



Licenciatura em Informática

3ºano – Regime Diurno

Lisboa, 20 de Julho de 2014

Projeto Global

Utilizador virtual

Trabalho realizado por:

Pedro Gonçalves nº1881

Agradecimentos

Agradeço á minha família e namorada, por todo o apoio que me deram ao longo de toda esta fase universitária da minha vida.

Aos docentes do instituto, pelos conhecimentos transmitidos durante todo o curso. Um agradecimento especial aos que foram meus professores, colegas e a todas as funcionárias da faculdade, todos foram um bom apoio neste percurso.

Resumo

O trabalho apresentado visa aprofundar os conhecimentos adquiridos ao longo de todo este percurso, está englobado no projeto global de final de curso e vai servir para criar uma biblioteca em que cada utilizador vai ter a sua máquina virtual alocada numa nuvem, mas para podermos desenvolver este trabalho temos que aprofundar os nossos conhecimentos sobre as quatro tecnologias a si associadas, que são Virtualização,

cloud computing, ADO.NET e SQL,

A virtualização vai ser utilizada para virtualizar os sistemas operativos de cada utilizador, o cloud computing vamos utilizar na alocação das máquinas numa nuvem, que pode ser privada ou pública.

O ADO.NET vai ser utilizado para podermos fazer a interligação entre a aplicação que vamos criar e a Base de dados, que neste caso será em SQL, mas poderia não ser, em relação ao SQL vai ser a linguagem utilizada para a criação das base de dados, assim como para a sua alteração e eliminação.

Numa segunda parte do trabalho iremos ver como será feito todo o desenvolvimento deste trabalho, e de seguida vê-lo a trabalhar com todas as tecnologias utilizadas.

Palavras-chave: Virtualização, Máquinas Virtuais, Sistemas operativos, Computação em nuvem, Linguagem de programação, Base de dados

Abstract

The presented work aims to deepen the knowledge acquired throughout this route is encompassed in the global project final course and will serve to create a library in which each user will have their allocated virtual machine in a cloud, but in order to develop this work we have to deepen our knowledge of the four technologies associated with them, which are Virtualization, cloud computing, ADO.NET and SQL, Virtualization will be used to virtualize operating systems each user, the cloud will use in the allocation of machines in a cloud that can be private or public.

The ADO.NET will be used in order to make the link between the application that we create and the database, which in this case will be in SQL, but could not be compared to SQL will be the language used to create the base data, as well as for its modification and disposal.

In a second part we will see how it will be done throughout the development of this work, and then see it work with all technologies used.

Keywords: Virtualization, Virtual Machines, Operating Systems, Cloud Computing, Programming Languages, Database

Índice

Resumo	3
Abstract.....	4
Índice de figuras	Erro! Marcador não definido.
Introdução	8
Objectivo	9
Gerais	9
Especificos.....	9
Estado da Arte	10
Virtualização	10
O que é a virtualização	10
O que é a virtualização nos dias de hoje	11
Vantagens	12
Desvantagens.....	14
Cloud Computing	16
História.....	16
Cloud Computing nos dias de hoje.....	16
Vantagens	18
Desvantagens.....	18
ADO.NET	19
História.....	19
ADO.net no seu estado atual	19
Vantagens	21
Desvantagens.....	21
SQL	21
História.....	21
SQL nos dias de hoje	22
Desenvolvimento Biblioteca Virtual	25
Pré-requisitos.....	25
Estrutura da aplicação	26
Modelo relacional.....	26
Construção da aplicação.....	29
Maquinas Virtuais.....	43
Conclusão	45
Bibliografia	46

Anexos.....	47
Pagina Utilizador.....	47
XAML.....	47
C#.....	48
Pagina Livros	52
XAML.....	52
C#.....	53
Pagina Editoras	60
XAML.....	60
C#.....	61
Página Autores.....	67
XAML.....	67
C#.....	68

Índice de figuras

Figura 1	página 26
Figura 2	página 27
Figura 3	página 27
Figura 4	página 28
Figura 5	página 28
Figura 6	página 29
Figura 7	página 31
Figura 8	página 31
Figura 9	página 32
Figura 10	página 33
Figura 11	página 34
Figura 12	página 35
Figura 13	página 36
Figura 14	página 37
Figura 15	página 38
Figura 16	página 39
Figura 17	página 40
Figura 18	página 41
Figura 19	página 42
Figura 20	página 42
Figura 21	página 43
Figura 22	página 43

Introdução

Este trabalho visa aprofundar os conhecimentos sobre virtualização, cloud computing, ADO.NET e SQL. Para isso vai ser feito um estudo científico sobre cada um dos temas, para que possamos perceber e compreender melhor o trabalho que vai ser desenvolvido. A virtualização de sistemas operativos tem ganho muito espaço no mundo da tecnologia nos últimos anos, em que a sua principal função é a partir de um dispositivo de *hardware* termos várias máquinas virtuais em que cada uma pode ter o seu sistema operativo, mesmo que um seja Windows, outro Mac Os ou Linux. Esta evolução permitiu aos utilizadores tanto na ótica empresarial como residencial baixarem os seus custos ao nível do *hardware*. O objetivo deste trabalho é perceber o que a virtualização trouxe aos utilizadores e às empresas que hoje em dia vivem muito da mesma, quais as vantagens e desvantagens agregadas a virtualização. Vamos também apresentar as outras três tecnologias associadas a este trabalho, que é o *cloud computing*, ADO.NET e SQL, em relação ao *cloud computing* vamos estudar a sua evolução histórica, como pode ser utilizado quanto aos seus serviços, e quais as arquiteturas utilizadas na sua implementação, assim como quais as suas vantagens e desvantagens.

No ADO.NET vamos perceber o porquê desta linguagem ter sido criada e com que funcionalidade, estudar como pode ser utilizada na ótica do programador, em relação á sua fácil ligação com a base de dados, que trabalha e atualiza os dados em ambiente conectado, mas que também podemos manipular os dados em ambiente desconectado, sendo essa a sua grande vantagem. No SQL, vamos conhecer melhor este tipo de linguagem de base de dados, quanto ao seu padrão e ao seu desenvolvimento, como é feito o backup, segurança e qual a sua necessidade de utilização nos dias de hoje.

No estado da arte vamos aprofundar o conhecimento sobre a área que vamos desenvolver no projeto global. Quanto á sua contextualização vai ter a criação de uma biblioteca em que cada utilizador vai ter a sua própria máquina virtual, alocada numa nuvem e poder através da sua conta de utilizador, consoante as permissões que forem dadas, fazer uma determinada pesquisa sem que para isso tenha o que quer que seja alocado na sua máquina real.

Na outra parte em que se divide o trabalho iremos ver como foi criado todo o projeto quanto á sua implementação prática, onde irá ser criada a aplicação, as base de dados e todo o laboratório virtual.

Para o Desenvolvimento deste projeto na sua componente prática iremos usar as seguintes ferramentas:

- Microsoft Visual Studio 2013 para utilizarmos a linguagem C# e a criação da *window wpf application*
- SQL Server 2014 Management Studio para a estrutura de Base de dados
- *VMware Workstation 11* para a criação das máquinas de forma a podermos simular a biblioteca virtual com a criação das diversas máquinas

Objectivo

Gerais

O que se pretende com o desenvolvimento do projeto é a criação de uma aplicação em desktop, que possa ser acedida pelos utilizadores e administradores através de sistemas operativos virtualizados, permitindo aos utilizadores consultar a biblioteca virtual, descarregar os ficheiros em formato .pdf, e as credenciais de administrador vão ter a possibilidade de alterar, inserir, eliminar e atualizar.

Específicos

- a) Permitir ao utilizador aceder ao sistema operativo virtualizado, colocando as suas credenciais, e poder consultar ou alterar a Base de dados consoante as suas permissões, utilizando as contas de utilizadores criadas no *Active Directory*
- b) Dar a possibilidade ao utilizador de pesquisar pelos vários campos de pesquisa, existentes na aplicação.
- c) Permitir que o Administrador possa editar, inserir e remover em três página diferentes, isto é, na página de livros, na página de autores e na página de editoras.
- d) Exportar o conteúdo da pesquisa para formato de .pdf.

Estado da Arte

Virtualização

O que é a virtualização

Segundo Juciely Rodrigues(2011) no final da década de 50 já existia relatos da virtualização com o Time-Sharing-System(TSS), através de um computador vários utilizadores podiam aceder em simultâneo.

Contudo Marcos Laureano(2006) defende que o conceito de máquina virtual não é novo começou no inicio de 1960, as máquinas virtuais inicialmente foram desenvolvidas para centralizar os sistemas de computador utilizados no ambiente VM/370 da IBM, este sistema servia para que cada máquina virtual simulasse uma réplica física da máquina real, e assim os utilizadores tinham a sensação que aquele *hardware* era exclusivo para si mesmo, assim usufruía-se melhor do *hardware*, que na época tinha elevados custos. Vanderlei Pollon(2008) cita que um grupo de pesquisadores da IBM precisava de fazer testes, então foi criada uma forma de dividir as máquinas. Assim sendo, estas máquinas geriam os seus próprios recursos, possibilitando os pesquisadores de fazerem testes em diferentes condições de uso, sem alterar outras partes do sistema, mas naquela altura esta plataforma demorou muito tempo a ser lançado. A versão TSS/ 360, quando chegou, era um sistema robusto quanto ao seu funcionamento. Logo após o fracasso do TSS/360, no inicio dos anos 70, a IBM desenvolveu o CP/CMS, que mais tarde ficou conhecido como VM/370.

Na década de 80, a virtualização perdeu um pouco a importância devido ao elevado uso dos computadores pessoais, pois ficou mais barato ter um computador para cada utilizador, do que fazer um elevado investimento para grandes sistemas de computação. Mais tarde, na década de 90, com o aparecimento da linguagem Java e os sistemas de *hardware* com mais funcionalidades, regressam os interesses pela virtualização. Sendo que esta tecnologia ganhou a sua grande força por via do VMware, fundada em 1998, empresa responsável por ter tornado a virtualização cada vez mais popular. Após o aparecimento da VMware, outras empresas foram criando ferramentas de virtualização.

O que é a virtualização nos dias de hoje

Hoje em dia a virtualização tem uma utilização muita vasta, isto porque podemos ter virtualização de sistemas operativos, virtualização de servidores, virtualização de aplicações, paravirtualização, virtualização total, sendo que nos dias de hoje é mais utilizada a virtualização de servidores e de sistemas Operativos.

A virtualização de sistemas operativos é utilizada no dia-a-dia tanto por empresas como pelos clientes residenciais para virtualizar diversas versões de sistemas operativos, enquanto a virtualização de servidores é mais utilizado no âmbito de clientes empresariais.

Segundo Alexandre Carissimi(2009) quando implementamos uma máquina virtual temos que considerar dois pontos de vista, a de um processo e a do núcleo do sistema operativo, sendo que uma máquina virtual de processo fornece ao utilizador um ambiente de execução para uma única aplicação através de uma interface virtual(ABI) a qual é mapeada para uma interface real composta pelas chamadas de sistema e pelo conjunto de instruções não-privilegiadas(*user ISA*), podemos olhar para isto na prática como sendo um processo chamado de aplicação a ser executado dentro de um outro processo, a máquina virtual, este tipo de máquina virtual só existe enquanto a aplicação estiver a ser executada.

Quanto ás máquinas virtuais de sistema, conhecidas por VMM(*Virtual Monitor Machine*) ou Hypervisor, é o que nos oferece um ambiente de execução completo onde existe os sistemas operativos e as aplicações, este tipo de máquinas virtuais estão ativas até existir a ordem para serem paradas.

Existem dois tipos de *hypervisors*, os nativos que são do tipo I e os hóspedes que são do tipo II, em que os primeiros são uma camada de *software* colocada entre o *hardware* físico e as máquinas virtuais, e o do tipo II são executados sobre o sistema operativo como se fossem um processo deste. Temos ainda duas técnicas de *hypervisors*, que é a virtualização total e a paravirtualização em que na primeira o sistema operativo hóspede não precisa de ser modificado para ser executado sobre o *Hypervisor*, enquanto na segunda técnica o sistema operativo hóspede tem que ser modificado para ser executado sobre o *hypervisor*.

Hoje em dia as grandes empresas como a AMD e a Intel tem investido mais em processadores de *hardware* para suportar com melhor *performance* a virtualização.

Alexandre Carissimi (2009) defende que podemos classificar a virtualização em três grandes categorias:

- Nível de hardware: é quando a camada de virtualização é colocada diretamente sobre a máquina física e a apresenta às camadas superiores como um *hardware* abstrato similar ao original. Como falamos na história da virtualização, esta categoria corresponde ao tipo de virtualização utilizada na década de 60.
- Nível do sistema operativo: permite ao utilizador criar partições lógicas numa plataforma de maneira a que cada partição seja vista como uma máquina isolada, mas que partilha do mesmo sistema operativo. Aqui a camada de virtualização encontra-se entre o sistema operativo e as aplicações. Como exemplos temos o Linux Vserver, OpenVZ entre outros.
- Nível de linguagens de programação: neste caso a camada de virtualização é um programa de aplicação do sistema operativo. Tem como objetivo definir uma qualquer máquina sobre a qual está a ser desenvolvida uma aplicação numa linguagem de programação de alto nível, temos como exemplo a máquina virtual Java e a *Microsoft Common Language Infrastructure*.

Contudo para Vanderlei Pollon(2008) a virtualização pode ser dividida em oito grandes categorias: virtualização de sistema operativos, virtualização do servidor de aplicação, virtualização de aplicação, virtualização de gestão, virtualização de rede, virtualização de *hardware*, virtualização de *storage* e virtualização de serviço, mas sendo a virtualização um tema tão vasto e sempre em constante mudança existem muitas opiniões sobre as grandes categorias em que podemos dividir a virtualização, porque também tem muito a ver com a utilização que é dada por parte do utilizador residencial e empresarial.

