**ESCOLA PROFISSIONAL DE CHAVES**

**Curso Técnico de Informática de Gestão**

**RELATÓRIO FINAL**

**Prova de Aptidão Profissional**

**Tiago José Barros Fernandes   
a3013**

**Chaves, 2018/2019**

# SUMÁRIO

[SUMÁRIO 2](#_Toc9969511)

[Introdução 4](#_Toc9969512)

[1 - ANÁLISE DO PROBLEMA 5](#_Toc9969513)

[1.1 - Aplicações disponíveis no mercado 5](#_Toc9969514)

[1.1.1 - Boonzi 5](#_Toc9969515)

[1.1.2 - Tosh finance 15](#_Toc9969516)

[1.2 - Aplicação GestMyMoney 24](#_Toc9969517)

[2 - BASE DE DADOS 26](#_Toc9969518)

[2.1 - Definição de Base de Dados 26](#_Toc9969519)

[2.2 - Definição de Sistema de Gestão de Base de Dados 26](#_Toc9969520)

[2.3 - Entidades e Relacionamentos (ER) 27](#_Toc9969521)

[2.4 - Base de Dados Relacional 30](#_Toc9969522)

[2.5 - Tabelas 31](#_Toc9969523)

[2.6 - Modelo Relacional 35](#_Toc9969524)

[3 - Manual do Programador 36](#_Toc9969525)

[3.1 - Base de Dados 37](#_Toc9969526)

[3.2 - Aplicação 44](#_Toc9969527)

[3.3 - Ficheiro “*Connect.cs*” 45](#_Toc9969528)

[3.4 - Ficheiro “SessaoSistema” 46](#_Toc9969529)

[3.5 - Formulário “Form1” 46](#_Toc9969530)

[3.6 - Formulário “VisaoGeral” 47](#_Toc9969531)

[4 - [Escreva aqui o título do 4º CAPÍTULO] 56](#_Toc9969532)

[Considerações Finais 57](#_Toc9969533)

[Bibliografia 58](#_Toc9969534)

[Anexo A – [Seleccione isto e escreva aqui o título do anexo] 59](#_Toc9969535)

[Anexo ? – [Seleccione isto e escreva aqui o título do anexo] 60](#_Toc9969536)

Introdução

Eu escolhi este tema pela necessidade de que as pessoas têm em gerir as suas despesas do dia-a-dia. Neste projeto irei utilizar como linguagem de programação o C# para desenvolver uma aplicação que permita gerir as despesas do quotidiano, cujos objetivos são consolidar, aprofundar e aplicar os conhecimentos adquiridos ao longo do curso, realizar uma aplicação de gestão de despesas do quotidiano em C# na qual seja possível registar digitalmente as despesas e rendimentos e que permita a sua consulta segundo vários filtros.

O relatório apresentado é estruturado em dois capítulos, a análise do problema que é constituído por uma descrição da mesma e está dividido em dois subcapítulos. O primeiro que descreve as aplicações consultadas para a realização do projeto, a aplicação “Boonzi” e o *site* “Toshl Finance”. O segundo aborda uma breve descrição da aplicação que se pretende codificar para trabalho da PAP, já com base em aplicações semelhantes e consultadas.

O segundo capítulo é sobre a base de dados, que apresenta uma descrição da mesma, assim como definição de sistemas de base de dados e os modelos de entidades e relacionamentos, base de dados relacional, tabelas e modelo relacional do projeto de PAP.

1. ANÁLISE DO PROBLEMA

Neste capítulo irá ser apresentada uma breve descrição de aplicações com o intuito de tirar ideias para a elaboração da prova de aptidão profissional. Assim, serão analisadas algumas aplicações existentes no mercado bem como explicadas as funcionalidades das mesmas

Aplicação é um termo que provém do vocábulo latim *applicatĭo* e que faz referência à ação e ao efeito de aplicar ou de se aplicar (pôr uma coisa em cima de outra, utilizar, empregar ou executar algo, atribuir).

Uma aplicação informática é um tipo de *software* que permite que o utilizador realize um ou vários tipos de trabalho. Os processadores de texto e as folhas de cálculo são exemplos de aplicações informáticas, ao passo que os sistemas operativos ou os programas utilitários (que cumprem tarefas de manutenção) não fazem parte destes programas.

As aplicações podem ser desenvolvidas à medida (para satisfazer as necessidades específicas do utilizador) ou fazer parte de um pacote integrado (como é o caso do Microsoft Office).

* 1. Aplicações disponíveis no mercado

Nesta secção irão ser apresentadas diversas aplicações disponíveis no mercado e encontradas mediante pesquisas na www. Serão mostradas as interfaces e as suas funcionalidades.

* + 1. Boonzi

O Boonzi é uma aplicação que ajuda o utilizador a gerir o dinheiro e planear o futuro, em termos de despesas. Encontra-se disponível em <http://www.boonzi.pt/>. Possui uma versão gratuita, mas a versão completa é paga e mais completa com versão para telemóvel e sincronização com a nuvem. O ambiente de trabalho da aplicação mostra-se na figura seguinte.



Figura 1 - Ambiente de trabalho da aplicação

Quando é aberta a aplicação pela primeira vez visualiza-se a seguinte interface para a criação do perfil do utilizador, onde cria o perfil e escolhe a divisa, ou seja, a moeda.

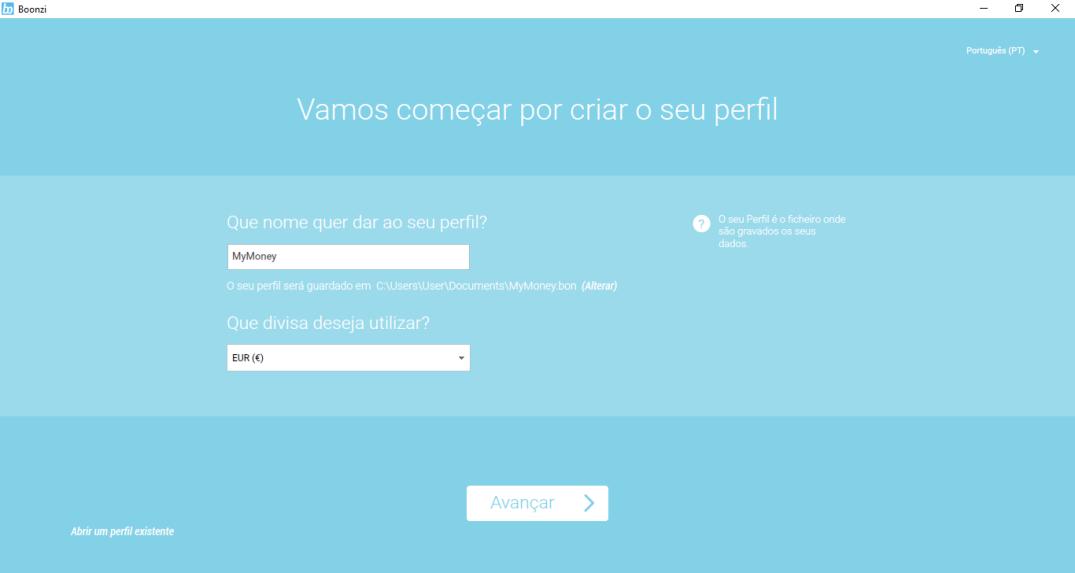


Figura 2 - Criação do perfil

Clicando em avançar aparece a possibilidade de criar uma senha.

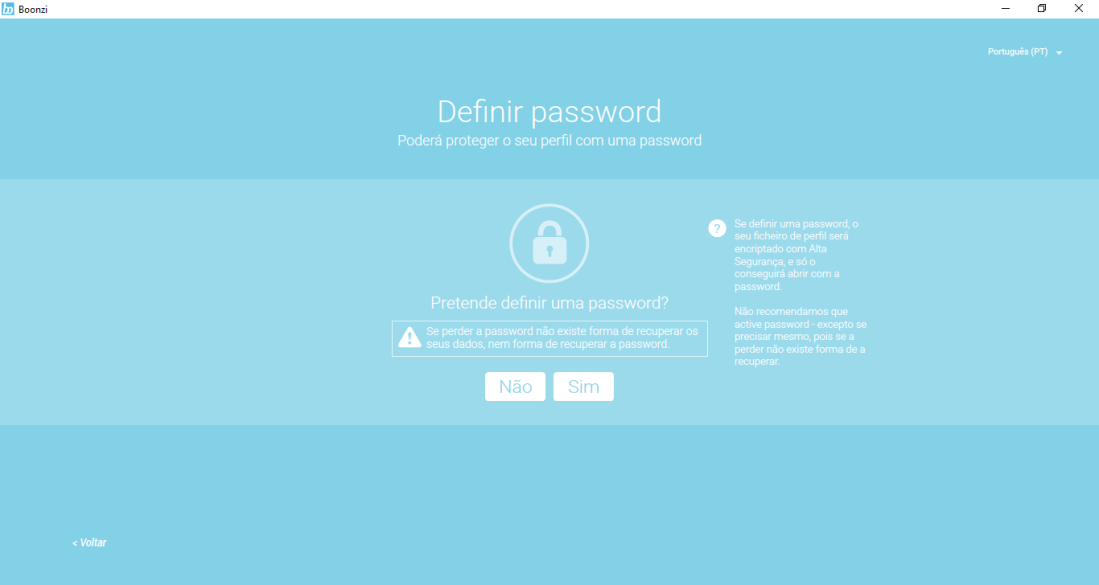


Figura 3 - Definir senha

A senha tem que respeitar alguns critérios de segurança, como ter no mínimo 8 carateres, e incluir um número e uma letra maiúscula.

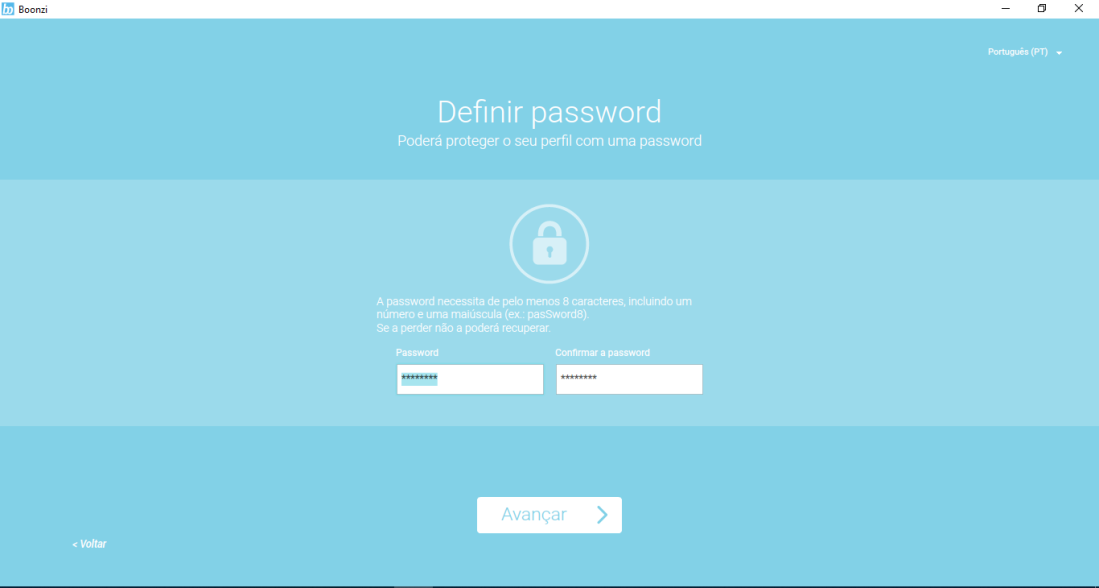


Figura 4 - Criar senha

Avançando é pedido para escolher as entidades e categorias do respetivo país.



Figura 5 - Entidades do país

A aplicação é constituída por um menu lateral com as opções: Visão Geral, transações, Relatórios (dividido em Categorias, Entidades, Mapa detalhado e evolução) Orçamentos, Agenda, Configuração (dividido em Perfil, Contas, Categorias, Entidades, Sincronizar, Mobile e Ficha técnica), Ajuda e todas as contas do utilizador.

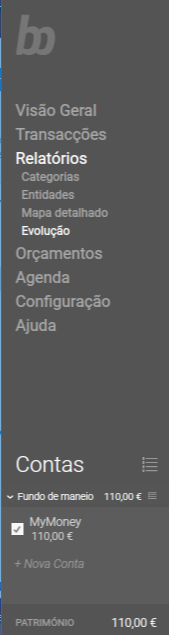


Figura 6 - Menu lateral

O menu transações apresenta todas as transações realizadas segundo os filtros pretendidos, e a respetiva janela para registar as despesas e as receitas.

Na janela de registar as despesas ou receitas é escolhida a entidade, data, conta, categoria, quantia, notas e a opção de anexar ficheiros e fotos. Também é possível ao clicar numa transação ver os detalhes da mesma, alterar e eliminar assim como a possibilidade de exportar os resultados para PDF, CSV.

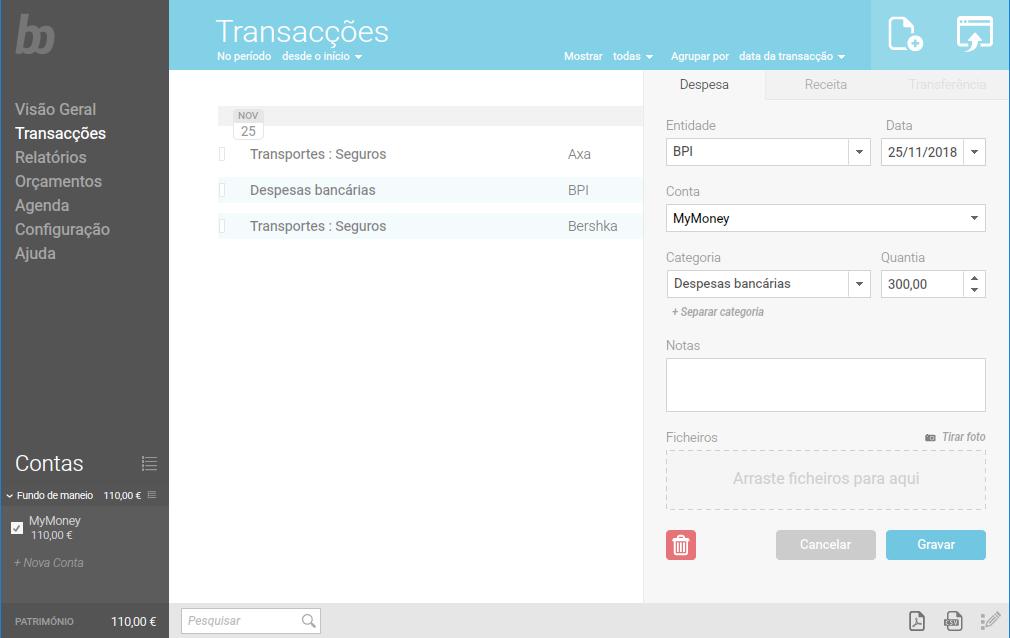


Figura 7 – Transações

No menu dos relatórios são apresentados gráficos circulares, de barras e linhas das categorias, entidades e evolução num determinado período de tempo e também a possibilidade de exportar os gráficos para PDF e CSV.

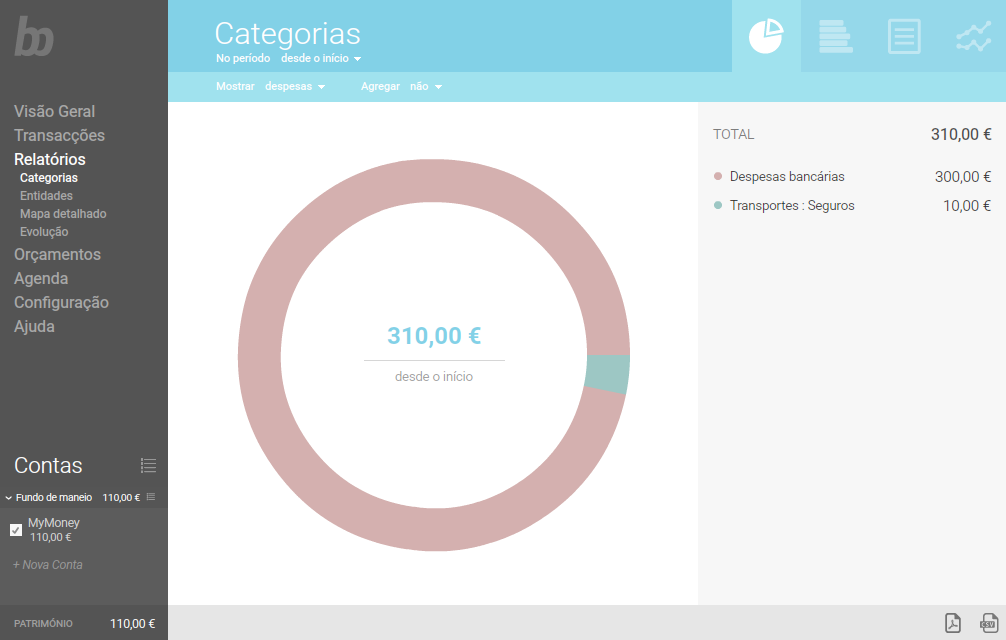


Figura 8 - Relatórios

No mapa detalhado são apresentadas todas as despesas e receitas divididas pelas entidades e referencias e por data, e também a possibilidade de exportar os resultados para PDF e CSV.



Figura 9 - Mapa detalhado

Na aplicação é também possível fazer orçamentos para prever as despesas e receitas para o mês atual. São apresentas as categorias, as despesas e receitas previstas, assim como o dinheiro que começou, gastou, ganhou e com quanto dinheiro terminará.



Figura 10 - Orçamento

É possível também agendar movimentos para um dia desejado, semelhante à criação de transações, e os movimentos podem ser consultados, alterados e eliminados.

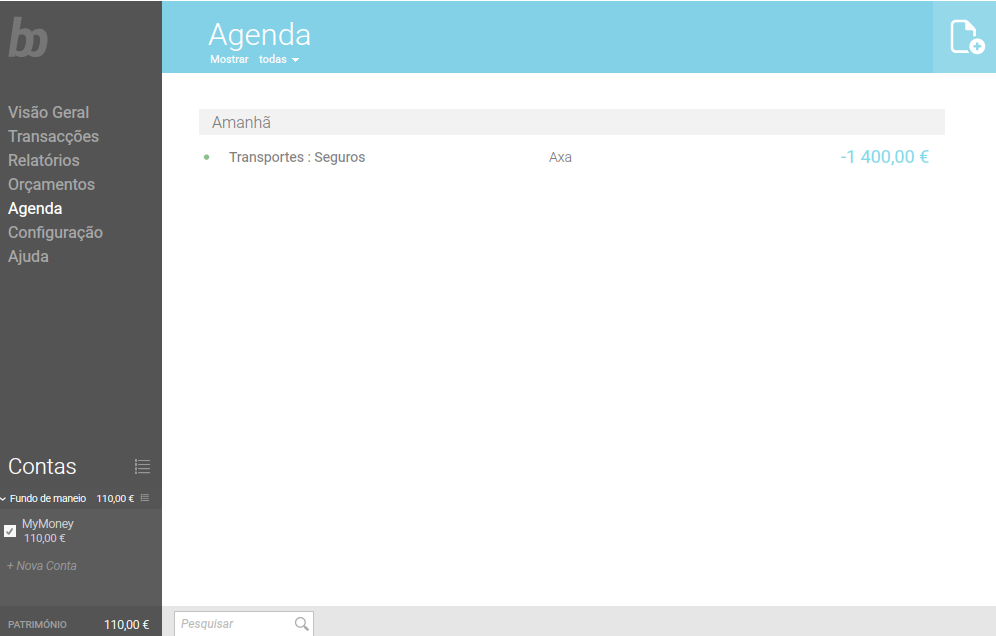


Figura 11 - Agenda

E por fim apresenta-se o menu de configurações em que é possível alterar a localização dos ficheiros de dados, idioma, divisa, o formato da divisa, o formato da data, a diretoria dos anexos das transações, da cópia de segurança automática e opção se o utilizador pretende efetuar a sincronização ao entrar e sair da aplicação.



