INSTITUTO POLITÉCNICO DE COIMBRA

ISCAC | COIMBRA BUSINESS SCHOOL


# MusicSpot

## Trabalho de Base de Dados

2º ANO DE LICENCIATURA DE INFORMÁTICA DE GESTÃO

**PEDRO DAVID DE JESUS ALMEIDA (2018069272)**

**BRUNO MIRANDA (2018067857)**

**Junho | 2020**

Coimbra

# Introdução

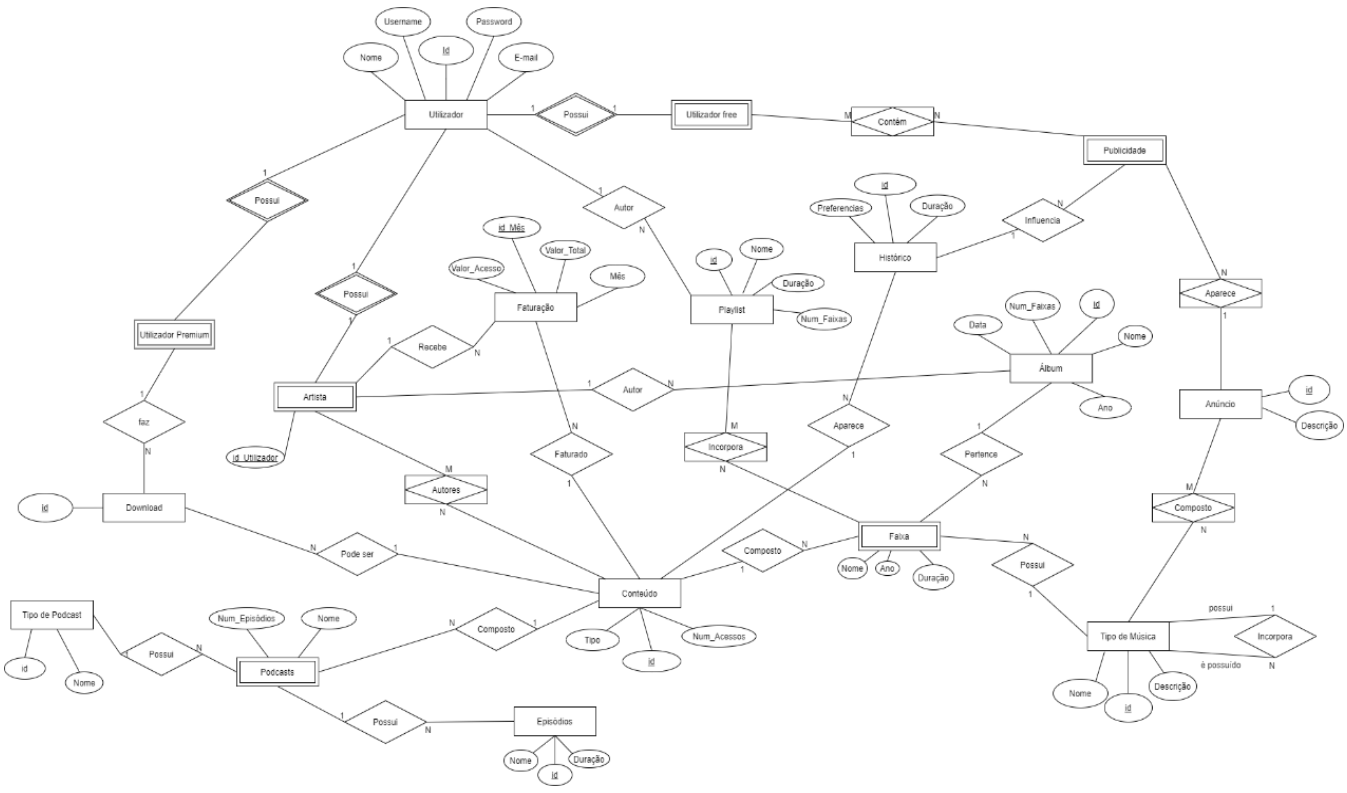
No âmbito da avaliação contínua à disciplina de Base de Dados, foi proposto o desenvolvimento de uma aplicação. Antes da elaboração do projeto proposto, tivemos de escolher um tema perante as opções dadas pelos professores. Depois de todos os temas analisados, chegamos a um consenso e optamos pelo tema 2, escolhendo "Gestão de serviço de streaming de música com publicidade dirigida", que mais tarde daria o nome á nossa aplicação de "MusicSpot". Este projeto foi desenvolvido com o objetivo de criar uma plataforma simples, funcional e útil, de forma a que os potenciais utilizadores ouçam a sua música sem qualquer tipo de problema.

Para a realização deste projeto, utilizamos o Lucichart para a realização do Diagrama Entidade-Relação da base de dados da aplicação, o labder para a realização do Diagrama Relacional, MySQL para a base de dados da app e utilizamos o Balsamiq Wireframes para desenvolvermos as mockups da aplicação, Microsoft PowerApps para desenvolvimento de interface de input e output, todas elas plataformas gratuitas, seguras e de fácil utilização, possuindo diversas ferramentas apelativas e bastante úteis, nas quais através da sua utilização, podemos criar o nosso modelo de aplicação como tínhamos idealizado.

# Diagrama Entidade-Relação

Um diagrama entidade relação é um tipo de fluxograma que ilustra "entidades", por exemplo, pessoas, objetos ou conceitos, relacionando-se entre si dentro de um sistema. Diagramas entidade relação são os mais usados para projetar e ceder bancos de dados relacionais nas áreas de engenharia de software, sistemas de informação empresarias, na educação e na pesquisa. Tal como referimos na introdução para a realização deste diagrama utilizamos o Lucidchart pois foi o que nos pareceu o melhor entre vários para a realização deste.

Na seguinte figura podemos ver o diagrama entidade-relação da base de dados da nossa aplicação.

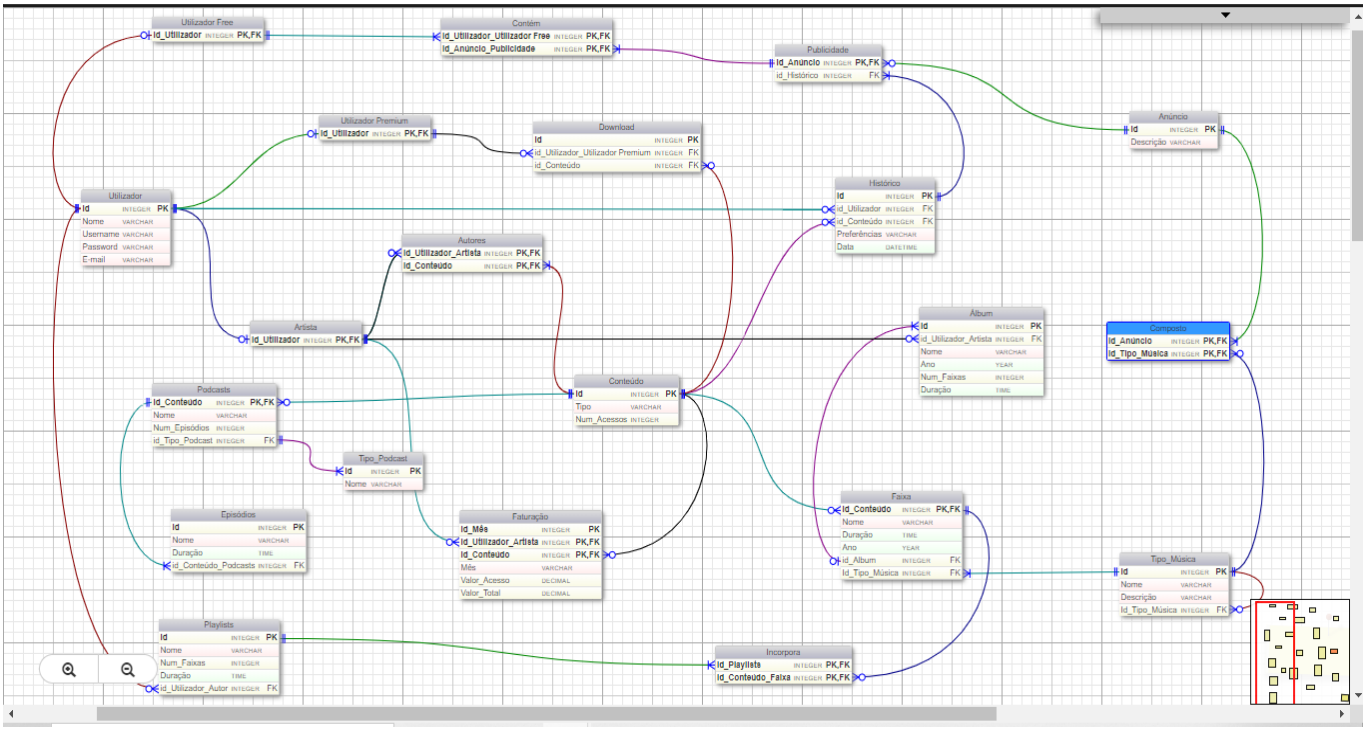


# Diagrama Relacional

Um diagrama relacional molda os dados num conjunto de relações (tabelas) que são constituídas por um conjunto de atributos (coluna ou campos) que definem características relevantes da entidade (conceito, objeto) que representam.