Vantagens

Segundo Vanderlei Pollon(2008) as vantagens da virtualização são:

- Consolidação de servidores, isto é, podemos migrar servidores físicos distribuídos para partições lógicas ou máquinas virtuais de um ou mais servidores, com maior porte e maior capacidade de escalabilidade, fazendo assim que não se desperdice recursos de *hardware* não utilizados;

- Gestão centralizada de servidores, permite-nos gerir de forma centralizada os servidores e podemos mover dispositivos virtuais, como memória RAM, capacidade de processamento, espaço em disco, dispositivos e portas, de uma máquina virtual para a outra, em caso de falta ou esgotamento dos mesmos, e é possível configurar esta gestão de recursos cujo o serviço seja considerado mais importante.
- Agilidade na recuperação de desastres, por exemplo se considerarmos uma máquina virtual como um ficheiro de arquivo em caso de algum “desastre” será só fazer uma cópia da máquina virtual para outro *hardware*.
- Redução no consumo de energia dos *datacenters*, sendo que os *datacenters* têm um elevado consumo de energia em todo o seu *hardware* a virtualização foi a solução encontrada para ultrapassar esse problema porque possibilita que exista menos *hardware* disponível, logo existe menos consumo de energia para a manutenção e refrigeração.

Contudo para Pedro Silva(2009) a virtualização tem quatro grandes vantagens para o utilizador:

- Compatibilidade: Todas as máquinas virtuais são compatíveis com todos os computadores x86 padrão. A máquina virtual hospeda sistemas operativos *guest*, aplicativos próprios e todos os dispositivos de *hardware* de uma máquina real, por esta mesma razão as máquinas virtuais são compatíveis com todos os sistemas operativos, aplicativos e drivers de dispositivo padrão x86.
- Isolamento: As máquinas virtuais são isoladas umas das outras, como se fossem dispositivos de *hardware* completamente diferentes. Embora todas as máquinas virtuais possam estar alojadas na mesma máquina física, ficam totalmente isoladas umas das outras, esta é uma das razões pela qual a disponibilidade e a segurança dos aplicativos em execução em ambientes virtuais é muito superior aos do sistema tradicional não virtualizado.
- Encapsulamento: As máquinas virtuais encapsulam um ambiente de computação completo, isto é, uma máquina virtual funciona como um arquivo de *software* em que nós podemos mover e copiar de um local para o outro, isto porque as máquinas virtuais não são nada mais nada menos que um container de *software* que reúne um conjunto completo de recursos virtuais de hardware, um sistema operativo e todos os aplicativos dentro de um pacote de *software* e assim as máquinas virtuais tornam-se muito mais compactas e fáceis de gerir.

- Independência do *hardware*: As máquinas virtuais são executadas independentemente do *hardware* agregado. Podemos configurar uma máquina virtual com componentes de *hardware* virtuais (por exemplo CPU, placa de rede, controladora SCSI), e podem ser completamente diferentes dos componentes físicos da máquina subjacente, até as máquinas virtuais no mesmo servidor podem utilizar diferentes sistemas operativos como Windows, Linux, Mac OS, etc. Esta independência também permite executar uma mistura heterogénea¹ de sistemas operativos e aplicativos num único computador físico.

Desvantagens

Uma das desvantagens é que as máquinas virtuais consomem muito processamento e Memória, e por essa mesma razão exige máquinas com elevado desempenho e de configuração mais sofisticada, com um custo muito mais elevado.

Alexandre Carissimi(2009) defende que a virtualização tem elevados problemas ao nível da segurança, gestão e desempenho. Sendo que o *hypervisor* é uma camada de software e por isso é vulnerável, por essa mesma razão hoje em dia as máquinas virtuais são menos seguras em relação às máquinas físicas, para além de tudo existem muitas opiniões contraditórias sobre este tema. Podemos ainda concluir que para além de tudo ainda há uma grande desvantagem na alocação de máquinas virtuais em servidores, isto porque quando uma é afetada, todas o serão em simultâneo. Mas todos sabemos que existe a possibilidade de replicarmos máquinas virtuais com muita facilidade, assim como a facilidade que temos na sua portabilidade.

Quanto à gestão das máquinas virtuais ainda existem muitas melhorias a serem feitas, porque qualquer máquina virtual precisa de ser criada, monitorizada, configurada, existem sistemas como o *Xen*, *VMWare player*, *Virtual PC 2007*, *VirtualBox* que ajudam a solucionar estes problemas, mas ainda existem deficiências como a gestão relacionada com a segurança na correlação de eventos e ações feitas como o root² em ambientes virtuais. Em relação ao desempenho, existem várias dúvidas quanto aos custos agregados ao processamento introduzido pela camada de virtualização, e quanto

¹ Consideramos uma mistura heterogénea como sendo uma mistura em que ambos estão juntos mas não se misturam. ou seja o solvente não consegue dissolver o soluto

² É a raiz de um sistema

a possibilidade de instalar n máquinas sem colocar o desempenho de outras em causa (escalabilidade).

Cloud Computing

História

Emerson Alecrim (2008) considera que o *cloud computing* é de difícil precisão sobre a sua origem, sendo que como falamos anteriormente na história da virtualização, iniciada em 1960, podemos então juntar que na mesma década o físico Joseph Carl Robnett Licklider criou a internet, fazendo parte da ARPA(*Advanced Research Projects Agency*), tendo a tarefa de dar outras funcionalidades a um computador que não só a de ser uma “poderosa calculadora”, e percebeu que os computadores podiam ser usados de forma a estarem interligados para a partilha de dados e para comunicarem uns com os outros, apartir daqui foi criado a *Intergalactic Computer Network* que posteriormente deu origem á famosa ARPANET, por sua vez foi a “porta de entrada” para a internet. Portanto ao juntarmos os criadores do TTS(John McCarthy) e o criador da internet J.C.R. Licklider podemos perceber que nesta altura terá sido a criação daquilo que chamamos de computação em nuvem ou *cloud computing*

Cloud Computing nos dias de hoje

No nosso dia-a-dia estamos habituados a armazenar arquivos e dados de uma forma *on premise*, isto é, nos nossos próprios computadores, o *cloud computing* veio mudar tudo isto, sendo que hoje em dia com a ótima conexão a internet que temos tanto no âmbito residencial como empresarial, com esta tecnologia podemos ter tudo alojado numa nuvem, portanto quando temos alguma aplicação não temos que nos preocupar com nada porque o fornecedor irá tratar de tudo, desde o desenvolvimento até ao *backup*, a única necessidade do utilizador será o acesso a internet.

Como exemplo temos o Google docs, em que o utilizador não tem qualquer necessidade de ter software instalado na sua máquina, necessita só de ter acesso a internet para poder editar textos, fazer apresentações em slides entre outras funções. Assim como *Windows Azure*, em que temos servidores alojados numa nuvem e a partir da internet, de forma *on-line* podemos criar máquinas virtuais, desenvolver aplicações multitarefas entre muitas outras funções.

Segundo A. Ohri (2014) existem três grandes participantes na nuvem, que é o fornecedor do serviço, o desenvolvedor e o utilizador, sendo que o desenvolvedor é o Pedro Gonçalves

responsável por disponibilizar, gerir e monitorizar toda a infraestrutura da nuvem, desenvolver as aplicações no espaço disponibilizado para o utilizador final, e o utilizador final é quem vai usufruir de todo este serviço disponibilizado na computação em nuvem.

Uma das empresas que atualmente mais investe em *cloud computing* é a Microsoft. O *cloud computing* é caracterizado por permitir ao utilizador uma elevada escalabilidade porque a nuvem dá a ideia ao utilizador que tem todos os recursos disponíveis de forma rápida, por deixar que o utilizador faça a sua própria escolha quanto à necessidade de querer novos recursos, a qualquer hora, o facto de permitir ao utilizador o acesso a partir de qualquer plataforma que tenha acesso à rede.

No *cloud computing* existem quatro tipos de serviços, são o PaaS, DaaS, IaaS e TaaS, em que o **IaaS** é a camada inferior da arquitetura e responsável por ter toda a infraestrutura necessária para as camadas intermédias e superiores, tem servidores capazes de executar *softwares* customizados e de operar em diferentes sistemas operativos, temos como exemplo o *Amazon Elastic Compute Cloud*, o **PaaS** é a plataforma como serviço, que é utilizada mais no âmbito do programador, permite que o desenvolvedor não se preocupe com a necessidade de ter mais processadores ou memória, oferece uma infraestrutura compatível com diversos sistemas operativos, linguagens de programação e ambientes de desenvolvimento, como exemplo temos o *Google App Engine*, o **SaaS** é o software como serviço, é a camada mais alta da arquitetura da computação em nuvem, tem a total responsabilidade de disponibilizar as aplicações completas ao utilizador final, que temos como exemplos mais conhecidos para os utilizadores, o *Microsoft Azure*, *Chrome OS*, entre outros. O **TaaS** que é o teste ao serviço permite ao utilizador testar as aplicações e sistemas remotamente, simulando o comportamento das aplicações como se estivessem em execução.

A. Ohri(2014) afirma também que existem quatro tipos de modelos de implementação que são o público, privado, comunidade e híbrido, em que no modelo **público** a nuvem é disponibilizada para os utilizadores de forma geral ou para grandes grupos industriais, em que a nuvem implementada por um prestador dos serviços assegura todo o desempenho e segurança da mesma.

As nuvens **privadas** são exclusivamente para uma organização, em que a gestão da rede pode ser feita pela própria organização ou por uma empresa contratada para o fazer, neste caso a responsabilidade da infraestrutura passa a ser do utilizador. O modelo

comunidade é a possibilidade de várias organizações partilharem a mesma infraestrutura de nuvem, e suporta uma comunidade específica que partilha as mesmas preocupações como missão, assim como a segurança, políticas, estas podem ser também geridas pelo utilizador ou por terceiros, pode existir remotamente ou localmente. Por ultimo na nuvem **híbrida** a infraestrutura é composta por dois ou mais modelos de implementação, sendo que cada nuvem permanece como uma entidade única, apenas unidos pelo uso de tecnologia proprietária ou padronizada, e assim garante a portabilidade de dados e aplicações, uma das suas características é o seu uso para executar tarefas periódicas que são mais facilmente implementadas em nuvens públicas.

Vantagens

As principais vantagens encontradas no *cloud computing* são:

- O facto do utilizador poder ter acesso ás suas aplicações e dados a partir de qualquer lugar desde que tenha acesso a internet, tendo maior mobilidade e flexibilidade;
- Os seus custos, sendo que o utilizador terá que pagar só consoante as suas necessidades, sem desperdiçar recursos;
- A possibilidade do utilizador poder ampliar os seus recursos, graças a sua escalabilidade;
- Uma empresa não tem a necessidade de comprar grandes recursos de *hardware* e não tem que assumir qualquer responsabilidade sobre a infraestrutura contratada;
- A facilidade de utilização dos serviços de partilha de recursos, e da confiabilidade dos serviços, visto que as empresas que os fornecem são empresas que têm uma grande reputação e conseguem manter os dados seguros através de cópias de segurança.

Desvantagens

Por outro lado também existem desvantagens quanto ao *cloud computing* como:

- O facto de ser obrigatório ao utilizador estar ligado a internet para poder aceder aos seus dados na nuvem

- A recuperação de dados, caso a nuvem deixe de funcionar é impossível recuperá-los, além de ser muito fiável podem sempre existir erros;
- O utilizador estar ligado a internet para aceder a nuvem também faz com que esteja mais vulnerável a ataques informáticos aos seus dados, como é do conhecimento geral, a segurança informática é um tópico transversal em qualquer tema da área da informática.

ADO.NET

História

O ADO.NET foi desenvolvido pela Microsoft para facilitar a comunicação com as base de dados, é um conjunto de classes do .Net Framework, é uma evolução do ADO (*ActiveX Data Objects*), sendo que o ADO e o ADO.NET quanto á sua arquitetura nada têm a ver, porque foi toda ela alterada.

Damien Foggon (2006) conta que o ADO.NET foi criado pela Microsoft para facilitar o acesso ás base de dados. No final da década de 80 e inicio dos anos 90, os fornecedores de servidores de base de dados todos enfrentaram o mesmo problema, queriam que os clientes quando usassem a sua aplicação pudessem aceder a uma base de dados á sua escolha e por esse mesmo motivo tinha que existir uma linguagem universal que pudesse ser utilizada por todos. Nessa altura a solução dos fornecedores de base de dados veio com o chamado *Open Database Connectivity*(ODBC), o qual é um conjunto comum de funções e interfaces acordado por todos os principais fornecedores de base de dados, e na época foi implementado em todos os servidores dos principais fornecedores.

ADO.net no seu estado atual

Hoje em dia o ADO.NET é muito utilizado pelos programadores, porque quando foi criado pela Microsoft foi a pensar nos dias em que vivemos,ligados a internet, e por isso mesmo o ADO.NET tem muitas características interessantes para o âmbito do programador, como por exemplo o facto de podermos trabalhar em ambiente desconectado, isto é, não é necessário estar ligado a base de dados para podermos manipular os dados, uma característica muito importante para os programadores que

têm a necessidade de ter elevado desempenho na criação de sites web ou aplicações, outra novidade com o ADO.NET é a sua integração com o XML, ADO.NET faz a sua comunicação com o XML permitindo a fácil comunicação entre plataformas heterogéneas, ainda ao contrário do ADO o ADO.NET pode utilizar tudo como fonte, desde que o utilizador possa aceder via OleDb.

Segundo John Sharp(2012) existem dois grandes grupos no ADO.NET:

- *Managed providers* que são os objetos utilizados para escrever e ler na base de dados, por exemplo: Connection, DataAdapter, Command e DataReader, para cada plataforma existem classes de acessos a dados diferentes, ou seja, para o SQL server e para Oracle por exemplo são diferentes;
- *DataClasses* neste caso são os objetos que podem armazenar e manipular dados, sem saberem a origem dos dados, nem o que significam, por exemplo: DataSets, DataTables, DataRow, DataColumns, DataRelations Constraints e DataView.

John Sharp(2012) descreve-nos ainda as funções de cada objeto, sendo que:

- *Connection*: é responsável por estabelecer a ligação;
- *Command*: é utilizado para enviar comandos SQL para a base de dados, como por exemplo executar *Stored Procedures*;
- *DataReader*: é o objeto que lê um conjunto de dados de forma sequencial, é muito utilizado para preenchimento de controlos e não exige a alteração de dados;
- *DataAdapter*: este é um *container* para objetos *Command*, os quais fazem o *DataAdapter* interagir com a Base de dados, fazendo pesquisas, inserções, alterações e exclusões;
- *DataSet*: este objeto armazena os dados em memória, independentemente do tipo de dados, tem *DataTable*, *DataColumn*, *DataRow*, *Constraints*, *DataRelations*.

