Licenciatura em Informática

Turma B

Relatório

MyGym

Gestão de clientes e aulas de um ginásio

Realizado por: João Paulo Lourenço Martins nº 8144

Coordenado por: Prof. Dr. Pedro Brandão

Lisboa

2017/2018

Agradecimentos

Gostaria de agradecer o apoio por parte da minha família e da minha namorada ao longo do meu percurso académico e na realização deste projecto, sem esquecer todos os professores que me ajudaram a atingir o nível pretendido e pela disponibilidade para esclarecer dúvidas que foram surgindo, mais especificamente o Professor Pedro Crispim, o Professor José Neves e o Professor doutor Pedro Brandão.

Resumo

Este projecto tem como objectivo o desenvolvimento de uma aplicação ASP.NET em que a autenticação e a recolha de dados é feita a partir de um controlador de domínio e um servidor SQL.

O tema escolhido foi uma aplicação que possibilita inscrever e gerir aulas de grupo em um ginásio, tendo dois pontos de utilização, um a nível do cliente e outro pela parte do administrador.

As técnicas de aprendizagem e ferramentas, que foram utilizadas ao longo do curso, foram implementadas de forma a desenvolver a aplicação, a estruturar o modelo de rede e a esquematizar a base de dados.

Palavras – chave: ASP.Net, Windows Server 2012, Visual Studio, SQL Server

Abstract

This project aims to develop a ASP.NET *application in which authentication and data collection is done from a domain controller and a SQL server.*
The chosen theme was an application that allows to enroll and manage group classes in a gym, having two points of use, one at the client level and another on the part of the administrator.
The learning techniques and tools that were used throughout the course were implemented in order to develop the application, to structure the network model and to schematize the database.

Keywords: ASP .Net, Windows Server 2012, Visual Studio, SQL Server

Índice

Introdução

A Virtualização surgiu nos anos 60 e teve como principal pioneira a IBM. Esta é uma tecnologia que permite criar vários ambientes simulados ou recursos dedicados a partir de um único sistema de *hardware* físico. O software chamado *hypervisor* conecta-se directamente ao *hardware* e possibilita a divisão de um único sistema em ambientes distintos, separados e seguros, conhecidos como máquinas virtuais. Essas máquinas virtuais dependem da capacidade do *hypervisor* em emular os recursos da máquina física e distribuí-los da forma mais adequada. A tecnologia de virtualização está presente no projeto, na virtualização dos seguintes servidores: *Domain Controler*, Servidor de Routing, Servidor SQL e o computador Cliente.

A aplicação denominada por MyGym foi concebida para um ginásio, e tem como objectivo fazer a consulta e inscrição em aulas de grupo. Contem dois perfis de utilizador destintos, tais como:

- **Clientes** – têm acesso ao mapa de aulas e possibilidade de fazer uma inscrição.
- **Administrador** – responsável por adicionar/remover modalidades de aulas de grupo e gerir informação dos clientes.

A aplicação foi instalada no computador Cliente e desenvolvida no *Microsoft Visual Studio* 2015, com a linguagem de programação C# e o *design* WPF.

Estado de arte

Virtualização

Define-se virtualização como sendo a criação de um recurso virtual, como um servidor, desktop, sistema operacional, armazenamento ou uma rede, tendo como principal objetivo oferecer aos recursos utilizados uma maior escalabilidade. (Techopedia, 2018)

Apesar de ser uma ideia antiga, o seu surgimento deu-se na década de 1960, a virtualização nos dias de hoje é extremamente importante para o mundo cada vez mais digital em que estamos presentes, devido às suas inúmeras vantagens, tais como:

Melhor aproveitamento da infra-estrutura existente, ao executar vários serviços em um servidor ou conjunto de máquinas, por exemplo, aproveitando a capacidade de processamento destes equipamentos o mais próximo possível da sua totalidade. O número de equipamento é menor, com o melhor aproveitamento dos recursos já existentes, a necessidade de aquisição de novos equipamentos diminui, assim como os consequentes gastos com a instalação, espaço físico, refrigeração, manutenção, consumo de energia, entre outros.

Gestão centralizada, dependendo da solução de virtualização utilizada fica mais fácil monitorizar os serviços em execução, já que a sua gestão é feita de maneira centralizada.

Implementação mais rápida, dependendo da aplicação, a virtualização pode permitir a sua implementação de forma mais rápida, uma vez que a infra-estrutura já está instalada.

Diversidade de plataformas, devido a uma grande diversidade de plataformas é possível assim realizar testes de desempenho de determinada aplicação em cada uma delas. Segurança e confiabilidade, como cada máquina virtual funciona de maneira independente das outras, um problema que possa surgir em uma delas, como uma vulnerabilidade de segurança, não afectará as restantes.

Migração e ampliação mais fácil, mudar o serviço de ambiente de virtualização é uma tarefa que pode ser feita rapidamente, assim como a ampliação da infra-estrutura. (Infowester, 2013)

Tipos de Virtualização

Virtualização de Servidores

Conhecidos como *hypervisor*, são os mais usados entre os três, como exemplo atualmente existe o *Hyper-V* (*Microsoft*) e o ESX (*Vsphere*) da *Vmware.*

O *hypervisor* é um sistema operativo funcional, ou seja, substitui integralmente o padrão de instalação dos sistemas operativos convencionais, como o Windows Server e Linux. (tiespecialistas, 2013)

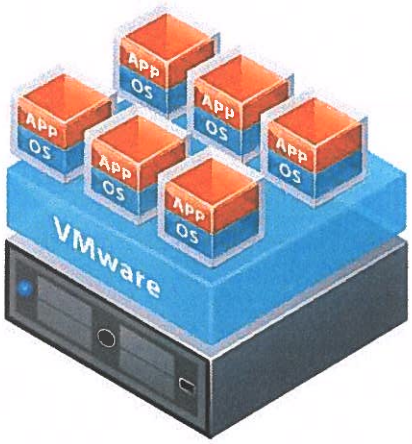

Figura 1 – Virtualização de servidores
Fonte: hoopdigital, 2013

Observando a figura da solução VMware representada anteriormente, há uma camada de virtualização a substituir o sistema operativo convencional que proporciona através da sua tecnologia criar múltiplas máquinas virtuais totalmente independentes nos seus recursos. Permitindo inúmeras vantagens tais como, redução de espaço físico, redução do consumo de energia e administração centralizada. (tiespecialistas, 2013)

Virtualização de Aplicações

A virtualização de aplicações é a possibilidade de aceder e utilizar as aplicações de forma remota, sem a necessidade da instalação das mesmas na máquina do utilizador.

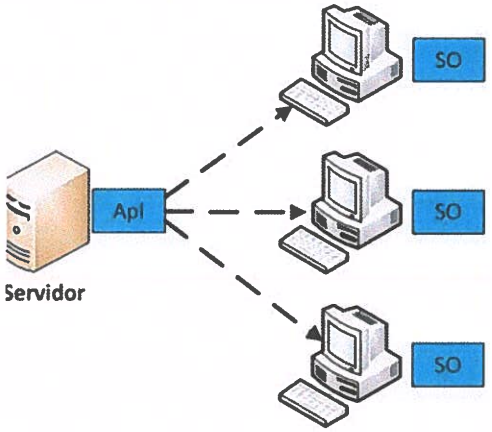

Figura 2 – Virtualização de Aplicações
Fonte: hoopdigital, 2013

Primordialmente este acesso era feito apenas por *LAN* ou através de uma *VPN*, sendo obrigado a configuração de um cliente, mas com o surgimento do mundo *Web*, hoje é possível o acesso via *internet*, de forma simples, fácil e segura. Todo o processamento é realizado no servidor onde foram instaladas as aplicações, tornando assim uma acesso de forma rápida e segura. (tiespecialistas, 2013)

Virtualização de Desktop

Baseada no conceito de virtualização de aplicações, a virtualização de desktop é mais centralizada em um hypervisor, ou seja, um ambiente *desktop* como por exemplo o Windows Professional não fica instalado localmente mas sim no hypervisor e depois é distribuido pelos vários utilizadores. (tiespecialistas, 2013)

Microsoft Windows Server 2012 R2

O Windows Server 2012 é uma plataforma comprovada de classe empresarial, que contribui para a construção de ambientes *Cloud*, simplificando a implementação de trabalho em *datacenters* com suporte para automação de diversas tarefas. (tiespecialistas, 2013)
Com uma interface de utilizador inovadora, com ferramentas poderosas de administração, suporte do *Windows PowerShell* melhorado e centenas de novos recursos nas áreas de redes, armazenamento e virtualização, o Windows Server 2012 pode ajudar as empresas a oferecer mais reduzindo os custos.

Evoluções e inovações

O sistema operativo *Windows Server* 2012 R2, fez um conjunto de evoluções e inovações técnicas em relação aos seus predecessores, tais como:

Active Directory Recycle Bin que permite recuperar os objetos que são apagados no *Active* Directory. *Active Directory Administrative Center (ADAC)* e *PowerShell*, fornece a possibilidade de criar e gerir contas de utilizador, grupos e unidades organizacionais. *Cloning Domain Controllers,* permite clonar um controlador de domínio que seja virtualizado através de uma plataforma como o *Hyper-V*. *Fine-Grained Password Policy*, as politicas de password foram melhoradas na medida que existe a opção de criar e administrar as próprias passwords utilizando o *Active Directory Administrative Center*. *IP Address Management*, permite localizar os servidores de endereçamento na rede de forma a administrá-los através de uma única interface de utilizador. *NIC Teaming*, permite que várias interfaces de rede sejam agrupadas numa única interface. *EAP-TTLS*, novo protocol de autenticação, é utilizado em conjunto com a autenticação do protocolo 802.1x. DNS, é possivel a instalação e remoção do DNS via *PowerShell*. (Microsoft, 2017)

Active Directory

O Active Directory (AD) é um Sistema operativo em *network* da Microsoft, que foi originalmente construído em conjunto com o Windows 2000 e está incluído na maior parte dos sistemas operativos Windows Server. É uma base de dados cuja a informação é armazenada em um ficheiro com o nome de NTDS.dit, contém informação sobre todos os objetos, tais como, utilizadores, computadores, impressoras, ficheiros e pastas partilhadas, em uma network organizada, sendo a mesma definida no *Active Directory Schema*. É um software que organiza e guarda informação sendo esta usada para autenticar e autorizar os utilizadores e computadores presentes na *network*. É composto por objetos, ou seja, todos os recursos da *network* são representados como um objeto no *Active Directory*, esses objetos possuem propriedades que são chamados de atributos dos objectos. (Microsoft, 2017)

Estrutura do Active Directory

No Active Directory a informação está organizada em uma estrutura lógica, isto permite encontrar um objeto pelo nome em vez da sua localização física, o que permite ter uma estrutura física da rede de maneira a ser mais "*user-friendly*". É constituído por domínios, árvores, florestas e unidades organizacionais (OU).

Domínio

O domínio é uma forma de network onde todas as contas de utilizadores, computadores, impressoras e outros objetos estão registados em uma base de dados central, em um dos Controladores de Domínio. A autenticação ocorre nos controladores de domínio e cada utilizador com uma única conta pode fazer login em qualquer máquina (desde que esta esteja no domínio) e usufruir dos recursos da network para os quais o administrador lhe der permissões. Sendo composto por uma estrutura hierárquica de recipientes e objetos, um nome de domínio DNS como identificador exclusivo e um serviço de segurança que autentica e autoriza qualquer acesso a recursos através de contas no domínio ou em outros domínios.

Árvores

Uma árvore é um grupo ou um arranjo hierárquico de um ou mais domínios de Windows 2000 que partilham um *namespace* semelhante. Têm as seguintes características, o nome do domínio de um domínio filho é o nome relativo desse domínio filho anexado com o nome do domínio pai. Todos os domínios dentro de uma única árvore, partilham um esquema em comum, o que é uma definição formal de todos os tipos de objetos que se pode armazenar em uma implementação do *Active Directory.* Todos os domínios dentro de uma única árvore, partilham um *Global Catalog* em comum, o que é o repositório central de informação sobre os objetos presentes na árvore.

Florestas

Uma floresta é um grupo ou um arranjo hierárquico de uma ou mais árvores que formam um namespace em separado. Todas as árvores presentes em uma floresta partilham um esquema em comum. Árvores presentes em uma floresta têm diferentes nomes de estruturas, de acordo com os seus domínios.Todos os domínios em uma floresta partilham um Global Catalog em comum.
Domínios presentes em uma floresta exercem de forma independente, mas a floresta permite comunicações entre toda a organização.

