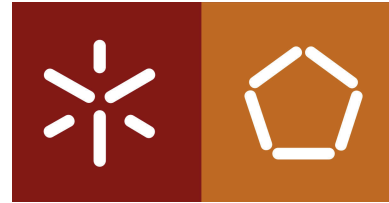


SpotifUM



Programação Orientada aos Objetos - Projecto prático



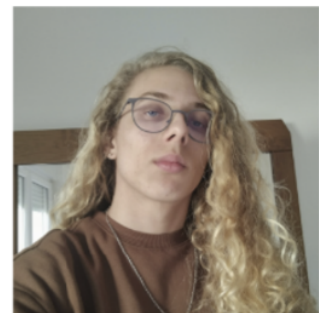
SpotifUM



Gabriel Dantas
a107291



Simão Oliveira
a107322



José Fernandes
a106937

13.05.2025

Grupo TP24: Gabriel Dantas (a107291), José Fernandes (a106937), Simão Oliveira(a107322)

Universidade do Minho - Licenciatura em Engenharia Informática, Programação Orientada aos Objetos

Índice

1. Resumo	1
2. Arquitetura de classes	1
2.1. Model	1
2.1.1. Classe Music	1
2.1.2. Classe MultimediaMusic	2
2.1.3. Classe Album	2
2.1.4. Classe Plan	2
2.1.5. Classe User	3
2.1.6. Classe MusicReproduction	3
2.1.7. Classe Playlist	3
2.1.8. Classe PlaylistRandom	4
2.1.9. Classe PlaylistFavorites	4
2.2. Facade	4
2.2.1. Classe SpotifUM	4
2.3. Controlador	5
2.3.1. Classe Controller	5
2.4. View	5
2.4.1. Classe Menu	6
2.4.2. Classe LoginMenu	6
2.4.3. Classe MainMenu	7
2.4.4. Classe ReproductionMenu	8
2.4.5. Classe ProfileMenu	10
2.4.6. Classe ProfileEditorMenu	11
2.4.7. Classe PlanMenu	11
2.4.8. Classe PlaylistMenu	12
2.4.9. Classe PlaylistCreationMenu	13
2.4.10. Classe AdminMenu	14
2.4.11. Classe StatisticsMenu	16
2.5. Utils	18
2.5.1. Classe BrowserOpener	18
2.5.2. Classe MusicPlayer	18
2.5.3. Classe Serialization	18
2.6. Excessões	18
3. Funcionalidades Avançadas	19

3.1. Reprodução das músicas	19
3.2. Criação das playlists “Feito para você”	20
4. Testes	20
5. Conclusão e trabalho futuro	20

1. Resumo

O presente relatório documenta o desenvolvimento do sistema SpotifUM, uma aplicação de streaming de música desenvolvida no âmbito da unidade curricular de Programação Orientada aos Objetos. Este projeto implementa uma plataforma que simula funcionalidades de serviços de streaming musical, permitindo aos utilizadores reproduzir músicas, criar playlists personalizadas e interagir com um catálogo musical diversificado.

A aplicação foi estruturada seguindo o padrão arquitetural Model-View-Controller (MVC), garantindo uma clara separação de responsabilidades entre a lógica de negócio, a interface com o utilizador e o controle de fluxo da aplicação. Foram implementados diferentes tipos de utilizadores e planos de subscrição, possibilitando níveis variados de acesso às funcionalidades do sistema, desde um plano gratuito com recursos limitados até planos premium com funcionalidades avançadas.

O SpotifUM incorpora mecanismos de serialização para persistência de dados, reprodução de conteúdo multimédia e um sistema de estatísticas que permite analisar os padrões de utilização. Este documento descreve detalhadamente a arquitetura de classes, as principais funcionalidades implementadas e as soluções técnicas adotadas para responder aos requisitos especificados no enunciado do projeto.

2. Arquitetura de classes

Nesta secção descrevemos cada entidade da aplicação individualmente, explicando a sua função e os principais atributos. Está disponibilizado, na mesma diretoria deste relatório, o diagrama de classes que descreve a arquitetura do programa criado.

2.1. Model

O Model representa a camada de dados e lógica interna da aplicação. Aqui encontram-se as classes que definem as entidades principais do sistema, como Music, User ou Playlist. Esta camada é responsável por armazenar, gerir e atualizar os dados, mantendo a lógica do domínio separada da interface e do controlo.

2.1.1. Classe Music

A classe Music representa uma música na aplicação. Cada objeto desta classe contém informação essencial sobre uma música individual, incluindo o seu identificador, título, artista, género e duração. Esta classe é central para a aplicação, sendo usada em várias funcionalidades, como a reprodução de músicas, criação de playlists ou análise de estatísticas de reprodução. Além disso, permite aceder e modificar os dados da música, garantindo flexibilidade na gestão do conteúdo.

2.1.2. Classe `MultimediaMusic`

No enunciado era também mencionada a existência de músicas que possuem conteúdos multimédia associados. Para dar resposta a esse requisito, foi criada a classe `MultimediaMusic` como uma subclasse de `Music`. Esta herda todos os atributos e comportamentos da classe `Music`, acrescentando um novo atributo específico para este tipo de conteúdo.

Esta abordagem permite reutilizar a estrutura base da música, estendendo-a de forma natural para incluir o conteúdo multimédia adicional.

Quando o utilizador reproduzir uma música deste tipo, é aberto o respetivo url associado no *browser* default do dispositivo.

2.1.3. Classe `Album`

Cada música da aplicação está associada a um álbum, seja este um single ou parte de uma coleção mais extensa. Para representar essa relação, foi criada a classe `Album`, responsável por armazenar informação relevante sobre o álbum, como o seu nome, o artista responsável e a lista de músicas que o compõem.

Esta classe permite organizar as músicas de forma estruturada e facilita funcionalidades como a reprodução de um álbum específico.

2.1.4. Classe `Plan`

Existem diferentes tipos de planos, que determinam as permissões atribuídas a cada utilizador. Estes planos regulam funcionalidades como o acesso a uma biblioteca pessoal, reprodução livre de músicas, possibilidade de passar faixas à frente, entre outras.

A classe `Plan` é abstrata, o que obriga todas as suas subclasses a implementarem métodos que definem como os pontos são incrementados quando o utilizador reproduz uma música e permissões específicas, tal como `canAccessLibrary()` que define se o utilizador tem acesso a uma biblioteca pessoal.

