INSTITUTO SUPERIOR DE TECNOLOGIAS AVANÇADAS

Projeto Global

Virtualização e Conceção de uma Biblioteca Digital

Licenciatura Informática 2015/2016 - Lisboa

Realizado por : Tiago Alexandre Veiga Peres de Soveral, Nº 1934

Coordenador : Professor Doutor Pedro Brandão

Lisboa 2016

(Esta página foi intencionalmente deixada em branco)

Dedicatória

Á minha mulher…

Agradecimentos

Á minha família pelo apoio e convicção em mim.

Aos meus estimados colegas de curso Nuno Mendes ("o master"), Ricardo Nunes, Ricardo Morais, Gonçalo Abreu, Nuno Azevedo, Duarte Bizarra e Hélder Barradas.

Aos meus orientadores Professor Doutor Pedro Brandão e Professor José Neves pelo suporte mesmo no pouco tempo que dispunham, pelas suas correções e incentivos.

A todos que direta ou indiretamente fizeram parte da minha formação, o meu muito Obrigado.

Resumo

Virtualização não é apenas uma "moda" que está a ser adotada, existem benefícios reais e mensuráveis para a sua implementação em *Data Centers*, mesmo em ambientes de Tecnologias de Informação (TI) de pequenas e médias dimensões.

Este documento descreve um trabalho autónomo de natureza académica realizado ao longo do ano letivo, no âmbito da disciplina de Projeto Global, onde o tema geral retratado foi exatamente a Virtualização e a conceção de uma biblioteca digital e a sua interligação à base de dados (BD).

Palavras-chave: Virtualização, base de dados, Hyper-V, Virtual Machine Manager, *hypervisor*.

Abstract

Virtualization is not just a "fashion" that is being adopted, there are real and measurable benefits to its implementation in data centers, even in Information Technology (IT) environments of small and medium dimensions.

This document describes an independent work This document describes an autonomous work of an academic nature carried out throughout the school year, under the Global Project discipline, where the depicted general theme was exactly the virtualization and the implementation of a digital library and its connection to the database (DB).

Palavras-chave: Virtualization, databases, Hyper-V, Virtual Machine Manager, *hypervisor*.

Índice

Índice de Figuras

Índice de Tabelas

Lista de Abreviaturas

AD – Active Directory

BD – Base de dados

CSV – Cluster Shared Volumes

DC – Domain controller

DHCP – Dynamic Host Configuration Protocol

DNS – Domain Name System

FQDN – Fully Qualified Domain Name

GPO – Group Policy Object

GUI – Graphical User Interface

IP – Internet Protocol

LAN – Local Area Network

NAT – Network Address Translation

NLB – Network Load Balancing

OLTA – Online Transaction Analytical

OLTP – Online Transaction Processing

OU – Organizational Unit

PDF – Portable Document Format

RDP – Remote Decktop User

SCVMM – System Center Virtual Machine Manager

TCP – Transmission Control Protocol

TI – Tecnologias de Informação

UDP – User Datagram Protoco

VM – Virtual Machine

WPF – Windows Presentation Foundation

XAML – Extensible Application Markup Language

1. Introdução

Virtualização é a técnica que permite particionar um único sistema computacional em vários outros, denominados máquinas virtuais. Cada máquina virtual oferece um ambiente completo similar a uma máquina física, sendo que cada uma destas pode ter o seu sistema operativo, serviços de rede e aplicações.

A virtualização tornou-se assim numa tecnologia fundamental para o desenvolvimento das infraestruturas de tecnologias de informação (TI).

1.1. Objetivos

O projeto global tem como objetivo principal a consolidação dos conhecimentos adquiridos ao longo do estudo académico e a sua demonstração desde a fase de planeamento até à fase de implementação e documentação do processo. A perceção de como tudo funciona e a maneira como se desenvolve, proporciona uma visão realista e prática da área de informática.

A criação de uma biblioteca digital e a sua disponibilização numa infraestrutura virtualizada adequada, foram as duas componentes exploradas neste projeto.

A implementação de soluções de virtualização e *cloud computing* apresenta vários objetivos tais como:

- Reutilização e partilha de recursos físicos;
- Ampliação da infraestrutura;
- Gestão centralizada;
- Independência de Hardware;
- Economia de espaço físico;

- Diversidade de plataformas;
- Entre outros.

Na implementação da solução apresentada foram utilizadas várias ferramentas adequadas à sua função, entre elas o VMware que contém os servidores virtuais criados, o Hyper-V que disponibiliza *desktops* virtuais, e o System Center Virtual Machine (SCVMM) Manager que gere os ambientes de virtualização. Nomeadamente também o SQL Server para o armazenamento e gestão da base de dados, o Visual Studio para o desenvolvimento da aplicação e a ADO.Net para a ligação com a BD, são outras das ferramentas utilizadas neste projeto.

1.2. Estrutura do documento

Este documento está organizado da seguinte forma:

- 2. Estado da Arte – Apresentação pormenorizada das tecnologias utilizadas na implementação do projeto.
- 3. Contextualização – Apresentação dos requisitos necessários ao desenvolvimento do projeto de forma sucinta.
- 4. Infraestrutura – Descrição da instalação e as configurações necessárias nos servidores ao nível da virtualização, rede, administração de sistemas e base de dados.
- 5. Aplicação – Descrição do funcionamento da aplicação biblioteca digital, a ligação á base de dados, a interface e o código utilizado para o desenvolvimento da mesma.
- 6. Conclusão – Constatação final dos resultados e abordagem crítica do trabalho realizado.

2. Estado de Arte

2.1. Virtualização

A virtualização é uma tecnologia já com alguma história, que remonta aos finais dos anos 60 e início dos anos 70, surgindo com a IBM.

O que é a virtualização e para que serve?

A virtualização é uma solução de partilha de recursos físicos de uma máquina (CPU, RAM, Disco Rígido) entre várias máquinas virtuais criadas[1]. Esta solução permite a alocação/realocação desses recursos sem que haja necessidade de agendamento ou de suspensão de serviços executados pelos administradores de sistemas. (Ruest & Ruest, 2009)

O conceito de virtualização descreve de modo geral a separação de recursos ou pedidos de serviços de um componente físico que fornece esse serviço. Um dos benefícios associados é a capacidade de disponibilizar mais recursos do que aqueles que existem na realidade. Por exemplo, a capacidade de memória virtual pode ser aumentada até mais do que a que existe fisicamente, recorrendo a permuta de dados com a memória do/s disco/s rígido/s.

Semelhantes técnicas de virtualização podem ser aplicadas em outras camadas das infraestruturas das tecnologias de informação (TI), tais como redes, *storage*[2], *hardware*[3] de computadores ou servidores, sistemas operativos e aplicações. Uma mistura destas técnicas proporciona uma camada de abstração entre a computação do *hardware* e as aplicações executadas no mesmo. (VMware, 2006)

[1] Computadores lógicos que apresentam resultados idênticos ao de um computador físico. Não existem fisicamente.
[2] Repositório físico de dados que providencia acesso a dados de forma rápida e fiável.
[3] Parte física de um computador, é formado pelos componentes eletrônicos, como por exemplo, circuitos de fios e luz, placas, utensílios, correntes, e qualquer outro material em estado físico

Como funciona?

Uma solução de virtualização é composta por dois elementos essenciais ao seu funcionamento, os quais denominam-se por *host* e *guest*. O primeiro, entende-se como o sistema operativo e os seus componentes de *hardware* provenientes da máquina física a fornecer o serviço. O segundo por sua vez, será o sistema virtualizado que é executado pelo *host*.

A virtualização ocorre quando é efetuada uma operação de emulação e partilha de recursos por parte de uma plataforma chamada Hypervisor[4], implementada no *host*, que recebe esses mesmos recursos a serem virtualizados, controlando-os e mantendo-os "invisíveis" em relação aos outros. (Rosa, 2010)

A plataforma Hypervisor pode ser classificada em dois tipos com o intuito de oferecer as funcionalidades descritas (Chenley, 2011):

- Hypervisor do Tipo 1 - denominado de *hypervisor bare metal*, é um software que interage diretamente sobre o *hardware* da máquina física, sendo completamente independente do sistema operativo do *host*.
- Hypervisor do Tipo 2 - denominado de *hypervisor hosted,* é um software que é executado sobre o sistema operativo do *host*, sendo a camada hypervisor como um segundo nível de *software*, e os Sistemas Operativos *guest* (das máquinas virtuais) executados num terceiro nível, acima do *hardware*.

[4] É uma plataforma que permite aplicar diversas técnicas de controlo de virtualização para utilizar, ao mesmo tempo, diferentes sistemas operativos no mesmo computador.

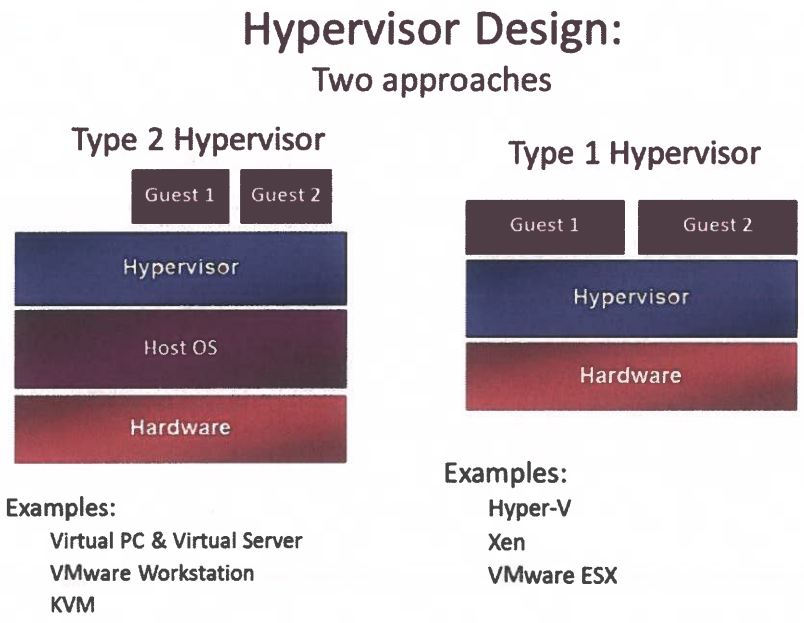

Figura 1 - Hypervisor Design (Chenley, 2011a)

Quais as suas vantagens?

As organizações de TI cada vez mais desejam usar a virtualização quando pretendem implementar aplicações ou serviços críticos necessários aos seus negócios ou para eventuais picos de carga. Nomeadamente o processamento de transações *on-line* (OLTP), bases de dados e soluções de análise de transações *on-line* (OLTA), que normalmente são executados em sistemas com 16 ou mais processadores e que exigem grandes quantidades de memória. Para esses picos de trabalho, e determinados períodos de tempo, a vantagem de adicionar mais processadores e memória virtual a uma máquina virtual são um requisito fundamental. (Advantages & Server, 2013)

Algumas das vantagens da virtualização já foram mencionadas, no entanto a sua utilização oferece vários outros benefícios apresentados de seguida (Adams, 2011):

- Melhoria de aproveitamento da infraestrutura existente;
- Diminuição do parque de máquinas;
- Gestão centralizada;
- Implementação mais célere;
- Utilização de sistemas legados;
- Diversidade de plataformas;
- Avaliação de novos sistemas ou atualizações;
- Segurança e confiabilidade;
- Facilidade de migração e ampliação de infraestrutura.

Soluções de virtualização

No mercado existem várias soluções de virtualização que atendem aos vários tipos de objetivos aos quais as empresas estão à procura. As mais conhecidas são o VMware, Microsoft, Citrix Xen e Virtual Box. (Adams, 2011)

2.2. Hyper – V

A tecnologia de virtualização disponibilizada pela Microsoft, integrada no seu sistema operativo, chamada Hyper-V, pode ser instalada na sua versão *Core* ou em instalações completas do sistemas operativo Windows Server. A diferença entre a sua versão core e a completa é a interface de visualização, ou seja, a primeira é composta apenas por linhas de comando, enquanto a segunda tem uma interface gráfica para com o utilizador (GUI[5]). O Hyper-V tem como base o conceito de *hypervisor*, onde é instalado uma camada de *software* que é executada entre o *hardware* e o sistema operativo do *host*. Apesar das máquinas virtuais, como os sistemas operativos *host,* serem instaladas em partições diferentes, estas acedem ao *hardware* da mesma forma democraticamente. (Rosa, 2010)

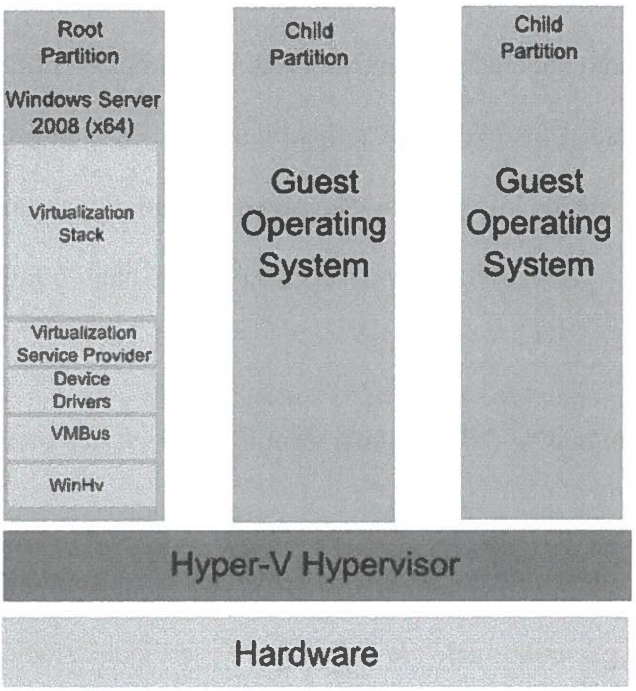

Figura 2 - Hyper-V ("Hyper-v_partitions," 2016)

[5] *Graphical User Interface*, é a interface gráfica visível pelo utilizador

Um dos benefícios de implementar servidores virtuais é a oportunidade de facultar alta-disponibilidade, tanto para aplicações ou serviços que por si só podem ter funcionalidade de alta-disponibilidade para aplicações ou serviços que não dispõem dessas funcionalidades. Nomeadamente o Hyper-V obtém alta disponibilidade com componentes como o *Failover Cluster* e o *Network Load Balancing*.

O *Failover Cluster* é um conjunto de servidores independentes que trabalham em conjunto para aumentar a disponibilidade e escalabilidade de funções de *cluster*[6]. Os servidores que fazem parte de um *cluster* (chamados de nós) são conectados por cabos físicos e por *software*. Se um ou mais dos nós do *cluster* falhar, o outro nó começará a fornecer o serviço (um processo conhecido como *failover*). Além disso, as funções de *cluster* são monitorizadas de maneira proactiva para verificar se estão a funcionar adequadamente. Se o funcionamento não estiver correto, tais funções serão reiniciadas ou movidas para outro nó. Os *Failover Cluster* também fornecem a funcionalidade CSV (Volume Compartilhado *Cluster*) que por sua vez, oferece um método consistente distribuído, o qual pode ser usado para aceder ao armazenamento partilhado em todos os nós. Com o recurso *Failover Cluster*, é possível obter o mínimo de interrupções de serviço.

O *Network Load Balancing* (NLB) é uma tecnologia de *cluster* oferecida pela Microsoft que permite aumentar e garantir a disponibilidade e escalabilidade de serviços de rede. O NLB permite a criação de um endereço IP virtual para que os utilizadores possam aceder as aplicações ou serviços garantidos pelo *cluster* em vários servidores. (Microsoft, 2012)

[6] Grupo de servidores a funcionar em conjunto de forma a manter alta disponibilidade.

2.3. SQL 2014

A base de dados (BD) não é mais que uma relação de informação. Uma lista de telefone é por exemplo uma BD com nomes, números e endereços das pessoas que vivem na região. Uma lista de telefone é certamente uma BD omnipresente e frequentemente utilizada, no entanto sofre com o seguinte:

- Elevado tempo de resposta – Encontrar o número de telefone de uma pessoa pode ser demorado, especialmente se o livro de telefone contiver um grande número de entradas.
- Ordenação unilateral – A lista telefónica é indexada apenas por os últimos nomes, assim encontrar os nomes das pessoas que vivem em um determinado endereço, em teoria, não é um uso prático para esta base de dados.
- Atualização não periódica – A partir do momento em que a lista de telefone é impressa, a informação que contém torna-se cada vez menos precisa, pois as pessoas deslocam-se para dentro ou fora de uma região, alteraram os seus números de telefone, ou mudam-se para outro local dentro da mesma região.

Os mesmos inconvenientes atribuídos a listas de telefone podem ser também aplicados a qualquer sistema manual de armazenamento de dados, como os registos dos pacientes armazenados num armário de arquivo. Devido à complexidade das BDs em papel, algumas das primeiras aplicações informáticas desenvolvidas foram sistemas de base de dados, que são mecanismos informatizados de armazenamento e recuperação dos mesmos. Um sistema que armazena BDs eletronicamente em vez de papel é capaz de recuperar dados mais rapidamente, indexando os dados de várias maneiras, e fornecendo informações de modo célere a quem a pesquisa.(Beaulieu, 2009)

O *Structured Query Language* (SQL) é uma linguagem que é utilizada para comunicar com uma base de dados. De acordo com ANSI (*American National Standards Institute*), é a linguagem padrão para sistemas de gestão de base de dados relacionais. O SQL inclui instruções que são usadas para executar tarefas como atualizar dados ou recuperar dados de uma BD. Alguns dos sistemas de gestão bases de dados relacionais mais comuns na utilização de SQL são: Oracle, Sybase, Microsoft SQL Server, Access, Ingres, etc. Embora a maioria dos sistemas de base de dados utilizem SQL, a maioria deles também têm as suas próprias extensões proprietárias adicionais que são normalmente utilizadas apenas em sistemas proprietários. No entanto, os comandos padrão do SQL, como "Select", "Insert", "Update", "Delete", "Create" e "Drop" podem ser utilizados para realizar quase todas as operações necessárias numa base de dados. ("Interactive Online SQL Training," 2016)

O SQL Server 2012 traz novas funcionalidades, melhorias de performance e capacidades expansão para a *Cloud*. Melhorias ao nível do mecanismo da BD como tabelas com otimização de memória, replicação das BDs para Azure, redundância e alta disponibilidade. (Mistry & Misner, 2014)

2.4. ADO.NET

ADO.NET é o componente de acesso de dados .NET Framework da Microsoft. A Microsoft construiu a ADO.NET como "uma melhoria evolutiva" em relação às versões anteriores do ADO. É certamente verdade que o modelo objeto ADO.NET leva muito pouca relação com versões anteriores do ADO.(Rebecca M.Riordan, 2002)

O ADO é um conjunto de classes de acesso a dados relacionais e fornece transparência usando uma interface comum para aceder a várias fontes de dados, utilizando um conjunto de objetos, tais como *Connection*, *Command* e *Records*, o *Parameter*, *Property*, *Field* e *Error*, e um conjunto de objetos subsidiários que são acedidos através de coleções de *Parameters*, *Properties*, *Fields* e *Errors* respetivamente. Todos esses objetos e conjuntos têm uma coleção de propriedades que contém objetos de propriedade. Os objetos ADO apresentam propriedades, métodos e eventos capazes de manipular o objeto e o seu conteúdo.(Lendvai & Shi, 2007)

Assim o ADO.NET fornece o acesso a bases de dados, como por exemplo o SQL Server, tal como outras fontes acedem via OLEDB, XML, ODBC. As aplicações podem utilizar o ADO.NET para estabelecer ligações a essas fontes de dados de modo a recuperar, manipular e atualizar dados. O ADO.NET pode ser utilizado num ambiente conectado ou desconectado. Num ambiente conectado, os utilizadores estão permanentemente ligados á base de dados, enquanto num ambiente desconectado, um subconjunto de dados pode ser copiado e modificado independentemente, e mais tarde introduzir essas mesmas alterações de novo na base de dados.(Ferreira, 2004)

Num ambiente desconectado utilizamos o *DataSet* para trazer todas as tabelas e relacionamentos necessários. Os *DataTables* semelhantes às tabelas da base de dados, gerem os valores de dados do código-fonte. Contém ainda outros componentes como *DataCollumn*,

DataRow, *DataRelations* e *DataViews*. Para conectar diretamente com a base de dados o ADO.NET contém vários tipos de *Data Providers*, incluindo o Microsoft SQL Server.(Patrick, 2010)

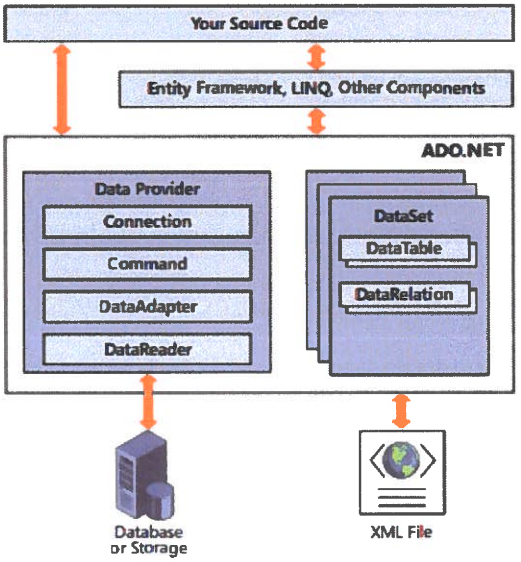

Figura 3 - ADO.NET (Patrick, 2010a)

3. Contextualização

O projeto consiste na criação de uma biblioteca digital, permitindo aos utilizadores credenciados o acesso ao conteúdo disponibilizado pela aplicação e a possibilidade de armazenamento de informação investigada por estes.

Para tal é necessário ter uma infraestrutura que suporte os requisitos da aplicação.

Tecnologia: a biblioteca tem um servidor físico com um *hypervisor*, que neste caso será simulado pelo computador portátil. No *hypervisor* estarão disponíveis as seguintes máquinas virtuais com as seguintes funções:

- Servidor Windows 2012 R2 Standard – Função de Controlador de Domínio (todo o cenário deve decorrer dentro de um domínio);
- Servidor Windows 2012 R2 Datacenter – Servidor SQL 2012 Enterprise, (servidor de base de dados para a aplicação bem como para a componente System Center VMM);
- Servidor Windows 2012 R2 Datacenter – Servidor Hyper-V e SC Virtual Machine Manager (disponibiliza a partir de um *template* a produção via Frabric de Máquinas Virtuais Cliente. Este servidor só é acessível aos administradores);
- Máquina Virtual Cliente com Windows Server 8.1 para acesso por parte dos utilizadores que utilizam a aplicação biblioteca a fim de consulta;
- Máquina Virtual Cliente com Windows Server 8.1 para acesso por parte dos utilizadores e administradores da aplicação afim de editar os conteúdos da aplicação biblioteca;
- O servidor com o SCVMM consiste numa implementação *standallone* e servirá exclusivamente para os administradores disponibilizarem máquinas virtuais

previamente definidas quando um utilizador as solicitar ao respetivo administrador. Os utilizadores não podem por si criar VM's.

Base de Dados:

- Construção de uma base de dados com SQL 2012 Enterprise.
- Criação das seguintes tabelas na base de dados: Autor, Titulo, Tipo de Documento (Livro, Jornal, Manuscrito, Imagem, etc), Editora, Fonte (caso não seja uma editora), PDF (a versão do documento em PDF), Capa (imagem da capa do documento, ou da 1ª página quando não for livro), Data (da edição ou produção do documento), Resumo (resumo do documento ficheiro de texto); e outros campos e tabelas necessárias para colocar todo este cenário a funcionar.
- O acesso á base de dados através da componente ADO.NET.

Interface de acesso à biblioteca:

- Existe um ícone no ambiente de trabalho dos computadores clientes que executará a aplicação da biblioteca. A interface foi desenvolvida para ser executada em dois modos, um para utilizadores (só consulta) outro para administradores (consulta e edição).
- O documento consultado fica disponível para ser feito o seu *download* em formato PDF.

Todo o cenário foi desenvolvido em VMware Workstation.

4. Infraestrutura

A infraestrutura é composta por todas as máquinas virtuais, servidores, computadores clientes e todas as configurações de sistemas como rede, permissões, políticas e instalações de aplicações. O utilizador comum ao fazer *login* na máquina cliente não necessita de fazer quaisquer configurações, pois são todas da responsabilidade da infraestrutura (*back-end*[7]).

Para este projeto e de modo a uniformizar a palavra passe (*password*) para todos os utilizadores criados, ficou definido como *password*: P@ssw0rd

Apesar de ser igual para todos cumpre com os requisitos de complexidade exigidos pela Microsoft que são:

- Não conter partes significativas do nome da conta ou nome do utilizador
- Ter, no mínimo, seis caracteres de comprimento
- Conter caracteres de três das seguintes quatro categorias:
    - Caracteres maiúsculos do alfabeto (de A a Z)
    - Caracteres minúsculos do alfabeto (de a a z)
    - Dígitos (de 0 a 9)
    - Caracteres não alfabéticos (por exemplo: !, $, #, %)

[7] Tudo o que não é visível para o utilizador mas que faz funcionar todo o sistema.

4.1. VMware

A infraestrutura implementada neste projeto ficou assente em *hypervisor* do tipo 2 VMware Workstation.

A nível de rede no VMware ficaram criados dois *switches* virtuais, NAT e VMnet4 (Figura 4), onde o primeiro está em modo NAT com acesso externo (à Internet) e o segundo em modo *Host-only*[8]. Cada máquina virtual foi configurada com duas placas de rede, Internet e LAN, em que uma é interligada ao *virtual switch* NAT e outra interligada ao *virtual switch* VMnet4 respetivamente.

Os três servidores virtuais criados contemplam as seguintes características:

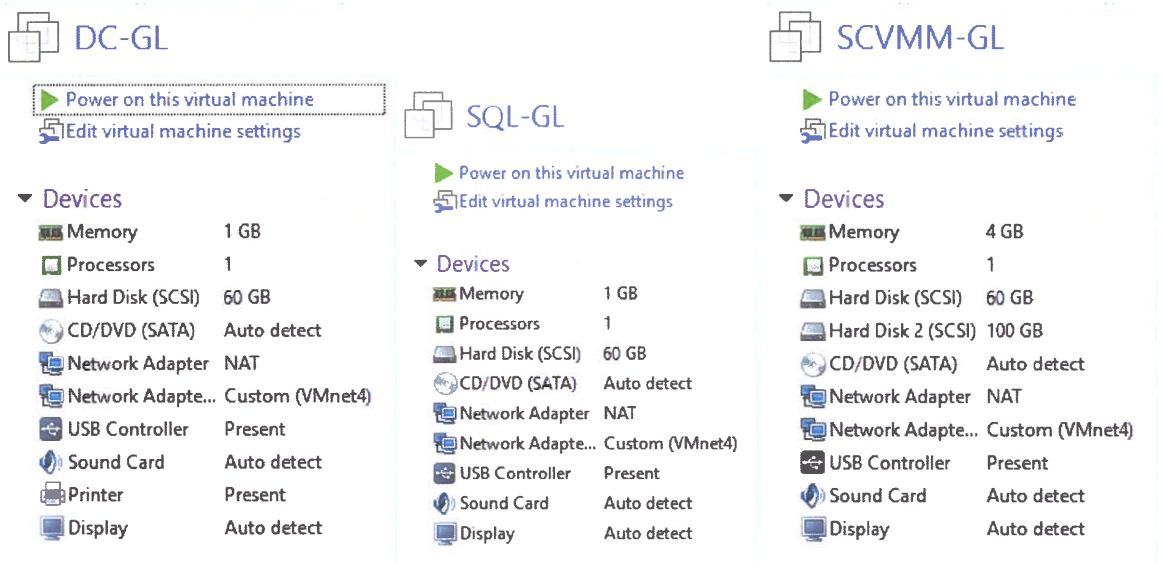

Figura 4- VMware Workstation VMs

No servidor SCVMM-GL é necessário efetuar duas alterações de modo a permitir a instalação da *role* Hyper-V (referida mais à frente).

[8] Conecta as VMs numa rede privada, sem acesso a nenhuma rede externa, apenas entre as VMs e o sistema *host*.

1ª Alteração - Com a máquina virtual desligada, selecionar nos *settings* do CPU a caixa "*virtualize intel vt-x/ept or amd-v/rvi*" e a caixa "*virtualized cpu performance counters*".

