Sistema de Música Richardjbl

Cauã Richlin, Arthur Zimmermann

Engenharia de Software
Universidade da Região de Joinville (UNIVILLE) – Joinville, SC – Brazil

caua.richlin@univille.br, arthur.zimmermann@univille.br

1. Introdução

Estamos desenvolvendo uma plataforma de música que promete revolucionar a forma como os usuários interagem com suas playlists e artistas, inspirando-se no modelo do Spotify. Com esta plataforma, os usuários poderão explorar um vasto catálogo de músicas, gerenciar álbuns e criar playlists personalizadas. Além disso, a interação entre os usuários será incentivada, permitindo a troca de descobertas musicais e o compartilhamento de playlists. É um aplicativo para qualquer um que curte ouvir música boa, onde você pode ouvir seus artistas favoritos e descobrir novos lançamentos na hora.

2. Requisitos Funcionais

Os requisitos funcionais do Sistema Richardjbl serão apresentados em forma de história de usuário.

2.1. História de Usuário 01

Como um usuário, eu quero me cadastrar no sistema de música, fornecendo meu nome, e-mail e senha, para poder acessar minha biblioteca de músicas e playlists personalizadas.

Diagrama de Classe (UML):

- **Entidades**:
    - **Usuário**: Contém atributos como `id`, `nome`, `email`, `senha`.

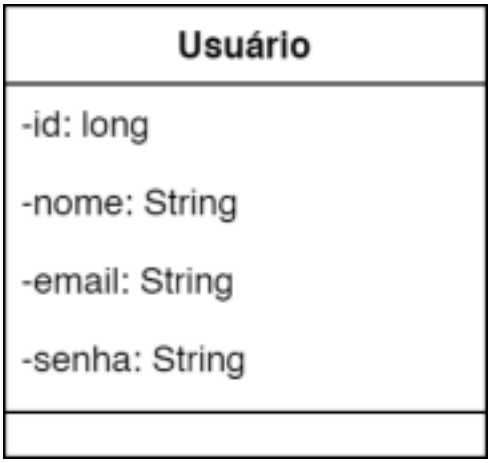

Figura 1. Diagrama de classe das entidades da História de Usuário 01.

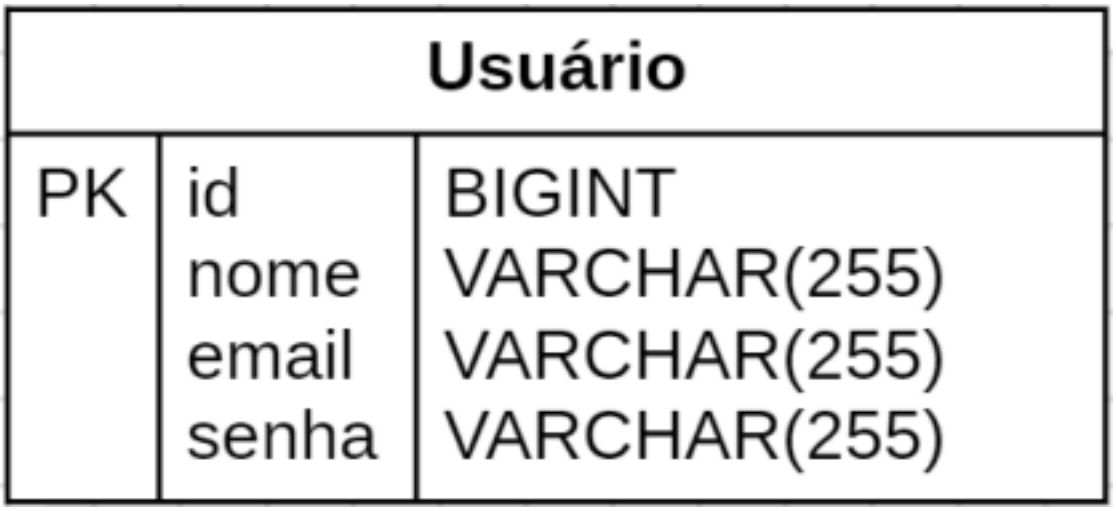

Figura 2. Modelo Entidade Relacionamento da História de Usuário 01.

2.2. História de Usuário 02

Como um usuário, eu quero criar e organizar minhas playlists pessoais, para poder gerenciar minhas músicas e artistas favoritos de forma personalizada.