Quando um utilizador altera o seu plano, os pontos acumulados até então são automaticamente transferidos para o novo plano, garantindo que não há perda de progresso ou recompensas. Adicionalmente, no caso de migração para um plano sem permissões de acesso à biblioteca — como acontece com planos gratuitos — a biblioteca pessoal do utilizador não é eliminada. Apesar de o utilizador não poder interagir com a biblioteca durante esse período, os seus dados permanecem intactos e associados à sua conta. Isto garante que, caso o utilizador volte a subcrever um plano com permissões de biblioteca no futuro, a sua biblioteca anterior estará disponível tal como foi deixada, assegurando uma experiência contínua e sem perdas de informação.

Esta estrutura permite acrescentar facilmente novos planos no futuro, bastando definir os comportamentos relativos aos pontos e às permissões. Da mesma

forma, é possível adicionar novas permissões à arquitetura, garantindo flexibilidade e extensibilidade.

Atualmente, a aplicação implementa três planos:

- PlanFree
- PlanPremiumBase
- PlanPremiumTop

2.1.5. Classe User

A classe User representa um utilizador da aplicação. Cada utilizador possui um conjunto de informações pessoais e está associado a um plano de subscrição (Plan) que determina as funcionalidades a que tem acesso.

Os utilizadores podem ouvir músicas, acumular pontos, e dependendo do plano que possuem, podem criar playlists, aceder a uma biblioteca pessoal ou reproduzir conteúdos de forma mais personalizada.

Cada utilizador pode também aceder a playlists públicas de outros utilizadores, promovendo partilha e descoberta de músicas.

2.1.6. Classe MusicReproduction

A classe MusicReproduction representa um registo individual de reprodução de uma música por parte de um utilizador. Esta classe foi criada com o objetivo de permitir a análise de hábitos de audição ao longo do tempo, servindo especialmente para responder a queries relacionadas com estatísticas em intervalos temporais.

Cada objeto desta classe associa uma música a uma data específica em que foi reproduzida, permitindo assim construir históricos de reprodução detalhados por utilizador.

Esta estrutura simples, mas eficaz, permite realizar operações como contagens de reproduções por período de tempo, identificação de músicas mais ouvidas num dado intervalo, ou análise da evolução do gosto musical ao longo do tempo.

2.1.7. Classe Playlist

A classe Playlist representa uma coleção de músicas agrupadas sob um determinado nome. Pode ser criada manualmente por um utilizador ou automaticamente pelo sistema — por exemplo, quando o utilizador solicita a geração de uma playlist com músicas de um determinado género musical e uma duração máxima.

As playlists estão agrupadas em dois locais distintos: as privadas, que são exclusivas de cada utilizador e apenas acessíveis por este, e as públicas, criadas pelos utilizadores com o objetivo de serem partilhadas com a comunidade. No caso das playlists públicas, qualquer utilizador pode adicioná-las à sua biblioteca

peçoal, embora apenas o autor original tenha permissões para as modificar. Qualquer alteração feita pelo autor — como adicionar ou remover músicas — será refletida automaticamente nas bibliotecas de todos os utilizadores que tenham essa playlist adicionada, dado que estas foram adicionadas á sua biblioteca através de composição.

A reprodução das playlists está sujeita às permissões do plano de cada utilizador, sendo que apenas alguns planos permitem essa funcionalidade. A reprodução pode ser feita de forma sequencial ou em modo aleatório, consoante a preferência do utilizador.

2.1.8. Classe `PlaylistRandom`

A classe `PlaylistRandom` é uma subclasse da classe `Playlist`, diferenciando-se desta apenas em alguns comportamentos específicos, como o seu construtor, que define automaticamente o autor como “SpotifUM”. Uma instância desta classe é criada unicamente quando um utilizador solicita ao sistema a reprodução de uma playlist aleatória. Assim que a reprodução termina, a playlist é descartada.

Optou-se por implementar esta funcionalidade como uma subclasse para distinguir, a nível estrutural, playlists temporárias geradas pelo sistema daquelas criadas manualmente pelos utilizadores.

2.1.9. Classe `PlaylistFavorites`

De forma análoga à classe `PlaylistRandom`, a classe `PlaylistFavorites` é instanciada apenas quando o utilizador escolhe a opção de reprodução “Feito para você”, que gera uma playlist com base nas suas músicas mais reproduzidas, respeitando preferências como tempo máximo e conteúdo explícito. Também neste caso, o autor é definido como “SpotifUM” e o seu nome como “Feito para Você” e a playlist não é adicionada de forma permanente à biblioteca do utilizador, sendo utilizada apenas para efeitos de reprodução pontual.

Apesar de ambas as funcionalidades poderem ter sido implementadas utilizando apenas instâncias da classe `Playlist` esta abordagem permite uma organização mais explícita e clara das intenções do sistema, facilitando também futuras extensões específicas para este tipo de playlists.

2.2. Facade

No padrão arquitetural da aplicação, a facade tem como objetivo fornecer uma interface única e simplificada para aceder à lógica de negócio. Esse papel é assumido pela classe `SpotifUM`, que centraliza todas as operações e dados do sistema, servindo de ponto de contacto entre o controlador e o modelo.

2.2.1. Classe `SpotifUM`

A classe `SpotifUM` representa o núcleo funcional da aplicação, articulando a comunicação entre as diferentes entidades do sistema, como utilizadores, músicas,

playlists e álbuns. Mantém estruturas de dados otimizadas, utilizando coleções como Map para garantir acesso rápido e eficiente aos dados.

Para além da gestão das entidades, o SpotifUM também é responsável por manter estatísticas de utilização — como o número de reproduções por artista ou género musical — e por responder às interrogações propostas no enunciado.

Adicionalmente, esta classe é responsável por lançar todas as exceções que possam ocorrer durante a execução das operações, deixando a sua gestão e comunicação à interface do utilizador a cargo do Controller. Esta separação assegura uma arquitetura modular e bem organizada, onde a lógica de negócio se encontra isolada da interface.

2.3. Controlador

O *Controller* faz a ligação entre a interface do utilizador (*View*) e a lógica de negócio da aplicação. No SpotifUM, essa responsabilidade é assumida pela classe *Controller*, que interpreta as ações do utilizador, valida os dados recebidos, coordena as operações e gere o tratamento de exceções.