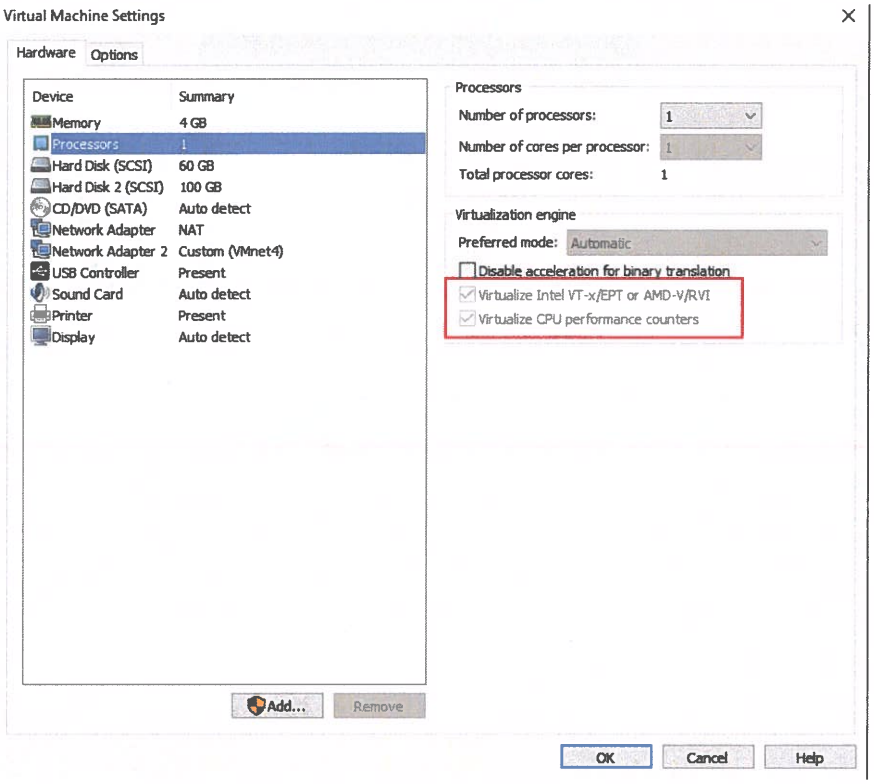

Figura 5 - VM SCVMM Settings

2º Alteração - Alterar as configurações da máquina virtual. Na diretoria da respetiva máquina virtual editar o ficheiro ".vmx" e acrescentar no final as seguintes linhas:

hypervisor.cpuid.v0 = "FALSE"

mce.enable = "TRUE"

vhu.enable = "TRUE"

4.2. Diagrama

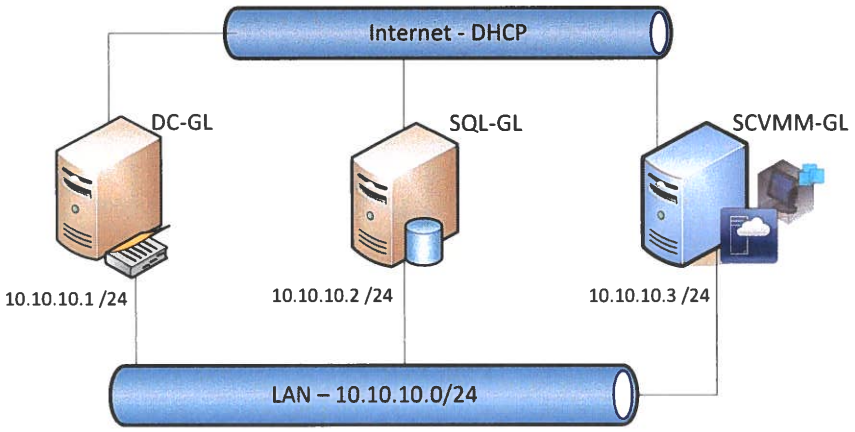

Figura 6 - Servers & Network

4.3. Servidor Domain Controller – DC-GL

Para realizar login neste servidor utilizar a seguinte conta de domínio:

User: geeklibrary\administrator

Pass: P@ssw0rd

4.3.1. Características do Servidor:

- Tipo: Servidor Virtual (VMware)
- Sistema Operativo: Windows Server 2012 R2 Standard
- Memória: 1 GB
- Hostname: DC-GL
- Configurações de interface LAN:
  - 10.10.10.1 / 255.255.255.0
- Configuração de interface INTERNET:
  - DHCP
- Configuração de Storage:
  - Sistema (C:) - 60GB

- Serviços Principais:
  - Active Directory
  - DNS
  - DHCP (para interface LAN)
  - File Share

4.3.2. Domain Controller

Para promover o servidor DC-GL a controlador de domínio (Domain Controller - DC), é imprescindível a instalação da *role* de Active Directory (AD). A AD é um serviço de domínio que serve essencialmente para armazenar dados e gerir as interações utilizador/domínio, autenticação e processos de início de sessão do utilizador, criação gestão de políticas de grupo, e gestão tanto de utilizadores como servidores/computadores que fazem parte desse mesmo domínio.

A estrutura lógica da AD pode ir da mais simples à mais complexa e consiste em vários termos, entre os quais Objetos, Unidades Organizacionais, Domínio, Árvores de Domínio e Floresta (Forest).

Quando é necessário criar um segundo domínio e temos um domínio "pai" com seus domínios "filhos", chamamos de árvore de domínio. Ao conjunto dessas várias árvores denominamos de Floresta.

Tando a Árvore como a Floresta pode ser feita de um único domínio, e é a situação realizada neste projeto, com apenas um controlador de domínio (o primeiro e o único).

Níveis funcionais de domínio e floresta:

Os níveis funcionais de domínio e floresta permitem ativar funcionalidades no ambiente de Serviços de Domínio da AD (Active Directory Domain Service - AD DS). Existem vários níveis diferentes disponíveis que podem ser selecionados consoante o ambiente de rede. Neste

caso e por ser uma instalação de raiz sem quaisquer precedentes de outros domínios e sistemas operativos antigos, foi selecionado o nível funcional mais recente de domínio e floresta.

Instalação da Role do Ative Directory:

No Server Manager clique em "Add roles and features".

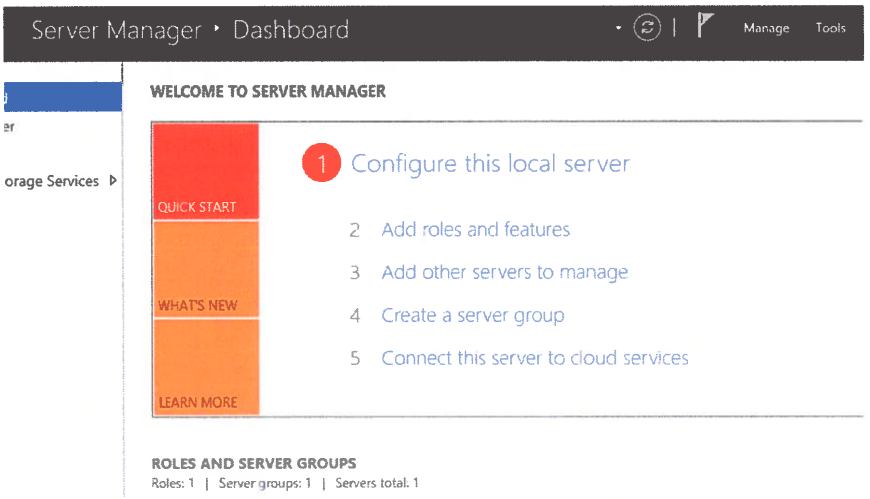

Figura 7 - Add Roles and features

Em tipo de instalação clique "Next"

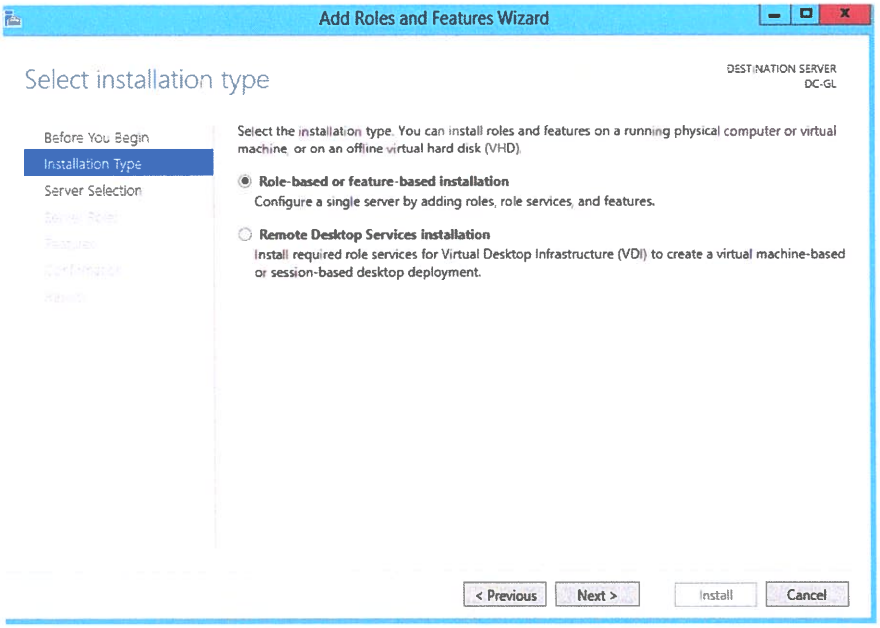

Figura 8 - Role based or feature based installation

No Servidor a que se destina a *Role* clique "Next"

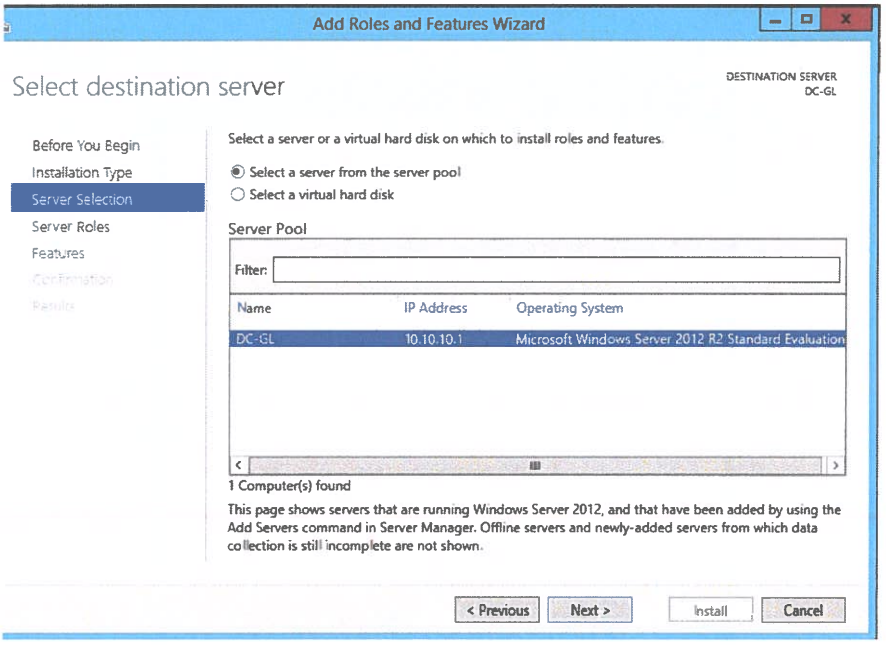

Figura 9 - Select destination server

Selecionar a caixa da role "Active Directory Domain Services" (AD DS).

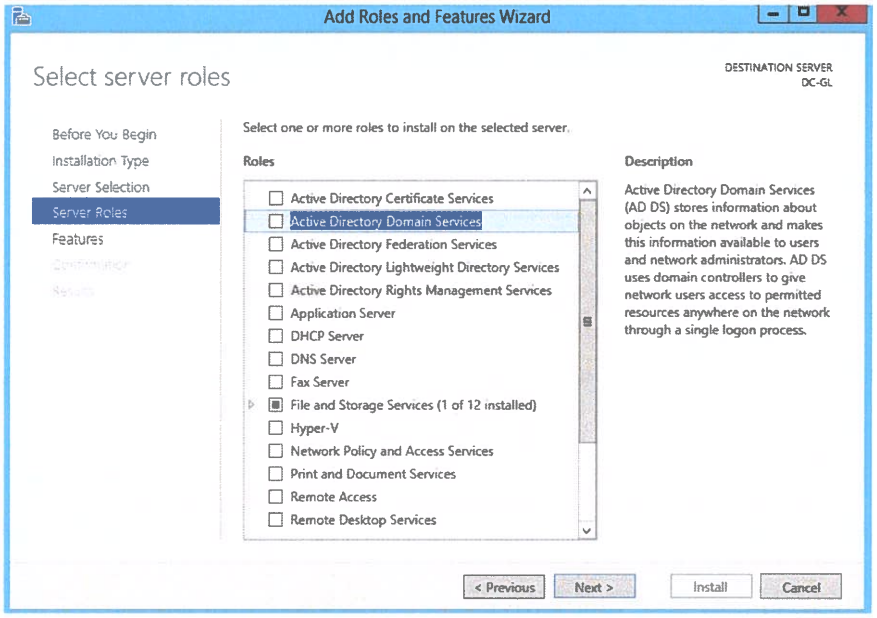

Figura 10 - Server Role Active Directory

De seguida vai aparecer a indicação das *features* necessárias à instalação da respetiva *role*. Clique em "Add Features"

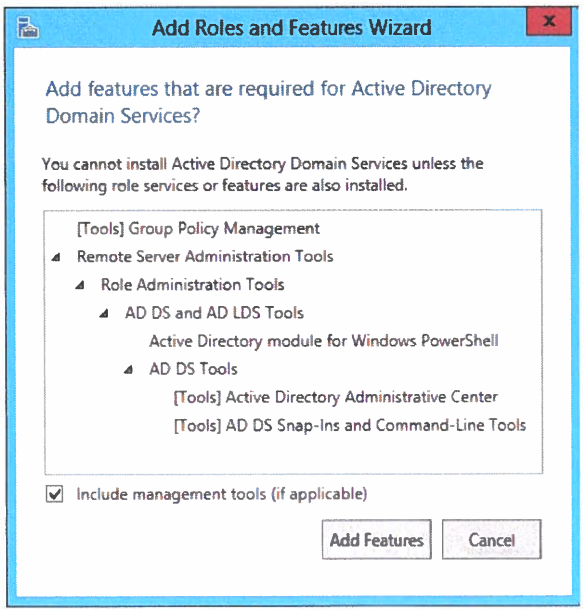

Figura 11 - Active Directory required features

Neste momento não será necessário adicionar mais qualquer *role* ou *feature*, pelo que deverá carregar em "Next" nas seguintes janelas e "Install" na última janela.

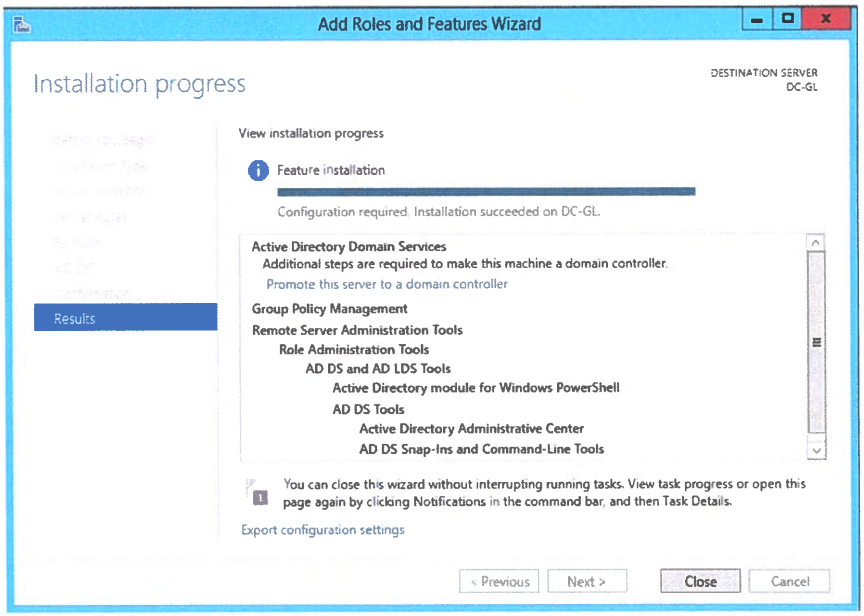

Figura 12 - AD DS installation progress

Após a instalação da *role* é necessário promover o servidor a Domain Controler, Clique em "Promote this server to a domain controller"

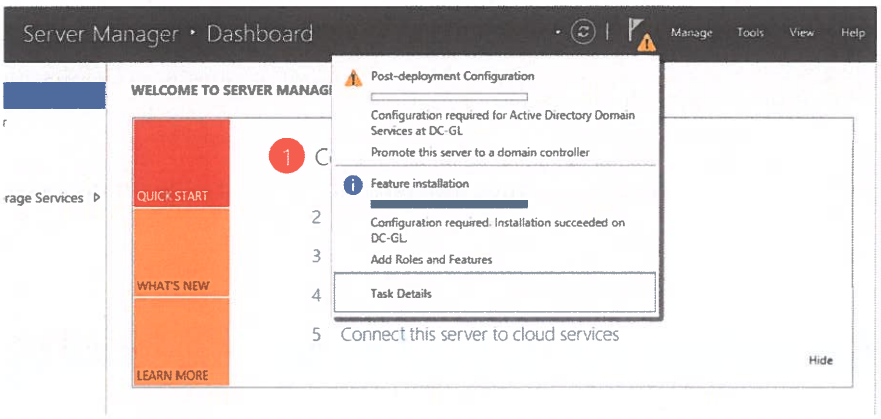

Figura 13 - Promote to a domain controller

A partir deste momento irá ser realizada as configurações do domínio.

Selecionar "Add a new Forest" e introduzir o nome domínio: geeklibrary.local

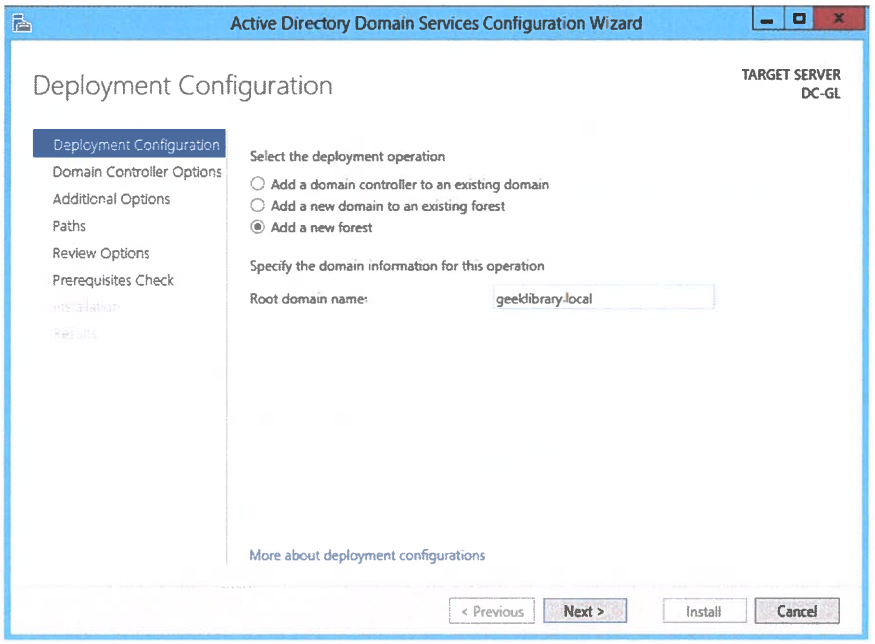

Figura 14 - Add a new forest

Selecionar o nível funcional tanto da floresta como do domínio para "Windows Server 2012 R2" conforme previamente explicado.

Selecionar a caixa "Domain Name System (DNS) server", pois pretende-se que este seja também o servidor de DNS deste domínio.

E será também neste ponto que definimos a *password* para o caso de ser necessário restaurar o serviço de diretório.

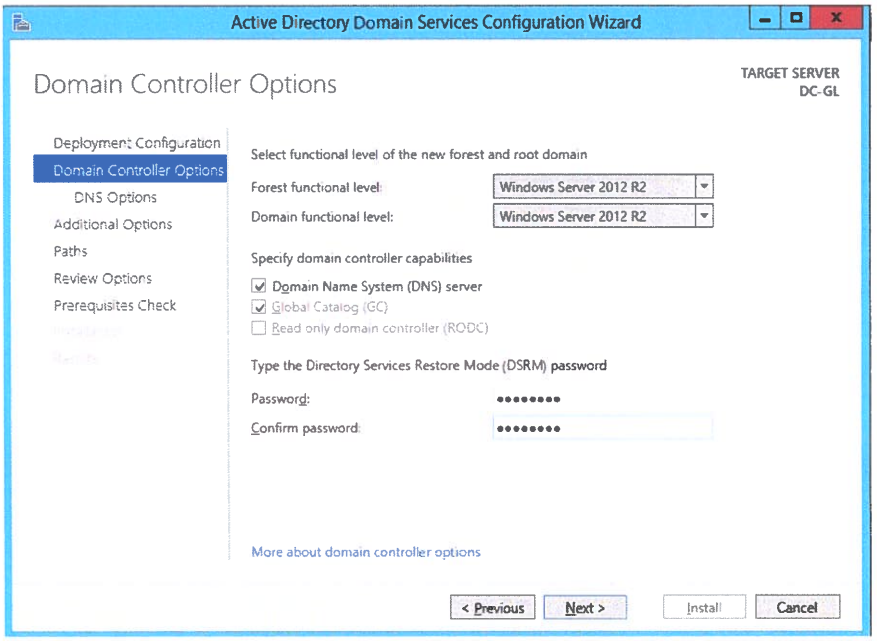

Figura 15 - Domain Controller Option

Na opção do DNS é apresentado um *warning* referindo que não foi possível encontrar a *root* de DNS, neste caso refere-se ao ".local" do domínio criado. É exatamente o que se pretende, não ter nenhuma dependência do nível de DNS acima. Clique em "Next".

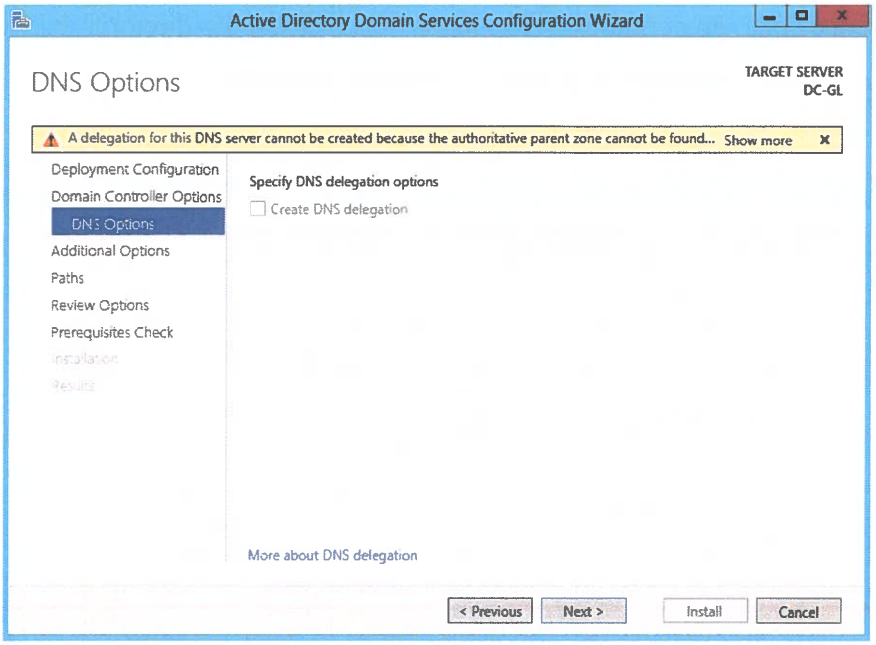

Figura 16 - DNS delegation Option

No nome de NETBIOS, colocar GEEKLIBRARY.

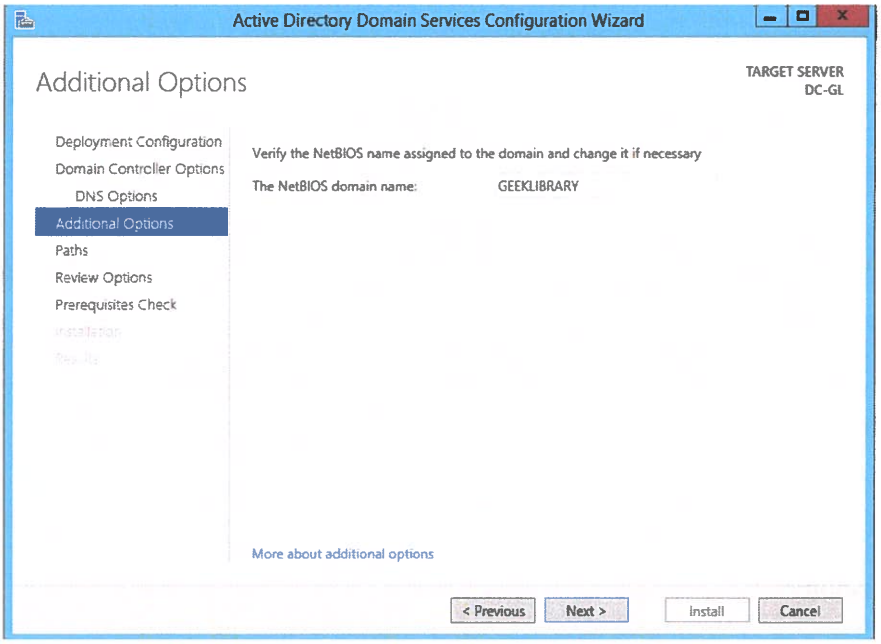

Figura 17 - NETBIOS domain name

Nesta caixa é onde são definidos os caminhos (localização) das diretorias para a Database, Log files e SYSVOL. Para este projeto manteve-se as definições predefinidas.

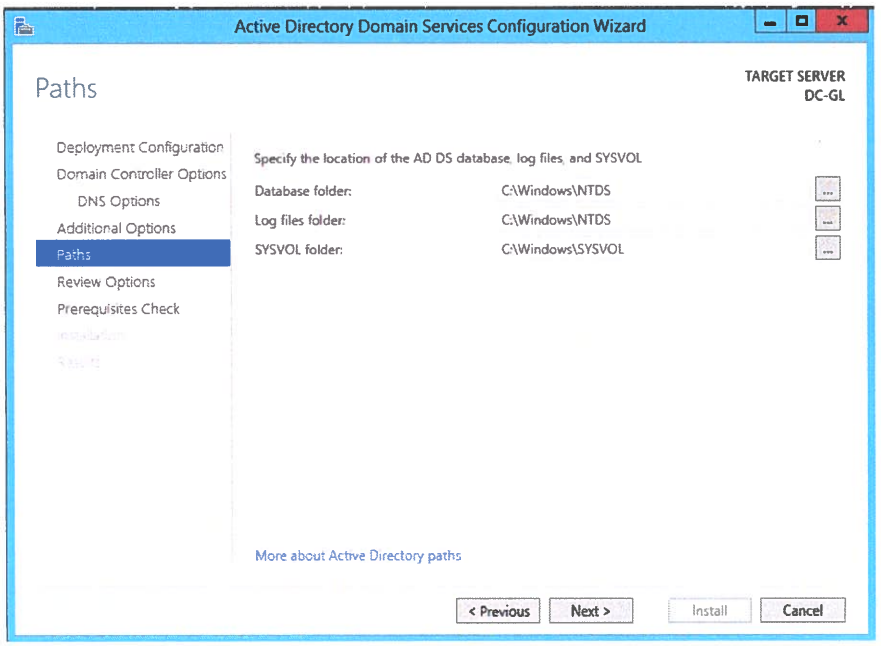

Figura 18 - AD DS folder paths

Validar e fazer "Next" nas duas janelas seguintes que são apenas um sumário para rever as configurações e o script de PowerShell gerado com as mesmas.

Na última janela verificar que os pré requisitos passaram com sucesso e de seguida clique em "Install".

Após a instalação o servidor é reiniciado.

Agora o servidor é um controlador de domínio com o Serviço Active Directory e DNS prontos a funcionar.

4.3.3. Active Directory – User and Computers

A consola dos "Users and Computer" pode ser considerada uma das mais importantes para a gestão de objetos da AD, entre os quais utilizadores, grupos, computadores, domínios, Organizational Units (OUs), etc.

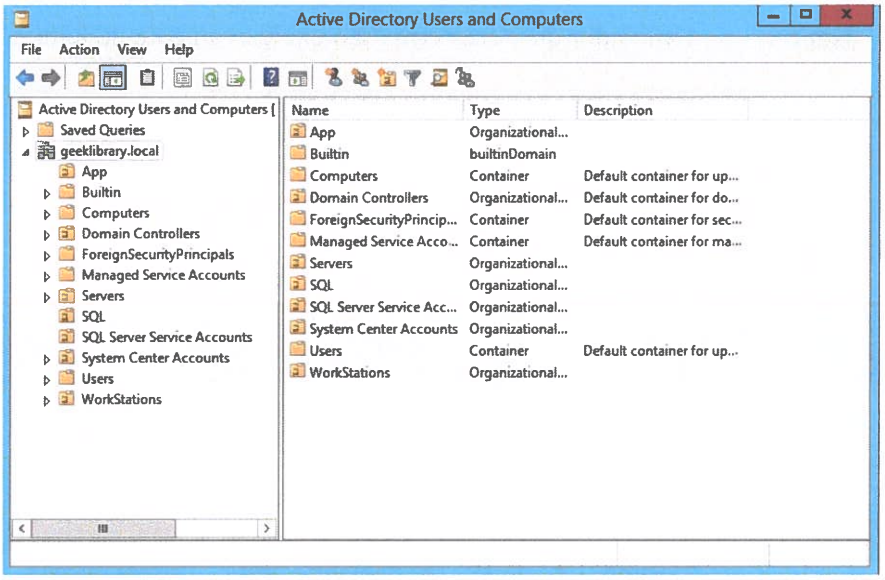

Figura 19 - Users and Computer console

Foram criadas as seguintes OUs:

Tabela 1 - Organizational Units

OU	Propósito
App	Para os utilizadores e grupos a serem utilizados pela aplicação da biblioteca
Servers	Para os servidores exceto o controlador de domínio.
SQL	Utilizadores a serem utilizados tanto para o servidor SQL como para o SQL Management Studio
SQL Service Accounts	Conta de serviço para o SQL a ser utilizado pelo SCVMM
System Center Accounts	Contas a serem utilizadas para o SCVMM
WorkStations	Local onde ficam os computadores clientes

Na OU App foram criados os seguintes utilizadores e grupos:

Tabela 2 - Utilizadores da OU App

Utilizadores	Descrição
master	Utilizador que terá permissão de administrar a aplicação
user1	Utilizador que terá permissão exclusivamente de leitura
user2	Utilizador que terá permissão exclusivamente de leitura

Tabela 3 - Grupos da OU App

Grupos	Descrição	Membros
AppAdmin	Grupo a que pertencem os utilizadores para ter permissão de administrar a aplicação	master
AppUsers	Grupo a que pertencem os utilizadores para ter permissão de administrar a aplicação	user1; user2
RDPUsers	Grupo a que pertencem os utilizadores para ter acesso remoto às máquinas cliente.	master; user1

A OU "Servers" serve para colocar todos os servidores exceto o controlador de domínio. Esta OU será usada para associar as GPOs (Group Policy Objects) que se pretende que afetem os respetivos servidores.

Na OU SQL foi criado o utilizador "sqladmin" e o grupo a que pertence "SQLAdmins".

Na OU SQL Services Acounts foi criada apenas a conta "sql_svc" que servirá como conta de serviço para base de dados do System Center Virtual Machine Manager.

