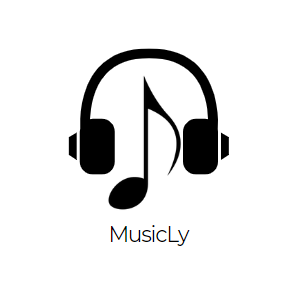
**DOCUMENTO DE REQUISITOS**

**MusicLy**

****

**Aluno:** Luana da Costa Cordeiro

**Data:** 28/05/2024

**Sumário**

[**1. Introdução 3**](#_38qczcheis4r)

[1.1. Requisitos 3](#_96ujnav8zqbb)

[1.1.1. O que são requisitos funcionais? 3](#_kin5kehcvqtr)

[1.1.2. O que são requisitos não funcionais? 4](#_upj3zh29bthf)

[1.2. Descrição do Usuário 4](#_appx0eih565m)

[1.2.1. Atores 4](#_mv9oz6s7k43g)

[**2. Requisitos Funcionais 5**](#_as3zrcntmh3)

[2.1. Sistema de Login (RF001) 5](#_yqrcrx5b8u01)

[2.2. Gerenciamento de Músicas para Administradores (RF002) 5](#_82894odod1ug)

[2.3. Gerenciamento de Álbuns para Administradores (RF003) 6](#_15ys9ajsy3g5)

[2.4. Gerenciamento de Artistas para Administradores (RF004) 6](#_hc9mhwip2dfq)

[2.5. Gerenciamento de Gêneros para Administradores (RF005) 6](#_5t69imyb08hh)

[2.6. Gerenciamento de Playlists para Usuários (RF006) 7](#_98xdgmp2n5w2)

[2.7. Gerenciamento de Álbuns para Artistas (RF007) 7](#_zbrxr6fkes0)

[**3. Requisitos Não-Funcionais 8**](#_z1gkcro39u5c)

[3.1. Usabilidade (RNF001) 8](#_gj7kzcnp1fiq)

[3.1.1. Responsividade 8](#_8zspj7kieohe)

[3.1.2. Validação de Campos 8](#_2ubme1gxhuh2)

[3.1.3. Mensagens de Aviso 8](#_boe7dwtjm6ab)

[3.2. Banco de Dados MySQL (RNF002) 8](#_oiw8haz0cyjl)

[3.3. PHP (RNF003) 9](#_yct9ph3x65ls)

[3.4. Frameworks (RNF004) 10](#_g24l6um08y42)

[3.4.1. Tailwind 10](#_uslur2y08np3)

[3.4.2. Bootstrap 11](#_ffpyoud2eis6)

[3.5. Requisítos Mínimos Para o Uso (RNF005) 11](#_89nftg8awpn3)

[3.5.1. Bootstrap 12](#_o78sxc44azuh)

[3.5.2. PHP 12](#_n59eiz4090dc)

[3.5.2.1. PHP 8.0 12](#_9xoyebz6lbqq)

[3.5.3. PDO 13](#_7air0wlgc8xt)

[**4. Referências Bibliográficas 13**](#_9nwkb4qw3exz)

# 1. Introdução

O MusicLy é um projeto que funciona como um Catálogo de Músicas. O administrador pode criar, observar, atualizar ou deletar músicas, álbuns, artistas e gêneros. É de desejo que o projeto também tenha diferentes atividades para o usuário comum e o artista. O usuário pode observar as músicas, álbuns, artistas e gêneros presentes no catálogo e pode criar playlists, adicionando músicas. O artista pode criar álbuns e adicionar músicas nesses álbuns, além de ver outros artistas e gêneros presentes também no catálogo. Esta aplicação pode ser utilizada por lojas de discos ou aplicativos que buscam armazenar as músicas, artistas e álbuns presentes em seu estabelecimento/software.

Este documento apresenta uma descrição dos requisitos do software MusicLy, detalhando os requisitos funcionais e não funcionais. Além disso, descreve os tipos de usuários e suas interações com o sistema, bem como as tecnologias e frameworks utilizados no desenvolvimento do projeto.

