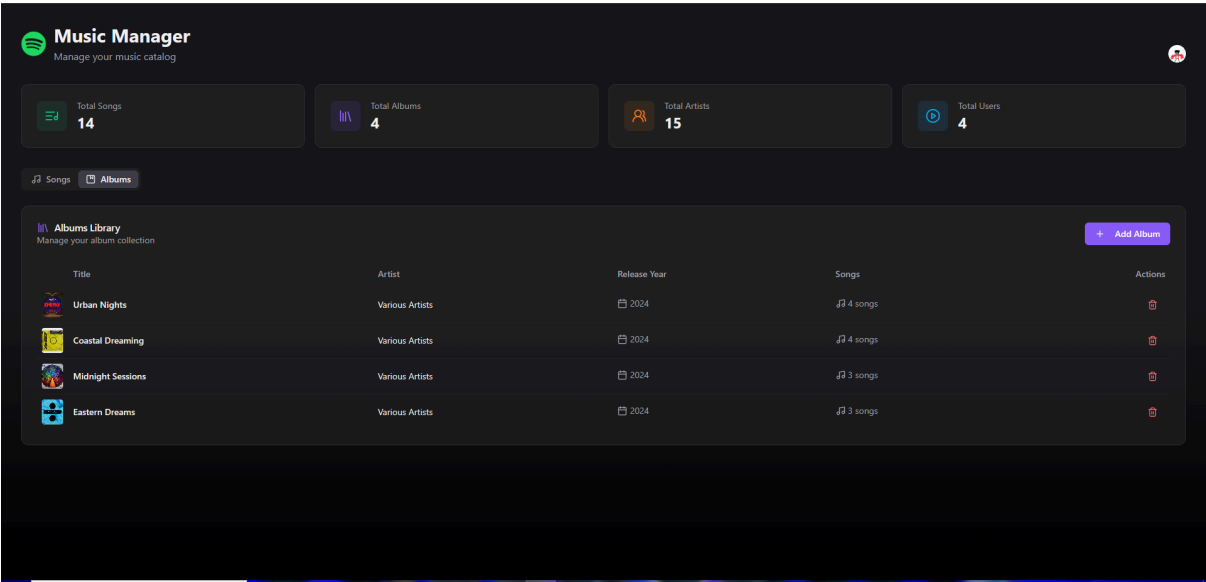
Trang quản trị là khu vực dành riêng cho quản trị viên để theo dõi, quản lý và cập nhật toàn bộ nội dung âm nhạc trong hệ thống. Giao diện hiện đại, trực quan giúp thao tác dễ dàng và hiệu quả.

Quyền truy cập:

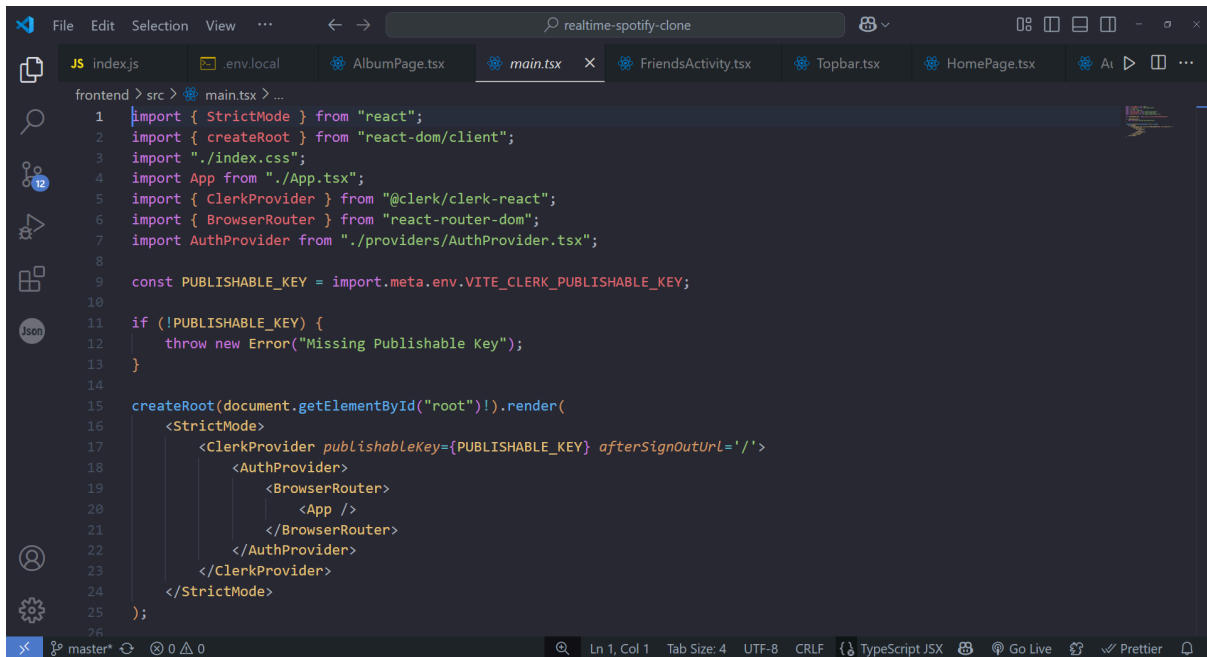
- Chỉ người dùng có quyền **admin** mới có thể truy cập trang này.
- Các thao tác quan trọng như xóa, thêm bài hát hoặc album chỉ hiển thị cho admin.