Para a OU System Center Accounts foram apenas um grupo "SCVMMAdmins" a qual pertencem todos os seguintes utilizadores:

Tabela 4 - Utilizadores da OU System Center Accounts

Utilizadores	Descrição
scvmm_admin	Administrador do Systems Center Virtual Machine Manager (SCVMM)
scvmm_runas	Conta para executar o serviço do SCVMM

Na OU Worksation é onde ficaram todas a maquinas clientes (neste caso Windows 8.1).

Sempre que uma nova máquina é adicionada ao domínio fica associada automaticamente à OU Computers sendo necessário mover manualmente para outra OU pretendida.

Como as máquinas clientes serão criadas através do SCVMM de modo a ficarem logo na OU correta WorkStations, onde já estão associadas algumas GPOs, foi necessário alterar a *default* OU.

Para realizar esta operação, abrir o command prompt no servidor DC-GL, e dentro do System32 executar o seguinte comando:

redircmp ou=WorkStations,DC=geeklibrary,dc=local

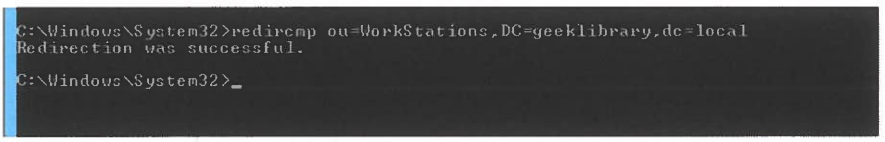

Figura 20 - Default OU redirect

A partir deste momento a *default* OU passou a ser a OU WorkStation.

4.3.4. Active Directory – Group Policy

A consola Politicas de grupo fornece uma gestão e configuração centralizadas do sistema operativo e das aplicações que são executadas no mesmo.

As configurações das políticas de grupo de domínio estão armazenadas em Group Policy Object (GPO) e dividem-se entre "*User Configuration*" (Configuração do Utilizador), que contém as definições que são aplicadas aos utilizadores quando iniciam sessão, e a "*Computer Configuration*" (Configuração do Computador) que contém as definições que são aplicadas aos computadores quando estes iniciam.

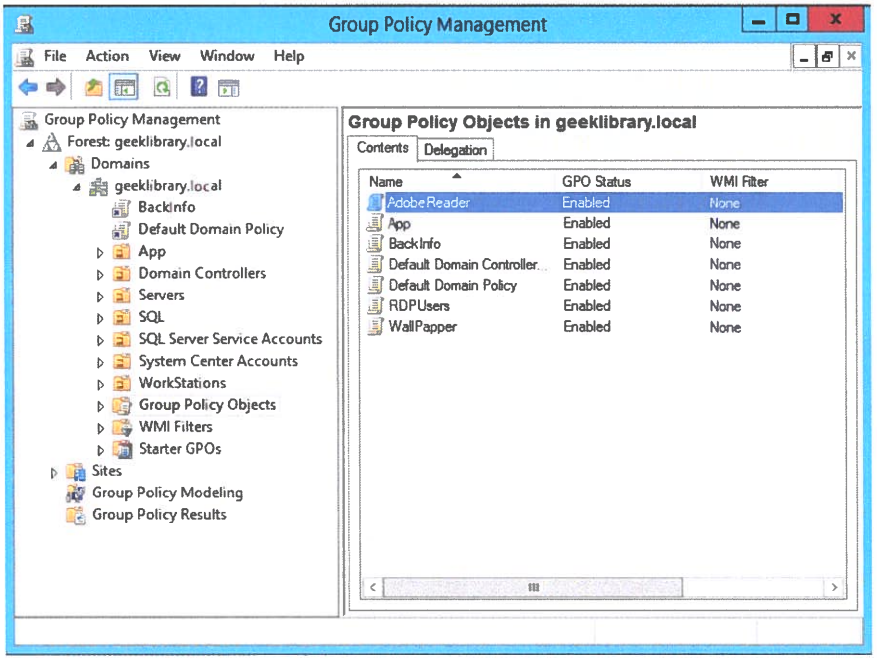

Figura 21 - Group Policy Management

Foram criadas as seguintes GPOs:

A GPO Adobe Reader vai instalar o *software* Adobe Reader, que serve para abrir os PDFs, nas máquinas clientes Windows 8.1.

A GPO App vai criar um atalho no ambiente de trabalho das máquinas clientes para executar a aplicação da biblioteca GeekLibrary.

A GPO BackInfo irá apresentar no ambiente de trabalho uma série de informações úteis sobre o computador, sempre que um dos utilizadores que pertence ao grupo de segurança do domínio "BackInfo" realizar login numa máquina.

A GPO RDPUsers irá colocar no grupo local das máquinas o grupo de segurança do domínio "RDPUsers" permitindo que os utilizadores que façam parte deste grupo possam aceder remotamente às máquinas. Sendo que as máquinas cliente são criadas no SCVMM, os utilizadores comuns não têm permissão para efetuar *login* como se estivessem localmente no computador, daí ser dado acesso remoto via LAN, que num caso real de produção poderia ser para efetuar acesso através de um *Thin client*[9].

A GPO WallPaper divide-se em duas partes, uma para colocar uma imagem como *background* do ambiente de trabalho, definida em "User Configuration" e outra que irá colocar uma imagem na janela de *login* e de *lock* definida em "Computer Configurations".

Tabela 5 - OUs e GPOs

OUs	GPOs
App	WallPaper
WorkStation	AdobeReader App RDPUsers WallPaper

[9] O Thinclient é um equipamento que não possui, na sua estrutura interna, (HD, Processador, Memória), e utiliza os recursos de um Servidor, através de *Terminal Service* (Ex: *Remote Desktop*)

4.3.5. DNS Server

O serviço DNS fornece a resolução de nomes para redes baseadas em TCP/IP, permitindo que os computadores cliente utilizem nomes, em vez de endereços IP numéricos, para identificarem outros recursos na rede.

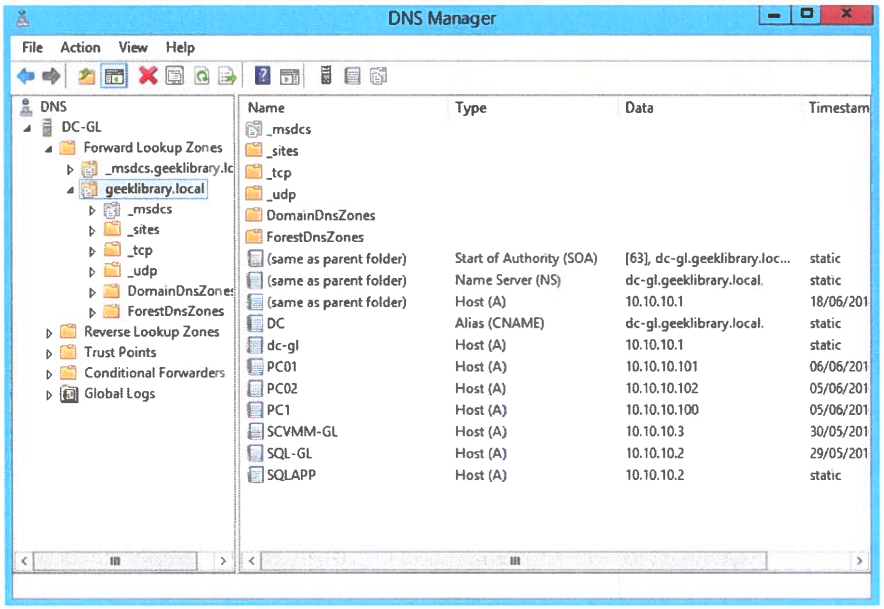

Figura 22 - DNS console

Aqui há poucas configurações a fazer, sendo que o serviço DNS será utilizado exclusivamente para a rede interna LAN. Apenas foi criado um novo registo (DNS record) para o IP "10.10.10.2" com o nome de "SQLAPP", ficando assim com o FQDN (Fully Qualified Domain Name) em "sqlapp.geeklibrary.local". Este registo é considerado um registo estático e servirá para a *connection string* da aplicação que não suporta caracteres especiais como tem o nome "SQL-GL" do servidor SQL.

4.3.6. DHCP Server

Quando se fala em redes, existem alguns recursos utilizados que facilitam muito a nossa vida, mas nem os percebemos. Um deles é o protocolo DHCP. Em inglês Dynamic Host Configuration Protocol (que significa algo como Protocolo de Configuração Dinâmica de Endereços de Rede), é um protocolo utilizado em redes de computadores e que permite às máquinas obterem um endereço IP automaticamente.

A *role* DHCP tem a função de gerir endereços IP e informações relacionadas de forma centralizada. Isto permite configurar definições da rede de cliente num servidor, em vez de configurá-las em cada computador cliente.

Resumidamente, utilizando um modelo cliente-servidor, o DHCP faz o seguinte:

- Quando um cliente se conecta a uma rede ele envia um pacote com um pedido de configurações DHCP.
- O servidor DHCP gere um intervalo de IPs disponíveis juntamente com as informações e parâmetros necessários (*default gateway*, nome de domínio, DNS, etc).
- Quando este servidor recebe um pedido, ele entrega um dos endereços disponíveis e configurações para o cliente.

Para realizar a instalação da role DHCP ir a "Add Roles and features".

Clique em "Next" até chegar a janela das *Roles* e selecionar a *role* "DHCP Server"

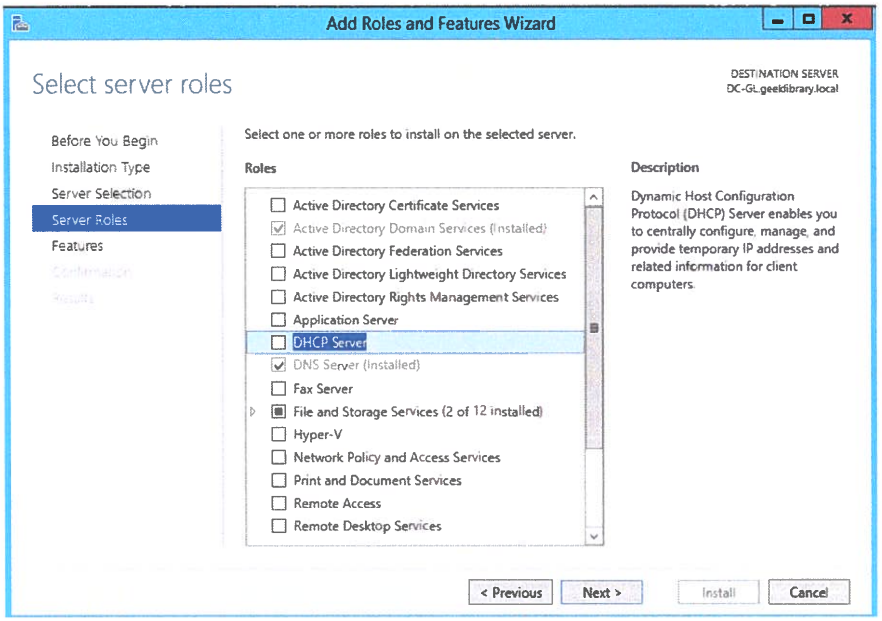

Figura 23 - Server Role DHCP

Aparece uma nova janela com as *features* necessárias a este serviço, clique em "*Add Features*"

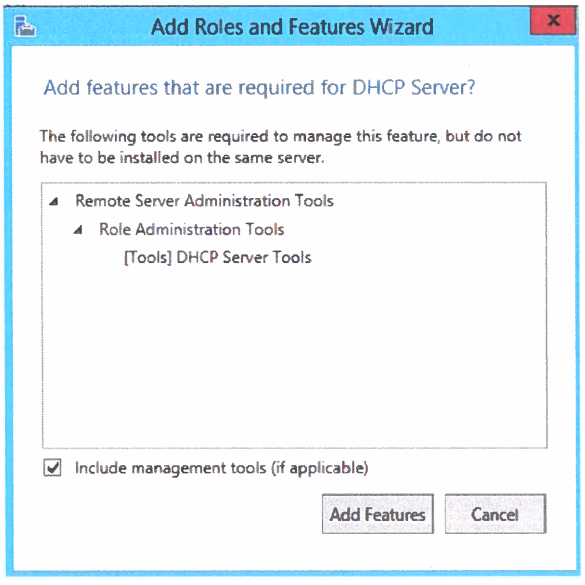

Figura 24 - DHCP required features

Clique em "Next" até à última janela e depois em "Install".

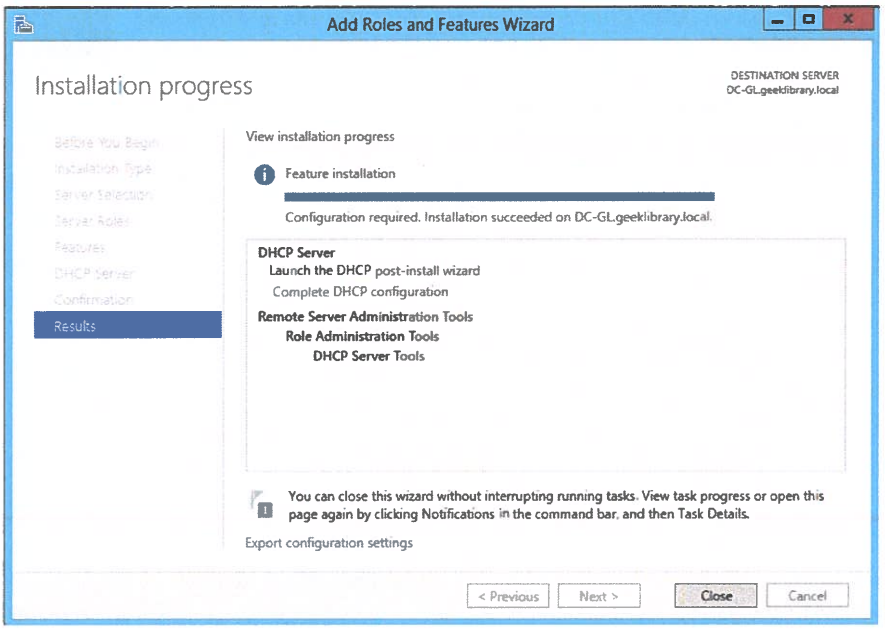

Figura 25- DHCP Install

Após a instalação necessário terminar as configurações. Clique em "Complete DHCP configuration"

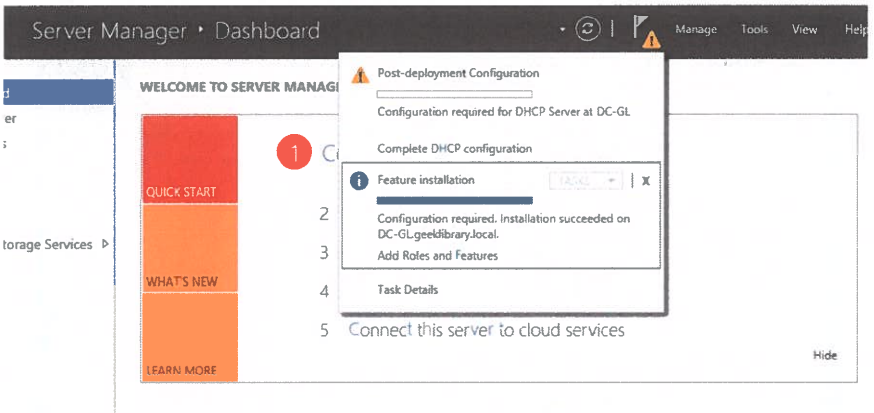

Figura 26- DHCP complete configuration

O Nome do *Scope* serve para distinguir mais facilmente no caso de haver multiplos *scopes* com diversos intervalos de rede (ex: impressora, wireless, PCs clientes). Neste caso vamos apenas utilizar um único *scope*.

Figura 27 - DHCP scope name

Na caixa do intervalo de IP coloca o IP de início e de fim que se pretende atribuir e a máscara de rede. O primeiro IP a ser atribuído é o 10.10.10.100 e o ultimo 10.10.10.150 da *subnet* 10.10.10.0 /24 que é que está a ser utilizada neste projeto.

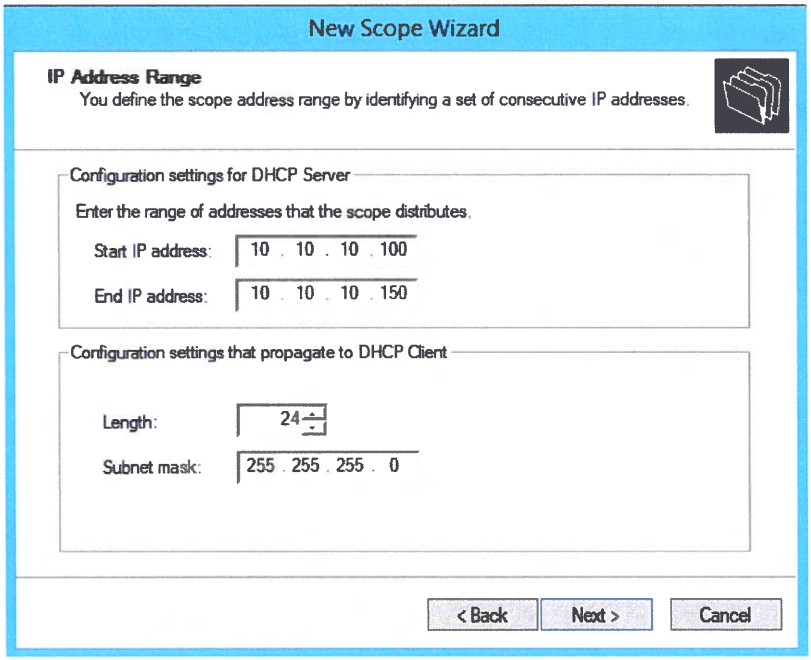

Figura 28 - DHCP scope range

Realizar "Next" nas janelas seguintes até à configuração do servidor DNS e validar que as configurações que aparecem automaticamente estão corretas.

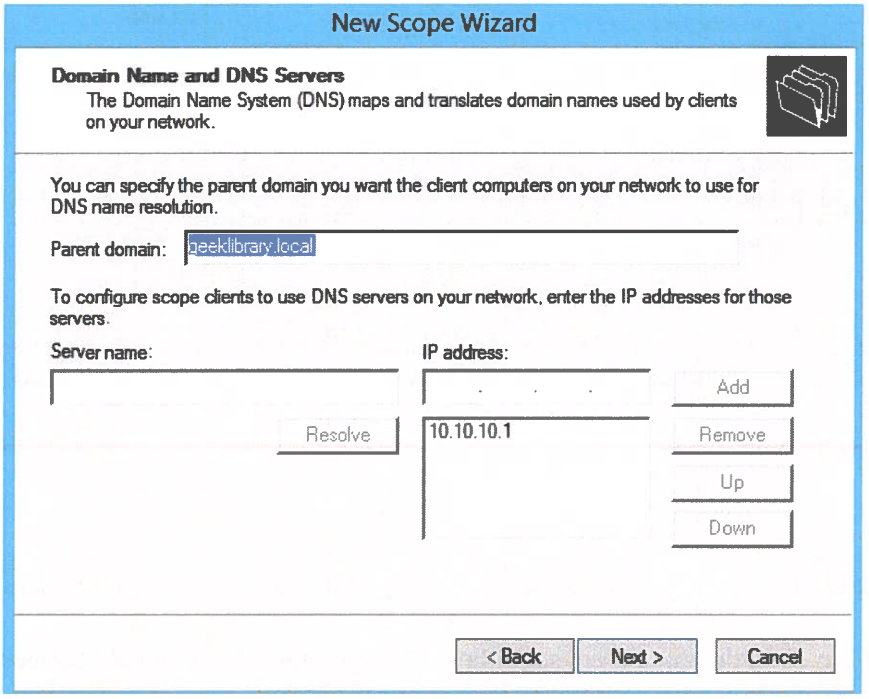

Figura 29 - DHCP configuration DNS Server

Realizar "Next" nas seguintes janelas e ativar o *scope* na última.

Deste modo ficará as configurações com o visto a verde e significa que a partir deste momento o serviço está pronto a fornecer configurações de rede (IPs, DNS, Gateway) aos computadores clientes ou até servidores que tiverem as suas interfaces de rede configuradas como DHCP.

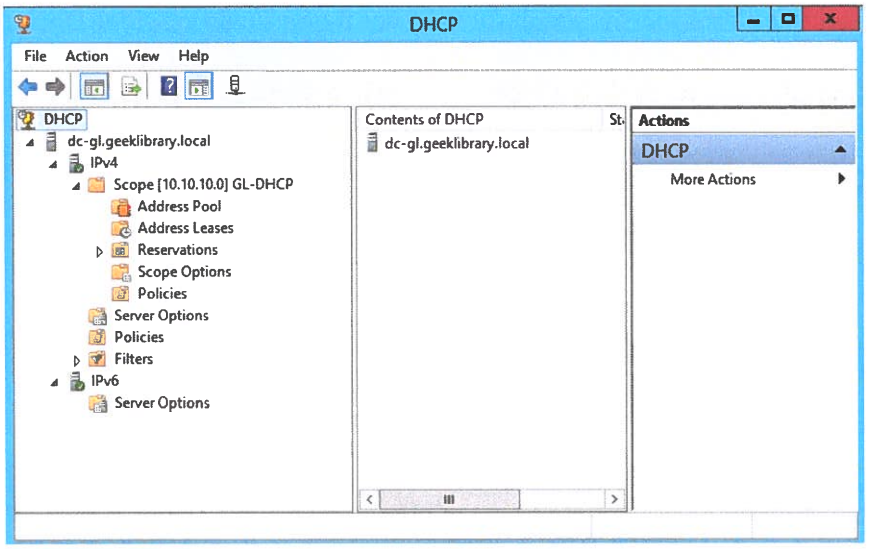

Figura 30 - DHCP console

É possível verificar as configurações e editar caso seja necessário, fazendo clique com botão direito em cima de *scope* e em seguida em "Properties". Nesta janela também é possível verificar o tempo, 8 dias, que um IP fica atribuído a um cliente até ser renovado.

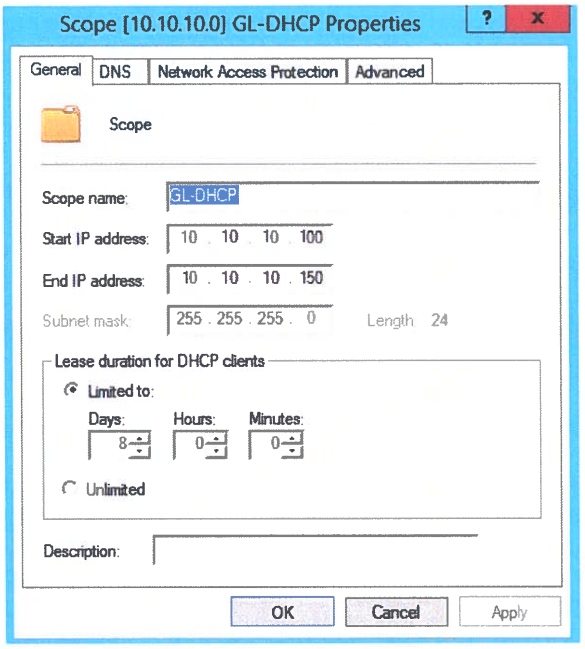

Figura 31- DHCP properties

4.3.7. File Share

A disponibilização de recursos aos clientes é provavelmente a tarefa mais comum nas implementações baseadas em Windows.

No Servidor DC-GL existe uma pasta partilhada com o nome "Share" que disponibiliza diversos conteúdos que são utilizados pelas GPOs e pelos computadores clientes:

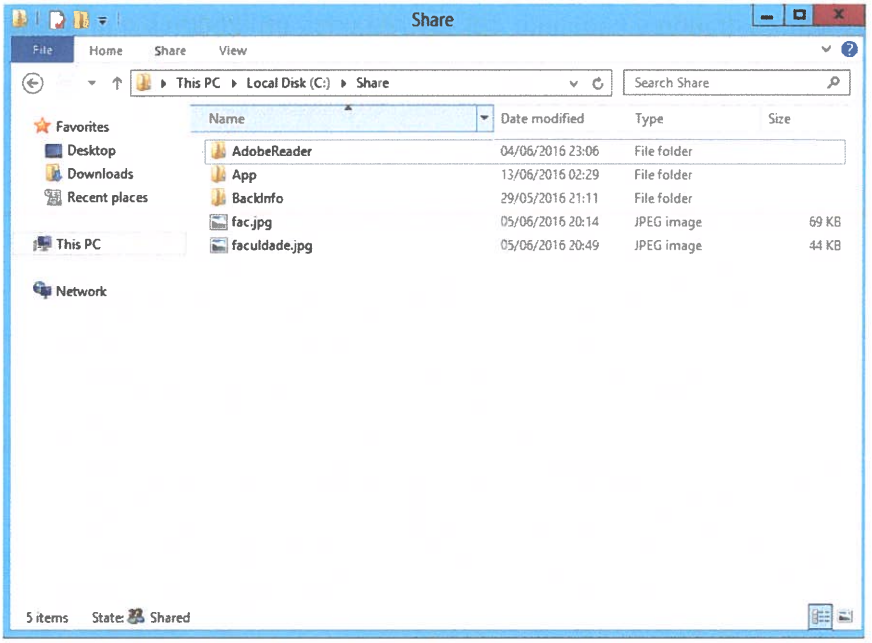

Figura 32 - DC File Share

A diretoria AdobeReader contém os ficheiros para a instalação via GPO do Adobe Reader nos computadores clientes.

A diretoria APP contem a aplicação da biblioteca. Os comutadores clientes não têm a aplicação instalada, mas sim um atalho nos seus ambientes de trabalho que "aponta" para o executável da aplicação nesta pasta partilhada. Portanto qualquer correção, alteração ou atualização que seja realizada na aplicação, esta não tem de ser reinstalada em cada posto de trabalho cliente.

Na pasta BackInfo estão os executáveis que geram a informação de sistema no ambiente de trabalho.

As imagens fac.jpg e faculdade.jpg são as imagens que através de GPO ficam como imagem de ambiente de trabalho e de *login* nos computadores clientes.

Por questões de segurança as permissões de adicionar, editar e apagar estão apenas acessíveis aos administradores enquanto que os restantes utilizadores tem apenas acesso de leitura, e estão restringidos de efetuar qualquer alteração tantos nas diretorias como nos ficheiros.

4.4. Servidor SQL - SQL-GL

Uma base de dados é uma ferramenta concebida para a recolha e organização de informações, e este servidor foi criado para servir esse propósito.

Para realizar login neste servidor utilizar a seguinte conta de domínio:

User: geeklibrary\sqladmin

Pass: P@ssw0rd

4.4.1. Características do Servidor:

- Tipo: Servidor Virtual (VMware)
- Sistema Operativo: Windows Server 2012 R2 Datacenter
- Memória: 1 GB
- Hostname: SQL-GL
- Configurações de interface LAN:
  - 10.10.10.2 / 255.255.255.0
- Configuração de interface INTERNET:
  - DHCP
- Configuração de Storage:
  - Sistema (C:) - 60GB

- Conta local
  - user: administrator
  - Password: P@ssw0rd
- Serviços Principais
  - SQL Server

4.4.2. SQL Server Enterprise 2012

Executar o *setup* e selecionar a primeira opção "New SQL Server stand-alone installation …"

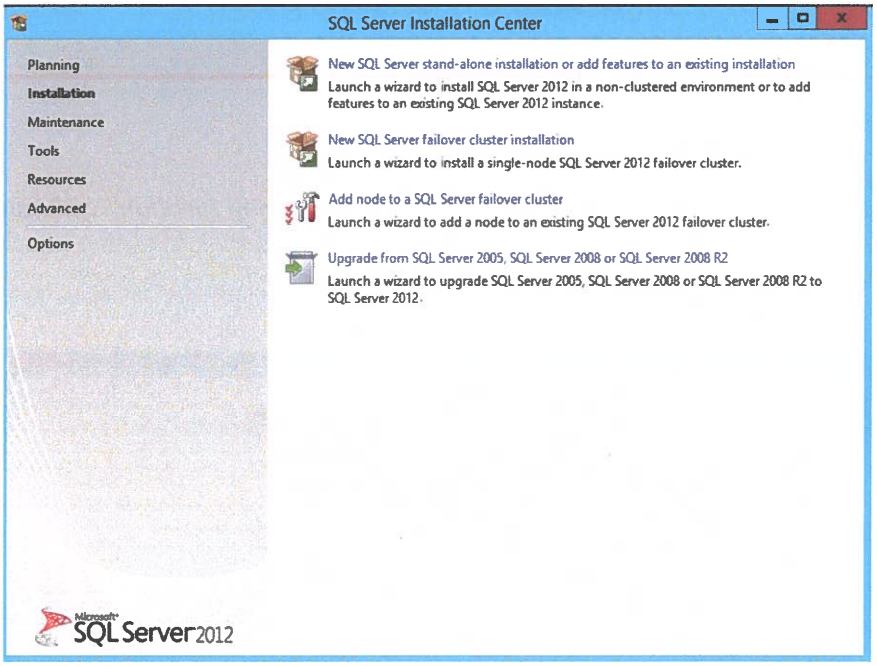

Figura 33 - SQL Installation setup

Em "Feature Selection" selecionar as funcionalidades Database engine Services, Client Tools Connectivity and Management Tools.