Figura 12 – Perfil



Figura 13 - Perfil (continuação)

O separador de contas apresenta todas as contas do utilizador e a possibilidade de criar mais.

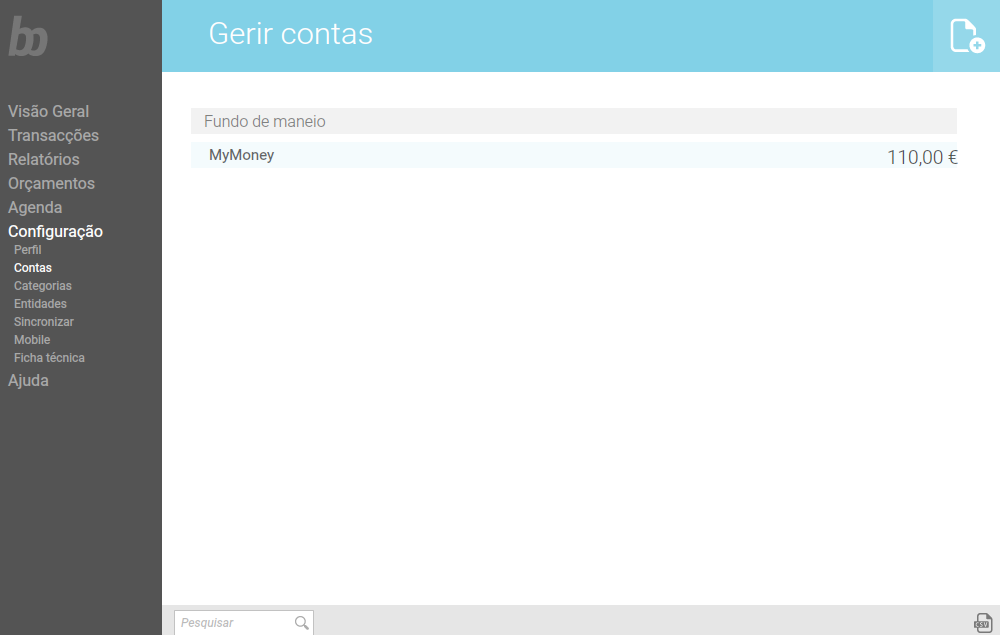


Figura 14 - Contas

O separador de categorias e entidades têm o mesmo propósito de consultar, alterar, eliminar e adicionar mais.

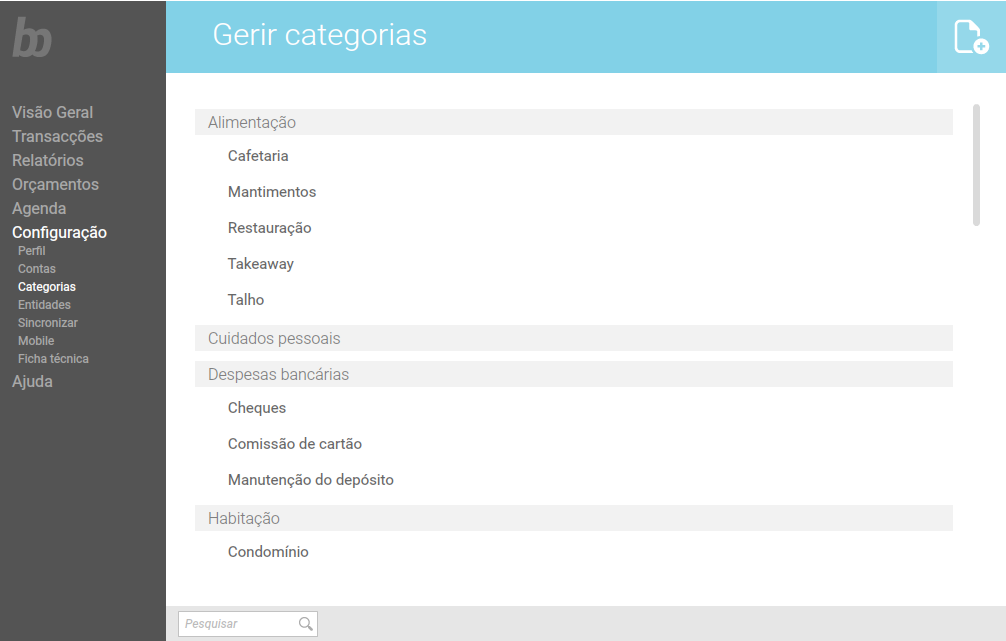


Figura 15 - Categorias



Figura 16 - Entidades

Também existe uma versão para dispositivos móveis e uma versão que permite a sincronização de todos os dispositivos à nuvem.

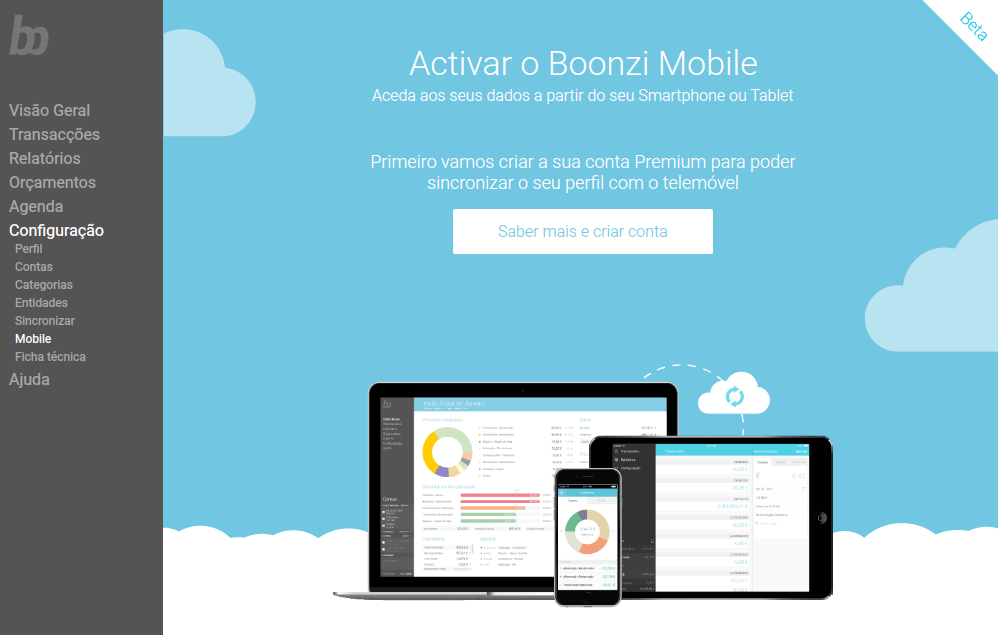


Figura 17 - Telemóvel

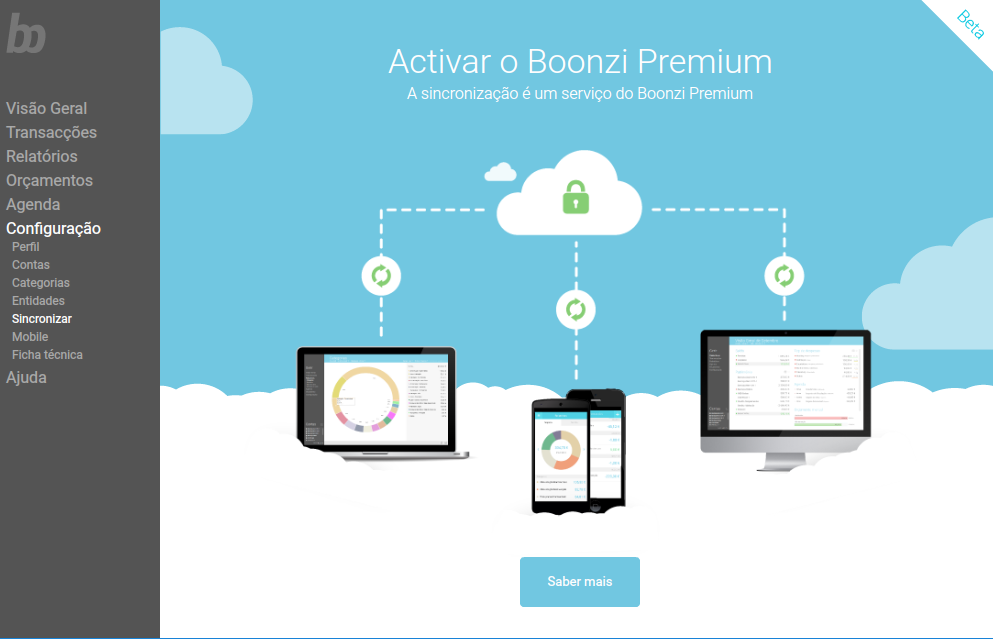


Figura 18 - Sincronizar

Por fim mostra-se a ficha técnica que apresenta todas as informações da aplicação, subscrição prémio e da empresa que criou a aplicação o site dos ícones usados e os apoios.



Figura 19 - Ficha técnica

* + 1. Tosh finance

O *Toshl Finance* encontra-se disponível em página *web* em <https://toshl.com/pt/> e em aplicativo para iOS (*iPod* *Touch*, *iPhone* e *iPad*), *Android*, *Windows Phone* e *Symbian* onde é possível registar todas as despesas e receitas pessoais e gerir de um modo simples e fácil. É também pago, mas possui versão grátis.

O *site* é composto por um menu lateral com todas as funções disponíveis, a parte central que apresenta o resumo mensal com gráficos de barras e menu lateral posicionado ao lado direito que apresenta todas as transações realizadas, organizadas por data, conta e categoria e com a opção de pesquisa.



Figura 20 - Página principal

O menu de despesas e receitas são também compostos por um menu lateral que apresenta todas as transações divididas por conta, data, categoria e com a opção de pesquisa, assim como o botão para criar novas transações.

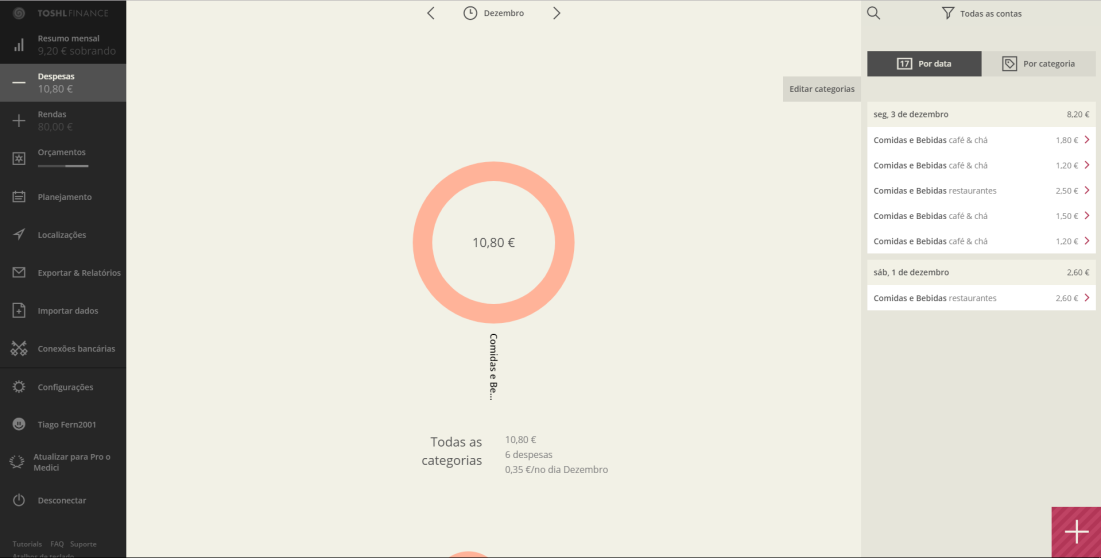


Figura 21 - Menu despesas



Figura 22 - Menu rendas

A página de adicionar uma despesa ou renda é composto por um campo onde introduz a quantia, o tipo de moeda, categoria, marcações, data, localização, descrição, repetir, lembrete, anexos e o respetivo botão para guardar

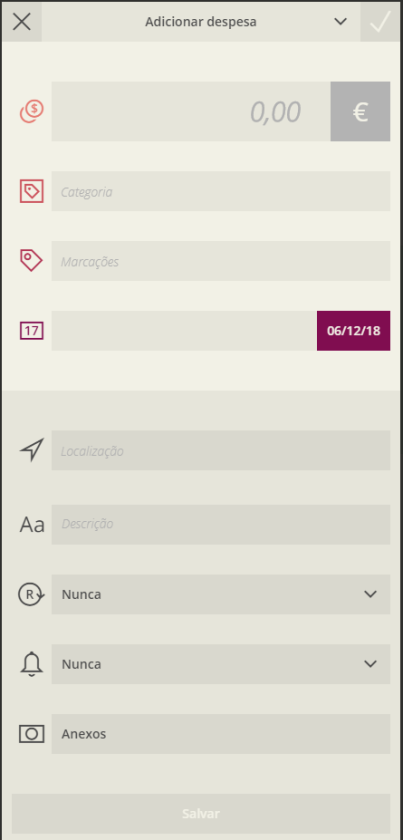


Figura 23 - Adicionar despesa

A página de adicionar uma transferência serve para realizar uma transferência de quantia de uma conta para a outra é semelhante ao de adicionar despesa, mas tem o campo para escolher a conta de origem e a conta de destino.

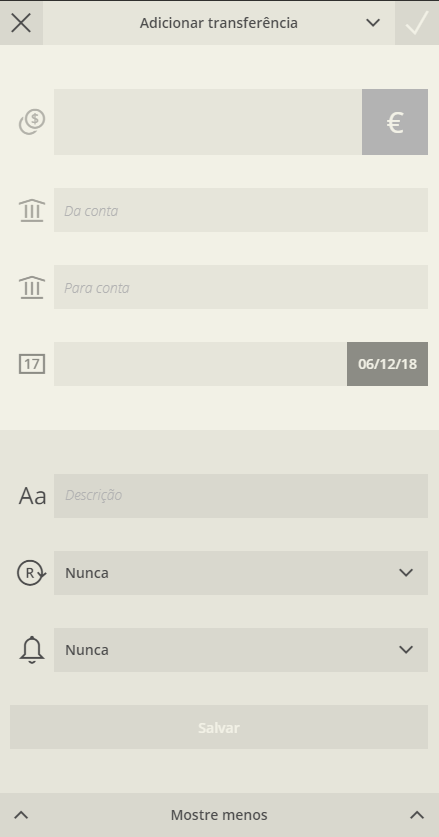


Figura 24 - Adicionar transferência

A página dos orçamentos apresenta a página lateral com os vários orçamentos planeados para o mês e para as várias contas, assim como o campo para uma pesquisa específica e um botão para criar um novo orçamento. Na parte central é apresentado o gráfico respetivo para o orçamento selecionado ao lado.

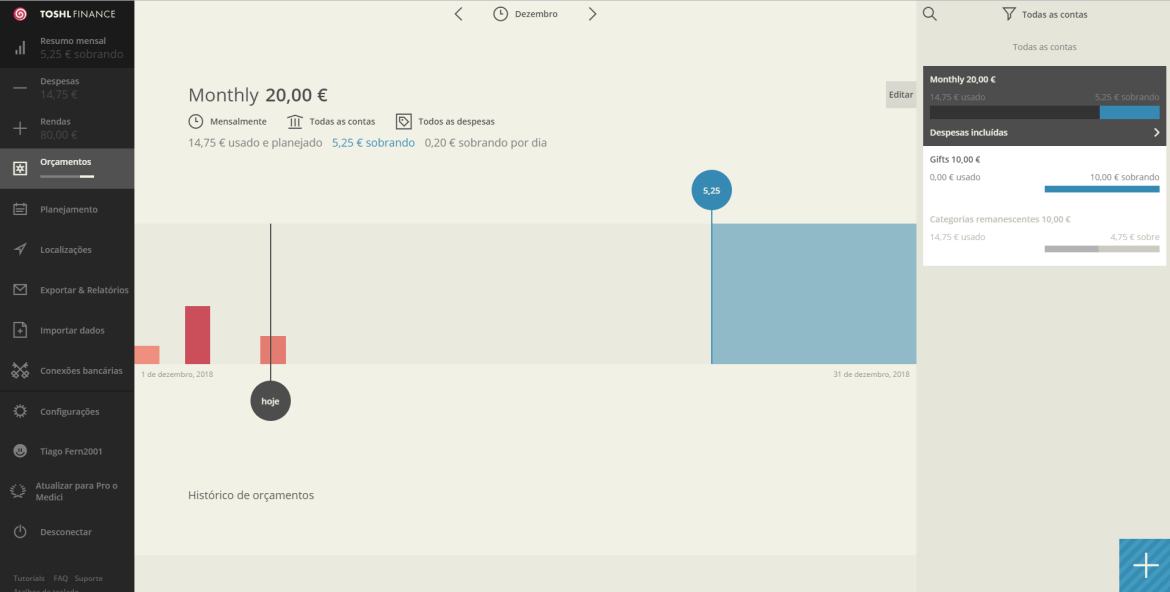


Figura 25 - Orçamentos

Existe também a opção de editar um orçamento onde é possível alterar o tipo de orçamento, escolher a conta, a repetição, a quantia, uma opção que permite mover os fundos restantes para o próximo período e escolher a quantidade que permite mover, o título, e os botões para excluir e salvar.

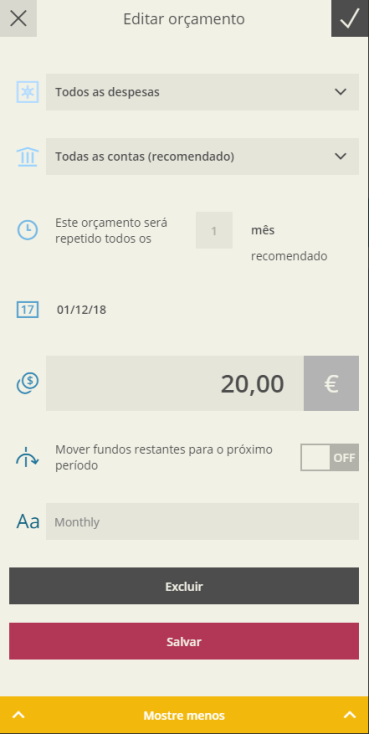


Figura 26 - Editar orçamento

O planeamento é uma função paga, mas apresenta imagens exemplos da sua interface, que é semelhante aos orçamentos com o menu lateral com as várias opções das quais contas, despesas e rendas das categorias, marcações das despesas e rendas, localização e uma opção se o utilizador deseja incluir estimativas futuras ou não no gráfico assim como o campo para fazer uma pesquisa especifica. Ao centro são apresentados os gráficos de planeamentos de balanços, despesas, rendas e valor líquido num determinado mês.



Figura 27 - Planeamento

Na página das localizações é apresentado o “Google Maps” com os pontos marcados na localização exata das despesas organizadas por categorias que se apresentam e podem ser consultadas no menu lateral direito.



Figura 28 - Localizações

O *site* dispõe também de uma opção para exportar os relatórios, escolhendo se pretende exportar apenas as despesas, as receitas ou ambas num mês específico, durante todo o tempo, ou personalizando um intervalo de tempo mais específico e com os filtros selecionados no menu lateral em vários ficheiros, dos quais “CSV”, “PDF”, “Excel”, “Google Drive”, “Evernote” e imprimir diretamente.



Figura 29 - Exportar relatórios

É possível também importar um ficheiro de dados para o site.