2.3.1. Classe Controller

No projeto SpotifUM, a classe *Controller* desempenha o papel de coordenador central da aplicação, fazendo a ponte entre a interface textual (menus) e a classe *SpotifUM*, que contém a lógica e os dados da aplicação. Sempre que a interface do utilizador solicita uma ação — como registar um novo utilizador, iniciar sessão, criar playlists, reproduzir músicas ou consultar estatísticas — é o *Controller* que interpreta essa ação e delega a sua execução à facade *SpotifUM*.

O *Controller* valida previamente os dados recebidos, garante que as condições de execução são cumpridas, e trata as exceções que possam surgir do núcleo lógico. Desta forma, evita que a interface tenha de lidar com detalhes internos ou erros inesperados, fornecendo-lhe apenas respostas claras, controladas e orientadas para o utilizador final.

Com esta abordagem, o *Controller* garante a separação de responsabilidades, contribuindo para a robustez, testabilidade e extensibilidade da aplicação.

2.4. View

A *View* é responsável por apresentar a interface textual da aplicação ao utilizador, fornecendo os menus disponíveis e recolhendo as interações. Atua como a camada de visualização no padrão MVC, mantendo-se independente da lógica de negócio e do armazenamento dos dados. Toda a interação do utilizador com o sistema é feita através desta componente, que interpreta os comandos e os encaminha ao *Controller* para serem executados.

2.4.1. Classe Menu

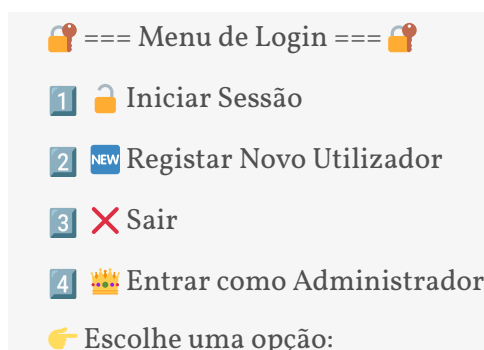
Esta classe abstrata implementa dois métodos abstratos essenciais: `show()`, que exibe as opções disponíveis no menu atual, e `handleInput()`, que lê o input do utilizador, valida os dados introduzidos, executa ações diretamente ou, quando necessário, comunica com o Controller para efetuar operações mais complexas. Este mecanismo permite, por exemplo, alternar entre diferentes menus, interagir com playlists, músicas ou informações de perfil, bem como controlar o estado da aplicação (como o logout ou o encerramento da aplicação).

Para facilitar a navegação, a View conta com um atributo `menuManager`, responsável por manter e atualizar o menu ativo, permitindo uma transição clara entre os diferentes contextos da aplicação. A entrada do utilizador é lida através de um objeto `Scanner`, e a variável `isRunning` determina se a aplicação deve continuar a executar ou encerrar.

De seguida, serão apresentados todos os menus implementados pela aplicação, explicando a utilidade de cada um e ilustrando algumas das suas funcionalidades com exemplos visuais representativos da interface textual. Optou-se por mostrar apenas as funcionalidades que considerámos mais relevantes, com o objetivo de manter a apresentação clara, objetiva e centrada nos aspetos mais significativos da experiência do utilizador. Ainda assim, todas as opções e operações estão disponíveis na aplicação e podem ser exploradas livremente por quem a utilizar.

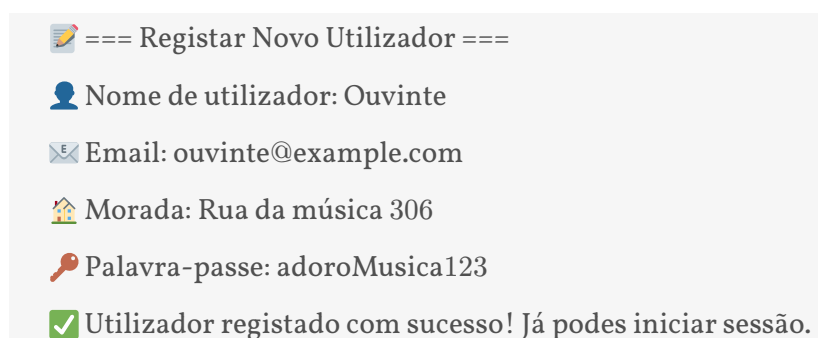
2.4.2. Classe LoginMenu

O menu de login é o ponto de entrada da aplicação e tem como objetivo permitir o acesso dos utilizadores ao sistema, quer sejam utilizadores comuns ou administradores.



Este menu apresenta quatro opções principais:

- **Iniciar Sessão** → Permite um utilizador já registado entrar na sua conta, inserindo o seu nome de utilizador único e a respetiva password.
- **Registar Novo Utilizador** → Inicia o processo de criação de um novo utilizador no sistema, pedindo os respetivos dados, estes utilizadores são registados com o Plano Free.

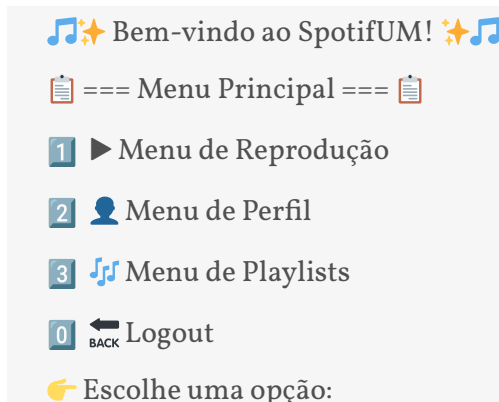


- **Sair** → Fechar a aplicação.
- **Entrar como Administrador** → Acesso ao menu de administrador que irá ser posteriormente demonstrado.

Cada uma destas escolhas conduz o utilizador a menus ou operações distintas, sendo todas devidamente validadas para garantir a segurança e integridade dos dados introduzidos.

2.4.3. Classe MainMenu

Após um utilizador iniciar sessão com sucesso, é-lhe apresentado o menu principal. Este menu fornece um número diferente de opções, dependendo do plano associado ao utilizador.



Como o plano Free não dá acesso a uma biblioteca própria, o utilizador correspondente não dispõe da opção Menu de Playlists. Esta funcionalidade permite a gestão da biblioteca do utilizador e será abordada mais adiante.