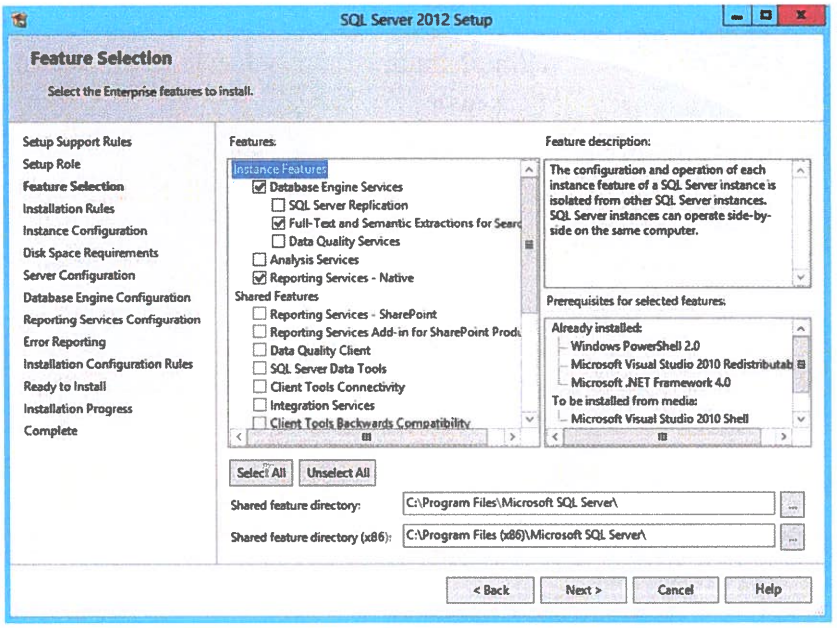

Figura 34 - SQL - Feature Selection

Fazer clique em "Next" até chegar á janela "*Server Configuration*" e alterar o "*Startup Type*" conforme imagem e clique em "Next".

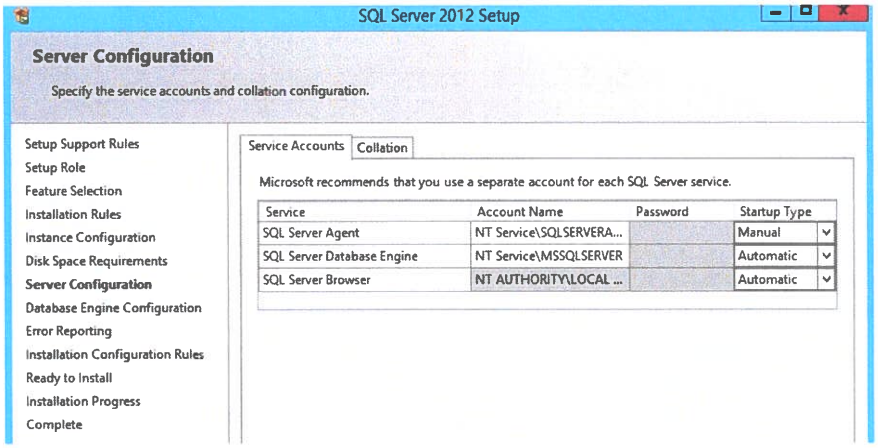

Figura 35 - SQL service configuration

Na janela seguinte selecionar o modo de autenticação em "Mixed mode" que permite realizar a autenticação através do SQL Server e do Windows. É neste ponto que se define a *password* para o utilizador local do SQL server "sa". De seguida clique em "Add Current User" e depois em "Add", e adicione os grupos de segurança de domínio "SQLAdmins" e "SCVMMAdmins".

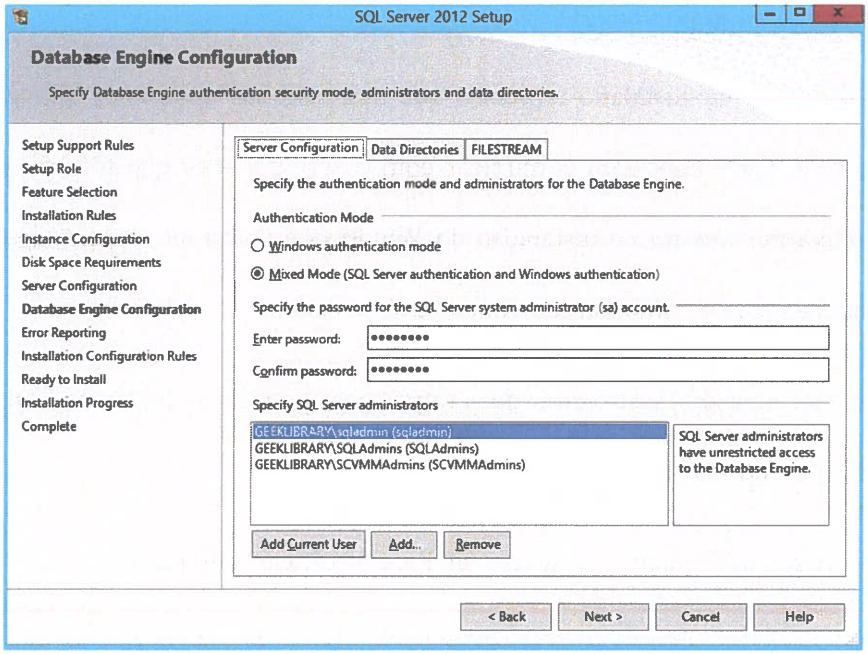

Figura 36 - SQL authentication mode

Clique em "Next" até à última janela e despois em "Install".

Fica assim instalado o SQL Enterprise 2012 com a instância *default* MSSQLSERVER bem com as permissões necessárias. É nesta instância que vão ser criadas todas as base de dados utilizadas neste projeto, tanto para aplicação da biblioteca como para o System Center Virtual Machine Manager.

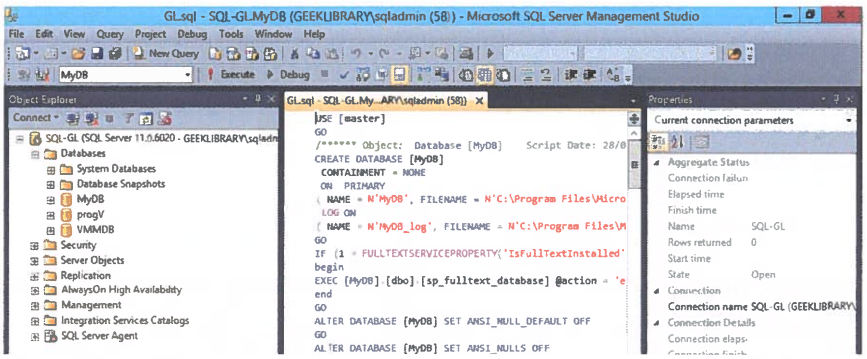

Figura 37 - SQL Management Studio

4.4.3. Firewall SQL

Após o SQL estar instalado é preciso que o acesso das aplicações através de outros servidores na rede LAN consigam comunicar com o servidor. Por questões de segurança a *firewall* está ativa por *default* na instalação do Windows e como tal contém algumas portas fechadas essenciais para a comunicação com a base de dados.

Na *firewall* será necessário criar duas regras para permitir o acesso ao SQL, uma com o protocolo TCP e outra UDP.

Criar um nova regra, selecionar o protocolo TCP e especificar o número da porta "1433".

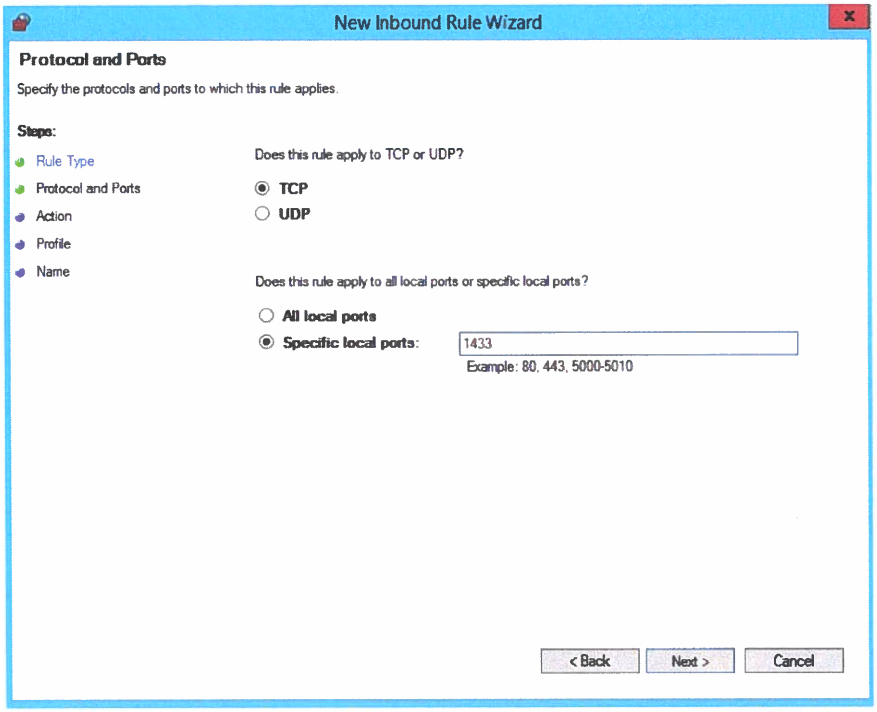

Figura 38 - SQL firewall port

Para permitir a ligação selecionar "Allow the connection"

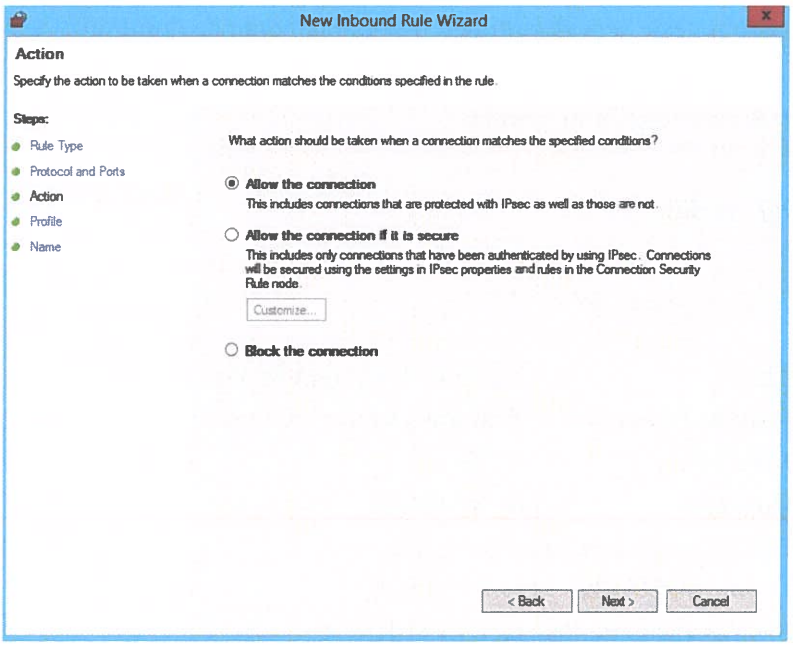

Figura 39 - SQL firewall allow

Especificar o nome da regra criada "SQL-TCP".

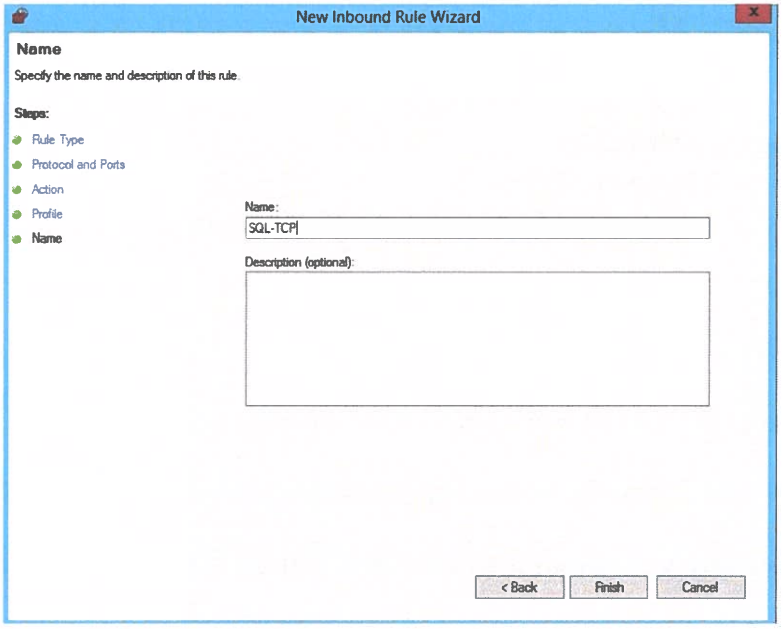

Figura 40 - SQL firewall name

Realizar os mesmos passos para criar a regra mas para o protocolo UDP, onde a porta é a "1434".

4.5. Servidor SC Virtual Machine Manager - SCVMM-GL

Para realizar login neste servidor utilizar a seguinte conta de domínio:

User: geeklibrary\scvmm_admin

Pass: P@ssw0rd

4.5.1. Características do Servidor:

- Tipo: Servidor Virtual (VMware)
- Sistema Operativo: Windows Server 2012 R2 Datacenter
- Memória: 4 GB
- Hostname: SCVMM-GL
- Configurações de interface LAN:
  - 10.10.10.3 / 255.255.255.0
- Configuração de interface INTERNET:
  - DHCP
- Configuração de Storage:
  - Sistema (C:) – 60GB
  - Hyper-V (H) – 100GB
- Conta local
  - user: administrator
  - Password: P@sww0rd
- Serviços Principais
  - Hyper-V Server
  - System Center Virtual Machine Manager

4.5.2. Hyper-V

A instalação da *role* Hyper-V (*hypervisor* da Microsoft) vai permitir criar máquinas virtuais neste servidor. Contudo, e como este servidor é já uma máquina virtual e que a partir deste momento passará a ser possível ter máquinas virtuais, estamos perante uma situação de *nested virtualization* (um *hypervisor* dentro de outro *hypervisor*). Esta situação só é possível realizar caso tenham sido realizadas as configurações descritas para este servidor (SCVMM-SQL) no ponto VMware.

Para instalar a role de Hyper-V, clique "Add roles and features", fazer "Next" até à Janela de "Server Roles" e selecionar a *role* "Hyper-V".

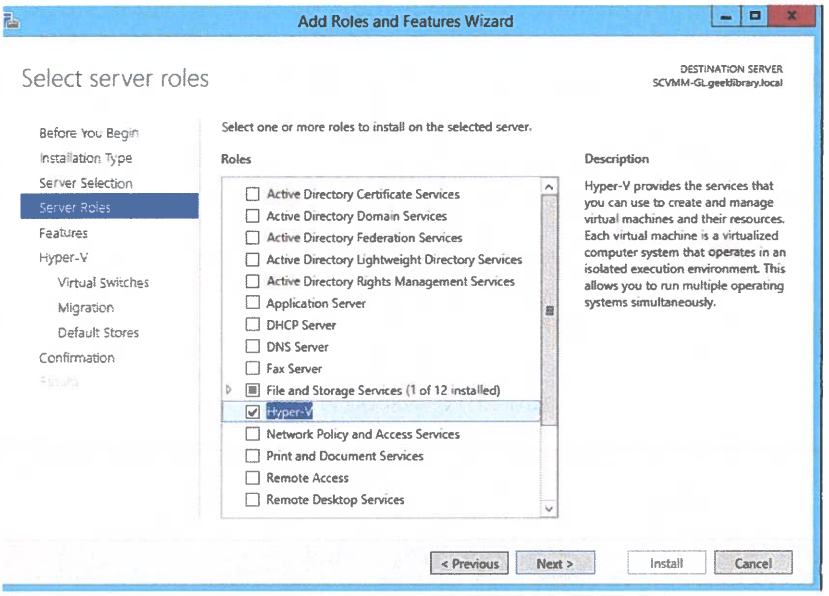

Figura Figura 41 - Server Roles Hyper-V

Na janela seguinte selecionar onde é criado o *virtual switch* associado a uma placa de rede. A placa de rede selecionada é "Prod" que pertence à rede interna LAN. Mais tarde será configurado um novo *virtual switch* para acesso à internet.

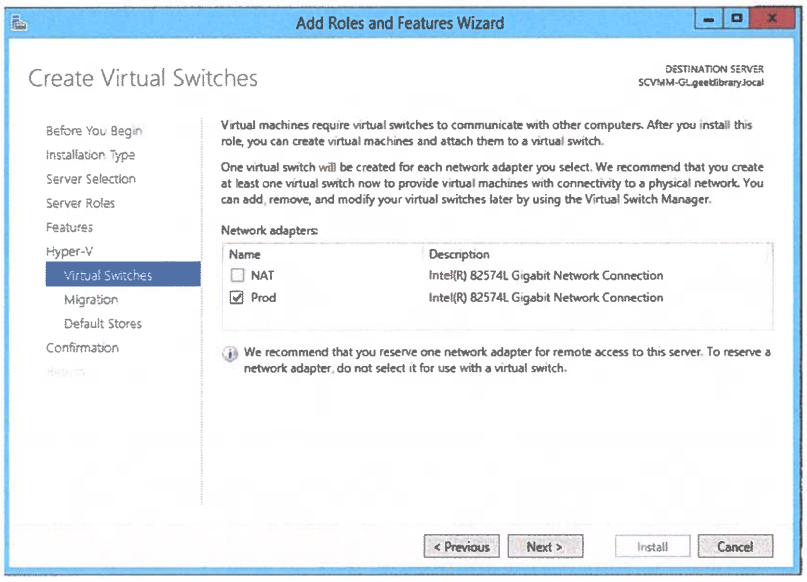

Figura 42- Hyper-V Role Virtual Switch

Fazer "Next" na janela seguinte, pois como se trata de uma instalação *standalone* (um único servidor com Hyper-V) não existirá qualquer configuração de migração de maquinas virtuais, que existe em implementações Cluster (dois ou mais servidores com Hyper-V com possibilidade para mover máquinas virtuais entre os mesmos).

Na janela de Default Storage selecionar as diretorias que servem de repositório para as máquinas virtuais e para os seus discos virtuais. Neste caso foi criado à parte do disco do sistema operativo (C: com 60 GB) um novo disco (H: com 100GB), exclusivamente para o arquivo das máquinas a serem criadas.

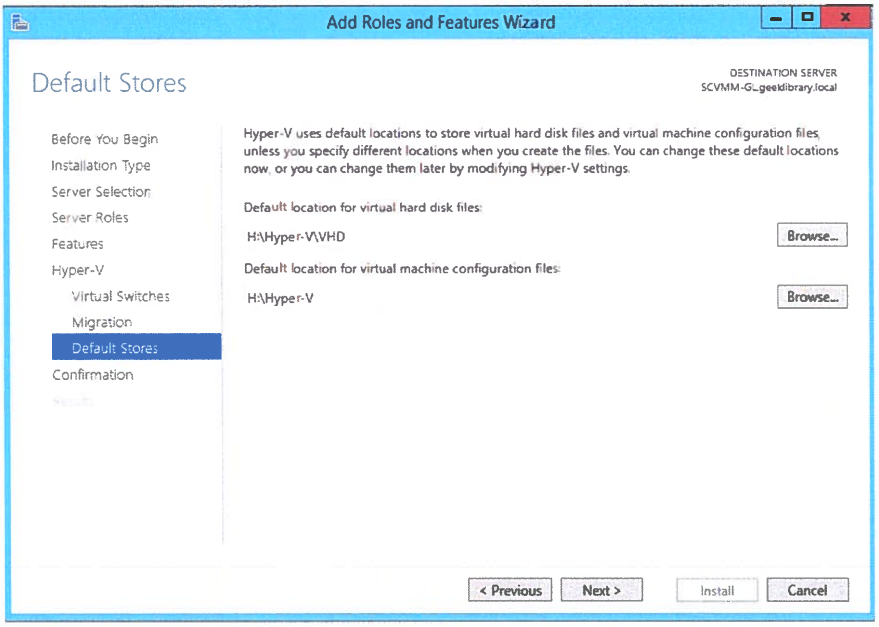

Figura 43- Hyper-V default stores

Fazer clique em "Install".

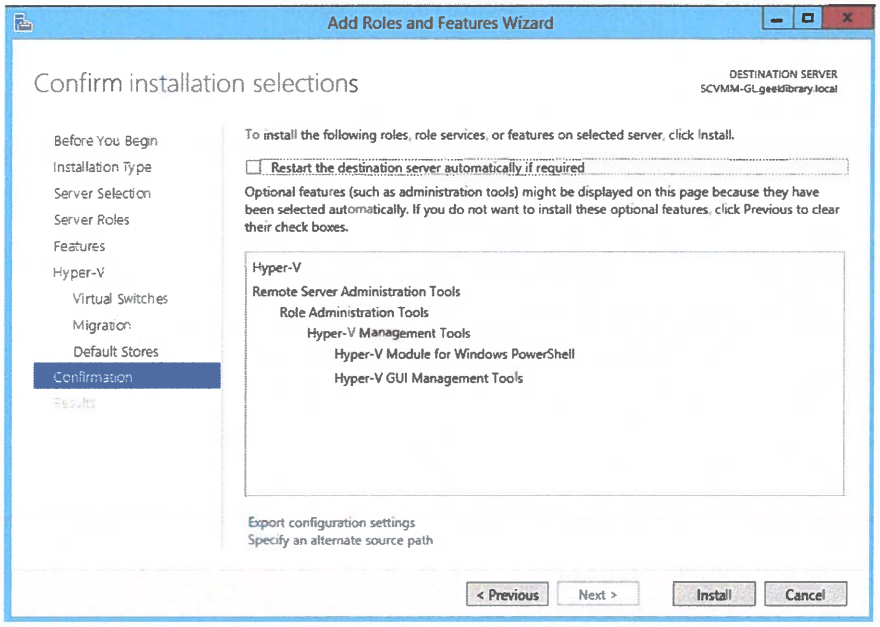

Figura 44 - Hyper-V confimation role

No final é necessário reiniciar o servidor de modo a finalizar a instalação da respetiva *role*.

Após o *restart* do servidor e validar que a instalação do Hyper-V foi realizada com sucesso, segue-se a instalação do SCVMM.

4.5.3. System Center Virtual Machine Manager

Contudo e antes de se executar a instalação principal é necessário instalar uns pré requisitos, pela seguinte ordem:

- ADK
- Sqlncli
- SqlCmdLnUtils

Na instalação do ADK (Windows Assessment and Deployment Kit (ADK) for Windows 8.1.) selecionar as opções "Deployment Tools" and "Windows Preinstallation Environment".

Executar sqlncli.exe e de seguida SqlCmdLnUtils.exe

Executar o *setup* do VMM

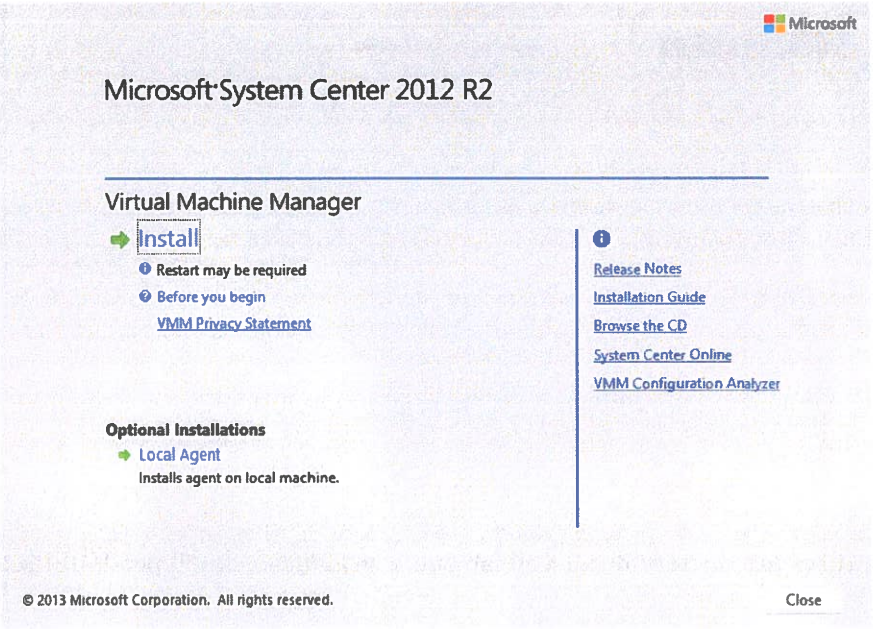

Figura 45- Virtual Machine Manager setup

Selecionar a caixa "VMM management server" e validar que a caixa "VMM console" fica também selecionada e clique em "Next".

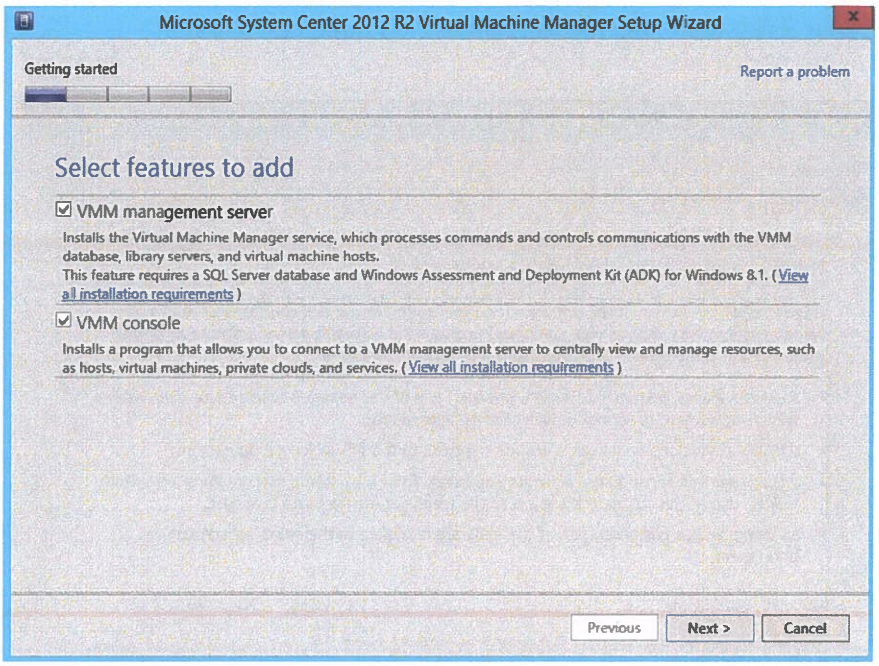

Figura 46 - VMM setup

Nesta janela indicar o nome, organização e colocar a chave de produto.

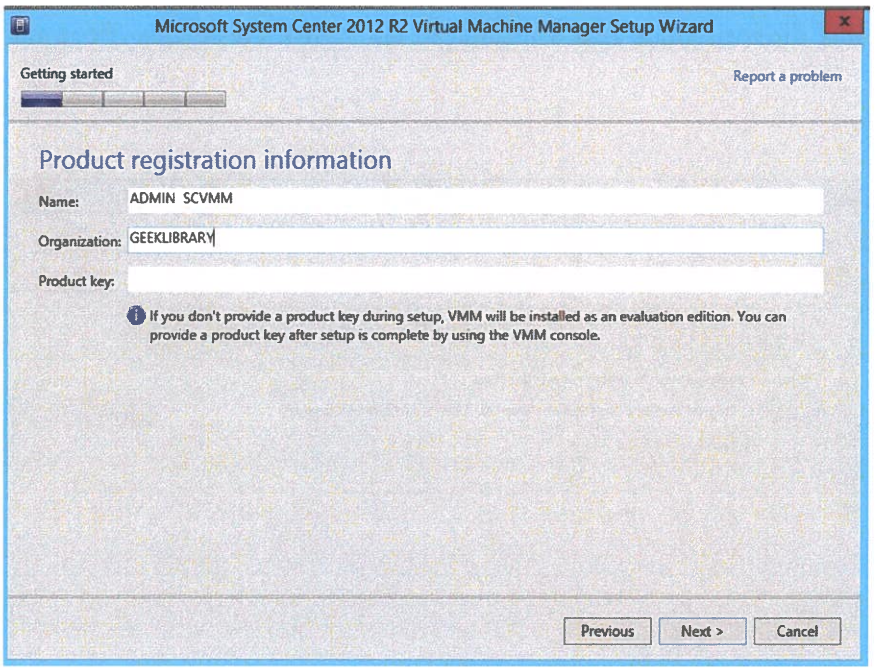

Figura 47 - VMM registration

Selecionar a caixa do "license agreement" e clique em "Next".

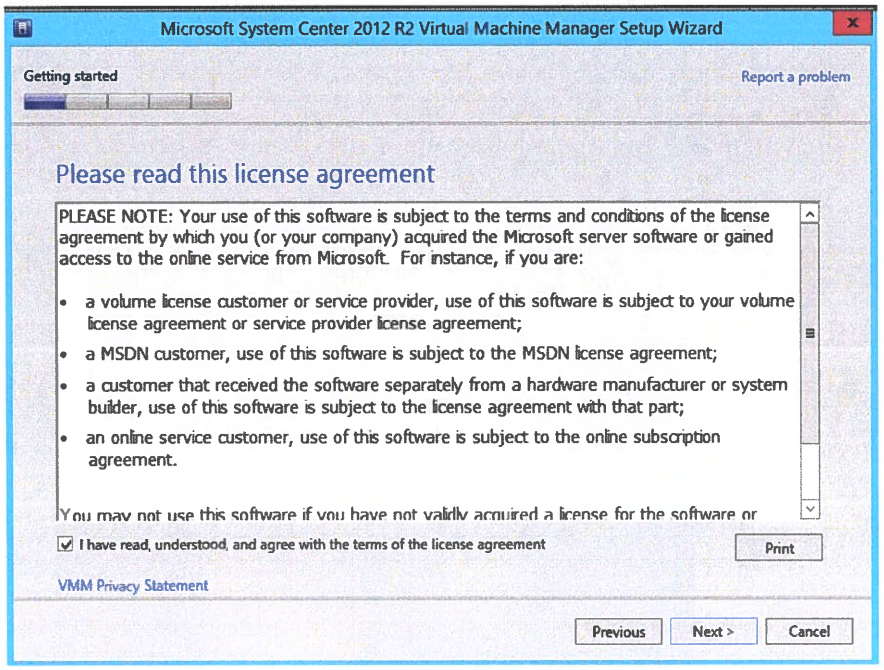

Figura 48 - VMM license agreement

Indicar a diretoria onde fica a instalação do VMM. Neste caso ficou na localização *default*.