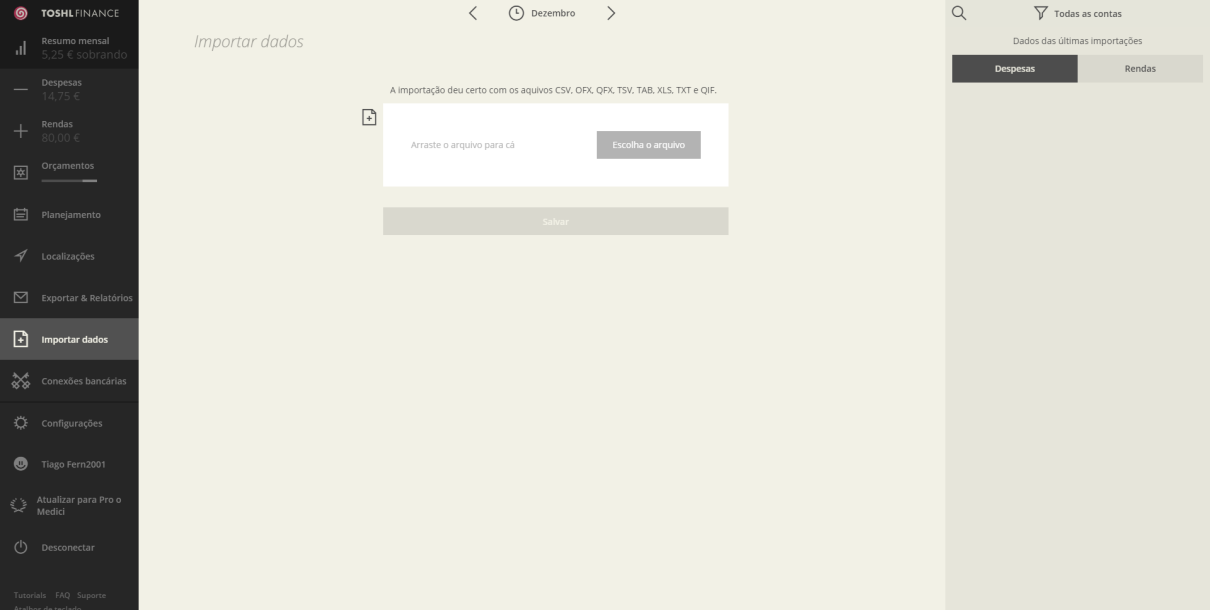


Figura 30 - Importar dados

O site dispõe também de conexão direta das contas do utilizador a um banco para obter o saldo e as transações realizadas.

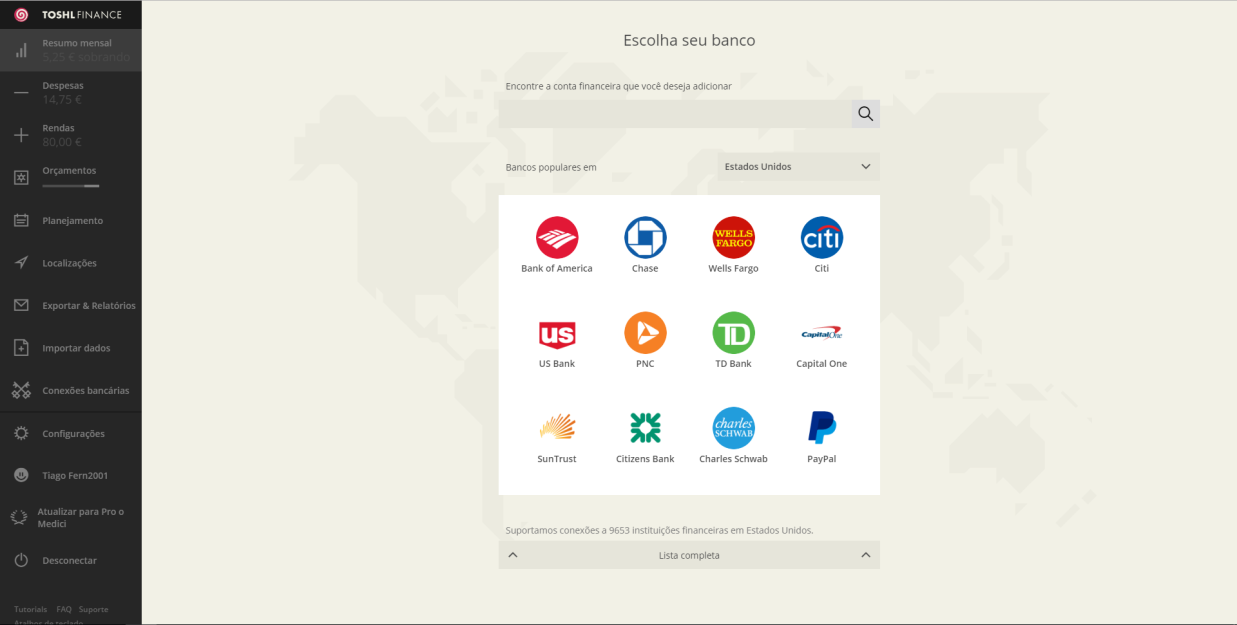


Figura 31 - Conexões bancárias

Na página das configurações é possível configurar a política do site, o país, a linguagem e o início do mês financeiro

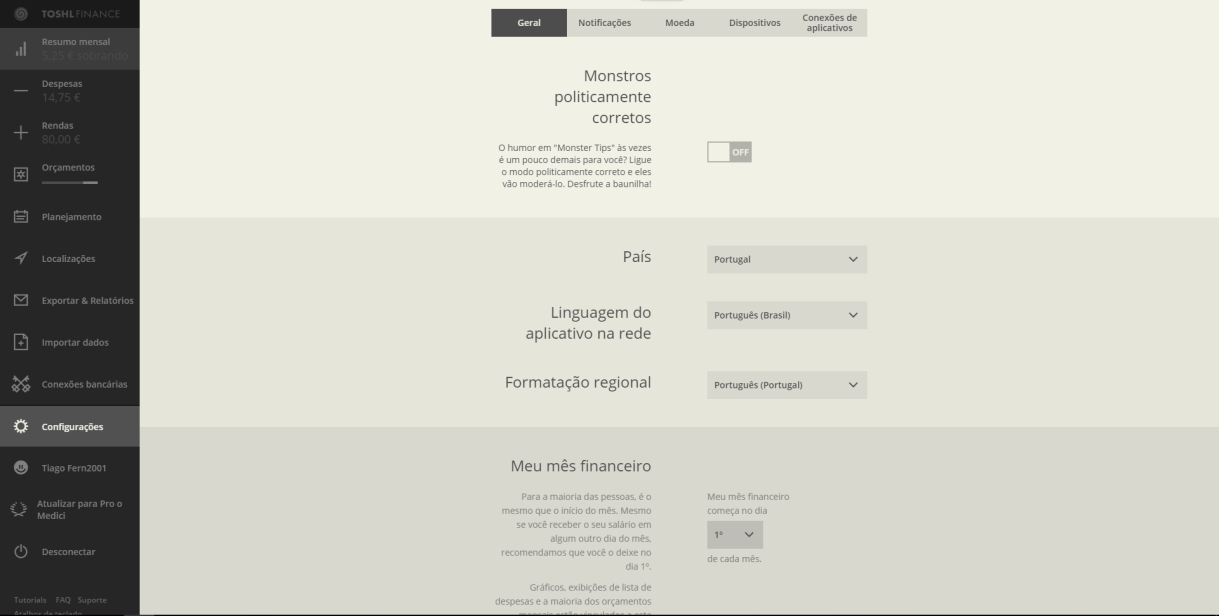


Figura 32 - Configurações

É possível ativar e desativar as notificações na aplicação e no *e-mail*, das quais o recebimento de notificações das despesas e rendimentos, dos orçamentos, dicas, noticias sobre o “Toshl”, relatórios mensais e lembretes diários para inserir as despesas

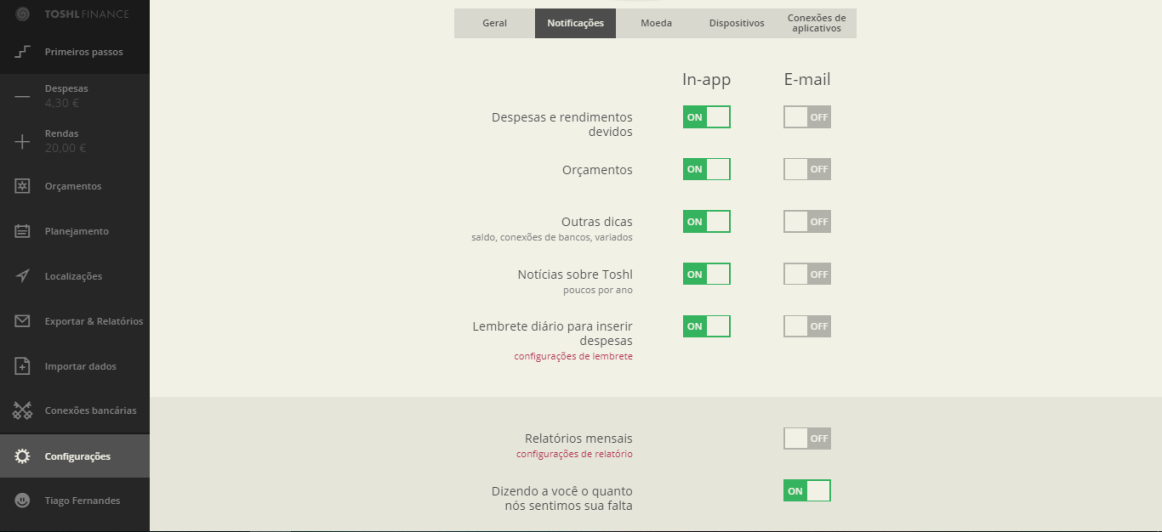


Figura 33 - Configurar notificações

O *site* possui também uma extensa variedade de moeda usadas no mundo para poder ser utilizado em qualquer país e introduzir a moeda que usa sem ter que efetuar conversões, tanto na moeda ativa como na moeda principal.



Figura 34 - Configurar moeda

Nas configurações também um separador que apresenta dispositivos móveis, assim como os conectados à conta.

E por fim as configurações da conta de utilizador, onde é possível alterar o nome, *E-mail*, palavra passe, autenticação de dois fatores, as contas conectadas, assim como o estado da subscrição e a opção de eliminar ou resetar a conta.

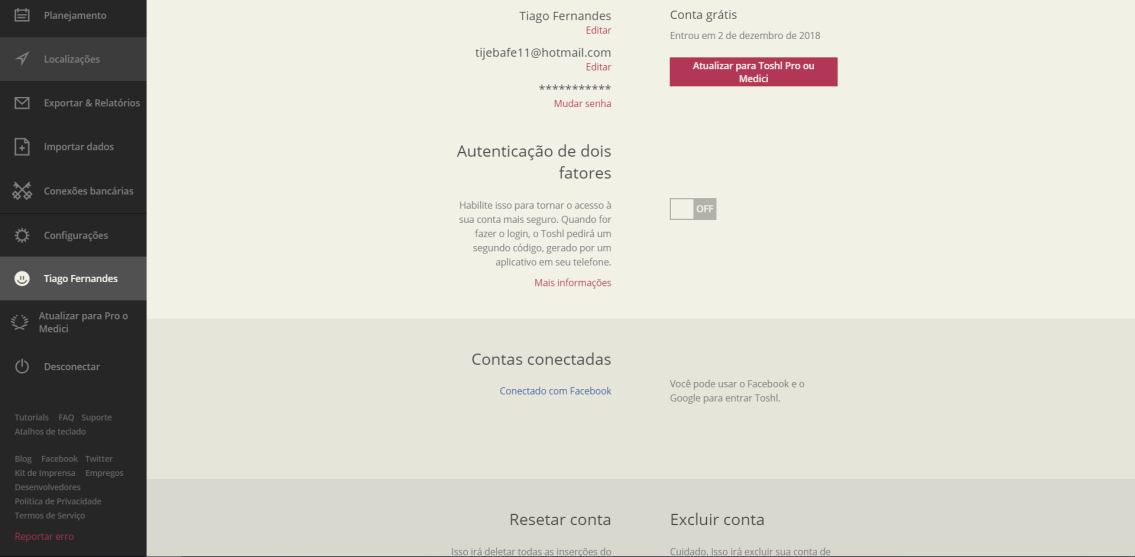


Figura 35 - Conta de utilizador

* 1. Aplicação GestMyMoney

Depois de analisar as duas aplicações apresentadas irá ser apresentada uma breve descrição da aplicação que se pretende codificar para trabalho da PAP.

Nesta aplicação o objetivo principal é gerir as despesas diárias do utilizador, na qual poderá ser introduzido os rendimentos, gastos e eventualmente algumas receitas, e obtenção dos respetivos gráficos. Nos rendimentos é possível registar se é salário, semanada, depósito… nos gastos e receitas é possível introduzir a quantia, lugar, localidade e data, apresenta também gráficos dos resultados finais segundo vários filtros e resultados ao final do dia, semana, mês, ano e se tem lucro ou prejuízo, é possível criar categorias, subcategorias, contas e tipos de rendimentos e para segurança o utilizador pode criar uma palavra passe para aceder á aplicação.

A minha projeção para o projeto é a seguinte:



Figura 36 - Visão geral da aplicação



Figura 37 - Despesas da aplicação

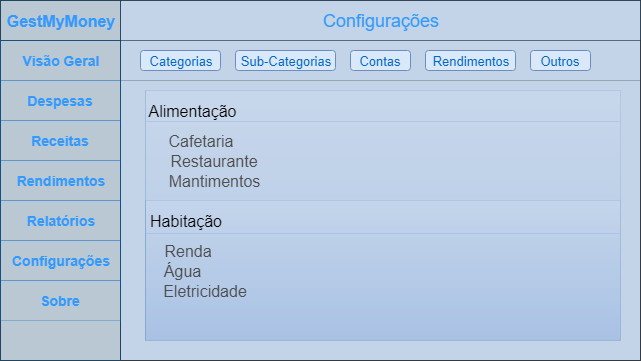


Figura 38 - Configurações da aplicação

1. BASE DE DADOS

Neste capítulo irá ser apresentado uma descrição de base de dados e a sua composição, nomeadamente definição de base de dados, definição de sistemas de gestão de bases de dados, entidades e relacionamentos, modelos ER, composição de base de dados relacionais, tabelas e o modelo relacional de todas as tabelas.

* 1. Definição de Base de Dados

Uma base de dados é uma ferramenta de recolha e organização de informações.

As bases de dados são repositórios de informação e podem armazenar informações sobre pessoas, produtos, encomendas ou qualquer outro assunto, ou seja, é uma coleção de dados ou itens estruturados de determinada maneira que permite a sua consulta, atualização e outros tipos de operação processados por meios informáticos.

São de vital importância para empresas e há duas décadas se tornaram a principal peça dos sistemas de informação. Normalmente existem por vários anos sem alterações em sua estrutura. São operados pelos [Sistemas de Gestão de Base de Dados](https://pt.wikipedia.org/wiki/Sistema_de_gerenciamento_de_banco_de_dados) (SGBD), que surgiram na década de 70.

Muitas Base de Dados começam por ser uma lista num programa de processamento de texto ou numa folha de cálculo. À medida que a lista cresce, começam a aparecer inconsistências e repetições nos dados. Os dados tornam-se difíceis de compreender num formato de lista e as maneiras para efetuar pesquisas ou selecionar subconjuntos de dados para revisão são limitadas. Assim que estes problemas começam a aparecer, convém transferir os dados para uma base de dados criada pelo sistema de gestão de bases de dados (SGBD) como o Access.

* 1. Definição de Sistema de Gestão de Base de Dados

Inicialmente, os dados ficavam dependentes dos programas que os geravam e manipulavam, para alterar a estrutura da informação era necessário alterar todo o programa, e consequentemente introduzir novamente todos os dados.

Surgiu, por isto, a necessidade de guardar a informação de forma que o acesso e a gestão fossem feitos de forma mais flexível, dando assim origem ao aparecimento dos SGBD.

Um (SGBD) Sistema de Gestão de Bases de Dados (ou Gestor de Bases de Dados) é o software que gere o armazenamento, manipulação, alteração e pesquisa dos dados existentes na base de dados. Os dados são independentes dos programas que os manipulam, assim o seu objetivo é registar e manter a informação que for considerada necessária à organização/pessoa que gere o sistema, disponibilizando-a automaticamente para os mais diversos fins.

Através de um SGBD, pode realizar-se um vasto conjunto de operações de gestão de base de dados, das quais:

* Inserção, edição e apagamento de registos;
* Critérios de visualização de registos;
* Indexação e ordenação da informação contida nos registos;
* Operações estatísticas sobre os dados;
* Criação de ecrãs de apresentação;
* Acesso à informação através das tecnologias Internet;
* Impressão de relatórios;
* Automatização de funções;
* Programação;

Algumas vantagens na utilização duma base de dados:

* Diminuição de espaço físico ocupado;
* Maior integridade dos dados;
* Menos redundância;
* Mais facilidade na partilha de dados;
* Maior facilidade de manutenção;
* Isolamento entre objetos de dados, o que protege a integridade da origem dos dados;
* Facilidade de mudança na criação de diferentes mapas com diferentes objetivos;

Requisitos fundamentais de um SGBD

* **Segurança** - proteção da base de dados contra acessos não autorizados;
* **Integridade** - validação de operações que coloquem em risco a consistência dos dados;
* **Controlo de concorrência nos acessos** - coordenação da partilha dos dados pelos vários utilizadores (SGBDs multiutilizador);
* **Recuperação de falhas** - restaurar a integridade da base de dados depois da ocorrência de uma falha. Mecanismos de recuperação (fundamentalmente baseados em redundância): *backups*, *transaction logging* (ficheiro *transaction log*, dados para repor as últimas transações).
  1. Entidades e Relacionamentos (ER)

Um diagrama entidade relacionamento (ER) é um tipo de fluxograma que ilustra como “entidades”, p. ex., pessoas, objetos ou conceitos, se relacionam entre si dentro de um sistema. Diagramas ER são mais utilizados para projetar ou detetar erros em bases de dados relacionais nas áreas de engenharia de software, sistemas de informação empresarial, educação e pesquisa. Também conhecidos como DERs, ou modelos ER, usam um conjunto definido de símbolos, tais como retângulos, losangos, ovais e linhas de conexão para representar a interconetividade de entidades, relacionamentos e seus atributos. Eles espelham estruturas gramaticais, onde entidades são substantivos e relacionamentos são verbos.

Na sua notação original, proposta por Peter Chen (idealizador do modelo e do diagrama), as entidades deveriam ser representadas por retângulos, os seus atributos por elipses e os relacionamentos por losangos, ligados às entidades por linhas, contendo também uma cardinalidade (1-1, 1-n ou n-n). Porém, notações mais modernas abandonaram o uso de elipses para atributos e passaram a utilizar o formato mais utilizado na UML, em que os atributos já aparecem listados na própria entidade. Essa forma torna o diagrama mais limpo e fácil de ser lido.

Existem três tipos de relacionamentos:

* **Relacionamento 1:1 (lê-se relacionamento um para um) -** Indica que as tabelas que têm relacionamento apenas entre si;
* **Relacionamento 1:n (lê-se um para muitos) -** A chave primária da tabela que tem o lado 1 vai para a tabela do lado N. No lado N ela é chamada de chave estrangeira;
* **Relacionamento n:n (lê-se muitos para muitos) -** Quando tabelas têm entre si relacionamento n:n, é necessário criar uma nova tabela com as chaves primárias das tabelas envolvidas, ficando assim uma chave composta, ou seja, formada por diversos campos-chave de outras tabelas. O relacionamento então se reduz para uma relacionamento 1:n, sendo que o lado n ficará com a nova tabela criada.