Unidades Organizacionais (OU)

Uma Unidade Organizacional tem como função organizar objetos dentro de um domínio contido em grupos administrativos lógicos. Uma OU pode conter objetos como contas de utilizadores, grupos, computadores, impressoras, aplicações, partilha de ficheiros e até outras unidades organizacionais. A hierarquia de uma unidade organizacional em um domínio é independente da estrutura hierárquica de uma OU em outros domínios.

Controladores de Domínio

Os controladores de domínio são a componente que hospeda todas as funcionalidades e protocolos do *Active Directory*. As funções dos controladores de domínio incluem o seguinte:

Cada controlador de domínio guarda uma cópia de toda a informação dos controladores de domínio existentes no mesmo domínio, gere alterações feitas a essa informação e trata de replicar as mudanças feitas para os outros controladores de domínio. Ter mais do que um controlador de domínio permite ter tolerância a erros. Se um controlador de domínio ficar *offline*, outro controlador de domínio pode oferecer todas as funções necessárias, tais como guardar alterações no *Active Directory*. Os controladores de domínio gerenciam todos os aspetos de interação de um utilizador com o domínio, tais como localizar objetos do AD e validar as tentativas de *login* do utilizador.

ADO.NET

ADO.NET é um conjunto de classes do .NET *Framework*, desenvolvidas para facilitar o acesso das aplicações à base de dados de diversos tipos como *Access*, SQL Server, Oracle etc. Oferece tecnologia para ligar à base de dados, executar comandos e obter resultados, estes resultados são processados diretamente e colocados em objetos *DataSet* que possui um conjunto de tabelas e as relações entre elas.

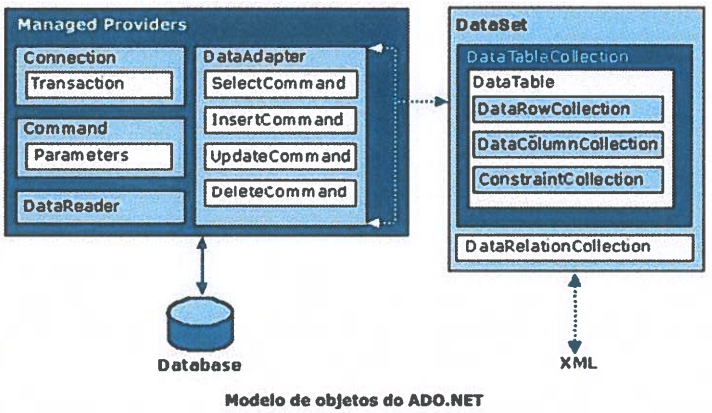

Figura 3 – Modelo de objetos
Fonte: www.macoratti.net/ado_net1.htm

O grupo *Managed Providers* é constituído pelos seguintes objetos de manipulação e leitura de dados:

Connection, que estabelece a ligação com a fonte de dados. *Command*, é utilizado para o envio de comandos SQL para a base de dados. *DataReade*r é um objeto que lê um conjunto de dados de forma sequencial e em *ReadOnly. DataAdapter* é um container para objetos do tipo *Command*, através desses comandos um *DataAdapter* interage com a base de dados, fazendo pesquisas, inserções, alterações e exclusões. *DataSet* é um objeto que armazena os dados, funciona com uma base de dados em memória, possuindo coelções de tabelas (*DataTables*), campos (*DataColumns*), registos (*DataRows*), *constraints* (*Constraints*) e relacionamentos (*DataRelations*). Os dados armazenados são depois manipulados em XML. Por fim, o *DataView* é muito semelhante ao objeto *View* contido em várias bases de dados. Através deste objeto podemos filtrar e ordenar registos de tabelas. (Microsoft, 2017)

Microsoft Azure

Cloud computing tem sido um dos assuntos mais discutidos atualmente no mundo da informática, seja pela oportunidade de redução de custos em cenários empresariais ou pelo poder de processamento e alta disponibilidade via internet que oferece para os seus utilizadores. Neste cenário, a Microsoft lançou uma plataforma denominada de *Microsoft Azure*, a partir de datacenters da Microsoft espalhados pelo mundo. Uma das principais capacidades que destaca uma plataforma como o *Microsoft Azure* é o seu poder de processamento, através de hospedagem e execução de processos com alta escalabilidade e provisionamento dinâmico.

O *Microsoft Azure* oferece três grupos principais de funcionalidades para a execução de processos, aplicações e serviços em *cloud*:

Computing, através de um conjunto de máquina virtuais disponíveis para uso, o Microsoft Azure oferece um poder de processamento ilimitado para aplicações que exigem alto volume de processamento. Armazenamento, aplicações executadas na *cloud* exigem tipos específicos de armazenamento preparado para uma alta escalabilidade e resposta. Da mesma forma, aplicações na cloud podem exigir um espaço ilimitado e crescente para o armazenamento de objetos diversos, para essa funcionalidade a plataforma, o *Azure* disponibiliza o *Azure Storage*, como parte das capacidades de armazenamento. Gestão, para aplicações e processos executados na plataforma *Microsoft Azure*, existem ferramentas de gestão e administração, assim como recursos para concepção de máquinas virtuais, configuração de serviços e *deployment* de aplicações, disponíveis para o utilizador. (Microsoft, 2017)

Roles do Microsoft Azure

Para o desenvolvimento de aplicações no *Microsoft Azure* é necessário conhecer o conceito de *Azure Services*. Azure Services consiste em uma fronteira de isolamento entre processos, através de elementos chamados roles. Cada *role* pode dispor de diversos *endpoints* para comunicação com elementos internos e externos, através de protocolos como HTPP, HTTPS e TCP.

Assim, temos na plataforma *Windows Azure* três tipos de *roles*:

Web Role, uma *role* que é alojada no IIS presente numa instância do *Microsoft Azure Service*. *Worker Role*, uma *role* que é executável como um serviço ou processo, por exemplo, um serviço de encoder, um consumidor de mensagens de filas, etc. *VM Role*, uma role que é representada por uma máquina virtual customizada, uma *Virtual Machine* (vhd). Nesse cenário, o cliente é responsável pela criação e configuração do sistema operacional da VM que será hospedada no *Windows Azure.*

Contextualização

No desenvolvimento deste projeto foram utilizados várias ferramentas e tecnologias como por exemplo a plataforma VMware onde foi feita a virtualização do mesmo, uso do Microsoft Visual Studio que permitiu desenvolver uma aplicação que comunica com o servidor SQL e um ambiente constituído por um controlador de domínio, um servidor SQL e um cliente.

Estas ferramentas e tecnologias estão ligadas ao mundo actual, sendo a plataforma VMware usada em ambientes de desenvolvimento onde é necessário testar uma aplicação em várias plataformas e a virtualização pode ser utilizada para produção com total segurança.

O uso de um controlador de domínio e um servidor SQL são ferramentas do sistema operativo Windows Server 2012 R2 predominantes em um ambiente empresarial nos dias actuais, pois a partir da funcionalidade Active Directory uma infraestrutura torna-se escalável, gerenciável e segura, sendo possivel aos utilizadores ter uma única senha que fornece todos os recursos necessários e disponíveis na rede de uma empresa e proporcionar um maior controlo e gestão da infraestrutura por parte do administrador.

Desenvolvimento

Para o desenvolvimento do projeto foi realizado, em primeiro lugar, a montagem de um laboratório constituido por quatro máquinas virtuais, três com o sistema operativo Windows Server 2012 R2 e uma com o sistema operativo Windows 8.1, instaladas e configuradas na plataforma VMware Workstation, as seguintes tabelas resumem as informações de cada máquina virtual.

Tabela 1 – Descrição das máquinas virtuais
Fonte: do autor

VM	Função	Domínio	Tipo Rede	Memória	Processador
Domain	Controlador de Domínio	A003domain.local	Host-Only	2 GB	1 vCPU/2 cores
Router	Servidor de Routing	A003domain.local	NAT/Host-Only	2 GB	1 vCPU/1 core
SQL	Servidor Sql	A003domain.local	Host-Only	2 GB	1 vCPU/1 core
Cliente	Cliente Windows 8.1	A003domain.local	Host-Only	2 GB	1 vCPU/2 cores

Tabela 2 – Descrição da rede
Fonte: do autor

VM	Tipo Rede	Segmento de Rede	Ip	Subnet Mask	Gateway	DNS
Domain	Host-Only	192.168.1.0	192.168.1.100	255.255.255.0	192.168.1.103	192.168.1.100
Router	NAT Host-Only	192.168.1.0	Auto 192.168.1.103	Auto 255.255.255.0	Auto	Auto 192.168.1.100
SQL	Host-Only	192.168.1.0	192.168.1.101	255.255.255.0	192.168.1.103	192.168.1.100
Cliente	Host-Only	192.168.1.0	192.168.1.102	255.255.255.0	192.168.1.103	192.168.1.100

Para fazer a instalação de uma máquina virtual na plataforma VMware, o primeiro passo é clicar em ***File***, depois clicar em ***New Virtual Machine*** e escolher o tipo de instalação ***Typical***, de seguida é seleccionado o disco .iso com sistema operativo a instalar e na janela seguinte selecciona-se o nome do sistema operativo. Para fazer a configuração das especificações de cada máquina virtual é feito um clique com o lado direito do rato em cima do nome da máquina virtual pretendida, escolhe-se a opção ***Settings*** e na janela ***Hardware*** apresentada é feita a configuração.

Após feitas as configurações necessárias, iniciamos a máquina virtual, para isso com a máquina seleccionada é feito um clique em ***Power on this virtual machine***, procedendo desta forma à instalação do sistema operativo.

Concluída a instalação de cada sistema operativo na máquina virtual correspondente, foi feito o início de sessão com o nome de utilizador "Administrator" e com a password "P@ssw0rd003" (ambos defenidos na instalação) em cada máquina virtual procendo à atribuição do respectivo nome, para isso na janela ***Server Manager*** no campo ***Computer Name*** foi inserido o nome da respectiva máquina virtual, efectuando a operação *reboot* necessária.
De seguida, na máquina virtual ***Domain*** foi feita a configuração da placa de rede Host-Only com a informação demonstrada na tabela acima. Para isso no painel ***Network and Sharing Center***, é feito um clique em ***Change Adapter Settings***, clique com o lado direito do rato na placa de rede ***Host-only,*** seleccionar a opção ***Properties*** e nas propriedades do ***Internet Protocol Version 4*** inserir a informação pretendida.

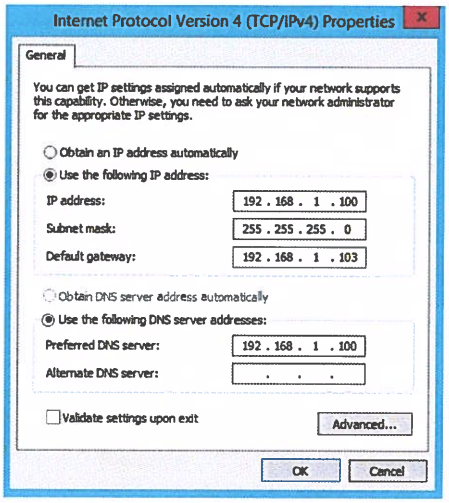

Figura 4 – Ipv4 Domain
Fonte: do autor

Por último, de modo a converter a VM em um controlador de domínio foi instalada/configurada a *Role Active Directory Domain* Services criando o domínio A003domain.local. Para isso na janela ***Server Manager*** clicar em ***Manage*** e seleccionar ***Add Roles and Features,*** na janela que seguinte seleccionar a opção ***Active Directory Domain Services,*** na configuração foi adicionado o nome do domínio como A003domain.local.