O menu principal apresenta, assim, três (no plano Free) ou quatro (nos planos Premium) opções principais:

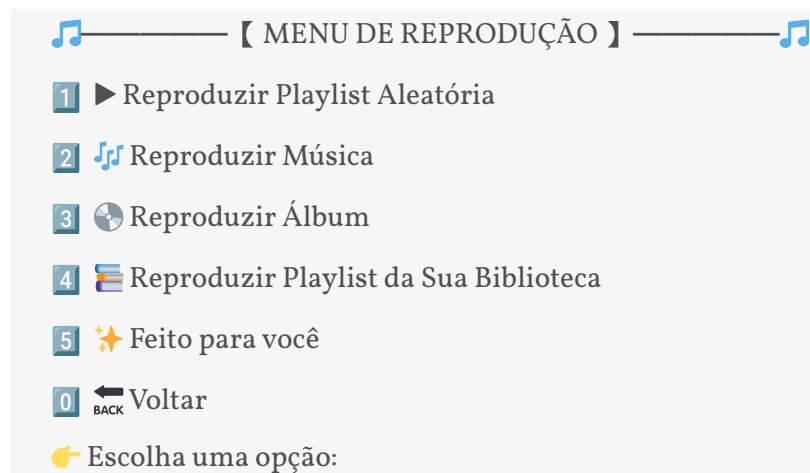
- **Menu de Reprodução** → Permite aceder ao menu de reprodução do sistema, onde o utilizador pode ouvir os conteúdos disponíveis na aplicação.
- **Menu de Perfil** → Dá acesso ao menu de gestão do perfil do utilizador, permitindo visualizar e editar as suas informações pessoais e o seu Plano.

- **Menu de Playlists** → Permite gerir e personalizar as playlists associadas à conta do utilizador.
- **Logout** → Termina a sessão do utilizador e retorna ao menu inicial da aplicação.

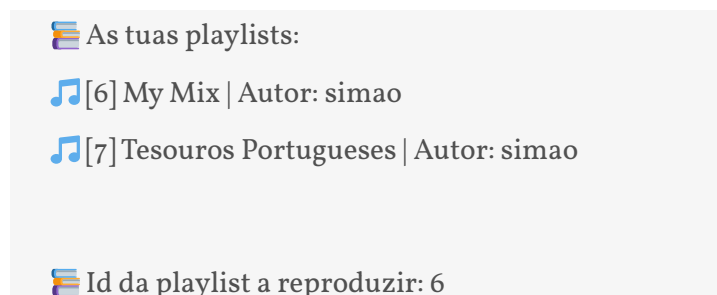
2.4.4. Classe `ReproductionMenu`

Este menu é o menu mais importante da aplicação, pois permite a reprodução de músicas por parte do utilizador.

Tal como o menu principal, as funcionalidades que este menu fornece dependem das permissões do plano do utilizador. Não iremos mostrar o menu para cada plano, mas especificaremos quais o plano necessário para que cada opção seja disponibilizada ao utilizador quando explicarmos as diferentes funcionalidades.



- **Reproduzir Playlist Aleatória** (Utilizadores Free apenas têm acesso a esta funcionalidade) → Reproduz uma das playlists com conteúdo aleatório produzidas pelo sistema.
- **Reproduzir Música** (Utilizadores PremiumBase e PremiumTop) → Permite reproduzir uma música do sistema, através do seu nome.
- **Reproduzir Álbum** (Utilizadores PremiumBase e PremiumTop) → Permite reproduzir um álbum do sistema através do seu nome.
- **Reproduzir Playlist da Sua Biblioteca** (Utilizadores PremiumBase e PremiumTop) → Reproduz uma playlist da biblioteca do utilizador atual à sua escolha.



Deseja reproduzir a playlist em modo Aleatório? (s/n)

s



 Playlist em modo Aleatório.

Deseja reproduzir em modo de Letra? (s/n)

s

 A tocar: "My Mix"

 Comandos disponíveis:

▶ 'f' → próxima |  'b' → anterior |  'q' → sair

 Agora a tocar: Bohemian Rhapsody - Queen

Is this the real life? Is this just fantasy?

Caught in a landslide, no escape from reality

Open your eyes, look up to the skies and see

 Música com conteúdo explícito.

 Agora a tocar: SICKO MODE - Travis Scott

Astro'

Yeah Sun is down, freezing cold

 Fim da reprodução 

- **Feito Para Você** (Utilizadores PremiumTop) → Reproduz uma Playlist baseada nas músicas ouvidas pelo utilizador com um limite de tempo opcional ou 60 minutos por omissão, pode ser também colocado um filtro de apenas músicas explícitas.
- **Voltar** → Permite regressar ao menu principal.

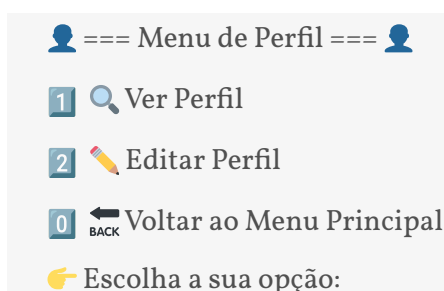
Independentemente do tipo de conteúdo que o utilizador pretende reproduzir, o funcionamento da reprodução de cada música é semelhante. O utilizador fornece ao menu o nome do conteúdo que pretende ouvir. No caso do utilizador pedir qualquer funcionalidade que não seja a playlist aleatória, o menu pergunta se quer ouvir a playlist com uma ordem aleatória, se pretende ouvir em modo letra (que determina se o programa irá apenas imprimir as lyrics na tela ou se coloca o áudio da música a tocar), ou, no caso de o utilizador escolher o "Feito para você" (PlaylistFavoritos) requisitado pelo enunciado, se quer apenas músicas explícitas e se quer um limite de duração (em minutos), esta terá a duração de 60 minutos no caso do utilizador dizer que não pretende introduzir limite de duração.

O utilizador pode durante a reprodução passar a música á frente e retroceder para a música anterior (no caso do plano permitir) ou parar a reprodução e voltar ao menu de reprodução.

