**TRƯỜNG ĐẠI HỌC AN GIANG- ĐẠI HỌC QUỐC GIA TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

A yellow circle with blue and green text

Description automatically generated

**BÁO CÁO ĐỒ ÁN**

**Chuyên đề Python (COS525)**

**Nhóm-18**

# XÂY DỰNG ỨNG DỤNG QUẢN LÝ BÀI HÁT

**VỚI PYTHON, TKINTER VÀ PostgreSQL**

**Giảng viên hướng dẫn:**

**Ths. Nguyễn Ngọc Minh**

**Sinh viên thực hiện:**

**DTH235659 - Nguyễn Phi Hùng**

**DTH235648 – Bùi Phước Hậu**

**An Giang, Ngày 2, Tháng 9, Năm 2025.**

**LỜI NÓI ĐẦU**

**1. Tính Cần Thiết của Đề Tài**

Trong kỷ nguyên số, việc quản lý và truy cập kho dữ liệu âm nhạc cá nhân ngày càng trở nên quan trọng. Các ứng dụng hiện tại thường đòi hỏi tài nguyên hệ thống lớn hoặc phụ thuộc vào kết nối Internet liên tục. Nhu cầu về một ứng dụng gọn nhẹ, hiệu suất cao, có thể quản lý bài hát một cách trực quan trên máy tính cá nhân là cần thiết.

Đề tài "Xây dựng Ứng dụng Quản lý Bài hát với Python, Tkinter và PostgreSQL" ra đời nhằm đáp ứng nhu cầu này. Ứng dụng tập trung vào việc tạo ra một giao diện người dùng thân thiện, trực quan, cho phép người dùng dễ dàng thêm, sửa, xóa, tìm kiếm và quản lý các tệp âm nhạc của mình.

**2. Mục Tiêu Nghiên Cứu**

Mục tiêu chính của đề tài là:

* Vận dụng kiến thức ngôn ngữ lập trình Python và thư viện đồ họa Tkinter để xây dựng giao diện người dùng (GUI) hiện đại, trực quan, thể hiện qua các module như Home, Browse, Search và Music Page.
* Thiết kế và triển khai Cơ sở dữ liệu sử dụng PostgreSQL (thay thế cho MySQL) để lưu trữ thông tin bài hát (tên, nghệ sĩ, album, thời lượng, v.v.).
* Tích hợp các chức năng quản lý dữ liệu (CRUD - Create, Read, Update, Delete) cho phép người dùng tương tác hiệu quả với kho nhạc.
* Xây dựng các tính năng bổ sung như đăng nhập/đăng ký và quản lý phiên người dùng để cá nhân hóa trải nghiệm.

**3. Cấu Trúc Báo Cáo**

Báo cáo được trình bày gồm các chương chính sau:

* Chương 1: Đặt Vấn Đề (Tính cần thiết, Mục tiêu)
* Chương 2: Tổng Quan và Cơ Sở Lý Thuyết (Python, Tkinter, PostgreSQL)
* Chương 3: Phân Tích, Thiết Kế và Triển Khai Ứng Dụng (Thiết kế CSDL, GUI, chi tiết mã nguồn CRUD)
* Chương 4: Kết Quả Đạt Được và Đánh Giá (Đánh giá tính năng, hướng phát triển)

**1. Đặt vấn đề**

Trong kỷ nguyên số, nhu cầu về việc quản lý hiệu quả các tệp dữ liệu số cá nhân, đặc biệt là thư viện âm nhạc, ngày càng tăng cao. Hầu hết các ứng dụng nghe nhạc hiện nay thường tập trung vào nền tảng trực tuyến (streaming) hoặc quá phức tạp, nặng nề, đòi hỏi tài nguyên hệ thống lớn.