Sendo o ADO.NET uma linguagem que veio ajudar muito os programadores, na sua criação, foi pensada a criação de vários objetos, assim como o exemplo dos *providers*, porque vejamos *providers* ou fornecedores vieram facilitar em muito a linguagem. Na ligação a uma base de dados SQL ou Oracle existem *providers*.

Como principais *namespace* do ADO.NET temos, o *System.Data*, *System.Data.SqlClient*, *System.Data.OracleClient*, *System.Data.OleDb*. Para fazer a

ligação com a base de dados existe o *SqlConnection*, *OracleConnection*, *OleDbConnection*, *OdbcConnection*, para executar comandos existe o *SqlCommand*, *OracleCommand*, *OleDbCommand*, *OdbcCommand*.

Vantagens

O ADO.NET pelas características que tem traz várias vantagens, entre elas:

- Interoperabilidade, através do padrão XML;
- Escalabilidade, com recursos como o pooling³ de conexões e de dados desconectados;
- Produtividade, por ser um modelo de fácil compreensão e com modelos que nos permitem o desenvolvimento rápido de camadas para acedermos a dados;
- alto desempenho, com a possibilidade de o utilizador poder trabalhar em ambiente desconectado, permite ter um elevado aumento de *performance* principalmente quando é necessário aceder muitas vezes à base de dados

Desvantagens

Quanto ao ADO.NET existe apenas uma grande desvantagem que é, o facto de os dados nem sempre estarem atualizados, porque é necessário que para isso acontecer exista uma constante ligação ao servidor.

SQL

História

SQL(Structured Query Language), ou seja linguagem de consulta estruturada, é uma linguagem de pesquisa declarativa para base de dados relacionais.

Segundo Luis Damas(2005) o SQL deu inicio nos anos 70, fruto do trabalho produtivo do colega de pesquisa da IBM Dr. E. F. Codd que desenvolveu o SEQUEL(*Structured English Query Language*) e que ultimamente se transformou em SQL(*structured Query Language*).

³ Variedade de serviços, ou neste caso uma variedade de conexões

A IBM juntamente com outros fornecedores de base de dados relacionais queria um método padronizado para poder aceder e manipular dados numa base de dados relacional. Sendo a IBM a primeira a desenvolver esta teoria, foi a Oracle a primeira a fornecer e a comercializar esta tecnologia. Com o passar do tempo, o SQL tornou-se muito popular, chamando a atenção da ANSI(*American National Standards Institute*), a qual lançou padrões para o SQL entre 1986 e 2006, em várias versões. Sendo assim desde 1986 que os programadores e desenvolvedores tiveram a sua tarefa mais simplificada, porque assim conseguem a partir de uma só linguagem, aceder às várias plataformas de base de dados relacionais, apenas com pequenos ajustes entre elas.

SQL nos dias de hoje

O SQL(*Structured Query Language*) foi criada para manipular as base de dados com maior facilidade, é uma linguagem declarativa para base de dados relacionais, ou seja o facto de ser declarativa torna-a muito mais simplificada para quem se inicia na mesma, originalmente foi baseado em álgebra relacional. O SQL é uma linguagem de fácil uso e simplicidade, muito pelo facto de ser uma linguagem padronizada.

Uma das grandes diferenças do SQL em relação a outras linguagens de Base de dados é que no caso do SQL a consulta mostra-nos o resultado e não o caminho até ao resultado. No SQL podemos criar comandos, através de *queries*, para pesquisar, inserir, eliminar registos de uma base de dados. O software mais utilizado em base dados para SQL é o MySql

Hoje em dia o SQL é a linguagem de estrutura de base de dados mais utilizada e como tal podemos ver algumas das suas características e regras. No SQL temos várias instruções, como:

- Instruções de ligação SQL, o Connect e disconnect;
- Controlo do SQL, a instrução Call e return;
- Instrução para os dados SQL, o Select, Insert, Update, Delete;
- Para fazer diagnósticos SQL, como o Get Diagnostics;
- Instruções para esquemas SQL, o Alter, Create e DROP;

Segundo Brian Knight, Devian Knight, Jessica M. Moss, Mike Davis, Chris Rock (2014) existe uma plataforma ou uma arquitetura da Microsoft, o SSIS(*Server Integration Services*) que se encaixa na plataforma de inteligência do negócio, *Business*

Intelligent(BI), mas além do SSIS, existem outras tecnologias como tecnologia de reescrita de código completo ETL, mas como o SSIS é a tecnologia com maior velocidade de carregamento, logo também será a mais utilizada. Na tecnologia SSIS existem ferramentas como o *import and export wizard* que é o método mais fácil para mover dados de fontes como o *Oracle, Sql Server* entre outros, e está disponível em quase todas as versões do SQL Server, também temos o SSDT (*SQL Server Data Tools*) que é o local onde o desenvolvedor estará mais tempo alocado á medida que vai desenvolvendo os seus pacotes.

Itzik Ben-Gan(2012) afirma que o utilizador nunca deve partilhar o servidor de produção com outros servidores. Ao nível de segurança no SQL server temos a opção de Windows Authentication e Mixed Mode, em que na primeira utilizamos a segurança do Windows e na segunda utilizamos diretamente a segurança do SQL. No SQL existem dois tipos de base de dados, que é a de sistema e as definidas pelo utilizador, em que as base de dados de sistema são, Master (utilizada para guardar a informação sobre o que se passa no sistema), MSDB(executa tarefas predeterminadas no tempo, como backups), Model(é utilizada como um modelo de todas as novas base de dados que vierem a ser criadas), TempDb(armazena todos os dados temporários necessários para o correto funcionamento do SQL Server), e cada uma destas base de dados contém pelo menos os ficheiros *primary data file* e *log files*.

A segurança no SQL é de alto nível, e um dos pontos fundamentais para a segurança é no processo do Login. Existem três tipos de utilizadores, os Built-In Users e System Roles, os Database Users e os Guest User. No caso de termos a necessidade que vários utilizadores accedam aos mesmo serviços, podemos criar roles, e as roles que podem ser criadas são:

- Public role (adicionada a todas as BD criadas pelo utilizador)
- Server Roles (aplicadas na instalação do SQL ao nível do servidor, e nas Server Roles existem outras roles, como o SysAdmin, Server Admin, SetupAdmin entre outros)
- Database Roles (permitem um conjunto de tarefas de administração de base de dados, e contém outras roles, como o *Db_owner, Db_datareader*, entre outros)

No SQL foram criados dois modelos de recuperação, em que o primeiro é o *Full model*, o qual guarda todas as transações no log, muito útil para as empresas, permite efetuar a recuperação até ao último backup total, e o segundo é o *simple model* que é otimizado para poucas transações, e mantém o log com espaço reduzido.

No SQL existe um parâmetro muito importante que é a integridade de dados, integridade de identidade ou de tabela, requer que todas as linhas tenham uma chave primária, a integridade de domínio que determina um conjunto de valores que são ou não permitidos para essa linha, como ver se aceita valores *Nulls* ou não, e por fim a integridade referencial em que esta assegura que os relacionamentos entre a chave primária e estrangeira são mantidos. A chave primária é uma coluna ou combinação de colunas, que identifica univocamente uma linha, enquanto a chave estrangeira aponta para a chave primária de outra tabela ou da mesma tabela, em que a sua finalidade é garantir a total integridade dos dados.

Desenvolvimento Biblioteca Virtual

A aplicação visa aprofundar todos os conhecimentos adquiridos ao longo da licenciatura, no contexto de projeto global.

Como tal, foi proposto ao júri o desenvolvimento de uma biblioteca virtual, onde os utilizadores poderão consultar todos os dados de um livro, quanto aos administradores poderão consultar, inserir, editar e remover, e a esta aplicação demos o nome de biblioteca virtual.

Pré-requisitos

Foi proposto e aceite pelo júri como pré-requisitos para este trabalho, a possibilidade de consulta de dados, como os dados do livro, autores e editoras, assim como a inclusão e construção de uma base de dados SQL, um sistema de autenticação próprio e um sistema de gestão e administração da aplicação.

Para um utilizador poder consultar os dados, fico definido:

- Criação de um ícone no ambiente de trabalho para aceder a aplicação
- Autenticação do utilizador, com autenticação própria
- Pesquisar pelo(s) campo(s) de pesquisa com caixa de texto para cada um
- Visualização dos resultados da pesquisa
- Clicar o botão para poder descarregar o resultado da pesquisa em .pdf
- Existirá ainda uma barra de Menus que estará disponível consoante as credenciais, neste caso estará só disponível para as credenciais Administrador.

Estrutura da aplicação

Modelo relacional

Como foi definido no trabalho, iremos desenvolver toda a estrutura de base de dados em SQL, e visto os utilizadores terem autenticação própria, a estrutura do diagrama SQL será a seguinte:

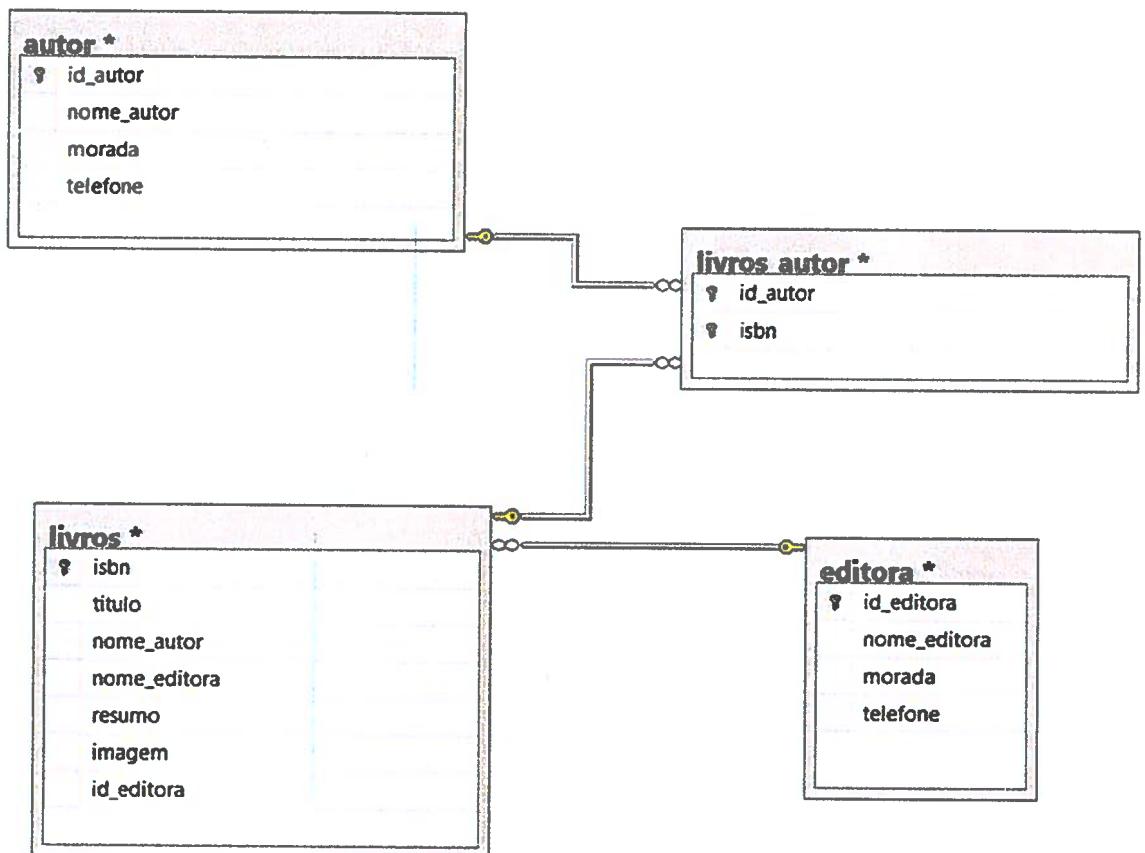


Figura 1 Tabelas relacionais SQL

Como podemos ver de uma forma mais aprofundada, as tabelas estão todas relacionadas entre si, começando pela tabela livros que está relacionada com a tabela editora, em que a sua relação é de 1:n, visto que uma editora pode ter vários livros mas um livro só pode ter uma editora, a mesma tabela livros tem uma tabela relacional livros_autor, esta tabela serve para relacionar a tabela livros com a tabela autor através das suas chaves primárias, por ultimo temos a tabela autor que por sua vez também está relacionada com a tabela livros, através da tabela livros_autor.

A tabela livros_autor é importante também porque sendo uma tabela relacional evita muita replicação de dados.

Tabela livros

A tabela livros, vai ser utilizada para guardar todos os dados dos livros, como o isbn, que será a sua identidade única, o titulo, nome do autor, nome da editora, o resumo do livro e a imagem da capa.

livros *	
?	isbn
?	titulo
?	nome_autor
?	nome_editora
?	resumo
?	imagem
?	id_editora

Figura 2 Tabela livros

Tabela livros autor

A tabela livros_autor será utilizada para relacionar a tabela livros com a tabela autor. É nesta tabela que serão feitas as cláusulas entre as duas tabelas, de maneira a que não hajam dados replicados, e que os dados sejam integrais.

livros autor *	
?	id_autor
?	isbn

Figura 3 tabela livros_autor

Tabela autor

A tabela autor é a tabela utilizada para guardar todos os dados do(s) autor(es) de um determinado livro, dados esses que não estão introduzidos na tabela livros porque são únicos do autor, assim como a morada e o telefone.

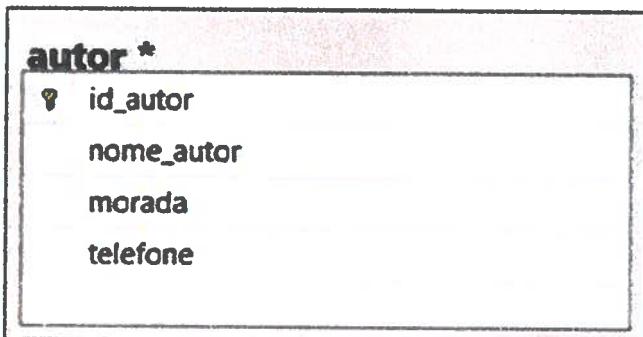


Figura 4 tabela autor

Tabela editora

A tabela editora vai servir para guardar todos os dados relativos á editora, que não estão contidos na tabela livros, assim como a morada e o telefone, ou seja, os seus dados pessoais.



Figura 5 Tabela editora

Construção da aplicação

A aplicação foi construída recorrendo ao *software Microsoft Visual Studio 2013*, utilizando a linguagem c#, e desenvolvida em *WPF Application*. Toda a estrutura de base de dados foi adicionada como uma *Data Connection*, tendo sido criada no SQL.