Para a realização deste diagrama recorremos à ferramenta DER – Livre na plataforma labder.

Com isto segue a seguinte figura que apresenta o diagrama relacional da base de dados da nossa aplicação.



# Decisões Tomadas

Começando pelo utilizador, optámos por fazer especializações através de ligações 1:1 na medida em que um utilizador apenas poderá ter um registo em determinada subscrição, quer seja free, premium ou artista. Definimos a chave estrangeira oriunda do utilizador, Id_Utilizador, para chave primária das especializações.

Optámos por uma entidade conteúdo com duas especializações que vão ser o miolo da nossa aplicação, faixas e podcasts. Para tal optámos por relações 1:N, onde estas especializações recebiam como chave estrangeira a chave primária de conteúdo, sendo esta as suas próprias chaves primárias.

Entre Faixa e Tipo_Música um relacionamento de 1:N na medida em que uma faixa apenas possui um tipo de musica mas para cada tipo de música podem existir muitas faixas. O mesmo acontece entre as entidades podcasts e Tipo_Podcasts.

Na entidade Tipo_Música implementámos uma relação recursiva pois um estilo musical pode ser composto por vários, por exemplo Funk Rock.

Entre Faixa e Álbum um relacionamento, 1:N, pois uma faixa pertence apenas a um álbum mas este último pode conter uma ou muitas faixas. Entre Podcasts e Episódios um relacionamento nos mesmos moldes, uma vez que um podcast pode possuir um ou vários episódios, enquanto que este último apenas pertence a único podcast.

Para a criação de playlists criámos uma terceira entidade (entidade associativa), Incorpora, que se relaciona através de relações do tipo 1:N com as entidades playlists e faixas. Desta forma é possível que uma playlist seja composta por várias faixas e que uma faixa incorpore várias playlists. Entidade associativa herdou (como atributo) de cada uma das entidades iniciais as suas chaves primárias, sendo a chave primária da entidade associativa formada pela concatenação destas.

No que toca às playlists, criámos uma relação 1:N com utilizadores de forma a registar o autor das playlists. Uma playlist é criada apenas por um utilizador mas, um utilizador pode criar várias playlists.

Para registar os autores dos conteúdos, foi necessário ocorrer a uma entidade associativa de nome “Autores”, na medida em que um determinado conteúdo pode ter como criador vários artistas e, um artista pode criar vários conteúdos.

No que toca à faturação dos conteúdos aos artistas, feito através de uma entidade associativa "Faturação" que herdou, como atributo, das entidades Artista e Conteúdo as suas chaves primárias. A chave primária da entidade associativa é formada pela concatenação dos atributos das chaves herdadas e por um atributo adicional Id_Mês. Desta forma é possível fazer registo todos os meses da mesma faixa para o mesmo artista.

O histórico foi onde tivemos mais dificuldades na medida em que várias faixas/podcasts o podem incorporar e, a mesma faixa/podcast pode ocorrer várias vezes nesse mesmo histórico, levando a uma relação M:N. Contudo era fulcral um histórico ter o id_Conteudo e com a utilização de uma entidade associativa isso não iria acontecer.

Primeiramente, tínhamos que um utilizador só podia ter um histórico, e que era obrigado a ter histórico, mesmo que tenha acabado de criar a conta e que ainda não tenha ouvido nenhuma música, o que obviamente estava errado.

A nossa solução foi essa mesma entidade associativa ser o histórico, fazendo ligação com as entidades Utilizador e Conteúdo através das tão conhecidas relações do tipo 1:N.

Para chave primária de histórico criámos um novo atributo do tipo inteiro. Assim é possível o mesmo conteúdo ser registado no histórico de determinado utilizador quantas vezes este o reproduzir.

Histórico está relacionado, também, com a entidade Publicidade. Esta relação permitirá a plataforma escolher a publicidade a enviar nas contas free. É aqui que entra o atributo Preferências da entidade Histórico. Este atributo foi adicionado para registar o estilo musical mais ouvido pelo utilizador através de uma filtragem no histórico, de forma a facilitar a seleção dos anúncios a entrar na publicidade.

Por sua vez, a entidade Publicidade possui uma relação 1:N com a entidade Anúncio, de onde vai herdar a chave primária como chave estrangeira, sendo esta a sua própria chave primária, ou seja, Publicidade é uma entidade fraca sendo dependente para identificação de Anúncio, a entidade proprietária.

Possui também uma relação M:N com a entidade Utilizador Free através da entidade associativa Contém, que se relaciona com as duas entidades iniciais através de relações 1:N. Isto pois a uma conta free pode ser dirigida várias publicidades, e a mesma publicidade pode ser enviada a vários utilizadores.

Anúncio está relacionada com a entidade Tipo_Música através de uma terceira entidade (associativa) “Composto”. Isto porque um estilo musical pode possuir vários anúncios dedicados a si e, um anúncio, pode abranger vários estilos musicais e não apenas um.

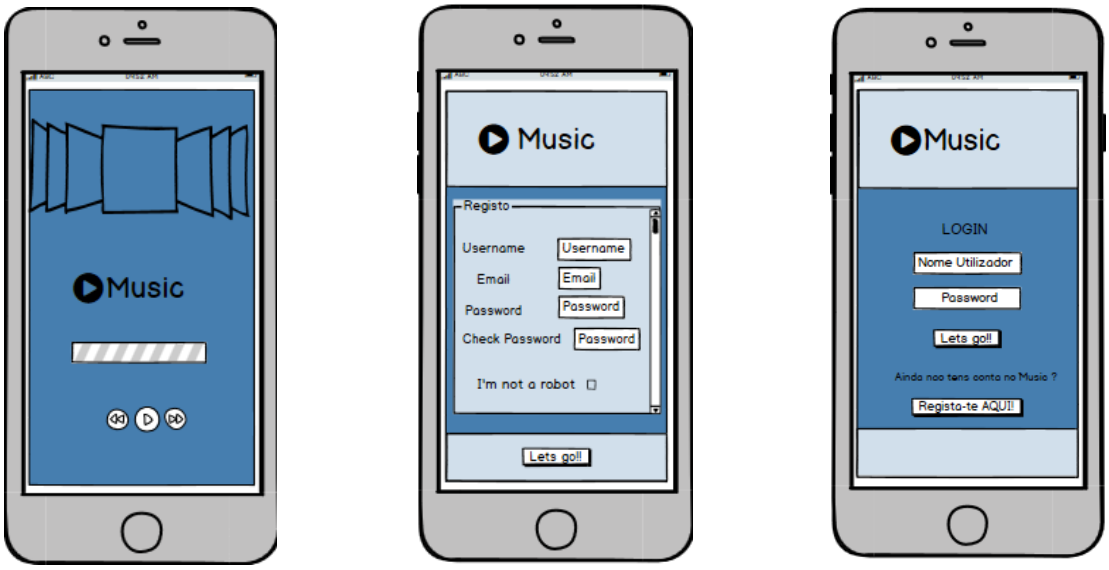
A forma de implementação do Download vai de encontro à do histórico. Uma entidade associativa que se relaciona com as entidades Utilizador Premium e Conteúdo através de relações do tipo 1:N.

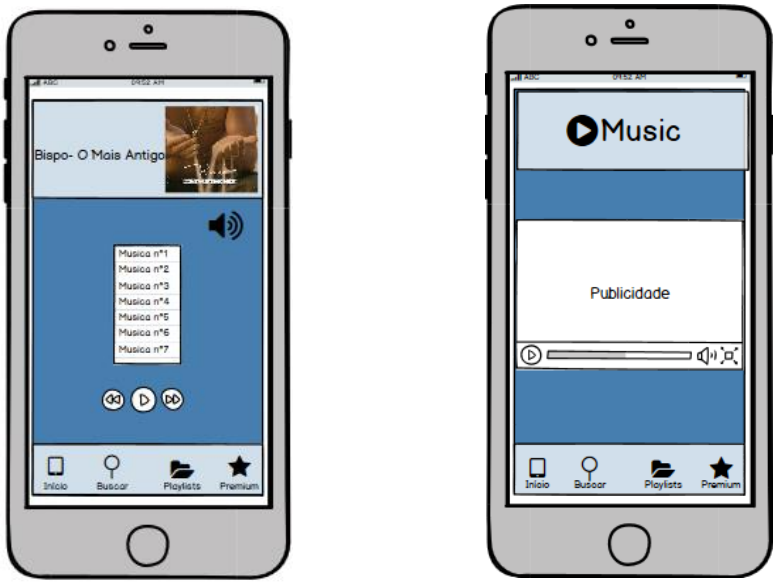
A chave primária de Download é um atributo do tipo inteiro adicional, além daqueles herdados (chaves primárias das entidades Utilizador Premium e Conteúdo), de forma a determinado utilizador conseguir fazer o download de um especifico conteúdo as vezes que pretender.

