**TÀI LIỆU THIẾT KẾ BAN ĐẦU**

**(Initial Design Document - IDD)**

**Xây dựng website nghe nhạc trực tuyến**

**Môn học: Công nghệ Phần mềm**

**Lớp học phần: INT2208E 23**

**Giảng viên: PGS.TS Đặng Đức Hạnh**

**Nhóm sinh viên thực hiện:**

|  |  |
| --- | --- |
| **Cao Đức Chung** | **22028015** |
| **Đàm Quang Đạt** | **22028026** |
| **Nhâm Ngọc Đức** | **22028096** |
| **Nguyễn Mạnh Quân** | **22028171** |
| **Lê Văn Thắng** | **22028313** |

**Hà Nội, ngày 05 tháng 5 năm 2024**

# 

# **Mục lục**

Contents

[**Mục lục** 1](#_Toc165823607)

[**Bảng ghi nhận thay đổi** 1](#_Toc165823608)

[**1.** **Giới thiệu** 2](#_Toc165823609)

[**1.1.** **Mục đích** 2](#_Toc165823610)

[**1.2.** **Đối tượng hướng đến và đề xuất cách đọc** 2](#_Toc165823611)

[**1.3.** **Tài liệu tham khảo** 3](#_Toc165823612)

[**1.4.** **Bố cục tài liệu** 3](#_Toc165823613)

[**2.** **Nội dung chính** 4](#_Toc165823614)

[**1.1.** **Cấu trúc hệ thống** 4](#_Toc165823615)

[**1.2.** **Các Diagram cho UC chính** 5](#_Toc165823616)

[**1.2.1.** **Sequence Diagram** 5](#_Toc165823617)

[**1.2.2.** **Class Diagram** 10](#_Toc165823618)

[**1.3.** **Lược đồ cơ sở dữ liệu** 16](#_Toc165823619)

[**1.4.** **Mô phỏng UI** 17](#_Toc165823620)

# **Bảng ghi nhận thay đổi**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên | Thời gian | Mục đích thay đổi | Phiên bản |
| Quang Đạt | 5/5/2024 | Khởi tạo mẫu tài liệu | 1.0 |
| Quang Đạt | 5/5/2024 | Thêm nội dung playlist | 1.1 |

# **Giới thiệu**

## **Mục đích**

Tài liệu này được tạo nên với mục đích cung cấp cái nhìn tổng quan về các thành phần của dự án theo một hướng chi tiết về kĩ thuật hơn.

Mục đích của Tài liệu thiết kế ban đầu là tách biệt mối quan tâm của các nhà đầu cơ của hệ thống (như được mô tả bởi mô hình Use case trong tài liệu SRS và các yêu cầu của hệ thống) khỏi các mối quan tâm của các nhà thiết kế hệ thống. Việc thực hiện Use case cung cấp một cấu trúc trong mô hình thiết kế, tổ chức các tạo tác liên quan đến Use case nhưng thuộc về mô hình thiết kế. Tài liệu này bao gồm mô hình tổng quan của hệ thống, các Class\_diagram và Sequence\_diagram của Use case theo các đối tượng cộng tác.

## **Đối tượng hướng đến và đề xuất cách đọc**

* **Vai trò người thiết kế:** Vai trò của người thiết kế xác định trách nhiệm, hoạt động, thuộc tính và mối quan hệ của một hoặc một số Class và xác định cách chúng sẽ được điều chỉnh theo môi trường thực hiện. Ngoài ra, vai trò của nhà thiết kế có thể chịu trách nhiệm đối với một hoặc nhiều gói thiết kế hoặc các Sub system thiết kế, bao gồm mọi Class thuộc sở hữu của các gói hoặc Sub system.
* Các đối tượng mà báo cáo này hướng đến bao gồm:
* Nhà phát triển: Người thực hiện nhiệm vụ phát triển hệ thống từ đầu vào là bản thiết kế và tài liệu để tạo thành đầu ra là một phiên bản có thể chạy được.
* Khách hàng: Khách hàng là người đặt hàng hệ thống và muốn có một hệ thống mới (system-to-be) tốt hơn hệ thống hiện thời (system-as-is). Trong khóa học này, khách hàng có thể coi như là giáo viên.
* Người viết tài liệu: Người sẽ viết tài liệu trong tương lai (các báo cáo, biên bản).

## **Tài liệu tham khảo**

* Các slides bài giảng trên courses.uet.vnu.edu.vn
* Mẫu tài liệu đặc tả yêu cầu phần mềm chuẩn IEEE.