Tính năng chính:

- **Thống kê tổng quan**
 - **Tổng số bài hát** (Total Songs)
 - **Tổng số album** (Total Albums)
 - **Tổng số nghệ sĩ** (Total Artists)
 - **Tổng số người dùng** (Total Users)
- **Quản lý thư viện bài hát**
 - Hiển thị danh sách bài hát: tên bài, nghệ sĩ, ngày phát hành.
 - Hình ảnh thumbnail của từng bài hát được hiển thị trực quan.
 - **Xóa bài hát**: Admin có thể xóa bài trực tiếp bằng nút thùng rác.
- **Chuyển đổi giữa bài hát và album**
 - Người dùng có thể chọn giữa chế độ xem **Songs** hoặc **Albums** bằng nút toggle.
- **Thêm bài hát**
 - Nút Add Song mở giao diện thêm mới với các thông tin như: tên bài hát, nghệ sĩ, album, ngày phát hành, ảnh đại diện và file nhạc.

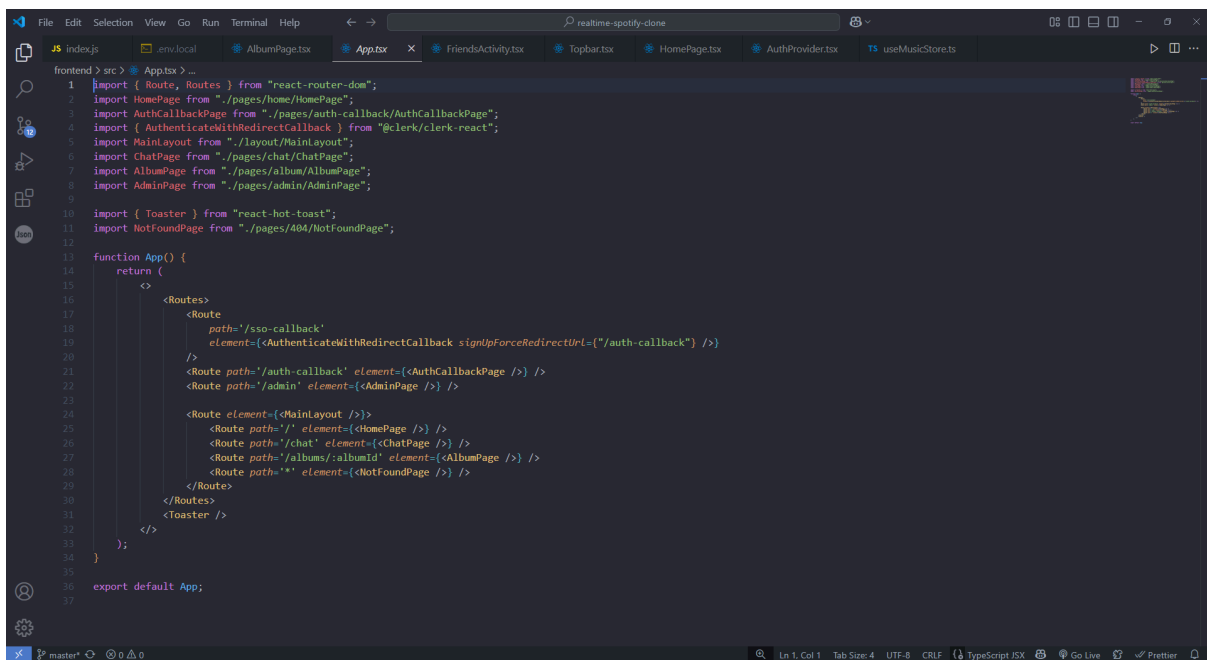


- Tiến trình thực hiện



```
1 import { StrictMode } from "react";
2 import { createRoot } from "react-dom/client";