# Mockups

São um modelo em escala ou de tamanho real e um projeto ou dispositivo, usado para o ensino, demonstração, avaliação de design, promoção e outros propósitos neste caso foi usado para demonstração e avaliação de design. Uma mockup é um protótipo que fornece pelo menos parte da funcionalidade de um sistema e permite o teste de um projeto.

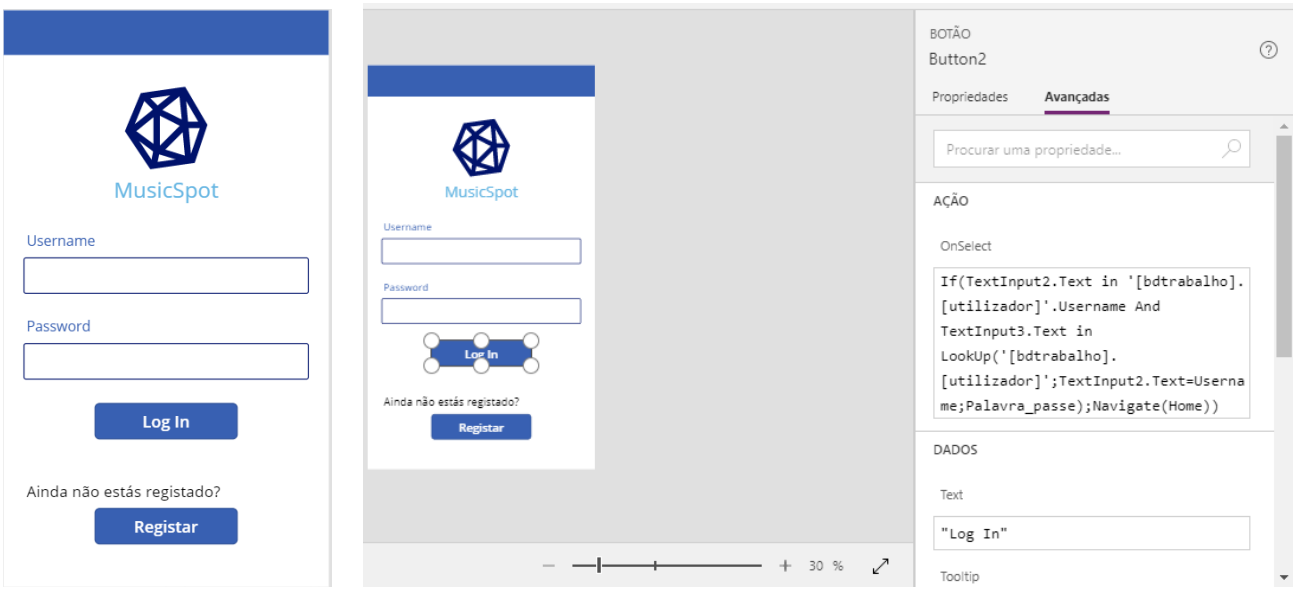
Com isto segue a seguinte figura que apresenta as diversas mockups da nossa aplicação.



# PowerApps

## Ecrã LogIn

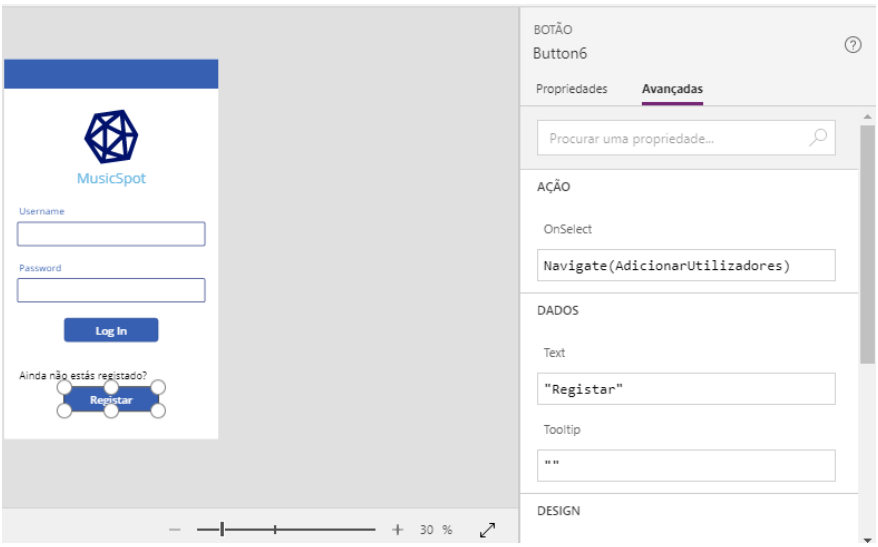



Ecrã destinado ao inicio de sessão pelos utilizadores na aplicação com duas entradas de texto, para registo do username e palavra-passe, respetivamente.

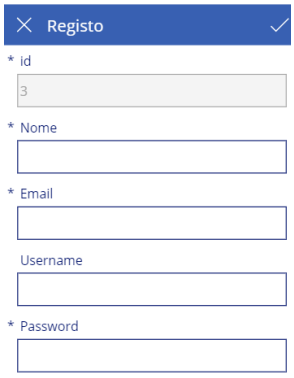
Através do Botão "Log In" fazemos a validação do utilizador pois este apenas avança na aplicação se tiver registado na base de dados, ou seja, se a condição seguinte se verificar: If(TextInput2.Text in '[bdtrabalho].[utilizador]'.Username And TextInput3.Text in LookUp('[bdtrabalho].[utilizador]';TextInput2.Text=Username;Palavra_passe); Navigate(Home)).

Primeiramente, a condição exigida era "If(TextInput2.Text in '[bdtrabalho].[utilizador]'.Username And TextInput3.Text in '[bdtrabalho].[utilizador]'.Palavra_passe;Navigate(Home))" mas apercebemo-nos que assim, a palavra-passe apenas tinha de estar registada na base de dados sem a necessidade de estar associada ao utilizador inserido, o que não é o pretendido.

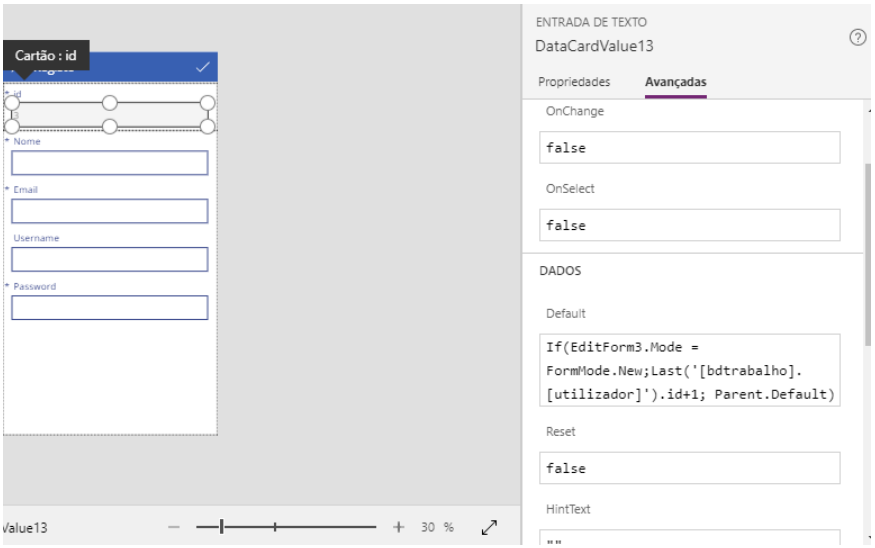
Caso o usuário da aplicação ainda não estar registado na mesma, implementámos o botão "Registar" com a indicação NewForm(EditForm3);; Navigate(AdicionarUtilizadores) para fazer a navegação para o ecrã próprio de registo, onde vai poder criar uma conta, no estado New quando se pretende criar um registo novo.