Os meus diagramas de entidades e relacionamentos são os seguintes:



Figura 39 – Relação Utilizador Rendimentos

O utilizador tem vários rendimentos e vários rendimentos pertencem a um só utilizador.



Figura 40 – Relação Contas Rendimentos

Uma conta tem vários rendimentos e vários rendimentos utilizam apenas uma conta.



Figura 41 – Relação Rendimentos Tipos de Rendimentos

Vários rendimentos utilizam um tipo de rendimento e um tipo de rendimento é utilizado por vários rendimentos.



Figura 42 – Relação Utilizador Gastos

O utilizador regista vários gastos e vários gastos são registados por um utilizador.



Figura 43 – Relação Utilizador Receitas

O utilizador regista várias receitas e várias receitas são registadas por um utilizador.



Figura 44 – Relação Gastos Categorias

Vários gastos utilizam a mesma categoria e uma categoria é utilizada por vários gastos.



Figura 45 – Relação Categoria Subcategorias

Uma categoria tem várias subcategorias e várias subcategorias pertencem a uma categoria.

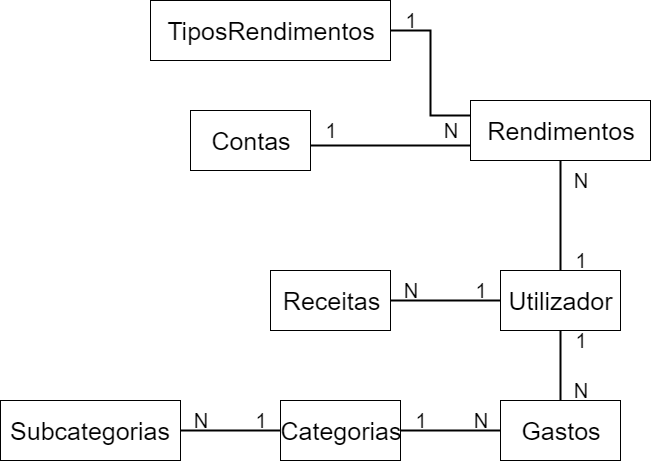


Figura 46 - Modelo ER

O utilizador tem várias opções, das quais inserir os seus rendimentos, receitas e gastos. Cada rendimento utiliza um tipo de rendimento e uma conta, mas que por sua vez um tipo de rendimento é utilizado por vários rendimentos, assim como as contas. Pode também registar as receitas e os gastos, e um gasto tem uma categoria, que por sua vez uma categoria é dividida em várias subcategorias, e uma categoria é utilizada por vários gastos.

* 1. Base de Dados Relacional

A arquitetura de uma base de dados relacional pode ser descrita de maneira informal ou formal. Na descrição informal a preocupação é em aspetos práticos da utilização, e são usados os termos tabela, linha e coluna. Na descrição formal existe a preocupação com a semântica formal do modelo e são usados os termos como relação (tabela), tupla(linha) e atributo(coluna).

Uma base de dados relacional é constituída por:

* **Tabela –** É um conjunto de dados dispostos em número infinito de colunas e número ilimitado de linhas
* **Campo –** Corresponde a uma das colunas da tabela. Local onde se armazena informação com uma mesma característica.
* **Registo –** Corresponde a uma das linhas da tabela. Conjunto de informação sobre os diferentes campos.
* **Chave primária** - corresponde a um campo ou conjunto de campos que distingue cada registo de todos os outros registos, dado que cria um índice automático em que as referências de cada registo têm de ser sempre diferentes.
* **Chave estrangeira** - é o campo que estabelece o relacionamento entre duas tabelas.
  1. Tabelas

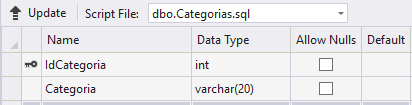


Figura 47 - Tabela Categorias

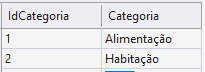


Figura 48 - Registos Categorias

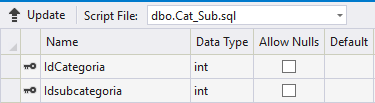


Figura 49 - Tabela Cat-Sub

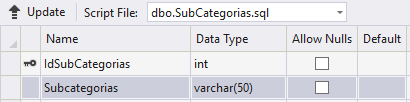


Figura 50 - Tabela SubCategorias

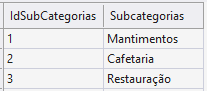


Figura 51 - Registos SubCategorias



Figura 52 - Tabela Contas



Figura 53 - Registos Contas

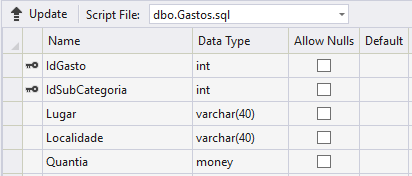


Figura 54 - Tabela Gastos

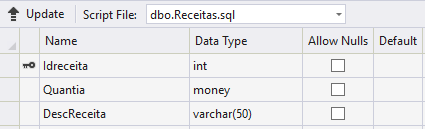


Figura 55 - Tabela Receitas

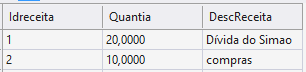


Figura 56 - Registos Receitas

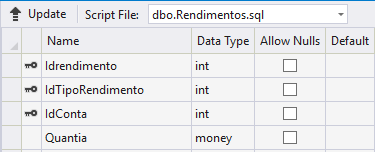


Figura 57 - Tabela Rendimentos

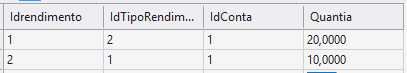


Figura 58 - Registos Rendimentos

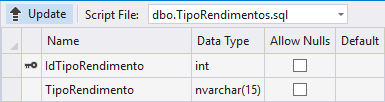


Figura 59 - Tabela Tipos de Rendimentos

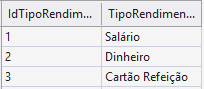


Figura 60 - Registos Tipos de Rendimentos

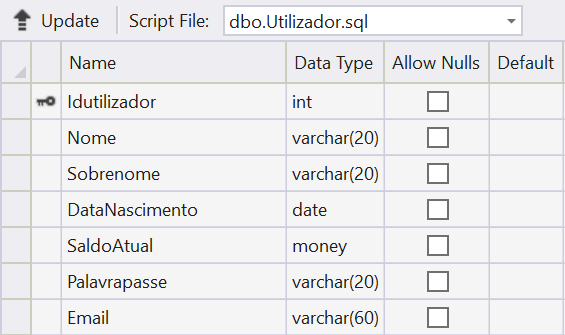


Figura 61 - Tabela Utilizador

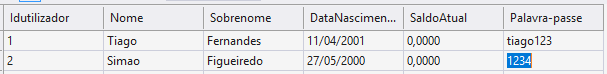


Figura 62 - Registos Utilizadores

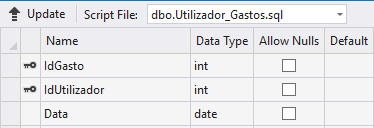


Figura 63 - Tabela Utilizador-Gastos

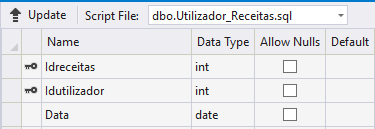


Figura 64 - Tabela Utilizador-Receitas

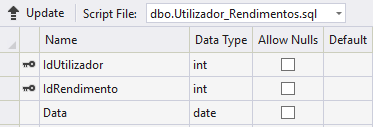


Figura 65 - Tabela Utilizador-Rendimentos

* 1. Modelo Relacional



Figura 66 - Modelo Relacional

1. Manual do Programador

Neste projeto usei o programa “Microsoft Visual Studio 2017” e a base de dados também criei dentro do mesmo em “*SQL Server Database File*” com a linguagem c# e sql.

De início abre-se o “visual Studio” e para criar o projeto é só clicar em “*create new Project…*” no canto inferior direito da página inicial

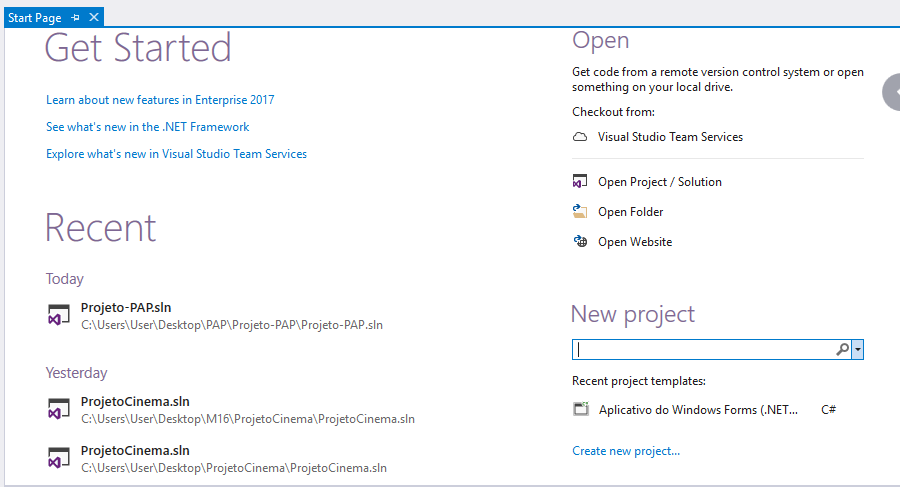


Figura 67 - Página inicial

De seguida na janela do lado esquerdo escolhe-se “visual C#”, o modelo em branco, o nome e a localização.

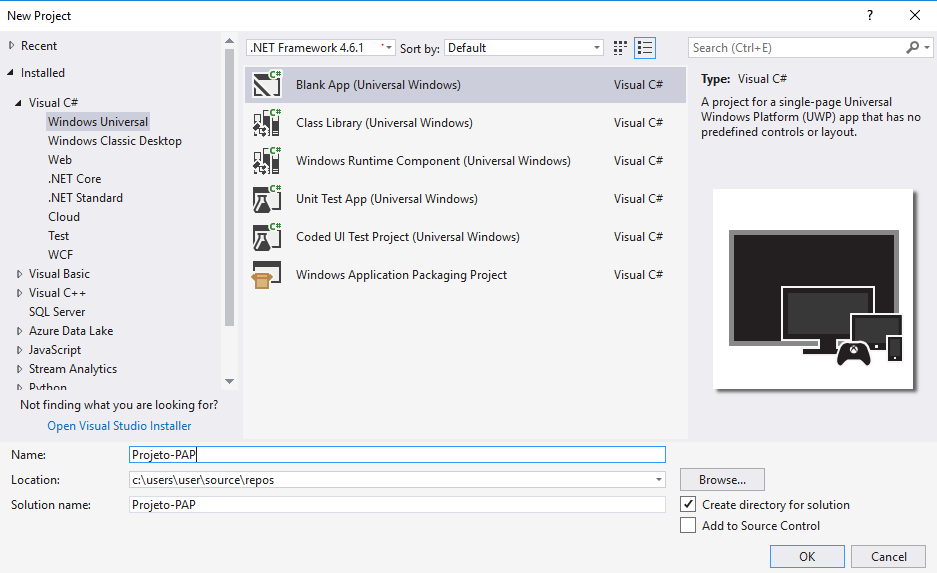


Figura 68 - Novo projeto

Em seguida cria-se a base de dados.

* 1. Base de Dados

Na criação da base de dados acede-se ao “*server explorer*” e clica-se no botão para conectar á base de dados.

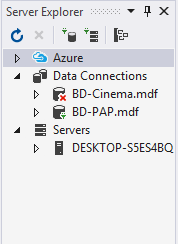


Figura 69 - Server explorer

No qual aparece a janela onde se escolhe a fonte de dados, no caso é “*Microsoft SQL Server Database File(SqlClient*)” e o nome, se já existir vai ligar, se não existir a base de dados com aquele nome vai criar uma nova.

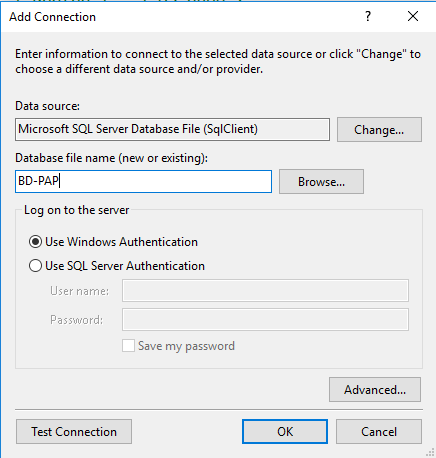


Figura 70 - Adicionar conexão

Com a base de dados criada, no “*server explorer*” abre-se “*Data Connections*”, a base de dados e com o botão direito em cima de “*Tables*” escolhe-se a opção “Add new table” para adicionar uma nova tabela á base de dados

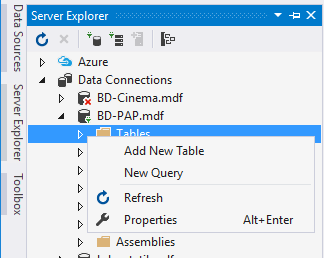


Figura 71 - Adicionar tabela

Em seguida define-se o nome, tipo de dados, o comprimento e se permite valores nulos.

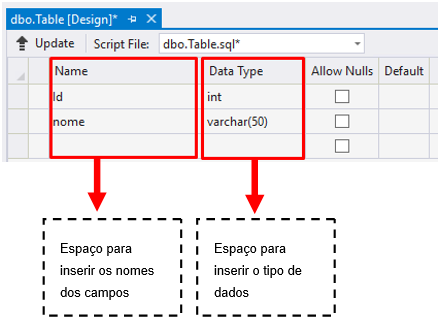
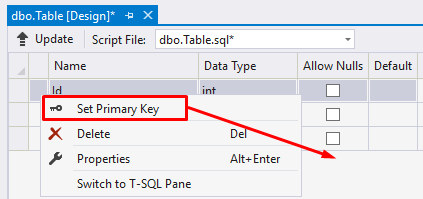


Figura 72 - Inserir campos

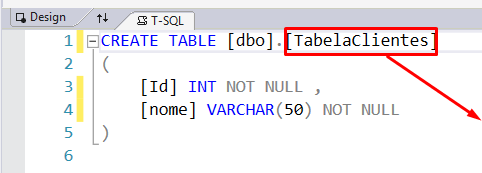
Selecionando-se um campo e com o botão direito é possível definir a chave primária.



Definir chave primária

Figura 73 - Definir chave primária

Na parte inferior tem o código para a criação da tabela e é onde se define o nome da mesma.



Definir o nome da tabela

Figura 74 - Nome da tabela

E por fim é necessário fazer “Update” para criar a tabela e guardar as alterações

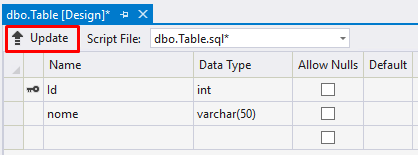


Figura 75 - Atualizar

Em seguida é necessário fazer a ligação á base de dados, para isso na janela “*Data Sources*” e clicar no botão “Add new data source”.

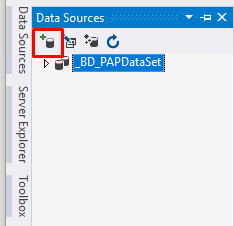


Figura 76 - *Data sources*

Na janela que aparece escolhe-se a opção “*Database*”.

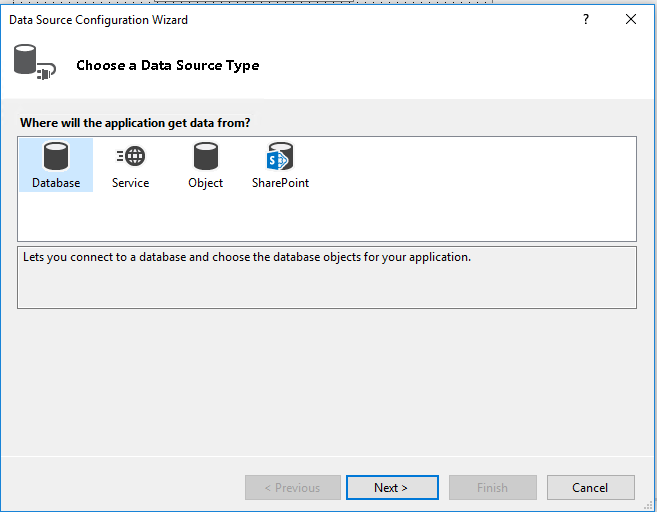


Figura 77 - Escolher “*Database”*

E a opção “*Dataset*” e prosseguir.

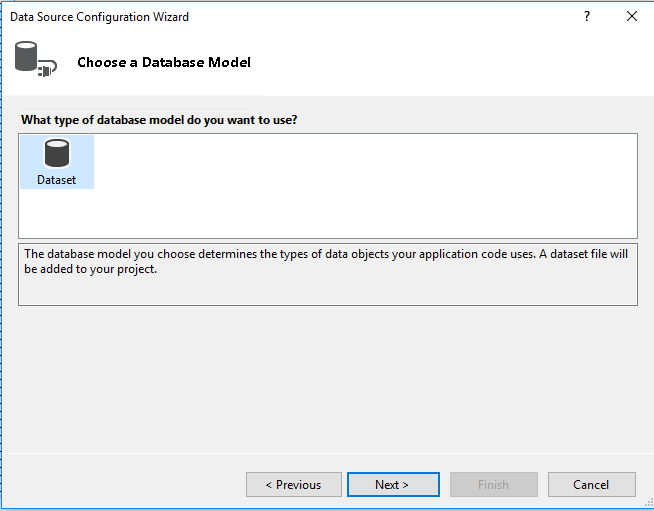


Figura 78 - Escolher "*dataset*"

Em seguida clica-se em “*New Connection…*”

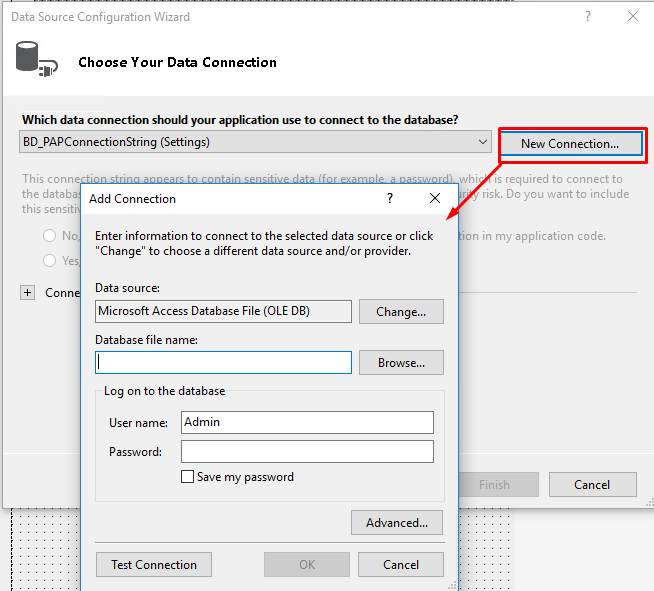


Figura 79 - Nova conexão

É necessário escolher a fonte de dados, para isso clica-se no botão “*Change*”, escolhe-se a opção “Microsoft SQL Server Database File”

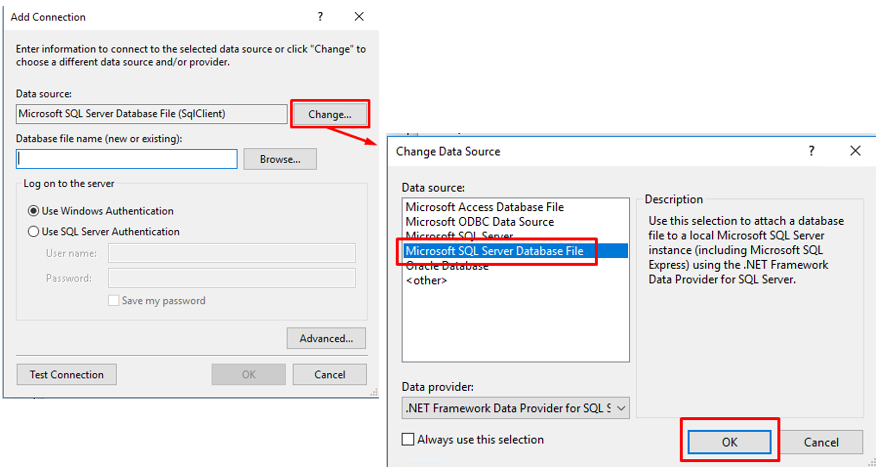


Figura 80 - Escolher o "*Data source*"

Em seguida vai-se buscar o ficheiro da base de dados, para isso clica-se no botão “Browser…”, vai-se á localização, escolhe-se o ficheiro .mdf que foi criado anteriormente e abrir.