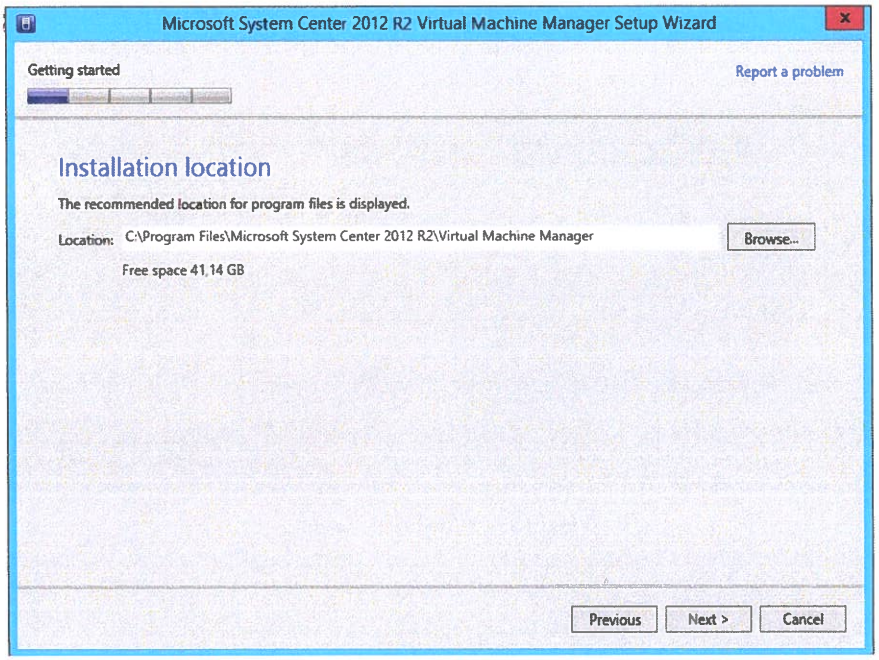

Figura 49 - VMM installation location

Na janela da configuração da bases de dados preencher o nome do servidor, as credenciais do utilizador "sql_svc" e nome da base de dados que se pretende criar "VMMDB".

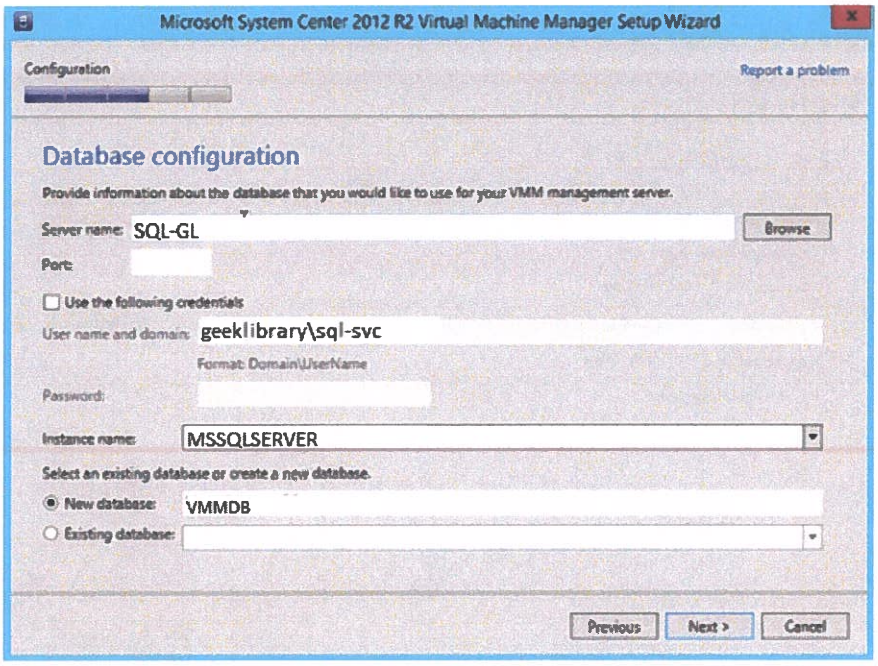

Figura 50 - VMM Database configuration

Para a conta de serviço do SCVMM colocar o utilizador de domínio "scvvm_admin".

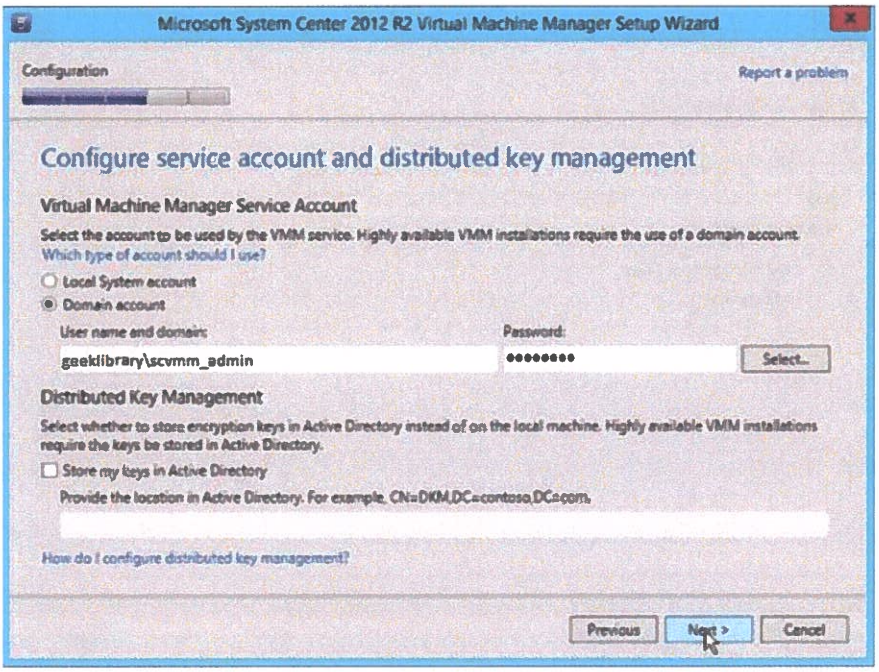

Figura 51 - VMM service account

Na janela do "Library Configurarion" é onde se define o repositório dos conteúdos que vão ser adicionados ao Library do VMM. Neste caso foi deixado na diretoria *default*.

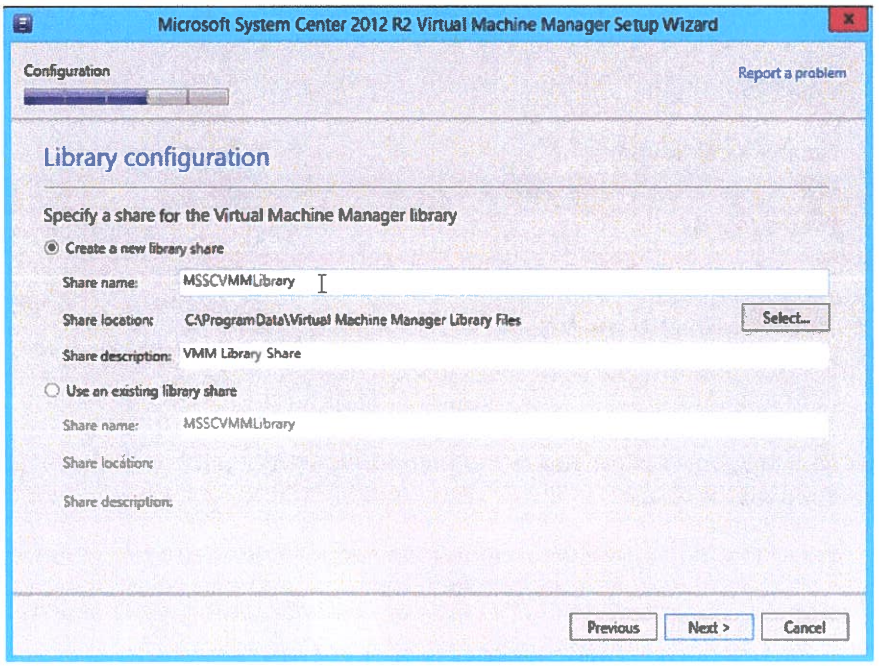

Figura 52 - VMM Library location

Clique em "Next" até ao fim e depois em "Install".

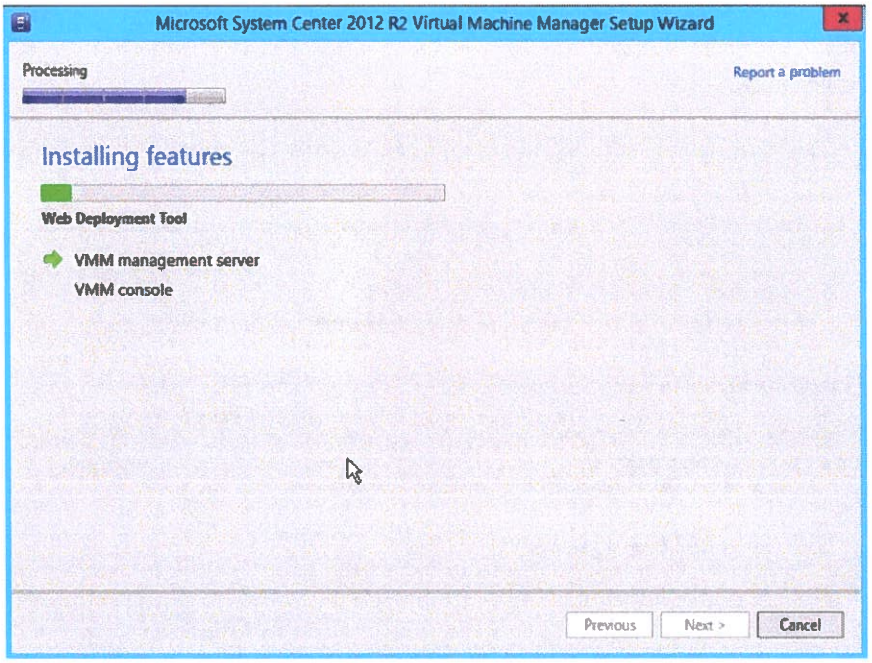

Figura 53 - VMM installing

Após s instalação irá aparecer um ícone do Virtual Machine Manager no desktop. Carregar e conectar ao servidor em que o nome do servidor é *localhost* (ele próprio) na porta 8100.

Figura 54 – SCVMM Connect to Server

Na consola do SCVMM no menu Fabric é onde a maioria das configurações são feitas.

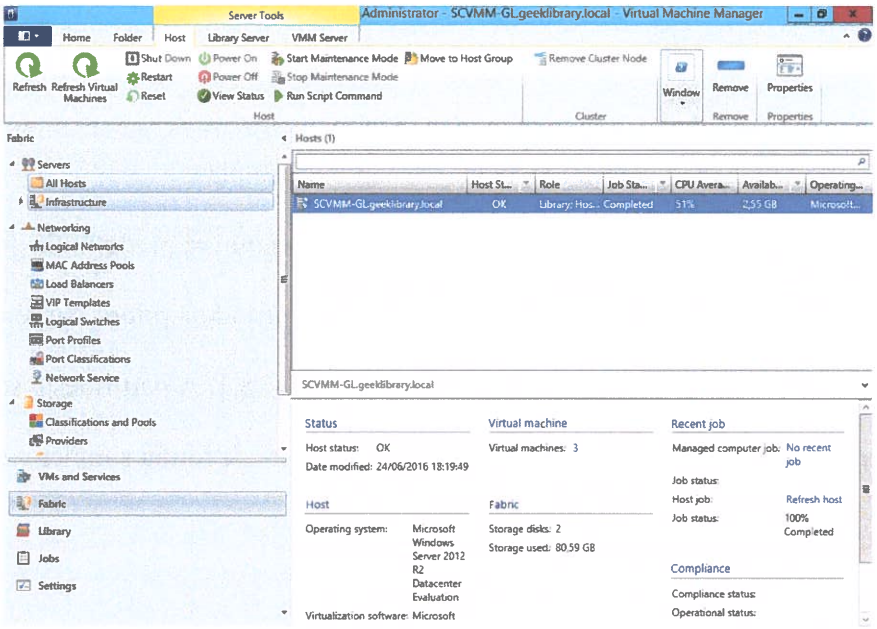

Figura 55 – SCVMM console

É necessário associar o *host* de Hyper-V ao SCVMM pois é no servidor com o *hypervisor* (SCVMM-GL) que as máquinas virtuais podem ser executadas. O SCVMM é apenas a consola que irá gerir as máquinas e os recursos das mesmas. Para associar o servidor, clicar no botão direito em All Hosts e selecionar "Add Hyper-V Hosts and Clusters " adicionando o servidor SCVMM-GL com a conta Run As Account do utilizador scvmm_runas.

4.5.4. System Center Virtual Machine Manager – Network

As máquinas virtuais criadas no VMM para terem acesso á rede LAN onde se encontram os servidores de domínio e base de dados, e para ter acesso à Internet, as suas placas de rede têm de estar de certa forma associadas às placas de rede do servidor SCVMM-GL. No VMM foi criado duas "Logical Networks" e associadas ás duas placas de rede do *host* mantendo os mesmo nomes. Na prática o que vai ser feito é um *virtual switch* ligado diretamente ás placas de rede do servidor SCVMM-GL através de uma *bridge*. No VMM foi criado duas VM Networks associadas, cada uma delas às Logical Networks. É às VM networks que as placas de rede de cada máquina virtual cliente se ligam.

4.5.5. System Center Virtual Machine Manager – Template

Um *template* é um modelo a ser seguido, com uma estrutura predefinida que facilita o desenvolvimento e criação do conteúdo a partir de algo construído à priori. Neste caso vai ser criado um *template* de uma máquina virtual cliente Windows 8.1. A partir deste será possível gerar inúmeras máquinas idênticas sempre que for necessário, de forma simples e rápida.

Antes da criação do *template* foi criada uma máquina virtual que vai servir de modelo para este. Para criar uma nova máquina virtual na consola do VMM, na janela VMs and Service, na barra de tarefas, selecionar Create Virtual Machine. A máquina virtual contém a seguinte configuração de *hardware*: 1 CPU (processador); 1 disco de 40GB; 2 placas de rede (uma ligada á VM Network LAN e a segunda à VM Network Internet); memória RAM configurada como memória dinâmica com valor mínimo de 512MB e máximo de 1GB, e o valor de arranque da máquina 512MB.

A memória dinâmica é útil pois permite às VMs consumir memória dinamicamente, com base na carga de trabalho atual, isto é, que consuma a memória física do servidor Hyper-V à medida que a for requisitando. Permitindo assim que a memória disponível possa ser utilizada para outros processos ou VMs.

Com a máquina virtual criada procedeu-se à instalação do Sistema Operativo Windows 8.1 Professional. Não é necessário configurar nenhum IP nas placas de rede, pois estas funcionam por DHCP. Foram feitos alguns testes para validar as GPOs e as aplicações.

Tendo a instalação sido realizada com sucesso e com a máquina desligada, na consola do VMM fazer botão direito em cima da VM e selecionar Create VM Template. A criação do *template* destrói a máquina tal como está atualmente, isto é, a máquina é convertida para *template* e deixa de estar disponível como VM. Para se manter a máquina e criar um *template* da mesma o melhor será realizar um clone[10]. O clone irá duplicar a máquina principal e depois fazer o *template* a partir do clone. O nome do *template* ficou "Windows 8.1 – Template" e nas configurações do hardware manteve-se as da máquina original. Na parte do OS Configuration foram adicionados alguns campos, esta parte é muito importante pois é onde se encontram

[10] Replicação igual à original. Uma cópia exata um do outro.

configurações como o licenciamento e associação da máquina ao domínio. Dentro desta janela em Operating system selecionar "64-bit edition of Windows 8.1", em Computer name colocar PC##. Durante a criação de uma máquina virtual através do *template*, o ## será substituído pelo número sequencial (Ex: 01, 02), ficando assim as respetivas máquinas com os nomes e *hostnames* com PC01, PC02, etc.

Em "Admin Password" foi colocado o nome "admin" que será um administrador local para todas as máquinas e no "Product Key" colocada licença do Windows.

Em Domain/Workgroup selecionar domínio e colocar "geeklibrary.local" com as respetivas credenciais e assim deste modo quando uma nova máquina for criada fica automaticamente associada ao domínio, podendo um utilizador de domínio fazer logo login.

Deste modo usando o *template* para criar uma nova máquina virtual, para além de ficar pronta num curto espaço de tempo, encontra-se com as configurações de domínio e licenciamento concluídas.

O *template* será guardado dentro da Library do VMM. Para visualizá-lo basta na consola do VMM ir a Library e depois VM Templates.

Para criar uma nova máquina virtual através do *template*, na consola do VMM ir a VMs and services e na barra de tarefas carregar em Create Virtual Machine. Dentro dessa janela selecionar "Use na existing virtual machine, VM Template, or virtual hard disk", fazer "Browse..." e selecionar o "Windows 8.1 – Template".

4.5.6. Maquina Virtual cliente Windows 8.1

O acesso ao computador cliente Windows 8.1 pode ser realizado através da consola do SCVMM, mas também é possível aos utilizadores acederem via *Remote Desktop (RDP)*.

Ambiente de trabalho o ícone da aplicação biblioteca "GeekLibrary", AdobeReader que permite visualizar os documentos (PDFs) que o utilizador pode fazer *download* da biblioteca e o ícone da OneDrive onde o utilizador pode guardar os respetivos documentos na *cloud* da Microsoft.

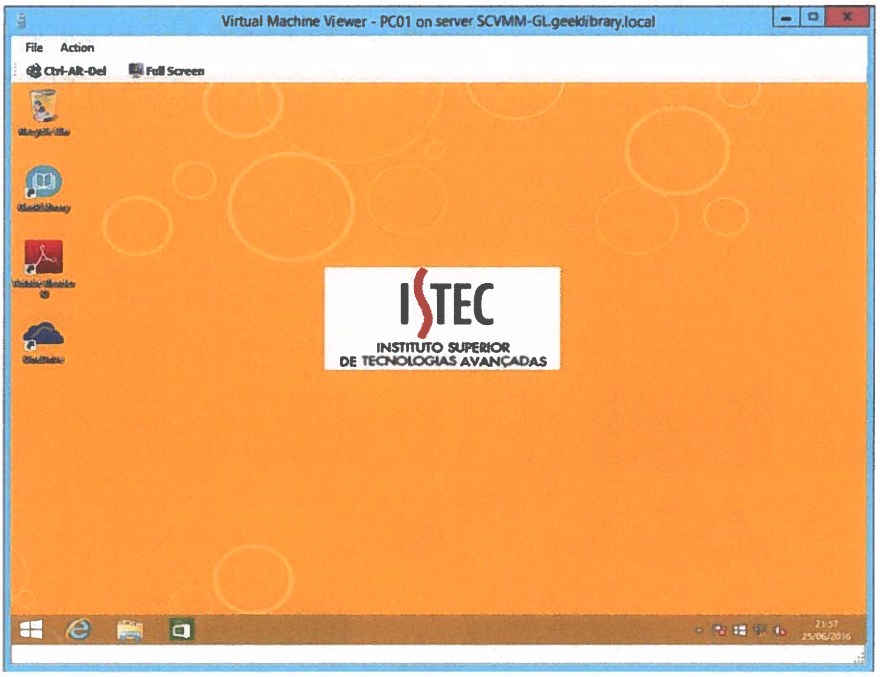

Figura 56 - VM cliente Windows 8.1

Este ambiente de trabalho é igual para todas a maquinas virtuais clientes (ex: PC01, PC02) criadas através do *template* no SCVMM.

5. Aplicação

A aplicação é uma biblioteca digital que tem como objetivo centralizar a informação, isto é, todos os conteúdos inseridos na biblioteca podem ser pesquisados e consultados. A esta aplicação deu-se o nome de GeekLibrary. Esta foi completamente desenvolvida na ferramenta Visual Studio 2015 em linguagem de programação C# e WPF (Windows Presentation Foundation) que é um subsistema gráfico da .NET com linguagem XAML[11].

Utilizando o modelo de três camadas, a aplicação contém uma interface com o utilizador, *layout* visual e interativo, uma camada de negócio, que estabelece a ligação entre o *layout* e o acesso á BD, e a camada *Data Access* que estabelece a ligação à própria BD.

5.1. Diagrama

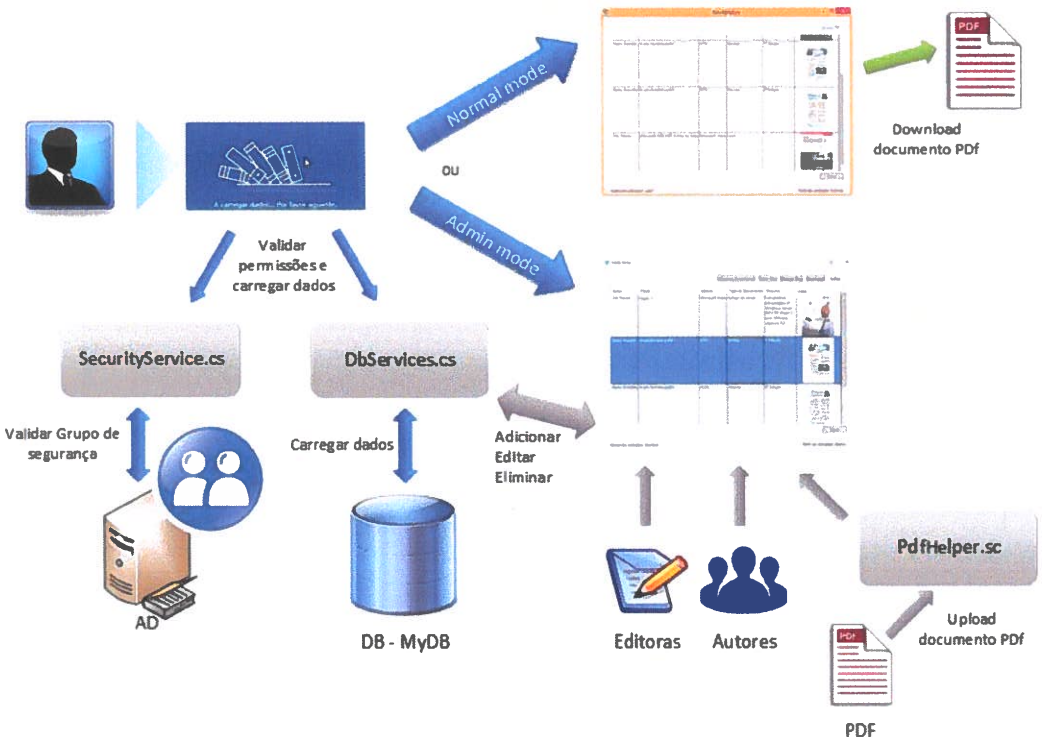

Figura 57 - Diagrama aplicacional

[11] XAML (*Extensible Markup Language*) é uma linguagem de marcação declarativa, utilizada para criar interfaces de utilizador de forma simples e rápida.

5.2. Base de Dados SQL

A base de dados é uma parte essencial para o funcionamento da aplicação, pois é aqui que vão ficar guardadas todas a informações como os nomes dos Autores, Editoras e o conteúdo dos documentos.

No servidor designado para esta tarefa (SQL-GL) foi criada a base de dados com o nome MyDB que será utilizada para conter todos os dados da aplicação biblioteca.

Apesar das permissões dos utilizadores serem definidas na programação da aplicação, se são administradores ou apenas consultores, por questões de segurança foram também colocados os grupos de domínio com as respetivas permissões de acesso à base de dados MyDB.

O grupo de segurança de domínio AppUsers foi colocado com apenas permissões de leitura (db_datareader) na base de dados.

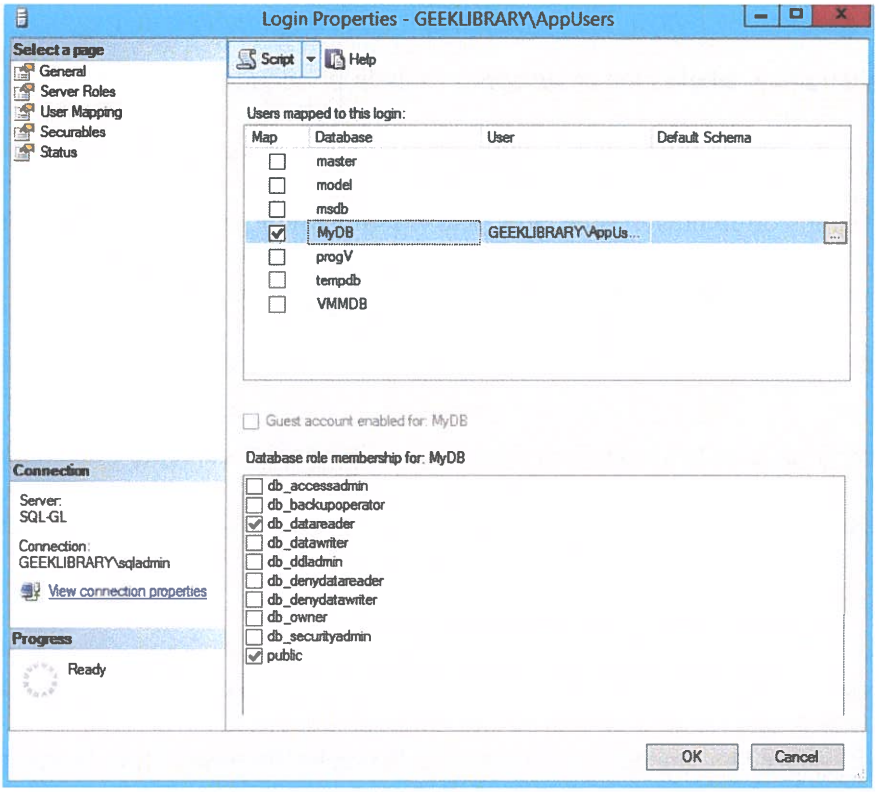

Figura 58 - DataBase User Mapping

Com a base de dados criada e as questões de segurança configuradas, foram cridas as tabelas e as relações entre si.

Foram criadas quatro tabelas *Document*, *Author*, *Publisher* e *DocumentType*.

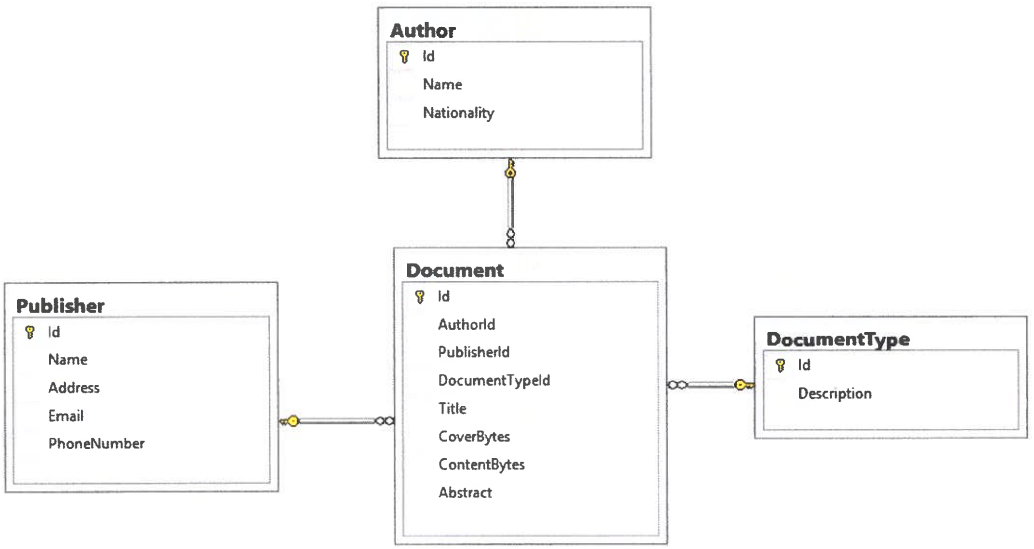

Figura 59 - SQL Diagram tables

Considera-se a tabela *Document* como a tabela principal em que o campo AutorId tem uma relação de um para muitos com o Id da tabela *Author*, pois é possível que possam haver vários documentos diferentes mas escritos pelo mesmo Autor. Assim e para que não se tenha de repetir o nome do Autor tanta vezes quanto os documentos adicionados que escreveu, o nome do Autor fica numa tabela à parte e é apenas criada uma relação entre tabelas.

O campo Id de cada tabela é um número gerado automaticamente e que nunca se repete de modo a não haver conflitos mesmo quando por exemplo existam dois livros diferentes do mesmo Autor.

Processo semelhante acontece para o campo PublisherId que tem uma relação com o Id da tabela *Publisher* e DocumentTypeId com o Id da tabela *DocumentType*. Desta forma não se

repetem informações poupando espaço na base de dados e melhorando a performance no seu acesso de escrita e leitura.

Na tabela *Document* existe ainda o campo ContentBytes que irá guardar o conteúdo dos documentos, e o CoverBytes que tem apenas a capa do documento guardado.

5.3. Código

No desenvolvimento, e principalmente para a criação das variáveis, utilizou-se o conceito de escrita CamelCase e que consiste em escrever palavras compostas ou frases, onde cada palavra é iniciada com maiúsculas e unidas sem espaços. Ex: MainWindow.

5.3.1. Interface gráfica

Na execução da aplicação a primeira janela é a "InitWindow", que contém uma imagem animada (ficheiro .gif) enquanto são executadas algumas tarefas em *background*, nomeadamente validar os dados do utilizador e carregar a base de dados.

Após a validação aparece a janela "MainWindow" que é janela principal da aplicação. Dependendo das permissões pode-se visualizar dois tipos de janelas:

Em modo administrador da aplicação, onde é possível, adicionar, editar e eliminar conteúdos.