## Ecrã AdicionarUtilizadores



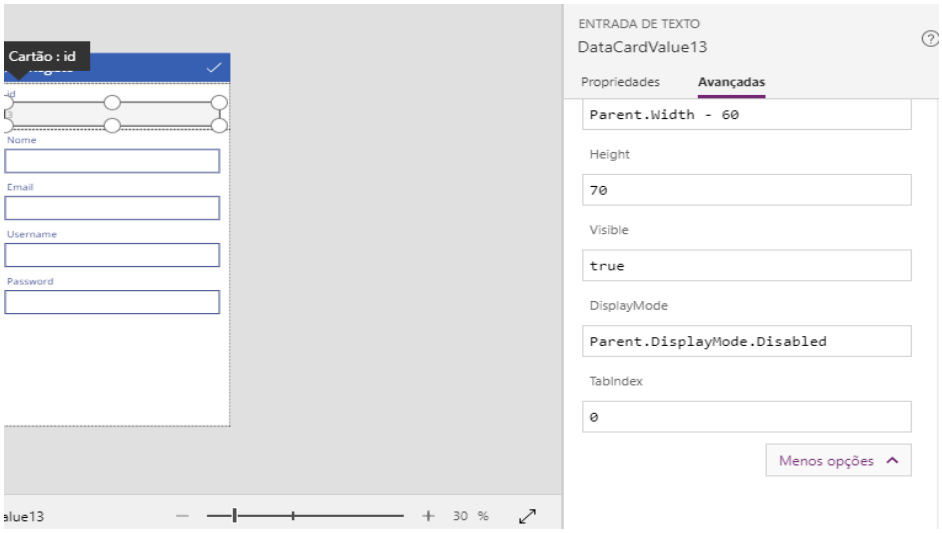

Este formulário está com origem de dados na tabela Utilizador ('[bdtrabalho].[utilizador]') pois é nessa entidade que estão contidos os registos que queremos apresentar. Encontra-se no estado New, isto é, na vista de criar registo novo.



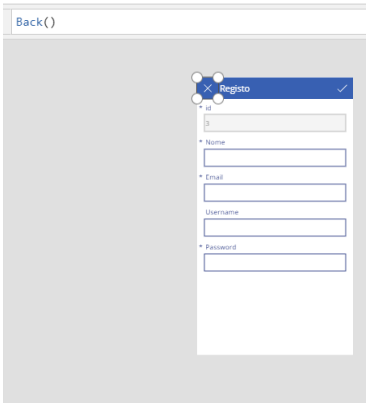
No campo “id”, para gestão de ID’s únicos (auto-increment), implementámos a seguinte condição: If(EditForm3.Mode =

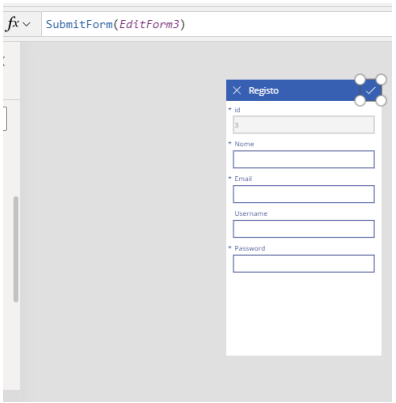
FormMode.New;Last('[bdtrabalho].[utilizador]').id+1; Parent.Default).

Esta o que faz é procurar o último ID existente na coleção de dados através da utilização da função Last. Caso o tipo de formulário for diferente de “New”, utilizar o valor por default.




Alterámos o modo de apresentação do campo para “Parent.DisplayMode.Disabled” pois assim não são permitidas alterações ao campo.



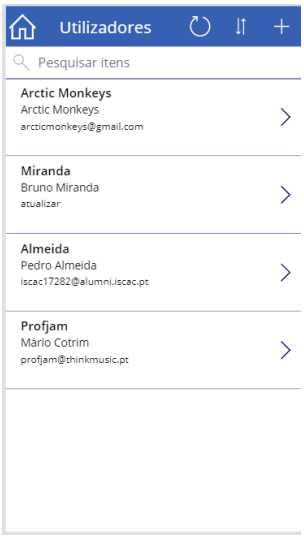

Por fim, no ícone “Cancelar” está atribuída a função “Back()”, de forma a quando este ícone for selecionado a aplicação navege para o ecrã anterior, e no ícone “Aceitar” a função “SubmitForm(EditForm3)” para fazer a submissão do

formulário e, consequentemente, criar o novo registo na base de dados, caso os campos estejam todos em conformidade.

## Ecrã Home


Ecrã com três botões: “Utilizador”, “Faixas”, “Playlists”. Servem para fazer a navegação para os respetivos ecrãs através da função “Navigate()”.

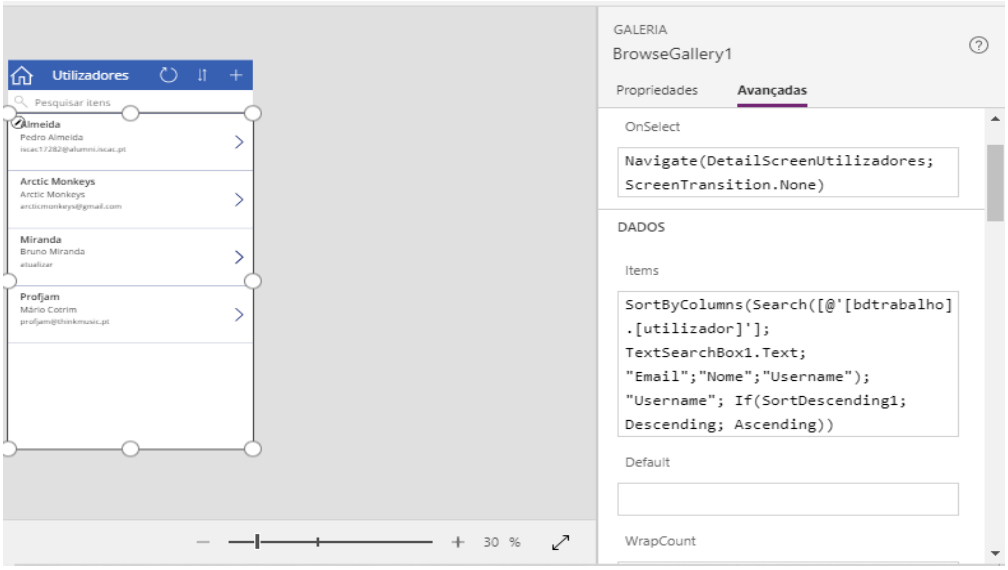
## Ecrã Utilizadores




Com origens de dados em ‘[bdtrabalho].[utilizador]', pois é nesta tabela que estão os dados a apresentar.

O ícone Home, implementado com a função Navigate(Home), premite ao usuário navegar para o ecrã Home.

O ícone Adicionar, implementado com NewForm(EditForm3);;Navigate(AdicionarUtilizadores), permite o registo de novos utilizadores pelo processo anteriormente falado.



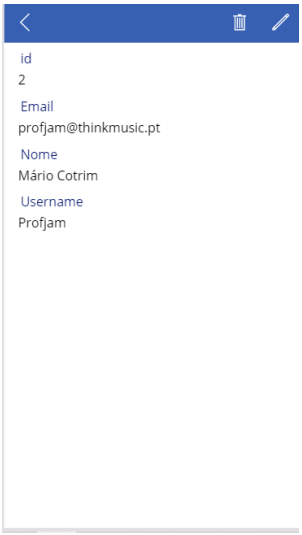

Items: SortByColumns(Search([@'[bdtrabalho].[utilizador]']; TextSearchBox1.Text; "Email";"Nome";"Username"); "Email"; If(SortDescending1; Descending; Ascending))

Utiliza o texto TextSearchBox para procurar combinações no campo email, nome e username, através da função Search.

Utiliza a função SortByColumns para ordenar por “Username”, descending ou ascending, dependendo da valor do controlo.

OnSelect: Função Navigate(DetailScreenUtilizadores; ScreenTransition.None) indica que ao carregar em um dos items da lista (BrowseGallery1), que o ecrã DetailScreenUtilizadores irá abrir.

## Ecrã DetailScreenUtilizadores

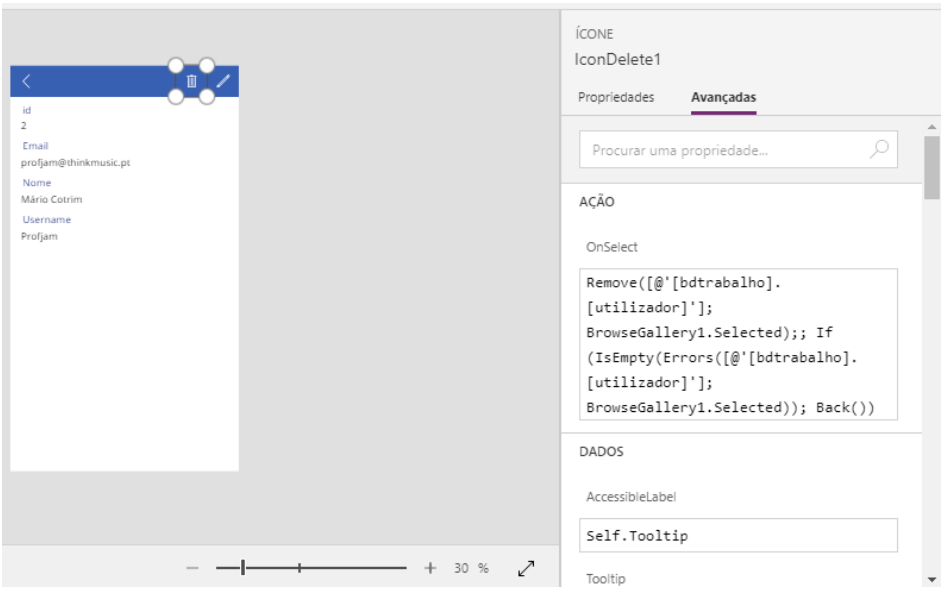



Origem de dados na tabela '[bdtrabalho].[utilizador]'.