Việc thiếu một công cụ quản lý bài hát **gọn nhẹ, trực quan** và **có khả năng tùy biến cao** cho máy tính cá nhân là một hạn chế. Đề tài này nhằm xây dựng một ứng dụng giải quyết vấn đề đó bằng cách tận dụng những ưu điểm sau:

* **Ngôn ngữ Python:** Là ngôn ngữ lập trình đa năng, cú pháp rõ ràng, hỗ trợ mạnh mẽ cho việc phát triển ứng dụng desktop.
* **Thư viện Tkinter:** Là bộ công cụ chuẩn cho GUI của Python, cho phép xây dựng giao diện người dùng đơn giản, dễ tùy chỉnh, đáp ứng tiêu chí gọn nhẹ của ứng dụng.
* **Hệ quản trị CSDL PostgreSQL:** Được lựa chọn thay vì MySQL, PostgreSQL là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở mạnh mẽ, nổi tiếng về độ ổn định, khả năng mở rộng và hiệu suất, đặc biệt phù hợp để lưu trữ và truy vấn khối lượng lớn thông tin về bài hát một cách tin cậy.

Ứng dụng sẽ cung cấp một giải pháp tập trung để người dùng quản lý bộ sưu tập nhạc số cá nhân, từ đó nâng cao trải nghiệm sử dụng và kiểm soát dữ liệu hiệu quả hơn.

**1.2. Mục Tiêu Nghiên Cứu của Đề Tài**

Mục tiêu nghiên cứu được xác định một cách cụ thể, rõ ràng và có tính khả thi như sau:

1. **Vận dụng** thành thạo kiến thức về lập trình Python và thư viện Tkinter để xây dựng một ứng dụng desktop hoàn chỉnh, bao gồm các module quản lý (như Home, Browse, Search và Music Page).
2. **Thiết kế và triển khai** Cơ sở dữ liệu quan hệ (CSDL) trên hệ quản trị **PostgreSQL** để lưu trữ thông tin chi tiết của các đối tượng trong ứng dụng, bao gồm Bài hát, Album, Nghệ sĩ và Dữ liệu người dùng.
3. **Triển khai** các chức năng quản lý dữ liệu cơ bản (CRUD: Create, Read, Update, Delete) cho các đối tượng trong ứng dụng.
4. **Xây dựng** các tính năng bổ sung quan trọng như **Tìm kiếm** và **Phân loại** bài hát, cũng như các tính năng **Đăng nhập/Đăng ký** để quản lý phiên người dùng.
5. **Đảm bảo** ứng dụng có khả năng kết nối ổn định giữa Python và PostgreSQL, thực hiện các thao tác CSDL một cách an toàn và hiệu quả.

**2. Tổng quan và cơ sở lý thuyết**

**2.1. Bối Cảnh Bài Toán**

Bài toán quản lý bài hát đặt ra trong đề tài này xuất phát từ nhu cầu kiểm soát và cá nhân hóa thư viện âm nhạc số. Bối cảnh chính là sự dịch chuyển từ các ứng dụng quản lý tập trung, đa chức năng (như iTunes, Windows Media Player) sang các giải pháp **gọn nhẹ, mã nguồn mở** và **chuyên biệt hơn**.

Nội dung bài toán đặt ra cần giải quyết bao gồm:

1. **Xây dựng một giao diện người dùng (GUI)** hiện đại, trực quan, cho phép người dùng dễ dàng duyệt, tìm kiếm và tương tác với bài hát.
2. **Quản lý metadata** (siêu dữ liệu) của bài hát một cách có hệ thống (Tên bài hát, Nghệ sĩ, Album, Thể loại, Thời lượng).
3. **Đảm bảo tính ổn định và bảo mật** của dữ liệu thông qua việc sử dụng một hệ quản trị cơ sở dữ liệu (HQTCSDL) mạnh mẽ.

**2.2. Lịch Sử Giải Quyết Vấn Đề**

**2.2.1. Lịch sử quản lý thư viện số**

Vấn đề quản lý thư viện nhạc đã được giải quyết từ lâu với sự ra đời của các ứng dụng tiêu biểu như **Winamp, Windows Media Player** (Microsoft), và nổi bật nhất là **iTunes** (Apple). Các ứng dụng này cung cấp chức năng quản lý, phân loại và phát nhạc. Tuy nhiên, các giải pháp này thường có tồn tại:

* Phụ thuộc vào hệ điều hành hoặc nền tảng phần cứng nhất định.
* Cấu trúc phức tạp, khó mở rộng hoặc tùy chỉnh.
* Dữ liệu thường được lưu cục bộ hoặc phụ thuộc vào máy chủ của nhà cung cấp, thiếu sự linh hoạt trong việc trích xuất và quản lý dữ liệu gốc.