## **Bố cục tài liệu**

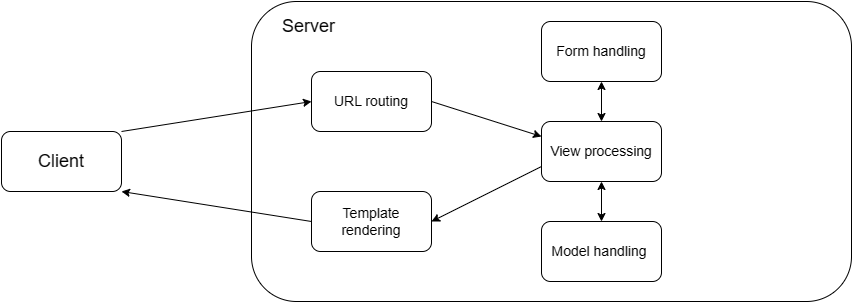
Tài liệu này gồm 4 phần chính:

* **Phần 1 Cấu trúc Hệ thống:** Biểu diễn tổng quan về các đối tượng, các Class trong hệ thống và cách chúng tương tác với nhau.
* **Phần 2 Sequence diagram, Class diagram cho các UC chính:** Một Use case sẽ được thực hiện thông qua các đối tượng cộng tác (collaborating objects). Có nhiều hình thức khác nhau để thể hiện tương tác giữa các đối tượng trong Use case. Từ các Squence diagram, nhanh chóng tạo ra sơ đồ lớp (class diagram), trong đó hiển thị các Class trong một gói hoặc các Class có sự cộng tác trong quá trình thực thi Use case.
* **Phần 3 Lược đồ cơ sở dữ liệu:** Biểu diễn tổng quan các bảng dữ liệu trong cơ sở dữ liệu, mối quan hệ và các ràng buộc giữa chúng
* **Phần 4 Mô phỏng UI:** Mô phỏng các chức năng.

# **Nội dung chính**

## **Cấu trúc hệ thống**

* + 1. **Tổng thể**

****

Trong hệ thống này bao gồm Client và Server, Client gửi yêu cầu và Server sẽ đáp lại tương ứng:

**1. URL Routing** : Khi người dùng gửi một yêu cầu, Django sử dụng URL của yêu cầu để xác định hàm View nào sẽ được gọi. Điều này được thực hiện trong ‘urls.py`, nơi đã định nghĩa danh sách các mẫu URL và các hàm View tương ứng của chúng.

**2. View Processing:** Hàm view nhận yêu cầu HTTP và xử lý nó. Tùy thuộc vào chi tiết của yêu cầu, View có thể xử lí ngay, thực hiện một số logic, hoặc tương tác với các dịch vụ khác để xử lí yêu cầu. Và cuối cùng là chỉ định template và dữ liệu để trả về cho Client.

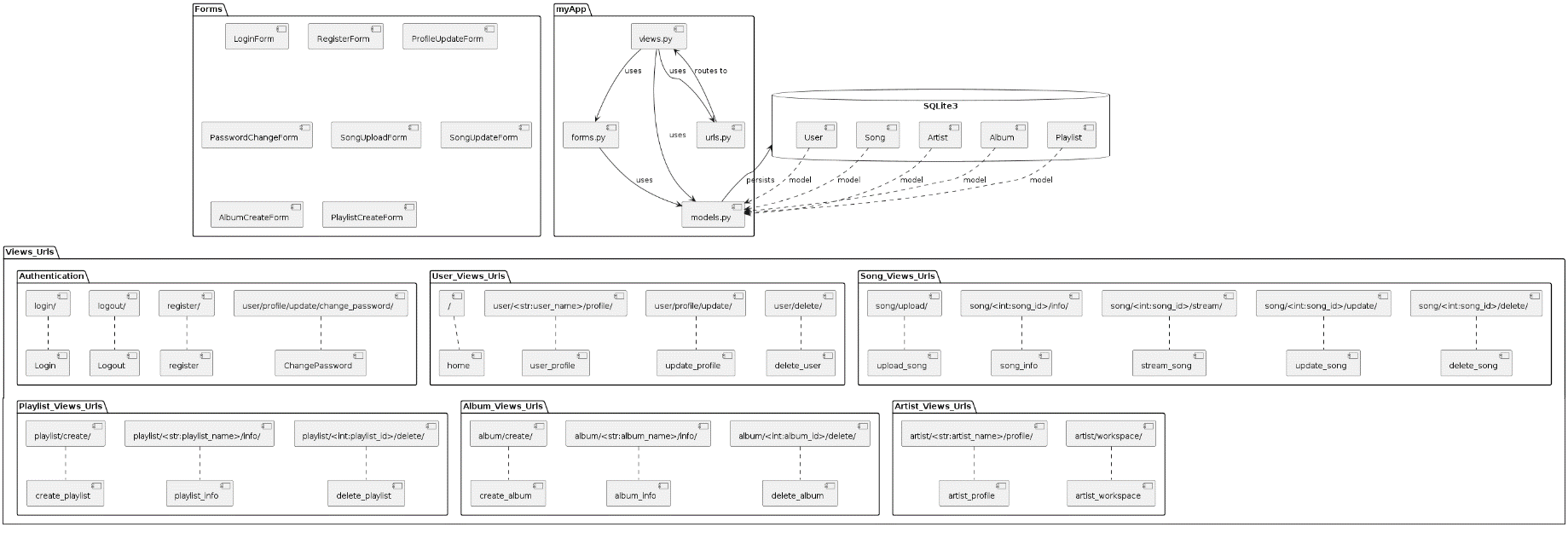
**3. Form Handling**: Nếu yêu cầu là một **biểu mẫu** được gửi (một yêu cầu POST), View sẽ tạo một thể hiện **form** và điền dữ liệu từ yêu cầu vào đó. Form sau đó được xác thực. Nếu form hợp lệ, dữ liệu được xử lý và người dùng được chuyển hướng đến trang thành công. Nếu form không hợp lệ, form sẽ được hiển thị lại với các thông báo lỗi.