App.config

Utilizamos o ficheiro de configuração *App.config* de forma a podermos ter uma *connection string* geral a todo o programa, sendo que é guardada neste ficheiro, e durante o programa iremos chamar só o nome dessa *connection string*, assim se houver necessidade de alterar a ligação á base de dados podemos apenas aceder ao ficheiro *app.config* e alterar a *connection string*.

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <configSections>
        </configSections>
    <connectionStrings>
        <add name="Teste.Properties.Settings.LivrosConnectionString"
            connectionString="Data Source=tiagoacer;Initial Catalog=Livros;Integrated Security=True"
            providerName="System.Data.SqlClient" />
        <add name="Teste.Properties.Settings.LivrosConnectionString1"
            connectionString="Data Source=WinServer;Initial Catalog=Livros;User ID=sa;Password=Admin2015"
            providerName="System.Data.SqlClient" />
        <add name="Teste.Properties.Settings.LivrosDataSet3" connectionString="Data Source=pc-ric;Initial Catalog=Livros;Integrated Security=True"
            providerName="System.Data.SqlClient" />
    </connectionStrings>
    <startup useLegacyV2RuntimeActivationPolicy="true">
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />
    </startup>
    <system.web>
        <authentication mode="Windows" />
        <authorization>
            <deny users="?" />
        </authorization>
    </system.web>
</configuration>
```

Figura 6 App.config

Utilizador

A página do utilizador será a página onde estão todos os campos necessários para a consulta dos livros. Sendo que esta página apenas permite consultar e descarregar o .pdf. A autenticação á mesma será feita consoante os utilizadores que tiverem criados no *Active Directory*.

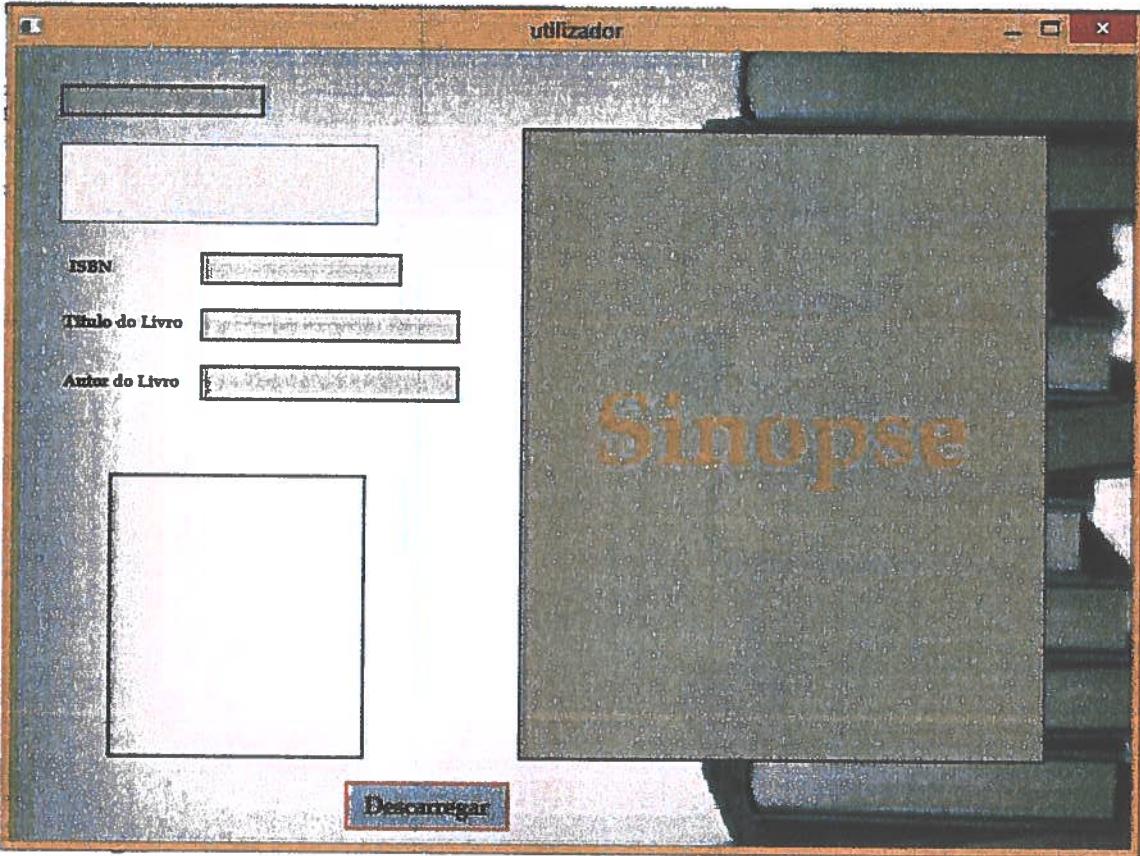


Figura 7 Utilizador - layout

Cod. Utilizador

O código que se segue será a *class* utilizador, que vai ser a *class* chamada quando o programa for iniciado, sendo que dentro dessa classe definimos que o txtNome(Título do Livro) será apenas de leitura, e não pode ser alterado.

```
public partial class utilizador : Window
{
    public utilizador()
    {
        InitializeComponent();
        txtNome.IsReadOnly = true;
    }
}
```

Figura 8

Agora criamos o evento *PreviewMouseUp dg_resultado* que será o evento sobre a *dataGrid* onde irá ser feita a pesquisa dos livros, na imagem seguinte iremos ver a chamada a *connection string* geral, criada no *App.config* como falada anteriormente.

```
private void dgResultado_PreviewMouseUp(object sender, MouseButtonEventArgs e)
{
    string connectionString = null;

    SqlConnection sql_conn;
    connectionString = ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectionString1"].ConnectionString;
    sql_conn = new SqlConnection(connectionString);
```

Figura 9

Dentro ainda do mesmo evento, como irá ser visível na figura seguinte iremos executar uma *query* para que possamos selecionar todos os campos da tabela livros, e consoante o campo existente irá ser mostrada na DataGridView dgResultado, o resultado da pesquisa. Ainda neste evento, quando ele for executado vamos renomear os cabeçalhos da tabela para os nomes pretendidos, para que não apareça os nomes da forma como estão nos cabeçalhos da base de dados.

```

try
{
    sql_conn.Open();
    lblSinopse.Visibility = Visibility.Hidden;
    string query = "SELECT * FROM livros";

    SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
    cmdlogin.ExecuteNonQuery();
    SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
    DataTable dt = new DataTable("livros");
    DataSet ds = new DataSet();
    da.Fill(ds);
    da.Fill(dt);
    var nome = dgResultado.SelectedIndex;
    DataRow selectedRow = dt.Rows[nome];

    txtISBN.Text = selectedRow["isbn"].ToString();
    txtNome.Text = selectedRow["titulo"].ToString();
    txtAutor.Text = selectedRow["nome_autor"].ToString();
    txtSinopse.Text = selectedRow["resumo"].ToString();
    txtPdf.Text = txtNome.Text;

    byte[] data = (byte[])selectedRow["imagem"];
    MemoryStream strm = new MemoryStream();
    strm.Write(data, 0, data.Length);
    strm.Position = 0;
    System.Drawing.Image img = System.Drawing.Image.FromStream(strm);

    BitmapImage bi = new BitmapImage();
    bi.BeginInit();
    MemoryStream ms = new MemoryStream();
    img.Save(ms, System.Drawing.Imaging.ImageFormat.Bmp);
    ms.Seek(0, SeekOrigin.Begin);
    bi.StreamSource = ms;
    bi.EndInit();
    img1.Source = bi;
}

```

Figura 10

O evento seguinte será o *PreviewKeyDown* que será executado na Text Box txtPesquisa. e onde é executada uma *query* para que possa fazer a pesquisa através de qualquer campo da tabela livros, seja o campo isbn, titulo, nome do autor ou nome da editora, qualquer um destes campos pode ser utilizado como pesquisa. Executa-se também o método clear para que quando faz uma pesquisa, ele vá limpando os resultados da mesma.

```

private void txtPesquisa_PreviewKeyDown(object sender, KeyEventArgs e)
{
    SqlConnection sql_conn;
    sql_conn = new SqlConnection(@"Data Source=pc-ric;Initial Catalog=Livros;Integrated Security=True");

    try
    {
        sql_conn.Open();
        string query = "SELECT isbn,titulo,nome_autor,nome_editora FROM livros where isbn like'" +
            this.txtPesquisa.Text + "%' OR titulo like'" + this.txtPesquisa.Text + "%' OR nome_autor like'" + this.txtPesquisa.Text + "%' OR nome_editora like'" + this.txtPesquisa.Text + "%'";
        SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
        cmdlogin.ExecuteNonQuery();

        txtAutor.Clear();
        txtISBN.Clear();
        txtNome.Clear();
        txtSinopse.Clear();

        SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
        DataTable dt = new DataTable("livros");
        da.Fill(dt);

        dgResultado.ItemsSource = dt.DefaultView;

        da.Update(dt);

        sql_conn.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

```

Figura 11

O evento *button_click* é utilizado para descarregar o pdf, como podemos ver na imagem seguinte, foi criada uma txtpdf que é uma texto box que não está visível no layout mas serve para receber o valor da *string* contida na texto box do titulo do livro, e assim conseguimos abrir o caminho do pdf com o nome do livro pretendido, como demonstra no código da imagem seguinte.

```
private void Button_Click(object sender, RoutedEventArgs e)
{
    string caminhoPdf = txtPdf.Text;

    try
    {
        System.Diagnostics.Process.Start("D:\\\" + caminhoPdf + ".pdf");
    }
    catch (Exception ex)
    {
        MessageBox.Show("Não é possível apresentar o PDF");
    }
}
```

Figura 12

Cod. Administrador

O código seguinte será para a apresentação e explicação da página livros, mas a página autor e editora utilizam as mesmas *class*, apenas alteram as *querys*, para que possam editar, alterar, remover, pesquisar consoante a página e assim sendo têm que executar a *query* igual á de livros mas que seja correspondente e página selecionada, autor ou editora.

Na imagem seguinte podemos ver o *layout* da aplicação de Administrador, onde tem os vários campos de preenchimento e os botões para executar os eventos, assim como o menu para podermos ir de página para página.

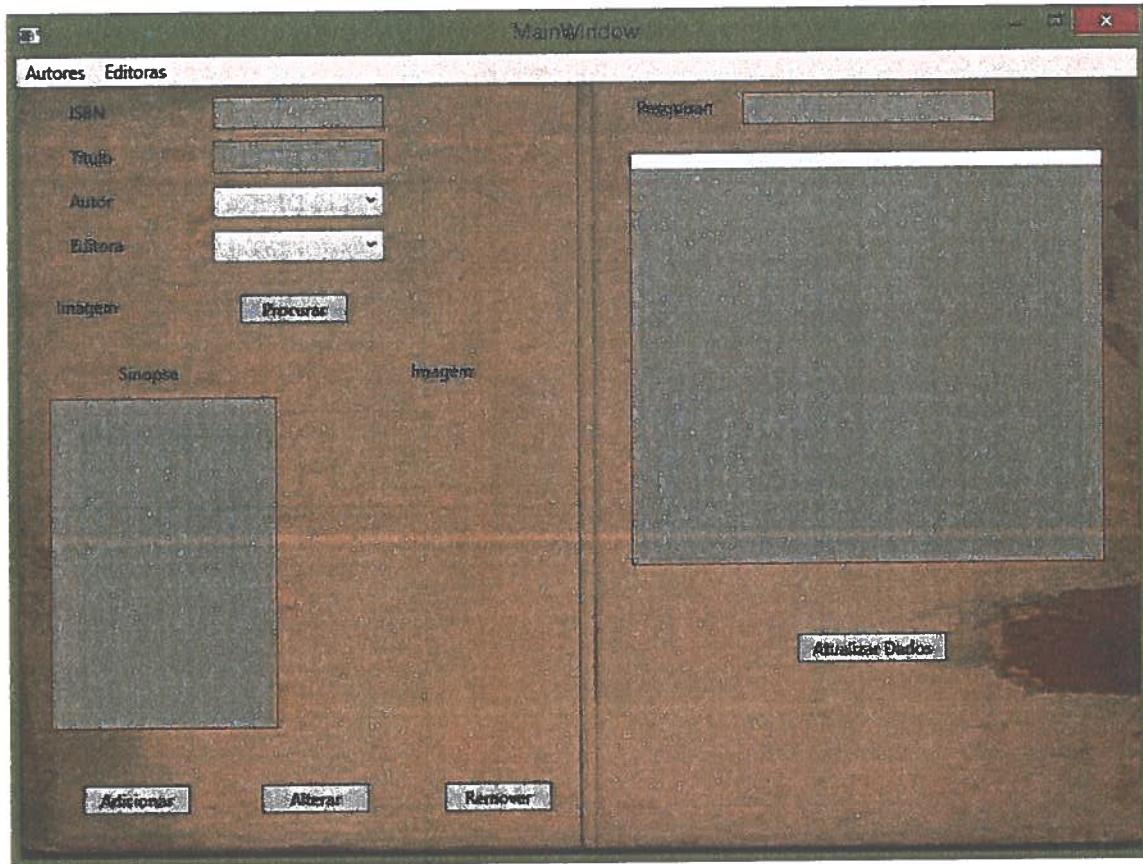


Figura 13

A class na imagem seguinte vai ser utilizada para preencher as comboBoxs da página principal livros, com os valores das tabelas editora e autor.

Será iniciado na altura de inicialização da página.