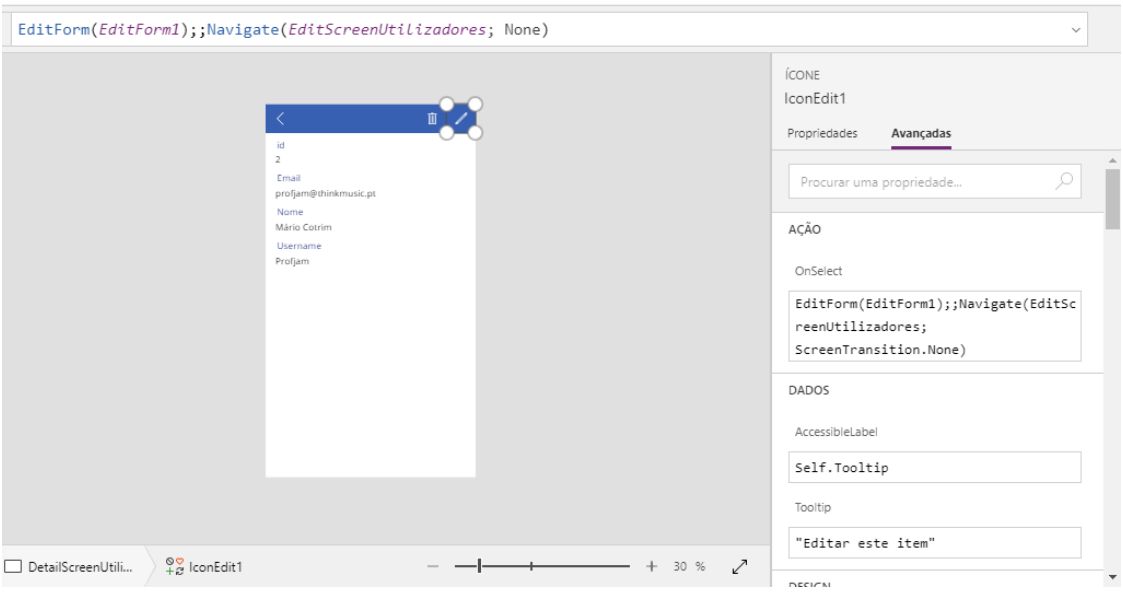
O formulário encontra-se no estado View, na vista de detalhe de um registo existente;

O registo a ser apresentado é selecionado através da propriedade BrowseGallery1.Selected, ou seja, apresentar o registo selecionado na lista de utilizadores no ecrã Utilizadores.

Um ícone de seta esquerda, implementado com a função Back(), para o utilizador poder voltar atrás na aplicação.

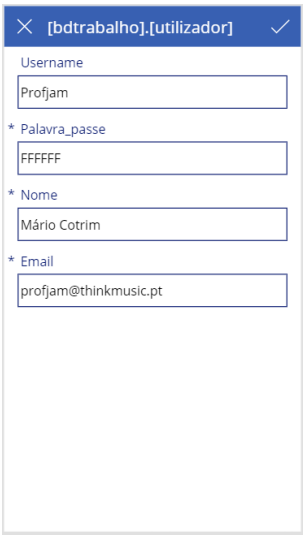



Remover registo através do icon "trash" que chama o seguinte código: Remove([@'[bdtrabalho].[utilizador]']; BrowseGallery1.Selected);; If (IsEmpty(Errors([@'[bdtrabalho].[utilizador]']; BrowseGallery1.Selected)); Back()). Remove o item selecionado no browsegallery1, depois de validados os erros. Retorna ao ecrã anterior, se for bem sucedido.




No ícone "Editar" está atribuída a função EditForm(EditForm1);;Navigate(EditScreenUtilizadores; ScreenTransition.None) de forma a quando este ícone for selecionado a aplicação navege para o formulário EditForm1 no estado Edit, isto é, na vista de edição do registo selecionado.

## Ecrã EditScreenUtilizadores

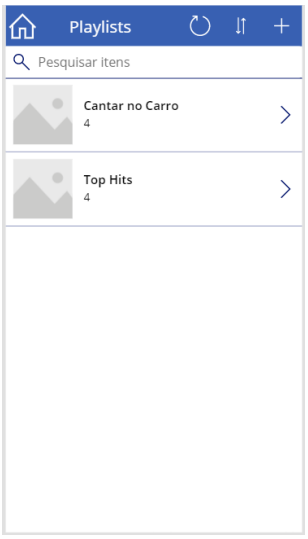



Formulário no estado Edit, onde se pode alterar os valores dos campos apresentados do registo selecionado.

Ícone Aceitar com a função SubmitForm(EditForm1), para validar e submeter a alteração no registo.

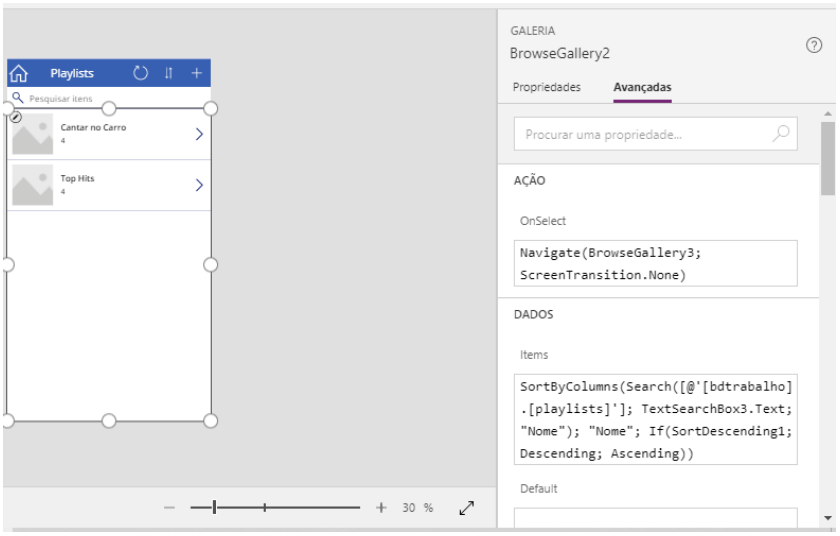
Ícone Cancelar com a função ResetForm(EditForm1);;Back() para voltar atrás na aplicação reiniciando o formulário EditForm1.

## Ecrã Playlists




Com origens de dados em '[bdtrabalho].[playlists]', pois é nesta tabela que estão os dados a apresentar.

O ícone Home, implementado com a função Navigate(Home), premite ao usuário navegar para o ecrã Home.

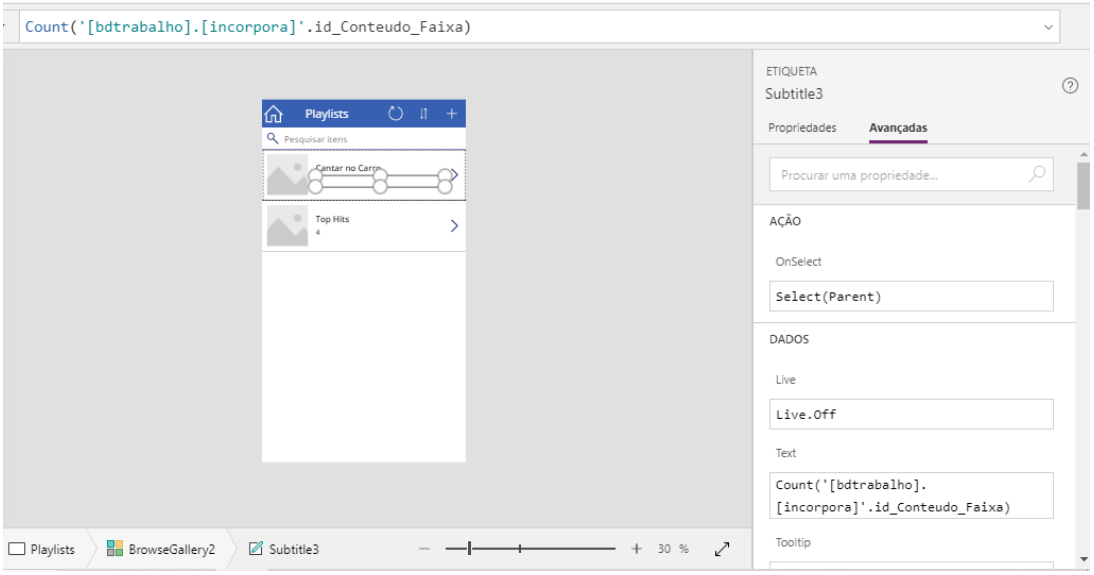



Items: SortByColumns(Search([@'[bdtrabalho].[playlists]']; TextSearchBox3.Text; "Nome"); "Nome"; If(SortDescending1; Descending; Ascending))

Utiliza o texto TextSearchBox3 para procurar combinações no campo nome através da função Search.