**4. Model Handling:** Trong quá trình xử lý yêu cầu, hàm View có thể tương tác với Model để truy vấn dữ liệu từ cơ sở dữ liệu. Model trong Django đại diện cho cấu trúc dữ liệu và chúng được định nghĩa trong tệp `models.py` của mỗi ứng dụng Django.

**5. Template Rendering:** Một khi hàm View đã xử lý yêu cầu, nó chỉ định một template kết với dữ liệu đã chuẩn bị và trả lại dưới dạng phản hồi cho **Client**. Các template là các tệp HTML mẫu có thể bao gồm các phần tử ngôn ngữ template Django, cho phép dữ liệu động được bao gồm trong HTML.

Ví dụ, khi một người dùng muốn xem hồ sơ của một nghệ sĩ, họ sẽ điều hướng đến một URL như `/artist/<str:artist\_name>/profile/`. Django khớp URL này với hàm **View** `artist\_profile`. Hàm này truy xuất nghệ sĩ và các bài hát và album của họ từ cơ sở dữ liệu thông qua class Artist trong **Model**, sau đó hiển thị **template** `user/artist/profile.html` với dữ liệu này. Trang HTML đã chuẩn bị sau đó được trả về trình duyệt của người dùng.

* + 1. **Chi tiết về các thành phần của Views, Forms, Models, Urls**

****

Trên đây là tổng quát về các thành phần của **Views, Forms, Models, Urls:**

**- Forms:**  Từng class form trong Form.py sẽ chịu trách nhiệm xử lí tương ứng khi được View gọi sử dụng, ví dụ khi Login, hàm Login trong Views.py sẽ tạo một Instance của class LoginForm, điền các dữ liệu nhận được từ người dùng, vào Instance này và nó sẽ kiểm tra tính Validate của yêu cầu, ví dụ như có thiếu trường nào không (Tên đăng nhập, mật khẩu)

**- Models:** Từng class model trong Model.py sẽ chịu trách hiện xử lí tương ứng khi được View gọi sử dụng, ví dụ khi Login, hàm Login trong Views.py sẽ tạo một Instance của class User, và thông qua Instance này lấy mật khẩu của người dùng có Tên đăng nhập được gửi trong yêu cầu từ Database, so sánh và xác thực yêu cầu Login  
 **- Urls – Views:** Như trên hình vẽ, từng url trong Urls.py được nối với hàm view xử lí tương ứng trong Views.py, hàm view sẽ kết hợp với form và model để xử lí yêu cầu của người dùng và trả về kết quả.