Figura 81 - Buscar a base de dados

Avançar para o próximo passo.

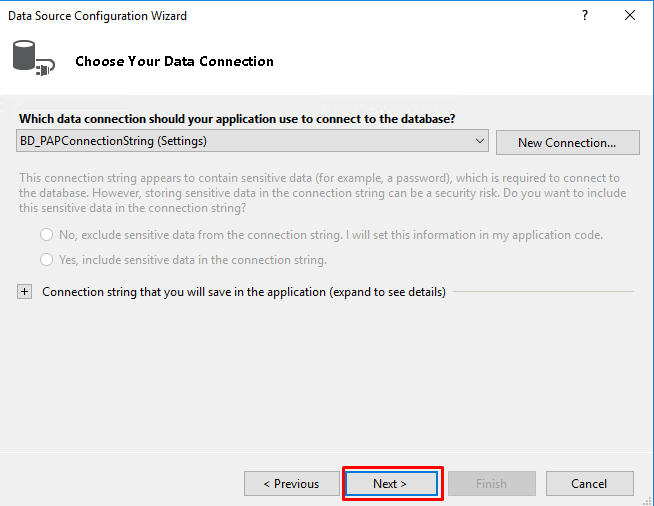


Figura 82 - Seguir o "*Data source*"

Aqui escolhe-se todas as tabelas criadas.

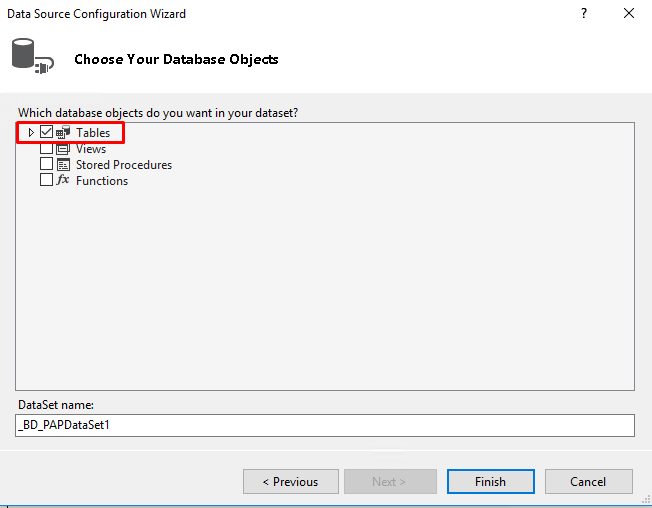


Figura 83 - Escolher as tabelas

E por fim é necessário efetuar as relações, para isso na janela do “Soluction Explorer”, e duplo clique no ficheiro do dataset que é .xsd

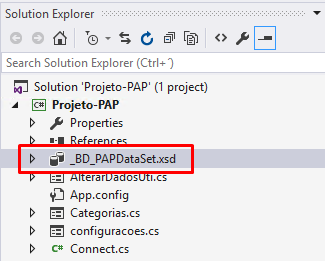


Figura 84 - Ficheiro .xsd

Em seguida adiciona-se a relação

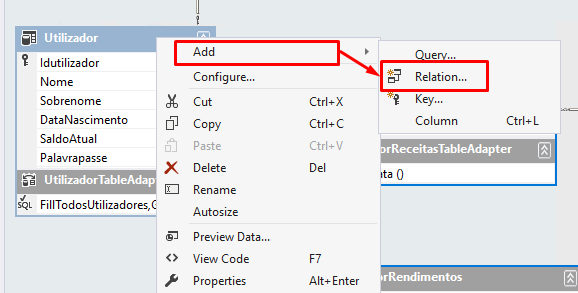


Figura 85 - Adicionar relação

Escolhe-se a tabela “pai” e “filha” e as respetivas colunas que se vão ligar e a opção ”Boht relacion and foreign key Constraint”

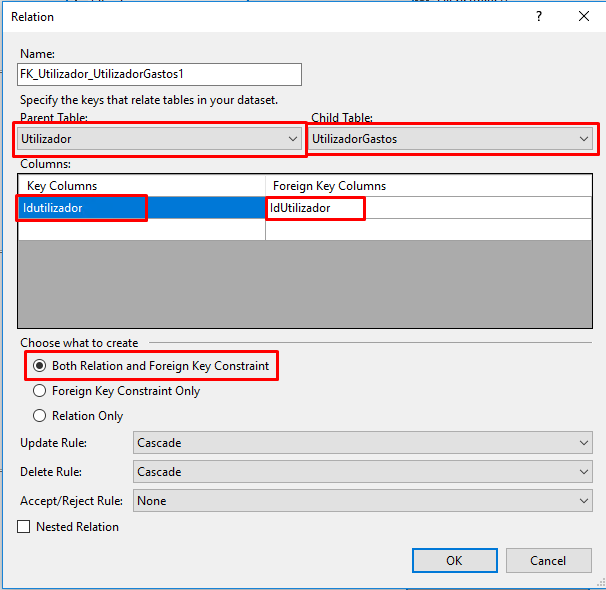


Figura 86 - Tabelas a relacionar

E por fim esta é a aparência das relações

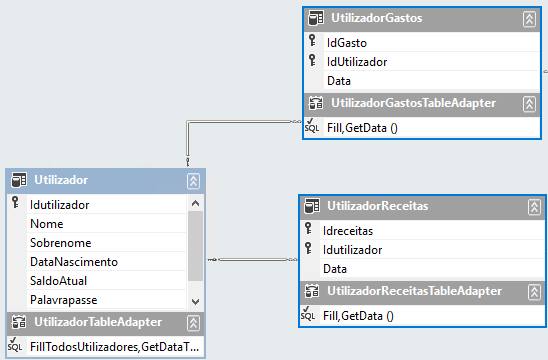


Figura 87 - Tabelas relacionadas

* 1. Aplicação

Todos os formulários têm o “try” e “catch” para o caso de existirem erros para informar, assim como “MessageBox’s” para a informação do utilizador se existem erros ou inseriu com sucesso

Try

{

// comandos

}

catch (Exception ex)

{

MessageBox.Show("Não foi possível preencher\n\nErro:" + ex, "Erro!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

Assim como também o “finally” que tem os commandos que são sempre executados quando termina a condição “try” ou a “catch”

finally

{

obj.con.Close();

obj.con.Dispose();

}

* 1. Ficheiro “*Connect.cs*”

O ficheiro “*connect.cs*” serve para fazer a conexão com a base de dados

public SqlConnection con = new SqlConnection();

public SqlCommand cmd = new SqlCommand();

public string locate = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=" + Application.StartupPath + "\\BD\\BD-Cinema.mdf;Integrated Security=True; Connect Timeout=30;Trusted\_Connection=True";

crio a conexão, o sql comand e depois busco a localização do ficheiro .mdf que é atribuída á variável “locate”

Todos os formulários têm consultas ou “*inserts*” á base de dados, e para isso é necessário e efetuar a conexão á base de dados antes de qualquer consulta e é sempre necessário abrir, fechar e limpar a sua conexão

Connect obj = new Connect();

obj.con.ConnectionString = obj.locate;

obj.con.Open();

obj.con.Close();

obj.con.Dispose();

* 1. Ficheiro “SessaoSistema”

Este ficheiro é uma classe que serve para armazenar os dados do administrador que são registados no “login” e depois ficam armazenados para consulta

Primeiro defino o nome da classe, as variáveis e depois crio um evento para armazenar e retornar os valores do email e a palavra passe

public static class SessaoSistema

{

private static string \_Email;

private static string \_Palavrapasse;

public static string EmailUsuario

{

get { return SessaoSistema.\_Email; }

set { SessaoSistema.\_Email = value; }

}

public static string PasswordUsuario

{

get { return SessaoSistema.\_Palavrapasse; }

set { SessaoSistema.\_Palavrapasse = value; }

}

}

Depois nos formulários sempre que for preciso usar saber o id do utilizador ou a palavra passe basta usar o código “SessaoSistema.EmailUsuario” ou “SessaoSistema.PasswordUsuario”

* 1. Formulário “Form1”



Figura 88 - Formulário "Form1"

O formulário form1 é o formulário “pai” e principal, onde tem o menu com os botões para navegação na aplicação.

VisaoGeral a = new VisaoGeral();

a.MdiParent = this;

a.WindowState = FormWindowState.Maximized;

a.Show();

Dentro do evento “click” de cada botão tem o código para abrir um novo formulário, no caso o botão de adicionar outros vai abrir o seu formulário, no código primeiro cria-se um novo formulário, depois define-se que este é o formulário pai(no caso o formulário pai é o form1), depois defini a propriedade “WindowsState” como maximizada e por fim a propriedade “Show” para aparecer o formulário.

* 1. Formulário “VisaoGeral”

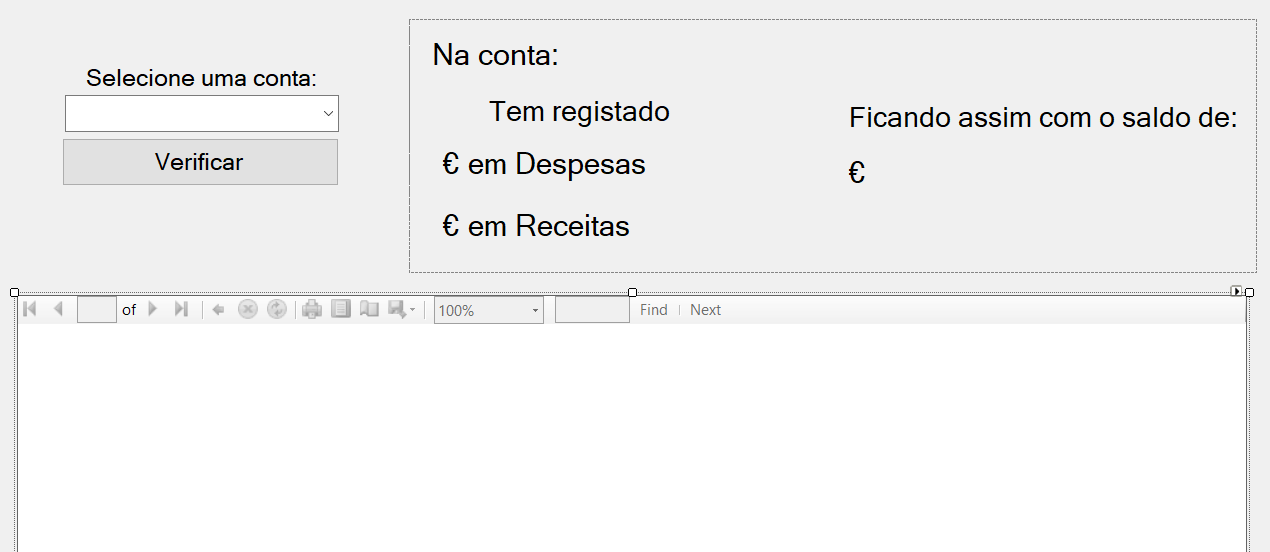


Figura 89 - Formulário "VisaoGeral"

O formulário “Visão geral é o formulário principal da aplicação onde é apresentado as despesas e as receitas efetuadas em cada conta, assim como o saldo total de cada conta”, para isso primeiro criei um argumento para preencher a combobox das contas

SqlCommand comando = new SqlCommand("Select IdConta, Conta From Contas", obj.con);

Cria o comando select

SqlDataReader reader = comando.ExecuteReader();

Retorna dados resultantes da consulta SQL

DataTable table = new DataTable();

Estrutura da tabela do BD ou de outra fonte

table.Load(reader);

Carrega dados na tabela de acordo com a instrução SQL passada

DataRow row = table.NewRow();

Representa uma linha de dados em um DataTable

row["Conta"] = "";

Para não gerar valor nulo e evitar erro de execução

contacomboBox.DataSource = table;

Atribui o objeto DataTable a instância do SQL Server a ser conectada

contacomboBox.ValueMember = "IdConta";

Obtemos o valor do membro (código da descrição)

contacomboBox.DisplayMember = "Conta";

Obtemos a descrição do membro

reader.Close();

Fecha objeto

reader.Dispose();

Libertamos o objeto da memória

E depois esse argumento é chamado no evento “load” do formulário

Em seguida dentro do evento “Click” do botão verificar tem várias consultas para obter o valor das despesas e das receitas, calculando assim o saldo atual em função da conta selecionada na combobox e em função também do utilizador que estiver a usar a aplicação

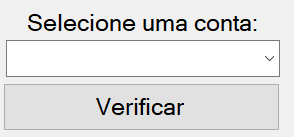


Figura 90 - ComboBox da Conta

Primeiro tem uma condição if para apenas entrar no ciclo se a opção que o utilizador selecionou na combobox da conta for diferente de 0, ou seja o utilizador se selecionar a primeira opção não vai acontecer nada

if (contacomboBox.SelectedIndex != 0)

Como os valores vão ser acrescentados na “label” no início tem sempre que definir o texto de cada “label”

contacomboBox.Text = "Na Conta: ";

TotalDespesaslabel.Text = "€ em Despesas";

TotalReceitaslabel.Text = "€ em receitas";

SaldoTotallabel.Text = "€";

Em seguida são criadas as variáveis com valor 0

int somaQntdDespesas = 0;

int somaQntdReceitas = 0;

int SaldoTotal = 0;

string contaselecionada="";

Todas as consultas estão dentro da condição try para no caso de existirem erros os reportar em uma “MessageBox” e para a aplicação não fechar

try

{

//comandos

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

E depois vai efetuar as consultas á base de dados, em que a primeira consulta é para saber qual é o id do utilizador para depois usar nas próximas consultas

string queryyy = "SELECT Idutilizador,Email FROM Utilizador WHERE Email LIKE '%" + SessaoSistema.EmailUsuario + "%' ";

SqlCommand cmd = new SqlCommand(queryyy, obj.con);

obj.con.Open();

SqlDataReader reader = cmd.ExecuteReader();

reader.Read();

string idutil = reader["Idutilizador"].ToString();

Vai adicionar á variável o id do utilizador

obj.con.Close();

Vai fazer uma consulta para obter o valor total das despesas

obj.con.Open();

string contDespesas = "select Sum(Quantia) From Gastos where IdConta ='" + (contacomboBox.SelectedIndex + 1) + "' And IdGasto in(select IdGasto From UtilizadorGastos where IdUtilizador='" + idutil + "')";

A consulta vai somar os valores todos do campo “Quantia” da tabela “Gastos” em que o “IdConta” é igual ao índex da “contacombobox” (pois estão por ordem por isso é possível obter o índex) e em que o “IdGasto” é igual a uma outra consulta que é o “IdGasto” na tabela “UtilizadorGastos” em que o IdUtilizador é igual ao que está a usar a aplicação

obj.cmd.Connection = obj.con;

obj.cmd.CommandText = contDespesas;

somaQntdDespesas = Convert.ToInt16(obj.cmd.ExecuteScalar());

O valor da soma é armazenado na variável “SomaQntDespesas”

obj.con.Close();

A consulta para obter as receitas é com o formato igual, só mudando os campos e as tabelas, que no caso são a “Receitas” e a “UtilizadorGasto”

obj.con.Open();

string contReceitas = "select Sum(Quantia) From Receitas where IdConta ='" + (contacomboBox.SelectedIndex + 1) + "' And Idreceita in(select Idreceitas From UtilizadorReceitas where Idutilizador='" + idutil + "')";

obj.cmd.Connection = obj.con;

obj.cmd.CommandText = contReceitas;

somaQntdReceitas = Convert.ToInt16(obj.cmd.ExecuteScalar());

O valor da soma é armazenado na variável “SomaQntReceitas”

obj.con.Close();

Como os valores ficaram armazenados nas variáveis falta agora introduzir nas respetivas “label’s”

NomeContalabel.Text = NomeContalabel.Text + " " + contacomboBox.Text;

A “label” do nome da conta vai tomar o valor do texto que já tem, introduzir mais um espaço e adicionar o texto da “contacombobox”

TotalDespesaslabel.Text = somaQntdDespesas + " " + TotalDespesaslabel.Text;

A “label” das depesas vai inserir o valor da variável, introduzir mais um espaço e tomar o valor do texto que já tem

TotalReceitaslabel.Text = somaQntdReceitas + " " + TotalReceitaslabel.Text;

A “label” das Receitas vai inserir o valor da variável, introduzir mais um espaço e tomar o valor do texto que já tem

SaldoTotal = somaQntdDespesas - somaQntdReceitas;

A variável do saldo vai subtrair a soma das receitas á soma das despesas para obter o saldo total

SaldoTotallabel.Text = SaldoTotal + " " + SaldoTotallabel.Text;

A “label” do saldo vai receber o valor da variável(o saldo), acrescentar um espaço e acrescentar o texto que já tinha

E o formulário apresenta também um gráfico com as despesas que fica dentro de um report viwer, e para isso é necessário instalar o componente

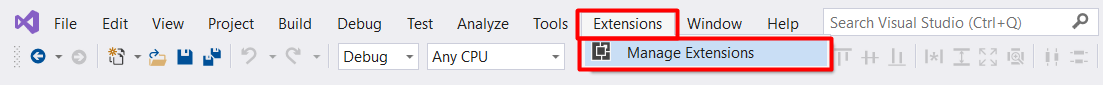


Figura 91 - Adicionar Extenções

No menu ir a “Extenções” e “Manage Extensions”;

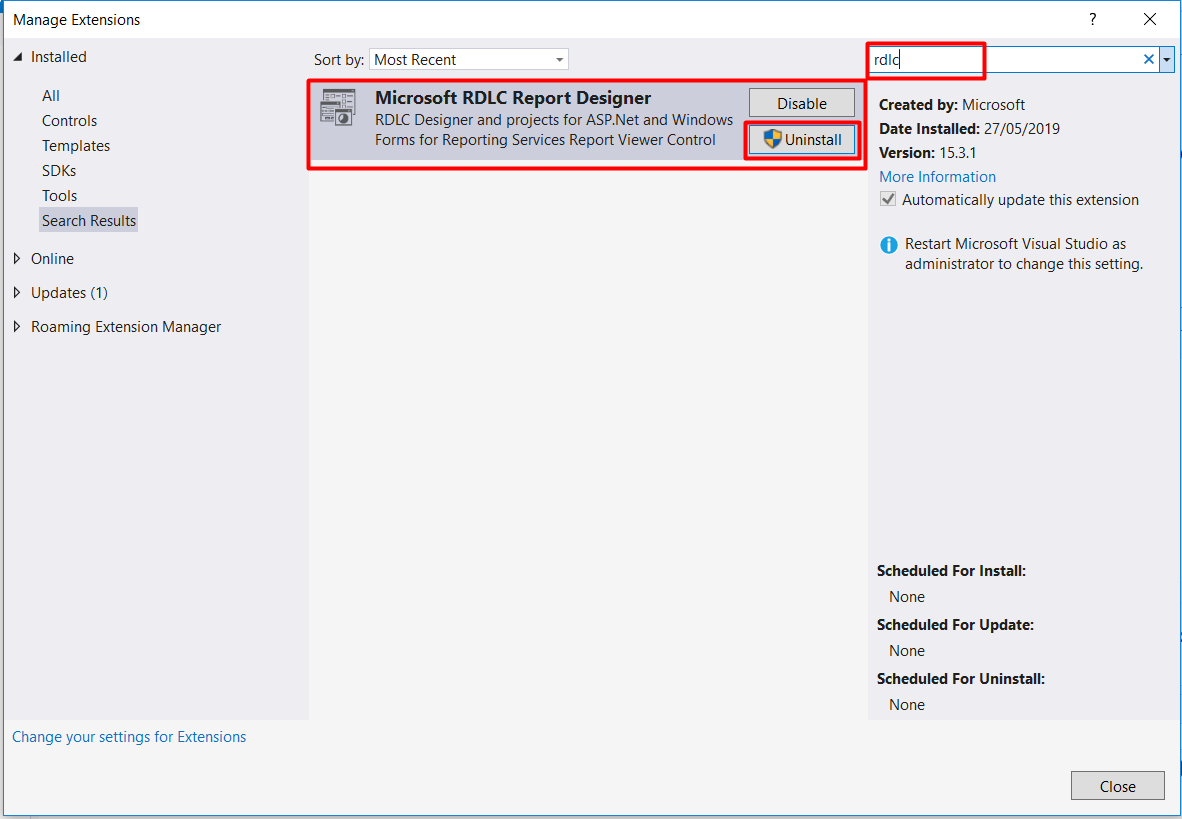


Figura 92 - Instalar rdlc

Na barra de pesquisa pesquisar “rdlc” e instalar o item “Microsoft RDLC Report Designer”, no caso o meu já está instalado;