## 1.1. Requisitos

Os requisitos de software são as necessidades e expectativas dos usuários e outras partes em relação ao sistema que está sendo desenvolvido**.** Eles definem o que o sistema deve fazer, como ele deve se comportar e quais são os seus limites e restrições. Os requisitos são divididos em requisitos funcionais e não funcionais, definindo o que o software deve fazer e como ele deve operar em diferentes ambientes e situações. Esses requisitos moldam a experiência do usuário, a eficiência operacional e a adaptabilidade do software.

### 1.1.1. O que são requisitos funcionais?

Os requisitos funcionais descrevem as ações específicas que um sistema ou aplicativo deve ser capaz de executar. Eles são as capacidades concretas e as operações que o software deve realizar para atender às necessidades e expectativas do usuário.

**Exemplos:**

* Processamento de transações
* Gerenciamento de usuários
* Busca e filtragem de dados
* Geração de relatórios

### 1.1.2. O que são requisitos não funcionais?

Os requisitos não funcionais especificam como o software deve fazer isso. Eles são cruciais para garantir a qualidade e a eficiência do software, abrangendo aspectos como desempenho, segurança, confiabilidade e usabilidade. Esses requisitos não estão diretamente ligados às funções específicas do software, mas sim à sua operação e ambiente.

**Exemplos:**

* Desempenho
* Segurança
* Confiabilidade
* Usabilidade

## 1.2. Descrição do Usuário

Um usuário, no contexto da tecnologia da informação, é qualquer pessoa que interage com um sistema computacional, seja ele um software, um aplicativo, um site ou um dispositivo eletrônico.

### 1.2.1. Atores

**Administrador/Funcionário:** O administrador ou o funcionário pode editar a tabela do artista, do álbum e das músicas, podendo criar, deletar ou modificar o conteúdo dessas tabelas.

**Artista:** O artista pode adicionar álbuns e também introduzir músicas em seus respectivos álbuns. O artista não pode modificar a tabela de artistas.

**Usuário Comum:** O usuário comum pode criar playlists e adicionar músicas nas playlists criadas. O usuário não pode adicionar músicas (diretamente na tabela), álbuns ou artistas.

# 2. Requisitos Funcionais

## 2.1. Sistema de Login (RF001)

O sistema deve permitir que o usuário crie sua conta e faça o login, caso o usuário e a senha sejam válidos.

**Entradas:** Nome de usuário e senha.

**Saídas:** Mensagens de sucesso ou erro.

**Pré-condições:** Nenhuma.

* O usuário deve ser redirecionado para a página inicial caso o login seja bem sucedido.
* O sistema deve exibir uma mensagem de erro caso o login ou a senha estejam incorretos.
* O sistema deve exibir uma mensagem ao criar uma conta.
* O sistema deve exibir uma mensagem caso os inputs estejam vazios.

## 2.2. Gerenciamento de Músicas para Administradores (RF002)

O sistema deve permitir que o administrador crie novas músicas no catálogo, atualize informações de músicas existentes, delete músicas do catálogo e visualize a lista completa de músicas.

**Pré-condições:** O usuário deve ter permissões de administrador.

**Entradas:**  Informações das músicas (idMusica, nome, álbum, gênero, data\_lanç, etc).

**Saídas**: Mensagem de sucesso ou erro.

* O sistema deve exibir uma mensagem caso o processo seja executado corretamente.
* O sistema deve exibir uma mensagem caso ocorra um erro.
* O sistema deve permitir que o administrador escolha o artista da música.
* O sistema deve permitir que o administrador escolha o gênero da música.
* O sistema deve permitir que o administrador escolha o álbum da música.

## 2.3. Gerenciamento de Álbuns para Administradores (RF003)

O sistema deve permitir que o administrador crie novos álbuns, atualize informações de álbuns existentes, delete álbuns do catálogo e visualize a lista completa de álbuns.

**Pré-condições:** O usuário deve ter permissões de administrador.

**Entradas:**  Informações dos álbuns (idalbum, nome, data\_lanç, etc).

**Saídas**: Mensagem de sucesso ou erro.

* O sistema deve exibir uma mensagem caso o processo seja executado corretamente.
* O sistema deve exibir uma mensagem caso ocorra um erro.

## 2.4. Gerenciamento de Artistas para Administradores (RF004)

O sistema deve permitir que o usuário crie novos artistas, atualize informações de artistas existentes, delete artistas do catálogo e visualize a lista completa de artistas.