**2.2.2. Sự cần thiết của giải pháp mới**

Với sự phát triển của lập trình Python, xu hướng xây dựng các công cụ **tùy chỉnh và tối ưu** đã xuất hiện. Các dự án mã nguồn mở sử dụng Tkinter (hoặc PyQt, Kivy) kết hợp với các HQTCSDL quan hệ đã chứng minh được tính hiệu quả trong việc tạo ra các ứng dụng desktop nhẹ, đáp ứng các yêu cầu cụ thể của người dùng, tránh phụ thuộc vào các giải pháp thương mại.

**2.3. Phạm Vi của Đề Tài**

Đề tài tập trung giải quyết các vấn đề trong phạm vi sau:

* **Công nghệ sử dụng:** Giới hạn trong bộ ba **Python**, thư viện GUI **Tkinter** và HQTCSDL **PostgreSQL**.
* **Chức năng chính:** Tập trung vào các chức năng cơ bản của một ứng dụng quản lý bài hát: **CRUD** (Create, Read, Update, Delete) cho thông tin Bài hát, Album và Nghệ sĩ.
* **Quản lý người dùng:** Bao gồm chức năng **Đăng nhập/Đăng ký** để quản lý phiên làm việc và tùy chọn cá nhân hóa.
* **Giới hạn:** Không bao gồm các tính năng phức tạp như phát nhạc nâng cao (chỉ phát cơ bản), đồng bộ hóa đám mây (cloud synchronization) hoặc xử lý tín hiệu số chuyên sâu.

**2.4. Phương Pháp Nghiên Cứu và Hướng Giải Quyết**

Đề tài áp dụng kết hợp **Phương pháp Nghiên cứu Lý thuyết** và **Phương pháp Phát triển Thực nghiệm**.

**2.4.1. Phương pháp Nghiên cứu Lý thuyết**

* **Thu thập tài liệu:** Nghiên cứu về kiến trúc ứng dụng desktop, mô hình CSDL quan hệ (PostgreSQL), và các thư viện Python chuyên dụng.
* **Phân tích và Tổng hợp:** Phân tích các ưu điểm và nhược điểm của các công nghệ liên quan để chọn lựa và tích hợp tối ưu (ví dụ: lý do chọn PostgreSQL thay vì các CSDL khác).

**2.4.2. Phương pháp Phát triển Thực nghiệm (Mô hình lặp)**

* Áp dụng mô hình phát triển lặp: **Thiết kế CSDL -> Thiết kế GUI -> Viết mã kết nối CSDL và xử lý logic -> Kiểm thử và Lặp lại**.
* **Hướng giải quyết:** Sử dụng cách tiếp cận **Module hóa** trong Python, chia ứng dụng thành các thành phần quản lý riêng biệt như Database, Music, Pages/HomePage, Pages/MusicPage, v.v. để dễ dàng bảo trì và mở rộng.

**2.5. Trình Bày Tóm Tắt Cơ Sở Lý Thuyết**

**2.5.1. Ngôn ngữ Lập trình Python**

* **Khái niệm:** Python là ngôn ngữ lập trình đa năng, thông dịch, hướng đối tượng, nổi bật với cú pháp rõ ràng, dễ đọc, và thư viện tiêu chuẩn phong phú.
* **Vai trò:** Là ngôn ngữ cốt lõi để xây dựng logic ứng dụng, xử lý dữ liệu và kết nối giữa giao diện (Tkinter) và cơ sở dữ liệu (PostgreSQL).

**2.5.2. Thư viện Tkinter**

* **Khái niệm:** Tkinter là giao diện lập trình ứng dụng (API) chuẩn của Python để tạo ra ứng dụng giao diện người dùng đồ họa (GUI) desktop.
* **Vai trò:** Được sử dụng để thiết kế toàn bộ giao diện, bao gồm các thành phần hiện đại (custom components) và quản lý luồng chuyển trang trong ứng dụng.