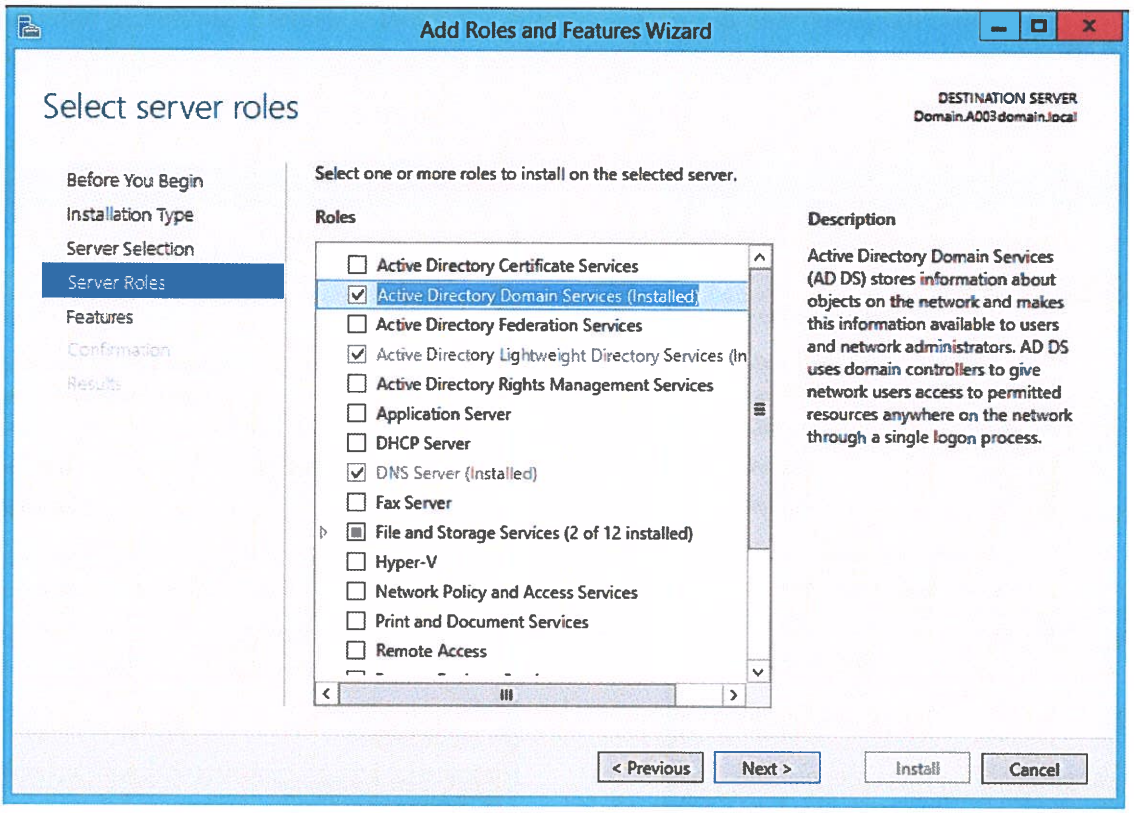

Figura 5 – Role Active Directory
Fonte: do autor

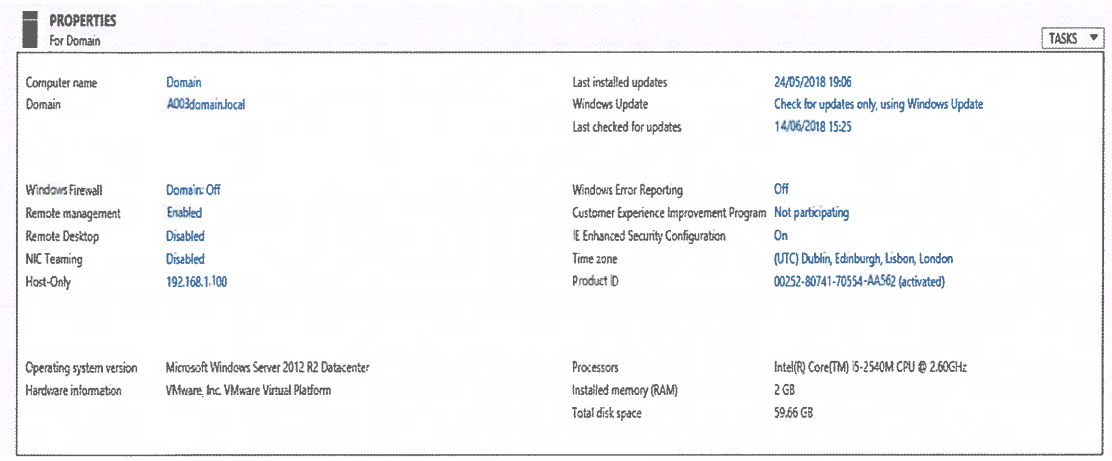

Figura 6 – Local Server Domain
Fonte: do autor

Terminado este processo a VM foi reiniciada e procedido a um novo *login*, desta vez com o domínio atribuído no nome de utilizador ou seja A003DOMAIN\Administrator.

De seguida nas máquinas virtuais restantes (Router, SQL, Cliente) foi realizada a configuração das placas de rede, adicionando no servidor de *routing* o endereço IP, *Subnet Mask*, DNS que neste caso é o endereço IP do controlador de domínio e nas restantes máquinas foi adicionado o endereço IP, *Subnet Mask*, DNS e *Default Gateway,* ou seja, o endereço IP do servidor de *routing*. Efetuando também a atribuição do ***Computer Name*** em cada VM.
As figuras que se seguem demonstram as configurações efetuadas.

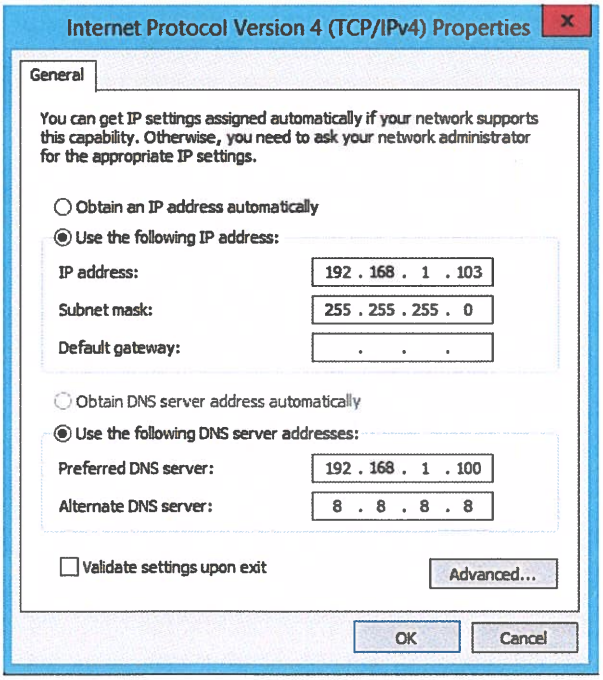

Figura 7 – Ipv4 Router Server
Fonte: do autor

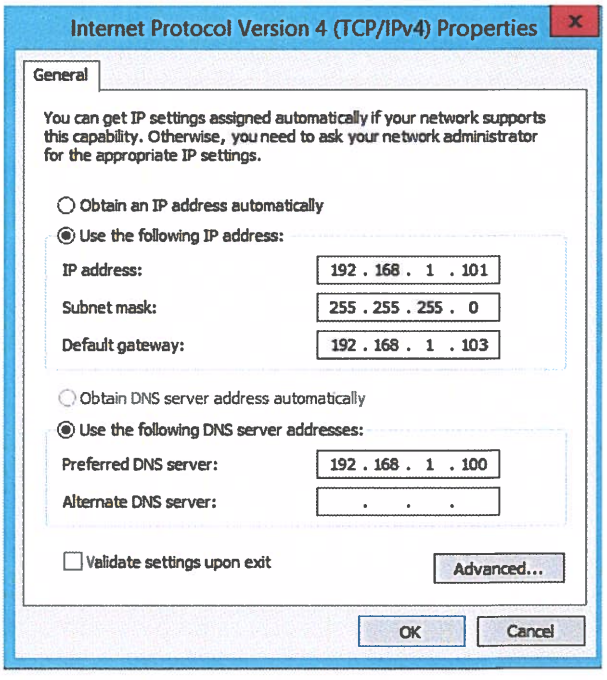

Figura 8 – Ipv4 SQL Server
Fonte: do autor

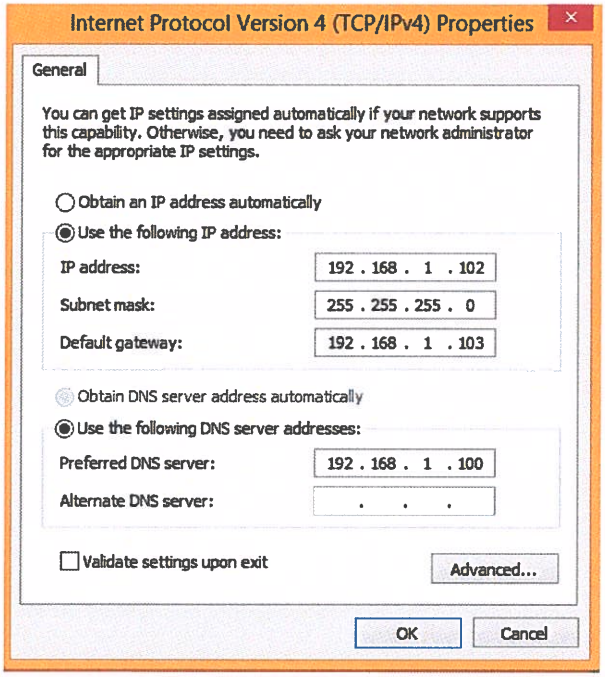

Figura 9 – Ipv4 Cliente
Fonte: do autor

Feita a configuração das placas de rede de cada máquina, as mesmas foram adicionadas ao domínio A003domain.local de forma a ficarem todas interligadas em um domínio. Para isso na janela ***Server Manager*** na opção ***Domain*** foi feito um clique em ***Change*** e no campo ***Domain*** introduzido o respectivo nome do domínio pretendido.

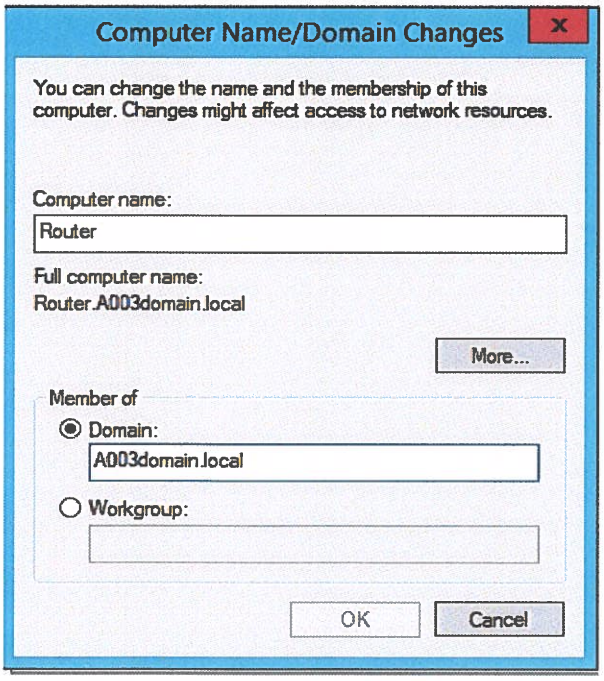

Figura 10 – Dominio Router
Fonte: do autor

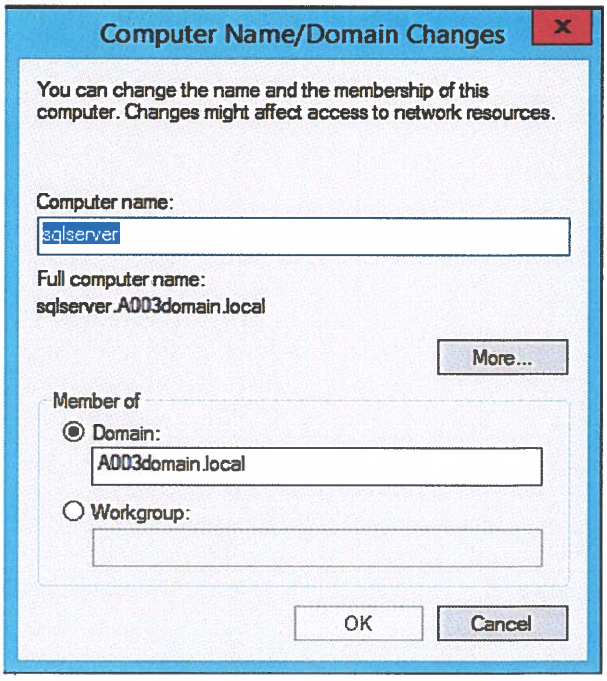

Figura 11 – Dominio SQL
Fonte: do autor

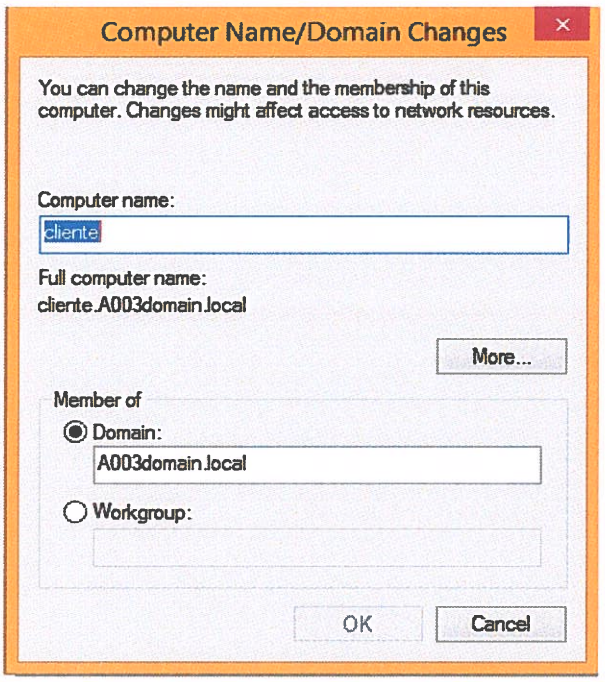

Figura 12 – Dominio cliente
Fonte: do autor

Após este procedimento, os servidores e cliente foram reniciados e em cada VM feito o login com o nome do utilizador A003DOMAIN\Administrator.