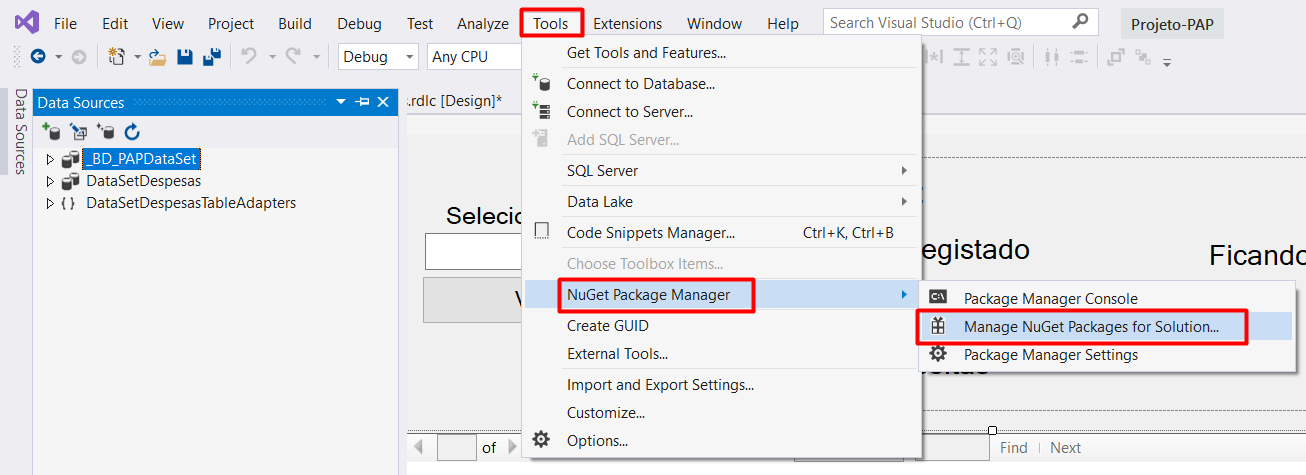


Figura 93 - Nuget Package Manager

No menu aceder a “Tools”, “Nuget Package Manager” e “Manage NuGet Packages for Soluction…”

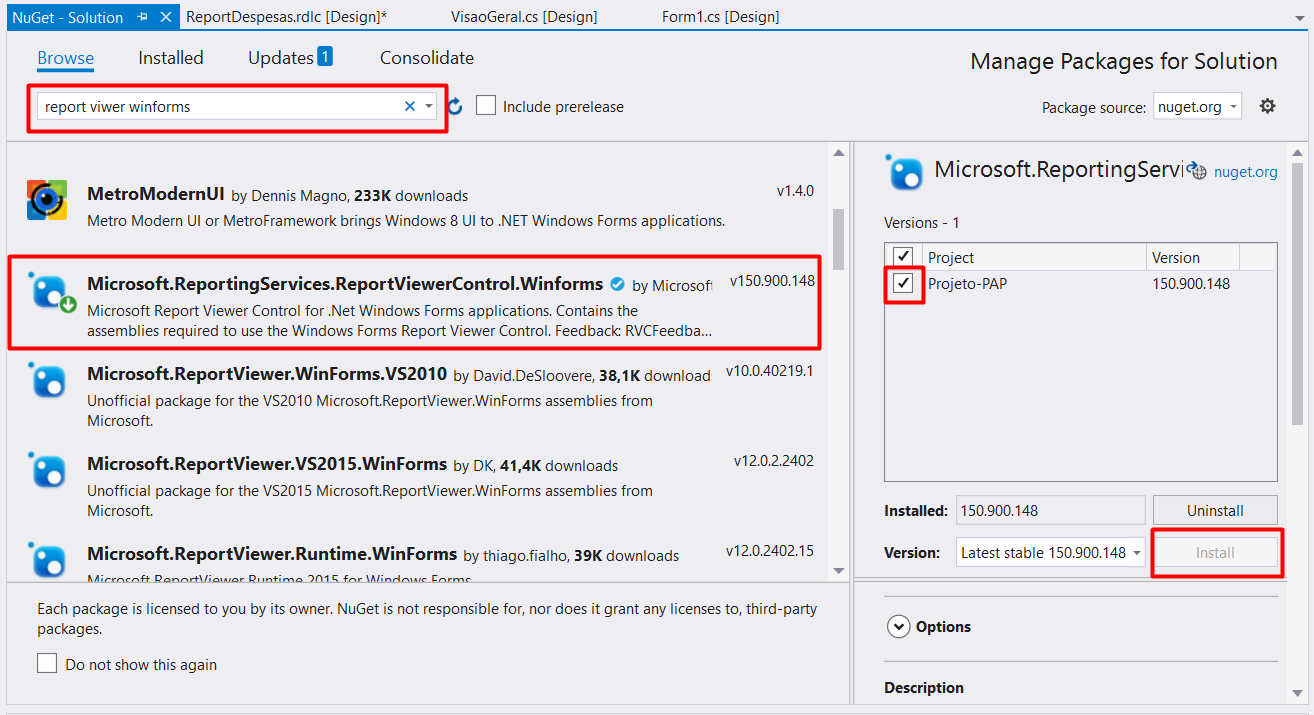


Figura 94 - Instalar o serviço de "reporting"

Em cima escolher “Browse”, na barra de pesquisa escrever “report viwer winforms”, escolher o item “Microsoft.ReportingServices.ReportViwerControl.Winforms”, ao lado selecionar o projeto e clicar em instalar

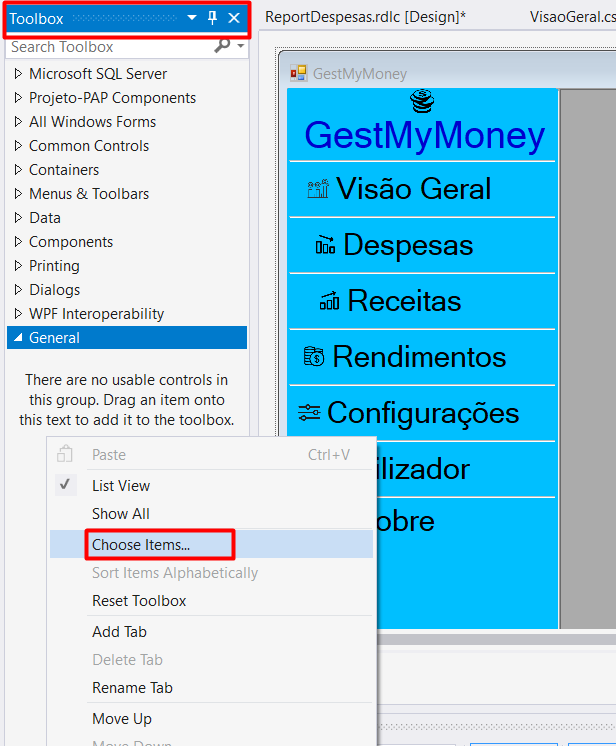


Figura 95 - Escolher itens

Na “toolbox” clicar com o botão direito e aceder a “Chose Items…”

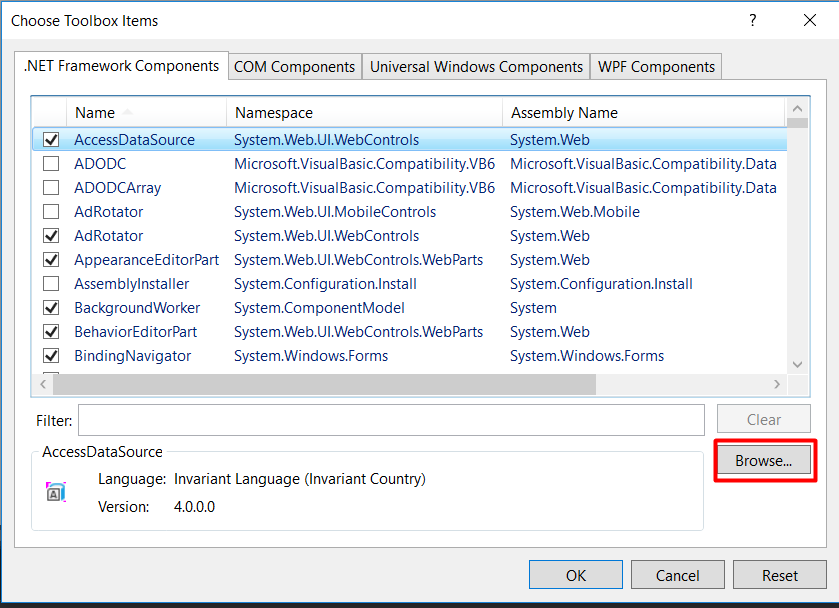


Figura 96 - Pesquisar item

Irá parecer uma caixa com todos os itens e é necessário ir ao botão “Browse”

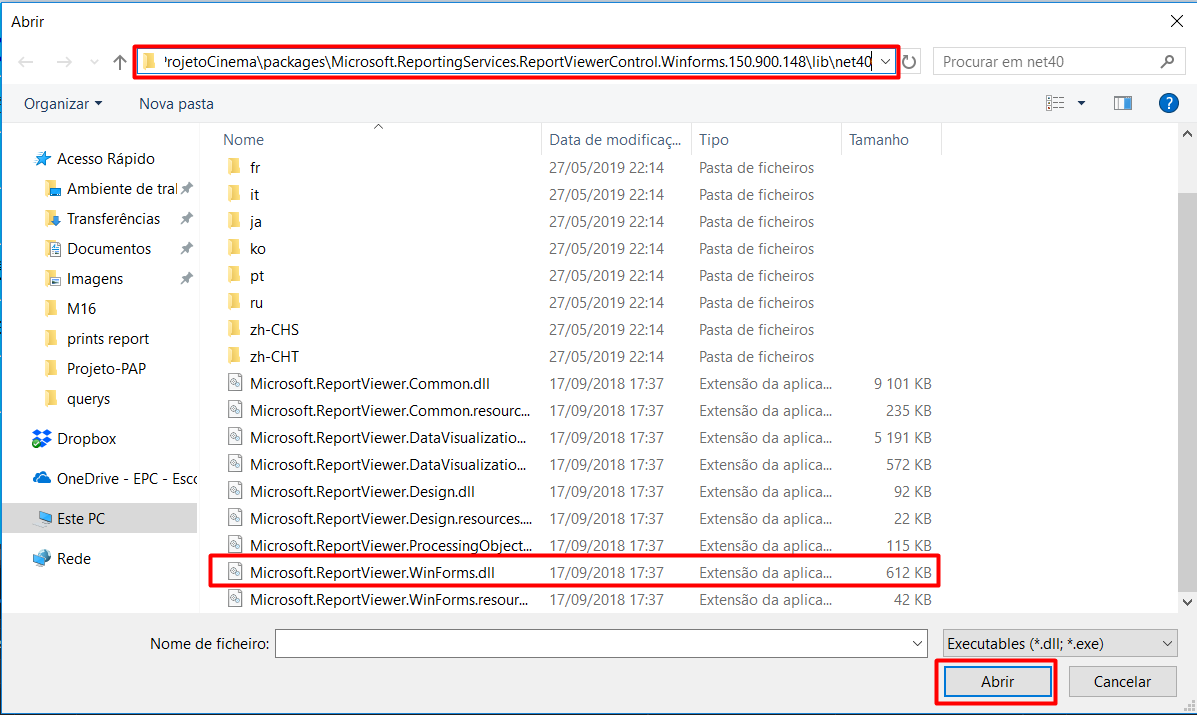


Figura 97 - Selecionar o item

Aceder á localização do ficheiro, dentro da pasta do projeto em “packages\Microsoft.ReportingServices.ReportViwerControl.Winforms.150.900.148\lib\net40”, selecionar o item “Microsoft.ReportViwer.Winforms.dll” e abrir

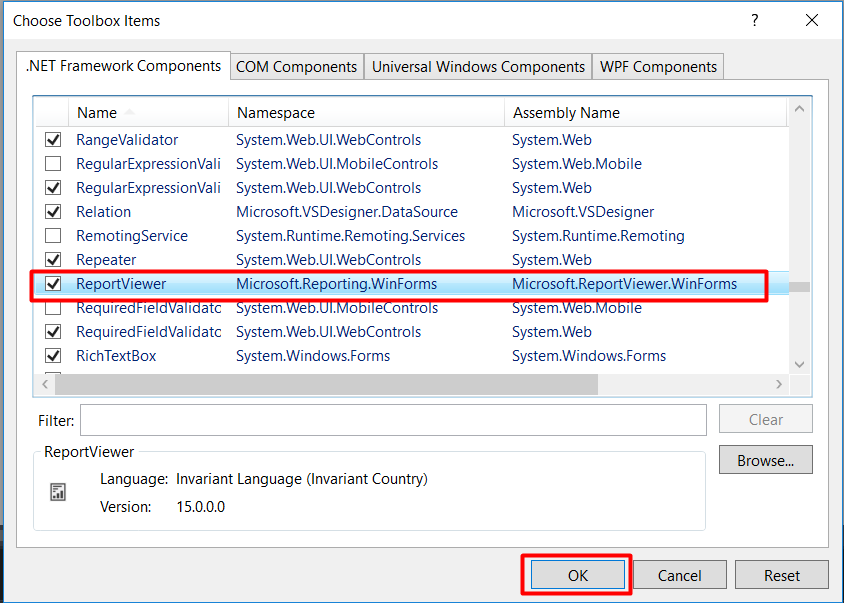


Figura 98 - Verificar o item na lista

Selecionar o item, e pressionar “OK”

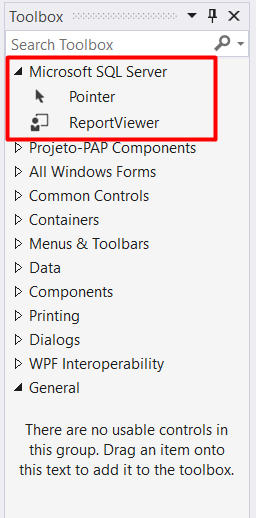


Figura 99 - "ReportViwer" na ToolBox

Por fim o “Report Viwer” aparecerá instalado na toolbox e pronto para ser usado

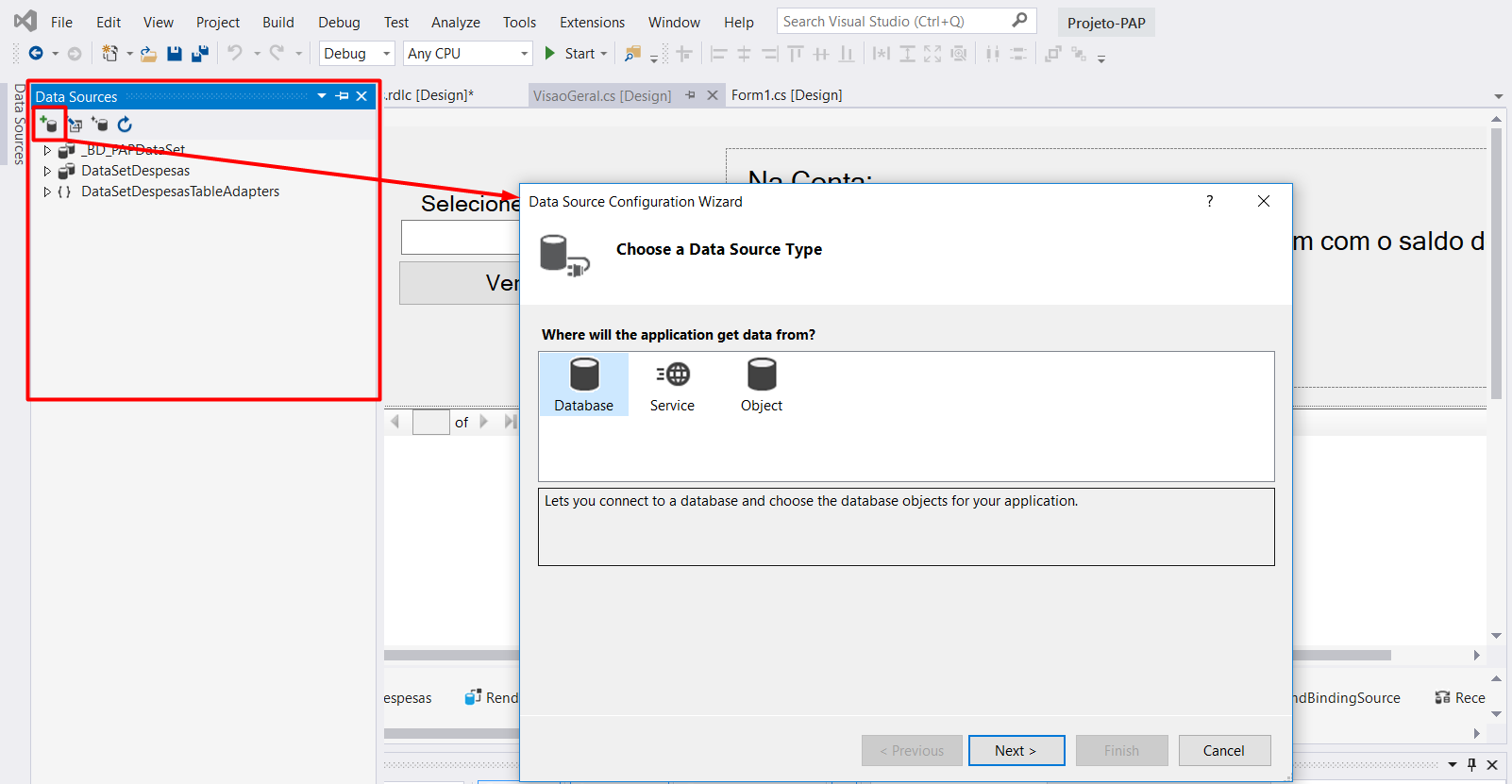


Figura - Adicionar Data Source

Em seguida é necessário criar um novo data source, para isso na janela “Data Sources” e no botão para adicionar novo “data source”