## **Các Diagram cho UC chính**

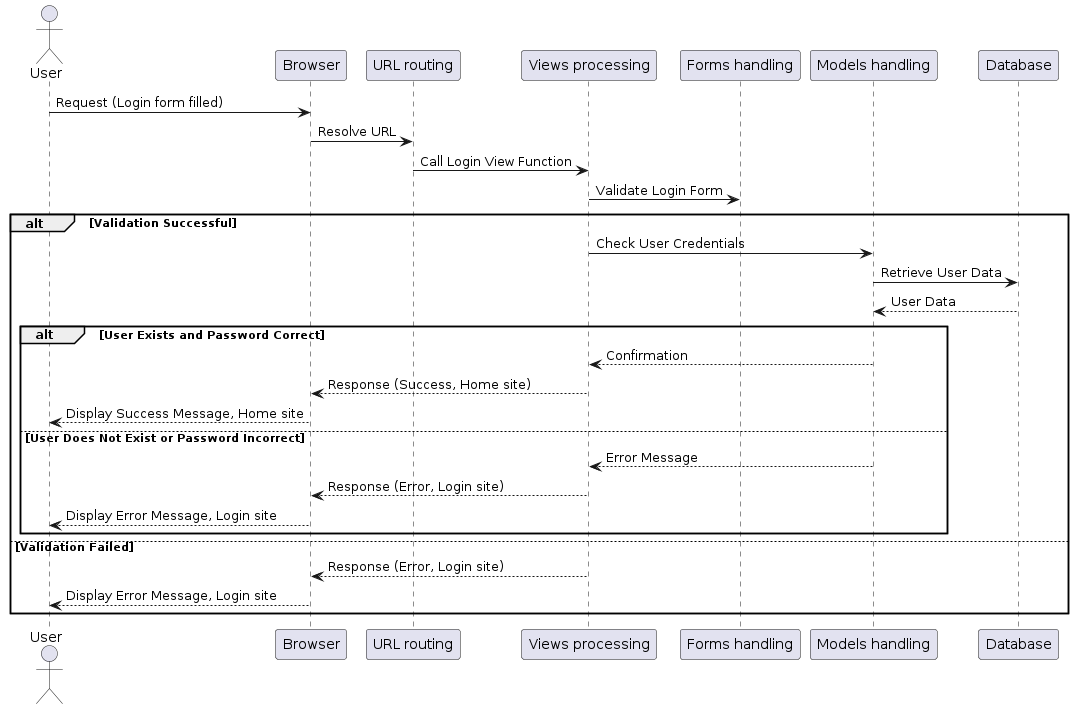
### **Sequence Diagram**

* **Đăng ký tài khoản**

Biểu đồ Register

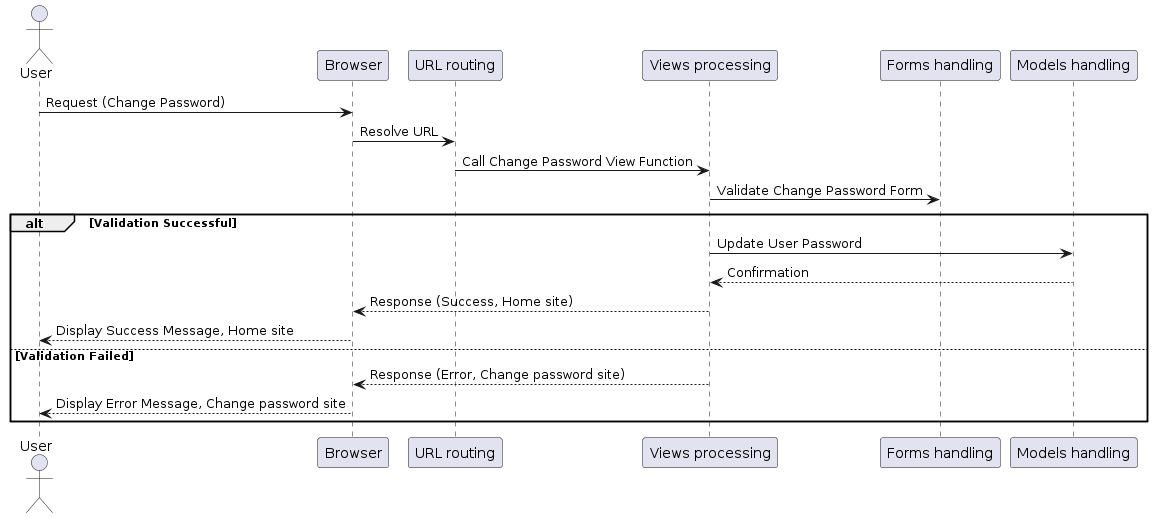

Biểu đồ 1.Register

* **Đăng nhập**

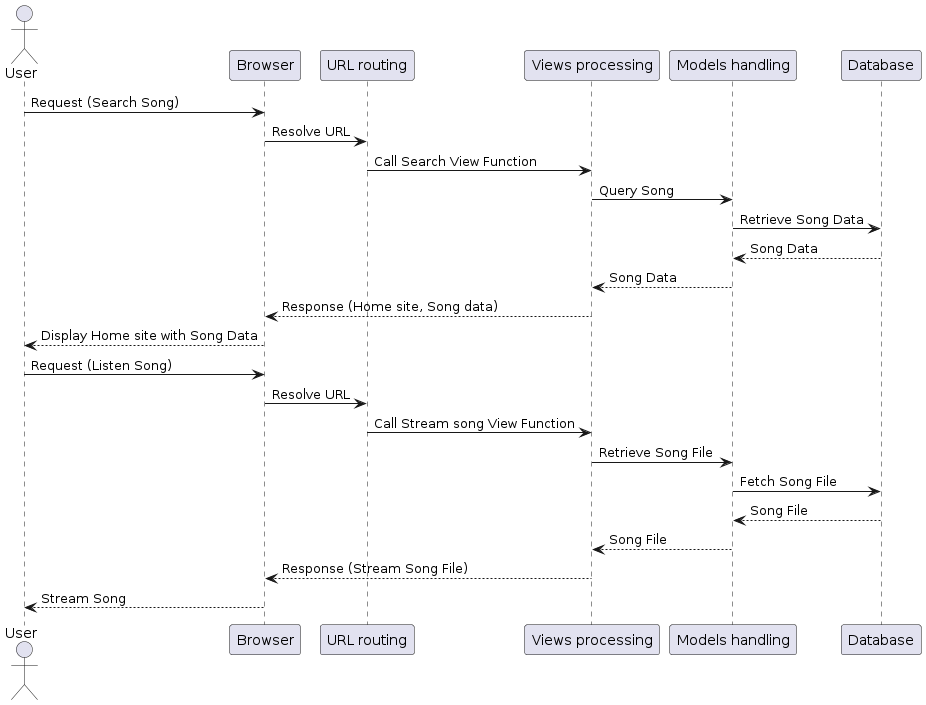
****

Biểu đồ 2. Login

