Relatório do Desafio Bridge - Desenvolvedor Mobile

Luiz Roberto Pereira Scolari

Setembro 2024

1 Introdução

Neste relatório, serão documentadas as etapas do desenvolvimento do desafio realizado para a posição de Desenvolvedor Mobile no Laboratório Bridge.

Todas as tarefas propostas e executadas serão abordadas detalhadamente, de modo a evidenciar as soluções desenvolvidas. Serão discutidos os métodos aplicados, as dificuldades encontradas ao longo do processo e as formas adotadas para resolver os problemas propostos.

2 Tarefas Propostas

Neste tópico, serão abordadas as atualizações e soluções das tarefas propostas realizadas no aplicativo.

2.1 Primeira Tarefa: Os Tech Leads tiraram férias! Atualize o Endereço de E-mail no Endpoints.dart

A primeira tarefa consistia em atualizar o endereço de e-mail no arquivo Endpoints.dart, utilizado para configurar os endpoints da aplicação.

Inicialmente, identifiquei o local no código onde o e-mail estava configurado. Após localizar a variável que continha o endereço de e-mail, realizei a atualização conforme solicitado, garantindo que o novo endereço fosse implementado corretamente.

```
3       static const _email = 'luizrpscolari@gmail.com';
```

Figure 1: Endereço de e-mail atualizado

2.2 Segunda Tarefa: Mudança no Ar! Atualize o Título da Tela "Home"

A segunda tarefa consistia em atualizar o título da tela "Home", a tela principal do aplicativo.

Inicialmente, identifiquei o local do código onde estava armazenada a variável do título e fiz a alteração solicitada, de "Lançamentos" para "Fresh Hits".

Figure 2: Título atualizado

2.3 Terceira Tarefa: O Botão de Voltar Precisa de um Estilo!

A terceira tarefa consistia em estilizar o botão "Voltar" presente na interface da aplicação. O botão original possuía uma cor azul e precisava ser atualizado para seguir o padrão visual do aplicativo.

Para resolver esse problema, identifiquei o componente responsável pelo botão de retorno, que estava dentro da classe `BackButton`. A tarefa envolveu alterar o código para aplicar um estilo customizado ao botão, utilizando o widget `ElevatedButton.icon`.

Figure 3: Botão atualizado

2.4 Quarta Tarefa: Hora de Organizar o "Global Top 50" – As Músicas com Preview Primeiro!

Na quarta tarefa, o objetivo era reorganizar a lista de músicas exibidas na aba "Global Top 50", de modo que as músicas com o *previewUrl* (prévia disponível) fossem exibidas primeiro, seguidas pelas músicas que não possuem essa prévia. Essa mudança tem como finalidade melhorar a experiência do usuário, destacando as músicas que oferecem uma prévia de áudio diretamente na interface.

Para implementar essa alteração, o código foi modificado da seguinte maneira:

Primeiro, foi realizada a filtragem das músicas com e sem *previewUrl*. Para isso, utilizei o método `where()` para separar as duas categorias: tracksWithPreviewUrl: músicas que possuem uma *previewUrl* válida. tracksWithoutPreviewUrl: músicas que não possuem uma *previewUrl*. Em seguida, combinei as

duas listas de modo que as músicas com *previewUrl* fossem exibidas primeiro. Isso foi feito criando uma nova lista chamada `sortedTracks`, que coloca os itens de `tracksWithPreviewUrl` no topo e os de `tracksWithoutPreviewUrl` logo abaixo. Por fim, no `ListView`, alterei o laço que percorre as músicas, utilizando `sortedTracks` em vez de `top`, garantindo que a nova ordem fosse refletida na interface.

2.5 Quinta Tarefa: Botão de Favoritar Inútil?! Dê Vida a Ele!

O objetivo desta tarefa foi adicionar a funcionalidade ao botão de "favoritar" na interface, permitindo ao usuário marcar ou desmarcar um item como favorito. Antes da modificação, o botão não possuía qualquer ação implementada, tornando-o inútil na interface do usuário

Alterações Realizadas:

Adicionar/remover favoritos: No código original, ao pressionar o botão, nenhuma ação era executada. Adicionei uma lógica condicional que verifica se o item já está marcado como favorito. Caso esteja, ele é removido da lista de favoritos; caso contrário, é adicionado. Isso foi implementado utilizando o método `remove` e `add` em uma lista de favoritos.