Feito o incio de sessão na máquina virtual *Router*, na janela ***Server Manager*** em ***Add Roles and Features***, foi realizada a instalação da *role* ***Remote Acess*** e a *feature* ***de Routing***, de forma a promover a mesma a um servidor de *routing*.

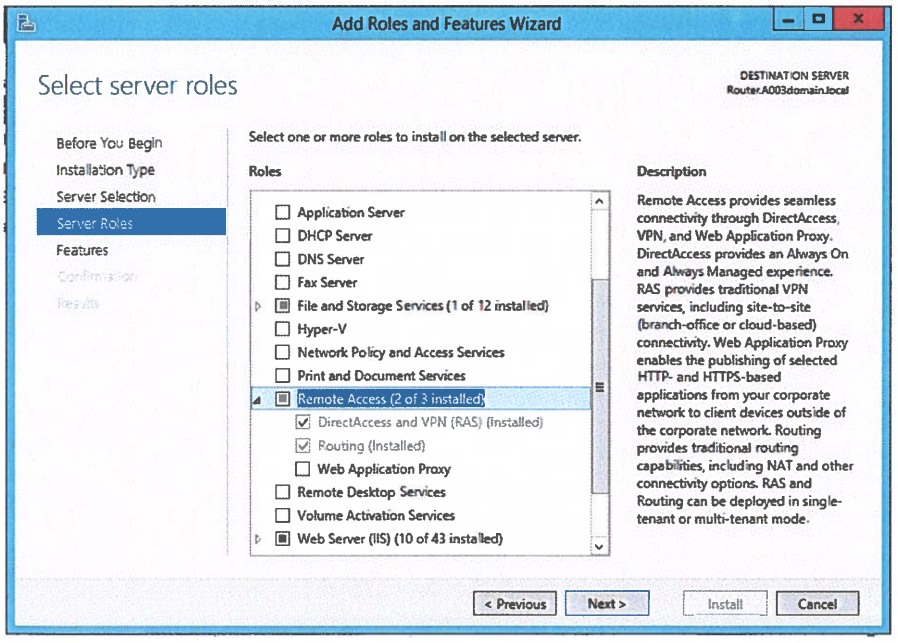

Figura 13 – Role Remote Acess
Fonte: do autor

Finalizada a instalação do ***Remote Acess*** e depois de ter sido verificado o acesso à *Internet* em cada máquina com sucesso, foi iniciada a sessão na máquina virtual SQL, efetuando a Instalação do *Sql Server* de modo a promover a mesma a um servidor SQL que tem como função guardar as informações provenientes da aplicação desenvolvida.

Para realizar a instalação do Sql Server foi feito um duplo clique no ficheiro executável de instalação, clique na opção ***New SQL Server stand-alone installation***, durante o processo de instalação foi adicionado o nome do servidor SQL que neste caso é SQLSERVER , dadas permissões ao utilizador que é administrador e adicionado como forma de login na base de dados a opção ***Windows Authentication.***

Figura 14 – Instalar SQL Server
Fonte: do autor

Concluída a instalação, realizou-se a abertura do *Microsoft SQL Management Studio*, introduzindo o *Server Name, Username e Password* com *Windows Authentication*. Esta ferramenta permite facilitar a gestão de uma base de dados SQL, local ou remotamente.

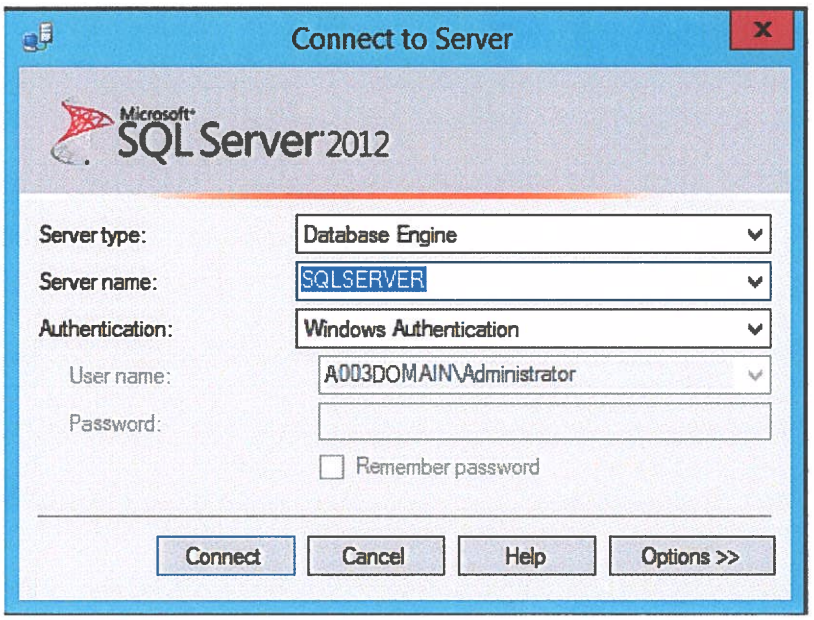

Figura 15 – Connect to Server
Fonte:do autor

Foi criada uma base de dados com o nome ginasio e três tabelas com o nome de Clientes, Logins e Modalidades. A tabela Clientes é constituída pelos campos iduser, nome, modalidade, foto, fotopath, tem como objectivo guardar e gerir toda a informação referente aos clientes. A tabela Logins pelos campos iduser, username, password, tem como objectivo guardar e gerir informação referente aos logins efetuados e a tabela Modalidades constituída pelos campos idmodalidade, nome, horas, que guarda a informação referente às modalidades presentes no ginásio.

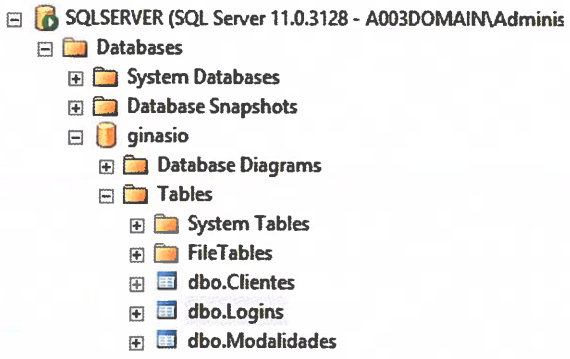

Figura 16 – SQLSERVER
Fonte: do autor

De seguida, na máquina virtual Cliente, foi instalado o programa Microsoft Visual Studio 2017 para proceder ao desenvolvimento da aplicação. Foi escolhida uma aplicação WPF com o *framework* .NET 4.6.1, foi adicionada uma *data entity* com o nome Model1 cuja a sua função é construir uma ligação entre a base de dados e a aplicação permitindo usar, editar e remover informação trocada entre ambas.

Para adicionar uma *data entity*, é feito um clique na opção ***Add New Item*** escolhe-se a opção ***ADO.NET Entity Data Model*** de seguida escolhemos o servidor SQL e a base de dados criada de modo a fazer uma nova *connection*, na janela que é apresentada escolhemos as tabelas necessárias adicionar ao projeto e de seguida é criada uma ligação entre a base de dados e a aplicação.