* **Đổi mật khẩu**

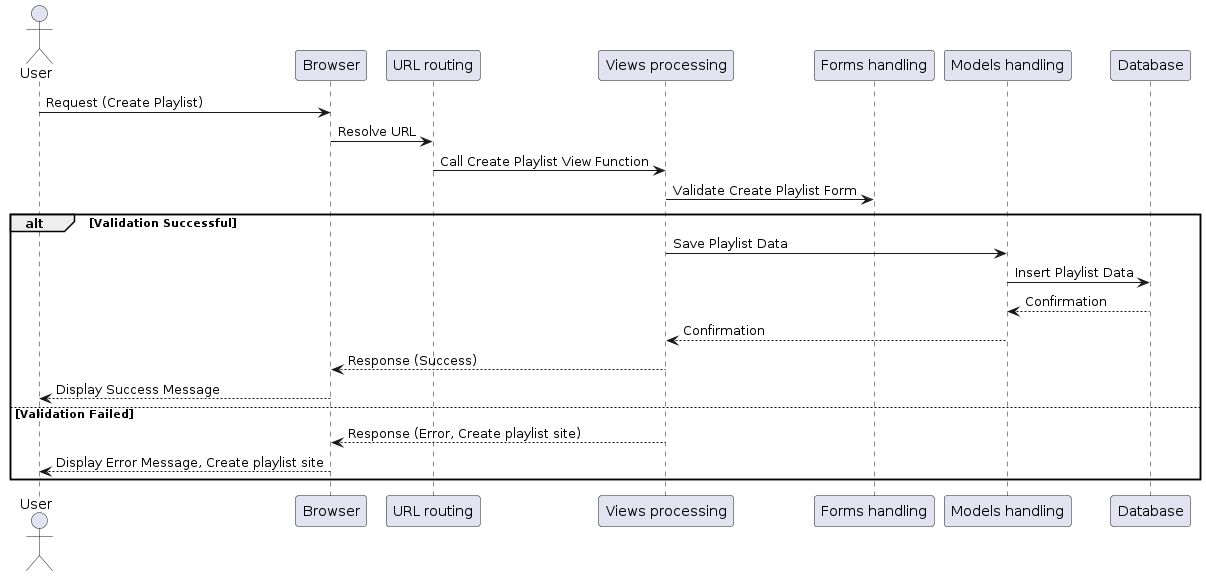


Biểu đồ 3. Change password

* **Tìm - Chơi bài hát**

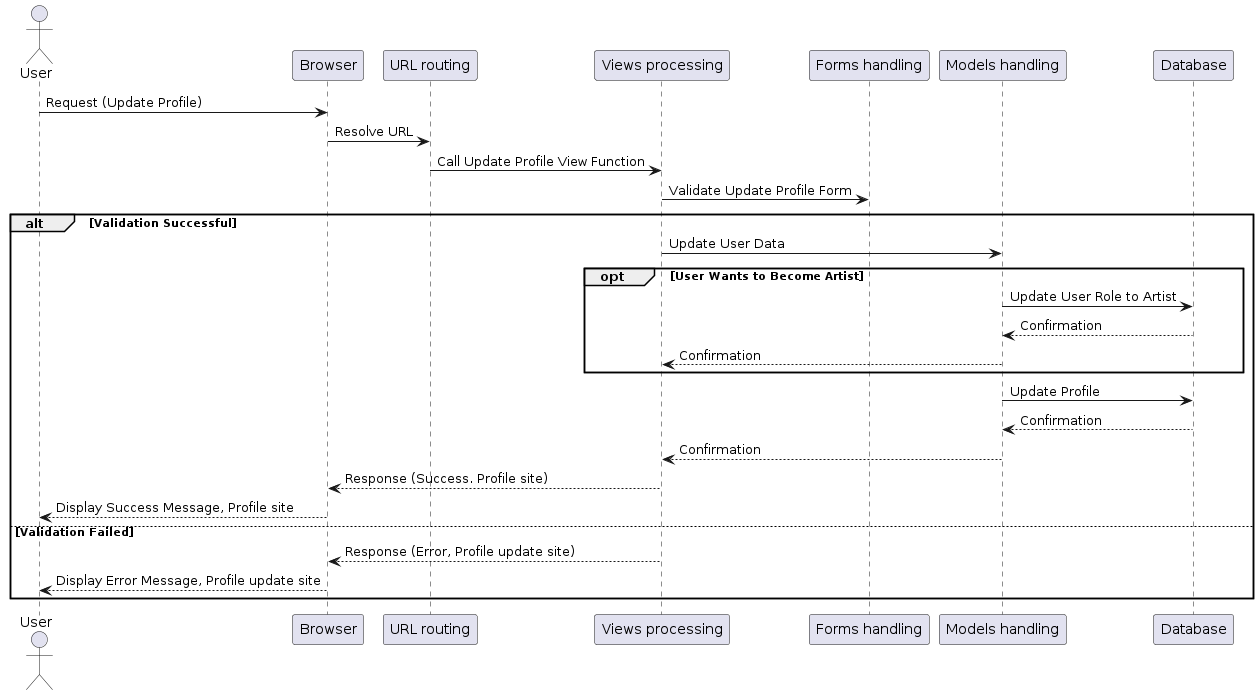
Biểu đồ 4.Search - Play song

* **Tạo playlist**

****

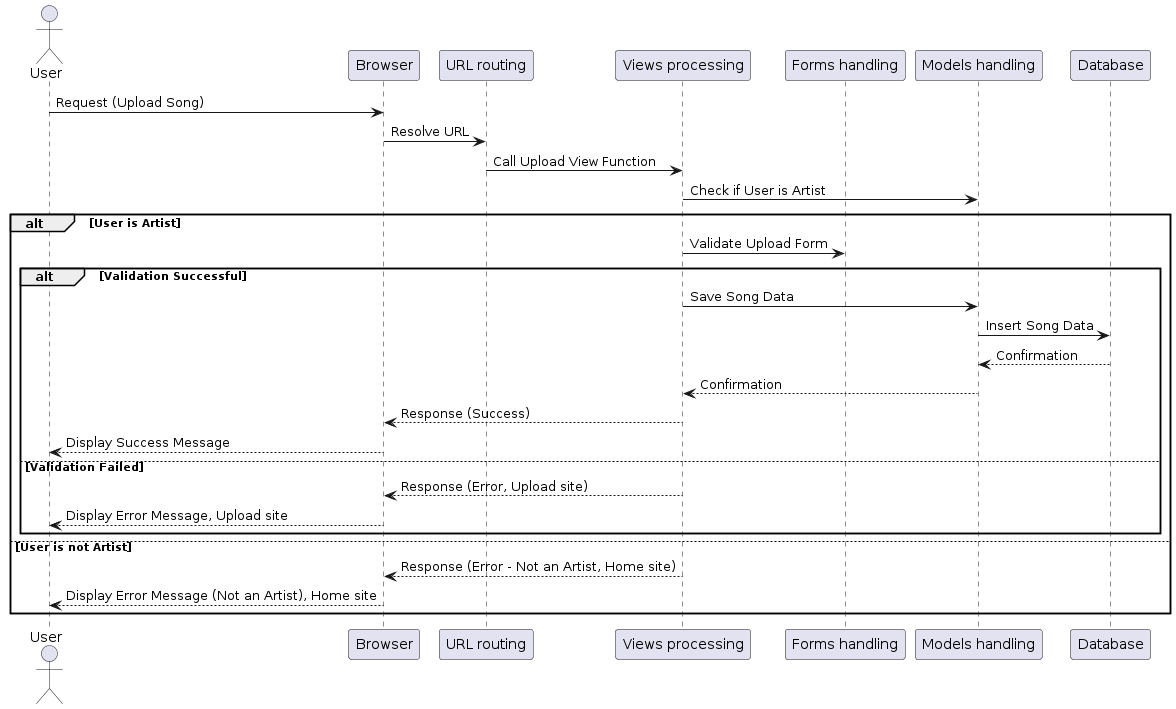
Biểu đồ 5. Create Playlist