```

public MainWindow()
{
    InitializeComponent();

    string connectionString = null;

    SqlConnection sql_conn;
    connectionString = ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectionString1"].ConnectionString;
    sql_conn = new SqlConnection(connectionString);
    sql_conn.Open();

    //Combosautor
    string query = "SELECT nome_autor FROM autor";
    SqlCommand cmdlogin = new SqlCommand(query, sql_conn);

    SqlDataReader dr = cmdlogin.ExecuteReader();
    while (dr.Read())
    {
        string sAutor = dr.GetString(0);
        cmbAutor.Items.Add(sAutor);
    }
    sql_conn.Close();
    sql_conn.Open();

    //ComboEditora
    string query1 = "SELECT nome_editora FROM editora";
    SqlCommand cmdlogin1 = new SqlCommand(query1, sql_conn);

    SqlDataReader dr1 = cmdlogin1.ExecuteReader();
    while (dr1.Read())
    {
        string sEditora = dr1.GetString(0);
        cmbEditora.Items.Add(sEditora);
    }
    sql_conn.Close();
}

```

Figura 14

Em primeiro lugar iremos ter o botão atualizar que é onde o Administrador tem a possibilidade de atualizar a base de dados, neste mesmo evento alteramos novamente o nome dos cabeçalhos, ou seja a forma como são apresentados na DataGridView, isto para que não apareçam com os mesmos nomes que existem na tabela do SQL

```

private void btnAtualizar_Click(object sender, RoutedEventArgs e)
{
    string connectionString = null;
    SqlConnection sql_conn;
    connectionString = ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectionString1"].ConnectionString;
    sql_conn = new SqlConnection(connectionString);

    try
    {
        sql_conn.Open();
        string query = "SELECT * FROM Livros";
        SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
        cmdlogin.ExecuteNonQuery();

        SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
        DataTable dt = new DataTable("autores");
        da.Fill(dt);

        dg1.ItemsSource = dt.DefaultView;
        da.Update(dt);

        this.dg1.Columns[0].Header = "ISBN";
        this.dg1.Columns[1].Header = "Título";
        this.dg1.Columns[2].Header = "Nome do Autor";
        this.dg1.Columns[3].Header = "Editora";
        this.dg1.Columns[4].Visibility = Visibility.Hidden;
        this.dg1.Columns[5].Visibility = Visibility.Hidden;
        this.dg1.Columns[6].Visibility = Visibility.Hidden;

        sql_conn.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

```

Figura 15

O próximo evento é o evento criado sobre a *DataGrid*, ou seja quando existir um *DoubleClick* com o rato sobre uma linha da *DataGrid*, vão aparecer todos os campos da aplicação preenchidos para que o Administrador depois possa editar, remover, alterar.

```
private void dg1_MouseDoubleClick(object sender, MouseButtonEventArgs e)
{
    string connectionString = null;
    SqlConnection sql_conn;
    connectionString = ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectionString1"].ConnectionString;
    sql_conn = new SqlConnection(connectionString);
    try
    {
        sql_conn.Open();
        string query = "SELECT * FROM livros";
        SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
        cmdlogin.ExecuteNonQuery();
        SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
        DataTable dt = new DataTable("livros");
        DataSet ds = new DataSet();
        da.Fill(ds);
        da.Fill(dt);
        var nome = dg1.SelectedIndex;
        DataRow selectedRow = dt.Rows[nome];
        txtISBN.Text = selectedRow["isbn"].ToString();
        txtTitulo.Text = selectedRow["titulo"].ToString();
        txtSinopse.Text = selectedRow["resumo"].ToString();
        cmbAutor.SelectedItem = selectedRow["nome_autor"].ToString();
        cmbEditora.SelectedItem = selectedRow["nome_editora"].ToString();
        byte[] data = (byte[])selectedRow["imagem"];
        MemoryStream strm = new MemoryStream();
        strm.Write(data, 0, data.Length);
        strm.Position = 0;
        System.Drawing.Image img = System.Drawing.Image.FromStream(strm);
        BitmapImage bi = new BitmapImage();
        bi.BeginInit();
        MemoryStream ms = new MemoryStream();
        img.Save(ms, System.Drawing.Imaging.ImageFormat.Bmp);
        ms.Seek(0, SeekOrigin.Begin);
        bi.StreamSource = ms;
        bi.EndInit();
        imgPreview.Source = bi;
    }
    catch (Exception ex)
```

Figura 16

O evento *btn_salvar* é disparado quando carregamos sobre o botão adicionar, ou seja quando editamos os campos da tabela, em primeiro lugar vai lançar uma mensagem de alerta para saber se tem a certeza de que quer adicionar as alterações e se a resposta for “sim”, então ai vai inserir todos os novos campos na tabela livros. Se for executado com êxito aparece uma mensagem no ecrã a dizer “conteúdo guardado”

```

private void btnSalvar_Click(object sender, RoutedEventArgs e)
{
    MessageBoxResult result = MessageBox.Show("Tem a Certeza?", "Informação", MessageBoxButton.YesNoCancel);
    switch (result)
    {
        case MessageBoxResult.Yes:
            string connectionString = null;
            SqlConnection sql_conn;
            connectionString = ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectionString"].ConnectionString;
            sql_conn = new SqlConnection(connectionString);

            byte[] imageData;
            imageData = File.ReadAllBytes(caminho);
            try
            {
                sql_conn.Open();
                string query = "insert into livros (isbn,titulo,nome_autor,nome_editora,resumo,image,id_editora) values('" + this.txtISBN.Text + "','" + this.txtTitulo.Text + ',
                [" + cmbAutor.SelectedItem.ToString() + "','" + cmbEditora.SelectedItem.ToString() + "','" + this.txtSinopse.Text + ',@DATA,' + @DATA + ')';
                SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
                cmdlogin.Parameters.Add("@DATA", imageData);
                cmdlogin.ExecuteNonQuery();
                sql_conn.Close();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }

            MessageBox.Show("Conteúdo Guardado");
            break;
        case MessageBoxResult.No:
            MessageBox.Show("Conteúdo não Guardado");
            break;
    }
}

```

Figura 17

O botão alterar que é apresentado na imagem seguinte, este evento vai executar uma query, que será executada quando o botão for carregado e essa mesma query vai fazer a atualização a base de dados com os novos dados inseridos.

```

private void btnAlterar_Click(object sender, RoutedEventArgs e)
{
    MessageBoxResult result1 = MessageBox.Show("Tem a certeza?", "Informação", MessageBoxButton.YesNoCancel);
    switch (result1)
    {
        case MessageBoxResult.Yes:
            string connectionString = null;
            SqlConnection sql_conn;
            connectionString = ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectionString1"].ConnectionString;
            sql_conn = new SqlConnection(connectionString);
            byte[] imageData;
            imageData = File.ReadAllBytes(caminho);
            try
            {
                sql_conn.Open();
                string query = "UPDATE livros set isbn=" + this.txtISBN.Text + ",titulo=" + this.txtTitulo.Text + ",nome_autor=" + this.cmbAutor.SelectedItem.ToString() + ",nome_editora=" +
                this.cmbEditora.SelectedItem.ToString() + ",resumo=" + this.txtSinopse.Text + ",image=@DATA where isbn=" + this.txtISBN.Text + "";
                SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
                cmdlogin.Parameters.Add("@DATA", imageData);
                cmdlogin.ExecuteNonQuery();

                sql_conn.Close();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            MessageBox.Show("Conteúdo Alterado");
            break;

        case MessageBoxResult.No:
            MessageBox.Show("Conteúdo não Alterado");
            break;
    }
}

```

Figura 18

O próximo passo será executar um evento relativamente parecido com os descritos nas imagens anteriores, sendo este o botão remover, vai ser feito o Delete sobre os campos preenchidos pelo utilizador, e sendo assim surgiram também no ecrã umas caixas de mensagem para confirmar se a ação a ser executada é mesmo a pretendida, por uma questão de segurança.

```
private void btnRemover_Click(object sender, RoutedEventArgs e)
{
    MessageBoxResult result2 = MessageBox.Show("Tem a Certeza?", "Informação", MessageBoxButton.YesNoCancel);

    switch (result2)
    {
        case MessageBoxResult.Yes:
            string connectionString = null;

            SqlConnection sql_conn;
            connectionString = ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectionString1"].ConnectionString;
            sql_conn = new SqlConnection(connectionString);

            try
            {
                sql_conn.Open();
                string query = "DELETE FROM livros WHERE isbn='" + this.txtISBN.Text + "'";
                SqlCommand cmdlogin = new SqlCommand(query, sql_conn);

                cmdlogin.ExecuteNonQuery();

                sql_conn.Close();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            MessageBox.Show("Conteúdo Removido");
            break;

        case MessageBoxResult.No:
            MessageBox.Show("Conteúdo Não Removido");
            break;
    }
}
```

Figura 19

O botão procurar está associado ao evento *click* e quando for pressionado, vai procurar o caminho da imagem selecionado pelo Administrador e de seguida guarda esse mesmo caminho, sendo que tem que ser extensão .jpeg

```
private void btnProcurar_Click(object sender, RoutedEventArgs e)
{
    Microsoft.Win32.OpenFileDialog openFileDialog1 = new Microsoft.Win32.OpenFileDialog();
    openFileDialog1.ShowDialog();
    openFileDialog1.Filter = "JPEG Files (*.jpeg)|*.jpeg";
    openFileDialog1.DefaultExt = ".jpeg";

    caminho = openFileDialog1.FileName;
    ImageSource imageSource = new BitmapImage(new Uri(caminho));
    imgPreview.Source = imageSource;
}
```

Figura 20

Existe também nesta página um evento igual ao da [figura 11](#), para que o administrador também possa fazer pesquisas sobre todos os dados que estão nas diversas tabelas.

Foram criados 2 botões em cada uma das páginas de Administrador (livros, autor, editora) para que cada página possa ser redirecionada para as outras duas. Nas duas imagens seguintes iremos poder ver a chamada desses dois menus, dentro da página livros, contudo nas outras páginas será exatamente igual, sendo que a única coisa que muda é o nome da página a redirecionar, se estiver na autor, será para a página livros e editora, na página editora é para a página autor e livros.

```
private void menuAutores_Click(object sender, RoutedEventArgs e)
{
    this.Hide();
    autores autoresnovo = new autores();
    autoresnovo.Show();
}
```

Figura 21

```
private void menuEditoras_Click(object sender, RoutedEventArgs e)
{
    this.Hide();
    editoras editorasnovo = new editoras();
    editorasnovo.Show();
}
```

Figura 22

Maquinas Virtuais

Depois de terminada a criação da aplicação podemos passar para a implementação da aplicação em ambiente virtual. Para que pudéssemos tornar este laboratório real, criamos um servidor como controlador de domínio, um servidor *SQL*, um servidor *hyper-v* e uma máquina cliente. Para testarmos a virtualização utilizamos o *VMware Workstation 11* como *hypervisor* do tipo II, e iremos utilizar como sistema operativo a instalar nos três servidores o *Windows Server 2012*.

Para utilizarmos esses três servidores necessitamos de:

DC

- 1 processador
- 2GB memória RAM
- 60GB Disco
- NIC – Host-only

Servidor-SQL

- 1 processador
- 2GB memória RAM
- 60GB Disco
- NIC – Host-only

Hyper-V

- 1 processador
- 4GB memória RAM
- 150GB Disco
- NIC – Host-only

Máquina Virtual DC

Na máquina virtual DC foi instalado o sistema operativo *Windows Server 2012*, esta máquina foi a máquina escolhida para ser promovida a controlador de domínio em que o domínio escolhido foi *virtualDC.local*, e todas máquinas vão estar neste domínio. No DC foram criados nos utilizadores do *Active Directory* os dois tipos de utilizadores que vão poder aceder á aplicação, um como utilizador que apenas pode consultar e outro como Administrador que pode executar várias ações na aplicação.

Máquina Virtual servidor-SQL

Foi também instalado o *Windows server 2012*, e adicionado ao DC, sendo que o que irá correr dentro desta máquina será o SQL, onde será feito todo o armazenamento e estrutura das base de dados. Para que os administradores tenham todas as permissões na execução de comandos foi feito o login no *SQL server* como sysAdmin.

Máquina Virtual Hyper-v

A máquina virtual *Hyper-V* será uma máquina também com a instalação do *Windows Server 2012*, sendo que tem a role *Hyper-V* instalada de modo a que possamos virtualizar a máquina cliente com a instalação do *Windows 8.1*.

Máquina Virtual cliente

A máquina cliente é a máquina virtualizada pela máquina virtual *Hyper-V*, onde é feita virtualização sobre virtualização. É nesta máquina que será instalado o *Windows 8.1* de modo a que possamos correr a aplicação criada, chamada de biblioteca virtual. No ambiente de trabalho desta máquina existe um atalho que inicia a aplicação.

Esta máquina está adicionada ao *virtualDC.local* de modo a que todas as credenciais de utilizador sejam validadas pelo *Active Directory*, e só pode aceder se pertencer aos *users and computers*.

Conclusão

A realização deste trabalho foi muito enriquecedora visto que permitiu-me enriquecer todos os conhecimentos adquiridos ao longo da licenciatura, tendo sido as unidades curriculares de programação, Sistema de Gestão de Base de dados, Administração de Redes as mais importantes para o desenvolvimento deste projeto, mas não sendo a pós-graduação menos importante no desenvolvimento de algumas fases deste projeto.

Assim como falei na introdução deste projeto, o objetivo deste trabalho era a criação de uma aplicação que tivesse dois tipos de contas, a conta de utilizador que apenas podia consultar e descarregar pdf, e a do Administrador que podia editar, inserir, remover, e penso que esse objetivo foi concluído com êxito. Não menos importante é a criação do ambiente virtual, onde tinha como objetivo criar os três tipos de servidores, DC, SQL e Hyper-V que tem uma máquina cliente virtualizada, onde iremos testar a aplicação e assim simular um ambiente real de uma empresa.

Concluindo todo o desenvolvimento da aplicação e do ambiente virtualizado, todos os objetivos foram cumpridos, foi utilizada a linguagem c# para o desenvolvimento da aplicação, que foi desenvolvida em *WPF Application*, e foi criado o ambiente virtual, com a criação dos três servidores DC, servidor-SQL, Hyper-V, todos eles instalados com o *Windows server 2012*, e a trabalhar num hypervisor do tipo II, com o *VMware Workstation 11*, penso que no fim deste projeto podemos ter uma noção da contenção de custos que uma empresa podia desenvolver com uma plataforma equivalente.

Bibliografia

Canal Comstor

<http://blogbrasil.comstor.com/bid/294730/O-que-SaaS-PaaS-e-IaaS>

Chamberlin, D. D., Astrahan, M. M., Eswaran, K. P., Griffiths, P. P., Lorie, R. A., Mehl, J. W., . . .

Wade,

IMBS DeveloperWorks (29/junho/2011)

<http://www.ibm.com/developerworks/br/library/l-virtual-machine-architectures/>

Malik, S. (2005). Pro ADO.NET 2.0. USA: Apress.

Svetlin Nakov, Veselin Kolev & CO (2013) Fundamentals Of Computer Programming with c#

Anexos

Pagina Utilizador

XAML