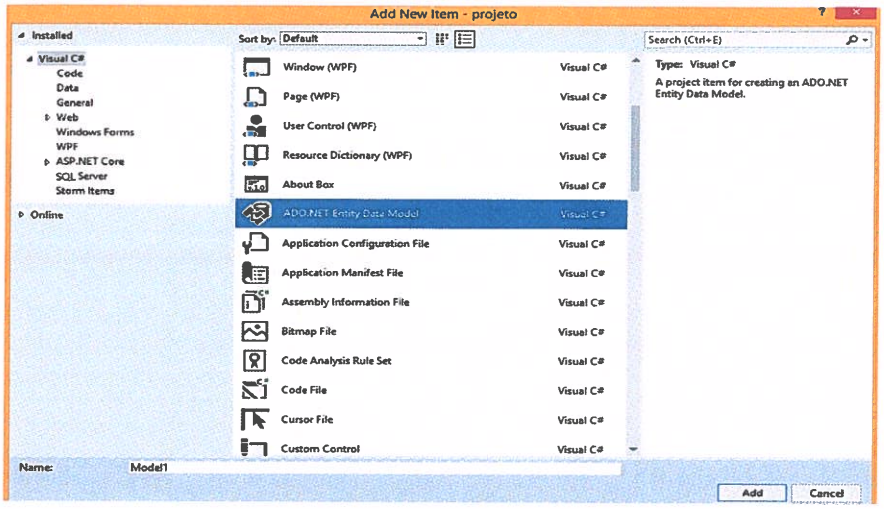

Figura 17 – Add New Item
Fonte: do autor

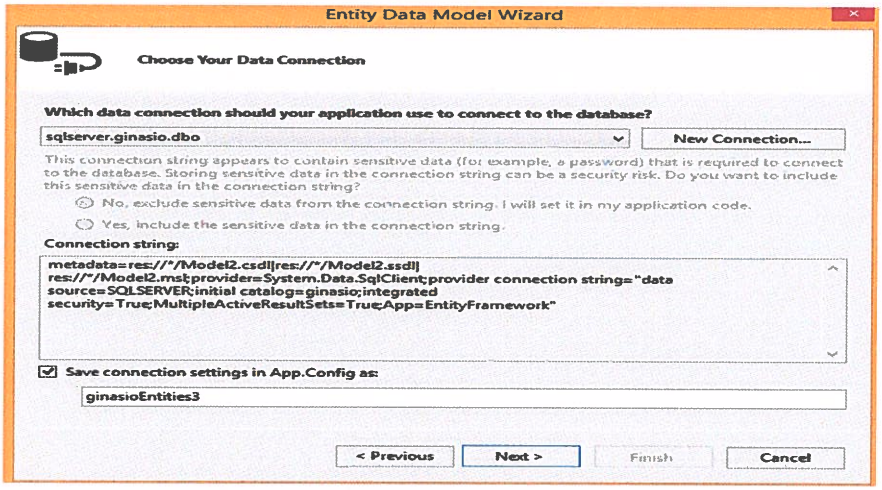

Figura 18 – Entity Data Model Wizard
Fonte: do autor

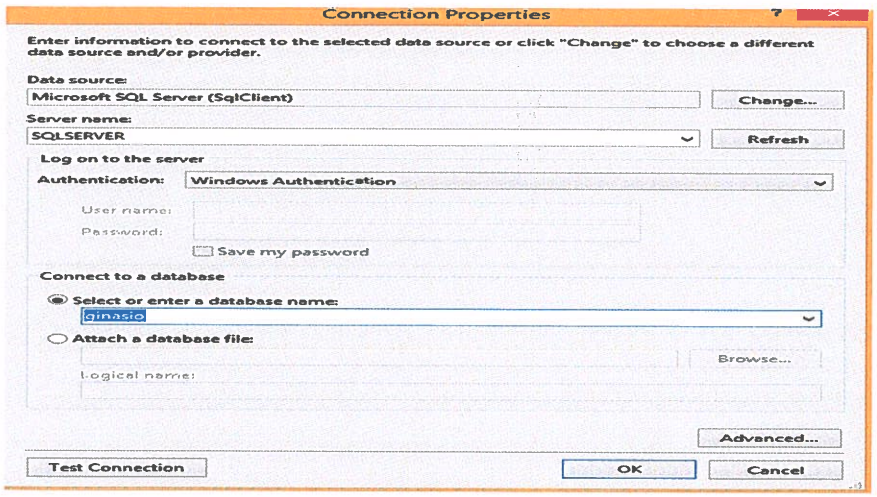

Figura 19 – Connection Properties
Fonte: do autor

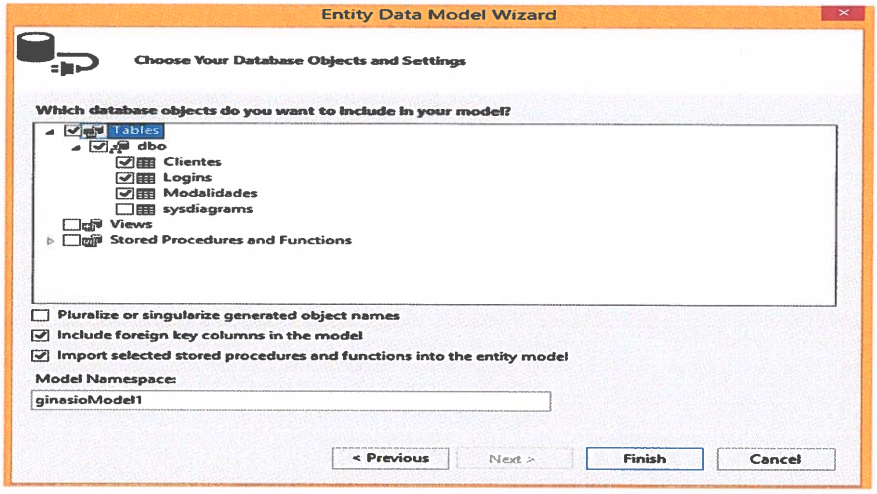

Figura 20 – Add New Item
Fonte: do autor

Desenvolvimento – Aplicação

No seguimento do tema deste projeto, foi desenvolvida uma aplicação constituida por quatro janelas (Registo, Login, User e Admin).

Na janela Registo o cliente pode fazer um registo ou caso tenha feito o mesmo pode efetuar um *login* carregando no botão Login que direciona para a respetiva janela.

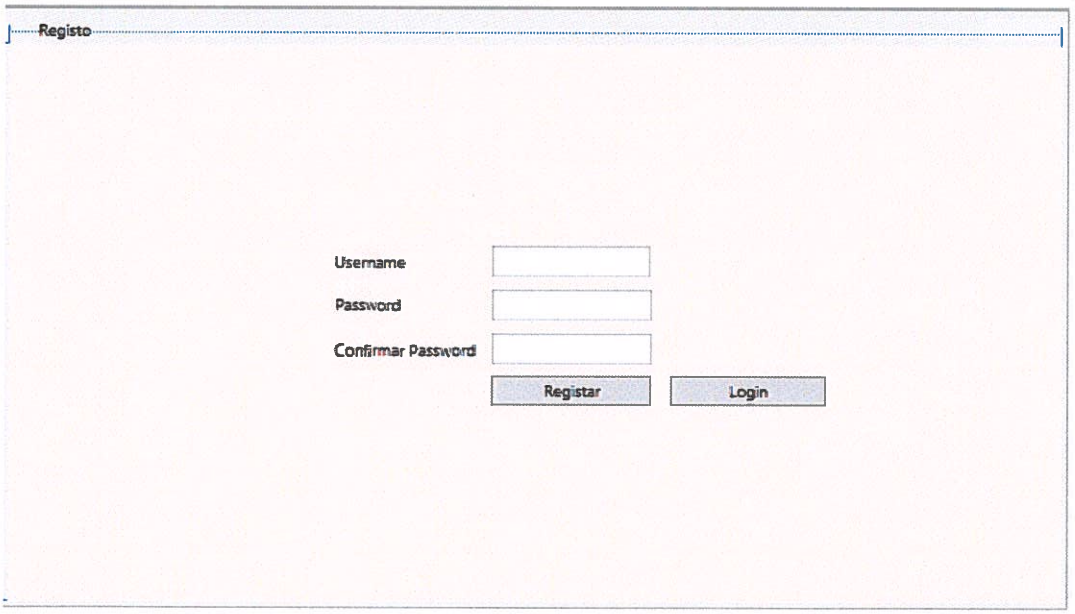

Figura 21 – Registo
Fonte:do autor

```
private bool registar(string user, string password)
{
var query = from Login in db.Logins
where Login.username == user
&& Login.password == password
select Login;
j = user;

Login u = new Login();
u.username = txtUserReg.Text;
u.password = txtPassReg.Password;

db.Logins.Add(u);
db.SaveChanges();
if (query.Any())
{
return true;
}
else
{
return false;
}

}

bool Validate()
{
if (txtUserReg.Text == "")
{
MessageBox.Show("Preencha o utilizador");
return false;
}
if (txtPassReg.Password == "")
{
MessageBox.Show("Preencha a password");
return false;
}



if (txtPassRegCon.Password != txtPassReg.Password)
{
MessageBox.Show("As Passwords não são iguais");
return false;
}
else
{
return true;
}
}
```

No bloco de código reprensentado anteriormente são utilizadas duas funções booleanas, a função Registar tem como objectivo verificar se as informações introduzidas nos campos respectivos já existem na base de dados e a função *Validate* com o objectivo de verificar se os campos *username* e *password* são devidamente preenchidos não sendo permitido campos vazios ou as passwords não serem iguais entre si.

```
private void btnReg_Click(object sender, RoutedEventArgs e)
{
if (Validate() == true)
{
if (registar(txtUserReg.Text, txtPassReg.Password) == true)
{

MessageBox.Show("Registo efetuado com sucesso," + j);
Login p = new Login();
p.Show();
this.Close();
}

else
{

MessageBox.Show("Registo nao efectuado, tente de novo");
}

}

else
{

MessageBox.Show("Login nao efectuado");
}
}

private void btnlog_Click(object sender, RoutedEventArgs e)
{
Login p = new Login();
p.Show();
this.Close();
}
}
```

Este bloco de código é responsável por receber os valores ***true*** ou ***false*** das funções *Validate* e Registar, caso o valor recebido seja ***true*** o cliente recebe uma mensagem a validar o seu registo e é redireccionado para a janela do *Login*, se o valor recebido for ***false*** é recebida uma mensagem a informar a necessidade de fazer um registo da forma correta.

Se o utilizador efectuou um registo de forma correta é rederecionado para a janela de login. A janela login tem como objectivo autenticar o respectivo utilizador e dependendo se o mesmo for classificado como cliente ou administrador é rederecionado para a sua interface correspondente.

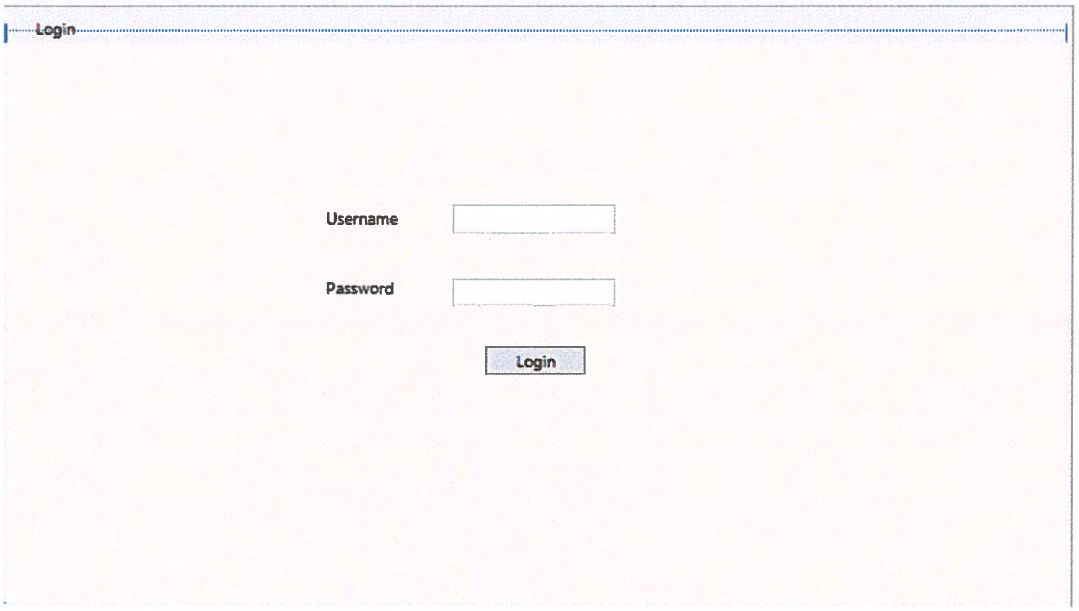

Figura 22 – Login
Fonte: do autor

```
private bool login(string user, string password)
{
var query = from Login in db.Logins
where Login.username == user
&& Login.password == password
select Login;
u = user;

if (query.Any())
{
return true;
}
else
{
return false;
}
}
bool Validate()
{
if (txtUser.Text == "")
{
MessageBox.Show("Preencha o utilizador");
return false;
}
if (txtPass.Password == "")
{
MessageBox.Show("Preencha a password");
return false;
}
else
{
return true;
```

```
}


}
private void btnLogin_Click(object sender, RoutedEventArgs e) //evento do botão login
{
if (Validate() == true)
{
if (login(txtUser.Text, txtPass.Password) == true)
{
if (txtUser.Text == "Administrator")
{
MessageBox.Show("Bem-vindo " + u);
MainWindow p = new MainWindow();
p.Show();
this.Close();
}
else
{
MessageBox.Show("Bem-vindo " + u);
UserPage p = new UserPage();
p.Show();
this.Close();

}
}
else
{

MessageBox.Show("Login nao efectuado" +
"Verifique os campos ou faça um registo");
}
}
```

No código representado anteriormente existem duas funções boleanas, a função login que confirma se os campos *username* e *password* estão presentes na tabela *logins* da base de dados e a função validate que verifica se não existem campos vazios.

Os valores ***true*** ou ***false*** são depois enviados para o evento clique do botão Login que por sua vez, caso seja recebido o valor ***true***, o utilizador é rederecionado para a sua *interface* correspondente, caso o valor seja ***false*** o utilizador é recebido com uma mensagem que informa a necessidade de verificar se os campos introduzidos estão corretos ou para efectuar um registo.

Depois de um login efetuado com sucesso o utilizador, caso seja considerado um cliente, é rederecionado para a sua interface onde lhe é permitido escolher a modalidade desejada e marcar/cancelar uma aula inserindo a sua informação.

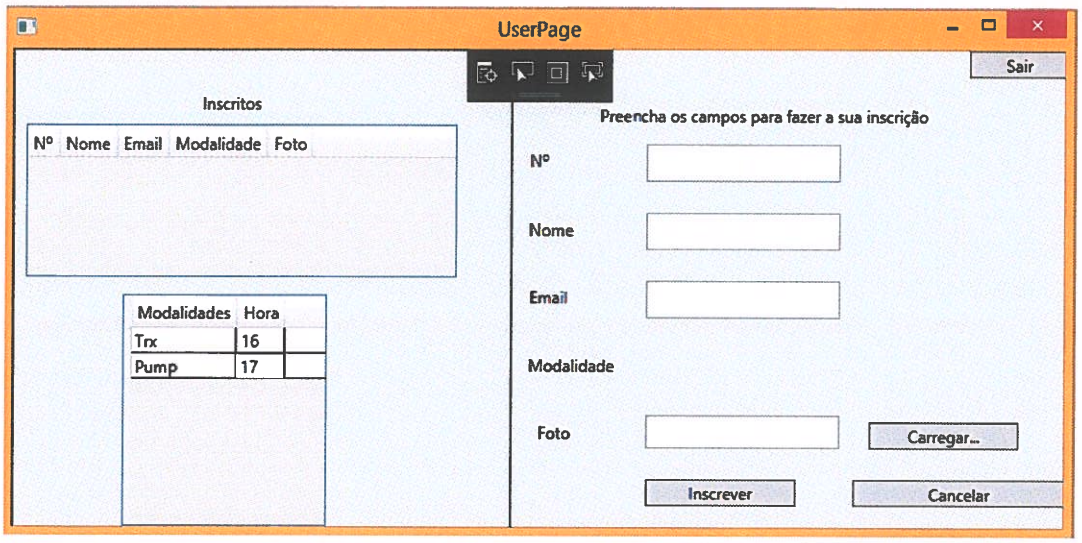

Figura 23 – UserPage
Fonte: do autor

```
private void btninscrever_Click(object sender, RoutedEventArgs e)
{
using (ginasioEntities2 db = new ginasioEntities2())
{



Cliente mycliente = new Cliente
{

iduser = int.Parse(txtuserid.Text),
nome = txtname.Text,
email = txtemail.Text,
modalidade = blockmodalidade.Text,
fotopath = txtfoto.Text,



};
db.Clientes.Add(mycliente);
db.SaveChanges();
gridcliente.ItemsSource = db.Clientes.ToList();
}
```

No bloco de código anterior esta representado o evento do botão Inscrever onde as informações do cliente (id, nome, email, modalidade) são introduzidas na base de dados por meio de uma data entity chamada **mycliente**.