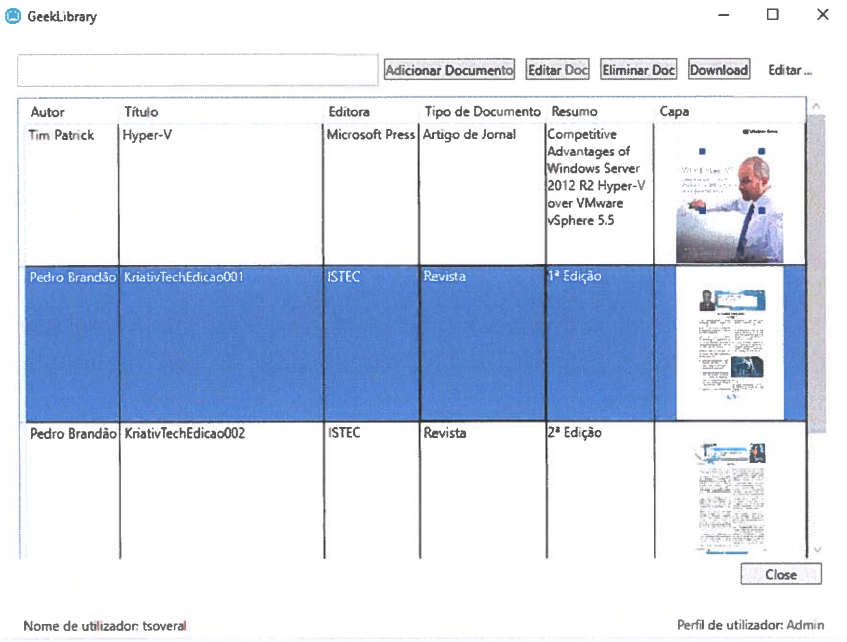

Figura 60 - GeekLibrary admin mode

Em modo utilizador normal onde é apenas possível consultar e fazer *download* dos conteúdos.

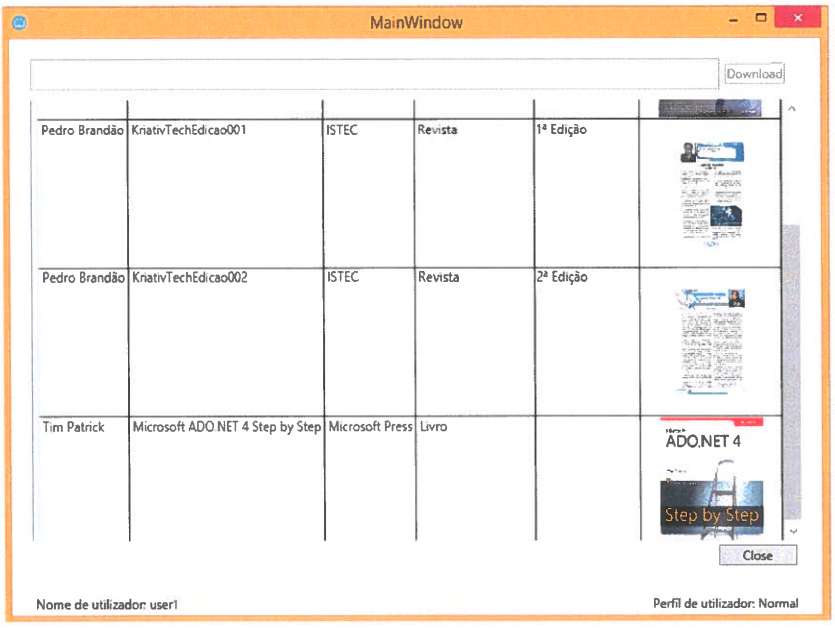

Figura 61 - GeekLibrary User mode

O tipo de perfil pode ser ainda verificado no final da janela da aplicação, onde do lado esquerdo está o nome do utilizador e do lado direito o perfil que o utilizador tem perante a aplicação (Admin ou Normal).

Um grande fator da aplicação é o facto de ser possivel pesquisar um documento. Escrevendo na caixa de texto que se encontra acima da tabela é possivel por exemplo, pesquisar por um autor onde aparecem todos os livros associados ao mesmo, ou pesquisar por uma editora e serem mostrados todos os documentos associados. Esta pesquisa é possivel pois o XAML tem acesso ao "DataContext" no C#, e este por sua vez implementa o componente "INotifyPropertyChanged", que faz com que seja possível definir o *binding*[12] na propriedade *text* da *textbox* a uma propriedade do C#, neste caso "*Searchquery*". Assim sempre que o texto for alterado na *textbox* é chamado o "*Set*" da propriedade e nesta altura é feita a pesquisa na BD e atualizados os dados na tabela. O valor *default* da caixa de texto é estar vazia e assim aparece toda a informação.

```
public partial class MainWindow : Window, INotifyPropertyChanged
{
    INotify

    private string searchQuery;
    2 references
    public string SearchQuery
    {
        get
        {
            return searchQuery;
        }

        set
        {
            searchQuery = value;
            NotifyPropertyChanged("SearchQuery");
            Refresh();
        }
    }
```

Figura 62 - Search Query, (fonte própria)

A tabela é criada no XAML utilizando uma DataGrid com as colunas Autor, Titulo, Editora, Tipo de Documento, Resumo e Capa. Estas colunas vão ser preenchidas, utilizando

[12] Permite ligar qualquer propriedade de um controlo a uma propriedade de um objeto, isto é, estabelece uma conexão entre a UI (interface de utilizador) da aplicação e a origem dos dados.

um *binding*, com tantos dados como os que existirem na tabela da base de dados. É na classe DbService que está a responsabilidade através de um "*Select*" de pesquisar os dados para importar para a tabela.

```
List<Document> result = new List<Document>();
DataTable dt = DbManager.Instance.GetDataTable(
                string.Format("select doc.* " +
                              "from Document doc " +
                                "inner join DocumentType dt " +
                                    "on dt.Id = doc.DocumentTypeId " +
                                "inner join Publisher pub " +
                                    "on pub.Id = doc.PublisherId " +
                                "inner join Author a " +
                                    "on a.Id = doc.AuthorId " +
                                "where doc.Title like '%{0}%' or " +
                                    "dt.Description like '%{0}%' or " +
                                    "pub.Name like '%{0}%' or " +
                                    "a.Name like '%{0}%'", search));
if (dt != null && dt.Rows.Count > 0)
{
    foreach (DataRow r in dt.Rows)
        result.Add(FillDocument(r));
}
return result;
```

Figura 63 - Select document, (fonte própria)

5.3.2. Autenticação na aplicação

A aplicação utiliza as informações dos utilizadores de domínio para saber quais as permissões que estes têm para poder utilizá-la. Utilizando as credencias já colocadas aquando o utilizador efetua o *login* na máquina, não é necessária qualquer autenticação extra para o uso da aplicação. A este tipo de solução chama-se *Single Sign-On* (SSO) e que permite a um utilizador entrar com o seu nome e *password* uma única vez e ter acesso a múltiplas aplicações.

É na classe "*SecurityServices.cs*" que são recebidas as informações do utilizador que se está a ligar à aplicação.

```
public AppUser GetUserInContext()
{
    AppUser result = new AppUser() { Username = Environment.UserName };
    System.Threading.Thread.Sleep(3000);
```

Figura 64 - AppUser, (fonte própria)

As informações do utilizador são validadas e se pertencerem ao "adminGroup" terão acesso a adicionar, editar e eliminar conteúdos (ex: documentos, autores, etc), ou se pelo contrário pertencerem ao "nomalGroup" têm apenas permissão de consulta que inclui a pesquisa (por titulo, autor, etc.) e a efetuar o *download* do respetivo documento.

Nas configurações do *Application Configuration File* "App.confg" é feita a correspondência entre adminGroup ao grupo de segurança do domínio AppAdmins e o normalGroup ao grupo de domínio AppUsers. Isto significa que um utilizador que seja adicionado a um destes grupos de segurança do domínio adquire as respetivas permissões na aplicação. Por exemplo, no caso do utilizador de domínio "user1" que pertence ao grupo de segurança AppUsers, corresponde ao normalGroup na aplicação, ficando apenas com permissões para consulta de documentos, assim como qualquer utilizador que pertença ao mesmo grupo do domínio.

```
using (ctx)
{
    result.DomainName = ctx.ConnectedServer;

    // find a user
    UserPrincipal user = UserPrincipal.FindByIdentity(ctx, result.Username);

    // find the group in question
    GroupPrincipal adminGroup = GroupPrincipal.FindByIdentity(ctx, ConfigHelper.Instance.GroupNameAdmin);
    if (adminGroup == null)
        throw new Exception("Não foi encontrado o grupo de acesso " + ConfigHelper.Instance.GroupNameAdmin);
    GroupPrincipal normalGroup = GroupPrincipal.FindByIdentity(ctx, ConfigHelper.Instance.GroupNameNormal);
    if (normalGroup == null)
        throw new Exception("Não foi encontrado o grupo de acesso " + ConfigHelper.Instance.GroupNameNormal);

    if (user != null)
    {
        // check if user is member of that group
        if (user.IsMemberOf(adminGroup))
            result.CurrentAccessMode = AccessMode.Admin;
        else if (user.IsMemberOf(normalGroup))
            result.CurrentAccessMode = AccessMode.Normal;
    }
```

Figura 65 - User Access

5.3.3. Ligação á Base de Dados

O acesso da aplicação à base de dados, SQL, foi realizada através da tecnologia ADO.Net.

Na App.config está a *ConnectionString* que estabelece a ligação ao servidor SQL à base de dados MyDB através da autenticação de domínio (respetivos utilizadores e grupos).

```
<add name="MyConnectionString" connectionString="Data Source=SQLAPP;Initial Catalog=MyDB;Integrated security=True"
  providerName="System.Data.SqlClient" />
```

Figura 66 - Connection String SQL

Após estabelecida a ligação à base de dados é na classe DbManager onde se encontra um conjunto de componentes do *Data Provider*, como *Connection* e *Command*, responsáveis por estabelecer a conexão com a base de dados e executar comandos na mesma. Esta classe é também responsável por carregar os dados para o *DataTable* (tabela) que faz parte do *DataSet*.

```
public DataTable GetDataTable(string selectStatement)
{
    DataTable dt = new DataTable();

    using (SqlConnection con = new SqlConnection(ConfigHelper.Instance.MyConnectionString))
    using (SqlDataAdapter adp = new SqlDataAdapter(selectStatement, con))
    {
        adp.Fill(dt);

    }

    return dt;
}

  public void ExecuteCommand(string cmdText)
  {
      using (SqlConnection con = new SqlConnection(ConfigHelper.Instance.MyConnectionString))
      using (SqlCommand cmd = new SqlCommand(cmdText, con))
      {
          SqlParameter p = new SqlParameter();

          con.Open();
          cmd.ExecuteNonQuery();
      }
  }
```

Figura 67 - SQLConnection

Na classe DbServices encontram-se os comandos do *Data Adapter* que funcionam como ponte entre os dados desconectados (*DataSet*), isto é a tabela visível pelo utilizador, e os conectados na base de dados. Os comandos *Select*, *Insert*, *Update* e *Delete* vão procurar, inserir, atualizar e apagar dados na tabela respetivamente. O método *Fill* altera os dados no *DataSet* para coincidir com os dados da base de dados.

5.3.4. Adicionar Documento

Adicionar novos documentos só pode ser efetuado por um utilizador administrador da aplicação.

Na janela principal carregar em "Adicionar Documento"

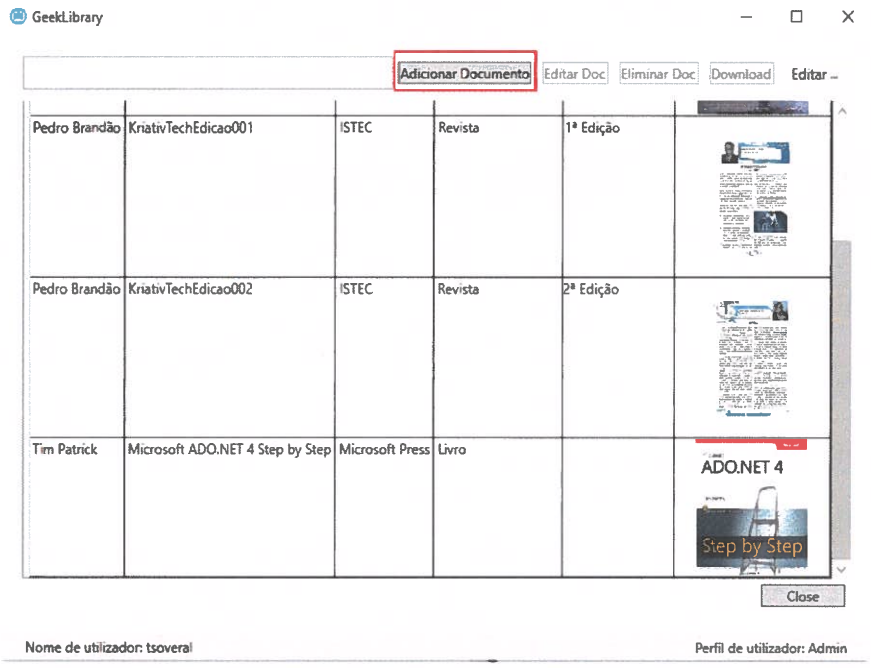

Figura 68 - Adicionar Documento

Vai abrir uma nova janela "Adicionar Documento". Nessa janela existe uma caixa de texto colocar Titulo do Documento, três *ComboBox* para selecionar o nome do Autor, a Editora e o Tipo de documento. Cada uma destas recebe o seu conteúdo através do *Biding* das

respetivas tabelas isto é, a *ComboBox* Autor vai listar todos os nomes dos autores existentes na tabela de autores para selecionar, através de um *binding*.

Na caixa de texto Resumo pode ser acrescentado o resumo do documento que se está a adicionar.

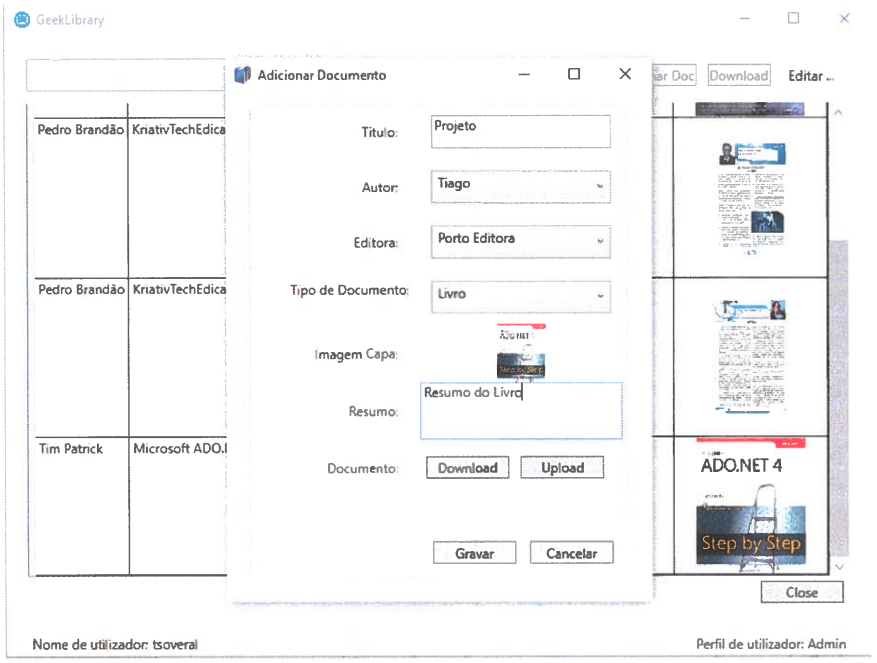

Figura 69 - Janela Adicionar Documento

Por baixo da caixa de texto Resumo aparecem dois botões *Download* e *Upload*, sendo que o primeiro só aparece disponível caso já exista um documento associado. O segundo botão serve para importar/adicionar um documento em modo PDF para a aplicação, aparecendo a imagem da capa (ou primeira página) do documento após este ser importado com sucesso.

Ao carregar no botão *Upload* (BtnUpload_Click) para importar um documento, este é convertido num *array* de *bytes* para ser guardado diretamente na base de dados. Para realizar a conversão é utilizada a classe PdfHelper. A conversão para o *array* de *bytes* é realizada para duas situações, uma para guardar efetivamente o documento todo (ContectBytes) e a segunda para guardar a capa (ConverBytes) que é visualizada como imagem na janela dos documentos.

```
private void BtnUpload_Click(object sender, RoutedEventArgs e)
{
    OpenFileDialog op = new OpenFileDialog();
    op.Title = "Selecione um PDF";
    op.Filter = "Ficheiros PDF|*.pdf";
    if (op.ShowDialog() == true)
    {

        Doc.CoverBytes = PdfHelper.Instance.GetCover(op.FileName);
        Doc.ContentBytes = PdfHelper.Instance.GetContent(op.FileName);

        MemoryStream stream = new MemoryStream(Doc.CoverBytes);
        BitmapImage image = new BitmapImage();
        image.BeginInit();
        image.StreamSource = stream;
        image.EndInit();
        ImgCover.Source = image;

        RefreshBtns();
    }
}
```

Figura 70 - Botão Upload

A visualização da capa do documento como imagem na janela principal e na janela de adicionar documento é feita através do processo inverso, converter o *array* de *bytes* para imagem (*bitmapimage*[13]), que é realizado através da classe BinaryImageConverter.cs.

No campo titulo da janela existe uma validação chamada "RequiredField" que valida se o campo se encontra preenchido. Se o campo estiver vazio fica com a caixa de texto a vermelho, com um ponto de exclamação como aviso e o botão "Gravar" fica desabilitado sendo necessário adicionar um título para poder efetuar a gravação.

```
<Binding Path="Doc.Title" Mode="TwoWay" UpdateSourceTrigger="PropertyChanged">
    <Binding.ValidationRules>
        <validators:RequiredField ErrorMessage="Titulo é obrigatório." />
    </Binding.ValidationRules>
</Binding>
```

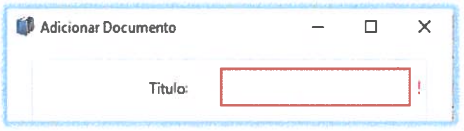

Figura 71 - Validação do título

Só após carregar no botão Gravar é que todos os dados vão ser guardados na base de dados. Esta tarefa é realizada pela classe DbService através do comando "*insert into*". Os

[13] Conjunto de pontos (pixéis) contidos num quadro, cada um destes pontos possuindo um ou vários valores que descrevem a sua cor

campos do Abstract do ContentBytes e do ConverBytes têm a particularidade de poderem ser inseridos com valores "Null" para o caso de não ter sido importado nenhum documento ou escrito algum texto no Resumo. É necessário também que os mesmos campos estejam com permissão de "Allow Null" na base dados SQL.

```
public bool AddDoc(Document doc)
{
    SqlCommand cmd = new SqlCommand("insert into Document (AuthorId,PublisherId,DocumentTypeId,Title,Abstract,CoverBytes,ContentBytes) " +
                                    "values(@AuthorId,@PublisherId,@DocumentTypeId,@Title,@Abstract,@CoverBytes,@ContentBytes)");
    cmd.Parameters.Add(new SqlParameter("@AuthorId", doc.AuthorId));
    cmd.Parameters.Add(new SqlParameter("@PublisherId", doc.PublisherId));
    cmd.Parameters.Add(new SqlParameter("@DocumentTypeId", doc.DocumentTypeId));
    cmd.Parameters.Add(new SqlParameter("@Title", doc.Title));
    //cmd.Parameters.Add(new SqlParameter("@Abstract", doc.Abstract));
    SqlParameter a = new SqlParameter("@Abstract", SqlDbType.NVarChar);
    if (string.IsNullOrEmpty(doc.Abstract))
        a.Value = DBNull.Value;
    else
        a.Value = doc.Abstract;
    cmd.Parameters.Add(a);
    SqlParameter coverParam = new SqlParameter("@CoverBytes", SqlDbType.VarBinary);
    if (doc.CoverBytes == null)
        coverParam.Value = DBNull.Value;
    else
        coverParam.Value = doc.CoverBytes;
    cmd.Parameters.Add(coverParam);
    SqlParameter contentParam = new SqlParameter("@ContentBytes", SqlDbType.VarBinary);
    if (doc.ContentBytes == null)
        contentParam.Value = DBNull.Value;
    else
        contentParam.Value = doc.ContentBytes;
    cmd.Parameters.Add(contentParam);
```

Figura 72 – SQL insert document

O botão "Cancelar" evita que os dados sejam guardados na base de dados e retorna á janela principal.

No caso de ser necessário alterar a informação inserida no documento, como por exemplo o título, o resumo ou até o ficheiro do documento, existem duas maneiras possíveis de o fazer. Uma delas é selecionar o documento e depois carregar no botão "Editar Doc", outra é percorrer a lista fazendo "duplo clique" (função gridDocs_MouseDoubleClick) sobre o documento escolhido. Para ambos os casos será apresentada uma nova janela "Editar Documento" semelhante à janela "Adicionar Documento". No entanto a janela é apresentada com os campos já pré-preenchidos (através de um *binding*), conforme as informações guardadas na base de dados, depois é só editar e guardar as respetivas alterações. As alterações vão ser guardadas na base de dados através do comando "*update*" na classe DbService.

```
private void gridDocs_MouseDoubleClick(object sender, MouseButtonEventArgs e)
{
    if(User.CurrentAccessMode != AccessMode.Admin)
    {
        MessageBox.Show("O seu perfil de utilizador não pode editar documentos.", "Acesso negado", MessageBoxButton.OK, MessageBoxImage.Error);
        return;
    }

    Document doc = (Document)gridDocs.SelectedItem;
    DocumentWindow window = new DocumentWindow(EditionMode.Edit, doc);
    bool? result = window.ShowDialog();
    if (result.HasValue && result.Value)
        Refresh();
}
```

No caso de ser apenas um utilizador de consulta (sem permissões para editar documentos) não estará visível o botão "Editar Doc" e se fizer "duplo clique" sobre o documento aparece uma caixa de texto com a mensagem "O seu perfil de utilizador não pode editar documentos".

Para eliminar um documento, na janela principal, selecionar um documento e fazer clique no botão "Eliminar Doc". Será ainda apresentada uma mensagem de confirmação para apagar definitivamente o documento, e em caso positivo o seu conteúdo é apagado na base de dados através do comando "Delete" na classe DbService.

```
public bool DeleteDoc(int id)
{
    DbManager.Instance.ExecuteCommand("delete from Document where Id = " + id);
    return true;
}
```

Figura 73 - Delete Document

Apagar um documento não elimina o Autor, Editora ou Tipo de Documento, pois estes campos podem estar a ser utilizados por outos documentos. Por exemplo, um autor pode ter vários livros, e apenas se pretende eliminar um desses livros.

5.3.5. Adicionar Autor, Editora e Tipo de Documento

Na janela principal existe um botão "Editar ..." que contém a lista para o acesso às informações dos Autores, Editoras e Tipos de Documento.

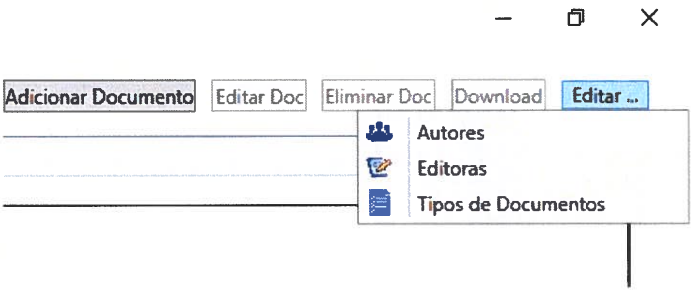

Figura 74 - Menu Autores, Editoras, Tipos de Documentos

Ao selecionar "Editoras" é apresentada uma janela com a informação das Editoras. Esta tabela apresenta os campos Nome, Endereço, Email e Contacto. É possível encontrar uma editora ou informação sobre a mesma através da pesquisa na caixa de texto. A tabela será atualizada com a informação pesquisada.

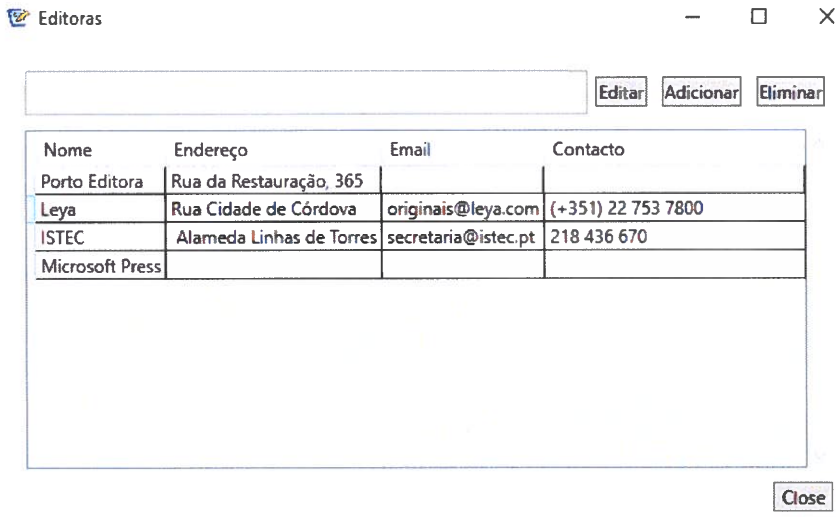

Figura 75 - Janela Editoras

É possível também adicionar, editar e eliminar editoras. Para adicionar uma nova editora carregar no botão "Adicionar" e surge uma janela para preencher os campos Nome, Endereço, Email, e Contacto.

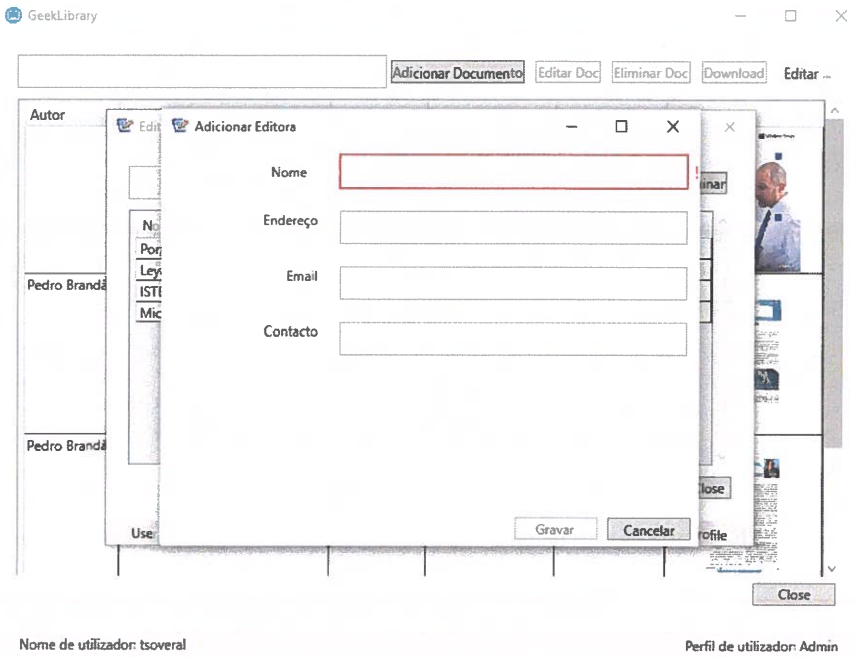

Figura 76 - Campos de adicionar editora

O campo Nome tem um método de validação igual ao campo título da janela "Adicionar Documento", em que não pode estar vazio. E só com o seu preenchimento o botão "Gravar" fica habilitado. Os restantes campos são aceites para gravação sem estarem preenchidos (valor *null*). No entanto o campo Email tem outro tipo de validação. Pode ser guardado sem nenhum valor, mas no caso de ser preenchido tem de respeitar o formato do endereço de email. Esta validação e feita através de uma expressão "regex[14]" na classe "ValidateEmail":

```
strmail = strmail.ToLower(); //Passa todas as letras do email para minusculas
//Link para a Regex: https://msdn.microsoft.com/en-us/library/01escwtf(v=vs.110).aspx
Regex rg = new Regex(@"^(?(""")(""".+?(?<!\\)""@)|(([0-9a-z]((\.(?!\.))|[-!#\$%&'\*\+/=\?\^`\{\}\|~\w])*)(?<=[0-9a-z])@))" +
    @"(?(\[)(\[(\d{1,3}\.){3}\d{1,3}\])|(([0-9a-z][-\w]*[0-9a-z]*\.)+[a-z0-9][\-a-z0-9]{0,22}[a-z0-9]))$");

if (!rg.IsMatch(strmail))
{
    return new ValidationResult(false, this.ErrorMessage);
}
```

Figura 77 - Regex de validação de email

Caso não cumpra com a validação a caixa de texto aparece a vermelho e o botão "Gravar" fica desabilitado.

[14] Uma expressão regular é uma notação para representar padrões em *strings*. Serve para validar entradas de dados ou fazer busca e extração de informações em textos

O botão "Gravar" (BtnSave_Click) para além de guardar as informações na base de dados vai através do "RefreshReferenceData" atualizar os dados na tabela da janela Editoras. Esta instrução encontra-se na classe DbService juntamente com comando "insert into" que irá guardar as novas informações na base de dados.

```
private void BtnSave_Click(object sender, RoutedEventArgs e)
{
    switch (Mode)
    {
        case EditionMode.Add:
            DbService.Instance.AddAut(Aut);
            DbService.Instance.RefreshReferenceData();
            this.DialogResult = true;
            break;
        case EditionMode.Edit:
            DbService.Instance.UpdateAuthors(Aut);
            DbService.Instance.RefreshReferenceData();
            this.DialogResult = true;
            break;
    }
}
```

Figura 78 - Botão guardar

Para editar os dados pode ser feito através do botão "Editar" ou "duplo clique" no documento pretendido. A janela para editar os campos é idêntica, sendo que os campos já aparecem preenchidos com as informações anteriores. As alterações são guardadas na base de dados através do comando "update" na tabela Publisher da base de dados.

```
public bool UpdatePublisher(Publisher pub)
{
    SqlCommand cmd = new SqlCommand("update Publisher set " +
                                    "Name = @Name, " +
                                    "Address = @Address, " +
                                    "Email = @Email, " +
                                    "PhoneNumber = @PhoneNumber " +
                    "where Id = @Id");

    cmd.Parameters.Add(new SqlParameter("@Id", pub.Id));
    cmd.Parameters.Add(new SqlParameter("@Name", pub.Name));
    cmd.Parameters.Add(new SqlParameter("@Address", pub.Address));
    cmd.Parameters.Add(new SqlParameter("@Email", pub.Email));
    cmd.Parameters.Add(new SqlParameter("@PhoneNumber", pub.PhoneNumber));
```

Figura 79 - Atualizar Editora no SQL

Para eliminar uma Editora basta selecionar a Editora e carregar no botão "Eliminar" e será apagada todas as informações referentes a essa editora. Supondo que a editora está a ser utilizada por um documento, não será possível eliminar a mesma, dando erro. Para a poder remover terá sempre de se desassociar de todos os documentos a que pertença.