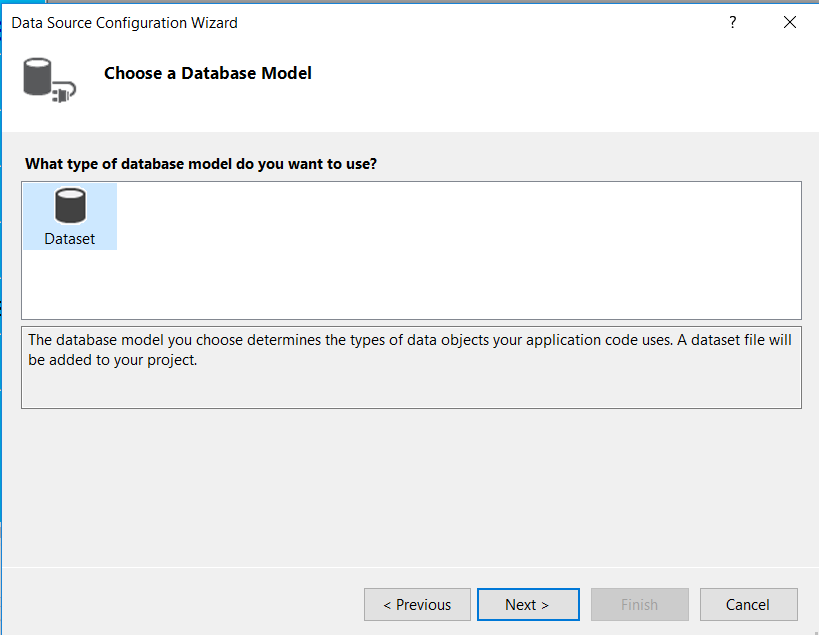


Figura – DataSet

Escolher a opção “DataSet”

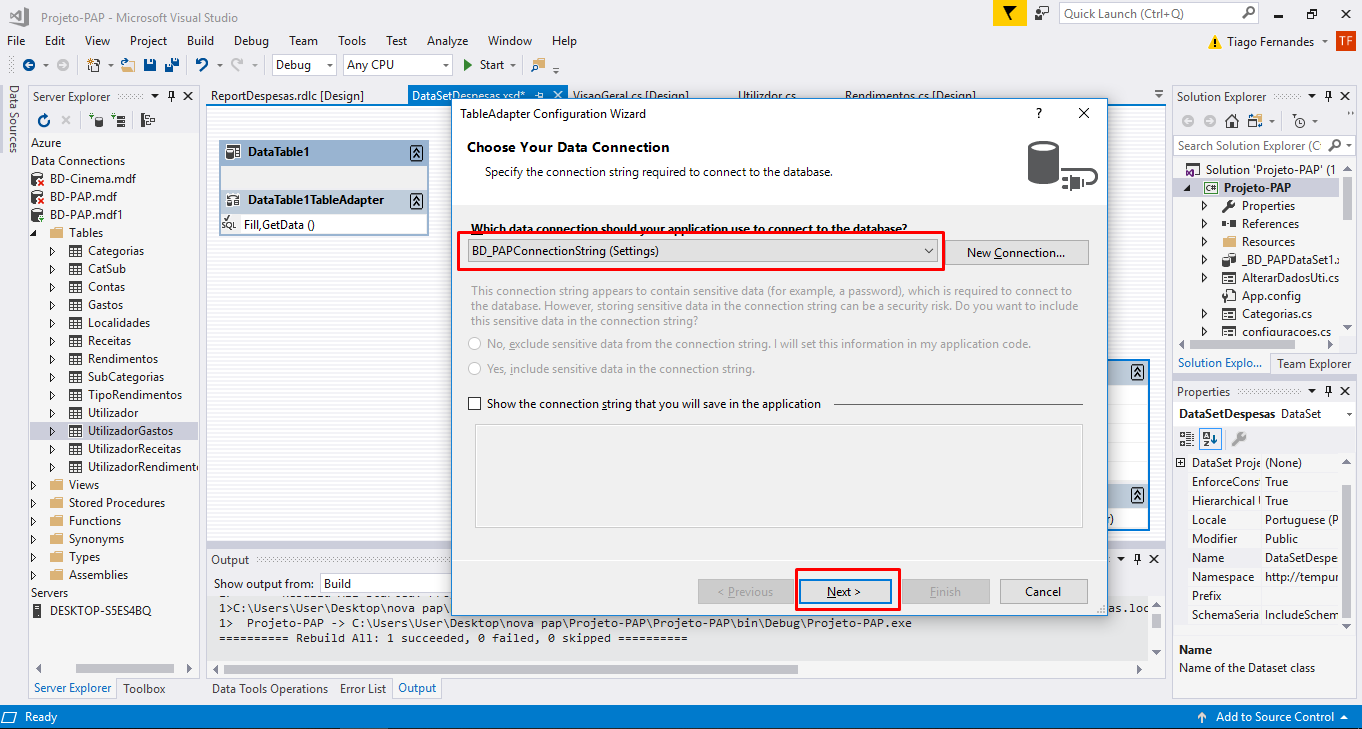


Figura - Escolher a conexão

Escolher a conexão á base de dados

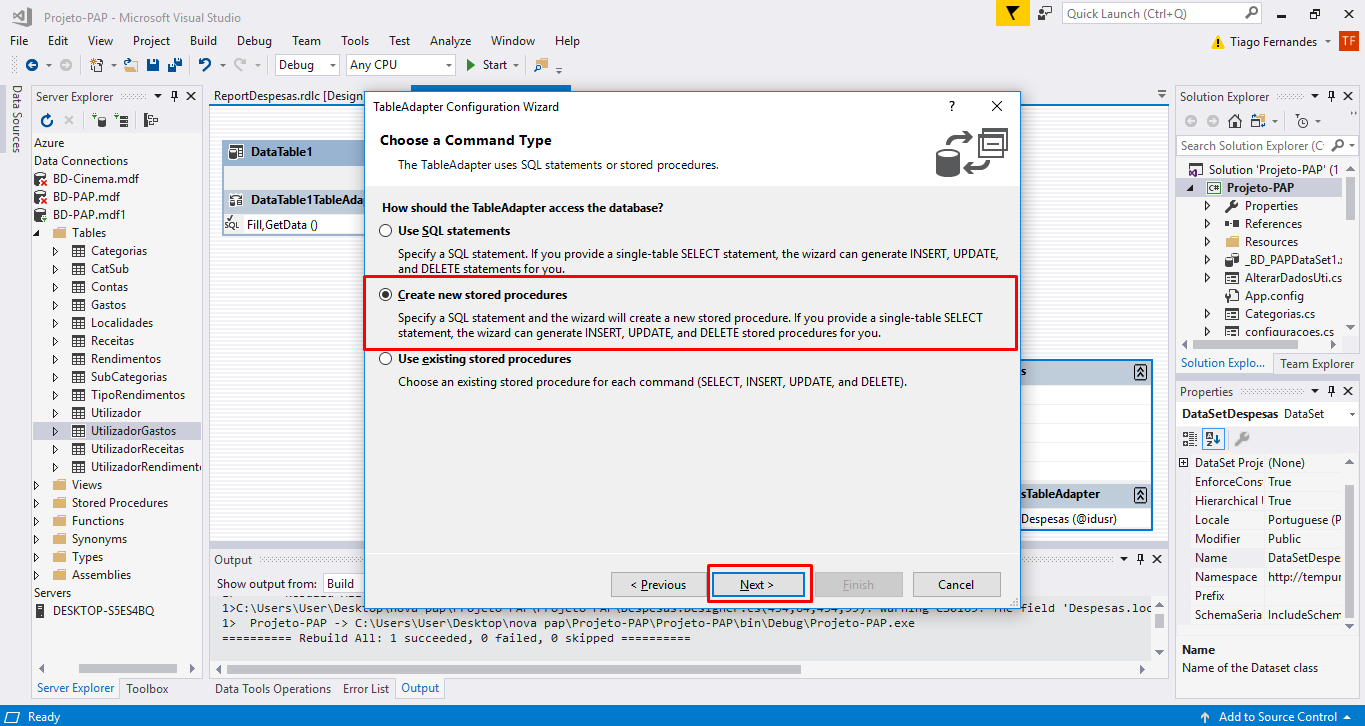


Figura - Criar novo procedimento

Selecionar a opção para criar novo procedimento

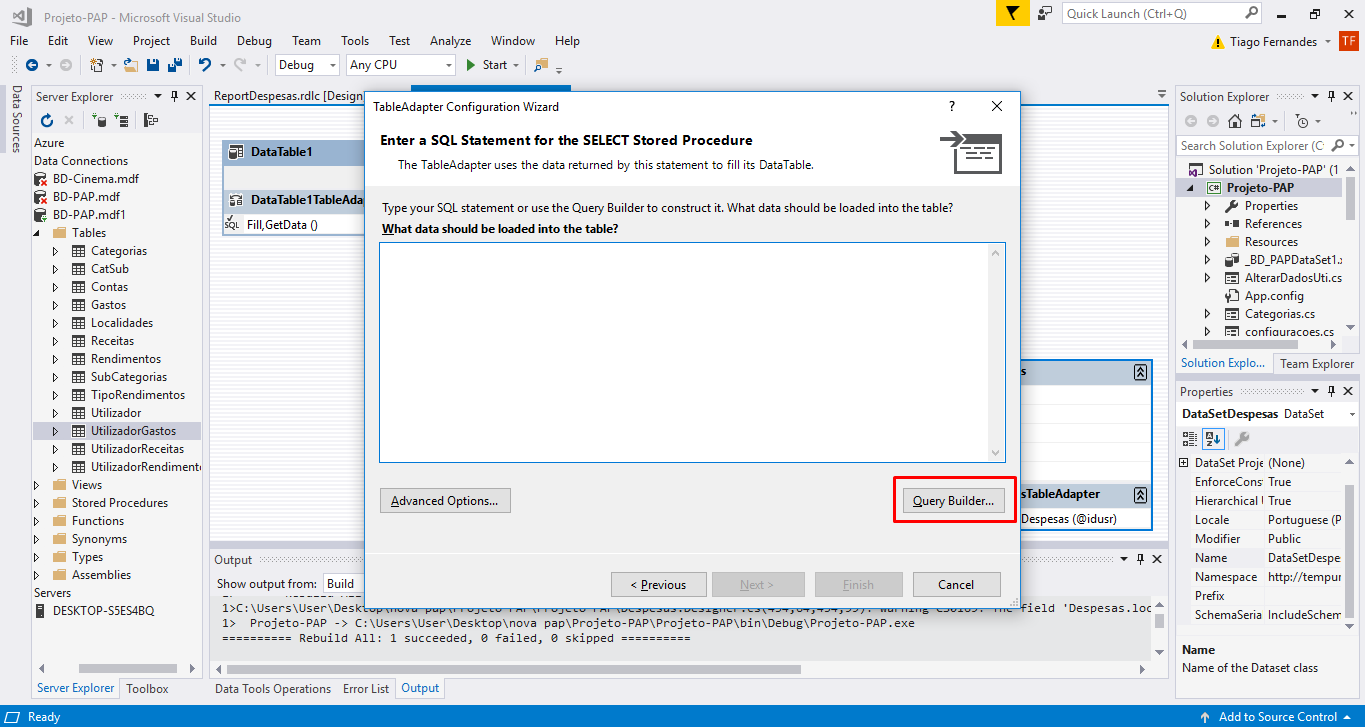


Figura - "Query Builder"

Usar o “Query Builder…”

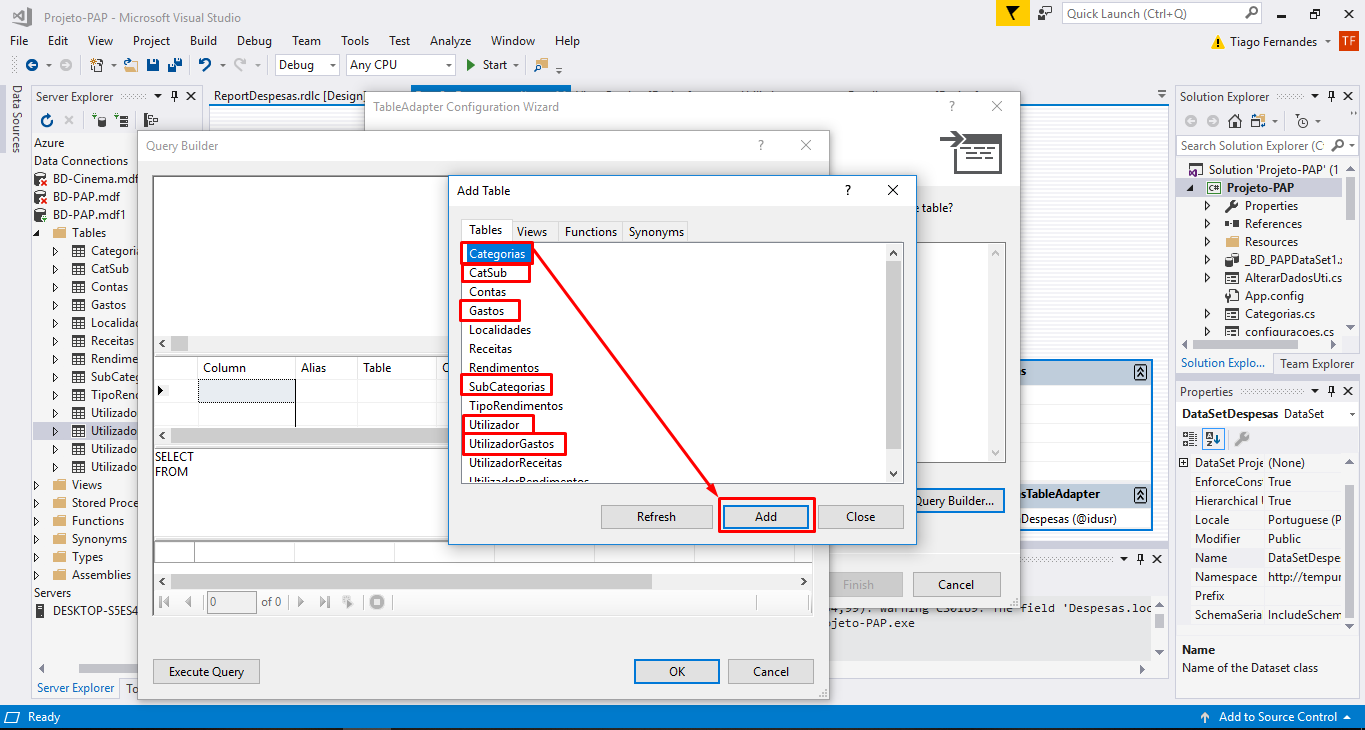


Figura - Adicionar tabelas

Irá ser aberta uma janela onde vamos selecionar as tabelas, uma de cada vez e clicar em adicionar



Figura – Relacionar tabelas dos Gastos

Com as tabelas “Categorias”, “CatSub”, “SubCategorias” e “Gastos” já é possível fazer as relações para obter as despesas em função das categorias, primeiro tem que se escolher os campos pretendidos, no caso é o “Categoria”, “Subcategorias”, “Lugar” e “Quantia” e automaticamente já são ser colocados no “select”

SELECT Gastos.Lugar, Gastos.Quantia, SubCategorias.Subcategorias, Categorias.Categoria, UtilizadorGastos.Data

FROM Gastos INNER JOIN

SubCategorias ON Gastos.IdSubCategoria = SubCategorias.IdSubCategorias INNER JOIN

CatSub ON SubCategorias.IdSubCategorias = CatSub.Idsubcategoria INNER JOIN

Categorias ON CatSub.IdCategoria = Categorias.IdCategoria

O “INNER JOIN” seleciona registos que possuem valores correspondentes nas duas tabelas., no caso o primeiro “Inner Join” vai consultar a tabela “SubCategorias” em que o campo “IdSubCategoria” dentro da tabela “Gastos” é igual ao campo “IdSubCategorias” da tabela “SubCategorias”

Mas como o programa possibilita a utilização de inúmeros utilizadores temos que relacionar a despesas com o utilizador, utilizando a tabela “UtilizadorGastos”

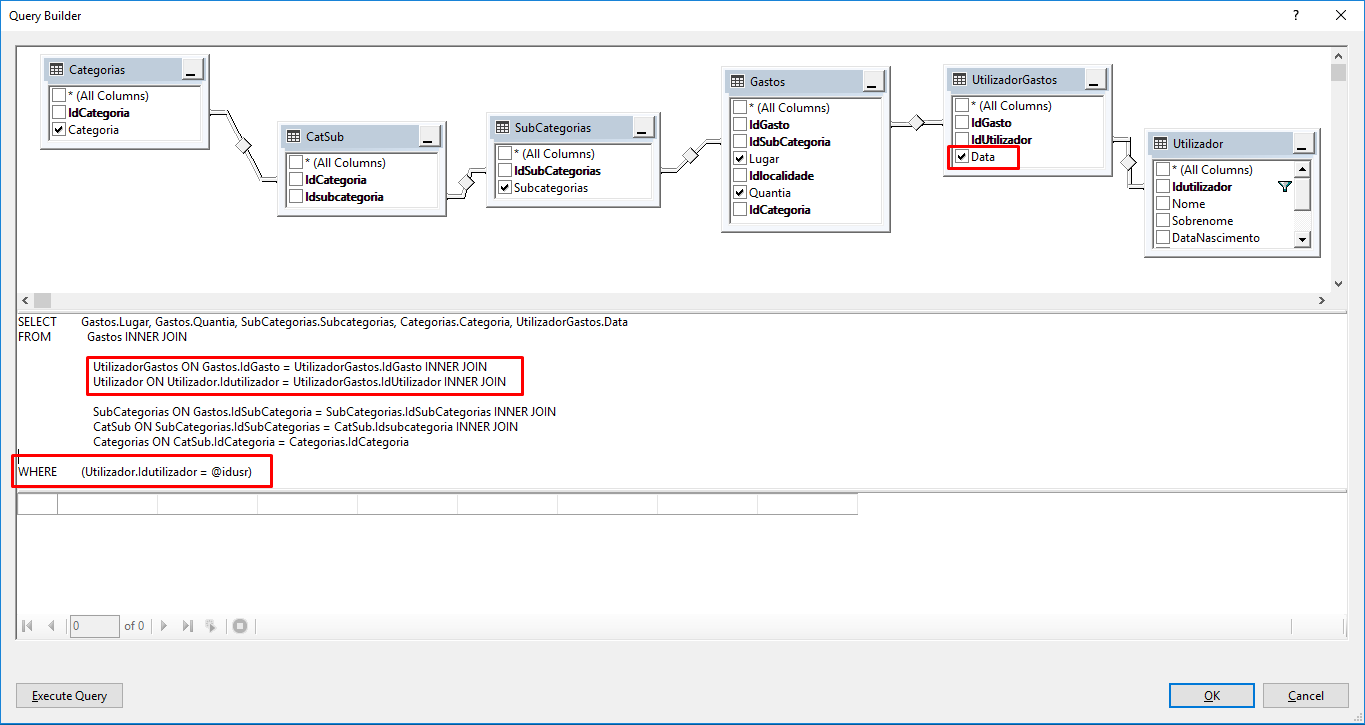


Figura - Relacionar o Utilizador

É necessário então adicionar mais as tabelas “Utilizador” e “UtilizadorGastos”, acrescentar mais dois “Inner Join”

UtilizadorGastos ON Gastos.IdGasto = UtilizadorGastos.IdGasto INNER JOIN

Utilizador ON Utilizador.Idutilizador = UtilizadorGastos.IdUtilizador INNER JOIN

E para escolher qual o id do utilizador pretendido usa-se a cláusula “Where” que é usada para extrair apenas os registros que atendem a uma condição especificada.