Quando a inscrição é efetuada o administrador recebe as informações do cliente podendo editar ou remover as mesmas, o seguinte bloco de código desmontra como é feito esse procedimento.

```
private void btneditar_Click(object sender, RoutedEventArgs e)
{
```

```
using (ginasioEntities2 db = new ginasioEntities2())
{
int iduser;
if (int.TryParse(txtid.Text, out iduser))
{
Cliente mycliente = db.Clientes.Where(x => x.iduser == iduser).FirstOrDefault();
mycliente.nome = txtnome.Text;
mycliente.email = txtemail.Text;


db.SaveChanges();
gridginasio.ItemsSource = db.Clientes.ToList();
}
```

Utilizando o id do utilizador como referência o administrador pode editar a informação do cliente. Outra opção presente na interface do administrador é a possibilidade de adicionar/editar uma modalidade e as suas respectivas horas, o seguinte bloco de código demonstra a mesma.

```
private void btnaddmod_Click(object sender, RoutedEventArgs e)
{
using (ginasioEntities2 db = new ginasioEntities2())
{
Modalidade modalidade = new Modalidade
{
Nome = txtaddmod.Text,
Horas = txthoras.Text,
};
db.Modalidades.Add(modalidade);
db.SaveChanges();
gridmoda.ItemsSource = db.Modalidades.ToList();
}
```

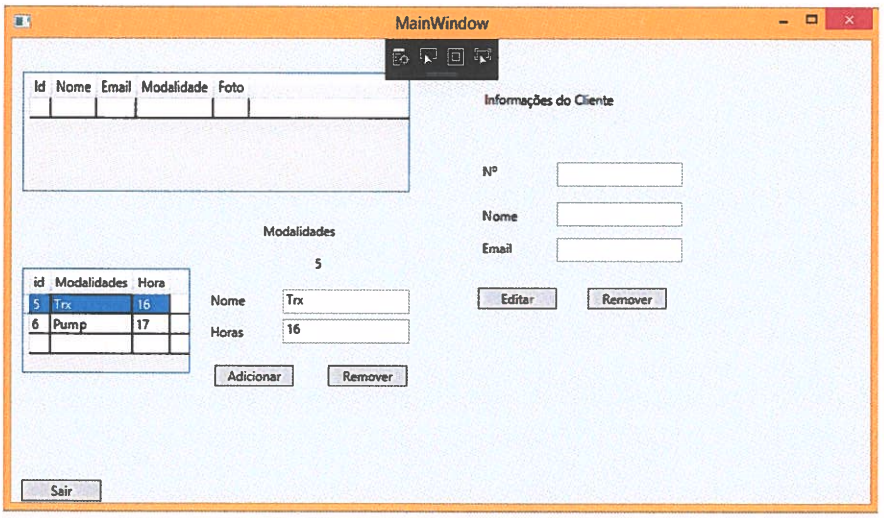

Figura 24 – Admin
Fonte: do autor

Conclusão

O objetivo deste projeto apontava para a criação de uma aplicação ASP.NET que fizesse a comunicação com uma base de dados SQL incorporada em um ambiente virtual, objectivo esse que foi atingido na totalidade.

Através do objectivo do projeto cheguei à conclusão que a tecnologia ASP.NET e a linguagem de programação C# permitem desenvolver aplicações com forte desempenho e versatilidade, foi um projeto interessante pois meteu em prática os conhecimentos adquiridos ao longo da licenciatura e do curso de Virtualização e Cloud Computing, tendo também uma vertente de pesquisa e auto-aprendizagem o que permitiu melhorar o meu conhecimento sobre as tecnologias utilizadas neste projeto.

Referências Bibliográficas

Microsoft, 2017. Consultado a 06/06/2018 em:
https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/

Rocha, V. (2013, 15 de Março). tiespecialistas.com.br. Consultado a 06/06/2018 em:
https://www.tiespecialistas.com.br/tipos-de-virtualizacao/

Tutorials Point (I) Pvt. Ltd. (2015), Entity Framework. Consultado a 15/06/2018 em:
https://www.tutorialspoint.com/entity_framework/index.htm

Tulloch, M., Windows Server Team (2012), *Introducing Windows Server 2012* (RTM Edition)

Techopedia. Consultado a 06/08/2018 em:
https://www.techopedia.com/definition/719/virtualization

Infowester "O que é a Virtualização?", 2013. Consultado a 15/06/2018 em:
https://www.infowester.com/virtualizacao.php

Anexos

Anexo I - Registo.xaml

```
<Window x:Class="projeto.Registo"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:local="clr-namespace:projeto"
      mc:Ignorable="d"
      Title="Registo" Height="450" Width="800" Background="FloralWhite">
   <Grid>
      <Grid HorizontalAlignment="Left" Height="132" Margin="237,146,0,0"
VerticalAlignment="Top" Width="388">
         <Grid.ColumnDefinitions>
            <ColumnDefinition Width="Auto"/>
            <ColumnDefinition Width="Auto"/>
            <ColumnDefinition Width="Auto"/>
         </Grid.ColumnDefinitions>
         <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="Auto"/>
         </Grid.RowDefinitions>
         <Label Content="Username" Grid.Column="0" Grid.Row="0" Margin="3"
HorizontalAlignment="Left" VerticalAlignment="Center"/>
         <Label Content="Password" Grid.Column="0" HorizontalAlignment="Left" Margin="3"
Grid.Row="1" VerticalAlignment="Center"/>
         <Label Content="Confirmar Password" Grid.Column="0" HorizontalAlignment="Left"
Margin="3" Grid.Row="2" VerticalAlignment="Center"/>
         <TextBox x:Name="txtUserReg" Grid.Column="1" Height="23" Margin="3"
Grid.Row="0" Width="120"/>
         <PasswordBox x:Name="txtPassReg" Grid.Column="1" Height="23" Margin="3"
Grid.Row="1" />
         <PasswordBox x:Name="txtPassRegCon" Grid.Column="1" Height="23" Margin="3"
Grid.Row="2" />

         <Button x:Name="btnReg" Content="Registar" Grid.Column="1" Height="23"
Grid.Row="3" Click="btnReg_Click" Margin="3"/>
         <Button x:Name="btnlog" Content="Login" Grid.Column="2" Margin="11,3,-127,3"
Grid.Row="3" Click="btnlog_Click" Height="23"/>
      </Grid>

   </Grid>
</Window>
```

Anexo II - Registo.xaml.cs

```
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;

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

      public static string j; //variável global para username


      ginasioEntities2 db = new ginasioEntities2(); //instancia a ligação à base de dados

            private bool registar(string user, string password) //confirma se o username e
password estão na tabela
      {



         var query = from Login in db.Logins
                  where Login.username == user
                  && Login.password == password
                  select Login;
         j = user;

         Login u = new Login();
         u.username = txtUserReg.Text;
         u.password = txtPassReg.Password;
```

```
        db.Logins.Add(u);
        db.SaveChanges();
        if (query.Any())
        {
            return true;
        }
        else
        {
            return false;
        }


    }

    bool Validate() //validar os campos username, password e confirmação de password
    {
        if (txtUserReg.Text == "")
        {
            MessageBox.Show("Preencha o utilizador");
            return false;
        }
        if (txtPassReg.Password == "")
        {
            MessageBox.Show("Preencha a password");
            return false;
        }
        if (txtPassRegCon.Password != txtPassReg.Password)
        {
            MessageBox.Show("As Passwords não são iguais");
            return false;


        }
        else
        {
            return true;
        }



    }

    private void btnReg_Click(object sender, RoutedEventArgs e) //evento do botão registar
    {
        if (Validate() == true)
        {
            if (registar(txtUserReg.Text, txtPassReg.Password) == true)
            {
```

```
                MessageBox.Show("Registo efetuado com sucesso," + j);
                Login p = new Login();
                p.Show();
                this.Close();




            }

            else
            {

                MessageBox.Show("Registo nao efectuado, tente de novo");
            }




        }

            else
            {

                MessageBox.Show("Login nao efectuado");
            }
        }

    private void btnlog_Click(object sender, RoutedEventArgs e) //evento do botão Login, ao clicar envia o cliente para a janela do login
    {
        Login p = new Login();
        p.Show();
        this.Close();
    }
}
}
```

Anexo III – Login.xaml

```
<Window x:Class="projeto.Login"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:local="clr-namespace:projeto"
      mc:Ignorable="d"
      Title="Login" Height="450" Width="800" Background="FloralWhite">
   <Grid>
      <Button x:Name="btnLogin" Content="Login" HorizontalAlignment="Left"
Margin="358,227,0,0" VerticalAlignment="Top" Width="75" Click="btnLogin_Click" />
      <TextBox x:Name="txtUser" HorizontalAlignment="Left" Height="23"
Margin="334,120,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"/>
      <PasswordBox x:Name="txtPass" HorizontalAlignment="Left" Margin="334,175,0,0"
VerticalAlignment="Top" Height="21" Width="120"/>
      <Label Content="Username" HorizontalAlignment="Left" Margin="234,117,0,0"
VerticalAlignment="Top" Width="80"/>
      <Label Content="Password" HorizontalAlignment="Left" Margin="234,170,0,0"
VerticalAlignment="Top" Width="80"/>

   </Grid>
</Window>
```

Anexo IV – Login.xaml.cs

```
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;

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




      }



      public static string u; //variável global para username



      ginasioEntities2 db = new ginasioEntities2(); //instancia a ligação à base de dados

      private bool login(string user, string password) //confirma se o username e password estão
na tabela
      {



         var query = from Login in db.Logins
                  where Login.username == user
```

```
                && Login.password == password
                select Login;
    u = user;

    if (query.Any())
    {
        return true;
    }
    else
    {
        return false;
    }
}


bool Validate() //valida os campos username e password
{
    if (txtUser.Text == "")
    {
        MessageBox.Show("Preencha o utilizador");
        return false;
    }
    if (txtPass.Password == "")
    {
        MessageBox.Show("Preencha a password");
        return false;
    }
    else
    {
        return true;
    }


}




private void btnLogin_Click(object sender, RoutedEventArgs e) //evento do botão login
{
    if (Validate() == true)
    {
        if (login(txtUser.Text, txtPass.Password) == true)
        {
            if (txtUser.Text == "Administrator")
            {
```

```
                    MessageBox.Show("Bem-vindo " + u);
                    MainWindow p = new MainWindow();
                    p.Show();
                    this.Close();



                }
                else
                {
                    MessageBox.Show("Bem-vindo " + u);
                    UserPage p = new UserPage();
                    p.Show();
                    this.Close();

                }



            }
            else
            {

                MessageBox.Show("Login nao efectuado" +
                                                "Verifique os campos ou faça um registo");
            }
        }
    }
}
```