3 import "./index.css";
4 import App from "./App.tsx";
5 import { ClerkProvider } from "@clerk/clerk-react";
6 import { BrowserRouter } from "react-router-dom";
7 import AuthProvider from "./providers/AuthProvider.tsx";
8
9 const PUBLISHABLE_KEY = import.meta.env.VITE_CLERK_PUBLISHABLE_KEY;
10
11 if (!PUBLISHABLE_KEY) {
12   throw new Error("Missing Publishable Key");
13 }
14
15 createRoot(document.getElementById("root")!).render(
16   <StrictMode>
17     <ClerkProvider publishableKey={PUBLISHABLE_KEY} afterSignOutUrl="/">
18       <AuthProvider>
19         <BrowserRouter>
20           <App />
21         </BrowserRouter>
22       </AuthProvider>
23     </ClerkProvider>
24   </StrictMode>
25 );
```

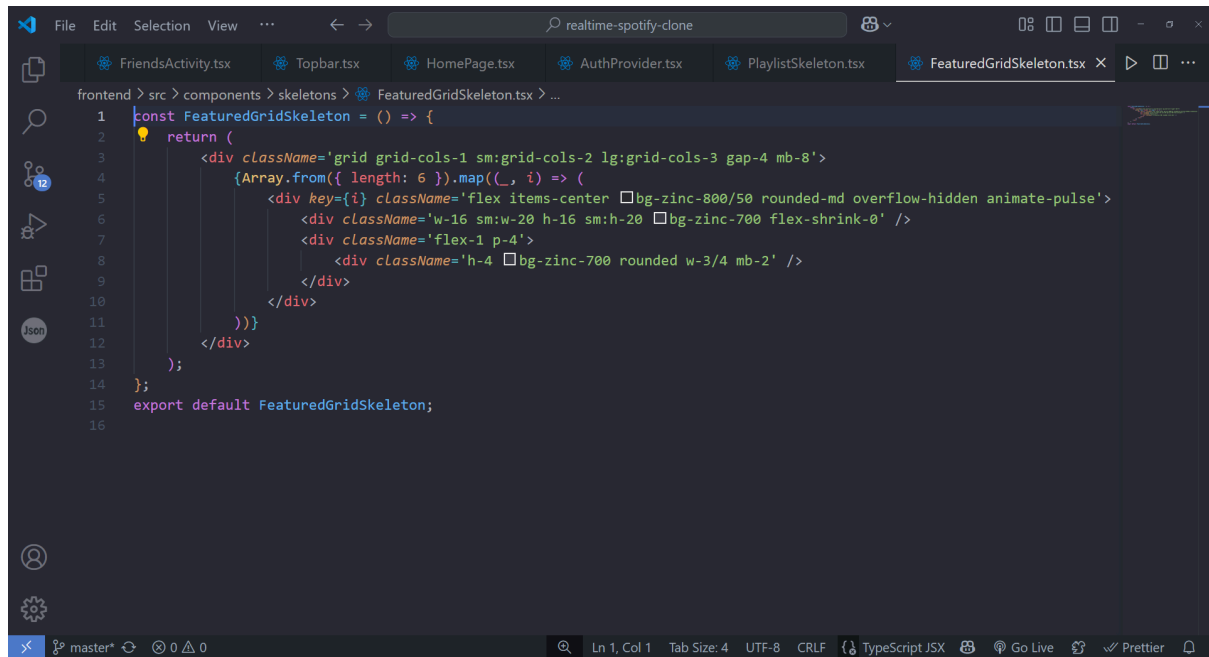
Component App sẽ được bọc bởi 2 lớp bảo mật từ Clerk và từ AuthProvider.