WHERE (Utilizador.Idutilizador = @idusr)

No caso quando o id do utilizador for igual ao introduzido

Uma imagem com captura de ecrã

Descrição gerada automaticamente

Figura - Executar a consulta

Em seguida executa-se a consulta, vai aparecer uma janela para introduzir o id do utilizador, que no caso para testar introduzi o número um

Uma imagem com captura de ecrã

Descrição gerada automaticamente

Figura - Confirmar os dados

E irão aparecer todos os registos que atendem á consulta especificada, com isto pronto vamos avançar

Uma imagem com captura de ecrã

Descrição gerada automaticamente

Figura - Avançar na configuração do Procedimento

Aparece novamente o código da consulta e é só avançar

Uma imagem com captura de ecrã

Descrição gerada automaticamente

Figura - Escolher o nome

Escolhe-se o nome do procedimento

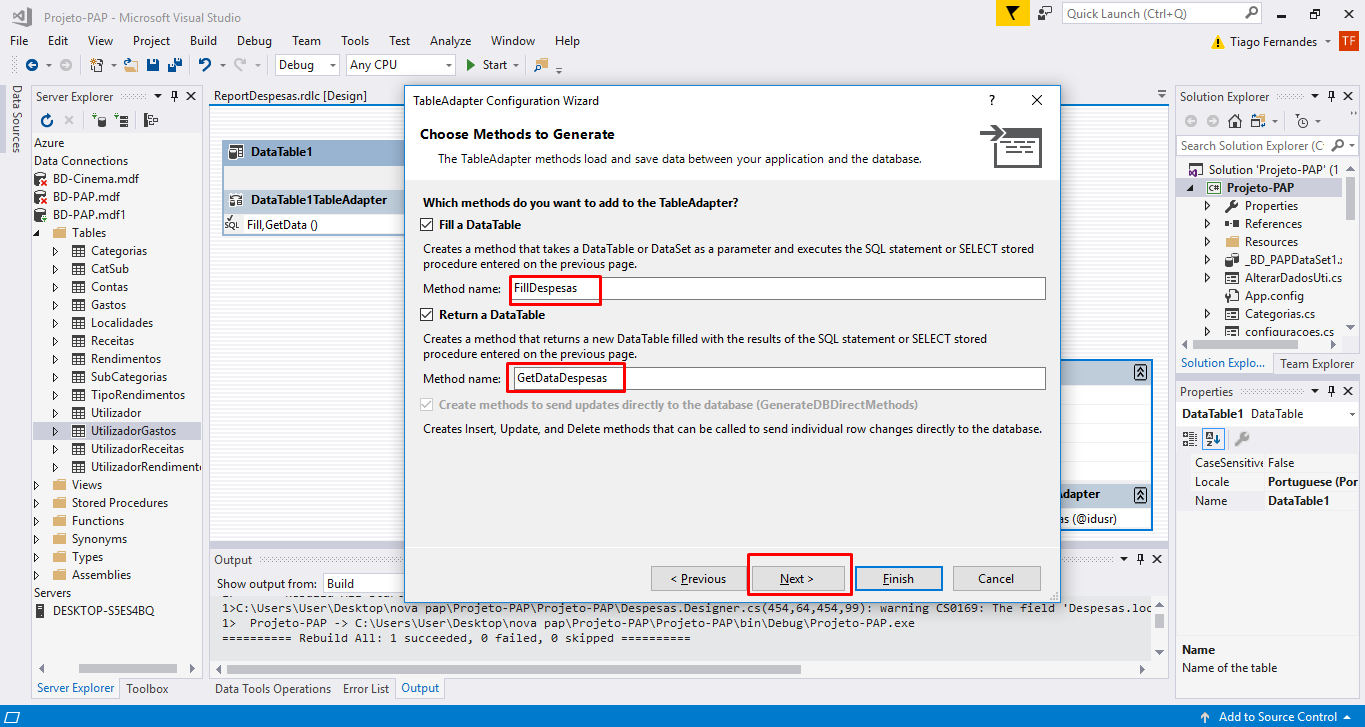


Figura - Nome do método

E o nome do método

Uma imagem com captura de ecrã

Descrição gerada automaticamente

Figura - Finalizar

Basta agora finalizar

Uma imagem com captura de ecrã

Descrição gerada automaticamente

Figura - Tabela "TblGraficoDespesas"

Aparece agora a tabela com os campos escolhidos e já alterei o nome da mesma para “tblGráficoDespesas”;

Com a consulta pronta falta agora criar o report

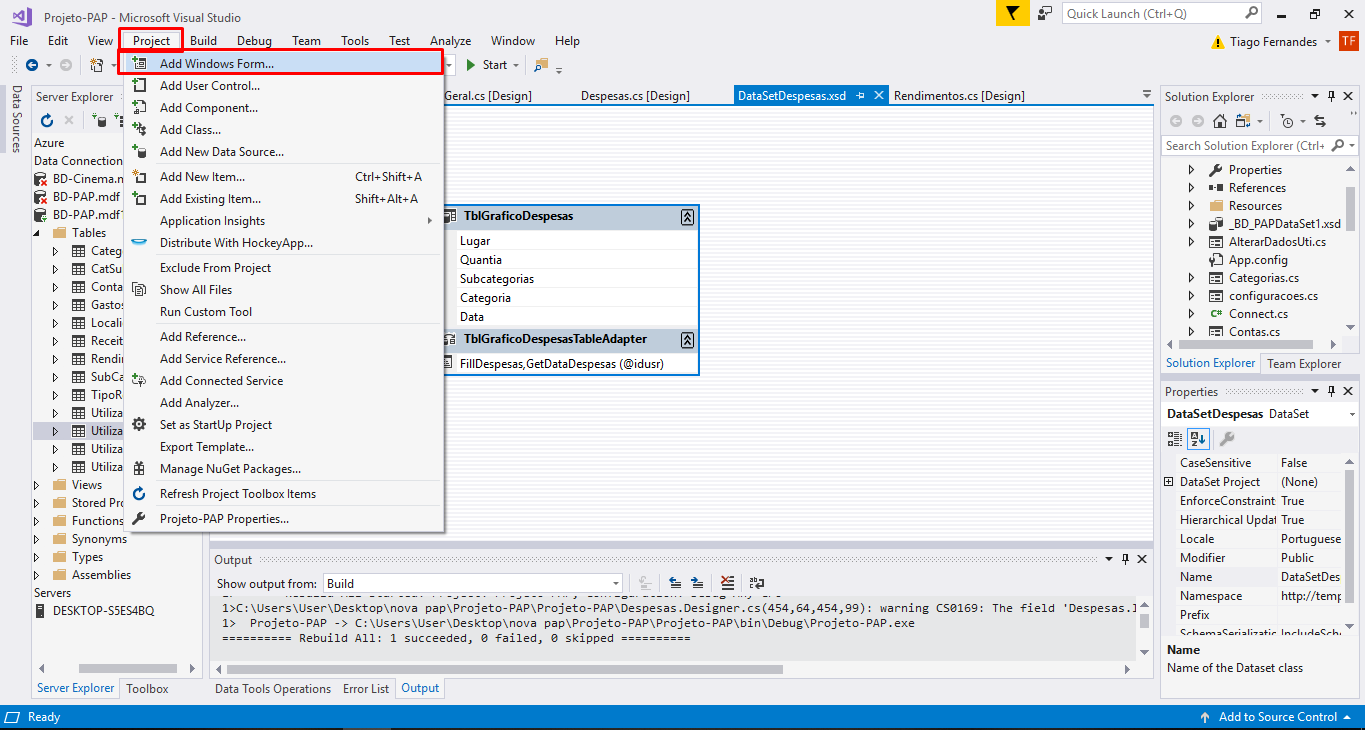


Figura - Adicionar “Windows form”

No menu do Visual Studio vamos a “Project”, “Add Windows Form…”

Uma imagem com captura de ecrã, interior

Descrição gerada automaticamente

Figura - Adicionar Report

No menu da esquerda seleciona-se “Visual C# Items” e ao centro seleciona-se “Report”, escolhe-se o nome basta adicionar

Uma imagem com captura de ecrã

Descrição gerada automaticamente

Figura - Nova “DataSet”

Na janela “Report Data” clica-se em “New”

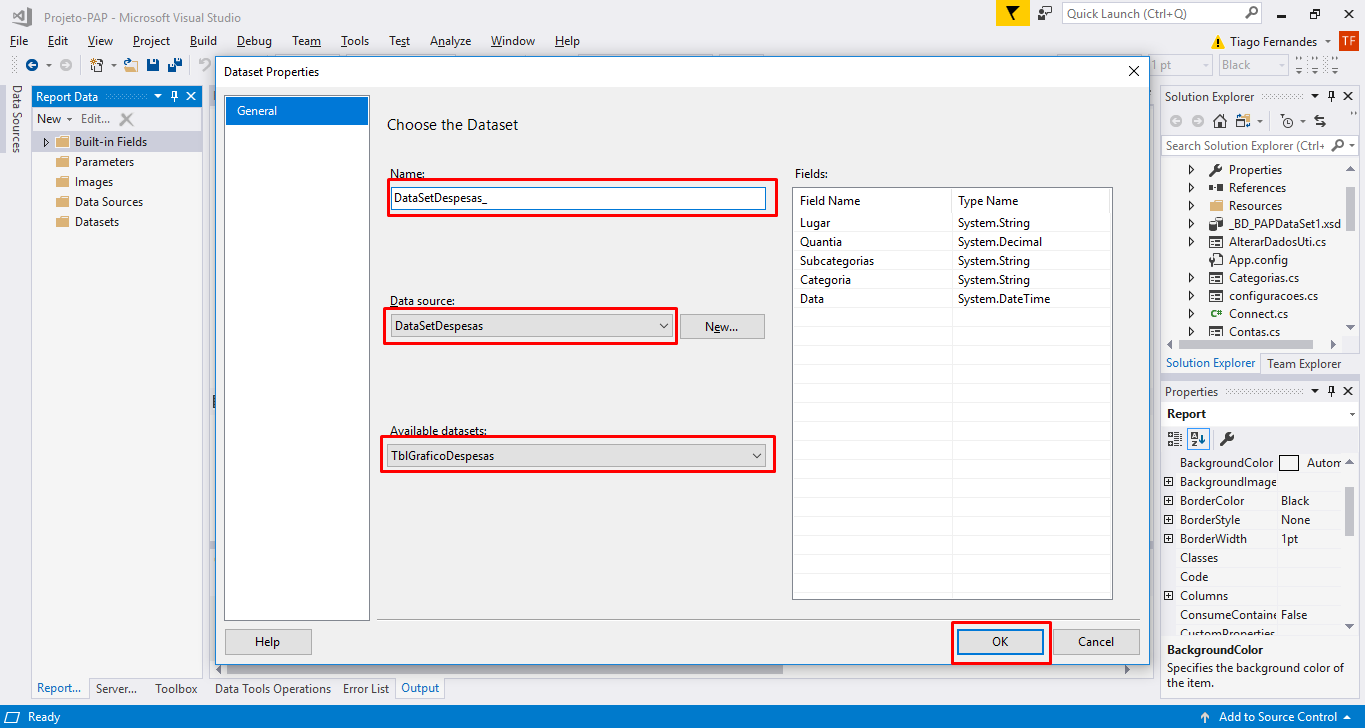


Figura - Campos da “DataSet”

Escolhe-se o nome, o data source a tabela

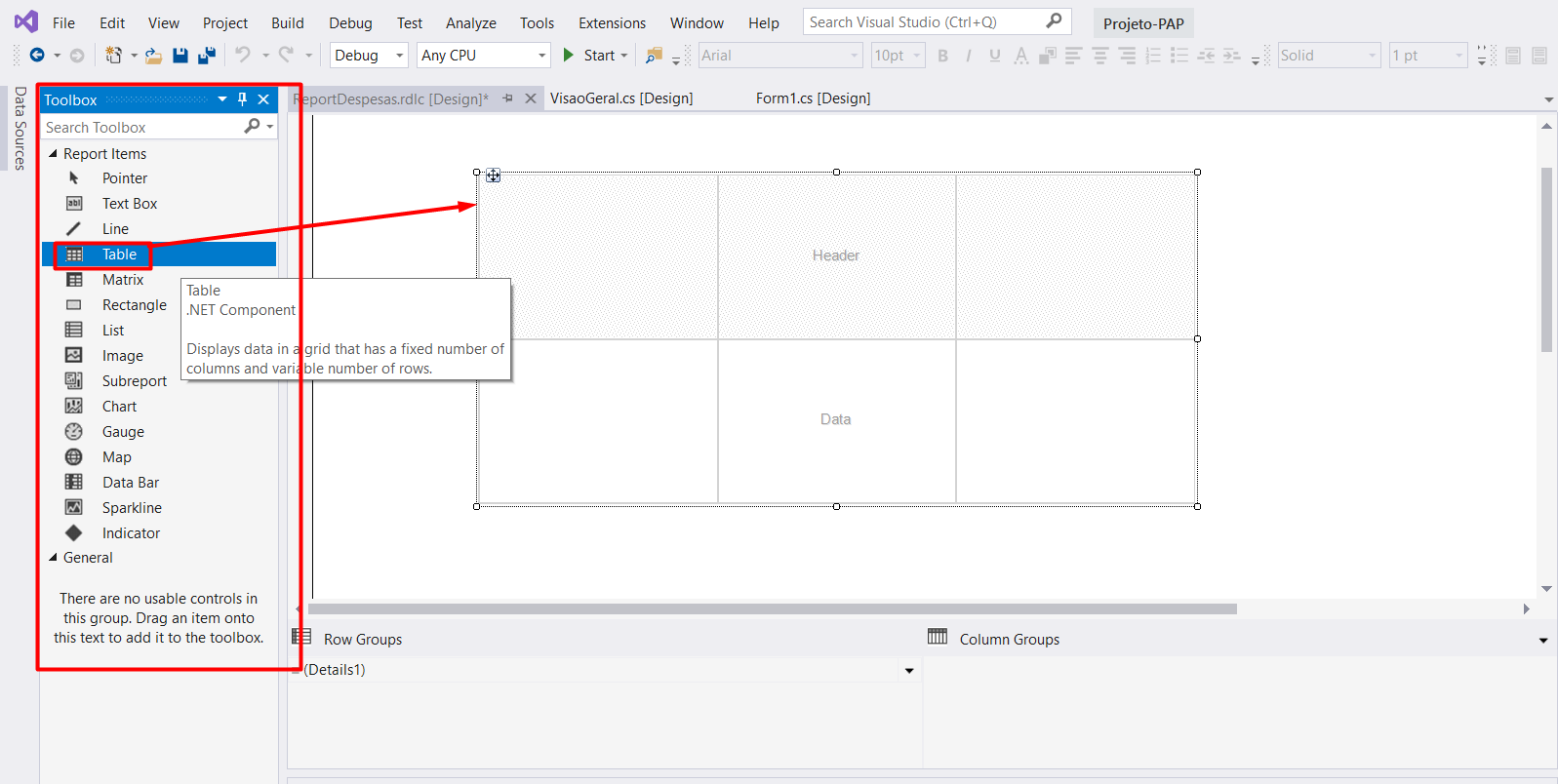


Figura - Adicionar Tabela

Na “ToolBox” arrasta-se o componente “Table” para o report

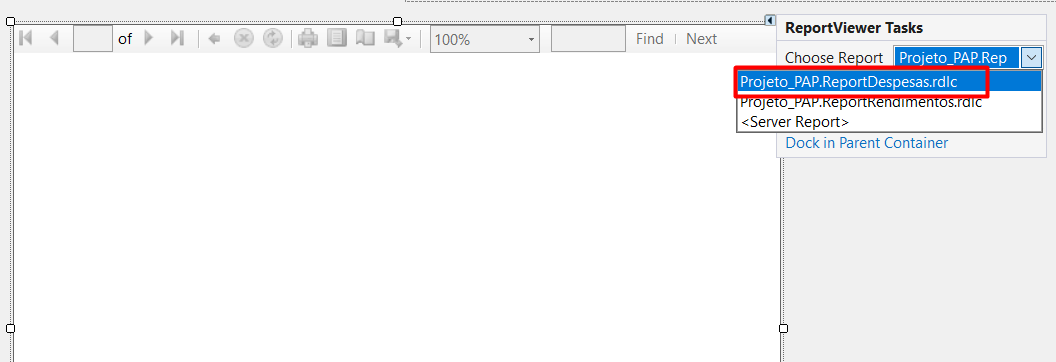


Figura - Escolher Report

Por fim basta adicionar o “report” ao “ReportViwer”

* 1. Formulário “Despesas”

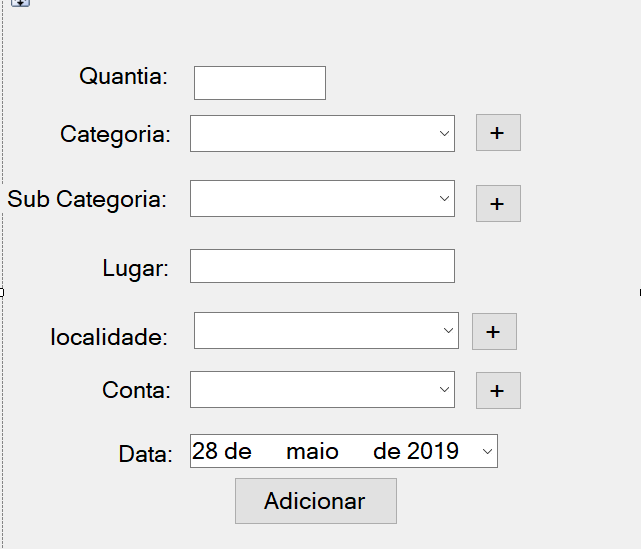


Figura - Formulário "Despesas"

O formulário despesas é serve para o utilizador registar as suas despesas, possui quatro combobox para preencher que são preenchidas cada um em um método e sempre com o mesmo formato de código, primeiro faz-se a ligação

private void preencheCategorias()

{

obj.con.ConnectionString = obj.locate;

obj.con.Open();

SqlCommand comando = new SqlCommand("Select IdCategoria, Categoria From Categorias", obj.con);

SqlDataReader reader = comando.ExecuteReader();

DataTable table = new DataTable();

table.Load(reader);

DataRow row = table.NewRow();

row["Categoria"] = "";

this.categoriaComboBox.DataSource = table;

this.categoriaComboBox.ValueMember = "IdCategoria";

this.categoriaComboBox.DisplayMember = "Categoria";

reader.Close();

reader.Dispose();

}

1. [Escreva aqui o título do 4º CAPÍTULO]

[Selecione estas frases e escreva aqui o texto para o 4º capítulo do corpo do relatório. Se precisar de criar mais capítulos:

1. Selecione toda esta página.
2. Copie.
3. Coloque o cursor antes da letra **C** das CONSIDERAÇÕES FINAIS da página seguinte.
4. Cole]

Considerações Finais

[Coloque aqui as considerações finais, conclusões, recomendações, etc.]

Bibliografia

Aplicação Boonzi. [s.n.] Boonzi® 2019. [Site]. Acesso:

<http://www.boonzi.pt/>

Site Tosh Finance. [s.n.] Toshl Inc. 2018. [Site]. Acesso:

<https://toshl.com/pt/>

Support Office: Noções básicas da Base de Dados. © Microsoft 2019. [Site]. Acesso:

<https://support.office.com/pt-pt/article/no%C3%A7%C3%B5es-b%C3%A1sicas-da-base-de-dados-a849ac16-07c7-4a31-9948-3c8c94a7c204>

Wiki Portugual-a-programar: Sistemas de Gestão de Bases de Dados. CC Attribution-Noncommercial-Share Alike 4.0 International 2018. [Site]. Acesso:

https://wiki.portugal-a-programar.pt/dev\_geral:bases\_de\_dados:sgbd

Loureiro, Henrique - C# 7.0 COM VISUAL STUDIO - CURSO COMPLETO. Edição 2017: Ed. FCA

Anexo A – [Seleccione isto e escreva aqui o título do anexo]

[Selecione esta frase e escreva aqui o texto para o anexo]

Anexo ? – [Seleccione isto e escreva aqui o título do anexo]

[Selecione esta frase e escreva aqui o texto para o anexo]