Anexo V – UserPage.xaml

```
<Window x:Class="projeto.UserPage"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:local="clr-namespace:projeto"
      mc:Ignorable="d"
      Title="User" Height="450" Width="800" Background="AliceBlue">
   <Grid>
      <Grid.ColumnDefinitions>
         <ColumnDefinition/>
      </Grid.ColumnDefinitions>
      <Grid HorizontalAlignment="Left" Height="299" Margin="377,57,0,0"
VerticalAlignment="Top" Width="405">
         <Grid.ColumnDefinitions>

            <ColumnDefinition Width="Auto"/>
            <ColumnDefinition Width="Auto"/>
            <ColumnDefinition Width="Auto"/>
         </Grid.ColumnDefinitions>
         <Grid.RowDefinitions>
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />



         </Grid.RowDefinitions>
         <Label x:Name="lbluserid" Content="N°" Grid.Column="0" Grid.Row="0"
Margin="3,12,0,13" HorizontalAlignment="Left" VerticalAlignment="Center" Height="26"
Width="58"/>
         <Label x:Name="lblname" Content="Nome" Grid.Column="0" Grid.Row="1"
Margin="3,12,0,13" HorizontalAlignment="Left" VerticalAlignment="Center" Height="26"
Width="58"/>
         <Label x:Name="lblemail" Content="Email" Grid.Column="0" Grid.Row="2"
Margin="3,11,0,11" HorizontalAlignment="Left" VerticalAlignment="Center" Height="26"
Width="58"/>
         <Label x:Name="lblmodalidade" Content="Modalidade" Grid.Column="0" Grid.Row="3"
Margin="3,11,0,11" HorizontalAlignment="Left" VerticalAlignment="Center" Height="26"
Width="82"/>
         <TextBox x:Name="txtuserid" HorizontalAlignment="Left" Grid.Column="1"
Grid.Row="0" Margin="10,13,0,10" VerticalAlignment="Center" Width="145" Height="28"/>
         <TextBox x:Name="txtname" HorizontalAlignment="Left" Grid.Column="1"
Grid.Row="1" Margin="10,13,0,10" VerticalAlignment="Center" Width="145" Height="28"/>
```

```
        <TextBox x:Name="txtemail" HorizontalAlignment="Left" Grid.Column="1"
Grid.Row="2" Margin="10,13,0,10" VerticalAlignment="Center" Width="145" Height="28"/>
        <TextBlock x:Name="blockmodalidade" HorizontalAlignment="Left" Grid.Column="1"
Grid.Row="3" Margin="10,13,0,10" VerticalAlignment="Center" Width="145" Height="28"/>
        <Button x:Name="btninscrever" Content="Inscrever" Grid.Column="1" Grid.Row="5"
Margin="10,59,33,-31" Click="btninscrever_Click"/>
        <Button x:Name="btncancelar" Content="Cancelar" Grid.Column="2" Grid.Row="5"
Margin="10,59,-5,-31" Click="btncancelar_Click"/>
        <Image x:Name="fotoImage" Grid.Column="2" Width="118" Grid.RowSpan="3"
HorizontalAlignment="Left" Margin="23,2,-25,0" />
        <Label x:Name="lblfoto" Content="Foto" HorizontalAlignment="Left"
Margin="10,10,0,0" Grid.Row="5" VerticalAlignment="Top"/>
        <TextBox x:Name="txtfoto" Grid.Column="1" HorizontalAlignment="Left" Height="25"
Margin="10,11,0,0" Grid.Row="5" TextWrapping="Wrap" VerticalAlignment="Top"
Width="145"/>
        <Button x:Name="btnCarregar" Content="Carregar..." Grid.Column="2"
HorizontalAlignment="Left" Margin="23,16,-19,0" Grid.Row="5" VerticalAlignment="Top"
Width="111" Click="btnCarregar_Click"/>
        <TextBlock x:Name="blockhora" Grid.Column="2" HorizontalAlignment="Left"
Margin="20,14,0,0" Grid.Row="3" Width="145" Height="28" TextWrapping="Wrap"
VerticalAlignment="Top"/>
    </Grid>
    <Button x:Name="btnsair" Content="Sair" HorizontalAlignment="Left" Margin="713,0,0,0"
VerticalAlignment="Top" Width="75" Click="btnsair_Click"/>
    <DataGrid x:Name="gridmoda" IsReadOnly="True" AutoGenerateColumns="False"
HorizontalAlignment="Left" Height="173" Margin="82,183,0,0" VerticalAlignment="Top"
Width="152" SelectionChanged="gridmoda_SelectionChanged">
      <DataGrid.Columns>
        <DataGridTextColumn Header="Modalidades" Binding="{Binding Nome}"/>
        <DataGridTextColumn Header="Hora" Binding="{Binding Horas}"/>
      </DataGrid.Columns>
    </DataGrid>
    <Border BorderBrush="Black" BorderThickness="1" HorizontalAlignment="Left"
Height="419" VerticalAlignment="Top" Width="372">
      <DataGrid x:Name="gridcliente" IsReadOnly="True" AutoGenerateColumns="False"
HorizontalAlignment="Left" Height="114" VerticalAlignment="Top" Width="321"
SelectionChanged="gridcliente_SelectionChanged" Margin="9,55,0,0">
        <DataGrid.Columns>
          <DataGridTextColumn Header="Nº" Binding="{Binding iduser}"/>
          <DataGridTextColumn Header="Nome" Binding="{Binding nome}"/>
          <DataGridTextColumn Header="Email" Binding="{Binding email}"/>
          <DataGridTextColumn Header="Modalidade" Binding="{Binding modalidade}"/>
          <DataGridTextColumn Header="Foto" Binding="{Binding fotopath}"/>
        </DataGrid.Columns>
      </DataGrid>
    </Border>
    <Label Content="Inscritos" HorizontalAlignment="Left" Margin="136,27,0,0"
VerticalAlignment="Top"/>
    <Label Content="Preencha os campos para fazer a sua inscrição" HorizontalAlignment="Left"
Margin="433,35,0,0" VerticalAlignment="Top"/></Grid></Window>
```

Anexo VI – UserPage.xaml.cs

```
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 Microsoft.Win32;
using iTextSharp;
using iTextSharp.text;
using iTextSharp.text.pdf;
using iTextSharp.text.pdf.draw;
using System.Text.RegularExpressions;

namespace projeto
{
   /// <summary>
   /// Interaction logic for UserPage.xaml
   /// </summary>
   public partial class UserPage : Window
   {
      public UserPage()
      {
         InitializeComponent();
                        loadginasio();
                        loadmoda();
      }

                protected void loadginasio() //carrega a tabela Clientes
                {

                        using (ginasioEntities2 db = new ginasioEntities2())
                        {
                                var qcliente = (from c in db.Clientes select c).ToList();
                                gridcliente.ItemsSource = qcliente;
```

```
            }
        }

        protected void loadmoda() //Carrega a tabela Modalidades
        {

            using (ginasioEntities2 db = new ginasioEntities2())
            {
                var qmodalidade = (from c in db.Modalidades select c).ToList();
                gridmoda.ItemsSource = qmodalidade;









            }
        }


        private void btninscrever_Click(object sender, RoutedEventArgs e) //evento do
botão inscrever, ao clicar insere as informações na tabela
    {
      using (ginasioEntities2 db = new ginasioEntities2())
      {


        Cliente mycliente = new Cliente
        {

          iduser = int.Parse(txtuserid.Text),
          nome = txtname.Text,
          email = txtemail.Text,
          modalidade = blockmodalidade.Text,
                                fotopath = txtfoto.Text,



                              };
        db.Clientes.Add(mycliente);
        db.SaveChanges();
        gridcliente.ItemsSource = db.Clientes.ToList();

      }

    }
```

```
    private void btncancelar_Click(object sender, RoutedEventArgs e) //evento do botão cancelar, ao clicar remove as informações da tabela
            {
        using (ginasioEntities2 db = new ginasioEntities2())
        {
            int iduser;
            if (int.TryParse(txtuserid.Text, out iduser))
            {
                Cliente mycliente = db.Clientes.Where(x => x.iduser == iduser).FirstOrDefault();
                db.Clientes.Remove(mycliente);
                db.SaveChanges();
                gridcliente.ItemsSource = db.Clientes.ToList();
                                        txtuserid.Text = String.Empty;
                                        txtname.Text = String.Empty;
                                        txtemail.Text = String.Empty;
                                        fotoImage.Source = null;
                                        txtfoto.Text = String.Empty;
                                    }
        }
    }

            private void btnCarregar_Click(object sender, RoutedEventArgs e) //carrega uma imagem utilizado a class utils.cs
            {
                    OpenFileDialog dlg = new OpenFileDialog();
                    dlg.DefaultExt = ".png";
                    dlg.Filter = "*.*|*.*|JPEG Files (*.jpeg)|*.jpeg|PNG Files (*.png)|*.png|JPG Files (*.jpg)|*.jpg|GIF Files (*.gif)|*.gif";
                    dlg.Title = "Foto";
                    Nullable<bool> result = dlg.ShowDialog();

                    if (result == true)
                    {
                            string fich = txtuserid.Text + System.IO.Path.GetExtension(dlg.FileName);
                            txtfoto.Text = utils.caminhoAppSubPasta("imagens", fich);
                            fotoImage.Source = utils.picpelocaminho(dlg.FileName);
                            utils.Save((BitmapImage)fotoImage.Source, txtfoto.Text);


                    }
            }

            private void gridcliente_SelectionChanged(object sender, SelectionChangedEventArgs e) //ao clicar na grid cliente mostra as informações nas textbox
            {
                    DataGrid lst = (DataGrid)sender;
                    if (lst.SelectedItem != null)
                    {
                            Cliente a = lst.SelectedItem as Cliente;
```

```
                    if (a.iduser.ToString() != null) txtuserid.Text = a.iduser.ToString();
                    if (a.nome != null) txtname.Text = a.nome;
                    if (a.fotopath != null)
                    {
                        txtfoto.Text = a.fotopath;
                        fotoImage.Source = utils.picpelocaminho(a.fotopath);

                    }


                }
            }

            private void btnsair_Click(object sender, RoutedEventArgs e) //sair da janela Userpage para a janela Login
            {
                Login p = new Login();
                p.Show();
                this.Close();
            }

            private void gridmoda_SelectionChanged(object sender, SelectionChangedEventArgs e) //ao clicar na grid moda mostra a informação na textblock Modalidade
            {
                DataGrid lst = (DataGrid)sender;
                if (lst.SelectedItem != null)
                {
                    Modalidade a = lst.SelectedItem as Modalidade;
                    if (a.Nome != null) blockmodalidade.Text = a.Nome;



                }
            }
        }
    }
```