Modelo Entidade Relacionamento (MER):

- **Entidades**:
  - **Música**: Contém `id`, `nome`, `artista`.
  - **Playlist:** Contém `id`, `nome`, `descrição.`
  - **Artista:** Contém `id`, `nome`, `gênero.`

Diagrama UML:

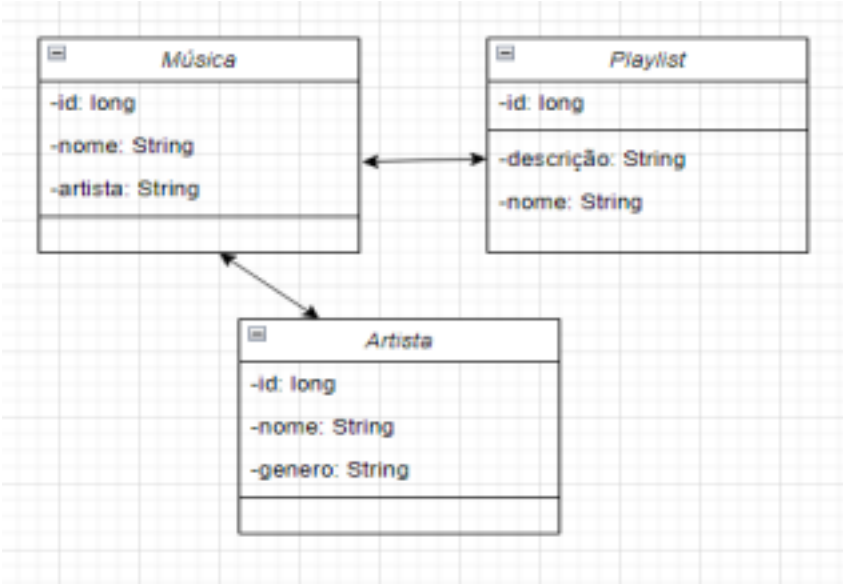

Figura 3. Diagrama de classe das entidades da História de Usuário 02.

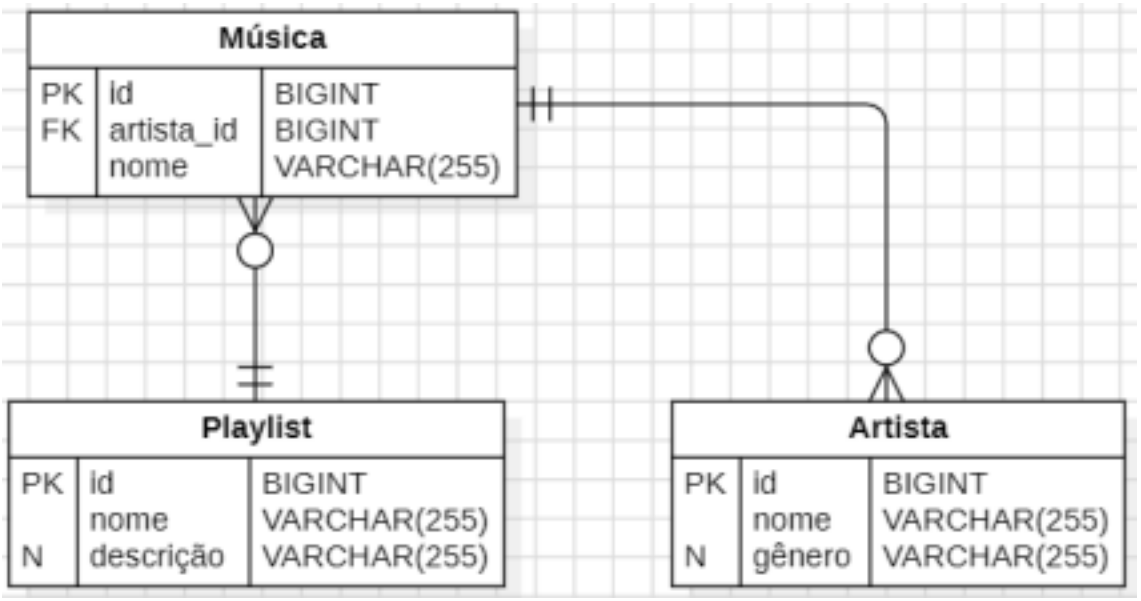

Figura 4. Modelo Entidade Relacionamento da História de Usuário 02.

3. Codificação

Para a construção de um sistema de música similar ao Spotify, iremos estruturar o banco de dados e os relacionamentos da seguinte forma:

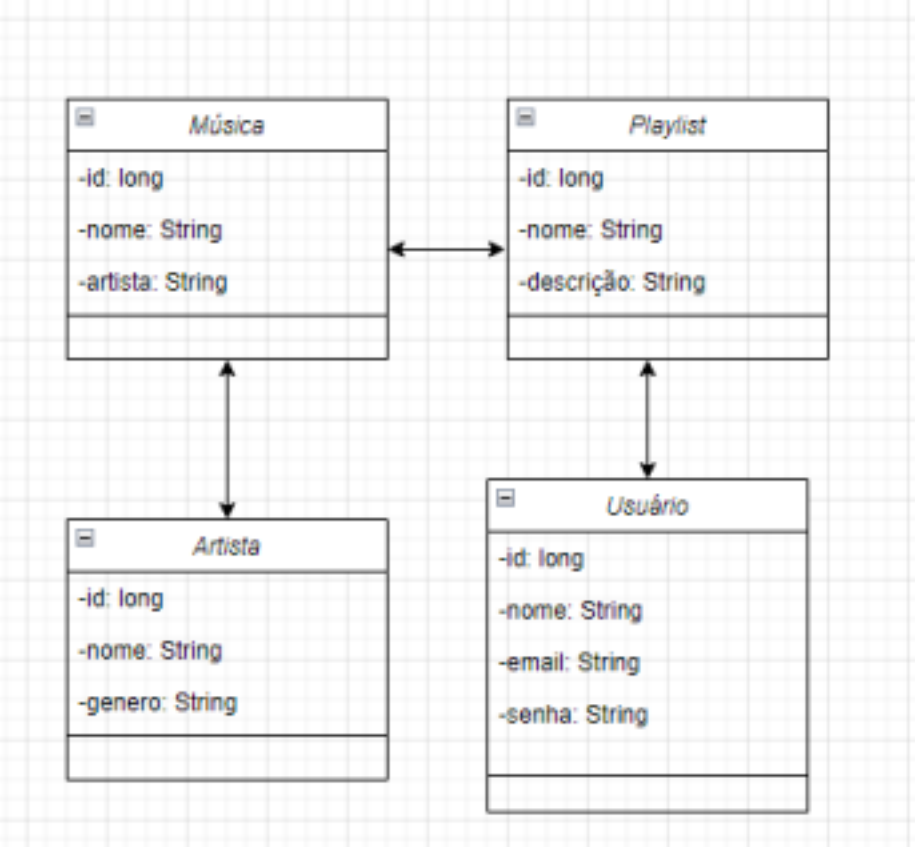

Figura 5. Diagrama de classe do Sistema de música Boombox

3.1. Entidade Música

Abaixo, mostramos um exemplo de como uma entidade pode ser codificada utilizando o framework JPA (Java Persistence API) para criar uma classe que represente o conceito de "Música" no sistema.

```
@Data
@NoArgsConstructor
@Entity
public class Musica {
    @Id
    @GeneratedValue
    private long id;
    @Column(nullable = false)
    private String nome;
    @Column(nullable = false)
    private long artista_id;
```

```
}
```

Figura 6. Código da entidade Música

4. Banco de dados

O banco de dados será estruturado utilizando tabelas que refletem as entidades do sistema e seus relacionamentos. Abaixo, apresentamos o Modelo Entidade Relacionamento (MER) completo do sistema Richardjbl.

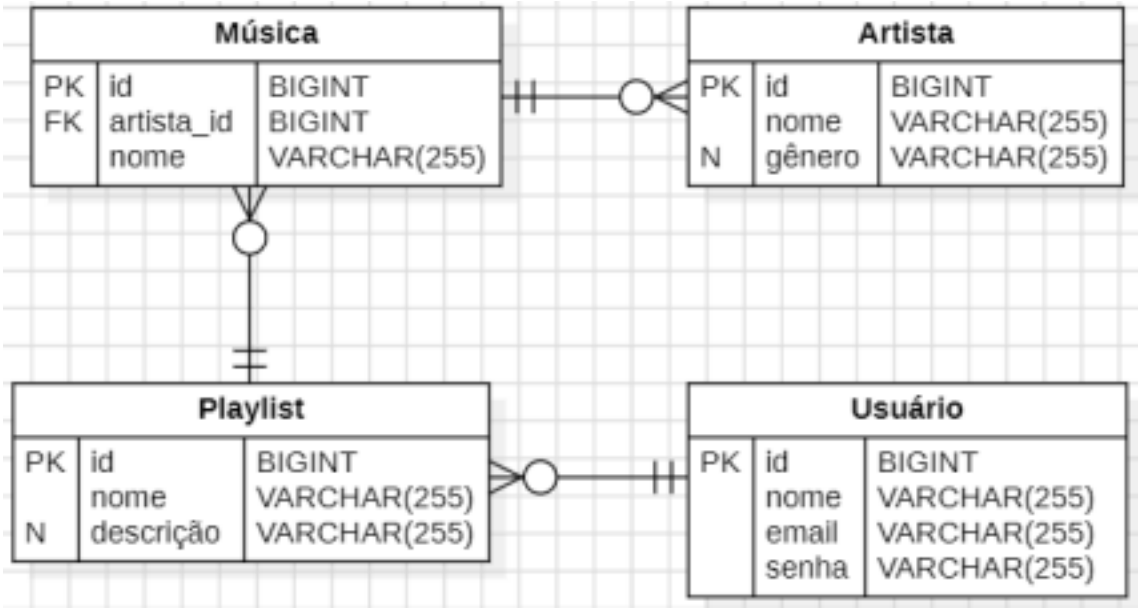

Figura 7. Modelo Entidade Relacionamento do Sistema de Música

5. Conclusão

Neste documento, destacamos as principais funcionalidades e entidades do sistema de música Richardjbl, inspirado em plataformas de streaming populares. O nosso foco é proporcionar uma experiência envolvente e personalizada, permitindo que os usuários se conectem facilmente com suas playlists, músicas e artistas favoritos. Ao desenvolver o sistema, buscamos oferecer uma experiência personalizada para o usuário, focando na interação com playlists, músicas e artistas favoritos.