O funcionamento das ações editar e eliminar são idênticos para os Autores e Tipo de documentos mudando apenas os campos a preencher, pois as janelas são semelhantes.

6. Conclusão

O rápido desenvolvimento e evolução da informática ao longo dos anos trouxe o aparecimento da virtualização e a sua contribuição para uma melhor gestão de infraestruturas nas empresas. Apesar do seu aparecimento em meados dos anos 70 só na última década é que existiu uma grande evolução e adoção desta tecnologia nas organizações. Os ganhos em poupança de investimento, gestão e manutenção, bem como de energia e redução do espaço físico necessário, superam o custo dispendioso deste tipo de implementações.

O projeto global foi implementado com sucesso, cumprindo todos os objetivos propostos demonstrando o conceito de virtualização e a interligação dos vários serviços entre os servidores, bem como a configuração e criação de um *template* base para criar novas máquinas virtuais. A criação da biblioteca digital foi também conseguida com todas as suas funções fundamentais associadas.

7. Bibliografia

Adams, M. (2011). Introduction to Virtualization, 61. Retrieved from https://www.ieee.li/pdf/viewgraphs/introduction_to_virtualization.pdf

Advantages, C., & Server, W. (2013). Why Hyper-V ?, (October), 1–44. Retrieved from http://download.microsoft.com/download/e/8/e/e8ecbd78-f07a-4a6f-9401-aa1760ed6985/competitive-advantages-of-windows-server-hyper-v-over-vmware-vsphere.pdf

Beaulieu, A. (2009). *Learning SQL. Database*. Retrieved from http://books.google.com/books?id=1PgCCVryjOQC

Chenley. (2011). Hypervisors. Retrieved from https://blogs.technet.microsoft.com/chenley/2011/02/09/hypervisors/

Ferreira, N. (2004). *Ado.net*. Porto. Retrieved from https://www.google.pt/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0ahUKEwj78bOt4JvLAhXIlxoKHWEwC50QFggbMAA&url=http://www.dei.isep.ipp.pt/~jtavares/ADAV/ADAV_2004_2005/Apresenta%C3%A7%C3%A3o_AdoDotNet.pdf&usg=AFQjCNFIwft5bb88EMrzT269S2rv7D

Interactive Online SQL Training. (2016). Retrieved from http://www.sqlcourse.com/intro.html

Lendvai, A. J., & Shi, H. (2007). ADO and ADO . NET Object Model Comparisons : A Relational Perspective, *7*(1), 330–337. Retrieved from http://paper.ijcsns.org/07_book/200701/200701B18.pdf

Microsoft. (2012). *Upgrating Your Skills to MCSA Windows Server 2012*.

Mistry, R., & Misner, S. (2014). *Introducing Microsoft SQL Server 2014 Technical Overview*.

Patrick, T. (2010). *Microsoft ADO.NET 4 Step by Step.*

Rebecca M.Riordan. (2002). *Microsoft ADO .NET.* (M. Press & O. M. Way, Eds.) (Vol. 2002).