Atualização do ícone de coração: Além de tornar o botão funcional, foi implementada uma troca visual para o ícone. Dependendo se o item está ou não marcado como favorito, o ícone de coração muda entre preenchido (`heart filled.png`) e contornado (`heart outline.png`). Essa troca de ícones reflete instantaneamente a ação do usuário, proporcionando feedback visual.

Figure 4: Botão de favoritos atualizado

2.6 Sexta Tarefa: Organize Seus Favoritos – Ordem Alfabética é a Nova Onda!

Nesta tarefa, o código foi alterado para organizar a lista de músicas favoritas em ordem alfabética. A função `buildFavoritesList` foi modificada para incluir a seguinte lógica:

Criação de Cópia da Lista: Foi criada uma nova lista `sortedFavorites` a partir da lista `favorites` usando `List.from()`. Isso garante que a lista original permaneça inalterada.

Ordenação da Lista: A nova lista foi ordenada utilizando o método `sort`, que aplica uma função de comparação baseada na propriedade `name` de cada objeto `Track`. A função `compareTo` foi utilizada para garantir a ordem lexicográfica correta.

Essas alterações melhoraram a usabilidade da aplicação, permitindo que os usuários visualizem suas músicas favoritas organizadas de forma mais acessível, facilitando a navegação e a localização de faixas específicas.

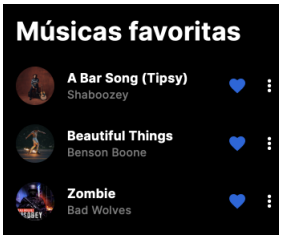

Figure 5: Ordem de músicas favoritas atualizada

2.7 Sétima Tarefa: Seguidores em Alta! Adicione a Quantidade de Seguidores para Cada Artista

Nesta tarefa, foi adicionado um novo atributo para representar a quantidade de seguidores de cada artista na classe `FullArtist`. As seguintes alterações foram realizadas:

Novo Atributo: A classe `FullArtist` foi estendida com um novo atributo `followers` do tipo `int`. Isso permite armazenar e manipular a contagem de seguidores de um artista.

Modificação do Construtor: O construtor da classe `FullArtist` foi atualizado para incluir o novo parâmetro `followers`. Isso assegura que ao criar uma instância de `FullArtist`, a quantidade de seguidores seja fornecida.

Exibição da Quantidade de Seguidores: No arquivo (`artist screen.dart`), a quantidade de seguidores é exibida utilizando a interpolação de strings. O código foi modificado para mostrar o número real de seguidores do artista, substituindo o valor fixo.

Figure 6: Quantidade de seguidores atualizada

2.8 Oitava Tarefa: Barra de Busca Desalinhada – Vamos Alinhar Tudo!

Nesta tarefa, o código da barra de busca foi ajustado para garantir que todos os elementos fossem devidamente alinhados, melhorando a experiência do usuário. As seguintes alterações foram realizadas:

Padding da Barra de Busca: O `contentPadding` foi configurado com EdgeInsets.symmetric(vertical: 12). Isso assegura que haja espaço adequado acima e abaixo do conteúdo, proporcionando um alinhamento mais equilibrado e uma aparência mais organizada.

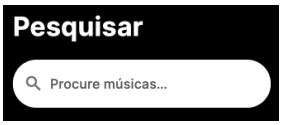

Figure 7: Texto de busca alinhado

2.9 Nona Tarefa: Nova Tela de Álbum – Hora de Brilhar com o Protótipo!

Nesta tarefa, a tela do álbum foi implementada para proporcionar uma experiência mais rica e interativa ao usuário. As seguintes mudanças foram implementadas:

Carregamento do Álbum: A função `loadAlbum()` foi implementada para buscar as informações do álbum com base no `albumId` fornecido. O álbum é carregado automaticamente quando a tela é inicializada, e o estado de carregamento é controlado por uma variável booleana `loading`. Durante o processo de carregamento, um indicador visual (`CircularProgressIndicator`) é exibido para fornecer feedback ao usuário até que as informações do álbum sejam completamente exibidas.