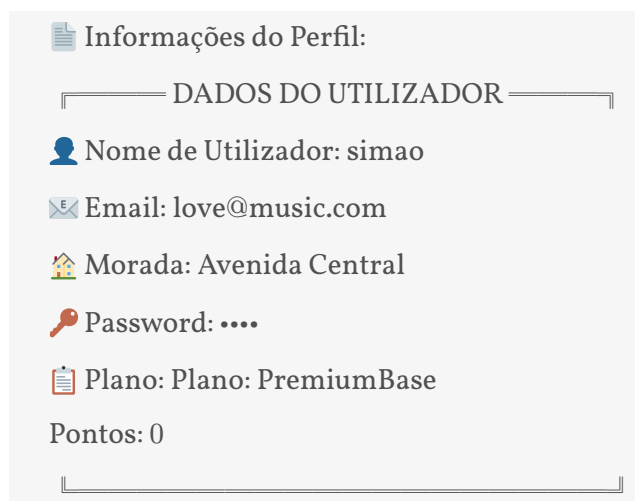
A lógica por trás da reprodução das músicas será explicada posteriormente na secção “Funcionalidades Avançadas”.

2.4.5. Classe ProfileMenu

Este menu é bastante simples, fornecendo funcionalidades relacionadas com a gestão do perfil do utilizador.



- **Ver Perfil** → Imprime todas as informações do utilizador, incluindo nome, email, morada, password, pontos e o seu plano.

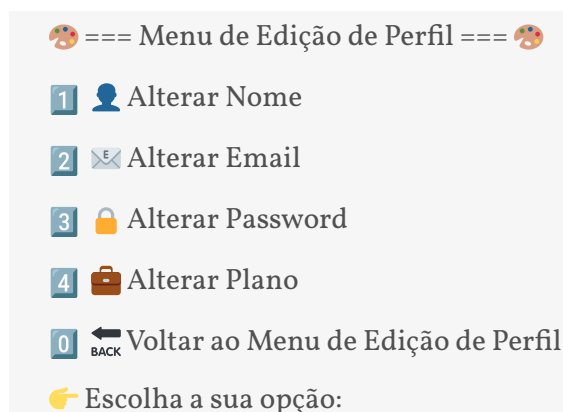


- **Editar Perfil** → Acede ao menu de edição de perfil, onde o utilizador pode modificar informações como nome de utilizador, palavra-passe e o seu plano.
- **Voltar ao Menu Principal** → Auto-explicativo.

Esta classe implementa uma interface simples mas funcional, permitindo aos utilizadores gerirem facilmente as suas informações pessoais, independentemente do tipo de conta que possuem. Todas as opções estão disponíveis para todos os tipos de utilizadores.

2.4.6. Classe ProfileEditorMenu

Este menu permite ao utilizador atualizar as suas informações pessoais e o seu plano de subscrição. Acessível a partir do menu de perfil, oferece várias opções para personalização da conta.



- **Alterar Nome** → Permite ao utilizador modificar o seu nome de utilizador. O sistema verifica se o novo nome escolhido está disponível e não é utilizado por outro utilizador.
- **Alterar Email** → Possibilita a atualização do endereço de email associado à conta. O sistema valida o formato do email e verifica se não está já registado por outra conta.
- **Alterar Password** → Permite ao utilizador definir uma nova palavra-passe para a sua conta. Por questões de segurança, o sistema solicita primeiro a palavra-passe atual antes de aceitar a alteração.
- **Alterar Plano** → Oferece ao utilizador a possibilidade de mudar o seu plano de subscrição. Ao fazer upgrade ou downgrade, o sistema ajusta automaticamente as permissões e funcionalidades disponíveis.
- **Voltar ao Menu de Perfil** → Retorna ao menu anterior (ProfileMenu).

2.4.7. Classe PlanMenu

Este menu permite ao utilizador escolher qual o plano ao qual pretende estar associado, oferecendo diferentes níveis de acesso às funcionalidades da aplicação. Ao selecionar um novo plano, o sistema altera as permissões e o método de ganho de pontos para os do respetivo plano escolhido, mantendo sempre os pontos acumulados no plano anterior.

👛 === Escolha o Seu Plano === 👛

- 1 🎧 Plano Free - Acesso limitado, mas grátis!
 - 2 💎 Plano PremiumBase - Acesso a criação de playlists!
 - 3 👑 Plano PremiumTop - Tudo incluído + benefícios extra! ✅ (Plano Atual)
 - 0 ⬅️ BACK Voltar ao Menu de Perfil
- 👉 Escolha a sua opção:

- **Free** → Plano básico gratuito com funcionalidades limitadas. Permite apenas a reprodução de playlists aleatórias geradas pelo sistema.
- **PremiumBase** → Plano intermédio que oferece acesso a funcionalidades adicionais como reprodução de músicas específicas, álbuns e playlists da biblioteca do utilizador.
- **PremiumTop** → Plano completo que inclui todas as funcionalidades disponíveis na aplicação, incluindo a playlist personalizada “Feito Para Você”.

2.4.8. Classe PlaylistMenu


Este menu gere todas as operações realizadas sobre a biblioteca pessoal do utilizador, permitindo a criação, modificação e gestão de playlists.


🎵🌟 === MENU DE PLAYLISTS === 🌟🎵


- 1 🆕 Menu de Criação de Playlist
 - 2 🌐 Tornar Playlist Pública
 - 3 🎵 Adicionar Música à Playlist
 - 4 🗑️ Remover Música de Playlist
 - 5 📁 Ver Minhas Playlists
 - 6 📥 Adicionar Playlist Pública à Biblioteca
 - 0 ⬅️ BACK Voltar ao Menu Principal
- 👉 Escolhe uma opção:


- **Menu de Criação de Playlists** → Acede ao menu de criação de playlists que será explicado posteriormente. Esta opção permite ao utilizador criar novas playlists personalizadas.
- **Tornar Playlist Pública** → Permite ao utilizador tornar uma das playlists criadas por si pública, possibilitando que outros utilizadores a adicionem à sua própria biblioteca e a reproduzam. Apesar de outros utilizadores poderem aceder à playlist, apenas o criador pode adicionar ou remover músicas da mesma.


- **Adicionar Música à Playlist** → Permite adicionar uma música existente no sistema a uma playlist do utilizador atual. O utilizador pode selecionar qualquer música disponível no catálogo e adicioná-la a uma das suas playlists.

 As tuas playlists:

 [6] My Mix | Autor: simao


 [7] Tesouros Portugueses | Autor: simao


 Id da playlist: 7


 Nome da música: Hino do Futebol Clube do Porto


☒ Música “Hino do Futebol Clube do Porto” adicionada à playlist “Tesouros Portugueses”.