```

<Window x:Class="Teste.utilizador"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="utilizador" Height="600" Width="800">
    <Window.Background>
        <ImageBrush ImageSource="User.png"/>
    </Window.Background>
    <Window.BorderBrush>
        <ImageBrush/>
    </Window.BorderBrush>
    <Window.OpacityMask>
        <ImageBrush ImageSource="User.png" Stretch="UniformToFill"/>
    </Window.OpacityMask>
    <Grid>
        <Grid.OpacityMask>
            <ImageBrush ImageSource="User.png"/>
        </Grid.OpacityMask>
        <TextBox x:Name="txtPesquisa" HorizontalAlignment="Left" Height="23"
Margin="32,26,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="142"
BorderBrush="#FF476D85" BorderThickness="2"
PreviewKeyDown="txtPesquisa_PreviewKeyDown" >
            <TextBox.Background>
                <SolidColorBrush Color="#FFCBC5BE" Opacity="0.5"/>
            </TextBox.Background>
        </TextBox>
        <TextBox Name="txtISBN" IsReadOnly="True" HorizontalAlignment="Left"
Height="23" Margin="130,145,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="142" Foreground="#FF476D85" BorderBrush="#FF476D85" BorderThickness="2">
            <TextBox.Background>
                <SolidColorBrush Color="#FFCBC5BE" Opacity="0.5"/>
            </TextBox.Background>
        </TextBox>
        <TextBox Name="txtNome" IsReadOnly="True" HorizontalAlignment="Left"
Height="23" Margin="130,185,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="183" BorderBrush="#FF476D85" BorderThickness="2">
            <TextBox.Background>
                <SolidColorBrush Color="#FFCBC5BE" Opacity="0.5"/>
            </TextBox.Background>
        </TextBox>
        <TextBox Name="txtAutor" IsReadOnly="True" HorizontalAlignment="Left"
Height="25" Margin="130,225,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="183" BorderThickness="2" BorderBrush="#FF476D85">
            <TextBox.Background>
                <SolidColorBrush Color="#FFCBC5BE" Opacity="0.5"/>
            </TextBox.Background>
        </TextBox>
        <TextBox Name="txtSinopse" VerticalScrollBarVisibility="Auto"
HorizontalAlignment="Left" Height="448" Margin="356,55,0,0" Padding="10"
TextWrapping="WrapWithOverflow" VerticalAlignment="Top" Width="370"

```

```

    BorderBrush="#FF476D85" BorderThickness="2" FontFamily="Book Antiqua"
    TextAlignment="Center" FontSize="14">
        <TextBox.Background>
            <SolidColorBrush Color="#FFCBC5BE" Opacity="0.97"/>
        </TextBox.Background>
    </TextBox>

    <Label Content="ISBN" HorizontalAlignment="Left" Margin="34,143,0,0"
    VerticalAlignment="Top" FontWeight="Bold" FontFamily="Book Antiqua"/>
    <Label Content="Titulo do Livro" HorizontalAlignment="Left"
Margin="30,182,0,0" VerticalAlignment="Top" FontFamily="Book Antiqua"
FontWeight="Bold"/>
    <Label Content="Autor do Livro" HorizontalAlignment="Left"
Margin="30,224,0,0" VerticalAlignment="Top" FontWeight="Bold" FontFamily="Book
Antiqua"/>
    <Label Name="lblSinopse" Content="Sinopse" HorizontalAlignment="Left"
Margin="400,217,0,0" VerticalAlignment="Top" RenderTransformOrigin="0.605,-0.308"
Width="281" FontFamily="Book Antiqua" FontWeight="Bold" BorderThickness="1"
Foreground="#FFF39200" FontSize="72" Background="{x:Null}" Height="103"
Panel.ZIndex="1" Opacity="0.3"/>

    <Image Name="img1" HorizontalAlignment="Left" Height="194"
Margin="68,309,0,0" VerticalAlignment="Top" Width="180" Stretch="Fill"
OpacityMask="Black"/>

    <Rectangle HorizontalAlignment="Left" Height="201" Margin="68,302,0,0"
Stroke="#FF476D85" VerticalAlignment="Top" Width="180" StrokeThickness="2"/>

    <DataGrid Name="dgResultado" HeadersVisibility="None"
GridLinesVisibility="None" HorizontalAlignment="Left" Margin="32,68,0,0"
VerticalAlignment="Top" Height="56" Width="223"
PreviewMouseUp="dgResultado_PreviewMouseUp"/>

    <Button Content="Descarregar" HorizontalAlignment="Left"
Margin="235,519,0,0" VerticalAlignment="Top" Width="115" BorderBrush="#FFF39200"
BorderThickness="2" FontFamily="Book Antiqua" FontWeight="Bold" FontSize="16"
Height="34" Click="Button_Click" >
        <Button.Background>
            <SolidColorBrush Color="#FF476D85" Opacity="0.3"/>
        </Button.Background>
    </Button>
    <TextBox Visibility="Hidden" x:Name="txtPdf" IsReadOnly="True"
HorizontalAlignment="Left" Height="25" Margin="376,24,0,0" TextWrapping="Wrap"
VerticalAlignment="Top" Width="183" BorderThickness="2" BorderBrush="#FF476D85">
        <TextBox.Background>
            <SolidColorBrush Color="#FFCBC5BE" Opacity="0.5"/>
        </TextBox.Background>
    </TextBox>

</Grid>
</Window>

```

C#

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;

```

Pedro Gonçalves

```

using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using System.IO;
using System.Data.SqlClient;
using System.Data;
using System.Drawing;

namespace Teste
{
    /// <summary>
    /// Interaction logic for utilizador.xaml
    /// </summary>
    public partial class utilizador : Window
    {
        public utilizador()
        {
            InitializeComponent();
            txtNome.IsReadOnly = true;
        }

        private void dgResultado_PreviewMouseUp(object sender,
MouseButtonEventArgs e)
        {
            SqlConnection sql_conn;
            sql_conn = new SqlConnection(@"Data Source=pc-ric;Initial
Catalog=Livros;Integrated Security=True");

            try
            {
                sql_conn.Open();

                lblSinopse.Visibility = Visibility.Hidden;
                string query = "SELECT * FROM livros";

                SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
                cmdlogin.ExecuteNonQuery();
                SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
                DataTable dt = new DataTable("livros");
                DataSet ds = new DataSet();
                da.Fill(ds);
                da.Fill(dt);
                var nome = dgResultado.SelectedIndex;
                DataRow selectedRow = dt.Rows[nome];

                txtISBN.Text = selectedRow["isbn"].ToString();
                txtNome.Text = selectedRow["titulo"].ToString();
                txtAutor.Text = selectedRow["nome_autor"].ToString();
                txtSinopse.Text = selectedRow["resumo"].ToString();
                txtPdf.Text = txtNome.Text;

                byte[] data = (byte[])selectedRow["imagem"];
                MemoryStream strm = new MemoryStream();
            }
        }
    }
}

```

```

        strm.Write(data, 0, data.Length);
        strm.Position = 0;
        System.Drawing.Image img = System.Drawing.Image.FromStream(strm);

        BitmapImage bi = new BitmapImage();
        bi.BeginInit();
        MemoryStream ms = new MemoryStream();
        img.Save(ms, System.Drawing.Imaging.ImageFormat.Bmp);
        ms.Seek(0, SeekOrigin.Begin);
        bi.StreamSource = ms;
        bi.EndInit();
        img1.Source = bi;

    }

    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

private void txtPesquisa_PreviewKeyDown(object sender, KeyEventArgs e)
{
    SqlConnection sql_conn;
    sql_conn = new SqlConnection(@"Data Source=pc-ric;Initial
Catalog=Livros;Integrated Security=True");

    try
    {

        sql_conn.Open();
        string query = "SELECT isbn,titulo,nome_autor,nome_editora FROM livros where isbn
like'" + 
        this.txtPesquisa.Text + "' OR titulo like'" + this.txtPesquisa.Text + "%' OR
nome_autor like'" + this.txtPesquisa.Text + "%' OR nome_editora like'" +
this.txtPesquisa.Text + "%'";
        SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
        cmdlogin.ExecuteNonQuery();

        txtAutor.Clear();
        txtISBN.Clear();
        txtNome.Clear();
        txtSinopse.Clear();

        SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
        DataTable dt = new DataTable("livros");
        da.Fill(dt);

        dgResultado.ItemsSource = dt.DefaultView;
        da.Update(dt);
        sql_conn.Close();
    }
    catch (Exception ex)
    {
}

```

```
        MessageBox.Show(ex.Message);
    }

private void Button_Click(object sender, RoutedEventArgs e)
{
    string caminhoPdf = txtPdf.Text;

    try
    {
        System.Diagnostics.Process.Start("D:\\\" + caminhoPdf + ".pdf");
    }
    catch (Exception ex)
    {
        MessageBox.Show("Não é possível apresentar o PDF");
    }
}

}
```

Pagina Livros

XAML

```

<Window
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="clr-namespace:Teste" x:Class="Teste.MainWindow"
    Title="MainWindow" Height="600" Width="800" Loaded="Window_Loaded"
    WindowStartupLocation="CenterScreen">

    <Window.Background>
        <ImageBrush ImageSource="Fundo.png"/>
    </Window.Background>
    <Grid Width="800" Height="800">
        <Grid.RowDefinitions>
            <RowDefinition Height="176*"/>
            <RowDefinition Height="149*"/>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition/>
            <ColumnDefinition Width="0*"/>
        </Grid.ColumnDefinitions>

        <TextBox Name="txtISBN" HorizontalAlignment="Left" Height="23"
Margin="137,28,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
BorderBrush="#FFB67551" Background="#FFCBC5BE"/>
        <TextBox Name="txtTitulo" HorizontalAlignment="Left" Height="23"
Margin="137,59,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
Background="#FFCBC5BE" BorderBrush="#FFB67551"/>
        <TextBox Name="txtSinopse" HorizontalAlignment="Left" Height="232"
Margin="22,239,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="159"
Background="#FFCBC5BE" BorderBrush="#FFB67551" Grid.RowSpan="2"/>
        <TextBox x:Name="txtPesquisa" HorizontalAlignment="Left" Height="23"
Margin="508,28,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="177"
BorderBrush="#FFB67551" Background="#FFCBC5BE"
PreviewKeyDown="txtPesquisa_PreviewKeyDown"/>

        <Button Name="btnSalvar" Content="Adicionar" HorizontalAlignment="Left"
Margin="44,77,0,0" VerticalAlignment="Top" Width="75" Click="btnSalvar_Click"
Grid.Row="1"/>
        <Button Name="btnAlterar" Content="Alterar" HorizontalAlignment="Left"
Margin="169,77,0,270" Width="75" Click="btnAlterar_Click" Grid.Row="1"/>
        <Button Name="btnRemover" Content="Remover" HorizontalAlignment="Left"
Margin="297,77,0,0" VerticalAlignment="Top" Width="75" Click="btnRemover_Click"
Grid.Row="1"/>
        <Button Name="btnAtualizar" Content="Atualizar Dados"
HorizontalAlignment="Left" Margin="545,408,0,0" VerticalAlignment="Top"
Width="104" Click="btnAtualizar_Click"/>
        <Button Name="btnProcurar" Content="Procurar" HorizontalAlignment="Left"
Margin="156,167,0,0" VerticalAlignment="Top" Width="75"
Click="btnProcurar_Click"/>

        <Label Content="ISBN" HorizontalAlignment="Left" Margin="32,25,0,0"
VerticalAlignment="Top"/>
        <Label Content="Titulo" HorizontalAlignment="Left" Margin="32,56,0,0"
VerticalAlignment="Top"/>
        <Label Content="Autor" HorizontalAlignment="Left" Margin="32,87,0,0"
VerticalAlignment="Top"/>
        <Label Content="Sinopse" HorizontalAlignment="Left" Margin="66,208,0,0"
VerticalAlignment="Top" RenderTransformOrigin="0.605,-0.308"/>

```

```

        <Label Content="Imagen" HorizontalAlignment="Left" Margin="22,161,0,0"
VerticalAlignment="Top"/>
        <Label Content="Imagen" HorizontalAlignment="Left" Margin="270,208,0,0"
VerticalAlignment="Top"/>
        <Label Content="Editora" HorizontalAlignment="Left" Margin="32,118,0,0"
VerticalAlignment="Top"/>
        <Label Content="Pesquisar:" HorizontalAlignment="Left"
Margin="429,25,0,0" VerticalAlignment="Top" Width="74"/>

        <DataGrid Name="dg1" IsReadOnly="True" AutoGenerateColumns="True"
HorizontalAlignment="Left" Margin="429,70,0,0" VerticalAlignment="Top"
Height="289" Width="331" BorderBrush="#FFB67551" Background="#FFCBC5BE"
MouseDoubleClick="dg1_MouseDoubleClick" CanUserResizeRows="False"/>

        <Image Name="imgPreview" HorizontalAlignment="Left" Height="115"
Margin="212,239,0,0" VerticalAlignment="Top" Width="160" Source="{Binding}"/>

        <ComboBox Name="cmbAutor" HorizontalAlignment="Left" Margin="137,91,0,0"
VerticalAlignment="Top" Width="120"/>
        <ComboBox Name="cmbEditora" HorizontalAlignment="Left"
Margin="137,122,0,0" VerticalAlignment="Top" Width="120"/>
<DockPanel Height="20" VerticalAlignment="Top">
        <Menu Name="MainMenu" DockPanel.Dock="Top">
            <MenuItem Name="menuAutores" Header="Autores"
Click="menuAutores_Click"/></MenuItem>
            <MenuItem Name="menuEditora" Header="Editoras"
Click="menuEditora_Click"/></MenuItem>

        </Menu>
    </DockPanel>
</Grid>
</Window>

```

C#

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using System.Data.SqlClient;
using System.Data;
using System.IO;
using System.Configuration;

namespace Teste
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {

```

```

public MainWindow()
{
    InitializeComponent();
}
private void Window_Loaded(object sender, RoutedEventArgs e)
{
    string connectionString = null;

    SqlConnection sql_conn;
    connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
    sql_conn = new SqlConnection(connectionString);
    sql_conn.Open();

    //Comboautor
    string query = "SELECT nome_autor FROM autor";
    SqlCommand cmdlogin = new SqlCommand(query, sql_conn);

    SqlDataReader dr = cmdlogin.ExecuteReader();
    while (dr.Read())
    {
        string sAutor = dr.GetString(0);
        cmbAutor.Items.Add(sAutor);
    }
    sql_conn.Close();
    sql_conn.Open();