Layout Implementado: O layout da tela foi implementado utilizando o widget `ImageGradientBackground` para exibir a imagem do álbum com um efeito de gradiente, proporcionando um visual mais atrativo. O nome do álbum, o nome do artista, a quantidade de músicas e o botão Tocar são exibidos de forma clara e organizada.

Lista de Faixas Implementada: As faixas do álbum são exibidas em uma lista, utilizando `ListTile` para cada música, com a imagem do álbum, o nome da música e o nome dos artistas. Os botões de favoritar e menu de opções foram integrados para cada faixa, tornando a interação mais fácil e intuitiva.

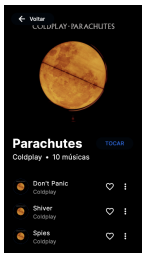

Figure 8: Texto de busca alinhado

3 Dificuldades Enfrentadas

Apesar de possuir uma base na em Kotlin, que me ajudou a entender conceitos de desenvolvimento mobile e orientação a objetos, foi um grande desafio realizar a Tarefa 9 com Flutter. O Flutter utiliza uma abordagem baseada em widgets, o que demanda uma forma diferente para a construção de interfaces. Embora eu tenha conseguido aplicar parte do meu conhecimento em Kotlin, principalmente no que diz respeito à estruturação de código e organização de componentes, a construção do layout da tela foi, sem dúvida, a parte mais desafiadora.

A tela de álbum, responsável por exibir informações como o nome do álbum, artista, lista de faixas, além de permitir a reprodução das músicas, exigiu um grande esforço na organização dos elementos visuais. Um dos maiores desafios foi a combinação de diferentes widgets, como `Row`, `Column`, `ListTile`, além de lidar com o espaçamento correto entre os componentes. Garantir que a interface estivesse visualmente agradável e funcional foi algo que demandou diversas tentativas e ajustes.

Outro ponto de dificuldade foi a manipulação de imagens de fundo, utilizando o widget `ImageGradientBackground`, de forma que elas se ajustassem corretamente ao restante da interface, sem comprometer a legibilidade dos textos e a harmonia visual da tela. A sobreposição dos textos com as imagens demandou um ajuste preciso nos gradientes e no alinhamento dos elementos, garantindo que o título do álbum e o nome do artista permanecessem sempre legíveis e bem destacados.

O gerenciamento de listas, especialmente na exibição das faixas do álbum, também foi um desafio. A utilização do `ListView` para listar as faixas, garantindo que cada faixa tivesse seu botão de favorito e menu de opções, foi uma parte trabalhosa, já que cada elemento precisava ser tratado individualmente, e o layout deveria permanecer consistente independentemente da quantidade de faixas.

Em suma, a pouca familiaridade com o Flutter tornou a construção do layout uma tarefa complexa, exigindo várias tentativas e ajustes. Porém, meu conhecimento prévio em desenvolvimento mobile com Kotlin me ajudou a entender os conceitos necessários e encontrar soluções ao longo do processo.

4 Pontos de melhoria

Apesar dos avanços feitos na construção da tela do álbum, reconheço que existem áreas em que poderia ter realizado um trabalho mais eficaz. Uma das principais dificuldades foi a disposição dos elementos na interface. Embora tenha conseguido implementar o layout, acredito que poderia ter explorado melhor as propriedades de espaçamento e alinhamento.

Além disso, seria benéfico implementar uma navegação mais fluida entre as telas, garantindo que a transição entre diferentes seções do aplicativo seja mais suave. A introdução de animações sutis durante essas transições poderia melhorar a experiência do usuário, tornando-a mais agradável e interativa.