- **Remover Música de Playlist** → Permite remover uma música existente de uma playlist do utilizador atual. Esta funcionalidade só está disponível para playlists criadas pelo próprio utilizador.
- **Ver Minhas Playlists** → Lista todas as playlists da biblioteca do utilizador, incluindo tanto as criadas pelo próprio como as playlists públicas adicionadas de outros utilizadores.
- **Adicionar Playlist Pública à Biblioteca** → Permite ao utilizador adicionar playlists criadas por outros utilizadores à sua biblioteca pessoal. Estas playlists podem ser reproduzidas normalmente, mas não podem ser modificadas.


 Playlists Públicas:


 [0] Top Global | Autor: Spotify


 [1] Rock Legends | Autor: RockFM

 [2] Hip-Hop Nation | Autor: HipHopTV

 [3] Electronic Vibes | Autor: EDM.com

 [4] Throwback Hits | Autor: OldiesFM

 [5] Portuguese Pride | Autor: RTP

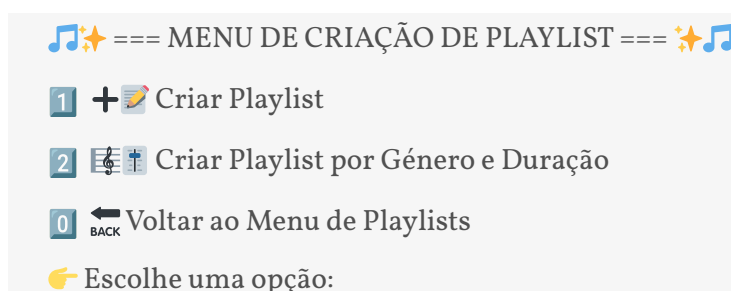
 Id da playlist pública que pretende adicionar: 0

☒ Playlist “Top Global” adicionada à biblioteca do utilizador.

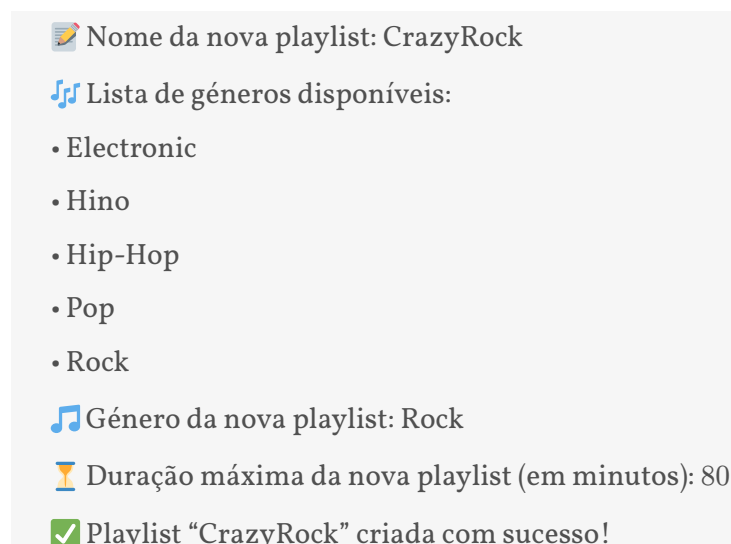
- **Voltar ao Menu Principal** → Permite regressar ao menu principal da aplicação.

2.4.9. Classe `PlaylistCreationMenu`

Este menu é responsável pela criação e adição de novas playlists à biblioteca do utilizador. Está disponível apenas para utilizadores com planos que incluem funcionalidades de biblioteca pessoal.



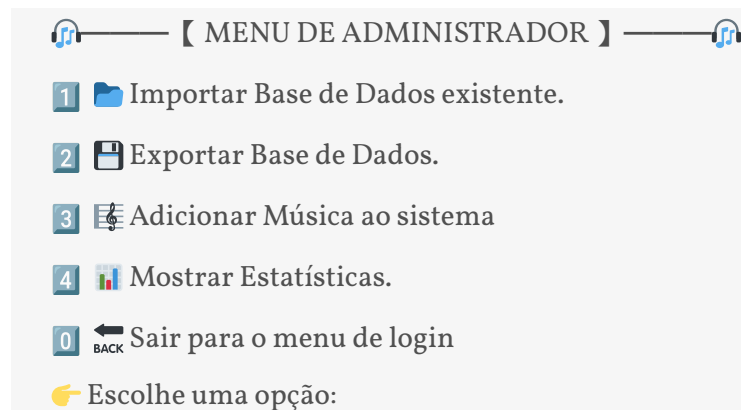
- **Criar Playlist** → Permite ao utilizador criar uma nova playlist com um nome à sua escolha. Esta playlist é então adicionada à biblioteca do utilizador, inicialmente vazia, podendo ser preenchida posteriormente.
- **Criar Playlist por Género e Duração** → Permite ao utilizador criar uma nova playlist, atribuindo-lhe um nome à sua escolha. Esta playlist é automaticamente preenchida com músicas pertencentes ao género selecionado pelo utilizador e cuja duração total não ultrapassa o valor máximo definido. Após criada, a playlist é adicionada à biblioteca pessoal do utilizador.



- **Voltar ao Menu de Playlists** → Regressa ao menu de playlists.

2.4.10. Classe AdminMenu

O menu de administrador é responsável pela gestão global da aplicação SpotifUM. Através deste menu, é possível importar e exportar toda a base de dados do sistema, adicionar novos álbuns e músicas ao repositório geral e aceder às diferentes estatísticas e consultas previstas no enunciado. Este menu é exclusivo para utilizadores com permissões de administrador, oferecendo ferramentas essenciais para a manutenção e supervisão da aplicação.



- **Importar Base de Dados existente** → Permite carregar para a aplicação um ficheiro binário previamente guardado, contendo um objeto serializado do tipo `SpotifUM`. Esta funcionalidade é útil para restaurar o estado completo do sistema, incluindo utilizadores, músicas, álbuns e playlists.
- **Exportar Base de Dados** → Guarda o estado atual da aplicação num ficheiro binário, utilizando serialização.
- **Adicionar Música ao sistema** → Permite ao administrador inserir uma nova música, introduzindo todos os dados relevantes como nome, artista, género, duração, álbum, entre outros. Caso o álbum associado à música ainda não exista no sistema, o administrador é informado de que será criado automaticamente, com a nova música já incluída.