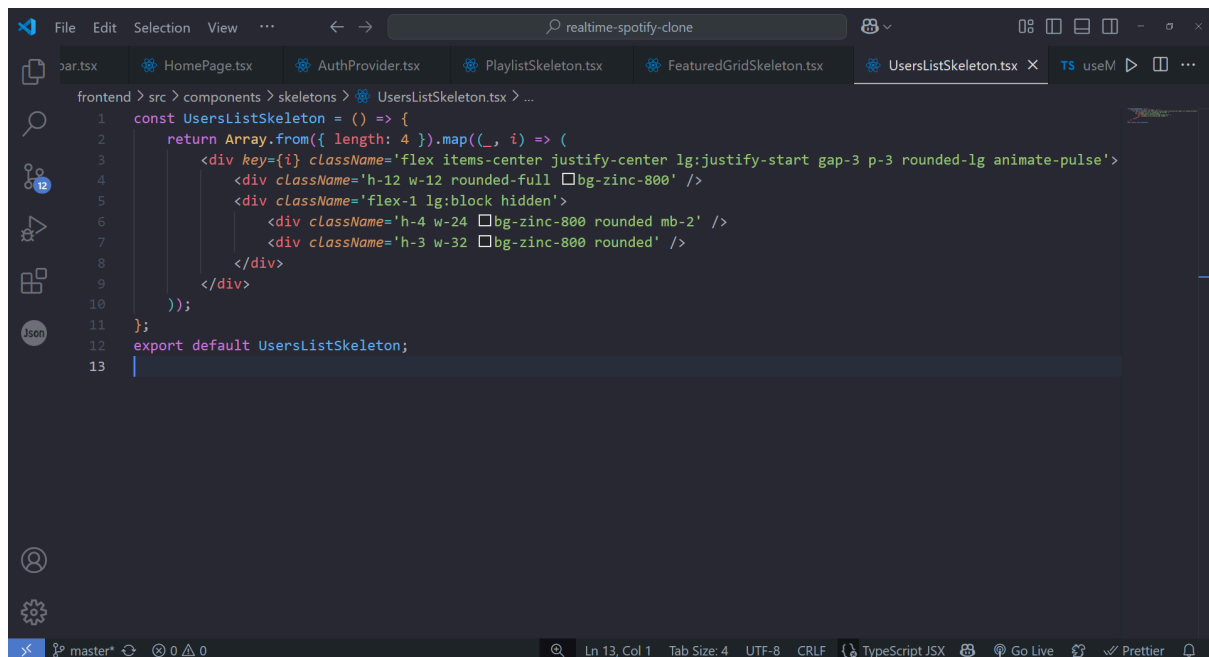
```
1 import { Route, Routes } from "react-router-dom";
2 import HomePage from "./pages/home/HomePage";
3 import AuthCallbackPage from "./pages/auth-callback/AuthCallbackPage";
4 import { AuthenticateWithRedirectCallback } from "@clerk/clerk-react";
5 import MainLayout from "./layout/MainLayout";
6 import ChatPage from "./pages/chat/ChatPage";
7 import AlbumPage from "./pages/album/AlbumPage";
8 import AdminPage from "./pages/admin/AdminPage";
9
10 import { Toast } from "react-hot-toast";
11 import NotFoundPage from "./pages/404/NotFoundPage";
12
13 function App() {
14   return (
15     <>
16       <Routes>
17         <Route
18           path="/sso-callback"
19           element={
20             <AuthenticateWithRedirectCallback
21               signUpForceRedirectUrl="/auth-callback" />
22           } />
23         <Route path="/auth-callback" element={
24           <AuthCallbackPage />
25         } />
26         <Route path="/admin" element={
27           <AdminPage />
28         } />
29         <Route element={
30           <MainLayout />
31         }>
32           <Route path="/" element={
33             <HomePage />
34           } />
35           <Route path="/chat" element={
36             <ChatPage />
37           } />
38           <Route path="/albums/:albumId" element={
39             <AlbumPage />
40           } />
41           <Route path="*" element={
42             <NotFoundPage />
43           } />
44         </Route>
45       </Routes>
46       <Toast />
47     </>
48   );
49 }
50
51 export default App;
```

Component App bao gồm các route.