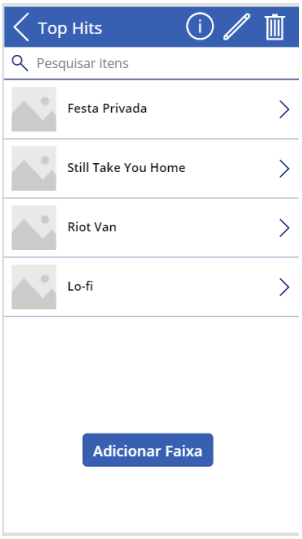
Utiliza a função SortByColumns para ordenar por "Nome", descending ou ascending, dependendo da valor do controlo.

OnSelect: Função Navigate(BrowseGallery3; ScreenTransition.None) indica que ao carregar em um dos items da lista (BrowseGallery2), irá fazer a navegação para BrowseGallery3.



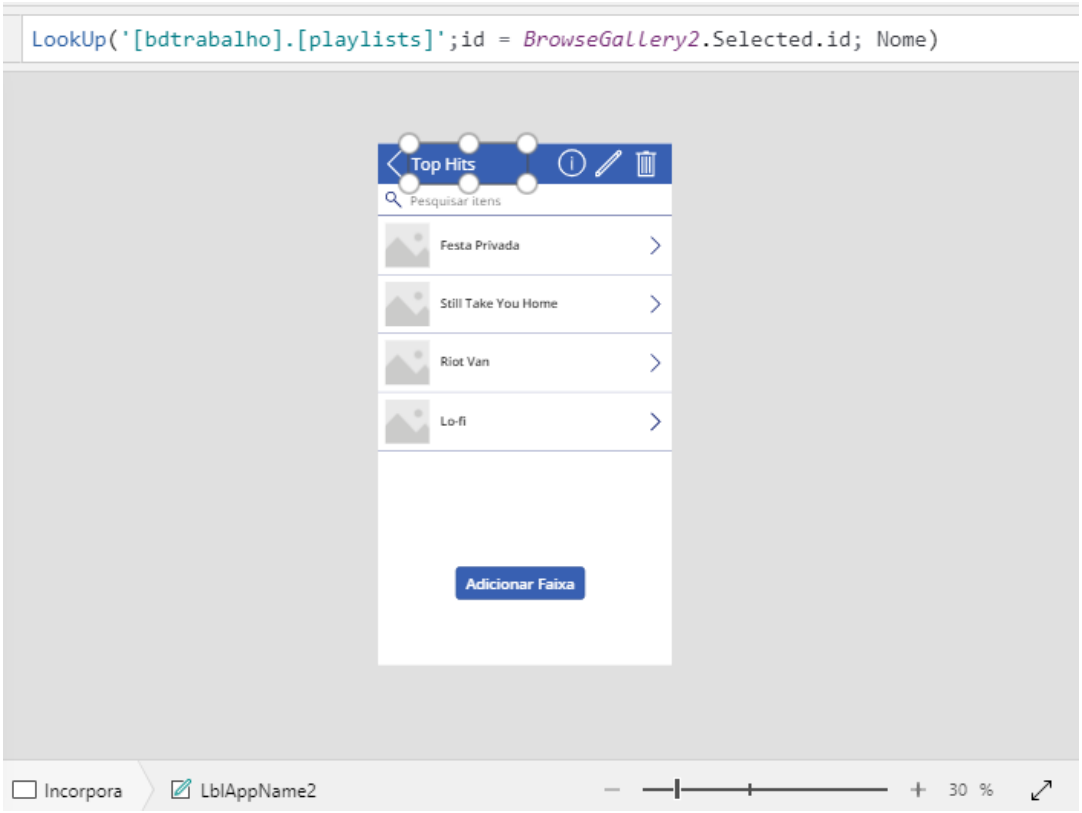

Subtítulo da lista faz referência ao número de faixas presentes na playlist e é obtido através do comando Count('[bdtrabalho].[incorpora]'.id_Conteudo_Faixa), que faz respetivamente a contagem de identificadores das faixas e devolve o total.

## Ecrã Incorpora




Com origens de dados em '[bdtrabalho].[incorpora]', pois é nesta tabela que estão os dados a apresentar.

Um ícone de seta esquerda, implementado com a função Back(), para o utilizador poder voltar atrás na aplicação.




Implementámos uma etiqueta com a função LookUp('[bdtrabalho].[playlists]';id = BrowseGallery2.Selected.id; Nome) de forma a apresentar o nome da playlist selecionada.

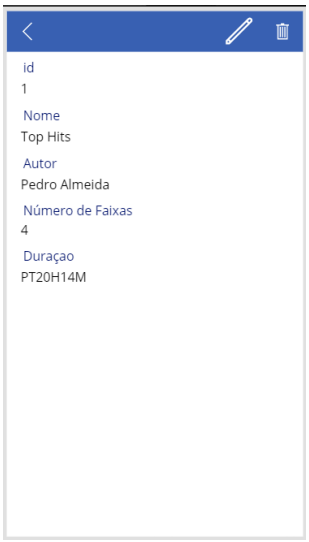
O ícone de informação está programado de forma a quando selecionado fazer a navegação para o ecrã dos detalhes da playlist - Navigate(DetailScreenPlaylists).

O ícone editar serve para fazer a navegação para o ecrã RemoverFaixa caso o utilizador pretenda remover alguma faixa da sua playlist. Está implementado com a função Navigate(RemoverFaixa).

Remover registo da playlist através do icon “trash” que chama o seguinte código: Remove([@'[bdtrabalho].[playlists]']; BrowseGallery2.Selected);; If (IsEmpty(Errors([@'[bdtrabalho].[playlists]']; BrowseGallery2.Selected)); Back()). Remove o item selecionado no browsegallery2 (lista de playlists), depois de validados os erros. Retorna ao ecrã anterior, se for bem sucedido.

O botão Adicionar Faixa faz a ligação com o ecrã onde vai possibilitar ao usuário adicionar faixas à playlist - Navigate(AdicionarFaixas).

## Ecrã DetailScreenPlaylists



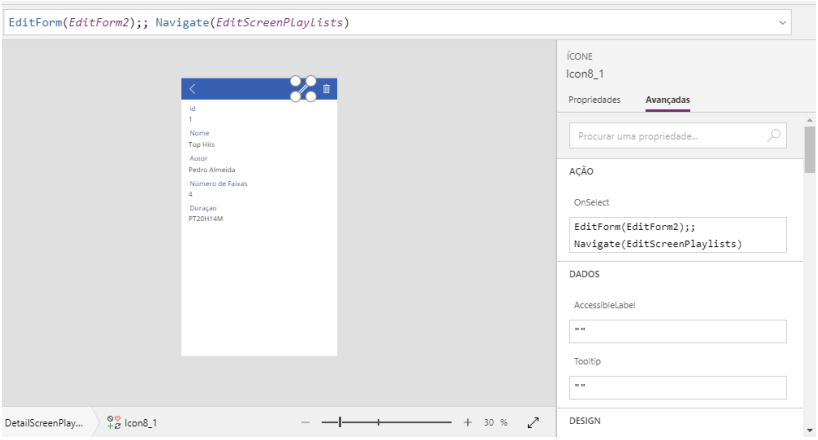

Origem de dados na tabela '[bdtrabalho].[playlists]'.

O formulário encontra-se no estado View, na vista de detalhe de um registo existente.

Um ícone de seta esquerda, implementado com a função Back(), para o utilizador poder voltar atrás na aplicação.

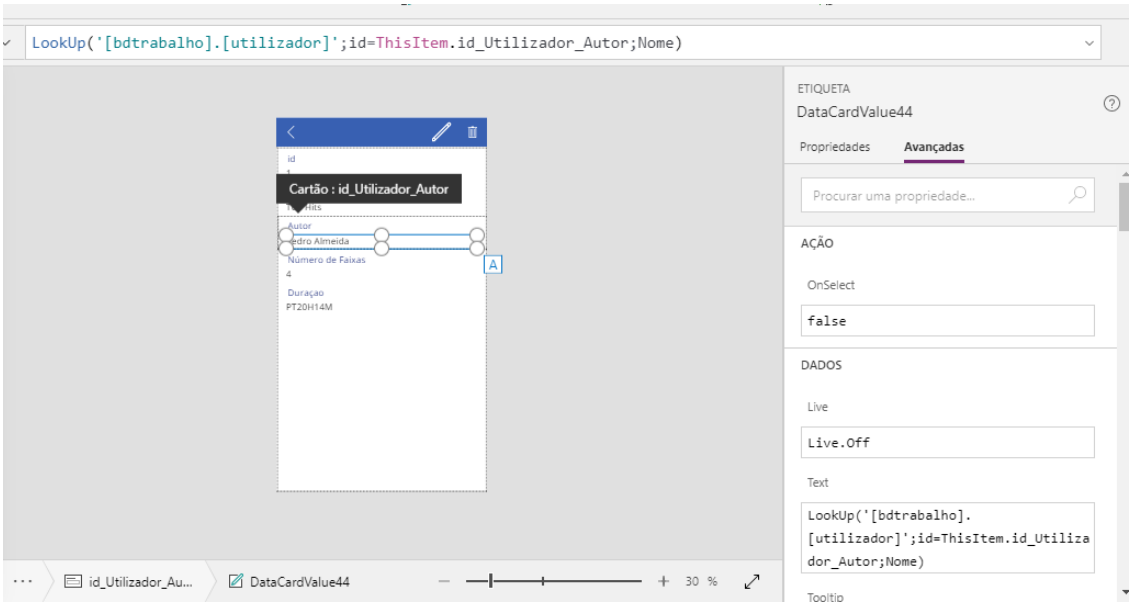
O registo a ser apresentado é selecionado através da propriedade BrowseGallery2.Selected, ou seja,

apresentar o registo selecionado na lista de playlists no ecrã Playlists.