🎵===== Adicionar Nova Música =====🎵

🎵 Nome da nova música: CoolMusic

👤 Nome do artista: CoolArtist

📁 Editora: CoolPublisher

📝 Letra da música: Cool lyrics for testing music creation

🎵 Figuras musicais: Musical Figures

🎵 Género musical: Rock

💿 Nome do álbum: CoolAlbum

⚠️ O álbum não existe.

Deseja criar um novo álbum com esse nome? (s/n): s

✅ Álbum criado com sucesso!

🕒 Duração da música (em segundos): 500

🚫 A sua música contém conteúdo explícito? (s/n): n

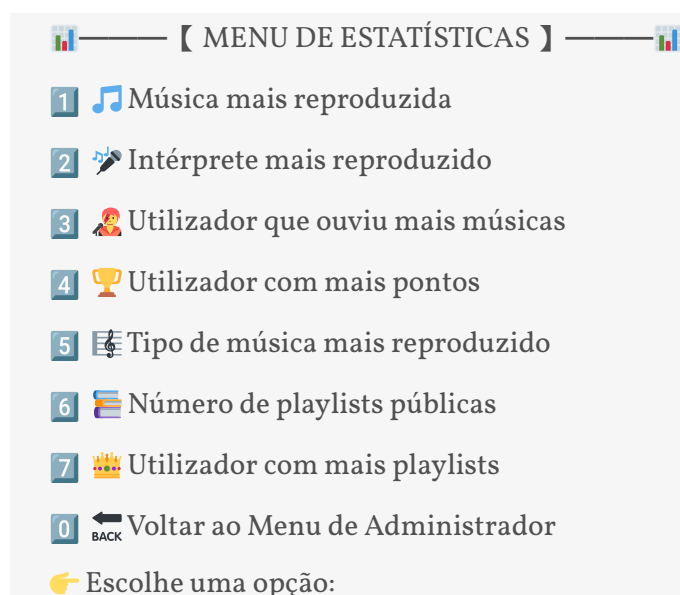
🌐 A sua música possui um URL associado? (s/n): n

✅ Música adicionada com sucesso!

- **Mostar Estatísticas** → Redireciona para o menu de estatísticas e interrogações. Neste submenu, o administrador pode consultar as respostas às queries previstas no enunciado, como artista mais reproduzido, género mais ouvido, utilizador com mais pontos, entre outras.
- **Sair para o menu de login** → Termina a sessão de administrador e retorna ao menu de login da aplicação.

2.4.11. Classe `StatisticsMenu`

O menu de estatísticas permite ao administrador consultar informação detalhada sobre a utilização e o desempenho do sistema SpotifUM. Estas estatísticas são geradas com base nos dados armazenados na aplicação e visam fornecer uma visão global sobre hábitos de utilização, popularidade de conteúdos e envolvimento dos utilizadores. Todas as interrogações apresentadas correspondem às queries previstas no enunciado do projeto.




- **Música mais reproduzida** → Apresenta a música com maior número total de reproduções no sistema, considerando todas as escutas realizadas por qualquer utilizador.
- **Intérprete mais reproduzido** → Identifica o artista ou banda com o maior número de reproduções somadas em todas as suas músicas disponíveis na plataforma.
- **Utilizador que ouviu mais músicas** → Mostra o utilizador com o maior número total de reproduções efetuadas, com o filtro opcional de intervalo de tempo.

Deseja ver as reproduções num certo intervalo de tempo? (s/n) s

Insira o intervalo de tempo inicial no formato DD-MM-YYYY: 03-03-2003

Insira o intervalo de tempo final no formato DD-MM-YYYY: 09-05-2025

 Utilizador que ouviu mais músicas entre 2003-03-03 e 2025-05-09: simao com 24 reproduções.

- **Utilizador com mais pontos** → Indica o utilizador que acumulou mais pontos ao longo do tempo, tendo em conta o tipo de plano e o seu comportamento na plataforma.
- **Tipo de música mais reproduzido** → Revela o género musical com maior número de reproduções globais.
- **Número de playlists públicas** → Informa quantas playlists foram tornadas públicas por utilizadores, estando acessíveis a qualquer pessoa com acesso ao sistema.
- **Utilizador com mais playlists** → Mostra qual o utilizador que criou mais playlists.

- **Voltar ao Menu de Administrador** → Retorna ao menu anterior.

2.5. Utils

As classes utilitárias da aplicação agrupam funcionalidades auxiliares que não pertencem diretamente à lógica de negócio, mas que facilitam a interação do utilizador com o sistema, bem como o funcionamento interno de algumas funcionalidades específicas.

2.5.1. Classe `BrowserOpener`

Classe auxiliar responsável por abrir o URL associado a uma música (caso exista) no browser predefinido do sistema operativo do utilizador. Esta funcionalidade é útil para músicas com conteúdo multimédia externo, como vídeos ou páginas de streaming.

2.5.2. Classe `MusicPlayer`

Classe dedicada à reprodução de áudio. Utiliza a biblioteca `javax.sound.sampled` para carregar ficheiros de som, devolvendo um objeto do tipo `Clip`. Este objeto é posteriormente utilizado no menu de reprodução para controlar a execução da música (início, fim, pausa, etc.).

2.5.3. Classe `Serialization`

Responsável por realizar operações de serialização e desserialização de objetos, nomeadamente da classe `SpotifUM`. Permite guardar o estado da aplicação num ficheiro binário e restaurá-lo posteriormente, assegurando a persistência dos dados entre sessões.

2.6. Excessões

A aplicação `SpotifUM` define um conjunto de exceções personalizadas que têm como objetivo melhorar o controlo de erros e tornar o código mais expressivo, robusto e legível. Estas exceções são utilizadas para sinalizar situações específicas de erro que podem ocorrer durante a execução das operações, permitindo que sejam tratadas de forma clara e apropriada no controlador.

Ao invés de utilizar exceções genéricas, o uso de exceções especializadas ajuda a identificar rapidamente a origem do problema e a tomar decisões mais informadas sobre o que apresentar ao utilizador.

Alguns exemplos de excessões personalizadas são:

- I. **`AlreadyExistsException`** → Lançada quando se tenta adicionar um elemento (por exemplo, uma playlist ou música) que já existe no sistema. Esta exceção previne duplicações e mantém a integridade dos dados.