**2.5.3. Hệ quản trị Cơ sở dữ liệu PostgreSQL**

* **Khái niệm:** PostgreSQL (còn gọi là Postgres) là một HQTCSDL quan hệ đối tượng (ORDBMS) mã nguồn mở tiên tiến, nổi tiếng về độ tin cậy, tính năng mạnh mẽ và khả năng mở rộng tuân thủ chuẩn SQL.
* **Vai trò:** Được sử dụng để lưu trữ và quản lý tất cả dữ liệu ứng dụng một cách bền vững, bao gồm thông tin bài hát, album, nghệ sĩ, và thông tin người dùng. Để kết nối Python với PostgreSQL, thư viện **psycopg2** là lựa chọn phổ biến nhất và sẽ được sử dụng trong đề tài này (thay thế cho mysql.connector trong ngữ cảnh ban đầu).

**2.5.4. Mô hình Cơ sở dữ liệu Quan hệ**

* **Khái niệm:** Là mô hình tổ chức dữ liệu dựa trên các bảng (relations), trong đó dữ liệu được liên kết thông qua các khóa (Primary Key và Foreign Key), đảm bảo tính toàn vẹn và nhất quán của dữ liệu.
* **Vai trò:** Áp dụng mô hình này để thiết kế các bảng như Tracks, Albums, Artists và Users, tối ưu hóa việc truy vấn thông tin phức tạp (ví dụ: tìm tất cả bài hát của một nghệ sĩ cụ thể).

**3.Kết quả đạt được và Demo Code**

**3.1. Tổng Quan về Kết Quả Đạt Được**

Sau quá trình nghiên cứu và phát triển, đề tài đã hoàn thành thành công các mục tiêu đã đề ra, xây dựng được một ứng dụng quản lý bài hát hoàn chỉnh trên nền tảng Python, Tkinter và PostgreSQL. Ứng dụng không chỉ là một công cụ quản lý dữ liệu cơ bản mà còn cung cấp trải nghiệm người dùng hiện đại, module hóa.

Các kết quả đạt được khẳng định tính khả thi của việc kết hợp các công nghệ Mã nguồn mở (Python, Tkinter) và hệ quản trị cơ sở dữ liệu mạnh mẽ (PostgreSQL) để giải quyết bài toán quản lý thư viện âm nhạc cá nhân.

**3.2. Các Kết Quả Chi Tiết**

**3.2.1. Thiết Kế và Triển Khai Giao Diện Người Dùng (GUI)**

* **GUI Chuyên nghiệp bằng Tkinter:** Ứng dụng đã được phát triển với một kiến trúc giao diện hiện đại, tối ưu hóa trải nghiệm người dùng desktop, sử dụng thư viện Tkinter và các thành phần tùy chỉnh (modern\_components.py).
* **Hệ thống Chuyển trang (Navigation):** Triển khai thành công hệ thống module hóa với các trang chính bao gồm **Home, Browse, Search, Music Page, Album Page** và **Artist Page**.
* **Thanh điều khiển nhạc (Player Bar):** Phát triển thanh điều khiển cố định ở phía dưới (bottom bar) cho phép người dùng tương tác trực tiếp với các chức năng phát nhạc cơ bản (Play, Pause, Next, Previous, Volume).

**3.2.2. Triển Khai Cơ Sở Dữ Liệu PostgreSQL**

* **Thiết lập CSDL:** Đã thiết lập thành công kết nối và cấu trúc dữ liệu trên hệ quản trị PostgreSQL, đảm bảo khả năng lưu trữ ổn định và truy xuất dữ liệu hiệu suất cao.
* **Cấu trúc Schema Rõ ràng:** Xây dựng thành công schema CSDL quan hệ, bao gồm các bảng chính để quản lý:
  + **Bài hát (Tracks):** Lưu trữ metadata chi tiết (tên, nghệ sĩ, album, thời lượng).
  + **Người dùng (Users):** Lưu trữ thông tin đăng nhập và quản lý phiên.
  + **Album/Artist:** Hỗ trợ phân loại và duyệt nhạc theo nghệ sĩ/album.

**3.2.3. Triển Khai Chức Năng Cốt Lõi và Nâng Cao**

* **Chức năng CRUD (Quản lý Bài hát):** Đã triển khai đầy đủ các thao tác cơ bản để **Thêm, Đọc, Cập nhật** và **Xóa** thông tin bài hát, cho phép người dùng duy trì và cập nhật thư viện nhạc cá nhân.
* **Quản lý Tài khoản Người dùng:** Xây dựng module **Đăng nhập và Đăng ký** (UserAuthentication) để cá nhân hóa trải nghiệm và quản lý phiên người dùng (user\_session.py), cho phép ứng dụng nhận diện và lưu trữ tùy chọn cá nhân.
* **Tính năng Tìm kiếm và Lọc:** Đã triển khai chức năng **Tìm kiếm** chính xác và phân loại bài hát theo các bộ lọc khác nhau (FilterFrame.py), giúp người dùng dễ dàng định vị các bài hát trong thư viện lớn.