* **Cập nhật thông tin – Đăng kí làm Nghệ sĩ**

****

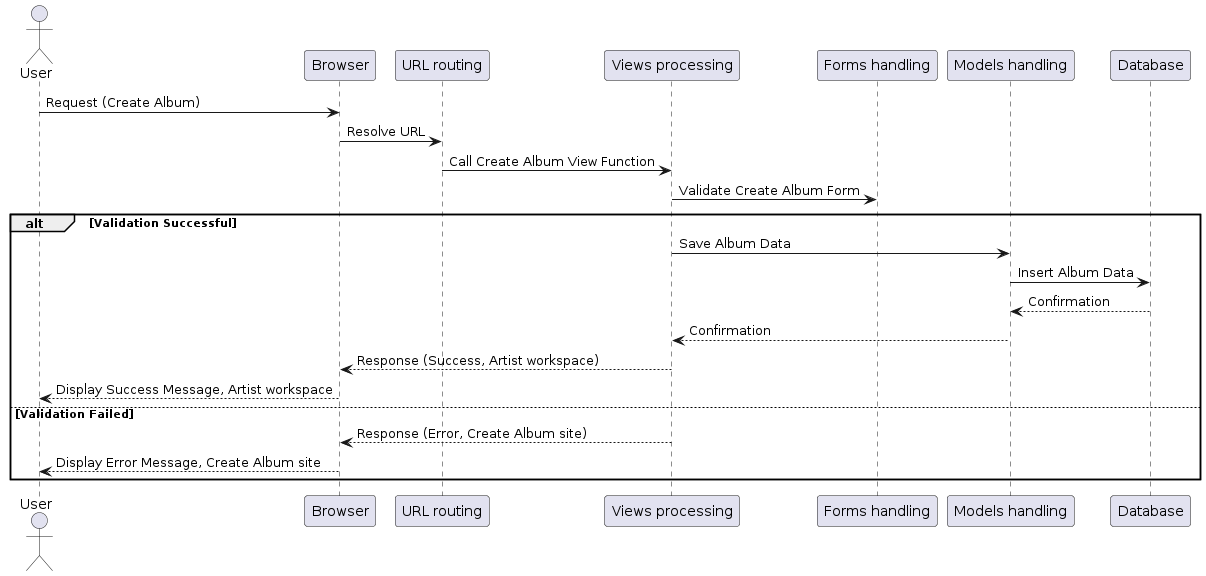
Biểu đồ 6. Update Profile - Become Artist

* **Phát hành nhạc**

****

Biểu đồ 7. Upload Song

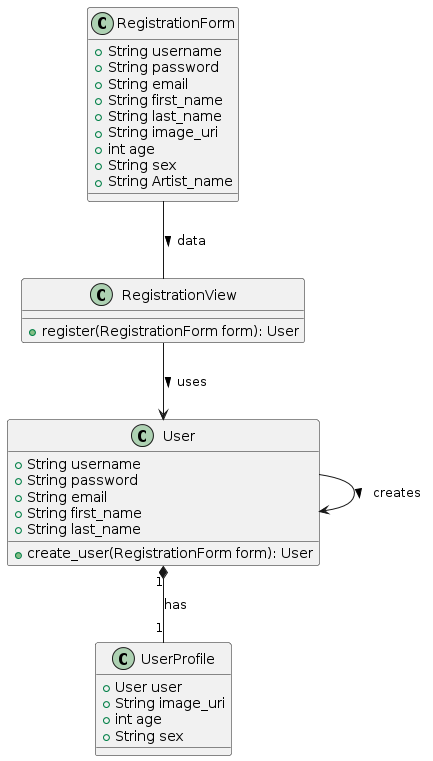
* **Tạo album**

****

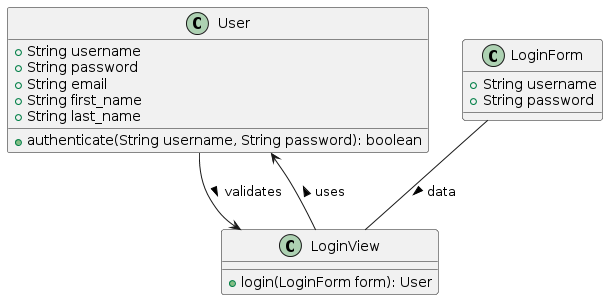
Biểu đồ 8. Create Album

### **Class Diagram**

* Các class liên quan đến Đăng ký

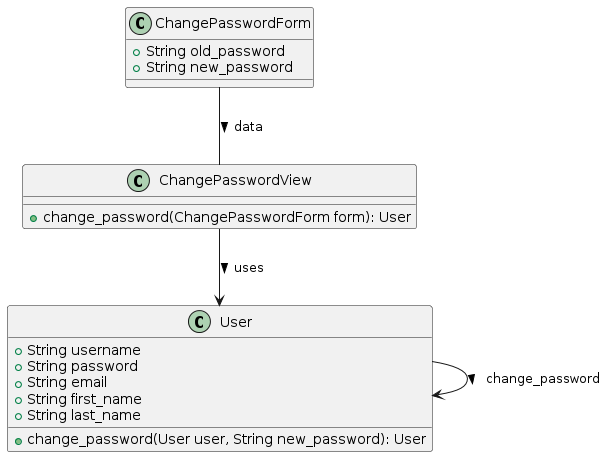


- Registration View sẽ sử dụng data trong Instance form của RegistrationForm, để thông qua class User trong Model tạo ra User và UserProfile tương ứng và lưu vào trong Database