2. **EmptyPlaylistException** → Utilizada quando se tenta reproduzir ou manipular uma playlist que não contém músicas. Esta exceção evita erros de execução e fornece feedback direto ao utilizador.

3. Funcionalidades Avançadas

Esta secção descreve algumas das funcionalidades mais complexas e técnicas implementadas no projeto SpotifUM. Estas funcionalidades exigem uma lógica mais elaborada, integração com bibliotecas externas ou uma gestão cuidadosa da interação entre camadas da aplicação.

3.1. Reprodução das músicas

Quando um utilizador inicia a reprodução de um conteúdo (por exemplo, uma playlist), a camada de interface (view) solicita ao controlador uma lista com os nomes de todas as músicas associadas ao mesmo.

Com base nas preferências do utilizador, como a escolha entre modo aleatório, reprodução com letra ou reprodução com som, inicia-se uma iteração sobre esta lista de músicas.

A cada iteração, a view invoca o método `playMusic(String musicName)`, o qual retorna um objeto do tipo `MusicInfo`, um DTO que encapsula todas as informações necessárias para a reprodução, nomeadamente:

Atributos da classe `MusicInfo`:

- Nome da música: `string`
- Nome do artista: `string`
- Letra da música: `string`
- Indicação se a música é explícita: `boolean`
- URL associado (caso seja uma música multimédia): `string`
- Mensagem adicional em caso de erro: `string`

Internamente, o método `playMusic` também é responsável por executar todas as atualizações necessárias relacionadas com a reprodução da música. Isto inclui a atribuição de pontos ao utilizador (de acordo com o plano que possui), o incremento do número de reproduções da música, a atualização das estatísticas relativas ao género musical e ao artista em questão, e o registo da reprodução no histórico pessoal do utilizador. Toda esta lógica é encapsulada dentro da camada de negócio, garantindo que os dados do sistema permanecem consistentes e atualizados.

A impressão da letra para o terminal é feito pela view, imprimindo, a cada 500 milissegundo, uma palavra da letra da música, dando a impressão de que está mesmo a tocar.

A reprodução do áudio é tratada pela view, com recurso à classe auxiliar `MusicPlayer`, que utiliza a biblioteca `javax.sound.sampled`. Durante a reprodução,

o utilizador pode interagir com o sistema através de comandos simples que permitem saltar para a próxima música, voltar à anterior ou terminar a reprodução. Estes comandos são processados dinamicamente, enquanto a música está a decorrer, permitindo uma experiência fluida, interativa e contínua.

Este mecanismo tornou-se particularmente desafiante de implementar, pois exigia um equilíbrio entre a reprodução automática e a capacidade de receber comandos do utilizador de forma fluida. O objetivo foi sempre garantir que o utilizador sentisse que está verdadeiramente a “ouvir” música, mesmo num ambiente de interface textual.

3.2. Criação das playlists “Feito para você”

Para a criação das playlists personalizadas ao gosto do utilizador, desenvolvemos um sistema que seleciona as N músicas mais ouvidas pelo mesmo. No caso de ser solicitado pelo utilizador, o sistema pode filtrar para incluir apenas músicas explícitas. Este processo de seleção continua até atingir a duração máxima fornecida ou os 60 minutos definidos em caso de omissão.

Após a seleção, estas músicas são compiladas numa playlist coerente que é reproduzida imediatamente para o utilizador, proporcionando uma experiência musical personalizada baseada nos seus próprios hábitos de escuta.

4. Testes

Os testes da aplicação foram organizados dentro do diretório tests, conforme a convenção do Maven, e foram executados diretamente através do comando `mvn test`. Optámos por realizar testes manuais para verificar o comportamento da aplicação a partir da interface textual, mas também escrevemos testes automatizados para validar funcionalidades críticas e recorrentes.

De forma a acelerar o processo e evitar repetição excessiva de código, utilizámos ferramentas de inteligência artificial para gerar os blocos mais repetitivos dos testes, como setups de dados e asserts de estrutura. Apesar disso, cada teste foi revisto manualmente por nós para garantir a lógica e validade dos mesmos.

Esta abordagem permitiu-nos manter uma boa cobertura funcional sem sacrificar demasiado tempo de desenvolvimento, garantindo que o sistema se comporta corretamente nas situações mais relevantes.

5. Conclusão e trabalho futuro

Gostamos bastante de desenvolver este projeto, tanto pelo tema — que nos foi familiar — como pela oportunidade de trabalhar com Java e aplicar os conceitos de programação orientada a objetos (POO). Ao longo do semestre, fomos percebendo melhor as vantagens deste paradigma, especialmente quando comparado com abordagens mais imperativas. No semestre anterior, já tínhamos desenvolvido um projeto semelhante em termos de temática, mas utilizando uma linguagem que não era orientada a objetos, e isso fez-nos perceber o quanto

a estruturação, a reutilização de código e a separação de responsabilidades são facilitadas com POO.

Inicialmente, tivemos alguma dificuldade em “desligar” do modo de pensar da unidade curricular LI3 (mais focada em baixo nível e performance), mas com o tempo — e com ajuda do professor — percebemos que esta unidade tem objetivos diferentes, mais virados para design e organização de software. A partir daí, conseguimos abraçar melhor a liberdade e o poder que a modelação com objetos nos oferece.

Foi um projeto trabalhoso, especialmente devido à quantidade de código necessário e à preocupação constante com a organização, de forma a não nos perdermos no meio das diferentes funcionalidades. Acreditamos que conseguimos cumprir todos os requisitos do enunciado, implementando-os de forma coerente com o espírito do projeto.

Ao longo do desenvolvimento, surgiram também dúvidas quanto a certas decisões, como por exemplo se deveríamos criar uma subclasse específica para músicas explícitas (`MusicaExplicita`) ou simplesmente usar um campo booleano na classe `Musica`. Optámos pela segunda opção, por simplicidade e eficiência. Embora se possa argumentar que um booleano extra consome mais memória, fizemos uma pequena pesquisa: mesmo que existissem 100 milhões de músicas no sistema, isso representaria apenas 12.5 MB adicionais — algo insignificante nos dias de hoje.

Em termos de trabalho futuro, consideramos implementar uma interface gráfica mais avançada por cima da aplicação atual, talvez durante o verão. Acreditamos que isso tornaria a experiência mais agradável e intuitiva, e seria um desafio interessante para continuar a evoluir a aplicação.