**Pré-condições:** O usuário deve ter permissões de administrador.

**Entradas:**  Informações dos artistas (idartista, nome, etc).

**Saídas**: Mensagem de sucesso ou erro.

* O sistema deve exibir uma mensagem caso o processo seja executado corretamente.
* O sistema deve exibir uma mensagem caso ocorra um erro.

## 2.5. Gerenciamento de Gêneros para Administradores (RF005)

O sistema deve permitir que o usuário adicione gêneros, atualize os gêneros existentes, delete gêneros do catálogo e visualize a lista completa de gêneros.

**Pré-condições:** O usuário deve ter permissões de administrador.

**Entradas:**  Informações do gênero (idgenero, nome, etc).

**Saídas**: Mensagem de sucesso ou erro.

* O sistema deve exibir uma mensagem caso o processo seja executado corretamente.
* O sistema deve exibir uma mensagem caso ocorra um erro.

## 2.6. Gerenciamento de Playlists para Usuários (RF006)

O sistema deve permitir que o usuário crie novas playlists, adicione músicas às playlists, remova músicas das playlists e visualize suas playlists e as músicas contidas nelas.

**Pré-condições:** O usuário deve estar autenticado.

**Entradas:**  Informações das playlists (nome, músicas, etc.).

**Saídas**: Mensagem de sucesso ou erro.

* O sistema deve exibir uma mensagem caso o processo seja executado corretamente.
* O sistema deve exibir uma mensagem caso ocorra um erro.

## 2.7. Gerenciamento de Álbuns para Artistas (RF007)

O sistema deve permitir que o artista crie novos álbuns, adicione músicas aos seus álbuns, atualize informações de seus álbuns e delete seus álbuns do catálogo.

**Pré-condições:** O usuário deve ter permissões de artista.

**Entradas:**  Informações dos álbuns (idalbum, nome, data\_lanç, etc).

**Saídas**: Mensagem de sucesso ou erro.

* O sistema deve exibir uma mensagem caso o processo seja executado corretamente.
* O sistema deve exibir uma mensagem caso ocorra um erro.

# 3. Requisitos Não-Funcionais

## 3.1. Usabilidade (RNF001)

A usabilidade em aplicativos é uma área crucial do design de experiência do usuário (UX), focada em garantir que os usuários possam interagir de forma eficiente e satisfatória com o aplicativo.

### 3.1.1. Responsividade

A forma como o software se ajusta a diferentes dispositivos e tamanhos de tela, sem comprometer a funcionalidade e a estética do aplicativo.

### 3.1.2. Validação de Campos

A validação dos campos de formulários veio com o objetivo de filtrar os dados para que essas informações sejam devidamente preenchidas. Desse modo, algo que poderia facilitar a validação e deixar mais intuitivo ao usuário seria a aplicação de máscaras nos campos de formulário.

### 3.1.3. Mensagens de Aviso

Mensagens de aviso são mensagens que aparecem quando o usuário realiza alguma ação, informando se a ação ocorreu de forma correta e atualizando o usuário sempre que ele interage com a interface da aplicação.

## 3.2. Banco de Dados MySQL (RNF002)

Um banco de dados nada mais é do que uma coleção de dados estruturados. Banco de dados é um um local onde dados são armazenados e gerenciados. A palavra relacional significa que os dados armazenados estão organizados em tabelas e cada tabela está relacionada de alguma maneira. Caso o software não suporte o modelo relacional, então chamamos DBMS.

MySQL é um sistema gerenciador de banco de dados relacional de código aberto usado na maioria das aplicações gratuitas para gerir suas bases de dados. O MySQL utiliza a linguagem SQL (Structure Query Language – Linguagem de Consulta Estruturada), que é a linguagem mais popular para inserir, acessar e gerenciar o conteúdo armazenado num banco de dados.

MySQL é um Banco de Dados relacional (RDBMS – *Relational Database Management Systems*) com um modelo de cliente-servidor. [RDBMS](https://pt.wikipedia.org/wiki/Banco_de_dados_relacional) é um software de código aberto ou serviço usado na criação e gerenciamento de bancos de dados baseados no modelo relacional.