No ícone "Editar" está atribuída a função EditForm(EditForm2);;Navigate(EditScreenPlaylists) de forma a quando este ícone for selecionado a aplicação navege para o formulário EditForm2 no estado Edit, isto é, na vista de edição do registo selecionado.

Remover registo através do icon "trash" que chama o seguinte código: Remove([@'[bdtrabalho].[playlists]']; BrowseGallery2.Selected);; If (IsEmpty(Errors([@'[bdtrabalho].[playlists]']; BrowseGallery2.Selected)); Back()). Remove o item selecionado no browsegallery2, depois de validados os erros. Retorna ao ecrã anterior, se for bem sucedido.




Para aparecer o nome do autor e não apenas o seu id, recorremos à função LookUp() - LookUp('[bdtrabalho].[utilizador]';id=ThisItem.id_Utilizador_Autor;Nome). A partir da tabela utilizador, a função vai verificar a condição "id=ThisItem.id_Utilizador_Autor" para as linhas na origem de dados e fornece o resultado da linha que corresponde à condição especificada com o seu respetivo campo Nome.

Um ícone de seta esquerda, implementado com a função Back(), para o utilizador poder voltar atrás na aplicação.

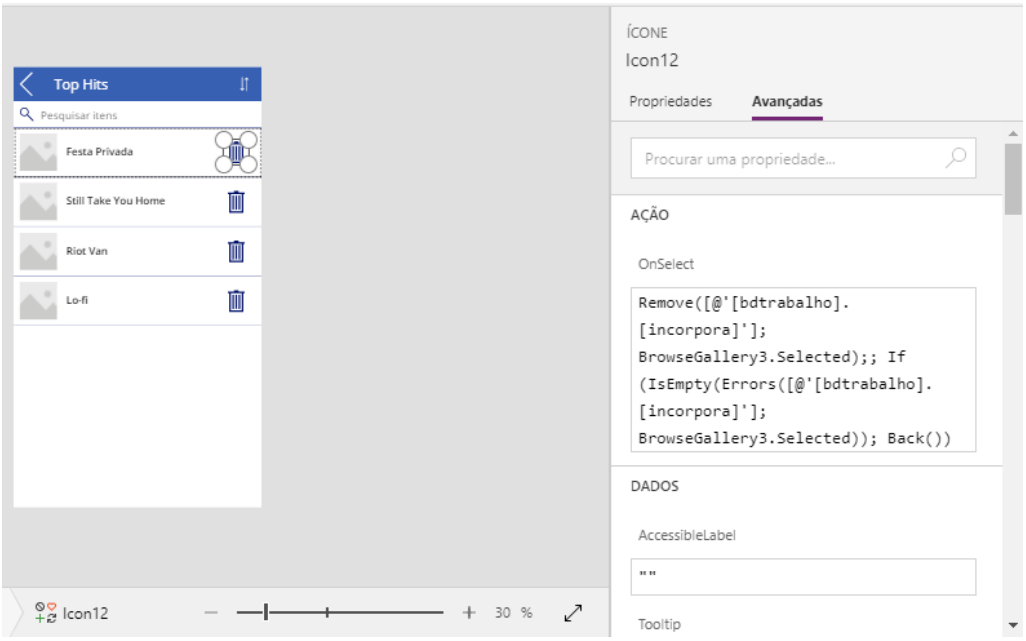
## Ecrã EditScreenPlaylists


Formulário no estado Edit, onde se pode alterar os valores dos campos apresentados do registo selecionado. Só faz sentido aparecer os campos Nome e Autor para alteração pois os restantes ou são obrigatoriamente fixos (id) ou o seu valor é atribuído automaticamente (Num_Faixas e Duração).

Ícone Aceitar com a função SubmitForm(EditForm2), para validar e submeter a alteração no registo.

Ícone Cancelar com a função ResetForm(EditForm2);;Back() para voltar atrás na aplicação reiniciando o formulário EditForm2.

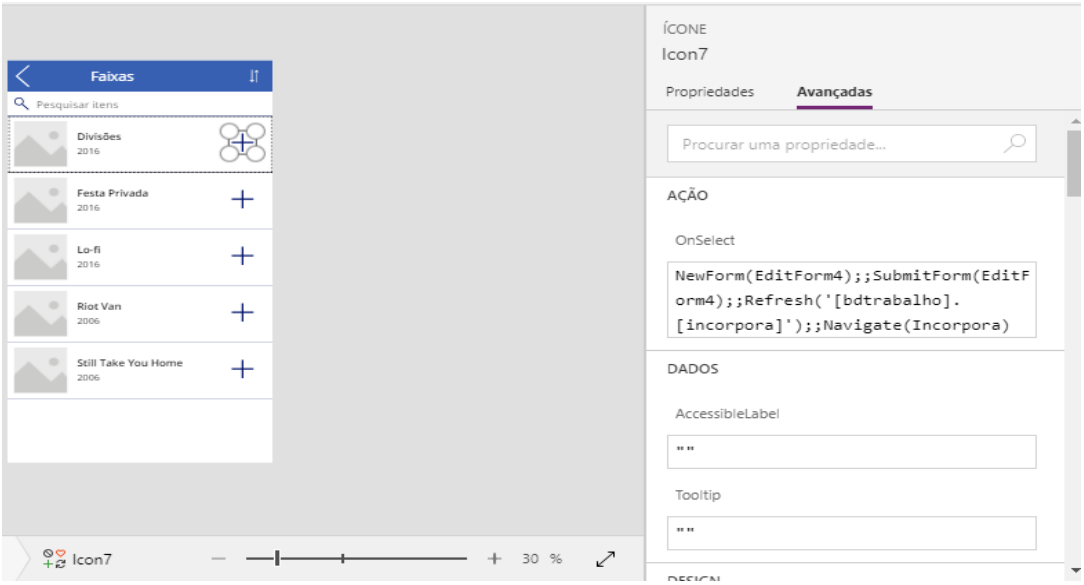
## Ecrã RemoverFaixa




Remover registo de faixa na playlist através do icon “trash” que chama o seguinte código: Remove([@'[bdtrabalho].[incorpora]'];
BrowseGallery3.Selected);; If (IsEmpty(Errors([@'[bdtrabalho].[incorpora]'];
BrowseGallery3.Selected)); Back()). Remove o item selecionado no browsegallery3, depois de validados os erros. Retorna ao ecrã anterior, se for bem sucedido.

## Ecrã EditAdicionarFaixas e ecrã AdicionarFaixas

- Ecrã AdicionarFaixas




Esta lista (BrowseGallery5) está com origem de dados na tabela Faixa ('[bdtrabalho].[faixa]') pois é nessa entidade que estão contidos os registos de faixas que queremos apresentar.