    //ComboEditora
    string query1 = "SELECT nome_editora FROM editora";
    SqlCommand cmdlogin1 = new SqlCommand(query1, sql_conn);

    SqlDataReader dr1 = cmdlogin1.ExecuteReader();
    while (dr1.Read())
    {
        string sEditora = dr1.GetString(0);
        cmbEditora.Items.Add(sEditora);
    }
    sql_conn.Close();
}

string caminho = "";

private void btnSalvar_Click(object sender, RoutedEventArgs e)
{
    MessageBoxResult result = MessageBox.Show("Tem a Certeza?", "Informação",
    MessageBoxButton.YesNoCancel);
    switch (result)
    {
        case MessageBoxResult.Yes:
            string connectionString = null;
            SqlConnection sql_conn;

```

```

        connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
    sql_conn = new SqlConnection(connectionString);

        byte[] imageData;
        imageData = File.ReadAllBytes(caminho);
        try
        {
            sql_conn.Open();
string query = "insert into livros
(isbn,titulo,nome_autor,nome_editora,resumo,imagem,id_editora) values('" +
this.txtISBN.Text + "','" +
        this.txtTitulo.Text + "','" + cmbAutor.SelectedItem.ToString() + "','" +
cmbEditora.SelectedItem.ToString() + "','" + this.txtSinopse.Text +
"',@DATA,'2')";
            SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
            cmdlogin.Parameters.Add("@DATA", imageData);
            cmdlogin.ExecuteNonQuery();
            sql_conn.Close();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }

        MessageBox.Show("Conteúdo Guardado");
        break;
        case MessageBoxResult.No:
        MessageBox.Show("Conteúdo não Guardado");
        break;
    }
}

private void btnAlterar_Click(object sender, RoutedEventArgs e)
{
    MessageBoxResult result1 = MessageBox.Show("Tem a Certeza?", "Informação", MessageBoxButton.YesNoCancel);
    switch (result1)
    {
        case MessageBoxResult.Yes:

            string connectionString = null;
            SqlConnection sql_conn;
            connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
            sql_conn = new SqlConnection(connectionString);
            byte[] imageData;
            imageData = File.ReadAllBytes(caminho);
            try
            {
                sql_conn.Open();
string query = "UPDATE livros set isbn='" + this.txtISBN.Text + "',titulo='" +
this.txtTitulo.Text + "',nome_autor='" + this.cmbAutor.SelectedItem.ToString() + "'",
nome_editora='"' +
            this.cmbEditora.SelectedItem.ToString() + "',resumo=''" +
this.txtSinopse.Text+ "',imagem=@DATA where isbn='" + this.txtISBN.Text + "'";

                SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
                cmdlogin.Parameters.Add("@DATA", imageData);
                cmdlogin.ExecuteNonQuery();
            }

```

```

        sql_conn.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    MessageBox.Show("Conteúdo Alterado");
    break;

    case MessageBoxResult.No:
        MessageBox.Show("Conteúdo não Alterado");
        break;
    }
}

private void btnRemover_Click(object sender, RoutedEventArgs e)
{
    MessageBoxResult result2 = MessageBox.Show("Tem a Certeza?", "Informação", MessageBoxButton.YesNoCancel);

    switch (result2)
    {
        case MessageBoxResult.Yes:

            string connectionString = null;

            SqlConnection sql_conn;
            connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
            sql_conn = new SqlConnection(connectionString);

            try
            {
                sql_conn.Open();
                string query = "DELETE FROM livros WHERE isbn='"
this.txtISBN.Text + "'";
                SqlCommand cmdlogin = new SqlCommand(query, sql_conn);

                cmdlogin.ExecuteNonQuery();

                sql_conn.Close();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            MessageBox.Show("Conteúdo Removido");
            break;

        case MessageBoxResult.No:
            MessageBox.Show("Conteúdo Não Removido");
            break;
    }
}

private void btnAtualizar_Click(object sender, RoutedEventArgs e)
{

```

```

        connectionString = null;
        SqlConnection sql_conn;
        connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
        sql_conn = new SqlConnection(connectionString);

        try
        {
            sql_conn.Open();
            string query = "SELECT * FROM Livros";
            SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
            cmdlogin.ExecuteNonQuery();

            SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
            DataTable dt = new DataTable("autores");
            da.Fill(dt);

            dg1.ItemsSource = dt.DefaultView;
            da.Update(dt);

            this.dg1.Columns[0].Header = "ISBN";
            this.dg1.Columns[1].Header = "Titulo";
            this.dg1.Columns[2].Header = "Nome do Autor";
            this.dg1.Columns[3].Header = "Editora";
            this.dg1.Columns[4].Visibility = Visibility.Hidden;
            this.dg1.Columns[5].Visibility = Visibility.Hidden;
            this.dg1.Columns[6].Visibility = Visibility.Hidden;

            sql_conn.Close();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }

    private void btnProcurar_Click(object sender, RoutedEventArgs e)
    {
        Microsoft.Win32.OpenFileDialog openFileDialog1 = new
Microsoft.Win32.OpenFileDialog();
        openFileDialog1.ShowDialog();
        openFileDialog1.Filter = "JPEG Files (*.jpeg)|*.jpeg";
        openFileDialog1.DefaultExt = ".jpeg";

        caminho = openFileDialog1.FileName;
        ImageSource imageSource = new BitmapImage(new Uri(caminho));
        imgPreview.Source = imageSource;
    }

    private void dg1_MouseDoubleClick(object sender, MouseButtonEventArgs e)
    {
        string connectionString = null;
        SqlConnection sql_conn;

```

```

        connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
        sql_conn = new SqlConnection(connectionString);
        try
        {sql_conn.Open();
            string query = "SELECT * FROM livros";
            SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
            cmdlogin.ExecuteNonQuery();
            SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
            DataTable dt = new DataTable("livros");
            DataSet ds = new DataSet();
            da.Fill(ds);
            da.Fill(dt);
            var nome = dg1.SelectedIndex;
            DataRow selectedRow = dt.Rows[nome];
            txtISBN.Text = selectedRow["isbn"].ToString();
            txtTitulo.Text = selectedRow["titulo"].ToString();
            txtSinopse.Text = selectedRow["resumo"].ToString();
            cmbAutor.SelectedItem = selectedRow["nome_autor"].ToString();
            cmbEditora.SelectedItem = selectedRow["nome_editora"].ToString();
            byte[] data = (byte[])selectedRow["imagem"];
            MemoryStream strm = new MemoryStream();
            strm.Write(data, 0, data.Length);
            strm.Position = 0;
            System.Drawing.Image img = System.Drawing.Image.FromStream(strm);
            BitmapImage bi = new BitmapImage();
            bi.BeginInit();
            MemoryStream ms = new MemoryStream();
            img.Save(ms, System.Drawing.Imaging.ImageFormat.Bmp);
            ms.Seek(0, SeekOrigin.Begin);
            bi.StreamSource = ms;
            bi.EndInit();
            imgPreview.Source = bi;}
        catch (Exception ex)
        { MessageBox.Show(ex.Message);}
    }

private void txtPesquisa_PreviewKeyDown(object sender, KeyEventArgs e)
{
    string connectionString = null;

    SqlConnection sql_conn;
    connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
    sql_conn = new SqlConnection(connectionString);

    try
    {

        sql_conn.Open();
        string query = "SELECT isbn,titulo,nome_autor,nome_editora FROM
livros where isbn like'" + this.txtPesquisa.Text + "%' OR titulo like'" +
            this.txtPesquisa.Text + "%' OR nome_autor like'" +
            this.txtPesquisa.Text + "%' OR nome_editora like'" + this.txtPesquisa.Text +
            "%'";
        SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
        cmdlogin.ExecuteNonQuery();
    }
}

```

```
SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
DataTable dt = new DataTable("autor");
da.Fill(dt);

dg1.ItemsSource = dt.DefaultView;

this.dg1.Columns[0].Header = "ISBN";
this.dg1.Columns[1].Header = "Titulo";
this.dg1.Columns[2].Header = "Nome do Autor";
this.dg1.Columns[3].Header = "Editora";
da.Update(dt);

sql_conn.Close();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
}

private void menuAutores_Click(object sender, RoutedEventArgs e)
{
    this.Hide();
    autores autoresnovo = new autores();
    autoresnovo.Show();
}

private void menuEditora_Click(object sender, RoutedEventArgs e)
{
    this.Hide();
    editoras editorasnovo = new editoras();
    editorasnovo.Show();
} } }
```

Pagina Editoras

XAML

```

<Window x:Class="Teste.editoras"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="editoras" Height="600" Width="800"
    WindowStartupLocation="CenterScreen">
    <Window.Background>
        <ImageBrush ImageSource="Fundo.png"/>
    </Window.Background>

    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="176*"/>
            <RowDefinition Height="149*"/>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition/>
            <ColumnDefinition Width="0*"/>
        </Grid.ColumnDefinitions>

        <DataGrid IsReadOnly="True" Name="dg1" AutoGenerateColumns="True"
HorizontalAlignment="Left" Margin="429,70,0,0" VerticalAlignment="Top"
Height="289" Width="331" BorderBrush="#FFB67551" Background="#FFCBC5BE"
CanUserResizeRows="False" MouseDoubleClick="dg1_MouseDoubleClick"/>

        <Button Name="btnAlterar" Content="Alterar" HorizontalAlignment="Left"
Margin="159,10,0,0" VerticalAlignment="Top" Width="75" Grid.Row="1"
Click="btnAlterar_Click" />
        <Button Name="btnRemover" Content="Remover" HorizontalAlignment="Left"
Margin="255,10,0,0" VerticalAlignment="Top" Width="75" Grid.Row="1"
Click="btnRemover_Click" />
        <Button Name="btnSalvar" Content="Adicionar" HorizontalAlignment="Left"
Margin="65,10,0,0" VerticalAlignment="Top" Width="75" Grid.Row="1"
Click="btnSalvar_Click" />
        <Button Name="btnAtualizar" Content="Atualizar Dados"
HorizontalAlignment="Left" Margin="544,56,0,0" VerticalAlignment="Top"
Width="104" Height="20" Grid.Row="1" RenderTransformOrigin="0.433,-0.792"
Click="btnAtualizar_Click" />

        <TextBox Name="txtNomeEditora" HorizontalAlignment="Left" Height="23"
Margin="194,136,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
BorderBrush="#FFB67551" Background="#FFCBC5BE"/>
        <TextBox Name="txtMorada" HorizontalAlignment="Left" Height="23"
Margin="194,167,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
Background="#FFCBC5BE" BorderBrush="#FFB67551"/>
        <TextBox x:Name="txtPesquisa" HorizontalAlignment="Left" Height="23"
Margin="508,28,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="177"
BorderBrush="#FFB67551" Background="#FFCBC5BE"
PreviewKeyDown="txtPesquisa_PreviewKeyDown" />
        <TextBox x:Name="txtTelefone" HorizontalAlignment="Left" Height="23"
Margin="194,198,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
Background="#FFCBC5BE" BorderBrush="#FFB67551"/>
        <TextBox x:Name="txtIdEditora" Visibility="Hidden"
HorizontalAlignment="Left" Height="23" Margin="194,92,0,0" TextWrapping="Wrap"
VerticalAlignment="Top" Width="120" BorderBrush="#FFB67551"
Background="#FFCBC5BE"/>

        <Label Content="Pesquisar:" HorizontalAlignment="Left"
Margin="429,25,0,0" VerticalAlignment="Top" Width="74"/>

```

```

        <Label Content="Editora" HorizontalAlignment="Left" Margin="89,133,0,0"
VerticalAlignment="Top"/>
        <Label Content="Morada" HorizontalAlignment="Left" Margin="89,164,0,0"
VerticalAlignment="Top"/>
        <Label Content="Telefone" HorizontalAlignment="Left" Margin="89,195,0,0"
VerticalAlignment="Top"/>

        <DockPanel Height="20" VerticalAlignment="Top">
            <Menu Name="MainMenu" DockPanel.Dock="Top">
                <MenuItem Name="menuLivros" Header="Livros"
Click="menuLivros_Click" /></MenuItem>
                <MenuItem Name="menuAutores" Header="Autores"
Click="menuAutores_Click" /></MenuItem>
            </Menu>
        </DockPanel>
    </Grid>
</Window>

```

C#

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using System.Data.SqlClient;
using System.Data;
using System.IO;
using System.Configuration;

namespace Teste
{
    /// <summary>
    /// Interaction logic for editoras.xaml
    /// </summary>
    public partial class editoras : Window
    {
        public editoras()
        {
            InitializeComponent();
        }

        private void btnAtualizar_Click(object sender, RoutedEventArgs e)
        {
            string connectionString = null;

            SqlConnection sql_conn;
            connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
            sql_conn = new SqlConnection(connectionString);

```

```

try
{
    sql_conn.Open();
    string query = "SELECT * FROM editora";
    SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
    cmdlogin.ExecuteNonQuery();

    SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
    DataTable dt = new DataTable("editora");
    da.Fill(dt);

    dg1.ItemsSource = dt.DefaultView;
    da.Update(dt);

    this.dg1.Columns[0].Visibility = Visibility.Hidden;

    this.dg1.Columns[1].Header = "Editora";
    this.dg1.Columns[2].Header = "Morada";
    this.dg1.Columns[3].Header = "Telefone";

    sql_conn.Close();
}

catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
}

private void dg1_MouseDoubleClick(object sender, MouseButtonEventArgs e)
{
    string connectionString = null;

    SqlConnection sql_conn;
    connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectionString1"].ConnectionString;
    sql_conn = new SqlConnection(connectionString);
    try
    {
        sql_conn.Open();

        string query = "SELECT * FROM editora";

        SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
        cmdlogin.ExecuteNonQuery();
        SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
        DataTable dt = new DataTable("editora");
        DataSet ds = new DataSet();
        da.Fill(ds);
        da.Fill(dt);
        var nome = dg1.SelectedIndex;
        DataRow selectedRow = dt.Rows[nome];

        txtIdEditora.Text = selectedRow["id_editora"].ToString();
        txtNomeEditora.Text = selectedRow["nome_editora"].ToString();
        txtMorada.Text = selectedRow["morada"].ToString();
        txtTelefone.Text = selectedRow["telefone"].ToString();
    }
    catch (Exception ex)
    {
}
}