**Vantagens:**

Uma das principais vantagens do MySQL é sua natureza de código aberto. Isso significa que qualquer pessoa pode baixar, usar e modificar o MySQL de acordo com suas necessidades.

* Código aberto.
* Facilidade de uso.
* Compatibilidade.
* Suporte da comunidade.
* Segurança.

## 3.3. PHP (RNF003)

PHP é um acrônimo recursivo para Hypertext Preprocessor. É uma linguagem de script open source de uso geral e adequada para o desenvolvimento web, podendo ser embutida dentro do HTML. Os Scripts PHP são executados no servidor e é gratuito para baixar e usar. O PHP pode gerar conteúdo de página dinâmico, criar, abrir, ler, escrever, excluir e fechar arquivos no servidor, coletar dados de formulário, enviar e receber cookies, adicionar, excluir, modificar dados em seu banco de dados, pode ser usado para controlar o acesso do usuário e criptografar dados.

**Vantagens:**

A melhor coisa em usar o PHP é que ele é extremamente simples para um iniciante, mas oferece muitos recursos avançados para um programador profissional.

* É executado em várias plataformas (Windows, Linux, Unix, Mac OS X, etc.).
* É compatível com quase todos os servidores usados ​​atualmente (Apache, IIS, etc.).
* Suporta uma ampla variedade de bancos de dado.
* É gratuito.
* É fácil de aprender e funciona com eficiência no lado do servidor.

## 3.4. Frameworks (RNF004)

Frameworks são estruturas compostas por um conjunto de códigos genéricos que permite o desenvolvimento de sistemas e aplicações. Um framework funciona como uma espécie de template ou modelo que, quando utilizado, oferece certos artifícios e elementos estruturais básicos para a criação de alguma aplicação ou software.

O framework utilizados no projeto foi:

* Bootstrap.

### 3.4.1. Tailwind

Tailwind CSS funciona verificando todos os seus arquivos HTML, componentes JavaScript e quaisquer outros modelos em busca de nomes de classes, gerando os estilos correspondentes e, em seguida, gravando-os em um arquivo CSS estático.

**Vantagens:**

* Utiliza um fluxo de trabalho que prioriza o utilitário para construir componentes complexos a partir de um conjunto restrito de utilitários primitivos.
* Estiliza elementos em estado interativos, como passar o mouse, focar e muito mais, usando modificadores condicionais.
* Gerencia a duplicação e mantém os projetos sustentáveis criando abstrações reutilizáveis.
* Cria interfaces de usuário totalmente responsivas que se adaptam a qualquer tamanho de tela usando modificadores responsivos.
* Otimiza o site para modo escuro diretamente em seu HTML usando o modificador de modo escuro.
* Personaliza a estrutura.

### 3.4.2. Bootstrap

Bootstrap é um kit de ferramentas de front-end poderoso e repleto de recursos. É um [framework web](https://pt.wikipedia.org/wiki/Framework_para_aplica%C3%A7%C3%B5es_web) com [código-fonte aberto](https://pt.wikipedia.org/wiki/Software_livre_e_de_c%C3%B3digo_aberto) para desenvolvimento de componentes de [interface](https://pt.wikipedia.org/wiki/Interface) e [front-end](https://pt.wikipedia.org/wiki/Front-end) para [sites](https://pt.wikipedia.org/wiki/Site) e [aplicações web](https://pt.wikipedia.org/wiki/Aplica%C3%A7%C3%A3o_web), usando [HTML](https://pt.wikipedia.org/wiki/HTML), [CSS](https://pt.wikipedia.org/wiki/Cascading_Style_Sheets) e [JavaScript](https://pt.wikipedia.org/wiki/JavaScript), baseado em modelos de [design](https://pt.wikipedia.org/wiki/Design) para a [tipografia](https://pt.wikipedia.org/wiki/Tipografia), melhorando a [experiência do usuário](https://pt.wikipedia.org/wiki/Experi%C3%AAncia_do_usu%C3%A1rio) em um site amigável e [responsivo](https://pt.wikipedia.org/wiki/Responsividade).