* Các class liên quan đến Đăng nhập

- LoginView sẽ sử dụng data trong Instance form của LoginForm và xác thực thông tin bằng hàm authenticate() của class User

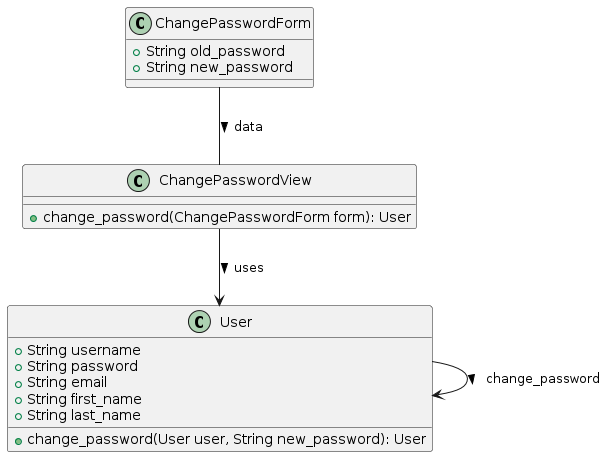
* Các class liên quan đến Đổi mật khẩu

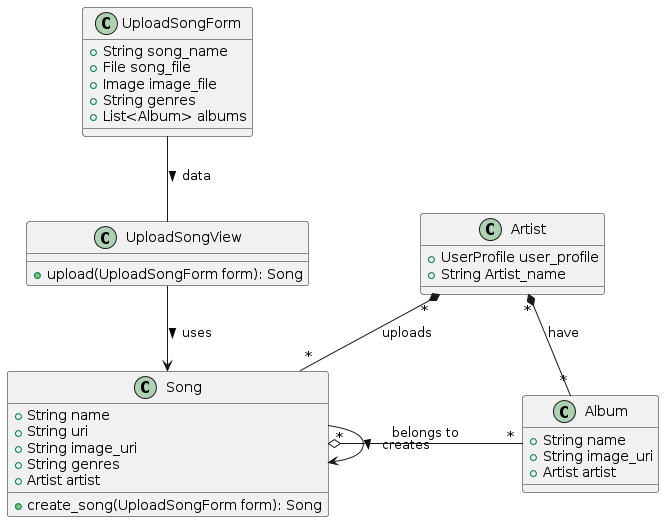


- ChangePasswordView sẽ sử dụng data trong Instance form của ChangePasswwordView và dùng hàm change\_password của class User để đổi mật khẩu

* Các class liên quan đến Chơi bài hát

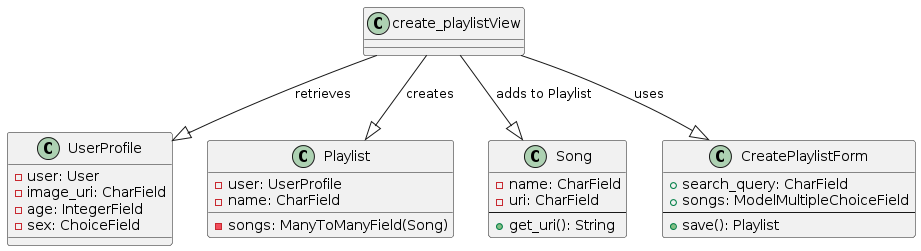
- stream\_songView sẽ lấy đường dẫn đến file bài hát trong hệ thống bằng hàm get\_uri() của class Song, sau đó tạo SteamHttpResponse để gửi đi

* Các class liên quan đến Phát hành nhạc
* Các class liên quan đến Tạo album



- UploadSongView sẽ dùng data từ Instance form của UploadSongForm để tạo bài hát từ hàm create\_song() của class Song, cùng với việc tạo các quan hệ với class Artist và Album

* Các class liên quan đến Tạo playlist



- create\_playlistView sẽ sử dụng data trong form để tạo Instance của Playlist, đồng thời tạo các mới quan hệ của Instance này với các Instance Song và UserProfile

## **Lược đồ cơ sở dữ liệu**

## **Mô phỏng UI**

|  |
| --- |
| **Đăng ký tài khoản** |
|  |
|  |
|  |

|  |
| --- |
| **Đăng nhập** |
|  |

|  |
| --- |
| **Trang chính** |
|  |

|  |
| --- |
| **Tìm kiếm nhạc** |
|  |

|  |
| --- |
| **Phát hành nhạc** |
|  |