Rosa, A. (2010). *Windows Server 2008R2.*

Ruest, D., & Ruest, N. (2009). *Virtualization: A Beginner's Guide.*

8. Anexos

Código da aplicação ADO.NET

```
1  using GeekLibrary.Utilities;
2  using System;
3  using System.Collections.Generic;
4  using System.Data;
5  using System.Data.SqlClient;
6  using System.Linq;
7  using System.Text;
8
9  namespace GeekLibrary.DataAccess
10 {
11     public class DbManager : GenericSingleton<DbManager>
12     {
13         public DataTable GetDataTable(string selectStatement)
14         {
15             DataTable dt = new DataTable();
16             using (SqlConnection con = new SqlConnection
                   (ConfigHelper.Instance.MyConnectionString))
17             using (SqlDataAdapter adp = new SqlDataAdapter(selectStatement,
                   con))
18             {
19                 adp.Fill(dt);
20             }
21             return dt;
22         }
23         public void ExecuteCommand(string cmdText)
24         {
25             using (SqlConnection con = new SqlConnection
                   (ConfigHelper.Instance.MyConnectionString))
26             using (SqlCommand cmd = new SqlCommand(cmdText, con))
27             {
28                 SqlParameter p = new SqlParameter();
29                 con.Open();
30                 cmd.ExecuteNonQuery();
31             }
32         }
33         public void ExecuteCommand(SqlCommand cmd)
34         {
35             if (cmd == null)
36                 return;
37             if (cmd.Connection == null)
38                 cmd.Connection = new SqlConnection
                      (ConfigHelper.Instance.MyConnectionString);
39             using (cmd)
40             {
41                 if (cmd.Connection.State != ConnectionState.Open)
42                     cmd.Connection.Open();
43                 cmd.ExecuteNonQuery();
44             }
45         }
46     }
47 }
48
```

```
1  using GeekLibrary.DataAccess;
2  using GeekLibrary.Models;
3  using GeekLibrary.Utilities;
4  using System;
5  using System.Collections.Generic;
6  using System.Data;
7  using System.Data.SqlClient;
8  using System.Linq;
9  using System.Text;
10
11 namespace GeekLibrary.Services
12 {
13     public class DbService : GenericSingleton<DbService>
14     {
15         public List<DocumentType> DocTypes { get; set; }
16         public List<Author> Authors { get; set; }
17         public List<Publisher> Publishers { get; set; }
18
19         //Refresh da tabela Autores
20         public void RefreshAuthors()
21         {
22             DataTable dt = DbManager.Instance.GetDataTable("select * from
                 Author");
23             if (dt != null && dt.Rows.Count > 0)
24             {
25                 Authors = new List<Author>();
26                 foreach (DataRow r in dt.Rows)
27                     Authors.Add(FillAuthor(r));
28             }
29         }
30
31         //Refresh da tabela Tipo de Documento
32         public void RefreshDocumentTypes()
33         {
34             DataTable dt = DbManager.Instance.GetDataTable("select * from
                 DocumentType");
35             if (dt != null && dt.Rows.Count > 0)
36             {
37                 DocTypes = new List<DocumentType>();
38                 foreach (DataRow r in dt.Rows)
39                     DocTypes.Add(FillDocumentType(r));
40             }
41         }
42
43         //Refresh da tabela Editora
44         public void RefreshPublishers()
45         {
46             DataTable dt = DbManager.Instance.GetDataTable("select * from
                 Publisher");
47             if (dt != null && dt.Rows.Count > 0)
48             {
49                 Publishers = new List<Publisher>();
50                 foreach (DataRow r in dt.Rows)
51                     Publishers.Add(FillPublisher(r));
52             }
53         }
```

```
54
55
56
57
58          //Faz o Refresh de todas as tabelas
59          public void RefreshReferenceData()
60          {
61              RefreshPublishers();
62              RefreshAuthors();
63              RefreshDocumentTypes();
64          }
65
66          public void InitializeReferenceData()
67          {
68              DataTable dt = DbManager.Instance.GetDataTable("select * from
                  DocumentType");
69              if (dt != null && dt.Rows.Count > 0)
70              {
71                  DocTypes = new List<DocumentType>();
72                  foreach (DataRow r in dt.Rows)
73                      DocTypes.Add(FillDocumentType(r));
74              }
75              dt = DbManager.Instance.GetDataTable("select * from Author");
76              if (dt != null && dt.Rows.Count > 0)
77              {
78                  Authors = new List<Author>();
79                  foreach (DataRow r in dt.Rows)
80                      Authors.Add(FillAuthor(r));
81              }
82              dt = DbManager.Instance.GetDataTable("select * from Publisher");
83              if (dt != null && dt.Rows.Count > 0)
84              {
85                  Publishers = new List<Publisher>();
86                  foreach (DataRow r in dt.Rows)
87                      Publishers.Add(FillPublisher(r));
88              }
89          }
90
91
92
93          public List<Document> GetAllDocs()
94          {
95              List<Document> result = new List<Document>();
96              DataTable dt = DbManager.Instance.GetDataTable("select * from
                  Document");
97              if(dt != null && dt.Rows.Count> 0)
98              {
99                  foreach (DataRow r in dt.Rows)
100                     result.Add(FillDocument(r));
101             }
102             return result;
103         }
104         public List<Publisher> GetPublishers()
105         {
106             List<Publisher> result = new List<Publisher>();
107             DataTable dt = DbManager.Instance.GetDataTable("select * from
```

```
                Publisher");
108             if (dt != null && dt.Rows.Count > 0)
109             {
110                 foreach (DataRow r in dt.Rows)
111                     result.Add(FillPublisher(r));
112             }
113             return result;
114         }
115         public List<Author> GetAuthors()
116         {
117             List<Author> result = new List<Author>();
118             DataTable dt = DbManager.Instance.GetDataTable("select * from
                  Author");
119             if (dt != null && dt.Rows.Count > 0)
120             {
121                 foreach (DataRow r in dt.Rows)
122                     result.Add(FillAuthor(r));
123             }
124             return result;
125         }
126         public List<DocumentType> GetDocumentTypes()
127         {
128             List<DocumentType> result = new List<DocumentType>();
129             DataTable dt = DbManager.Instance.GetDataTable("select * from
                  DocumentType");
130             if (dt != null && dt.Rows.Count > 0)
131             {
132                 foreach (DataRow r in dt.Rows)
133                     result.Add(FillDocumentType(r));
134             }
135             return result;
136         }
137
138
139
140         //Search
141         public List<Publisher> SearchPublishers(string search)
142         {
143             List<Publisher> result = new List<Publisher>();
144             DataTable dt = DbManager.Instance.GetDataTable(
145                                 string.Format("select * " +
146                                               "from Publisher " +
147                                                 "where Name like '%{0}%' or " +
148                                                     "Address like '%{0}%' or " +
149                                                     "Email like '%{0}%' or " +
150                                                     "PhoneNumber like '%{0}%'",
                        search));
151             if (dt != null && dt.Rows.Count > 0)
152             {
153                 foreach (DataRow r in dt.Rows)
154                     result.Add(FillPublisher(r));
155             }
156             return result;
157         }
158         public List<Document> SearchDocs(string search)
159         {
```

```
160             List<Document> result = new List<Document>();
161             DataTable dt = DbManager.Instance.GetDataTable(
162                             string.Format("select doc.* " +
163                                           "from Document doc " +
164                                             "inner join DocumentType dt " +
165                                                 "on dt.Id =
                    doc.DocumentTypeId " +
166                                           "inner join Publisher pub " +
167                                                 "on pub.Id = doc.PublisherId
                    " +
168                                           "inner join Author a " +
169                                                 "on a.Id = doc.AuthorId " +
170                                           "where doc.Title like '%{0}%' or
                    " +
171                                                 "dt.Description like '%{0}%'
                    or " +
172                                                 "pub.Name like '%{0}%' or " +
173                                                 "a.Name like '%{0}%'",
                    search));
174             if (dt != null && dt.Rows.Count > 0)
175             {
176                 foreach (DataRow r in dt.Rows)
177                     result.Add(FillDocument(r));
178             }
179             return result;
180         }
181         public List<DocumentType> SearchDocumentTypes(string search)
182         {
183             List<DocumentType> result = new List<DocumentType>();
184             DataTable dt = DbManager.Instance.GetDataTable(
185                             string.Format("select * " +
186                                           "from DocumentType " +
187                                             "where Description like '%{0}%'",
                     search));
188             if (dt != null && dt.Rows.Count > 0)
189             {
190                 foreach (DataRow r in dt.Rows)
191                     result.Add(FillDocumentType(r));
192             }
193             return result;
194         }
195         public List<Author> SearchAuthors(string search)
196         {
197             List<Author> result = new List<Author>();
198             DataTable dt = DbManager.Instance.GetDataTable(
199                             string.Format("select * " +
200                                           "from Author " +
201                                             "where Name like '%{0}%' or " +
202                                                 "Nationality like '%{0}%' " ,
                     search));
203             if (dt != null && dt.Rows.Count > 0)
204             {
205                 foreach (DataRow r in dt.Rows)
206                     result.Add(FillAuthor(r));
207             }
208             return result;
```

```
209         }
210
211
212         //Update
213         public bool UpdateDoc(Document doc)
214         {
215             SqlCommand cmd = new SqlCommand("update Document set " +
216                                             "AuthorId = @AuthorId, " +
217                                             "DocumentTypeId =
                    @DocumentTypeId, " +
218                                             "PublisherId = @PublisherId, "
                     +
219                                             "Title = @Title, " +
220                                             "Abstract = @Abstract, " +
221                                             "CoverBytes = @CoverBytes, " +
222                                             "ContentBytes = @ContentBytes
                    " +
223                                         "where Id = @Id");
224
225             cmd.Parameters.Add(new SqlParameter("@Id", doc.Id));
226             cmd.Parameters.Add(new SqlParameter("@AuthorId", doc.AuthorId));
227             cmd.Parameters.Add(new SqlParameter("@PublisherId",
                  doc.PublisherId));
228             cmd.Parameters.Add(new SqlParameter("@DocumentTypeId",
                  doc.DocumentTypeId));
229             cmd.Parameters.Add(new SqlParameter("@Title", doc.Title));
230             cmd.Parameters.Add(new SqlParameter("@Abstract", doc.Abstract));
231             SqlParameter coverParam = new SqlParameter("@CoverBytes",
                  SqlDbType.VarBinary);
232             if (doc.CoverBytes == null)
233                 coverParam.Value = DBNull.Value;
234             else
235                 coverParam.Value = doc.CoverBytes;
236             cmd.Parameters.Add(coverParam);
237             SqlParameter contentParam = new SqlParameter("@ContentBytes",
                  SqlDbType.VarBinary);
238             if (doc.ContentBytes == null)
239                 contentParam.Value = DBNull.Value;
240             else
241                 contentParam.Value = doc.ContentBytes;
242             cmd.Parameters.Add(contentParam);
243
244             DbManager.Instance.ExecuteCommand(cmd);
245             return true;
246         }
247         public bool UpdatePublisher(Publisher pub)
248         {
249             SqlCommand cmd = new SqlCommand("update Publisher set " +
250                                             "Name = @Name, " +
251                                             "Address = @Address, " +
252                                             "Email = @Email, " +
253                                             "PhoneNumber =
                    @PhoneNumber " +
254                                     "where Id = @Id");
255
256             cmd.Parameters.Add(new SqlParameter("@Id", pub.Id));
```

```
257             cmd.Parameters.Add(new SqlParameter("@Name", pub.Name));
258             cmd.Parameters.Add(new SqlParameter("@Address", pub.Address));
259             cmd.Parameters.Add(new SqlParameter("@Email", pub.Email));
260             cmd.Parameters.Add(new SqlParameter("@PhoneNumber",
                  pub.PhoneNumber));
261             DbManager.Instance.ExecuteCommand(cmd);
262             return true;
263         }
264         public bool UpdateDocumentType(DocumentType doctype)
265         {
266             SqlCommand cmd = new SqlCommand("update DocumentType set " +
267                                             "Description = @Description "
                    +
268                                         "where Id = @Id");
269
270             cmd.Parameters.Add(new SqlParameter("@Id", doctype.Id));
271             cmd.Parameters.Add(new SqlParameter("@Description",
                  doctype.Description));
272
273             DbManager.Instance.ExecuteCommand(cmd);
274             return true;
275         }
276         public bool UpdateAuthors(Author aut)
277         {
278             SqlCommand cmd = new SqlCommand("update Author set " +
279                                             "Name = @Name, " +
280                                             "Nationality = @Nationality "
                    +
281                                         "where Id = @Id");
282
283             cmd.Parameters.Add(new SqlParameter("@Id", aut.Id));
284             cmd.Parameters.Add(new SqlParameter("@Name", aut.Name));
285             cmd.Parameters.Add(new SqlParameter("@Nationality",
                  aut.Nationality));
286
287             DbManager.Instance.ExecuteCommand(cmd);
288             return true;
289         }
290
291
292         //Add
293         public bool AddDoc(Document doc)
294         {
295             SqlCommand cmd = new SqlCommand("insert into Document
                  (AuthorId,PublisherId,DocumentTypeId,Title,Abstract,CoverBytes,C
                  ontentBytes) " +
296                                         "values
                        (@AuthorId,@PublisherId,@DocumentTypeId,@Title,@Abstract,@
                        CoverBytes,@ContentBytes)");
297             cmd.Parameters.Add(new SqlParameter("@AuthorId", doc.AuthorId));
298             cmd.Parameters.Add(new SqlParameter("@PublisherId",
                  doc.PublisherId));
299             cmd.Parameters.Add(new SqlParameter("@DocumentTypeId",
                  doc.DocumentTypeId));
300             cmd.Parameters.Add(new SqlParameter("@Title", doc.Title));
301             SqlParameter a = new SqlParameter("@Abstract",
```

```
                SqlDbType.NVarChar);
302             if (string.IsNullOrEmpty(doc.Abstract))
303                 a.Value = DBNull.Value;
304             else
305                 a.Value = doc.Abstract;
306             cmd.Parameters.Add(a);
307             SqlParameter coverParam = new SqlParameter("@CoverBytes",
                  SqlDbType.VarBinary);
308             if (doc.CoverBytes == null)
309                 coverParam.Value = DBNull.Value;
310             else
311                 coverParam.Value = doc.CoverBytes;
312             cmd.Parameters.Add(coverParam);
313             SqlParameter contentParam = new SqlParameter("@ContentBytes",
                  SqlDbType.VarBinary);
314             if (doc.ContentBytes == null)
315                 contentParam.Value = DBNull.Value;
316             else
317                 contentParam.Value = doc.ContentBytes;
318             cmd.Parameters.Add(contentParam);
319
320             DbManager.Instance.ExecuteCommand(cmd);
321             return true;
322         }
323         public bool AddPub(Publisher pub)
324         {
325             SqlCommand cmd = new SqlCommand("insert into Publisher
                  (Name,Address,Email,PhoneNumber) " +
326                                             "values
                        (@Name,@Address,@Email,@PhoneNumber)");
327             cmd.Parameters.Add(new SqlParameter("@Name", pub.Name));
328
329             SqlParameter p = new SqlParameter("@Address", SqlDbType.NVarChar);
330             if (string.IsNullOrEmpty(pub.Address))
331                 p.Value = DBNull.Value;
332             else
333                 p.Value = pub.Address;
334             cmd.Parameters.Add(p);
335             p = new SqlParameter("@Email", SqlDbType.NVarChar);
336             if (string.IsNullOrEmpty(pub.Email))
337                 p.Value = DBNull.Value;
338             else
339                 p.Value = pub.Email;
340             cmd.Parameters.Add(p);
341
342             p = new SqlParameter("@PhoneNumber", SqlDbType.NVarChar);
343             if (string.IsNullOrEmpty(pub.PhoneNumber))
344                 p.Value = DBNull.Value;
345             else
346                 p.Value = pub.PhoneNumber;
347             cmd.Parameters.Add(p);
348             DbManager.Instance.ExecuteCommand(cmd);
349             return true;
350         }
351         public bool AddDockType(DocumentType docktype)
352         {
```

```
353             SqlCommand cmd = new SqlCommand("insert into DocumentType
                  (Description) " +
354                                             "values(@Description)");
355             cmd.Parameters.Add(new SqlParameter("@Description",
                  docktype.Description));
356             DbManager.Instance.ExecuteCommand(cmd);
357             return true;
358         }
359         public bool AddAut(Author aut)
360         {
361             SqlCommand cmd = new SqlCommand("insert into Author
                  (Name,Nationality) " +
362                                             "values(@Name,@Nationality)");
363             cmd.Parameters.Add(new SqlParameter("@Name",aut.Name));
364
365             SqlParameter n = new SqlParameter("@Nationality",
                  SqlDbType.NVarChar);
366             if (string.IsNullOrEmpty(aut.Nationality))
367                 n.Value = DBNull.Value;
368             else
369                 n.Value = aut.Nationality;
370             cmd.Parameters.Add(n);
371             DbManager.Instance.ExecuteCommand(cmd);
372             return true;
373         }
374
375
376         //Delete
377         public bool DeleteDoc(int id)
378         {
379             DbManager.Instance.ExecuteCommand("delete from Document where Id =
                    " + id);
380             return true;
381         }
382         public bool DeletePub(int id)
383         {
384             DbManager.Instance.ExecuteCommand("delete from Publisher where Id
                  = " + id);
385             return true;
386         }
387         public bool DeleteDockType(int id)
388         {
389             DbManager.Instance.ExecuteCommand("delete from DocumentType where
                  Id = " + id);
390             return true;
391         }
392         public bool DeleteAuthor(int id)
393         {
394             DbManager.Instance.ExecuteCommand("delete from Author where Id = "
                    + id);
395             return true;
396         }
397
398
399
400         //Fill
```

```
401         private Document FillDocument(DataRow row)
402         {
403             Document doc = new Document
404             {
405                 Id = Convert.ToInt32(row["Id"]),
406                 AuthorId = Convert.ToInt32(row["AuthorId"]),
407                 PublisherId = Convert.ToInt32(row["PublisherId"]),
408                 DocumentTypeId = Convert.ToInt32(row["DocumentTypeId"]),
409                 Title = row["Title"].ToString(),
410                 Abstract = row["Abstract"].ToString(),
411                 CoverBytes = row["CoverBytes"] == DBNull.Value ? null : (byte
                      [])row["CoverBytes"],
412                 ContentBytes = row["ContentBytes"] == DBNull.Value ? null :
                      (byte[])row["ContentBytes"]
413             };
414             doc.Author = Authors.FirstOrDefault(a => a.Id == doc.AuthorId);
415             doc.Publisher = Publishers.FirstOrDefault(p => p.Id ==
                  doc.PublisherId);
416             doc.DocType = DocTypes.FirstOrDefault(t => t.Id ==
                  doc.DocumentTypeId);
417             return doc;
418         }
419         private DocumentType FillDocumentType(DataRow row)
420         {
421             DocumentType type = new DocumentType
422             {
423                 Id = Convert.ToInt32(row["Id"]),
424                 Description = row["Description"].ToString()
425             };
426             return type;
427         }
428         private Publisher FillPublisher(DataRow row)
429         {
430             Publisher pub = new Publisher
431             {
432                 Id = Convert.ToInt32(row["Id"]),
433                 Name = row["Name"].ToString(),
434                 Address = row["Address"].ToString(),
435                 Email = row["Email"].ToString(),
436                 PhoneNumber = row["PhoneNumber"].ToString()
437             };
438             return pub;
439         }
440         private Author FillAuthor(DataRow row)
441         {
442             Author type = new Author
443             {
444                 Id = Convert.ToInt32(row["Id"]),
445                 Name = row["Name"].ToString(),
446                 Nationality = row["Nationality"].ToString()
447             };
448             return type;
449         }
450     }
451 }
```

```
1  <?xml version="1.0" encoding="utf-8" ?>
2  <configuration>
3    <configSections>
4    </configSections>
5    <startup>
6      <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5.1" />
7    </startup>
8    <connectionStrings>
9      <add name="MyConnectionString" connectionString="Data Source=SQLAPP;Initial
         Catalog=MyDB;Integrated security=True"
10       providerName="System.Data.SqlClient" />
11   </connectionStrings>
12   <appSettings>
13     <add key="GroupNameAdmin" value="AppAdmins"/>
14     <add key="GroupNameNormal" value="AppUsers"/>
15   </appSettings>
16 </configuration>
```

```
1  using GeekLibrary.Models;
2  using GeekLibrary.Services;
3  using System;
4  using System.Collections.Generic;
5  using System.Linq;
6  using System.Text;
7  using System.Threading.Tasks;
8  using System.Windows;
9  using System.Windows.Controls;
10 using System.Windows.Data;
11 using System.Windows.Documents;
12 using System.Windows.Input;
13 using System.Windows.Media;
14 using System.Windows.Media.Imaging;
15 using System.Windows.Shapes;
16
17 namespace GeekLibrary
18 {
19     public partial class EditPublisherWindow : Window
20     {
21         EditionMode Mode;
22
23         public Publisher Pub { get; set; }
24
25         public EditPublisherWindow(EditionMode mode, Publisher pub)
26         {
27             this.Mode = mode;
28             this.Pub = pub;
29             if (this.Pub == null)
30             {
31                 this.Pub = new Publisher();
32             }
33
34             DataContext = this;
35
36             InitializeComponent();
37
38
39
40             switch (Mode)
41             {
42                 case EditionMode.Add:
43                     this.Title = "Adicionar Editora";
44                     break;
45                 case EditionMode.Edit:
46                     this.Title = "Editar Editora";
47                     break;
48             }
49
50             TxtName.GetBindingExpression(TextBox.TextProperty).UpdateSource();
51             TxtAddress.GetBindingExpression(TextBox.TextProperty).UpdateSource
                 ();
52             TxtEmail.GetBindingExpression(TextBox.TextProperty).UpdateSource();
53             TxtPhoneNumber.GetBindingExpression
                 (TextBox.TextProperty).UpdateSource();
54
```

```
55          }
56
57          private void BtnSave_Click(object sender, RoutedEventArgs e)
58          {
59              switch (Mode)
60              {
61                  case EditionMode.Add:
62                      DbService.Instance.AddPub(Pub);
63                      DbService.Instance.RefreshReferenceData();
64                      this.DialogResult = true;
65                      break;
66                  case EditionMode.Edit:
67                      DbService.Instance.UpdatePublisher(Pub);
68                      DbService.Instance.RefreshReferenceData();
69                      this.DialogResult = true;
70                      break;
71              }
72          }
73
74          private void BtnCancel_Click(object sender, RoutedEventArgs e)
75          {
76              this.DialogResult = false;
77          }
78      }
79  }
80
```

```
<Window x:Class="GeekLibrary.EditPublisherWindow"
        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:GeekLibrary"
        xmlns:converters="clr-namespace:GeekLibrary.Converters"
        xmlns:validators="clr-namespace:GeekLibrary.Validators"
        mc:Ignorable="d"
        Title="Editora" Icon="images/editoras.png" Height="400" Width="500"
          WindowStartupLocation="CenterScreen">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="150" />
            <ColumnDefinition />
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition Height="50" />
            <RowDefinition Height="50" />
            <RowDefinition Height="50" />
            <RowDefinition Height="50" />
            <RowDefinition Height="*" />
            <RowDefinition Height="50" />
            <RowDefinition Height="70" />
        </Grid.RowDefinitions>
        <TextBlock Grid.Column="0" Grid.Row="0" Text="Nome"
          VerticalAlignment="Center" HorizontalAlignment="Right" Margin="0 0 20
           0" />
        <TextBox Name="TxtName" Grid.Column="1" Grid.Row="0" Height="30"
          Margin="10">
            <TextBox.Text>
                <Binding Path="Pub.Name" Mode="TwoWay"
                  UpdateSourceTrigger="PropertyChanged">
                    <Binding.ValidationRules>
                        <validators:RequiredField ErrorMessage="Nome é
                    obrigatório." />
                    </Binding.ValidationRules>
                </Binding>
            </TextBox.Text>
        </TextBox>
        <TextBlock Grid.Column="0" Grid.Row="1" Text="Endereço"
          VerticalAlignment="Center" HorizontalAlignment="Right"
          Margin="0,10,10,24" />
        <TextBox Name="TxtAddress" Text="{Binding Pub.Address}" Grid.Column="1"
           Grid.Row="1" Height="30" Margin="10"></TextBox>
        <TextBlock Grid.Column="0" Grid.Row="2" Text="Email"
          VerticalAlignment="Center" HorizontalAlignment="Right"
          Margin="0,10,10,24" />
        <TextBox Name="TxtEmail" Grid.Column="2" Grid.Row="2" Height="30"
          Margin="10">
            <TextBox.Text>
                <Binding Path="Pub.Email" Mode="TwoWay"
                  UpdateSourceTrigger="PropertyChanged">
                    <Binding.ValidationRules>
                        <validators:ValidateEmail ErrorMessage="Email é
                    obrigatório." />
```

```
43                      </Binding.ValidationRules>
44                  </Binding>
45              </TextBox.Text>
46          </TextBox>
47          <TextBlock Grid.Column="0" Grid.Row="3" Text="Contacto"
              VerticalAlignment="Center" HorizontalAlignment="Right"
              Margin="0,10,10,24" />
48          <TextBox Name="TxtPhoneNumber" Text="{Binding Pub.PhoneNumber}"
              Grid.Column="3" Grid.Row="3" Height="30" Margin="10"></TextBox>
49          <StackPanel Orientation="Horizontal" Grid.Column="1" Grid.Row="6"
              HorizontalAlignment="Right" VerticalAlignment="Bottom">
50              <Button x:Name="BtnSave" Content="Gravar" Height="20" Width="75"
                  Margin="5" Click="BtnSave_Click" >
51                  <Button.Style>
52                      <Style TargetType="Button">
53                          <Setter Property="IsEnabled" Value="False"/>
54                          <Style.Triggers>
55                              <MultiDataTrigger>
56                                  <MultiDataTrigger.Conditions>
57                                      <Condition Binding="{Binding Path=
                        (Validation.HasError), ElementName=TxtName}" Value="False"/
                        >
58                                      <Condition Binding="{Binding Path=
                        (Validation.HasError), ElementName=TxtEmail}"
                        Value="False"/>
59                                  </MultiDataTrigger.Conditions>
60                                  <Setter Property="IsEnabled" Value="True"/>
61                              </MultiDataTrigger>
62                          </Style.Triggers>
63                      </Style>
64                  </Button.Style>
65              </Button>
66              <Button x:Name="BtnCancel" Content="Cancelar" Height="20"
                  Width="75" Margin="5" Click="BtnCancel_Click" />
67          </StackPanel>
68      </Grid>
69  </Window>
70
```

```
1  using GeekLibrary.Models;
2  using GeekLibrary.Services;
3  using GeekLibrary.Utilities;
4  using System;
5  using System.Collections.Generic;
6  using System.ComponentModel;
7  using System.Data;
8  using System.Linq;
9  using System.Text;
10 using System.Threading.Tasks;
11 using System.Windows;
12 using System.Windows.Controls;
13 using System.Windows.Data;
14 using System.Windows.Documents;
15 using System.Windows.Input;
16 using System.Windows.Media;
17 using System.Windows.Media.Imaging;
18 using System.Windows.Shapes;
19
20 namespace GeekLibrary
21 {
22     /// <summary>
23     /// Interaction logic for MainWindow.xaml
24     /// </summary>
25     public partial class MainWindow : Window, INotifyPropertyChanged
26     {
27         #region INotify
28         public event PropertyChangedEventHandler PropertyChanged;
29
30         protected void NotifyPropertyChanged(string propertyName)
31         {
32             if (PropertyChanged != null)
33                 PropertyChanged(this, new PropertyChangedEventArgs
                      (propertyName));
34         }
35         #endregion
36
37         private string searchQuery;
38         public string SearchQuery
39         {
40             get
41             {
42                 return searchQuery;
43             }
44
45             set
46             {
47                 searchQuery = value;
48                 NotifyPropertyChanged("SearchQuery");
49                 Refresh();
50             }
51         }
52         public AppUser User { get; set; }
53         public List<Document> Docs { get; set; }
54
55         public MainWindow(AppUser user, List<Document> docs)
```

```
56          {
57              InitializeComponent();
58              this.User = user;
59              this.Docs = docs;
60              gridDocs.ItemsSource = Docs;
61              RefresBtnsState();
62              if(user.CurrentAccessMode != AccessMode.Admin)
63              {
64                  BtnAdd.Visibility = Visibility.Collapsed;
65                  BtnEdit.Visibility = Visibility.Collapsed;
66                  BtnDel.Visibility = Visibility.Collapsed;
67                  BtnMenu.Visibility = Visibility.Collapsed;
68                  LblProfile.Text = "Perfil de utilizador: Normal";
69              }
70              else
71              {
72                  LblProfile.Text = "Perfil de utilizador: Admin";
73              }
74
75              LblUser.Text = "Nome de utilizador: " + User.Username;
76              this.DataContext = this;
77          }
78
79          private void RefresBtnsState()
80          {
81              BtnEdit.IsEnabled = gridDocs.SelectedIndex > -1;
82              BtnDel.IsEnabled = gridDocs.SelectedIndex > -1;
83              BtnDownload.IsEnabled = gridDocs.SelectedIndex > -1 && ((Document)
                  gridDocs.SelectedItem).ContentBytes != null;
84          }
85
86          private void BtnAdd_Click(object sender, RoutedEventArgs e)
87          {
88              try
89              {
90                  DocumentWindow window = new DocumentWindow(EditionMode.Add);
91                  bool? result = window.ShowDialog();
92                  if (result.HasValue && result.Value)
93                      Refresh();
94              }
95              catch(Exception ex)
96              {
97                  MessageBox.Show(ex.Message, "Erro", MessageBoxButton.OK,
                      MessageBoxImage.Error);
98              }
99
100         }
101
102         private void BtnEdit_Click(object sender, RoutedEventArgs e)
103         {
104             try
105             {
106                 Document doc = (Document)gridDocs.SelectedItem;
107                 DocumentWindow window = new DocumentWindow(EditionMode.Edit,
                      doc);
108                 bool? result = window.ShowDialog();
```

```
109                 if (result.HasValue && result.Value)
110                     Refresh();
111             }
112             catch (Exception ex)
113             {
114                 MessageBox.Show(ex.Message, "Erro", MessageBoxButton.OK,
                      MessageBoxImage.Error);
115             }
116 
117         }
118         private void BtnDel_Click(object sender, RoutedEventArgs e)
119         {
120             try
121             {
122                 Document doc = (Document)gridDocs.SelectedItem;
123                 if (MessageBox.Show("Tem a certeza que pretende eliminar o
                      item selecionado?", "Confirmação", MessageBoxButton.YesNo,
                      MessageBoxImage.Question) == MessageBoxResult.Yes)
124                 {
125                     DbService.Instance.DeleteDoc(doc.Id);
126                     Refresh();
127                 }
128             }
129             catch (Exception ex)
130             {
131                 MessageBox.Show(ex.Message, "Erro", MessageBoxButton.OK,
                      MessageBoxImage.Error);
132             }
133 
134         }
135         private void BtnDownload_Click(object sender, RoutedEventArgs e)
136         {
137             try
138             {
139                 Document doc = (Document)gridDocs.SelectedItem;
140                 System.Windows.Forms.FolderBrowserDialog dialog = new
                      System.Windows.Forms.FolderBrowserDialog();
141                 if (dialog.ShowDialog() ==
                      System.Windows.Forms.DialogResult.OK)
142                 {
143                     string filename = System.IO.Path.Combine
                      (dialog.SelectedPath, doc.Title + ".pdf");
144                     PdfHelper.Instance.SaveBytesToFile(filename,
                      doc.ContentBytes);
145                     System.Diagnostics.Process.Start(filename);
146                 }
147             }
148             catch (Exception ex)
149             {
150                 MessageBox.Show(ex.Message, "Erro", MessageBoxButton.OK,
                      MessageBoxImage.Error);
151             }
152         }
153 
154         private void gridDocs_SelectedCellsChanged(object sender,
              SelectedCellsChangedEventArgs e)
```

```
155         {
156             RefresBtnsState();
157         }
158
159         public void Refresh()
160         {
161             Task.Run(() =>
162             {
163                 if (string.IsNullOrEmpty(SearchQuery))
164                 {
165                     this.Docs = DbService.Instance.GetAllDocs();
166                 }
167                 else
168                 {
169                     this.Docs = DbService.Instance.SearchDocs(SearchQuery);
170                 }
171                 UpdateUIAsync();
172             });
173         }
174
175         private void UpdateUIAsync()
176         {
177             Dispatcher.Invoke(new Action(() =>
178             {
179                 this.gridDocs.ItemsSource = Docs;
180                 RefresBtnsState();
181             }));
182         }
183
184         private void gridDocs_MouseDoubleClick(object sender,
              MouseButtonEventArgs e)
185         {
186             if(User.CurrentAccessMode != AccessMode.Admin)
187             {
188                 MessageBox.Show("O seu perfil de utilizador não pode editar
                      documentos.", "Acesso negado", MessageBoxButton.OK,
                      MessageBoxImage.Error);
189                 return;
190             }
191
192             Document doc = (Document)gridDocs.SelectedItem;
193             DocumentWindow window = new DocumentWindow(EditionMode.Edit, doc);
194             bool? result = window.ShowDialog();
195             if (result.HasValue && result.Value)
196                 Refresh();
197         }
198
199
200         //Menu
201         private void MenuAut_Click(object sender, RoutedEventArgs e)
202         {
203             AutorWindow authorshow = new AutorWindow();
204             authorshow.ShowDialog();
205         }
206
207         private void MenuEdit_Click(object sender, RoutedEventArgs e)
```

```
208         {
209             PublisherWindow publishewd = new PublisherWindow();
210             publishewd.ShowDialog();
211         }
212 
213         private void MenuType_Click(object sender, RoutedEventArgs e)
214         {
215             DocumentTypeWindow docktypewd = new DocumentTypeWindow();
216             docktypewd.ShowDialog();
217         }
218 
219         //close application
220         private void BtnClose_Click(object sender, RoutedEventArgs e)
221         {
222             System.Environment.Exit(0);
223         }
224     }
225 }
226 
```

```
<Window x:Class="GeekLibrary.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:GeekLibrary"
        xmlns:converters="clr-namespace:GeekLibrary.Converters"
        mc:Ignorable="d"
        Title="GeekLibrary" Icon="images/gl3.png" Height="600" Width="800"
          WindowStartupLocation="CenterScreen" MinWidth="500" MinHeight="300">
    <Window.Resources>
        <converters:BinaryImageConverter x:Key="imgConverter" />
    </Window.Resources>
    <DockPanel>
        <StatusBar DockPanel.Dock="Bottom">
            <StatusBar.ItemsPanel>
                <ItemsPanelTemplate>
                    <Grid>
                        <Grid.RowDefinitions>
                            <RowDefinition Height="*"/>
                        </Grid.RowDefinitions>
                        <Grid.ColumnDefinitions>
                            <ColumnDefinition Width="*"/>
                            <ColumnDefinition Width="*"/>
                        </Grid.ColumnDefinitions>
                    </Grid>
                </ItemsPanelTemplate>
            </StatusBar.ItemsPanel>
            <StatusBarItem>
                <TextBlock Name="LblUser" Margin="20 0 0 0">User</TextBlock>
            </StatusBarItem>
            <StatusBarItem Grid.Column="1" HorizontalContentAlignment="Right">
                <TextBlock Name="LblProfile" Margin="0 0 20 0">Profile</
                  TextBlock>
            </StatusBarItem>
        </StatusBar>

        <Grid Margin="20">
            <Grid.RowDefinitions>
                <RowDefinition Height="Auto" />
                <RowDefinition Height="Auto" />
                <RowDefinition Height="10" />
                <RowDefinition Height="*" />
                <RowDefinition Height="30" />
            </Grid.RowDefinitions>
            <Grid Grid.Row="0">
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="*" />
                    <ColumnDefinition Width="Auto" />
                </Grid.ColumnDefinitions>
                <TextBox Grid.Column="0" Name="TxtSearch" Tag="pesquisa"
                  Text="{Binding Path=SearchQuery, Mode=TwoWay,
                  Delay=500,UpdateSourceTrigger=PropertyChanged}"  />
                <StackPanel Grid.Column="1" Name="PnlButtons"
                  Orientation="Horizontal" HorizontalAlignment="Right">
```

```
52                      <Button Name="BtnAdd" Content="Adicionar Documento"
                        Click="BtnAdd_Click" Margin="5" />
53                      <Button Name="BtnEdit" Content="Editar Doc"
                        Click="BtnEdit_Click" Margin="5" />
54                      <Button Name="BtnDel" Content="Eliminar Doc"
                        Click="BtnDel_Click" Margin="5" />
55                      <Button Name="BtnDownload" Content="Download"
                        Click="BtnDownload_Click" Margin="5" />
56                      <Menu VerticalAlignment="Center" DockPanel.Dock="Top"
                        Margin="5" >
57                          <MenuItem Name="BtnMenu" Header="Editar ...">
58                              <MenuItem Header="Autores" Click="MenuAut_Click" >
59                                  <MenuItem.Icon>
60                                      <Image Source="/images/autores.png"  />
61                                  </MenuItem.Icon>
62                              </MenuItem>
63                              <MenuItem Header="Editoras"
                        Click="MenuEdit_Click" >
64                                  <MenuItem.Icon>
65                                      <Image Source="/images/editoras.png"  />
66                                  </MenuItem.Icon>
67                              </MenuItem>
68                              <MenuItem Header="Tipos de Documentos"
                        Click="MenuType_Click" >
69                                  <MenuItem.Icon>
70                                      <Image Source="/images/tipo.png"  />
71                                  </MenuItem.Icon>
72                              </MenuItem>
73                          </MenuItem>
74                      </Menu>
75
76                  </StackPanel>
77
78              </Grid>
79              <Grid Grid.Row="3">
80                  <ScrollViewer Grid.Row="3">
81                      <DataGrid Name="gridDocs" AutoGenerateColumns="False"
                        Height="Auto"
82                    ItemsSource="{Binding}"
83                    CanUserAddRows="False"
84                    CanUserDeleteRows="False"
85                    IsReadOnly="True"
86                    SelectedCellsChanged="gridDocs_SelectedCellsChanged"
87                    MouseDoubleClick="gridDocs_MouseDoubleClick"
88                    VerticalScrollBarVisibility="Auto">
89                          <DataGrid.Columns>
90                              <DataGridTextColumn Header="Autor"
                          Binding="{ Binding Author.Name }" />
91                              <DataGridTextColumn Header="Título"
                          Binding="{ Binding Title }" />
92                              <DataGridTextColumn Header="Editora"
                          Binding="{ Binding Publisher.Name }"  />
93                              <DataGridTextColumn Header="Tipo de Documento"
                          Binding="{ Binding DocType.Description }"  />
94                              <DataGridTextColumn Header="Resumo" Width="100"
                          Binding="{ Binding Abstract }" >
```

```
95                                  <DataGridTextColumn.ElementStyle>
96                                      <Style>
97                                          <Setter
                    Property="TextBlock.TextWrapping" Value="Wrap" />
98                                      </Style>
99                                  </DataGridTextColumn.ElementStyle>
100                             </DataGridTextColumn>
101                                 <DataGridTemplateColumn Width="100*"
                    Header="Capa">
102                                 <DataGridTemplateColumn.CellTemplate>
103                                     <DataTemplate>
104                                         <Image Width="100" Source="{Binding
                    CoverBytes, Converter={StaticResource imgConverter}}" />
105                                     </DataTemplate>
106                                 </DataGridTemplateColumn.CellTemplate>
107                             </DataGridTemplateColumn>
108                         </DataGrid.Columns>
109                     </DataGrid>
110                 </ScrollViewer>
111
112             </Grid>
113
114             <Button x:Name="BtnCancel" Grid.Row="4"
                  HorizontalAlignment="Right" Content="Close" Height="20"
                  Width="75" Margin="5" Click="BtnClose_Click"/>
115
116         </Grid>
117     </DockPanel>
118
119 </Window>
120
```

```
1  using GeekLibrary.Models;
2  using GeekLibrary.Services;
3  using GeekLibrary.Utilities;
4  using Microsoft.Win32;
5  using System;
6  using System.Collections.Generic;
7  using System.ComponentModel;
8  using System.IO;
9  using System.Linq;
10 using System.Text;
11 using System.Threading.Tasks;
12 using System.Windows;
13 using System.Windows.Controls;
14 using System.Windows.Data;
15 using System.Windows.Documents;
16 using System.Windows.Input;
17 using System.Windows.Media;
18 using System.Windows.Media.Imaging;
19 using System.Windows.Shapes;
20
21 namespace GeekLibrary
22 {
23     /// <summary>
24     /// Interaction logic for PublisherWindow.xaml
25     /// </summary>
26     public partial class PublisherWindow : Window, INotifyPropertyChanged
27     {
28         #region INotify
29         public event PropertyChangedEventHandler PropertyChanged;
30
31         protected void NotifyPropertyChanged(string propertyName)
32         {
33             if (PropertyChanged != null)
34                 PropertyChanged(this, new PropertyChangedEventArgs
                      (propertyName));
35         }
36         #endregion
37
38
39         private string searchQuery;
40         public string SearchQuery
41         {
42             get
43             {
44                 return searchQuery;
45             }
46
47             set
48             {
49                 searchQuery = value;
50                 NotifyPropertyChanged("SearchQuery");
51                 Refresh();
52             }
53         }
54
55         public List<Publisher> Publishers { get; set; }
```

```
56
57          public PublisherWindow()
58          {
59              InitializeComponent();
60              Refresh();
61              this.DataContext = this;
62          }
63
64          private void Refresh()
65          {
66              Task.Run(() =>
67              {
68                  if (string.IsNullOrEmpty(SearchQuery))
69                  {
70                      this.Publishers = DbService.Instance.GetPublishers();
71                  }
72                  else
73                  {
74                      this.Publishers = DbService.Instance.SearchPublishers
                        (SearchQuery);
75                  }
76                  UpdateUIAsync();
77              });
78          }
79          private void RefresBtnsState()
80          {
81              BtnEdit.IsEnabled = gridPublishers.SelectedIndex > -1;
82              BtnDel.IsEnabled = gridPublishers.SelectedIndex > -1;
83          }
84          private void UpdateUIAsync()
85          {
86              Dispatcher.Invoke(new Action(() =>
87              {
88                  this.gridPublishers.ItemsSource = Publishers;
89              }));
90          }
91          private void gridPublishers_MouseDoubleClick(object sender,
              MouseButtonEventArgs e)
92          {
93              Publisher Pub = (Publisher)gridPublishers.SelectedItem;
94              EditPublisherWindow editpublish = new EditPublisherWindow
                  (EditionMode.Edit, Pub);
95              bool? result = editpublish.ShowDialog();
96              if (result.HasValue && result.Value)
97                  Refresh();
98          }
99
100         private void BtnEdit_Click(object sender, RoutedEventArgs e)
101         {
102             try
103             {
104                 Publisher Pub = (Publisher)gridPublishers.SelectedItem;
105                 EditPublisherWindow editpublish = new EditPublisherWindow
                      (EditionMode.Edit, Pub);
106                 bool? result = editpublish.ShowDialog();
107                 if (result.HasValue && result.Value)
```

```
108                     Refresh();
109             }
110             catch (Exception ex)
111             {
112                 MessageBox.Show(ex.Message, "Erro", MessageBoxButton.OK,
                      MessageBoxImage.Error);
113             }
114         }
115
116         private void BtnAdd_Click(object sender, RoutedEventArgs e)
117         {
118             try
119             {
120                 Publisher Pub = (Publisher)gridPublishers.SelectedItem;
121                 EditPublisherWindow editpublish = new EditPublisherWindow
                      (EditionMode.Add, Pub);
122                 bool? result = editpublish.ShowDialog();
123                 if (result.HasValue && result.Value)
124                     Refresh();
125             }
126             catch (Exception ex)
127             {
128                 MessageBox.Show(ex.Message, "Erro", MessageBoxButton.OK,
                      MessageBoxImage.Error);
129             }
130         }
131
132         private void BtnDel_Click(object sender, RoutedEventArgs e)
133         {
134             try
135             {
136                 Publisher Pub = (Publisher)gridPublishers.SelectedItem;
137                 if (MessageBox.Show("Tem a certeza que pretende eliminar o
                      item selecionado?", "Confirmação", MessageBoxButton.YesNo,
                      MessageBoxImage.Question) == MessageBoxResult.Yes)
138                 {
139                     DbService.Instance.DeletePub(Pub.Id);
140                     Refresh();
141                 }
142             }
143             catch (Exception ex)
144             {
145                 MessageBox.Show(ex.Message, "Erro", MessageBoxButton.OK,
                      MessageBoxImage.Error);
146             }
147         }
148
149         private void BtnClose_Click(object sender, RoutedEventArgs e)
150         {
151             this.DialogResult = false;
152         }
153     }
154 }
155
```

```
<Window x:Class="GeekLibrary.PublisherWindow"
        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:GeekLibrary"
        xmlns:converters="clr-namespace:GeekLibrary.Converters"
        xmlns:validators="clr-namespace:GeekLibrary.Validators"
        mc:Ignorable="d"
        Title="Editoras" Icon="images/editoras.png" Height="400" Width="600"
          WindowStartupLocation="CenterScreen">
    <Window.Resources>
        <converters:BinaryImageConverter x:Key="imgConverter" />
    </Window.Resources>
    <DockPanel>
        <StatusBar DockPanel.Dock="Bottom">
            <StatusBar.ItemsPanel>
                <ItemsPanelTemplate>
                    <Grid>
                        <Grid.RowDefinitions>
                            <RowDefinition Height="*"/>
                        </Grid.RowDefinitions>
                        <Grid.ColumnDefinitions>
                            <ColumnDefinition Width="*"/>
                            <ColumnDefinition Width="*"/>
                        </Grid.ColumnDefinitions>
                    </Grid>
                </ItemsPanelTemplate>
            </StatusBar.ItemsPanel>
            <StatusBarItem>
                <TextBlock Name="LblUser" Margin="20 0 0 0">User</TextBlock>
            </StatusBarItem>
            <StatusBarItem Grid.Column="1" HorizontalContentAlignment="Right">
                <TextBlock Name="LblProfile" Margin="0 0 20 0">Profile</
                  TextBlock>
            </StatusBarItem>
        </StatusBar>

        <Grid Margin="20">
            <Grid.RowDefinitions>
                <RowDefinition Height="Auto" />
                <RowDefinition Height="Auto" />
                <RowDefinition Height="10" />
                <RowDefinition Height="*" />
                <RowDefinition Height="30" />
            </Grid.RowDefinitions>
            <Grid Grid.Row="0">
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="*" />
                    <ColumnDefinition Width="Auto" />
                </Grid.ColumnDefinitions>
                <TextBox Grid.Column="0" Name="TxtSearch" Text="{Binding
                  Path=SearchQuery, Mode=TwoWay, Delay=500,
                  UpdateSourceTrigger=PropertyChanged}" Tag="Insira Nome,
                  Endereço, Email ou Contacto" />
                <StackPanel Grid.Column="1" Name="PnlButtons"
```

```
                    Orientation="Horizontal" HorizontalAlignment="Right">
52                    <Button Name="BtnEdit" Content="Editar"  Margin="5"
                      Click="BtnEdit_Click" />
53                    <Button Name="BtnAdd" Content="Adicionar"  Margin="5"
                      Click="BtnAdd_Click" />
54                    <Button Name="BtnDel" Content="Eliminar"  Margin="5"
                      Click="BtnDel_Click" />
55                  </StackPanel>
56              </Grid>
57              <ScrollViewer Grid.Row="3">
58                  <DataGrid Name="gridPublishers" AutoGenerateColumns="False"
59                    ItemsSource="{Binding}"
60                    CanUserAddRows="False"
61                    CanUserDeleteRows="False"
62                    IsReadOnly="True"
63                    MouseDoubleClick="gridPublishers_MouseDoubleClick">
64                      <DataGrid.Columns>
65                          <DataGridTextColumn Header="Nome" Binding="{ Binding
                        Name }" />
66                          <DataGridTextColumn Header="Endereço"
                        Binding="{ Binding Address }" />
67                          <DataGridTextColumn Header="Email" Binding="{ Binding
                        Email }"  />
68                          <DataGridTextColumn Header="Contacto"
                        Binding="{ Binding PhoneNumber }" Width="100*"  />
69                      </DataGrid.Columns>
70                  </DataGrid>
71              </ScrollViewer>
72              <Button Name="BtnClose" Content="Close" Grid.Row="6" Width="40"
                  Height="20" HorizontalAlignment="Right"
                  VerticalAlignment="Bottom" Click="BtnClose_Click"></Button>
73          </Grid>
74
75
76      </DockPanel>
77  </Window>
78
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Globalization;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7  using System.Windows.Controls;
8
9  namespace GeekLibrary.Validators
10 {
11     public class RequiredField : ValidationRule
12     {
13         private String _errorMessage = String.Empty;
14         public string ErrorMessage
15         {
16             get { return _errorMessage; }
17             set { _errorMessage = value; }
18         }
19
20         public override ValidationResult Validate(object value, CultureInfo
             cultureInfo)
21         {
22             var str = value as string;
23
24             if (String.IsNullOrEmpty(str))
25             {
26                 return new ValidationResult(false, this.ErrorMessage);
27             }
28
29             return new ValidationResult(true, null);
30         }
31     }
32 }
33
```

```
1  using GeekLibrary.Models;
2  using GeekLibrary.Utilities;
3  using System;
4  using System.Collections.Generic;
5  using System.DirectoryServices.AccountManagement;
6  using System.Linq;
7  using System.Text;
8
9  namespace GeekLibrary.Services
10 {
11     public class SecurityService : GenericSingleton<SecurityService>
12     {
13         public AppUser GetUserInContext()
14         {
15             AppUser result = new AppUser() { Username = Environment.UserName };
16             System.Threading.Thread.Sleep(3000);
17
18             PrincipalContext ctx = null;
19             try
20             {
21                 ctx = new PrincipalContext(ContextType.Domain);
22             }
23             catch (Exception) { }
24             if (ctx == null)
25                 ctx = new PrincipalContext(ContextType.Machine);
26             using (ctx)
27             {
28                 result.DomainName = ctx.ConnectedServer;
29                 // find a user
30                 UserPrincipal user = UserPrincipal.FindByIdentity(ctx,
                     result.Username);
31                 // find the group in question
32                 GroupPrincipal adminGroup = GroupPrincipal.FindByIdentity(ctx,
                     ConfigHelper.Instance.GroupNameAdmin);
33                 if (adminGroup == null)
34                     throw new Exception("Não foi encontrado o grupo de acesso "
                         + ConfigHelper.Instance.GroupNameAdmin);
35                 GroupPrincipal normalGroup = GroupPrincipal.FindByIdentity(ctx,
                     ConfigHelper.Instance.GroupNameNormal);
36                 if (normalGroup == null)
37                     throw new Exception("Não foi encontrado o grupo de acesso "
                         + ConfigHelper.Instance.GroupNameNormal);
38                 if (user != null)
39                 {
40                     // check if user is member of that group
41                     if (user.IsMemberOf(adminGroup))
42                         result.CurrentAccessMode = AccessMode.Admin;
43                     else if (user.IsMemberOf(normalGroup))
44                         result.CurrentAccessMode = AccessMode.Normal;
45                 }
46             }
47             return result;
48         }
49     }
50 }
51
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Globalization;
4  using System.Linq;
5  using System.Text;
6  using System.Text.RegularExpressions;
7  using System.Threading.Tasks;
8  using System.Windows.Controls;
9
10 namespace GeekLibrary.Validators
11 {
12
13         public class ValidateEmail : ValidationRule
14         {
15             private String _errorMessage = String.Empty;
16             public string ErrorMessage
17             {
18                 get { return _errorMessage; }
19                 set { _errorMessage = value; }
20             }
21
22             public override ValidationResult Validate(object value, CultureInfo
                   cultureInfo)
23             {
24                 var strmail = value as string;
25
26                 if (!String.IsNullOrEmpty(strmail))
27                 {
28                     strmail = strmail.ToLower(); //Passa todas as letras do
                       email para minusculas
29                     //Link para a Regex: https://msdn.microsoft.com/en-us/
                       library/01escwtf(v=vs.110).aspx
30                     Regex rg = new Regex(@"^(?("")(""".+?(?<!\\)""@)|(([0-9a-z]
                       ((\.(?!\.))|[-!#\$%&'\*\+/=\?\^`\{\}\|~\w])*)(?<=[0-9a-z])
                       @))" +
31                         @"(?(\[)(\[(\d{1,3}\.){3}\d{1,3}\])|(([0-9a-z][-\w]*
                       [0-9a-z]*\.)+[a-z0-9][\-a-z0-9]{0,22}[a-z0-9]))$");
32
33                     if (!rg.IsMatch(strmail))
34                     {
35                         return new ValidationResult(false, this.ErrorMessage);
36                     }
37
38                 }
39                 return new ValidationResult(true, null);
40             }
41         }
42 }
43
44
```

```
1  <Window x:Class="GeekLibrary.InitWindow"
2          xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3          xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4          xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
5          xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
6          xmlns:local="clr-namespace:GeekLibrary"
7          xmlns:gif="http://wpfanimatedgif.codeplex.com"
8          mc:Ignorable="d"
9          Title="InitWindow" Height="150" Width="300" WindowStyle="None"
10         ResizeMode="NoResize" WindowStartupLocation="CenterScreen"
11         BorderThickness="1" Loaded="Window_Loaded">
12     <Grid>
13          <Image gif:ImageBehavior.AnimatedSource="images/load.gif"
             Stretch="Fill"></Image>
14         <Grid x:Name="LayoutRoot">
15
16             <Grid.RowDefinitions>
17                 <RowDefinition Height="20" />
18                 <RowDefinition Height="Auto" />
19                 <RowDefinition Height="10" />
20                 <RowDefinition Height="*" />
21                 <RowDefinition Height="Auto" />
22                 <RowDefinition Height="20" />
23             </Grid.RowDefinitions>
24             <StackPanel Grid.Row="5">
25                 <TextBlock Name="TxtStatus" HorizontalAlignment="Center"
                     TextBlock.Foreground="White"/>
26             </StackPanel>
27         </Grid>
28     </Grid>
29
30 </Window>
31
```

```
1  using GeekLibrary.Models;
2  using GeekLibrary.Services;
3  using System;
4  using System.Collections.Generic;
5  using System.Data;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows;
10 using System.Windows.Controls;
11 using System.Windows.Data;
12 using System.Windows.Documents;
13 using System.Windows.Input;
14 using System.Windows.Media;
15 using System.Windows.Media.Imaging;
16 using System.Windows.Navigation;
17 using System.Windows.Shapes;
18
19
20 namespace GeekLibrary
21 {
22     /// <summary>
23     /// Interaction logic for InitWindow.xaml
24     /// </summary>
25     public partial class InitWindow : Window
26     {
27         public AppUser User { get; set; }
28         public List<Document> Docs { get; set; }
29
30         public bool ErrorOccured { get; set; }
31
32         public InitWindow()
33         {
34             InitializeComponent();
35         }
36
37         private void Window_Loaded(object sender, RoutedEventArgs e)
38         {
39
40         }
41
42         protected override void OnContentRendered(EventArgs e)
43         {
44             base.OnContentRendered(e);
45             Task.Run(() =>
46             {
47                 TxtStatus.Dispatcher.Invoke(new Action(() => { TxtStatus.Text =
                       "A validar o acesso de " + Environment.UserName + "... Por
                     favor aguarde."; }));
48                 User = SecurityService.Instance.GetUserInContext();
49                 TxtStatus.Dispatcher.Invoke(new Action(() => { TxtStatus.Text =
                       "A carregar dados... Por favor aguarde."; }));
50                 DbService.Instance.InitializeReferenceData();
51                 Docs = DbService.Instance.GetAllDocs();
52
53             }).ContinueWith((t) =>
```

```
54              {
55
56                  if (t.IsFaulted)
57                  {
58                      ErrorOccured = true;
59                      Exception ex = t.Exception;
60                      while (ex is AggregateException && ex.InnerException !=
                        null)
61                          ex = ex.InnerException;
62                      MessageBox.Show(ex.Message, "Erro", MessageBoxButton.OK,
                        MessageBoxImage.Error);
63                      this.Close();
64                  }
65                  else
66                  {
67                      if (User.CurrentAccessMode == AccessMode.Unknown)
68                      {
69                          MessageBox.Show("O seu utilizador não pertence a nenhum
                           grupo com acesso à aplicação.", "Acesso negado",
                          MessageBoxButton.OK, MessageBoxImage.Error);
70                          this.Close();
71                      }
72
73                      MainWindow main = new MainWindow(User, Docs);
74                      main.Show();
75                      this.Close();
76
77                  }
78
79
80
81              }, TaskScheduler.FromCurrentSynchronizationContext());
82
83          }
84
85
86      }
87  }
88
```