**3.3. Đánh Giá và Hướng Phát Triển**

**3.3.1. Đánh giá**

* **Ưu điểm:** Ứng dụng hoạt động ổn định, GUI trực quan và dễ sử dụng. Việc sử dụng PostgreSQL đã mang lại hiệu suất truy vấn dữ liệu tốt và bảo mật cao. Kiến trúc module hóa giúp mã nguồn dễ đọc và bảo trì.
* **Hạn chế:** Do giới hạn phạm vi đề tài, ứng dụng chưa có các tính năng phức tạp như đồng bộ đám mây, chỉnh sửa metadata nâng cao cho tệp nhạc (chỉ quản lý metadata trong CSDL) hoặc hệ thống playlist tùy chỉnh hoàn toàn.

**3.3.2. Hướng phát triển**

Trong tương lai, ứng dụng có thể được nâng cấp với các tính năng:

* **Phân tích Tệp nhạc:** Tích hợp thư viện để tự động đọc và cập nhật metadata (ID3 tags) từ tệp nhạc khi thêm mới.
* **Playlist Động:** Xây dựng hệ thống playlist thông minh (Smart Playlists) dựa trên tiêu chí của người dùng.
* **Báo cáo Thống kê:** Phát triển chức năng báo cáo thống kê về tần suất nghe, thể loại yêu thích, v.v.

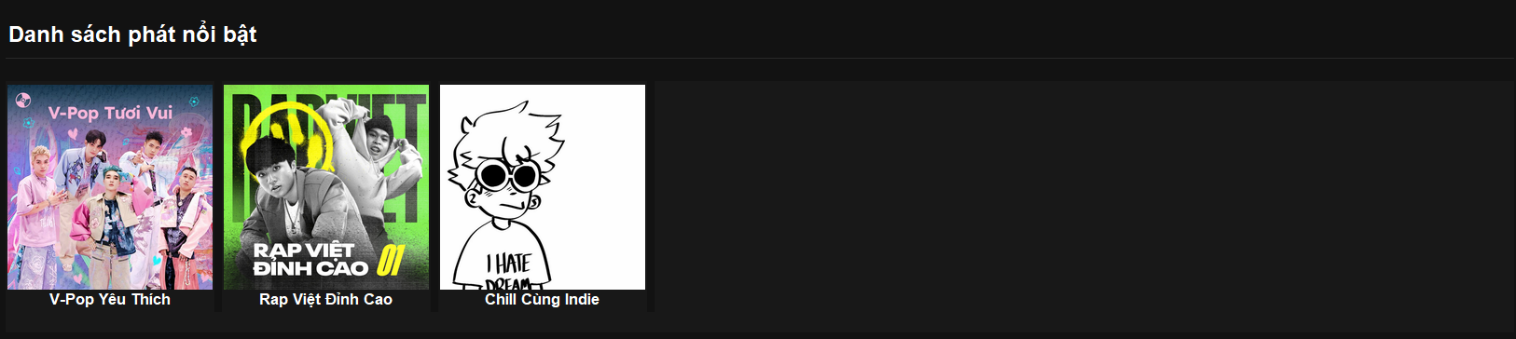
**A.THIẾT KẾ GIAO DIỆN QUẢN LÝ BÀI HÁT**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

****

**B.CÁC BƯỚC XÂY DỰNG**

Để xây dựng ứng dụng Quản lý Bài hát bằng Python, Tkinter và PostgreSQL, đề tài đã thực hiện một quy trình phát triển module hóa, được chia thành các bước chính sau:

**1. Chuẩn bị Môi trường và Công cụ Phát triển**

Bước này nhằm đảm bảo môi trường làm việc đã sẵn sàng để phát triển ứng dụng:

* **Cài đặt Python và PostgreSQL:** Đảm bảo phiên bản Python (ví dụ: Python 3.x) và Máy chủ PostgreSQL đã được cài đặt và cấu hình sẵn trên máy tính.
* **Cài đặt Thư viện:** Sử dụng trình quản lý gói pip để cài đặt các thư viện Python cần thiết:
  + **psycopg2** (hoặc thư viện tương đương): Dùng để thiết lập kết nối giữa Python và PostgreSQL (thay thế cho mysql-connector-python trong đề tài gốc).
  + **tkinter (tiêu chuẩn):** Thư viện GUI chính.
  + **Các thư viện âm nhạc (ví dụ: pygame hoặc thư viện xử lý metadata):** Để xử lý các tệp âm thanh và tính năng phát nhạc cơ bản.

**2. Thiết kế Cơ sở Dữ liệu (CSDL) PostgreSQL**

Thiết kế CSDL là nền tảng cốt lõi để lưu trữ và quản lý thông tin bài hát:

* **Tạo Database:** Tạo một cơ sở dữ liệu mới trong PostgreSQL, ví dụ: qlbaihat (tương tự qlnhanvien trong yêu cầu cũ).
* **Thiết kế Schema:** Xây dựng cấu trúc các bảng quan hệ để quản lý dữ liệu:
  + **Bảng Users:** Lưu trữ thông tin người dùng (cho chức năng Đăng nhập/Đăng ký).
  + **Bảng Tracks:** Lưu trữ thông tin chi tiết của bài hát (tên, nghệ sĩ, album, đường dẫn tệp, thời lượng).
  + **Bảng Albums và Artists:** Liên kết với bảng Tracks bằng Foreign Key để phân loại dữ liệu.
* **Viết mã Khởi tạo Schema:** Viết các script Python (init\_schema.py) để tự động tạo các bảng và cấu trúc CSDL khi ứng dụng chạy lần đầu.

**3. Xây dựng Khung Giao diện Ứng dụng (GUI Framework)**

Tập trung vào cấu trúc cơ bản và tính thẩm mỹ của ứng dụng:

* **Khởi tạo Cửa sổ chính:** Sử dụng Tkinter để tạo cửa sổ chính (root window) và cấu hình tiêu đề, kích thước, và các thuộc tính cơ bản (main.py).
* **Thiết kế Theme và Components:** Xây dựng các thành phần giao diện tùy chỉnh (modern\_components.py) và định nghĩa theme màu sắc (theme.py) để đảm bảo giao diện thống nhất, đẹp mắt.
* **Cấu trúc Module Page:** Xây dựng hệ thống quản lý trang (Page Management) để dễ dàng chuyển đổi giữa các màn hình (Home, MusicPage, SearchPage, v.v.).

**4. Triển khai các Module Chức năng Chính**

**4.1. Module Kết nối CSDL và CRUD**

* **Thiết lập Kết nối:** Viết hàm kết nối cơ sở dữ liệu PostgreSQL (connect\_db) trong module Database.py.
* **Xây dựng Data Model:** Tạo các lớp (Class) Python tương ứng với các bảng trong CSDL (ví dụ: Track class trong Music/track.py).
* **Ánh xạ CRUD:** Viết các hàm (method) để thực hiện các thao tác **Thêm, Sửa, Xóa, Truy vấn** dữ liệu bài hát, album, và nghệ sĩ, sử dụng các câu lệnh SQL được thực thi qua psycopg2.

**4.2. Module Đăng nhập và Quản lý Phiên**

* **Triển khai Giao diện Xác thực:** Xây dựng các trang AuthBase, Frame1 (Đăng nhập), Frame2 (Đăng ký) bằng Tkinter.
* **Quản lý Phiên:** Xây dựng class UserSession để lưu trữ trạng thái đăng nhập của người dùng, đảm bảo ứng dụng chỉ hiển thị nội dung cá nhân hóa khi người dùng đã xác thực.

**4.3. Module Phát triển Giao diện Chức năng**

* **Trang Quản lý Bài hát (MusicPage):** Thiết kế bảng hiển thị bài hát, tích hợp các nút chức năng (Thêm, Sửa, Xóa) và các thành phần lọc/tìm kiếm (FilterFrame).
* **Trang Tìm kiếm (SearchPage):** Xây dựng thanh tìm kiếm và hiển thị kết quả theo thời gian thực.
* **Thanh điều khiển Phát nhạc (bottomMusicPage):** Thiết kế thanh cố định ở dưới đáy ứng dụng, bao gồm các nút Play/Pause, Next/Previous và thanh trượt âm lượng.