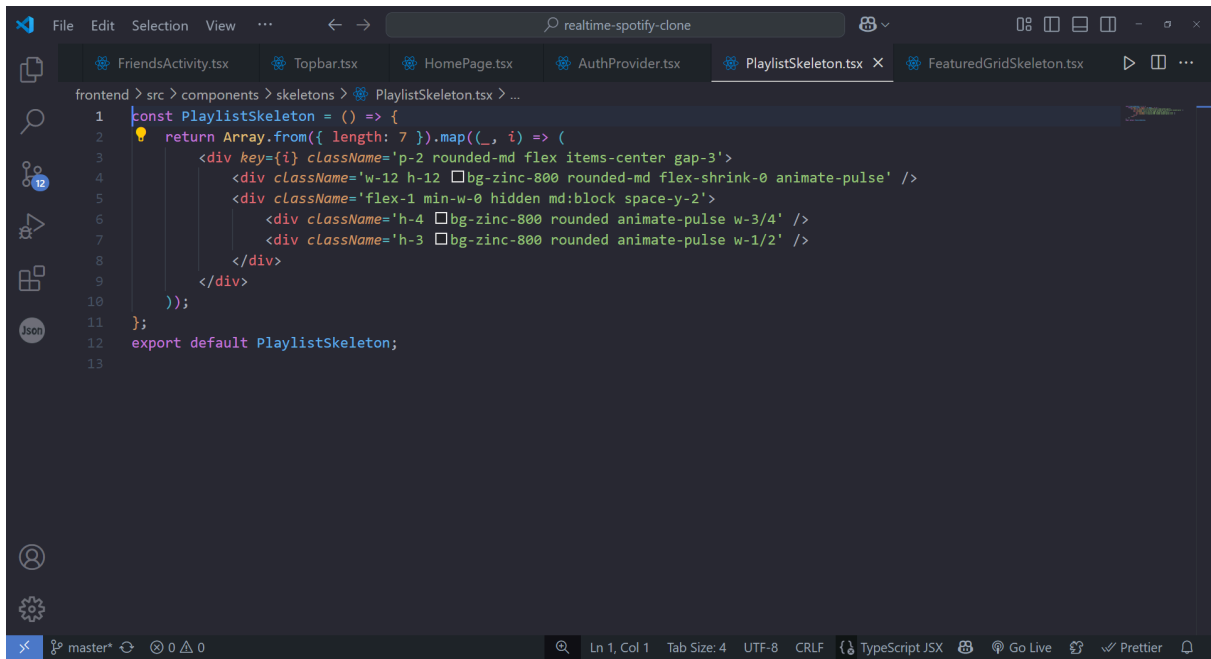
Các component skeleton để hiển thị placeholder khi đang load dữ liệu



```
frontend > src > components > skeletons > FeaturedGridSkeleton.tsx > ...
1  const FeaturedGridSkeleton = () => {
2    return (
3      <div className='grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-4 mb-8'>
4        {Array.from({ length: 6 }).map((_, i) => (
5          <div key={i} className='flex items-center justify-center bg-zinc-800/50 rounded-md overflow-hidden animate-pulse'>
6            <div className='w-16 sm:w-20 h-16 sm:h-20 bg-zinc-700 flex-shrink-0' />
7            <div className='flex-1 p-4'>
8              <div className='h-4 bg-zinc-700 rounded w-3/4 mb-2' />
9            </div>
10          </div>
11        ))}
12      </div>
13    );
14  };
15  export default FeaturedGridSkeleton;
16
```



```
frontend > src > components > skeletons > UsersListSkeleton.tsx > ...
1  const UsersListSkeleton = () => {
2    return Array.from({ length: 4 }).map((_, i) => (
3      <div key={i} className='flex items-center justify-center lg:justify-start gap-3 p-3 rounded-lg animate-pulse'>
4        <div className='h-12 w-12 rounded-full bg-zinc-800' />
5        <div className='flex-1 lg:block hidden'>
6          <div className='h-4 w-24 bg-zinc-800 rounded mb-2' />
7          <div className='h-3 w-32 bg-zinc-800 rounded' />
8        </div>
9      </div>
10    ));
11  };
12  export default UsersListSkeleton;
13
```



```
1 const PlaylistSkeleton = () => {
2   return Array.from({ length: 7 }).map((_, i) => (
3     <div key={i} className='p-2 rounded-md flex items-center gap-3'>
4       <div className='w-12 h-12 bg-zinc-800 rounded-md flex-shrink-0 animate-pulse' />
5       <div className='flex-1 min-w-0 hidden md:block space-y-2'>
6         <div className='h-4 bg-zinc-800 rounded animate-pulse w-3/4' />
7         <div className='h-3 bg-zinc-800 rounded animate-pulse w-1/2' />
8       </div>
9     </div>
10   ));
11 };
12 export default PlaylistSkeleton;
13
```

Hiện thị placeholder (các ô vuông nhấp nháy) khi chờ dữ liệu loading.