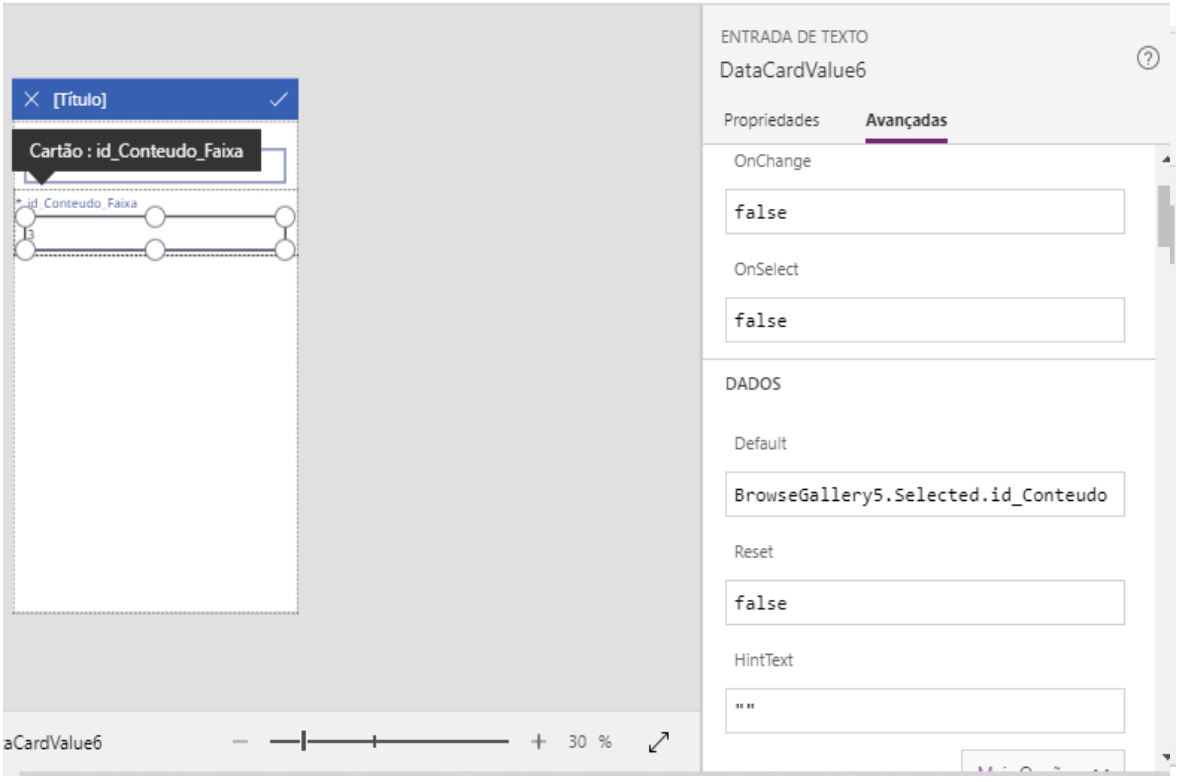
O ícone Adicionar com as funções NewForm(EditForm4);;SubmitForm(EditForm4);;Refresh('[bdtrabalho].[incorpora]');;Navigate(Incorpora) permite validar e adicionar o registo da faixa selecionada à playlist, isto é, criar uma nova form, EditForm4, com um formulário do tipo new mas sem necessidade de redirecionar o utilizador para o ecrã EditAdicionarFaixas, pois os valores necessários vão ser inseridos automaticamente. Assim, adicionámos a função SubmitForm(EditForm4) diretamente neste ícone.

Um ícone de seta esquerda, implementado com a função Back(), para o utilizador poder voltar atrás na aplicação.

- Ecrã EditAdicionarFaixas


Este ecrã foi criada com o objetivo de fazer, apenas, suporte ao ecrã AdicionarFaixas, ou seja, nenhum usuário irá ter contacto com este ecrã sendo que, é a partir dele que novas faixas são adicionadas ás playlists.

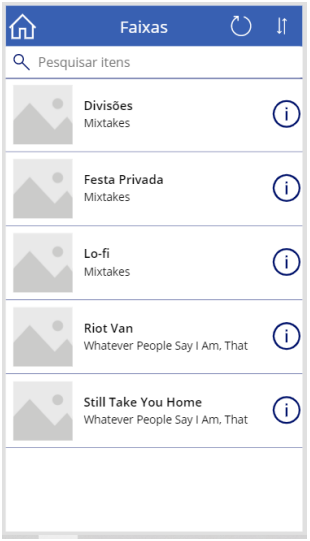
Este formulário está designado como EditForm4 e encontra-se no estado New, isto é, na vista de criar registo novo.




Os valores das entradas de texto são atribúidos automaticamente: id_Playlist é preenchido com o valor da playlist selecionada no ecrã Playlists (BrowseGallery2), ou seja, aquela em que o usuário está a trabalhar, mais precisamente a adicionar faixas, e o id_Conteudo_Faixa é preenchido com o valor da faixa selecionado no ecrã AdicionarFaixas (BrowseGallery5), aquela que o usuário pretende adicionar à playlist.

Através destes dois ecrãs o processo de adicionar faixas a uma playlists torna-se mais intuitivo para o utilizador porque não têm de saber nem o id da playlist nem o id da música para o realizar

## Ecrã Faixas




Com origens de dados em '[bdtrabalho].[faixa]', pois é nesta tabela que estão os dados a apresentar.

O ícone Home, implementado com a função Navigate(Home), premite ao usuário navegar para o ecrã Home.

Items: SortByColumns(Search([@'[bdtrabalho].[faixa]']; TextSearchBox4.Text; "Nome"); "Nome"; If(SortDescending1; Descending; Ascending))

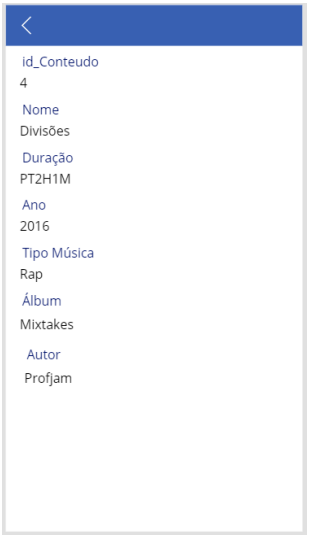
Utiliza o texto TextSearchBox4 para procurar combinações no campo nome através da função Search.

Utiliza a função SortByColumns para ordenar por "Nome", descending ou ascending, dependendo da valor do controlo.

OnSelect: Função Navigate(Player) indica que ao carregar em um dos items da lista (BrowseGallery4), irá fazer a navegação para o ecrã player.

O ícone informação presente em cada uma das faixas listadas serve para apresentar os detalhes das faixas, ou seja, tem implementado a função Navigate(DetailScreenFaixas) para aquando da seleção do item fazer a navegação para o ecrã DetailScreenFaixas.

## Ecrã DetailScreenFaixas

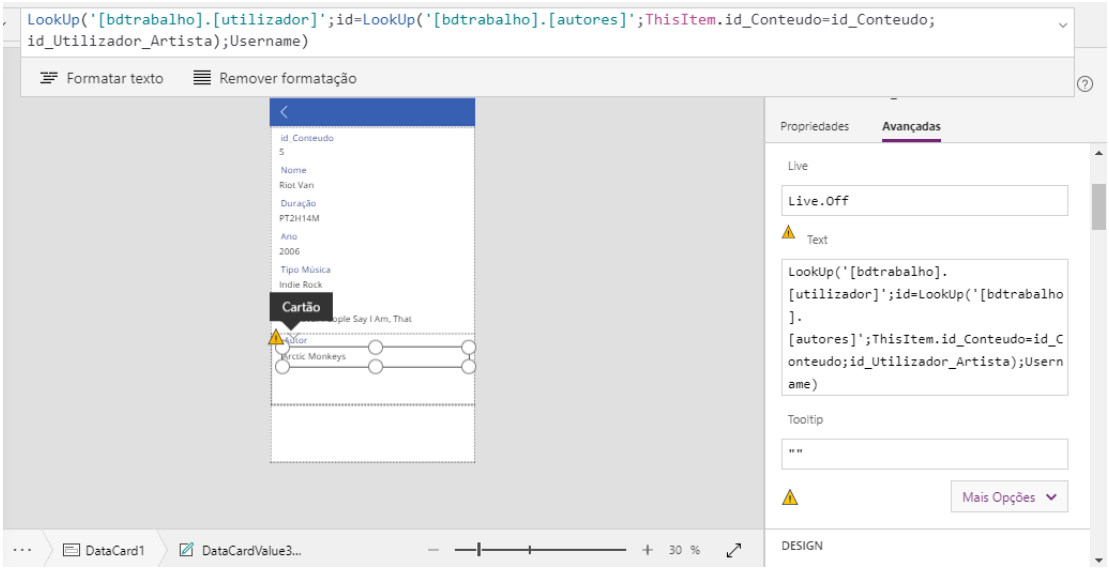



Origem de dados na tabela '[bdtrabalho].[faixa]'.

O formulário encontra-se no estado View, na vista de detalhe de um registo existente.

O registo a ser apresentado é selecionado através da propriedade BrowseGallery4.Selected, ou seja, apresentar o registo selecionado na lista de faixas no ecrã Faixas.

Um ícone de seta esquerda, implementado com a função Back(), para o utilizador poder voltar atrás na aplicação.




Para apresentar o autor da faixa selecionada pelo usuário recorremos a uma subconsulta, na medida em que implementámos um LookUp dentro de outro LookUp:
LookUp('[bdtrabalho].[utilizador]';id=LookUp('[bdtrabalho].[autores]';ThisItem.id_Conteudo=id_Conteudo;id_Utilizador_Artista);Username).

### Ecrã Player


Contém uma etiqueta de texto com a função BrowseGallery4.Selected.Nome para apresentar o nome da faixa selecionada pelo utilizador.

Um ícone de seta esquerda, implementado com a função Back(), para o utilizador poder voltar atrás na aplicação.

# Considerações Finais

Este trabalho possibilitou uma grande aprendizagem na área de Base de Dados SQL.

Ocorreram alguns problemas com a conta azure pois a subscrição foi terminada o que nos obrigou a reiniciar todo o progresso já feito.