```

```

        MessageBox.Show(ex.Message);
    }

}

private void btnSalvar_Click(object sender, RoutedEventArgs e)
{
    MessageBoxResult result = MessageBox.Show("Tem a Certeza?", "Informação", MessageBoxButton.YesNoCancel);

    switch (result)
    {
        case MessageBoxResult.Yes:
            string connectionString = null;

            SqlConnection sql_conn;
            connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
            sql_conn = new SqlConnection(connectionString);

            try
            {
                sql_conn.Open();
                string query = "insert into editora
(nome_editora,morada,telefone) values('" + this.txtNomeEditora.Text + "','" +
this.txtMorada.Text + "','" + this.txtTelefone.Text +
"')";
                SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
                cmdlogin.ExecuteNonQuery();

                sql_conn.Close();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }

            MessageBox.Show("Conteúdo Guardado");
            break;
        case MessageBoxResult.No:
            MessageBox.Show("Conteúdo não Guardado");
            break;
    }
}

private void btnAlterar_Click(object sender, RoutedEventArgs e)
{
    MessageBoxResult result1 = MessageBox.Show("Tem a Certeza?", "Informação", MessageBoxButton.YesNoCancel);

    switch (result1)
    {
        case MessageBoxResult.Yes:

            string connectionString = null;

            SqlConnection sql_conn;
            connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;

```

```

        sql_conn = new SqlConnection(connectionString);
        try
        {
            sql_conn.Open();
            string query = "UPDATE editora set nome_editora='" +
this.txtNomeEditora.Text + "',morada='" + this.txtMorada.Text + "',telefone=''" +
                this.txtTelefone.Text + "'where id_editora=''" +
this.txtIdEditora.Text + "' ";
            SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
            cmdlogin.ExecuteNonQuery();

            sql_conn.Close();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
        MessageBox.Show("Conteúdo Alterado");
        break;

        case MessageBoxResult.No:
            MessageBox.Show("Conteúdo não Alterado");
            break;
    }
}

private void btnRemover_Click(object sender, RoutedEventArgs e)
{
    MessageBoxResult result2 = MessageBox.Show("Tem a Certeza?", "Informação", MessageBoxButton.YesNoCancel);
    switch (result2)
    {
        case MessageBoxResult.Yes:

            string connectionString = null;

            SqlConnection sql_conn;
            connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
            sql_conn = new SqlConnection(connectionString);

            try
            {
                sql_conn.Open();
                string query = "DELETE FROM editora WHERE id_editora=''" +
this.txtIdEditora.Text + "'";
                SqlCommand cmdlogin = new SqlCommand(query, sql_conn);

                cmdlogin.ExecuteNonQuery();

                sql_conn.Close();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            MessageBox.Show("Conteúdo Removido");
            break;
    }
}

```

```

        case MessageBoxResult.No:
            MessageBox.Show("Conteúdo Não Removido");
            break;
    }

}

private void txtPesquisa_PreviewKeyDown(object sender, KeyEventArgs e)
{
    string connectionString = null;

    SqlConnection sql_conn;
    connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
    sql_conn = new SqlConnection(connectionString);
    try
    {

        sql_conn.Open();
        string query = "SELECT nome_editora,morada,telefone FROM editora
where nome_editora like'" + this.txtPesquisa.Text + "%' or morada like'" +
this.txtPesquisa.Text + "%'";
        SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
        cmdlogin.ExecuteNonQuery();
        SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
        DataTable dt = new DataTable("editora");
        da.Fill(dt);

        dg1.ItemsSource = dt.DefaultView;
        this.dg1.Columns[0].Header = "Editora";
        this.dg1.Columns[1].Header = "Morada";
        this.dg1.Columns[2].Header = "Telefone";
        da.Update(dt);

        sql_conn.Close();

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

private void menuLivros_Click(object sender, RoutedEventArgs e)
{
    this.Hide();
    MainWindow livros = new MainWindow();
    livros.Show();

}

private void menuAutores_Click(object sender, RoutedEventArgs e)
{
    this.Hide();
    autores autoresnovo = new autores();
}

```

```
        autoresnovo.Show();  
    }  
  
}  
}
```

Página Autores

XAML

```

<Window x:Class="Teste.autores"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="autores" Height="600" Width="800"
        WindowStartupLocation="CenterScreen">

    <Window.Background>
        <ImageBrush ImageSource="Fundo.png"/>
    </Window.Background>

    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="176*"/>
            <RowDefinition Height="149*"/>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition/>
            <ColumnDefinition Width="0*"/>
        </Grid.ColumnDefinitions>

        <DataGrid IsReadOnly="True" Name="dg1" AutoGenerateColumns="True"
HorizontalAlignment="Left" Margin="429,70,0,0" VerticalAlignment="Top"
Height="289" Width="331" BorderBrush="#FFB67551" Background="#FFCBC5BE"
MouseDoubleClick="dg1_MouseDoubleClick" CanUserResizeRows="False"/>

        <Button Name="btnAlterar" Content="Alterar" HorizontalAlignment="Left"
Margin="162,10,0,0" VerticalAlignment="Top" Width="75" Grid.Row="1"
Click="btnAlterar_Click"/>
        <Button Name="btnRemover" Content="Remover" HorizontalAlignment="Left"
Margin="255,10,0,0" VerticalAlignment="Top" Width="75" Grid.Row="1"
Click="btnRemover_Click"/>
        <Button Name="btnSalvar" Content="Adicionar" HorizontalAlignment="Left"
Margin="65,10,0,0" VerticalAlignment="Top" Width="75" Grid.Row="1"
Click="btnSalvar_Click"/>
        <Button Name="btnAtualizar" Content="Atualizar Dados"
HorizontalAlignment="Left" Margin="544,56,0,0" VerticalAlignment="Top"
Width="104" Height="20" Grid.Row="1" RenderTransformOrigin="0.433,-0.792"
Click="btnAtualizar_Click_1" />

        <TextBox Name="txtNomeAutor" HorizontalAlignment="Left" Height="23"
Margin="194,136,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
BorderBrush="#FFB67551" Background="#FFCBC5BE"/>
        <TextBox Name="txtMorada" HorizontalAlignment="Left" Height="23"
Margin="194,167,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
Background="#FFCBC5BE" BorderBrush="#FFB67551"/>
        <TextBox x:Name="txtPesquisa" HorizontalAlignment="Left" Height="23"
Margin="508,28,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="177"
BorderBrush="#FFB67551" Background="#FFCBC5BE"
PreviewKeyDown="txtPesquisa_PreviewKeyDown" />
        <TextBox x:Name="txtTelefone" HorizontalAlignment="Left" Height="23"
Margin="194,198,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
Background="#FFCBC5BE" BorderBrush="#FFB67551"/>
        <TextBox x:Name="txtIdAutor" Visibility="Hidden"
HorizontalAlignment="Left" Height="23" Margin="194,92,0,0" TextWrapping="Wrap"
VerticalAlignment="Top" Width="120" BorderBrush="#FFB67551"
Background="#FFCBC5BE"/>

```

```

        <Label Content="Pesquisar:" HorizontalAlignment="Left"
Margin="429,25,0,0" VerticalAlignment="Top" Width="74"/>
        <Label Content="Nome do Autor" HorizontalAlignment="Left"
Margin="89,133,0,0" VerticalAlignment="Top"/>
        <Label Content="Morada" HorizontalAlignment="Left" Margin="89,164,0,0"
VerticalAlignment="Top"/>
        <Label Content="Telefone" HorizontalAlignment="Left" Margin="89,195,0,0"
VerticalAlignment="Top"/>

        <DockPanel Height="20" VerticalAlignment="Top">
            <Menu Name="MainMenu" DockPanel.Dock="Top">
                <MenuItem Name="menuLivros" Header="Livros"
Click="menuLivros_Click"/></MenuItem>
                <MenuItem Name="menuEditoras" Header="Editoras"
Click="menuEditoras_Click"/></MenuItem>
            </Menu>
        </DockPanel>
    </Grid>
</Window>

```

C#

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using System.Data.SqlClient;
using System.Data;
using System.IO;
using System;
using System.Configuration;

namespace Teste
{
    /// <summary>
    /// Interaction logic for autores.xaml
    /// </summary>
    public partial class autores : Window
    {
        public autores()
        {
            InitializeComponent();
        }

        private void btnAtualizar_Click_1(object sender, RoutedEventArgs e)
        {
            string connectionString = null;
            SqlConnection sql_conn;
            connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
            Pedro Gonçalves

```

```

sql_conn = new SqlConnection(connectionString);
try
{
    sql_conn.Open();
    string query = "SELECT * FROM autor";
    SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
    cmdlogin.ExecuteNonQuery();

    SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
    DataTable dt = new DataTable("autor");
    da.Fill(dt);

    dg1.ItemsSource = dt.DefaultView;

    da.Update(dt);

    this.dg1.Columns[0].Visibility = Visibility.Hidden;
    this.dg1.Columns[1].Header = "Nome do Autor";
    this.dg1.Columns[2].Header = "Morada";
    this.dg1.Columns[3].Header = "Telefone";

    sql_conn.Close();

}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}

}

private void dg1_MouseDoubleClick(object sender, MouseButtonEventArgs e)
{
    string connectionString = null;

    SqlConnection sql_conn;
    connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
    sql_conn = new SqlConnection(connectionString);

    try
    {
        sql_conn.Open();

        string query = "SELECT * FROM autor";

        SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
        cmdlogin.ExecuteNonQuery();
        SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
        DataTable dt = new DataTable("autor");
        DataSet ds = new DataSet();
        da.Fill(ds);
        da.Fill(dt);
        var nome = dg1.SelectedIndex;
        DataRow selectedRow = dt.Rows[nome];

        txtIdAutor.Text = selectedRow["id_autor"].ToString();
        txtNomeAutor.Text = selectedRow["nome_autor"].ToString();
        txtMorada.Text= selectedRow["morada"].ToString();
        txtTelefone.Text = selectedRow["telefone"].ToString();
    }
}

```

```

        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }

    private void btnSalvar_Click(object sender, RoutedEventArgs e)
    {
        MessageBoxButtonResult result = MessageBox.Show("Tem a Certeza?", "Informação", MessageBoxButton.YesNoCancel);
        switch (result)
        {
            case MessageBoxResult.Yes:
                string connectionString = null;
                SqlConnection sql_conn;
                connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectionString1"].ConnectionString;
                sql_conn = new SqlConnection(connectionString);
                try
                {
                    sql_conn.Open();
                    string query = "insert into autor
(nome_autor,morada,telefone) values('" + this.txtNomeAutor.Text + "','" +
this.txtMorada.Text + "','" + this.txtTelefone.Text + "')";
                    SqlCommand cmdlogin = new SqlCommand(query, sql_conn);

                    cmdlogin.ExecuteNonQuery();

                    sql_conn.Close();
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message);
                }

                MessageBox.Show("Conteúdo Guardado");
                break;

            case MessageBoxResult.No:
                MessageBox.Show("Conteúdo não Guardado");
                break;
        }
    }

    private void btnAlterar_Click(object sender, RoutedEventArgs e)
    {
        MessageBoxButtonResult result1 = MessageBox.Show("Tem a Certeza?", "Informação", MessageBoxButton.YesNoCancel);
        switch (result1)
        {
            case MessageBoxResult.Yes:

                string connectionString = null;
                SqlConnection sql_conn;
                connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectionString1"].ConnectionString;

```

Pedro Gonçalves

```

        sql_conn = new SqlConnection(connectionString);
        try
        {
            sql_conn.Open();
            string query = "UPDATE autor set nome_autor='" + this.txtNomeAutor.Text +
            "',morada='"+ this.txtMorada.Text + "',telefone='"+ this.txtTelefone.Text +
            "'where id_autor='"+ this.txtIdAutor.Text + "'";
            SqlCommand cmdlogin = new SqlCommand(query, sql_conn);

            cmdlogin.ExecuteNonQuery();
            sql_conn.Close();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
        MessageBox.Show("Conteúdo Alterado");
        break;

        case MessageBoxResult.No:
            MessageBox.Show("Conteúdo não Alterado");
            break;
    }
}

private void btnRemover_Click(object sender, RoutedEventArgs e)
{
    MessageBoxResult result2 = MessageBox.Show("Tem a Certeza?", 
    "Informação", MessageBoxButton.YesNoCancel);

    switch (result2)
    {
        case MessageBoxResult.Yes:

            string connectionString = null;

            SqlConnection sql_conn;
            connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
            sql_conn = new SqlConnection(connectionString);

            try
            {
                sql_conn.Open();
                string query = "DELETE FROM autor WHERE id_autor='" +
this.txtIdAutor.Text + "'";
                SqlCommand cmdlogin = new SqlCommand(query, sql_conn);

                cmdlogin.ExecuteNonQuery();

                sql_conn.Close();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            MessageBox.Show("Conteúdo Removido");
            break;
    }
}

```

```

        case MessageBoxResult.No:
            MessageBox.Show("Conteúdo Não Removido");
            break;
    }
}

private void txtPesquisa_PreviewKeyDown(object sender, KeyEventArgs e)
{
    string connectionString = null;
    SqlConnection sql_conn;
    connectionString =
ConfigurationManager.ConnectionStrings["Teste.Properties.Settings.LivrosConnectio
nString1"].ConnectionString;
    sql_conn = new SqlConnection(connectionString);
    try
    {

        sql_conn.Open();
        string query = "SELECT nome_autor,morada,telefone FROM autor
where nome_autor like'" + this.txtPesquisa.Text + "%' OR morada like'" +
this.txtPesquisa.Text + "%'";
        SqlCommand cmdlogin = new SqlCommand(query, sql_conn);
        cmdlogin.ExecuteNonQuery();

        SqlDataAdapter da = new SqlDataAdapter(cmdlogin);
        DataTable dt = new DataTable("autor");
        da.Fill(dt);

        dg1.ItemsSource = dt.DefaultView;
        this.dg1.Columns[0].Header = "Nome do Autor";
        this.dg1.Columns[1].Header = "Morada";
        this.dg1.Columns[2].Header = "Telefone";

        da.Update(dt);

        sql_conn.Close();

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

private void menuLivros_Click(object sender, RoutedEventArgs e)
{
    this.Hide();
    MainWindow livros = new MainWindow();
    livros.Show();
}

private void menuEditoras_Click(object sender, RoutedEventArgs e)
{
    this.Hide();
    editoras editorasnovo = new editoras();
    editorasnovo.Show();
}
}

```