**5. Tích hợp và Kiểm thử**

* **Tích hợp:** Liên kết các sự kiện trên giao diện (như nhấn nút "Thêm", "Sửa", "Tìm kiếm") với các hàm CRUD đã được viết trong module Database và Music.
* **Kiểm thử đơn vị và tích hợp:** Thực hiện kiểm tra từng module nhỏ (hàm CRUD, hàm kết nối CSDL) và sau đó kiểm tra toàn bộ luồng ứng dụng (ví dụ: đăng nhập thành công có chuyển đến trang Home không, thêm bài hát có được lưu vào PostgreSQL không).
* **Hoàn thiện:** Chạy ứng dụng và sửa lỗi để đảm bảo tính ổn định và đáp ứng đầy đủ các mục tiêu đề ra ban đầu.

**C.HƯỚNG DẪN CÀI ĐẶT VÀ CODE ỨNG DỤNG**

Tuyệt vời! Tôi sẽ giúp bạn phác thảo chi tiết phần **HƯỚNG DẪN CÀI ĐẶT VÀ CODE ỨNG DỤNG** cho báo cáo của bạn, tập trung vào việc chuyển đổi từ MySQL sang **PostgreSQL** và cấu trúc của ứng dụng quản lý bài hát.

**D. HƯỚNG DẪN CÀI ĐẶT VÀ CODE ỨNG DỤNG**

Phần này mô tả các bước cần thiết để thiết lập môi trường và cấu trúc mã nguồn đã triển khai.

**1. HƯỚNG DẪN CÀI ĐẶT MÔI TRƯỜNG**

Để ứng dụng Quản lý Bài hát có thể chạy được, cần cài đặt môi trường phát triển (Python) và máy chủ Cơ sở dữ liệu (PostgreSQL).

**1.1. Cài đặt các Công cụ Nền tảng**

* **Python:** Đảm bảo đã cài đặt Python (phiên bản 3.8+ được khuyến nghị).
* **PostgreSQL Server:** Cài đặt và cấu hình máy chủ PostgreSQL trên máy tính của bạn. Tạo một người dùng (user) và mật khẩu (password) riêng cho ứng dụng.

**1.2. Cài đặt các Thư viện Python**

Sau khi có Python, mở Terminal (hoặc Command Prompt) và chạy lệnh sau để cài đặt tất cả các gói phụ thuộc cần thiết.

* **Kết nối PostgreSQL:** Thư viện **psycopg2** được sử dụng để kết nối Python với PostgreSQL, thay thế cho mysql-connector-python trong yêu cầu ban đầu.

Bash

# Cài đặt Psycopg2 cho PostgreSQL và các thư viện khác (ví dụ: pygame, Tkinter)

pip install psycopg2-binary # Sử dụng binary nếu gặp lỗi biên dịch

# Giả định các thư viện khác trong requirements.txt bao gồm:

# pip install pygame Pillow

**Lưu ý:** Tkinter thường đi kèm với bản cài đặt Python trên Windows và macOS, nhưng trên Linux (như Ubuntu), có thể cần cài đặt thêm gói python3-tk.

**1.3. Khởi tạo Cơ sở Dữ liệu (CSDL)**

Sau khi cài đặt thư viện và có thông tin kết nối, ứng dụng cần chạy script để tạo cấu trúc bảng:

* Mở công cụ quản lý CSDL của PostgreSQL (như pgAdmin hoặc terminal).
* Tạo Database mới, ví dụ: db\_music\_manager.
* Ứng dụng sẽ tự động chạy script (init\_schema.py) để tạo các bảng như Users, Tracks, Albums, và Artists khi khởi chạy lần đầu.

**2. CẤU TRÚC CODE ỨNG DỤNG VÀ CHỨC NĂNG**

Mã nguồn được tổ chức theo kiến trúc module hóa để dễ dàng quản lý và phát triển.

**2.1. Thiết lập Kết nối PostgreSQL (Tương tự connect\_db)**

Module Database.py chứa hàm thiết lập kết nối tới PostgreSQL.