Anexo VII – MainWindow.xaml

```
<Window x:Class="projeto.MainWindow"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:local="clr-namespace:projeto"
      mc:Ignorable="d"
      Title="Admin" Height="482.452" Width="836.779" Background="AliceBlue">
  <Grid Margin="0,0,2,0">
    <Grid.ColumnDefinitions>
      <ColumnDefinition Width="199*"/>
      <ColumnDefinition Width="65*"/>
    </Grid.ColumnDefinitions>
    <Grid x:Name="grid1" HorizontalAlignment="Left" Height="419" Margin="10,22,0,0"
VerticalAlignment="Top" Width="379">
      <DataGrid x:Name="gridginasio" AutoGenerateColumns="False" Grid.Column="0"
HorizontalAlignment="Left" Height="114" Margin="0,10,0,0" VerticalAlignment="Top"
Width="369" SelectionChanged="gridginasio_SelectionChanged" >

        <DataGrid.Columns>
          <DataGridTextColumn Header="Id" Binding="{Binding iduser}"/>
          <DataGridTextColumn Header="Nome" Binding="{Binding nome}"/>
          <DataGridTextColumn Header="Email" Binding="{Binding email}"/>
          <DataGridTextColumn Header="Modalidade" Binding="{Binding modalidade}"/>
          <DataGridTextColumn Header="Foto" Binding="{Binding fotopath}"/>
        </DataGrid.Columns>
      </DataGrid>
      <Button x:Name="btnsair" Content="Sair" HorizontalAlignment="Left"
Margin="0,399,0,0" VerticalAlignment="Top" Width="75" Click="btnsair_Click"/>
      <DataGrid x:Name="gridmoda" AutoGenerateColumns="False"
HorizontalAlignment="Left" Height="100" Margin="0,197,0,0" VerticalAlignment="Top"
Width="160" SelectionChanged="gridmoda_SelectionChanged">
        <DataGrid.Columns>
          <DataGridTextColumn Header="id" Binding="{Binding idmodalidade}"/>
          <DataGridTextColumn Header="Modalidades" Binding="{Binding Nome}"/>
          <DataGridTextColumn Header="Hora" Binding="{Binding Horas}"/>
        </DataGrid.Columns>
      </DataGrid>


      <TextBox x:Name="txtaddmod" HorizontalAlignment="Left" Height="23"
Margin="249,217,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"/>
      <Button x:Name="btnaddmod" Content="Adicionar" HorizontalAlignment="Left"
Margin="184,289,0,0" VerticalAlignment="Top" Width="75" Click="btnaddmod_Click"/>
      <Button x:Name="btnremovermod" Content="Remover" HorizontalAlignment="Left"
Margin="292,289,0,0" VerticalAlignment="Top" Width="75" Click="btnremovermod_Click"/>
      <Label Content="Modalidades" HorizontalAlignment="Left" Margin="225,148,0,0"
VerticalAlignment="Top"/>
```

```
        <Label Content="Nome" HorizontalAlignment="Left" Margin="175,214,0,0"
VerticalAlignment="Top"/>
        <Label Content="Horas" HorizontalAlignment="Left" Margin="175,244,0,0"
VerticalAlignment="Top"/>
        <TextBox x:Name="txthoras" HorizontalAlignment="Left" Height="23"
Margin="249,245,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"/>
        <TextBlock x:Name="blockidmod" HorizontalAlignment="Left" Margin="279,184,0,0"
TextWrapping="Wrap" VerticalAlignment="Top"/>


    </Grid>

    <Grid x:Name="grid2" HorizontalAlignment="Left" Height="419" Margin="405,22,0,0"
VerticalAlignment="Top" Width="412" Grid.ColumnSpan="2">

        <Label x:Name="lblnome" Content="Nome" HorizontalAlignment="Left"
Margin="40,133,0,0" VerticalAlignment="Top"/>
        <Label x:Name="lblemail" Content="Email" HorizontalAlignment="Left"
Margin="40,164,0,0" VerticalAlignment="Top"/>
        <Label Content="Informações do Cliente" HorizontalAlignment="Left"
Margin="41,22,0,0" VerticalAlignment="Top" RenderTransformOrigin="0.383,2.87"/>
        <TextBox x:Name="txtnome" HorizontalAlignment="Left" Height="23"
Margin="116,133,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"/>
        <TextBox x:Name="txtemail" HorizontalAlignment="Left" Height="23"
Margin="116,167,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"/>
        <Button x:Name="btneditar" Content="Editar" HorizontalAlignment="Left"
Margin="41,215,0,0" VerticalAlignment="Top" Width="75" Click="btneditar_Click"/>
        <Button x:Name="btnremover" Content="Remover" HorizontalAlignment="Left"
Margin="145,215,0,0" VerticalAlignment="Top" Width="75" RenderTransformOrigin="0.5,0.5"
Click="btnremover_Click"/>
        <Label x:Name="lblid" Content="N°" HorizontalAlignment="Left" Margin="40,91,0,0"
VerticalAlignment="Top"/>
        <TextBox x:Name="txtid" HorizontalAlignment="Left" Height="23" Margin="116,95,0,0"
TextWrapping="Wrap" VerticalAlignment="Top" Width="120"/>
    </Grid>
    <Image Grid.Column="1" HorizontalAlignment="Left" Height="100" Margin="50,121,0,0"
VerticalAlignment="Top" Width="100"/>
    <Image x:Name="fotoImage" Grid.Column="1" HorizontalAlignment="Left" Height="100"
Margin="50,121,0,0" VerticalAlignment="Top" Width="100"/>


  </Grid>

</Window>
```

Anexo VIII – MainWindow.xaml.cs

```
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 Microsoft.Win32;
using iTextSharp;
using iTextSharp.text;
using iTextSharp.text.pdf;
using iTextSharp.text.pdf.draw;
using System.Text.RegularExpressions;

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


    public MainWindow()
    {
      InitializeComponent();
      loadginasio();
                    loadmod();




    }


    protected void loadginasio() // carrega a tabela Clientes
    {

      using (ginasioEntities2 db = new ginasioEntities2())
      {
```

```
            var qginasio = (from c in db.Clientes select c).ToList();
            gridginasio.ItemsSource = qginasio;




        }
    }

            protected void loadmod() //carrega a tabela Modalidades
            {

                using (ginasioEntities2 db = new ginasioEntities2())
                {
                    var qmod = (from c in db.Modalidades select c).ToList();
                    gridmoda.ItemsSource = qmod;




                }
            }



            private void btneditar_Click(object sender, RoutedEventArgs e) //evento do botão
editar, edita a informação do cliente
    {
      using (ginasioEntities2 db = new ginasioEntities2())
      {
        int iduser;
        if (int.TryParse(txtid.Text, out iduser))
        {
          Cliente mycliente = db.Clientes.Where(x => x.iduser == iduser).FirstOrDefault();
          mycliente.nome = txtnome.Text;
          mycliente.email = txtemail.Text;


          db.SaveChanges();
          gridginasio.ItemsSource = db.Clientes.ToList();
```

```
        }

    }
}

    private void btnremover_Click(object sender, RoutedEventArgs e) //evento do botão remover, remove o cliente
    {
        using (ginasioEntities2 db = new ginasioEntities2())
        {
            int iduser;
            if (int.TryParse(txtid.Text, out iduser))
            {
                Cliente mycliente = db.Clientes.Where(x => x.iduser == iduser).FirstOrDefault();
                db.Clientes.Remove(mycliente);
                db.SaveChanges();
                gridginasio.ItemsSource = db.Clientes.ToList();
                                        txtid.Text = String.Empty;
                                        txtnome.Text = String.Empty;
                                        txtemail.Text = String.Empty;
                                        fotoImage.Source = null;
                                }
        }
    }



    private void gridginasio_SelectionChanged(object sender, SelectionChangedEventArgs e) //ao selecionar mostra a informação da grid ginasio na textbox
    {
        DataGrid lst = (DataGrid)sender;
        if (lst.SelectedItem != null)
        {
            Cliente a = lst.SelectedItem as Cliente;
            if (a.iduser.ToString() != null) txtid.Text = a.iduser.ToString();
            if (a.nome != null) txtnome.Text = a.nome;
                                if (a.email != null) txtemail.Text = a.email;
                                if (a.fotopath != null)
                                {

                                        fotoImage.Source = utils.picpelocaminho(a.fotopath);

                                }


                        }
    }
```

```
        private void btnsair_Click(object sender, RoutedEventArgs e) //evento do botão sair, sai da window admin e volta para o Login
        {
            Login p = new Login();
            p.Show();
            this.Close();
        }

        private void btnaddmod_Click(object sender, RoutedEventArgs e) //evento do botão addmod, adiciona uma modalidade
        {
            using (ginasioEntities2 db = new ginasioEntities2())
            {


                Modalidade modalidade = new Modalidade
                {


                    Nome = txtaddmod.Text,
                    Horas = txthoras.Text,




                };
                db.Modalidades.Add(modalidade);
                db.SaveChanges();
                gridmoda.ItemsSource = db.Modalidades.ToList();

            }
        }

        private void btnremovermod_Click(object sender, RoutedEventArgs e) //evento do botão removermod, remove uma modalidade
        {
            using (ginasioEntities2 db = new ginasioEntities2())
            {
                int idmodalidade;
                if (int.TryParse(blockidmod.Text, out idmodalidade))
                {
                    Modalidade mymodalidade = db.Modalidades.Where(x => x.idmodalidade == idmodalidade).FirstOrDefault();
                    mymodalidade.Nome = txtaddmod.Text;
                    mymodalidade.Horas = txthoras.Text;


                    db.SaveChanges();
                    gridmoda.ItemsSource = db.Modalidades.ToList();
```

```
                }

            }
        }

        private void gridmoda_SelectionChanged(object sender,
SelectionChangedEventArgs e) //ao selecionar um elemento da grid moda, mostra a
informação na textbox
        {
            DataGrid lst = (DataGrid)sender;
            if (lst.SelectedItem != null)
            {
                Modalidade a = lst.SelectedItem as Modalidade;
                if (a.idmodalidade.ToString() != null) blockidmod.Text =
a.idmodalidade.ToString();
                if (a.Nome != null) txtaddmod.Text = a.Nome;
                if (a.Horas != null) txthoras.Text = a.Horas;



            }
        }
    }


}
```

Anexo VIX – Utils.cs

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Media;
using System.Windows.Media.Imaging;

namespace projeto
{


   public static class utils //guarda a imagem inserida
   {
      public static void Save(this BitmapImage image, string filePath)
      {
         BitmapEncoder encoder = new PngBitmapEncoder();
         encoder.Frames.Add(BitmapFrame.Create(image));

         using (var fileStream = new System.IO.FileStream(filePath, System.IO.FileMode.Create))
         {
            encoder.Save(fileStream);
         }
      }



      public static string caminhoAppSubPasta(string pasta, string fich) //caminho a utilizar para
guardar a imagem
      {
         string resposta = Environment.CurrentDirectory.Substring(0,
Environment.CurrentDirectory.IndexOf("projeto")) + @"projeto -
Copy\projeto\projeto\bin\imagens";
         resposta = System.IO.Path.Combine(resposta, fich);
         return resposta;
      }


      public static BitmapImage picpelocaminho(string path)
      {
         BitmapImage bi = new BitmapImage();
         bi.BeginInit();
         bi.DecodePixelWidth = 600;
         bi.CacheOption = BitmapCacheOption.OnLoad;
         bi.CreateOptions = BitmapCreateOptions.DelayCreation;
         bi.UriSource = new Uri(path, UriKind.Absolute);
```

```
            bi.EndInit();
            return bi;
        }

//----------------Converte em função da imagem que for escolhida--------------------//

        public static byte[] ConvertBitmapSourceToByteArray(BitmapEncoder encoder, ImageSource
imageSource)
        {
            byte[] bytes = null;
            var bitmapSource = imageSource as BitmapSource;
            if (bitmapSource != null)
            {
                encoder.Frames.Add(BitmapFrame.Create(bitmapSource));
                using (var stream = new MemoryStream())
                {
                    encoder.Save(stream);
                    bytes = stream.ToArray();
                }
            }

            return bytes;
        }

        public static byte[] ConvertBitmapSourceToByteArray(BitmapSource image)
        {
            byte[] data;
            BitmapEncoder encoder = new JpegBitmapEncoder();
            encoder.Frames.Add(BitmapFrame.Create(image));
            using (MemoryStream ms = new MemoryStream())
            {
                encoder.Save(ms);
                data = ms.ToArray();
            }
            return data;
        }
        public static byte[] ConvertBitmapSourceToByteArray(ImageSource imageSource)
        {
            var image = imageSource as BitmapSource;
            byte[] data;
            BitmapEncoder encoder = new JpegBitmapEncoder();
            encoder.Frames.Add(BitmapFrame.Create(image));
            using (MemoryStream ms = new MemoryStream())
            {
                encoder.Save(ms);
                data = ms.ToArray();
            }
            return data;
        }
```

```
public static byte[] ConvertBitmapSourceToByteArray(Uri uri)
{
   var image = new BitmapImage(uri);
   byte[] data;
   BitmapEncoder encoder = new JpegBitmapEncoder();
   encoder.Frames.Add(BitmapFrame.Create(image));
   using (MemoryStream ms = new MemoryStream())
   {
      encoder.Save(ms);
      data = ms.ToArray();
   }
   return data;
}
public static byte[] ConvertBitmapSourceToByteArray(string filepath)
{
   var image = new BitmapImage(new Uri(filepath));
   byte[] data;
   BitmapEncoder encoder = new JpegBitmapEncoder();
   encoder.Frames.Add(BitmapFrame.Create(image));
   using (MemoryStream ms = new MemoryStream())
   {
      encoder.Save(ms);
      data = ms.ToArray();
   }
   return data;
}

public static BitmapImage ConvertByteArrayToBitmapImage(Byte[] bytes)
{
   var stream = new MemoryStream(bytes);
   stream.Seek(0, SeekOrigin.Begin);
   var image = new BitmapImage();
   image.BeginInit();
   image.StreamSource = stream;
   image.EndInit();
   return image;
}

public static byte[] BufferFromImage(BitmapImage imageSource)
{
   Stream stream = imageSource.StreamSource;
   byte[] buffer = null;

   if (stream != null && stream.Length > 0)
   {
      using (BinaryReader br = new BinaryReader(stream))
      {
         buffer = br.ReadBytes((Int32)stream.Length);
      }
   }
```

```
            return buffer;
        }


    }//classe
}
```