**Vantagens:**

* Velocidade de desenvolvimento.
* Compatibilidade.
* Responsividade.
* Open source.
* Fácil adoção.
* Personalização.
* Popularidade.
* Integração.

## 3.5. Requisítos Mínimos Para o Uso (RNF005)

São os requisitos de versão mínima do navegador para rodar o site, compatibilidade com os principais navegadores (Chrome, Firefox, Safari, Edge), requisitos de hardware e software para o servidor e os dispositivos clientes.

### 3.5.1. Bootstrap

* Navegadores alternativos que usam a versão mais recente do WebKit, Blink ou Gecko (tanto diretamente, quanto através da API web view da plataforma) não suportam explicitamente o Bootstrap.
* Suporte a navegadores modernos como Google Chrome, Mozilla Firefox, Safari, Microsoft Edge.
* Suporte básico ao Internet Explorer 11 (embora o Bootstrap 5 tenha removido o suporte oficial, projetos legados podem exigir).
* Organização do projeto que permita a inclusão dos arquivos principais do Bootstrap (CSS e JS) e suas dependências, como Popper.js e jQuery (caso necessário).
* Estrutura de pastas adequada para arquivos de estilos personalizados, scripts e assets.
* Minificação dos arquivos CSS e JS para melhorar o desempenho do carregamento da página.

### 3.5.2. PHP

* Configuração adequada do servidor para processar arquivos PHP (geralmente via módulo ou FastCGI).
* Apache, Nginx, IIS ou qualquer outro servidor web compatível com PHP.
* PHP é compatível com diversos sistemas operacionais, incluindo Linux, Windows, macOS e outros sistemas baseados em Unix.

#### 3.5.2.1. PHP 8.0

* Servidor Web: Apache 2.4 ou Nginx 1.18.
* Sistema Operacional: Ubuntu 20.04, Windows 10, macOS Catalina.
* Sistema Operacional: Ubuntu 20.04, Windows 10, macOS Catalina.
* Extensões: ctype, fileinfo, mbstring, openssl, PDO, tokenizer, xml.
* Memória: 1GB RAM (mínimo para produção).
* Banco de Dados: MySQL 5.7+, MariaDB 10.2+, PostgreSQL 10+, SQLite 3.31+.
* Ferramentas: Composer, Git.

### 3.5.3. PDO

* **PHP 5.1.0 ou superior:** PDO foi introduzido no PHP 5.1.0, então você precisa pelo menos dessa versão. No entanto, é recomendado usar uma versão mais recente do PHP, como PHP 7.x ou 8.x, para garantir suporte e segurança.
* **Servidor de banco de dados compatível:** O servidor de banco de dados que você planeja usar deve estar instalado e configurado adequadamente. Isso inclui servidores como MySQL, PostgreSQL, SQLite, etc.
* **Servidor Web:** Um servidor web como Apache, Nginx, ou IIS, configurado para processar arquivos PHP.
* **Sistema Operacional:** Qualquer sistema operacional compatível com PHP, como Linux, Windows, ou macOS.

# 4. Referências Bibliográficas

[https://pt.linkedin.com › pulse › requisitos-de-software-u…](https://pt.linkedin.com/pulse/requisitos-de-software-um-guia-abrangente-para-o-sistemas-almeida#:~:text=Os%20requisitos%20de%20software%20podem,os%20seus%20limites%20e%20restri%C3%A7%C3%B5es.)

<https://querobolsa.com.br/revista/requisitos-funcionais-e-nao-funcionais>

https://escolalbk.com.br

<https://rankmyapp.com>

https://coodesh.com

<https://www.hostinger.com.br/tutoriais/o-que-e-mysql>

<https://www.techtudo.com.br/noticias/2012/04/o-que-e-e-como-usar-o-mysql.ghtml>

https://awari.com.br

https://www.hostgator.com.br/blog/mysql-e-suas-vantagens/

<https://www.w3schools.com/>

<https://www.php.net/manual/pt_BR/intro-whatis.php>

<https://pt.wikipedia.org/wiki/Bootstrap_(framework_front-end)>

<https://tailwindcss.com/>

<https://tailwindcss.com/>

<https://www.hostmidia.com.br/blog/bootstrap-10-razoes-para-usar/>

https://www.dio.me

Acessado em 01/05/2024